88MW300/302

WLAN Microcontroller

IEEE 802.11n/g/b

Datasheet

Doc. No. MV-S109936-01, Rev. C
May 12, 2016, 2.00

Document Conventions

	**Note:** Provides related information or information of special importance.
	**Caution:** Indicates potential damage to hardware or software, or loss of data.
	**Warning:** Indicates a risk of personal injury.

Document Status

Doc Status: 2.00	Technical Publication: 0.xx

For more information, visit our website at: http://www.marvell.com

PRODUCT OVERVIEW

The Marvell® 88MW300/302 is a highly integrated, low-power WLAN Microcontroller System-on-Chip (SoC) solution designed for a broad array of smart devices for Internet of Things (IoT), wearables, accessories, Machine-to-Machine (M2M), home automation, and Smart Energy applications.

A high degree of integration enables very low system costs requiring only a single 3.3V power input, a 38.4 MHz crystal, and SPI Flash. The RF path needs only a lowpass filter for antenna connection.

The SoC includes a full-featured W LAN subsystem powered by proven and mature IEEE 802.11n/g/b Marvell technology. The WLAN subsystem integrates a WLAN MAC, baseband, and direct-conversion RF radio with integrated PA, LNA, and transmit/receive switch. It also integrates a CPU subsystem with integrated memory to run Marvell WLAN firmware to handle real time WLAN protocol processing to off-load many WLAN functions from the main application CPU.

The 88MW300/302 application subsystem is powered by an ARM Cortex-M4F CPU that operates up to 200 MHz. The device supports an integrated 512 KB SRAM, 128 KB mask ROM, and a QSPI interface to external Flash. An integrated Flash Controller with a 32 KB SRAM cache enables eXecute In Place (XIP) support for firmware from Flash.

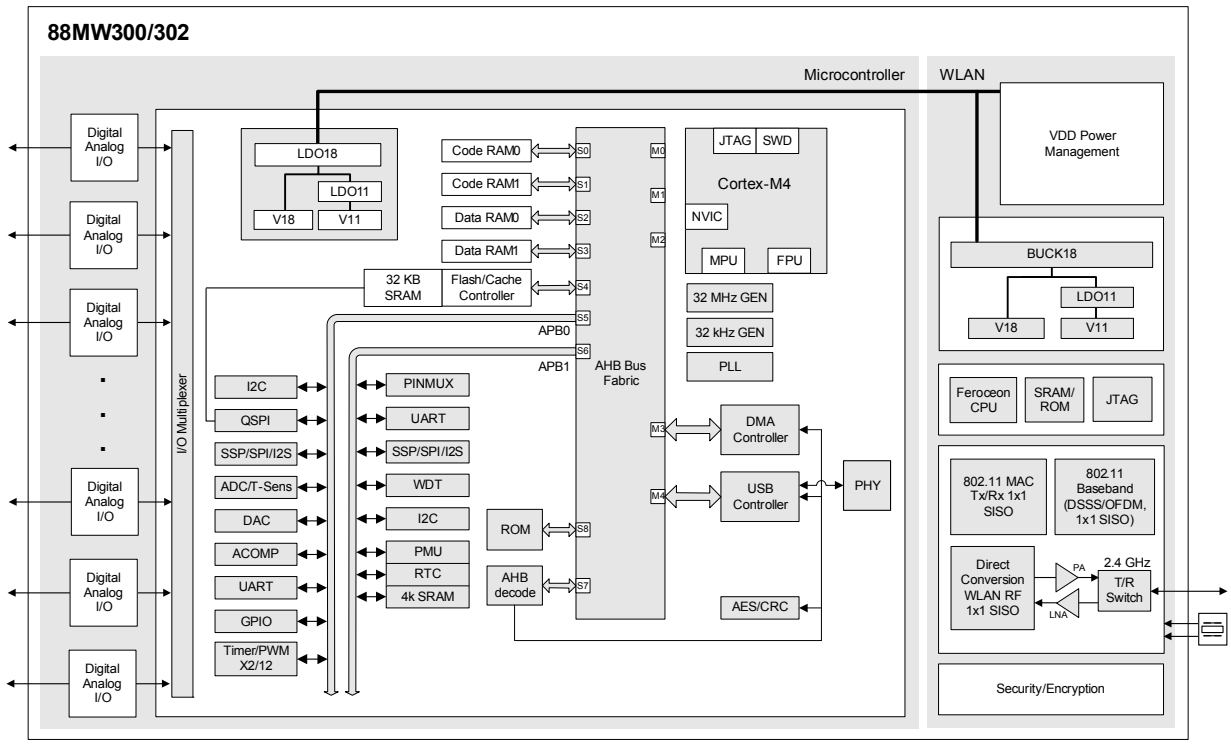
The SoC is designed for low-power operation and includes several low-power states and fast wake-up times. Multiple power domains and clocks can be individually shut down to save power. The SoC also has a high-efficiency internal PA that can be operated in low-power mode to save power. The microcontroller and WLAN subsystems can be placed into low-power states, independently, supporting a variety of application use cases. An internal DC-DC regulator provides the 1.8V rail for the WLAN subsystem.

The SoC provides a full array of peripheral interfaces including SSP/SPI/I^2S (3x), UART (3x), I^2C (2x), General Purpose Timers and PWM, ADC, DAC, Analog Comparator, and GPIOs. It also includes a hardware cryptographic engine, RTC, and Watchdog Timer. The 88MW302 includes a high speed USB On-The-Go (OTG) interface to enable USB audio, video, and other applications.

A complete set of digital and analog interfaces enable direct interfacing for I/O avoiding the need for external chips. The application CPU can be used to support custom application development avoiding the need for another microcontroller or processor.

Figure 1 shows an overall block diagram of the device.

Figure 1: Block Diagram



Applications

- White goods/appliances—refrigerator, washer, dryer, oven range, microwave, dishwasher, water heater, air conditioner
- Consumer devices and accessories—toys, speakers, headset, alarm clock, gaming accessory, remote control
- Home automation—smart outlet, light switch, security camera, thermostat, sprinkler controller, sensor, door lock, door bell, garage door, security system
- Personal health devices—weighing scale, glucometer, blood pressure monitor, fitness equipment
- IoT/wearables—coffee pot, rice cooker, vacuum cleaner, air purifier, smart watch, fitness bracelet, pet monitor
- Commercial/industrial—lighting, building automation, asset management, Point of Sale (POS) sales
- Gateways—Connecting IR, sub-Gig or Legacy RF, Bluetooth Smart, ZigBee, ZWave and other radios to Wi-Fi/IP network

Key Features

- Highly integrated SoC requiring very few external components for a full system operation
- Multiple low-power modes and fast wake-up times
- Full-featured, single stream 802.11n/g/b WLAN
- High-efficiency PA with a low-power (10 dB) mode
- Cortex-M4F application CPU for applications with integrated 512 KB SRAM and 128 KB mask ROM
- Flash Controller with embedded 32 KB SRAM cache to support XIP from external SPI Flash
- Secure boot
- Full set of digital and analog I/O interfaces

Power Management

- Power modes—active, idle, standby, sleep, shutoff, power-down
- Integrated high efficiency buck DC-DC converter
- Independent power domains
- Brown-out detection
- Integrated POR
- Wake-up through dedicated GPIO, IRQ, and RTC

Chip Package

- 88MW300—68-pin QFN, 8x8 mm
  - USB OTG not supported
  - 35 GPIOs
  - 2 GPTs
- 88MW302—88-pin QFN, 10x10 mm
  - USB OTG supported
  - 50 GPIOs
  - 4 GPTs

Table 1: Package Feature Differences[1]

Feature	68-Pin	88-Pin
GPIO	35 total GPIO_0 to GPIO_10 GPIO_16 GPIO_22 to GPIO_33 GPIO_39 to GPIO_49	50 total GPIO_0 to GPIO_49
USB 2.0 OTG	--	1
GPT	2	4

1. All I/O features are muxed on GPIOs, except WLAN RF TX/RX, USB, reference clock, and reset functionality.

Temperature

- Extended: -30 to 85°C
- Industrial: -40 to 85°C
- Storage: -55 to 125°C

Wireless

- IEEE 802.11n/g/b, 1x1 SISO 2.4 GHz and HT20
- Integrated CPU, memory, MAC, DSSS/OFDM baseband, direct conversion RF radio, encryption
- Antenna diversity
- CMOS and low-swing sine wave input clock
- Low-power with deep sleep and standby modes
- Pre-regulated supplies
- Integrated T/R switch, PA, and LNA
- Optional 802.11n features
- One Time Programmable (OTP) memory to eliminate need for external EEPROM

WLAN Rx Path

- Direct conversion architecture eliminates need for external SAW filter
- On-chip gain selectable LNA with optimized noise figure and power consumption
- High dynamic range AGC function in receive mode

WLAN Tx Path

- Integrated PA with power control
- Optimized Tx gain distribution for linearity and noise performance

WLAN Local Oscillator

- Fractional-N for multiple reference clock support
- Fine channel step

WLAN Encryption

- WEP 64- and 128-bit encryption with hardware TKIP processing (WPA)
- AES-CCMP hardware implementation as part of 802.11i security standard (WPA2)
- Enhanced AES engine performance
- AES-Cipher-Based Message Authentication Code (CMAC) as part of the 802.11w security standard
- WLAN Authentication and Privacy Infrastructure (WAPI)

IEEE 802.11 Standards

- 802.11 data rates of 1 and 2 Mbps
- 802.11b data rates of 5.5 and 11 Mbps
- 802.11g data rates 6, 9, 12, 18, 24, 36, 48, and 54 Mbps for multimedia content transmission
- 802.11g/b performance enhancements
- 802.11n compliant with maximum data rates up to 72.2 Mbps (20 MHz channel)
- 802.11d international roaming
- 802.11e quality of service
- 802.11h transmit power control
- 802.11i enhanced security
- 802.11k radio resource measurement
- 802.11n block acknowledgement extension
- 802.11r fast hand-off for AP roaming
- 802.11w protected management frames
- Fully supports clients (stations) implementing IEEE Power Save mode
- Wi-Fi direct connectivity

Microprocessor

Processor

- ARM Cortex-M4F, 32-bit
- 200 MHz main bus clock

Memory

- 128 KB ROM
- 512 KB RAM

Flash Controller

- Supports QSPI Flash devices
- Memory-mapped access to QSPI Flash devices
- 32 KB SRAM cache

Digital Interfaces

- 3x I^2S stereo
- 3x SPI master/slave
- 2x I^2C master/slave
- 3x UART
- 1x USB OTG 2.0, high-speed
- 1x QSPI
- Up to 50 GPIOs
- 2x wake-up pins

Analog

- 2-step ADC with integrated PGA and configurable resolution/speed
  - 12-bit/2 MHz sample(s) for fast conversion
  - 16-bit/16 kHz sample/s with voice quality
  - 8 single channels or 4 differential channels
- 2-Channel or 1 differential channel DAC, 10-bit/500 ksps
- 2 Analog Comparators with programmable speed/current
- On-die/off-chip temperature sensing and battery monitor

Counters/Timers/PWM

- General Purpose Timers (GPT) with LED PWM support
- Real Time Clock (RTC)
- CM4 system tick
- Watchdog Timer

Table of Contents

List of Figures

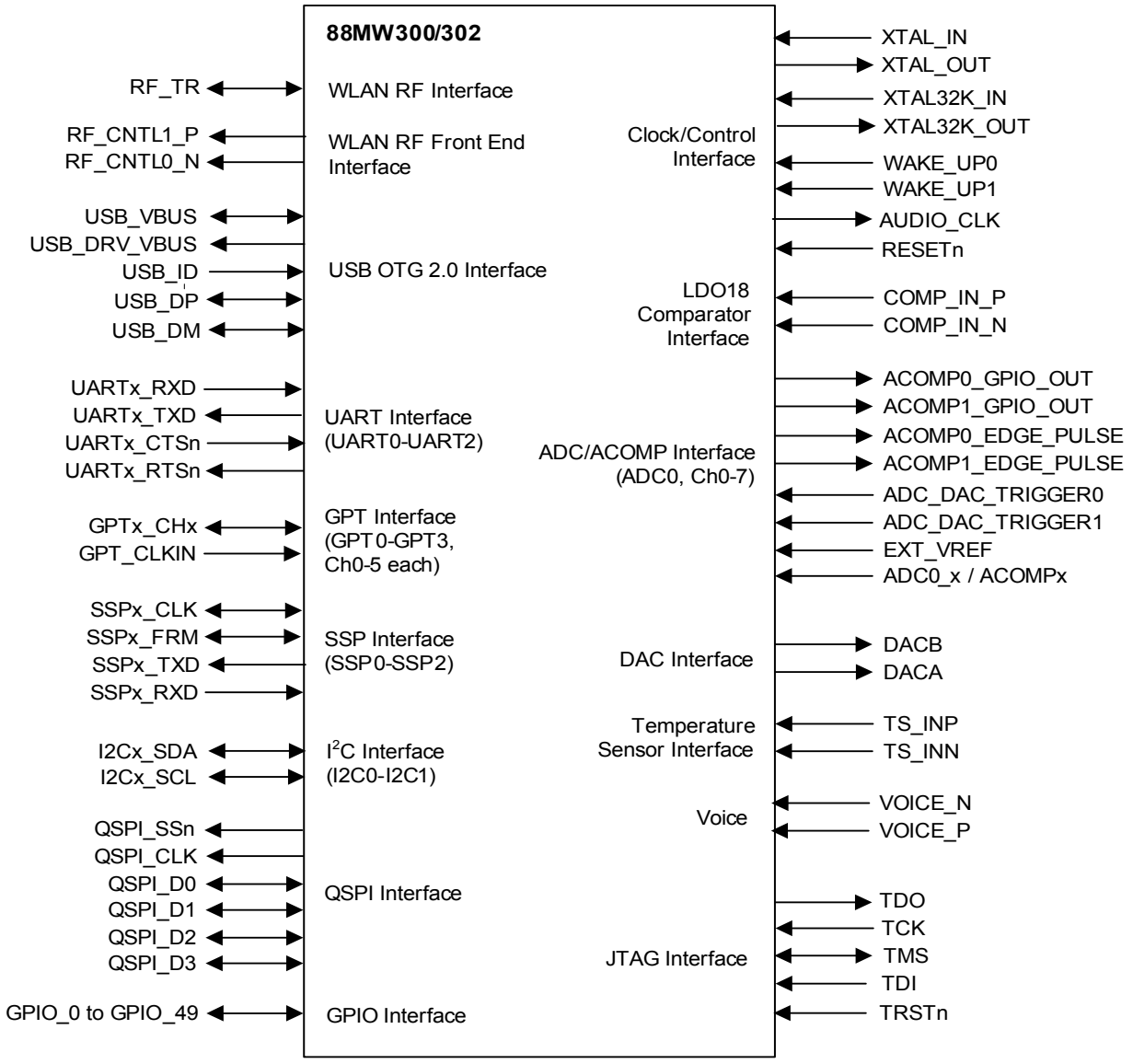
List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

1 Package

1.1 Signal Diagram

Figure 2: Signal Diagram[1 2 3 4]

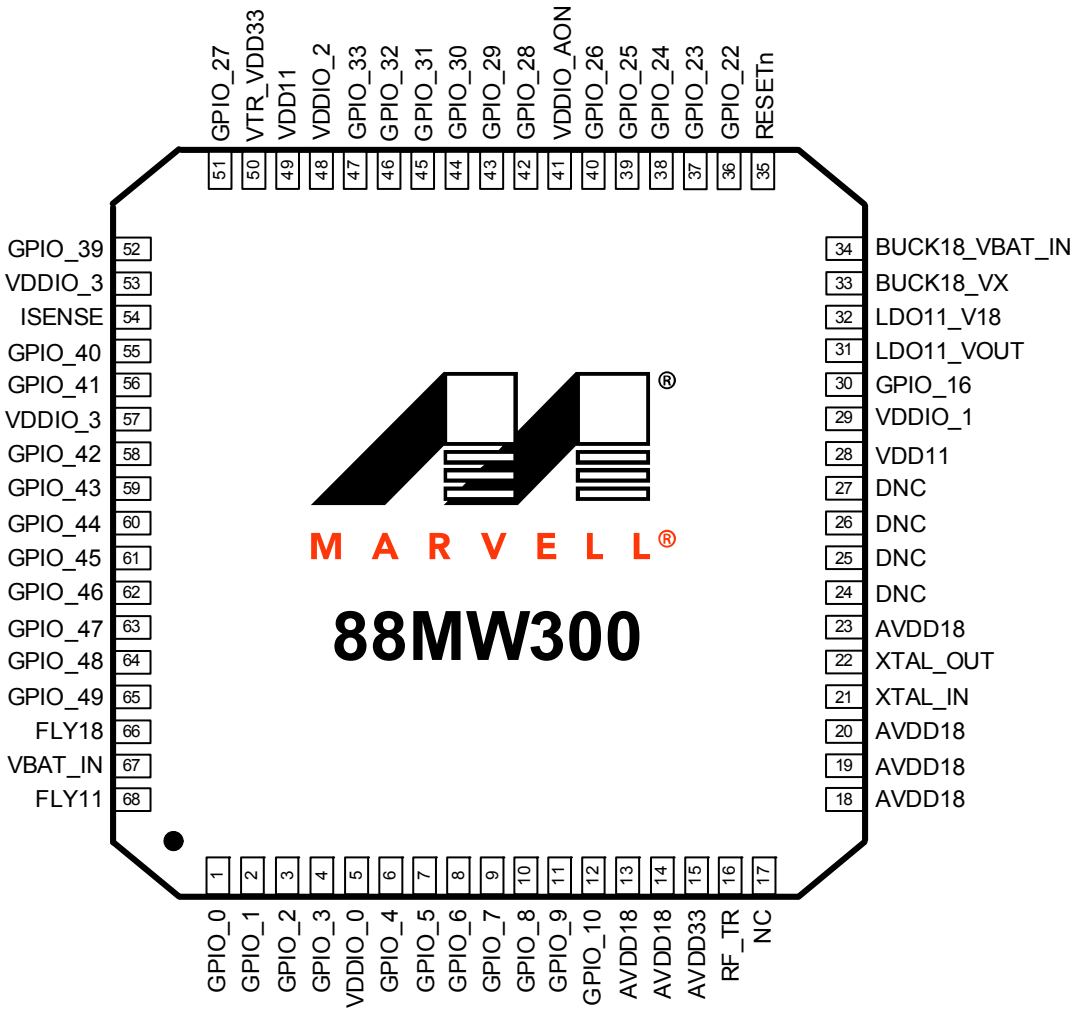


1. Signals are muxed on dedicated pins. See Section 1.4, Pin Description, on page 44 for dedicated pin / muxed signal descriptions.
2. Some pins/signals are available on the 88-pin QFN only. See Section 1.4, Pin Description, on page 44.
3. RF_TR, USB OTG, XTAL_IN/OUT, and RESETn pins are dedicated. Others are muxed on GPIOs.
4. See Table 16, Power and Ground, on page 64 for power signals.

1.2 Pinout

1.2.1 Pinout—68-Pin QFN

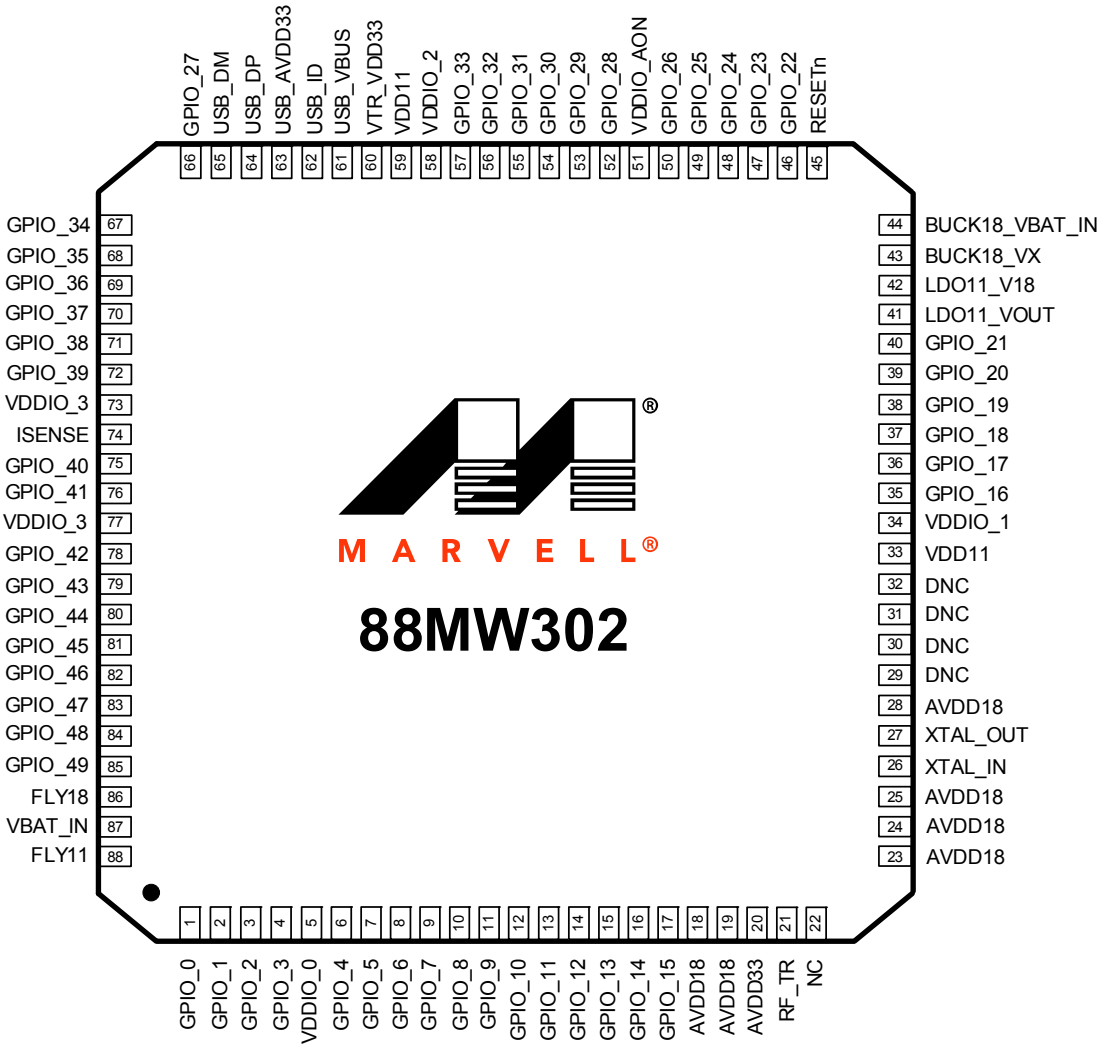
Figure 3: Pinout—68-Pin QFN[1]



1. Connect pin 17 to ground.

1.2.2 Pinout—88-Pin QFN

Figure 4: Pinout—88-Pin QFN[1]

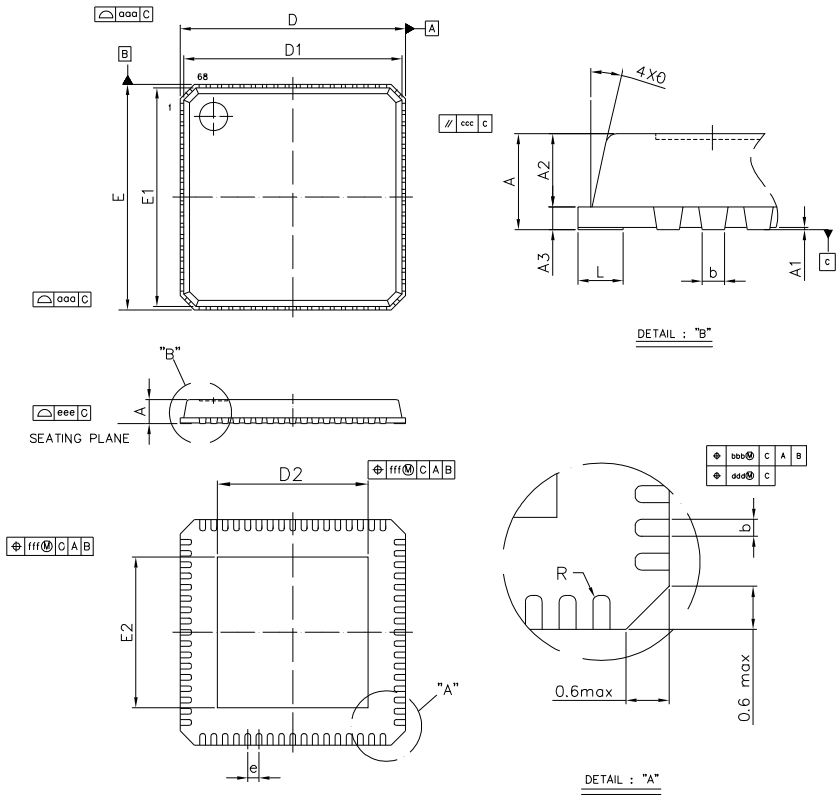


1. Connect pin 22 to ground.

1.3 Mechanical Drawing

1.3.1 Mechanical Drawing—68-Pin QFN

Figure 5: Mechanical Drawing—68-Pin QFN



Symbol	Dimension in mm			Dimension in inch		
	MIN	NOM	MAX	MIN	NOM	MAX
A	0.80	0.85	1.00	0.031	0.033	0.039
A1	0.00	0.02	0.05	0.000	0.001	0.002
A2	0.60	0.65	0.80	0.024	0.026	0.031
A3	0.20 REF			0.008 REF		
b	0.15	0.20	0.25	0.006	0.008	0.010
D/E	8.00 BSC			0.315 BSC		
D1/E1	7.75 BSC			0.305 BSC		
e	0.40 BSC			0.016 BSC		
L	0.30	0.40	0.50	0.012	0.016	0.020
θ	0°	---	14°	0°	---	14°
R	0.075	---	---	0.003	---	---
aaa	---	---	0.15	---	---	0.006
bbb	---	---	0.10	---	---	0.004
ccc	---	---	0.10	---	---	0.004
ddd	---	---	0.05	---	---	0.002
eee	---	---	0.08	---	---	0.003
fff	---	---	0.10	---	---	0.004

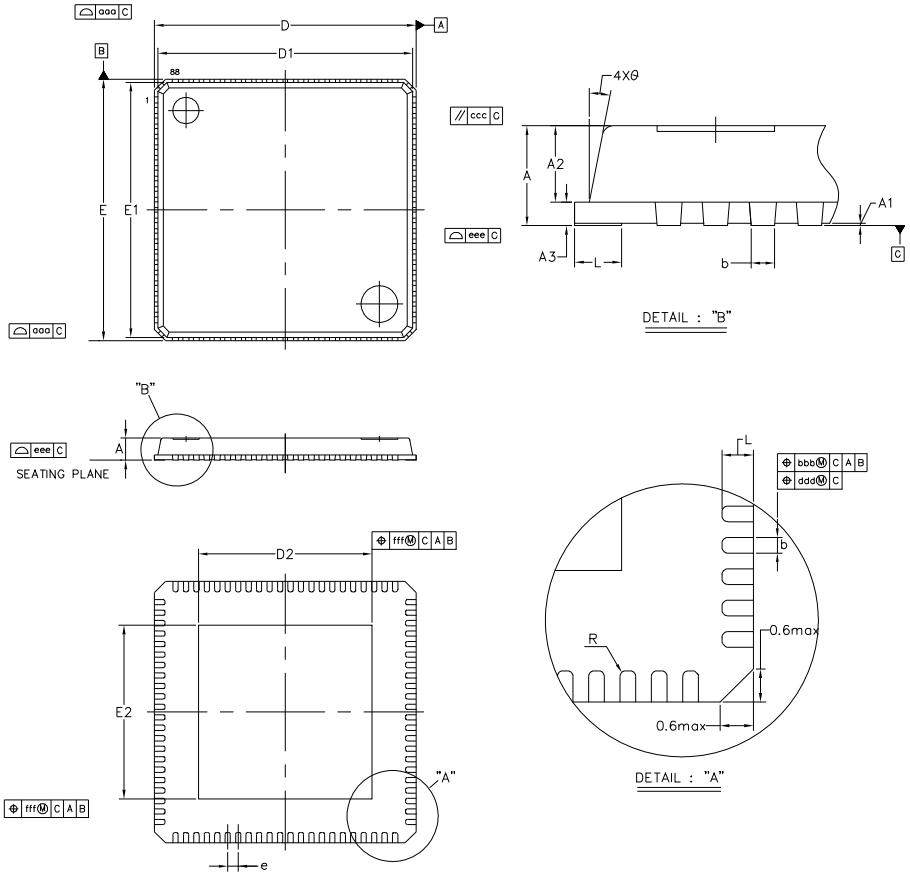
EPad Size Options			
Option	Symbol	Dimension in mm	Dimension in inch
Option #1	D_2	5.35 ± 0.15	.211 ± 0.006
	E_2	5.35 ± 0.15	.211 ± 0.006
Option #2	D_2	4.47 ± 0.15	.176 ± 0.006
	E_2	4.47 ± 0.15	.176 ± 0.006
Option #3	D_2	5.49 ± 0.15	.216 ±0.006
	E_2	5.49 ± 0.15	.216 ±0.006
Option #4	D_2	3.50 ± 0.15	.138 ±0.006
	E_2	3.50 ± 0.15	.138 ±0.006

NOTE:
1. CONTROLLING DIMENSION : MILLIMETER
2. REFERENCE DOCUMENT : JEDEC MO-220.

Note: QFN package uses Epad size Option #3 only. See Section 22.5, Package Thermal Conditions, on page 325 for electrical specifications. See Section 23.2, Package Marking, on page 348 for package marking.

1.3.2 Mechanical Drawing—88-Pin QFN

Figure 6: Mechanical Drawing—88-Pin QFN



Symbol	Dimension in mm			Dimension in inch		
	MIN	NOM	MAX	MIN	NOM	MAX
A	0.80	0.85	0.90	0.031	0.033	0.035
A1	0.00	0.02	0.05	0.000	0.001	0.002
A2	0.60	0.65	0.70	0.024	0.026	0.028
A3	0.20 REF			0.008 REF		
b	0.15	0.20	0.25	0.006	0.008	0.010
D/E	9.90	10.00	10.10	0.390	0.394	0.398
D1/E1	9.75 BSC			0.384 BSC		
e	0.40 BSC			0.016 BSC		
L	0.30	0.40	0.50	0.012	0.016	0.020
Θ	0°	---	14°	0°	---	14°
R	0.075	---	---	0.003	---	---
aaa	0.10			0.004		
bbb	0.07			0.003		
ccc	0.10			0.004		
ddd	0.05			0.002		
eee	0.08			0.003		
fff	0.10			0.004		

EPad Size Options			
Option	Symbol	Dimension in mm	Dimension in inch
Option #1	D_2	6.64 ± 0.15	.261 ± 0.006
	E_2	6.64 ± 0.15	.261 ± 0.006
Option #2	D_2	6.00 ± 0.15	.236 ± 0.006
	E_2	6.00 ± 0.15	.236 ± 0.006
Option #3	D_2	4.30 ± 0.15	.169 ± 0.006
	E_2	4.30 ± 0.15	.169 ± 0.006

NOTE:

1. CONTROLLING DIMENSION : MILLIMETER
2. REFERENCE DOCUMENT : JEDEC MO-220.

Note: QFN package uses Epad size Option #3 only. See Section 22.5, Package Thermal Conditions, on page 325 for electrical specifications. See Section 23.2, Package Marking, on page 348 for package marking.

1.4 Pin Description

Table 2: Pin Types

Pin Type	Description
I/O	Digital input/output
I	Digital input
O	Digital output
A, I	Analog input
A, O	Analog output
NC	No connect
DNC	Do not connect
PWR	Power
Ground	Ground

Table 3: WLAN RF Interface

88-Pin	68-Pin	Pin Name	Type	Supply	Description
21	16	RF_TR	A, I/O	AVDD18	WLAN RF Interface (2.4 GHz Transmit/Receive) Baseband input/output data

Table 4: WLAN RF Front End Interface

88-Pin	68-Pin	Pin Name	Type	Supply	Description
GPIO_44		RF_CNTL1_P	A, O	VDDIO_3	WLAN Radio Control 1 Power-down output high signal
GPIO_45		RF_CNTL0_N	A, O	VDDIO_3	WLAN Radio Control 0 Power-down output low signal

Table 5: USB 2.0 OTG Interface[1]

NOTE: Available on 88-pin package only (88MW302)

88-Pin	68-Pin	Pin Name	Type	Supply	Description
61	--	USB_VBUS	A, I/O	--	VBUS Selection Input in device mode; unused in host mode.
62	--	USB_ID	A, I	USB_AVDD33	USB 2.0 OTG IDPIN
63	--	USB_AVDD33	A, I	--	USB 3.3V Analog Power Supply See Table 16, Power and Ground, on page 64.
64	--	USB_DP	A, I/O	USB_AVDD33	USB 2.0 Bus Data+
65	--	USB_DM	A, I/O	USB_AVDD33	USB 2.0 Bus Data–
66	--	USB_DRV_VBUS	O	VDDIO_3	Drive 5V on VBUS 0 = do not drive VBUS 1 = drive 5V on VBUS The USB_DRV_VBUS port is connected to the SoC pad to drive an external power management chip to provide power for USB VBUS.

1. After POR, if USB is in host mode, USB_DPUSB_DM will be SE0. If USB is in device mode, USB_DP/USB_DM will be High-z.

Table 6: UART Interface[1]

88-Pin	68-Pin	Signal Name	Type	Supply	Description
GPIO_0		UART0_CTSn	I	VDDIO_0	UART 0 CTSn (active low)
GPIO_1		UART0_RTSn	O	VDDIO_0	UART 0 RTSn (active low)
GPIO_2		UART0_TXD	O	VDDIO_0	UART 0 TXD
GPIO_3		UART0_RXD	I	VDDIO_0	UART 0 RXD
GPIO_23		UART0_CTSn	I	VDDIO_AON	UART 0 CTSn (active low)
GPIO_24		UART0_RXD	I	VDDIO_AON	UART 0 RXD
GPIO_30		UART0_CTSn	I	VDDIO_2	UART 0 CTSn (active low)
GPIO_31		UART0_RTSn	O	VDDIO_2	UART 0 RTSn (active low)
GPIO_32		UART0_TXD	O	VDDIO_2	UART 0 TXD
GPIO_33		UART0_RXD	I	VDDIO_2	UART 0 RXD
GPIO_37	--	UART0_RTSn	O	VDDIO_3	UART 0 RTSn (active low)

Table 6: UART Interface[1] (Continued)

88-Pin	68-Pin	Signal Name	Type	Supply	Description
GPIO_27		UART0_TXD	O	VDDIO_3	UART 0 TXD
GPIO_11	--	UART1_CTSn	I	VDDIO_0	UART 1 CTSn (active low)
GPIO_12	--	UART1_RTSn	O	VDDIO_0	UART 1 RTSn (active low)
GPIO_13	--	UART1_TXD	O	VDDIO_0	UART 1 TXD
GPIO_14	--	UART1_RXD	I	VDDIO_0	UART 1 RXD
GPIO_35	--	UART1_CTSn	I	VDDIO_3	UART 1 CTSn (active low)
GPIO_36	--	UART1_RTSn	O	VDDIO_3	UART 1 RTSn (active low)
GPIO_38	--	UART1_TXD	O	VDDIO_3	UART 1 TXD
GPIO_39		UART1_RXD	I	VDDIO_3	UART 1 RXD
GPIO_42		UART1_CTSn	I	VDDIO_3	UART 1 CTSn (active low)
GPIO_43		UART1_RTSn	O	VDDIO_3	UART 1 RTSn (active low)
GPIO_44		UART1_TXD	O	VDDIO_3	UART 1 TXD
GPIO_45		UART1_RXD	I	VDDIO_3	UART 1 RXD
GPIO_7		UART2_CTSn	I	VDDIO_0	UART 2 CTSn (active low)
GPIO_8		UART2_RTSn	O	VDDIO_0	UART 2 RTSn (active low)
GPIO_9		UART2_TXD	O	VDDIO_0	UART 2 TXD
GPIO_10		UART2_RXD	I	VDDIO_0	UART 2 RXD
GPIO_46		UART2_CTSn	I	VDDIO_3	UART 2 CTSn (active low)
GPIO_47		UART2_RTSn	O	VDDIO_3	UART 2 RTSn (active low)
GPIO_48		UART2_TXD	O	VDDIO_3	UART 2 TXD
GPIO_49		UART2_RXD	I	VDDIO_3	UART 2 RXD

1. All UART signals are muxed on GPIO pins. See Table 11, GPIO Interface, on page 51 for GPIO muxing.

Table 7: GPT Interface[1]

88-Pin	68-Pin	Signal Name	Type	Supply	Description
GPIO_0		GPT0_CH0	I/O	VDDIO_0	General Purpose Timer 0, Channel 0
GPIO_1		GPT0_CH1	I/O	VDDIO_0	General Purpose Timer 0, Channel 1
GPIO_2		GPT0_CH2	I/O	VDDIO_0	General Purpose Timer 0, Channel 2
GPIO_3		GPT0_CH3	I/O	VDDIO_0	General Purpose Timer 0, Channel 3
GPIO_4		GPT0_CH4	I/O	VDDIO_0	General Purpose Timer 0, Channel 4
GPIO_5		GPT0_CH5	I/O	VDDIO_0	General Purpose Timer 0, Channel 5
GPIO_35	--	GPT0_CLKIN	I	VDDIO_3	General Purpose Timer 0, Clock Input
GPIO_28		GPT1_CH0	I/O	VDDIO_2	General Purpose Timer 1, Channel 0
GPIO_29		GPT1_CH1	I/O	VDDIO_2	General Purpose Timer 1, Channel 1
GPIO_30		GPT1_CH2	I/O	VDDIO_2	General Purpose Timer 1, Channel 2
GPIO_31		GPT1_CH3	I/O	VDDIO_2	General Purpose Timer 1, Channel 3
GPIO_32		GPT1_CH4	I/O	VDDIO_2	General Purpose Timer 1, Channel 4
GPIO_33		GPT1_CH5	I/O	VDDIO_2	General Purpose Timer 1, Channel 5
GPIO_24		GPT1_CH5	I/O	VDDIO_AON	General Purpose Timer 1, Channel 5
GPIO_36	--	GPT1_CLKIN	I	VDDIO_3	General Purpose Timer 1, Clock Input
GPIO_11	--	GPT2_CH0	I/O	VDDIO_0	General Purpose Timer 2, Channel 0
GPIO_12	--	GPT2_CH1	I/O	VDDIO_0	General Purpose Timer 2, Channel 1
GPIO_13	--	GPT2_CH2	I/O	VDDIO_0	General Purpose Timer 2, Channel 2
GPIO_14	--	GPT2_CH3	I/O	VDDIO_0	General Purpose Timer 2, Channel 3
GPIO_15	--	GPT2_CH4	I/O	VDDIO_0	General Purpose Timer 2, Channel 4
GPIO_37	--	GPT2_CH5	I/O	VDDIO_3	General Purpose Timer 2, Channel 5
GPIO_38	--	GPT2_CLKIN	I	VDDIO_3	General Purpose Timer 2, Clock Input
GPIO_17	--	GPT3_CH0	I/O	VDDIO_1	General Purpose Timer 3, Channel 0
GPIO_18	--	GPT3_CH1	I/O	VDDIO_1	General Purpose Timer 3, Channel 1
GPIO_19	--	GPT3_CH2	I/O	VDDIO_1	General Purpose Timer 3, Channel 2
GPIO_20	--	GPT3_CH3	I/O	VDDIO_1	General Purpose Timer 3, Channel 3

Table 7: GPT Interface[1] (Continued)

88-Pin	68-Pin	Signal Name	Type	Supply	Description
GPIO_21	--	GPT3_CH4	I/O	VDDIO_1	General Purpose Timer 3, Channel 4
GPIO_34	--	GPT3_CH5	I/O	VDDIO_3	General Purpose Timer 3, Channel 5
GPIO_39		GPT3_CLKIN	I	VDDIO_3	General Purpose Timer 3, Clock Input

1. All GPT signals are muxed on GPIO pins. See Table 11, GPIO Interface, on page 51 for GPIO muxing.

Table 8: SSP Interface[1]

88-Pin	68-Pin	Pin Name	Type	Supply	Description
GPIO_0		SSP0_CLK	I/O	VDDIO_0	SSP 0 Serial Clock
GPIO_1		SSP0_FRM	I/O	VDDIO_0	SSP 0 Frame Indicator
GPIO_2		SSP0_TXD	O	VDDIO_0	SSP 0 TXD
GPIO_3		SSP0_RXD	I	VDDIO_0	SSP 0 RXD
GPIO_30		SSP0_CLK	I/O	VDDIO_2	SSP 0 Serial Clock
GPIO_31		SSP0_FRM	I/O	VDDIO_2	SSP 0 Frame Indicator
GPIO_32		SSP0_TXD	O	VDDIO_2	SSP 0 TXD
GPIO_33		SSP0_RXD	I	VDDIO_2	SSP 0 RXD
GPIO_11	--	SSP1_CLK	I/O	VDDIO_0	SSP 1 Serial Clock
GPIO_12	--	SSP1_FRM	I/O	VDDIO_0	SSP 1 Frame Indicator
GPIO_13	--	SSP1_TXD	O	VDDIO_0	SSP 1 TXD
GPIO_14	--	SSP1_RXD	I	VDDIO_0	SSP 1 RXD
GPIO_18	--	SSP1_CLK	I/O	VDDIO_1	SSP 1 Serial Clock
GPIO_19	--	SSP1_FRM	I/O	VDDIO_1	SSP 1 Frame Indicator
GPIO_20	--	SSP1_TXD	O	VDDIO_1	SSP 1 TXD
GPIO_21	--	SSP1_RXD	I	VDDIO_1	SSP 1 RXD
GPIO_35	--	SSP1_CLK	I/O	VDDIO_3	SSP 1 Serial Clock
GPIO_36	--	SSP1_FRM	I/O	VDDIO_3	SSP 1 Frame Indicator

Table 8: SSP Interface[1] (Continued)

88-Pin	68-Pin	Pin Name	Type	Supply	Description
GPIO_38	--	SSP1_TXD	O	VDDIO_3	SSP 1 TXD
GPIO_39		SSP1_RXD	I	VDDIO_3	SSP 1 RXD
GPIO_42		SSP1_CLK	I/O	VDDIO_3	SSP 1 Serial Clock
GPIO_43		SSP1_FRM	I/O	VDDIO_3	SSP 1 Frame Indicator
GPIO_44		SSP1_TXD	O	VDDIO_3	SSP 1 TXD
GPIO_45		SSP1_RXD	I	VDDIO_3	SSP 1 RXD
GPIO_7		SSP2_CLK	I/O	VDDIO_0	SSP 2 Serial Clock
GPIO_8		SSP2_FRM	I/O	VDDIO_0	SSP 2 Frame Indicator
GPIO_9		SSP2_TXD	O	VDDIO_0	SSP 2 TXD
GPIO_10		SSP2_RXD	I	VDDIO_0	SSP 2 RXD
GPIO_46		SSP2_CLK	I/O	VDDIO_3	SSP 2 Serial Clock
GPIO_47		SSP2_FRM	I/O	VDDIO_3	SSP 2 Frame Indicator
GPIO_48		SSP2_TXD	O	VDDIO_3	SSP 2 TXD
GPIO_49		SSP2_RXD	I	VDDIO_3	SSP 2 RXD

1. All SSP signals are muxed on GPIO pins. See Table 11, GPIO Interface, on page 51 for GPIO muxing.

Table 9: I²C Interface[1]

88-Pin	68-Pin	Pin Name	Type	Supply	Description
GPIO_4 GPIO_7		I2C0_SDA	I/O	VDDIO_0	I²C 0 SDA
GPIO_5 GPIO_8		I2C0_SCL	I/O	VDDIO_0	I²C 0 SCL
GPIO_6 GPIO_9		I2C1_SDA	I/O	VDDIO_0	I²C 1 SDA
GPIO_10		I2C1_SCL	I/O	VDDIO_0	I²C 1 SCL
GPIO_20	--	I2C0_SDA	I/O	VDDIO_1	I²C 0 SDA
GPIO_21	--	I2C0_SCL	I/O	VDDIO_1	I²C 0 SCL
GPIO_18	--	I2C1_SDA	I/O	VDDIO_1	I²C 1 SDA
GPIO_17 GPIO_19	--	I2C1_SCL	I/O	VDDIO_1	I²C 1 SCL
GPIO_25		I2C1_SDA	I/O	VDDIO_AON	I²C 1 SDA
GPIO_26		I2C1_SCL	I/O	VDDIO_AON	I²C 1 SCL
GPIO_28		I2C0_SDA	I/O	VDDIO_2	I²C 1 SDA
GPIO_29		I2C0_SCL	I/O	VDDIO_2	I²C 1 SCL

1. All I²C signals are muxed on GPIO pins. See Table 11, GPIO Interface, on page 51 for GPIO muxing.

Table 10: QSPI Interface[1]

88-Pin	68-Pin	Pin Name	Type	Supply	Description
GPIO_28		QSPI_SSn	O	VDDIO_2	QSPI Chip Select (active low)
GPIO_29		QSPI_CLK	O	VDDIO_2	QSPI Clock
GPIO_30		QSPI_D0	I/O	VDDIO_2	QSPI Data 0
GPIO_31		QSPI_D1	I/O	VDDIO_2	QSPI Data 1
GPIO_32		QSPI_D2	I/O	VDDIO_2	QSPI Data 2
GPIO_33		QSPI_D3	I/O	VDDIO_2	QSPI Data 3

1. QSPI signals are used for external Flash only. All QSPI signals are muxed on GPIO pins. See Table 11, GPIO Interface, on page 51 for GPIO muxing.

Table 11: GPIO Interface [12]

88-Pin	68-Pin	Pin/Signal Name	Type	Supply	Description
1	1	GPIO_0	I/O	VDDIO_0	General Purpose I/O 0
		GPT0_CH0	I/O		General Purpose Timer 0, Channel 0
		UART0_CTSn	I		UART 0 CTSn (active low)
		SSP0_CLK	I/O		SSP 0 Serial Clock
2	2	GPIO_1	I/O	VDDIO_0	General Purpose I/O 1
		GPT0_CH1	I/O		General Purpose Timer 0, Channel 1
		UART0_RTSn	O		UART 0 RTSn (active low)
		SSP0_FRM	I/O		SSP 0 Frame Indicator
3	3	GPIO_2	I/O	VDDIO_0	General Purpose I/O 2
		GPT0_CH2	I/O		General Purpose Timer 0, Channel 2
		UART0_TXD	O		UART 0 TXD
		SSP0_TXD	O		SSP 0 TXD
4	4	GPIO_3	I/O	VDDIO_0	General Purpose I/O 3
		GPT0_CH3	I/O		General Purpose Timer 0, Channel 3
		UART0_RXD	I		UART 0 RXD
		SSP0_RXD	I		SSP 0 RXD
6	6	GPIO_4	I/O	VDDIO_0	General Purpose I/O 4
		GPT0_CH4	I/O		General Purpose Timer 0, Channel 4
		I2C0_SDA	I/O		I^2C 0 SDA
		AUDIO_CLK	O		Audio Clock AUPLL Audio clock output provided by Audio PLL for external codec.
7	7	GPIO_5	I/O	VDDIO_0	General Purpose I/O 5
		GPT0_CH5	I/O		General Purpose Timer 0, Channel 5
		I2C0_SCL	I/O		I^2C 0 SCL

Table 11: GPIO Interface [12] (Continued)

88-Pin	68-Pin	Pin/Signal Name	Type	Supply	Description
8	8	GPIO_6	I/O	VDDIO_0	General Purpose I/O 6
		TDO	O		JTAG Test Data
		I2C1_SDA	I/O		I^2C 1 SDA
9	9	GPIO_7	I/O	VDDIO_0	General Purpose I/O 7
		TCK	I		JTAG Test Clock
		UART2_CTSn	I		UART 2 CTSn (active low)
		SSP2_CLK	I/O		SSP 2 Serial Clock
		I2C0_SDA	I/O		I^2C 0 SDA
10	10	GPIO_8	I/O	VDDIO_0	General Purpose I/O 8
		TMS	I/O		JTAG Controller Select
		UART2_RTSn	O		UART 2 RTSn (active low)
		SSP2_FRM	I/O		SSP_2 Frame Indicator
		I2C0_SCL	I/O		I^2C 0 SCL
11	11	GPIO_9	I/O	VDDIO_0	General Purpose I/O 9
		TDI	I		JTAG Test Data
		UART2_TXD	O		UART 2 TXD
		SSP2_TXD	O		SSP 2 TXD
		I2C1_SDA	I/O		I^2C 1 SDA
12	12	GPIO_10	I/O	VDDIO_0	General Purpose I/O 10
		TRSTn	I		JTAG Test Reset (active low)
		UART2_RXD	I		UART 2 RXD
		SSP2_TXD	O		SSP 2 RXD
		I2C1_SCL	I/O		I^2C 1 SCL

Table 11: GPIO Interface [12] (Continued)

88-Pin	68-Pin	Pin/Signal Name	Type	Supply	Description
13	--	GPIO_11	I/O	VDDIO_0	General Purpose I/O 11
		GPT2_CH0	I/O		General Purpose Timer 2, Channel 0
		UART1_CTSn	I		UART 1 CTSn (active low)
		SSP1_CLK	I/O		SSP 1 Serial Clock
14	--	GPIO_12	I/O	VDDIO_0	General Purpose I/O 12
		GPT2_CH1	I/O		General Purpose Timer 2, Channel 1
		UART1_RTSn	O		UART 1 RTSn (active low)
		SSP1_FRM	I/O		SSP 1 Frame Indicator
15	--	GPIO_13	I/O	VDDIO_0	General Purpose I/O 13
		GPT2_CH2	I/O		General Purpose Timer 2, Channel 2
		UART1_TXD	O		UART 1 TXD
		SSP1_TXD	O		SSP 1 TXD
16	--	GPIO_14	I/O	VDDIO_0	General Purpose I/O 14
		GPT2_CH3	I/O		General Purpose Timer 2, Channel 3
		UART1_RXD	I		UART 1 RXD
		SSP1_RXD	I		SSP 1 RXD
17	--	GPIO_15	I/O	VDDIO_0	General Purpose I/O 15
		GPT2_CH4	I/O		General Purpose Timer 2, Channel 4
35	30	GPIO_16	I/O	VDDIO_1	General Purpose I/O 16
		CON[5]	I/O		Configuration Bit See Table 17, Configuration Pins, on page 65.
		AUDIO_CLK	O		Audio Clock AUPLL Audio clock output provided by Audio PLL for external codec.

Table 11: GPIO Interface [12] (Continued)

88-Pin	68-Pin	Pin/Signal Name	Type	Supply	Description
36	--	GPIO_17	I/O	VDDIO_1	General Purpose I/O 17
		GPT3_CH0	I/O		General Purpose Timer 3, Channel 0
		I2C1_SCL	I/O		I^2C 1 SCL
37	--	GPIO_18	I/O	VDDIO_1	General Purpose I/O 18
		GPT3_CH1	I/O		General Purpose Timer 3, Channel 1
		I2C1_SDA	I/O		I^2C 1 SDA
		SSP1_CLK	I/O		SSP 1 Serial Clock
38	--	GPIO_19	I/O	VDDIO_1	General Purpose I/O 19
		GPT3_CH2	I/O		General Purpose Timer 3, Channel 2
		I2C1_SCL	I/O		I^2C 1 SDA
		SSP1_FRM	I/O		SSP 1 Frame Indicator
39	--	GPIO_20	I/O	VDDIO_1	General Purpose I/O 20
		GPT3_CH3	I/O		General Purpose Timer 3, Channel 3
		I2C0_SDA	I/O		I^2C 0 SDA
		SSP1_TXD	O		SSP 1 TXD
40	--	GPIO_21	I/O	VDDIO_1	General Purpose I/O 21
		GPT3_CH4	I/O		General Purpose Timer 3, Channel 4
		I2C0_SCL	I/O		I^2C 0 SCL
		SSP1_RXD	I		SSP 1 RXD
46	36	GPIO_22	I/O	VDDIO_ AON	General Purpose I/O 22
		WAKE_UP0	I		Wake-Up 0

Table 11: GPIO Interface [12] (Continued)

88-Pin	68-Pin	Pin/Signal Name	Type	Supply	Description
47	37	GPIO_23	I/O	VDDIO_AON	General Purpose I/O 23
		UART0_CTSn	I		UART 0 CTSn (active low)
		WAKE_UP1	I		Wake-Up 1
		COMP_IN_P	I		LDO18 Comparator Input, Positive Positive input to LDO18 comparator.
48	38	GPIO_24	I/O	VDDIO_AON	General Purpose I/O 24
		UART0_RXD	I		UART 0 RXD
		GPT1_CH5	I/O		General Purpose Timer 1, Channel 5
		COMP_IN_N	I		LDO18 Comparator Input, Negative Negative input to LDO18 comparator.
49	39	GPIO_25	I/O	VDDIO_AON	General Purpose I/O 25
		XTAL32K_IN	I		32.768 kHz Crystal Input
		I2C1_SDA	I/O		I^2C 1 SDA
50	40	GPIO_26	I/O	VDDIO_AON	General Purpose I/O 26
		XTAL32K_OUT	O		32.768 kHz Crystal Output
		I2C1_SCL	I/O		I^2C 1 SCL
66	51	GPIO_27	I/O	VDDIO_3	General Purpose I/O 27
	--	USB_DRV_VBUS	O		Drive 5V on VBUS
	51	UART0_TXD	O		UART 0 TXD
	51	CON[4]	I/O		Configuration Bit See Table 17, Configuration Pins, on page 65.
52	42	GPIO_28	I/O	VDDIO_2	General Purpose I/O 28
		QSPI_SSn	O		QSPI Chip Select (active low)
		I2C0_SDA	I/O		I^2C 0 SDA
		GPT1_CH0	I/O		General Purpose Timer 1, Channel 0

Table 11: GPIO Interface [12] (Continued)

88-Pin	68-Pin	Pin/Signal Name	Type	Supply	Description
53	43	GPIO_29	I/O	VDDIO_2	General Purpose I/O 29
		QSPI_CLK	O		QSPI Clock
		I2C0_SCL	I/O		I^2C 0 SCL
		GPT1_CH1	I/O		General Purpose Timer 1, Channel 1
54	44	GPIO_30	I/O	VDDIO_2	General Purpose I/O 30
		QSPI_D0	I/O		QSPI Data 0
		UART0_CTSn	I		UART 0 CTSn (active low)
		SSP0_CLK	I/O		SSP 0 Serial Clock
		GPT1_CH2	I/O		General Purpose Timer 1, Channel 2
55	45	GPIO_31	I/O	VDDIO_2	General Purpose I/O 31
		QSPI_D1	I/O		QSPI Data 1
		UART0_RTSn	O		UART 0 RTSn (active low)
		SSP0_FRM	I/O		SSP 0 Frame Indicator
		GPT1_CH3	I/O		General Purpose Timer 1, Channel 3
56	46	GPIO_32	I/O	VDDIO_2	General Purpose I/O 32
		QSPI_D2	I/O		QSPI Data 2
		UART0_TXD	O		UART 0 TXD
		SSP0_TXD	O		SSP 0 TXD
		GPT1_CH4	I/O		General Purpose Timer 1, Channel 4
57	47	GPIO_33	I/O	VDDIO_2	General Purpose I/O 33
		QSPI_D3	I/O		QSPI Data 3
		UART0_RXD	I		UART 0 RXD
		SSP0_RXD	I		SSP 0 RXD
		GPT1_CH5	I/O		General Purpose Timer 1, Channel 5

Table 11: GPIO Interface [12] (Continued)

88-Pin	68-Pin	Pin/Signal Name	Type	Supply	Description
67	--	GPIO_34	I/O	VDDIO_3	General Purpose I/O 34
		GPT3_CH5	I/O		General Purpose Timer 3, Channel 5
68	--	GPIO_35	I/O	VDDIO_3	General Purpose I/O 35
		GPT0_CLKIN	I		General Purpose Timer 0, Clock Input
		UART1_CTSn	I		UART 1 CTSn (active low)
		SSP1_CLK	I/O		SSP 1 Serial Clock
69	--	GPIO_36	I/O	VDDIO_3	General Purpose I/O 36
		GPT1_CLKIN	I		General Purpose Timer 1, Clock Input
		UART1_RTSn	O		UART 1 RTSn (active low)
		SSP1_FRM	I/O		SSP 1 Frame Indicator
70	--	GPIO_37	I/O	VDDIO_3	General Purpose I/O 37
		GPT2_CH5	I/O		General Purpose Timer 2, Channel 5
		UART0_RTSn	O		UART 0 RTSn (active low)
71	--	GPIO_38	I/O	VDDIO_3	General Purpose I/O 38
		GPT2_CLKIN	I		General Purpose Timer 2, Clock Input
		UART1_TXD	O		UART 1 TXD
		SSP1_TXD	O		SSP 1 TXD
72	52	GPIO_39	I/O	VDDIO_3	General Purpose I/O 39
		GPT3_CLKIN	I		General Purpose Timer 2, Clock Input
		UART1_RXD	I		UART 1 RXD
		SSP1_RXD	I		SSP 1 RXD

Table 11: GPIO Interface [12] (Continued)

88-Pin	68-Pin	Pin/Signal Name	Type	Supply	Description
75	55	GPIO_40	I/O	VDDIO_3	General Purpose I/O 40
		ADC_DAC_TRIGGER0	I		ADC/DAC External Trigger 0
		ACOMP0_GPIO_OUT	O		ACOMP0 GPIO Output ACOMP0 output synchronous or asynchronous level signals.
		ACOMP1_GPIO_OUT	O		ACOMP1 GPIO Output ACOMP1 output synchronous or asynchronous level signals.
76	56	GPIO_41	I/O	VDDIO_3	General Purpose I/O 41
		ADC_DAC_TRIGGER1	I		ADC/DAC External Trigger 1
		ACOMP0_EDGE_ PULSE	O		ACOMP Edge Pulse 0 Output pulse aligned with synchronized comparison result.
		ACOMP1_EDGE_ PULSE	O		ACOMP Edge Pulse 1 Output pulse aligned with synchronized comparison result.
78	58	GPIO_42	I/O	VDDIO_3	General Purpose I/O 42
		ADC0_0 / ACOMP0 / TS_INP / VOICE_P	A, I		ADC0 Channel 0 ACOMP0 Channel 0 ACOMP1 Channel 0 Temperature sensor remote sensing positive input Voice sensing positive input
		UART1_CTSn	I		UART 1 CTSn (active low)
		SSP1_CLK	I/O		SSP 1 Serial Clock

Table 11: GPIO Interface [12] (Continued)

88-Pin	68-Pin	Pin/Signal Name	Type	Supply	Description
79	59	GPIO_43	I/O	VDDIO_3	General Purpose I/O 43
		ADC0_1 / ACOMP1 / TS_INN / DACB / VOICE_N	A, I/O		ADC0 Channel 1 ACOMP0 Channel 1 ACOMP1 Channel 1 Temperature sensor remote sensing negative input Voice sensing negative input
		UART1_RTSn	O		UART 1 RTSn (active low)
		SSP1_FRM	I/O		SSP 1 Frame Indicator
80	60	GPIO_44	I/O	VDDIO_3	General Purpose I/O 44
		ADC0_2 / ACOMP2 / DACA	A, I/O		ADC0 Channel 2 ACOMP0 Channel 2 ACOMP1 Channel 2 DAC Channel A output
		UART1_TXD	O		UART 1 TXD
		SSP1_TXD	O		SSP 1 TXD
		RF_CNTL1_P	O		WLAN Radio Control 1
81	61	GPIO_45	I/O	VDDIO_3	General Purpose I/O 45
		ADC0_3 / ACOMP3 / EXT_VREF	A, I		ADC0 Channel 3 ACOMP0 Channel 3 ACOMP1 Channel 3 ADC or DAC external voltage reference input
		UART1_RXD	I		UART 1 RXD
		SSP1_RXD	I		SSP 1 RXD
		RF_CNTL0_N	O		WLAN Radio Control 0
82	62	GPIO_46	I/O	VDDIO_3	General Purpose I/O 46
		ADC0_4 / ACOMP4	A, I		ADC0 Channel 4 ACOMP0 Channel 4 ACOMP1 Channel 4
		UART2_CTSn	I		UART 2 CTSn (active low)
		SSP2_CLK	I/O		SSP 2 Serial Clock

Table 11: GPIO Interface [12] (Continued)

88-Pin	68-Pin	Pin/Signal Name	Type	Supply	Description
83	63	GPIO_47	I/O	VDDIO_3	General Purpose I/O 47
		ADC0_5 / ACOMP5	A, I		ADC0 Channel 5 ACOMP0 Channel 5 ACOMP1 Channel 5
		UART2_RTSn	O		UART 2 RTSn (active low)
		SSP2_FRM	I/O		SSP 2 Frame Indicator
84	64	GPIO_48	I/O	VDDIO_3	General Purpose I/O 48
		ADC0_6 / ACOMP6	A, I		ADC0 Channel 6 ACOMP0 Channel 6 ACOMP1 Channel 6
		UART2_TXD	O		UART 2 TXD
		SSP2_TXD	O		SSP 2 TXD
85	65	GPIO_49	I/O	VDDIO_3	General Purpose I/O 49
		ADC0_7 / ACOMP7	A, I		ADC0 Channel 7 ACOMP0 Channel 7 ACOMP1 Channel 7
		UART2_RXD	I		UART 2 RXD
		SSP2_RXD	I		SSP 2 RXD

1. GPIO_11 to GPIO_15, GPIO_17 to GPIO_21, GPIO_34 to GPIO_38 I/O and associated muxing available on 88-pin QFN only.
2. All GPIO pins are pull-up high after POR.

Table 12: Clock/Control Interface[1]

88-Pin	68-Pin	Pin/Signal Name	Type	Supply	Description
26	21	XTAL_IN	A, I	AVDD18	Crystal Oscillator Input
27	22	XTAL_OUT	A, O	AVDD18	Crystal Oscillator Output Connect to ground when an external oscillator used.
GPIO_25		XTAL32K_IN	A, I	VDDIO_AON	32.768 kHz Crystal Input
GPIO_26		XTAL32K_OUT	A, O	VDDIO_AON	32.768 kHz Crystal Output
GPIO_22		WAKE_UP0	I	VDDIO_AON	Wake-Up 0
GPIO_23		WAKE_UP1	I	VDDIO_AON	Wake-Up 1
GPIO_4 GPIO_16		AUDIO_CLK		VDDIO_0	Audio Clock AUPLL audio clock output provided by audio PLL for external codec.
45	35	RESETn	I	VDDIO_AON	Chip Reset (active low) Has internal hardwired non-programmable 10 kohm pull-up to VDDIO_AON.

1. The XTAL32K_IN/OUT and WAKE_UP0/1 signals are muxed on GPIO pins. See Table 11, GPIO Interface, on page 51 for GPIO muxing.

Table 13: ADC/DAC/ACOMP Interface[1]

88-Pin	68-Pin	Pin Name	Type	Supply	Description
GPIO_40		ACOMP0_GPIO_OUT	O	VDDIO_3	ACOMP0 GPIO Output ACOMP0 output synchronous or asynchronous level signals
		ACOMP1_GPIO_OUT	O	VDDIO_3	ACOMP1 GPIO Output ACOMP1 output synchronous or asynchronous level signals
		ADC_DAC_TRIGGER0	I	VDDIO_3	ADC/DAC External Trigger 0
GPIO_41		ACOMP0_EDGE_PULSE	O	VDDIO_3	ACOMP Edge Pulse 0 Output pulse aligned with synchronized comparison result.
		ACOMP1_EDGE_PULSE	O	VDDIO_3	ACOMP Edge Pulse 1 Output pulse aligned with synchronized comparison result.
		ADC_DAC_TRIGGER1	I	VDDIO_3	ADC/DAC External Trigger 1
GPIO_49		ADC0_7 / ACOMP7	A, I	VDDIO_3	ADC0 Channel 7 ACOMP0 Channel 7 ACOMP1 Channel 7
GPIO_48		ADC0_6 / ACOMP6	A, I	VDDIO_3	ADC0 Channel 6 ACOMP0 Channel 6 ACOMP1 Channel 6
GPIO_47		ADC0_5 / ACOMP5	A, I	VDDIO_3	ADC0 Channel 5 ACOMP0 Channel 5 ACOMP1 Channel 5
GPIO_46		ADC0_4 / ACOMP4	A, I	VDDIO_3	ADC0 Channel 4 ACOMP0 Channel 4 ACOMP1 Channel 4
GPIO_45		ADC0_3 / ACOMP3 / EXT_VREF	A, I	VDDIO_3	ADC0 Channel 3 ACOMP0 Channel 3 ACOMP1 Channel 3 ADC or DAC external voltage reference input
GPIO_44		ADC0_2 / ACOMP2 / DACA	A, I	VDDIO_3	ADC0 Channel 2 ACOMP0 Channel 2 ACOMP1 Channel 2 DAC Channel A output

Table 13: ADC/DAC/ACOMP Interface[1] (Continued)

88-Pin	68-Pin	Pin Name	Type	Supply	Description
GPIO_43		ADC0_1 / ACOMP1 / TS_INN / DACB / VOICE_N	A, I	VDDIO_3	ADC0 Channel 1 ACOMP0 Channel 1 ACOMP1 Channel 1 Temperature sensor remote sensing negative input Voice sensing negative input
GPIO_42		ADC0_0 / ACOMP0 / TS_INP / VOICE_P	A, I	VDDIO_3	ADC0 Channel 0 ACOMP0 Channel 0 ACOMP1 Channel 0 Temperature sensor remote sensing positive input Voice sensing positive input

1. All ADC/DAC/ACOMP signals are muxed on GPIO pins. See Table 11, GPIO Interface, on page 51 for GPIO muxing.

Table 14: LDO18 Comparator Interface[1]

88-Pin	68-Pin	Pin Name	Type	Supply	Description
GPIO_23		COMP_IN_P	A, I	VDDIO_AON	LDO18 Comparator Input, Positive Positive input to LDO18 comparator.
GPIO_24		COMP_IN_N	A, I	VDDIO_AON	LDO18 Comparator Input, Negative Positive input to LDO18 comparator.

1. All COMP signals are muxed on GPIO pins. See Table 11, GPIO Interface, on page 51 for GPIO muxing.

Table 15: JTAG Interface[1]

88-Pin	68-Pin	Signal Name	Type	Supply	Description
GPIO_6		TDO	O	VDDIO_0	JTAG Test Data
GPIO_7		TCK	I	VDDIO_0	JTAG Test Clock
GPIO_8		TMS	I/O	VDDIO_0	JTAG Controller Select
GPIO_9		TDI	I	VDDIO_0	JTAG Test Data
GPIO_10		TRSTn	I	VDDIO_0	JTAG Test Reset I/O (active low)

1. All JTAG signals are muxed on GPIO pins. See Table 11, GPIO Interface, on page 51 for GPIO muxing.

Table 16: Power and Ground

NOTE: See Section 22.2, Recommended Operating Conditions, on page 316 for ratings.

88-Pin	68-Pin	Pin Name	Type	Description
33 59	28 49	VDD11	PWR	1.1V Core Power Supply Input
5	5	VDDIO_0	PWR	1.8V/2.5V/3.3V Digital I/O Power Supply
34	29	VDDIO_1		
58	48	VDDIO_2		
77	57	VDDIO_3		
73	53			
51	41	VDDIO_AON		
60	50	VTR_VDD33	PWR	3.3V OTP Write Operation or Floating for OTP Read Operation
63	--	USB_AVDD33	PWR	3.3V USB Analog Power Supply
74	54	ISENSE	--	USB Current Source Connect pin to ground with resistance.
20	15	AVDD33	PWR	3.3V Analog Power Supply
18 19 23 24 25 28	13 14 18 19 20 23	AVDD18	PWR	1.8V Analog Power Supply
41	31	LDO11_VOUT	PWR	1.1V LV LDO Voltage Output
42	32	LDO11_V18	--	BUCK18 Inductor Connection
43	33	BUCK18_VX	--	BUCK18 Inductor Connection
44	34	BUCK18_VBAT_IN	PWR	BUCK18 Input 2.4V to 4.3V
87	67	VBAT_IN	PWR	LDO18 VBAT Input This pin connects to VBAT source 1.84V to 3.6V.
86	66	FLY18	--	1.8V LDO Fly Capacitor to Ground Connection
88	68	FLY11	--	1.1V LDO Fly Capacitor to Ground Connection
22	17	NC	NC	No Connect NOTE: CONNECT THESE PINS to GROUND.
29 30 31 32	24 25 26 27	DNC	DNC	Do Not Connect Do not connect these pins. Leave these pins floating.

1.5 Configuration Pins

Table 17 shows the pins used as configuration inputs to set parameters following a reset. The definition of these pins changes immediately after reset to their usual function. To set a configuration bit to 0, attach a 100 kΩ resistor from the pin to ground. No external circuitry is required to set a configuration bit to 1.

Table 17: Configuration Pins

Configuration Bits	Pin Name	Configuration Function
CON[5]	GPIO_16	Boot Options 00 = boot from UART 01 = reserved 10 = boot from USB 11 = boot from Flash (default)
CON[4]	GPIO_27	

THIS PAGE INTENTIONALLY LEFT BLANK

2 Core and System Control

2.1 Overview

The 88MW300/302 integrates a full-featured ARM Cortex-M4F core processor.

2.2 Cortex-M4F Core

The ARM Cortex-M4F processor provides a high performance and low-cost platform. It offers many new features, including a Thumb-2 instruction set, low interrupt latency, hardware division, memory protection unit, etc.

Details of the ARM Cortex-M4F core are available in the ARM Cortex-M4F r2p1 technical reference manual.

2.2.1 Features

- 32-bit ARM Cortex-M4F architecture optimized for embedded applications
- Cortex-M4F core can operate at up to 200 MHz
- Thumb-2 mixed 16/32-bit instruction set
- Hardware division and fast multiplier
- Little-endian memory space
- Memory protection unit (MPU) for protected operating system functionality
- Includes Nested Vectored Interrupt Controller (NVIC)
- SysTick Timer provided by Cortex-M4F core
- Wake-up Interrupt Controller (WIC) for waking up the CPU from reduced power modes
- Standard JTAG debug interface
- Serial Wire JTAG debug port (SWJ-DP)
- Enhanced system debug with extensive breakpoint

2.2.2 Memory Protection Unit (MPU)

The Memory Protection Unit (MPU) is used to improve the reliability of an embedded system by protecting critical data in the applications.

The MPU separates the memory into distinct regions and implements protection by preventing disallowed accesses. The MPU can manage as many as 8 protection regions. The protection area sizes are between 32 bytes and all 4 gigabytes of addressable memory.

The MPU is optional and can be bypassed for applications that do not need it.

2.2.3 Nested Vectored Interrupt Controller (NVIC)

The NVIC is integrated in the Cortex-M4F core. The tight coupling to the CPU allows for low interrupt latency and efficient processing of interrupts. Features include:

- Supports up to 63 interrupts
- Supports 16 interrupt priority levels, Level 0 is the highest interrupt priority
- Control system exceptions and peripheral interrupts
- Supports interrupt tail chaining
- Non-maskable interrupt

See Section 2.4, Memory Map, on page 69 for a detailed description of the interrupts.

2.2.4 SysTick Timer

The ARM Cortex-M4F includes a system tick timer (SysTick). This timer is dedicated to real-time operating systems, but could also be used as a standard downcounter. Features include:

- 24-bit downcounter
- Auto reload capability
- Maskable system interrupt generation when counter reaches 0
- Programmable clock source

2.3 System Control

The 88MW300/302 System Control unit provides several system features and control registers for memory space configuration, DMA handshake interface mapping, peripheral software reset, and USB control.

These registers must be configured correctly to ensure correct functionality of memories, DMA, USB, and other peripherals on-chip.

- DMA handshake interface mapping – The DMA_HS register enables mapping the DMA handshaking interface to the required DMA channel in order to perform DMA transfers for different peripherals on-chip.
- Peripheral software reset – The PERI_SW_RST register is used to program reset for various peripherals on the chip. Writing 0 to certain bits resets the corresponding module. It resets only the function clock domain.

See Appendix A.21.2, System Control Registers, on page 825 for a detailed description of the registers.

2.4 Memory Map

The 88MW300/302 includes 128 KB Boot ROM and ROM, 4 KB SRAM in AON domain, and 512 KB of SRAM on-chip. ROM is allocated on the CODE space. The 512 KB SRAM is allocated to the CODE and SRAM space.

The Cortex-M4F CPU accesses the CODE memory space using the ICODE or DCODE AHB bus interface, and accesses the SRAM space with the SYS AHB bus interface.

The 512 KB SRAM memory consists of 4 segments:

- RAM0, 192 KB
- RAM1, 192 KB
- RAM2, 64 KB
- RAM3, 64 KB

RAM0 and RAM1 are part of the CODE space, and RAM2 and RAM3 are part of the SRAM space. 192 KB of SRAM memory can be in Retention mode even in PM3 low-power mode. With this 192 KB retention SRAM, the chip can implement the fast wake-up from PM3 mode.

The 4 KB SRAM in the AON domain is mapped to the peripheral address space and begins at address 0x480C_0000.The on-chip peripherals are mapped to the peripheral address space. Accessing reserved portions of the peripheral address space does not cause a data abort or error response but does provide undetermined data.

Figure 7 and Table 18 show the system memory map.

Figure 7: System Memory Map Diagram

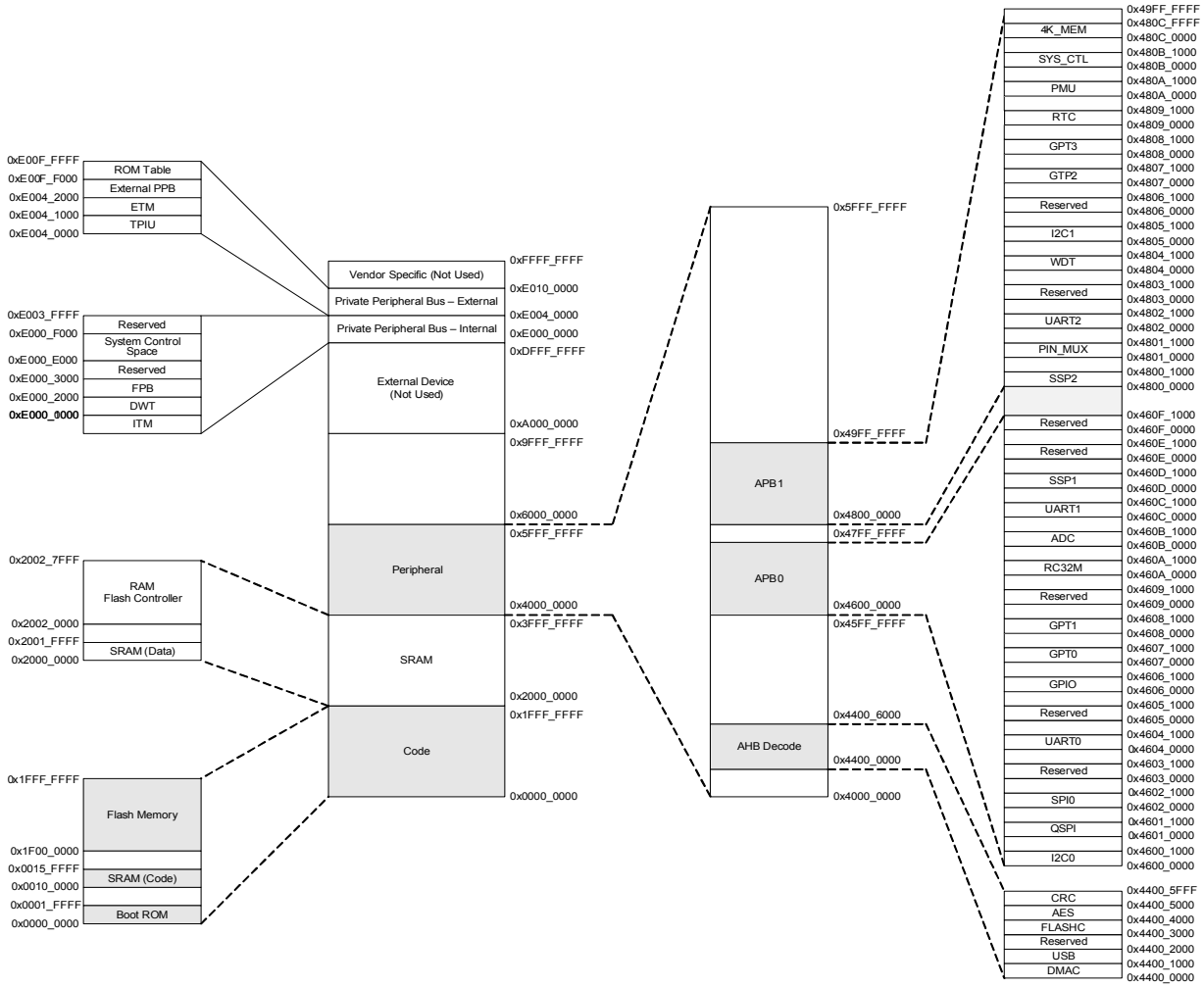


Table 18: System Address Memory Map

Module	Start Address	End Address
Memory		
ROM	0x0000_0000	0x0001_FFFF
Code RAM	0x0010_0000	0x0015_FFFF
Flash Memory	0x1F00_0000	0x1FFF_FFFF
SRAM RAM	0x2000_0000	0x2001_FFFF
SRAM Flash Controller	0x2002_0000	0x2002_7FFF
AHB		
DMAC	0x4400_0000	0x4400_0FFF
USB	0x4400_1000	0x4400_1FFF
Flash Controller	0x4400_3000	0x4400_3FFF
AES	0x4400_4000	0x4400_4FFF
CRC	0x4400_5000	0x4400_5FFF
APB0		
I2C0	0x4600_0000	0x4600_0FFF
QSPI	0x4601_0000	0x4601_0FFF
SSP0	0x4602_0000	0x4602_0FFF
UART0	0x4604_0000	0x4604_0FFF
GPIO	0x4606_0000	0x4606_0FFF
GPT0	0x4607_0000	0x4607_0FFF
GPT1	0x4608_0000	0x4608_0FFF
Reserved	0x4609_0000	0x4609_0FFF
RC32M	0x460A_0000	0x460A_0FFF
ADC	0x460B_0000	0x460B_00FF
DAC	0x460B_0200	0x460B_02FF
ACOMP	0x460B_0400	0x460B_04FF
UART1	0x460C_0000	0x460C_0FFF
SSP1	0x460D_0000	0x460C_0FFF
APB1		
SSP2	0x4800_0000	0x4800_0FFF

Table 18: System Address Memory Map (Continued)

Module	Start Address	End Address
Pin Mux	0x4801_0000	0x4801_0FFF
UART2	0x4802_0000	0x4802_0FFF
Watchdog Timer	0x4804_0000	0x4804_0FFF
I2C1	0x4805_0000	0x4805_0FFF
Reserved	0x4806_0000	0x4806_0FFF
GPT2	0x4807_0000	0x4807_0FFF
GPT3	0x4808_0000	0x4808_0FFF
RTC	0x4809_0000	0x4809_0FFF
PMU	0x480A_0000	0x480A_0FFF
SYS_CTL	0x480B_0000	0x480B_0FFF
4k_MEM	0x480C_0000	0x480C_0FFF

Accesses to unmapped addresses of RAM1 and RAM2 are provided with an error response. Writes to the ROM space, if any, also yield an error response.

Table 19 shows the available RAM blocks.

Table 19: RAM Blocks

RAM Blocks	Start Address	End Address
RAM0		
Bank0	0x0010_0000	0x0010_7FFF
Bank1	0x0010_8000	0x0010_FFFF
Bank2	0x0011_0000	0x0011_7FFF
Bank3	0x0011_8000	0x0011_FFFF
Bank4	0x0012_0000	0x0012_7FFF
Bank5	0x0012_8000	0x0012_FFFF
RAM1		
Bank6	0x0013_0000	0x0013_7FFF
Bank7	0x0013_8000	0x0013_FFFF
Bank8	0x0014_0000	0x0014_7FFF
Bank9	0x0014_8000	0x0014_FFFF
Bank10	0x0015_0000	0x0015_7FFF

Table 19: RAM Blocks (Continued)

RAM Blocks	Start Address	End Address
Bank11	0x0015_8000	0x0015_FFFF
RAM2		
Bank12	0x2000_0000	0x2000_7FFF
Bank13	0x2000_8000	0x2000_FFFF
RAM3		
Bank14	0x2001_0000	0x2001_7FFF
Bank15	0x2001_8000	0x2001_FFFF

Appendix A.21.2, System Control Registers, on page 825 shows a detailed description of the memory configuration register.

192 KB of the 512 KB SRAM can be in Retention mode in PM3 low-power mode. 160 KB retention SRAM is located in the CODE space, starting from address 0x0010_0000.

2.5 External Interrupts

2.5.1 Interrupts Accepted

The 88MW300/302 can accept external interrupts through the NVIC module in the Cortex-M4F processor. Table 20 shows the interrupts.

Table 20: External Interrupts

Name	Source	Type	Polarity	Map
WD Timeout	WDT	Level	Active High	INTNMI
LOCKUP	Cortex-M4F			
Ext. Pin 0	External	Configurable	Active High	INTIRQ[0]
Ext. Pin 1	External	Configurable	Active High	INTIRQ[1]
RTC INT	RTC	Level	Active High	INTIRQ[2]
CRC INT	CRC	Level	Active High	INTIRQ[3]
AES INT	AES	Level	Active High	INTIRQ[4]
I2C0 INT	I2C 0	Level	Active High	INTIRQ[5]
I2C1 INT	I2C 1	Level	Active High	INTIRQ[6]
DMAC INT	DMAC	Level	Active High	INTIRQ[8]
GPIO INT	GPIO	Level	Active High	INTIRQ[9]
SSP0 INT	SSP 0	Level	Active High	INTIRQ[10]
SSP1 INT	SSP 1	Level	Active High	INTIRQ[11]
SSP2 INT	SSP 2	Level	Active High	INTIRQ[12]
QSPI INT	QSPI	Level	Active High	INTIRQ[13]
GPT0 INT	GPT 0	Level	Active High	INTIRQ[14]
GPT1 INT	GPT 1	Level	Active High	INTIRQ[15]
GPT2 INT	GPT 2	Level	Active High	INTIRQ[16]
GPT3 INT	GPT 3	Level	Active High	INTIRQ[17]
UART0 INT	UART 0	Level	Active High	INTIRQ[18]
UART1 INT	UART 1	Level	Active High	INTIRQ[19]
UART2 INT	UART 2	Level	Active High	INTIRQ[20]
WDT INT	WDT	Level	Active High	INTIRQ[22]
ADC0 INT	GAU	Level	Active High	INTIRQ[24]
DAC INT	GAU	Level	Active High	INTIRQ[25]

Table 20: External Interrupts (Continued)

Name	Source	Type	Polarity	Map
ACOMP Wake-up INT	GAU	Level	Active High	INTIRQ[26]
ACOMP INT	GAU	Level	Active High	INTIRQ[27]
USB INT	USB	Level	Active High	INTIRQ[29]
PLL INT	PMU	Level	Active High	INTIRQ[31]
RC32M INT FUNC	RC32M	Level	Active High	INTIRQ[33]
External Pin INT	PMU	Level	Active High	INTIRQ[58:34]
ULP_COMP	PMU	Level	Active High	INTIRQ[60]
Brnout INT	Brnout	Level	Active High	INTIRQ[61]

External pin interrupts connected to INTIRQ[58:34] are generated using GPIOs in the design by programming the PMU.EXT_SEL_REG register bits to the required value.

2.5.2 GPIO Mapping of Interrupts

The PMU.EXT_SEL_REG0 register is used to select the GPIO connected to an external interrupt on the Cortex-M4F.

See Appendix A.20, PMU Address Block, on page 769 for information on register programming.

Table 21 shows the GPIO mapping of the external interrupts.

Table 21: GPIO Mapping to External Interrupts

External Interrupt Bit	GPIO Connected	External Select Register Value
34	GPIO[0]	ext_sel_reg0[0] = 1
	GPIO[1]	ext_sel_reg0[0] = 0
35	GPIO[2]	ext_sel_reg0[1] = 1
	GPIO[3]	ext_sel_reg0[1] = 0
36	GPIO[4]	ext_sel_reg0[2] = 1
	GPIO[5]	ext_sel_reg0[2] = 0
37	GPIO[6]	ext_sel_reg0[3] = 1
	GPIO[7]	ext_sel_reg0[3] = 0
38	GPIO[8]	ext_sel_reg0[4] = 1
	GPIO[9]	ext_sel_reg0[4] = 0
39	GPIO[10]	ext_sel_reg0[5] = 1
	GPIO[11]	ext_sel_reg0[5] = 0
40	GPIO[12]	ext_sel_reg0[6] = 1
	GPIO[13]	ext_sel_reg0[6] = 0
41	GPIO[14]	ext_sel_reg0[7] = 1
	GPIO[15]	ext_sel_reg0[7] = 0
42	GPIO[16]	ext_sel_reg0[8] = 1
	GPIO[17]	ext_sel_reg0[8] = 0
43	GPIO[18]	ext_sel_reg0[9] = 1
	GPIO[19]	ext_sel_reg0[9] = 0
44	GPIO[20]	ext_sel_reg0[10] = 1
	GPIO[21]	ext_sel_reg0[10] = 0
45	GPIO[22]	ext_sel_reg0[11] = 1
	GPIO[23]	ext_sel_reg0[11] = 0

Table 21: GPIO Mapping to External Interrupts (Continued)

External Interrupt Bit	GPIO Connected	External Select Register Value
46	GPIO[24]	ext_sel_reg0[12] = 1
	GPIO[25]	ext_sel_reg0[12] = 0
47	GPIO[26]	ext_sel_reg0[13] = 1
	GPIO[27]	ext_sel_reg0[13] = 0
48	GPIO[28]	ext_sel_reg0[14] = 1
	GPIO[29]	ext_sel_reg0[14] = 0
49	GPIO[30]	ext_sel_reg0[15] = 1
	GPIO[31]	ext_sel_reg0[15] = 0
50	GPIO[32]	ext_sel_reg0[16] = 1
	GPIO[33]	ext_sel_reg0[16] = 0
51	GPIO[34]	ext_sel_reg0[17] = 1
	GPIO[35]	ext_sel_reg0[17] = 0
52	GPIO[36]	ext_sel_reg0[18] = 1
	GPIO[37]	ext_sel_reg0[18] = 0
53	GPIO[38]	ext_sel_reg0[19] = 1
	GPIO[39]	ext_sel_reg0[19] = 0
54	GPIO[40]	ext_sel_reg0[20] = 1
	GPIO[41]	ext_sel_reg0[20] = 0
55	GPIO[42]	ext_sel_reg0[21] = 1
	GPIO[43]	ext_sel_reg0[21] = 0
56	GPIO[44]	ext_sel_reg0[22] = 1
	GPIO[45]	ext_sel_reg0[22] = 0
57	GPIO[46]	ext_sel_reg0[23] = 1
	GPIO[47]	ext_sel_reg0[23] = 0
58	GPIO[48]	ext_sel_reg0[24] = 1
	GPIO[49]	ext_sel_reg0[24] = 0

2.6 AHB Bus Fabric

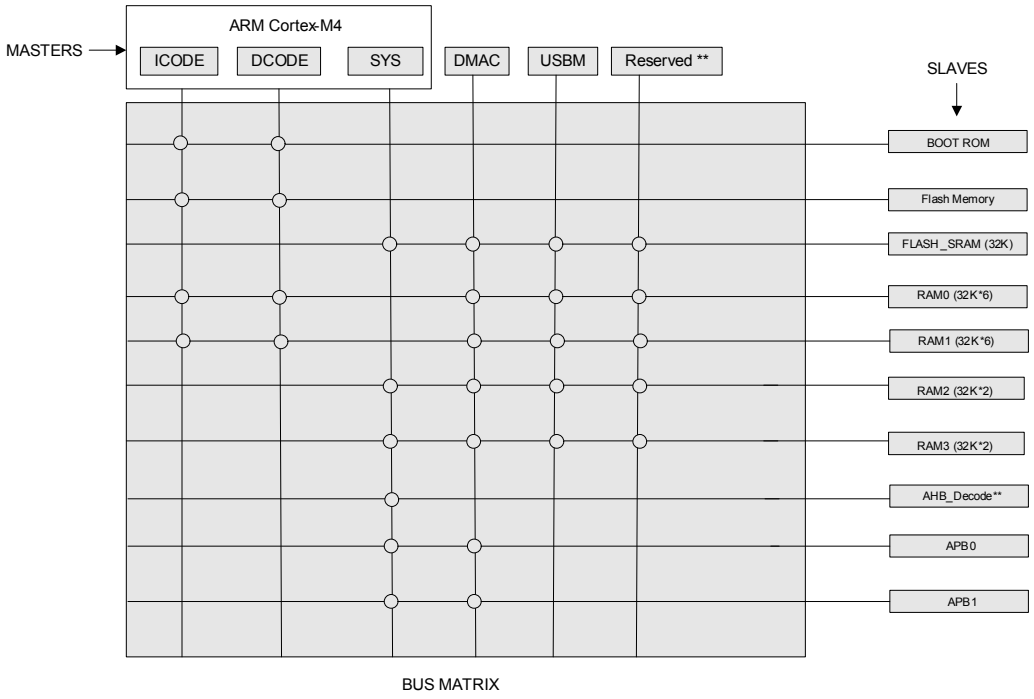
The 88MW300/302 AHB Bus Matrix is a low latency bus matrix which enables parallel access to a number of shared AHB slaves from a number of different AHB masters. The bus matrix routes the control data signals between the masters and slaves based on the address map (see Section 2.4, Memory Map, on page 69). The sparse connectivity feature allows for the configuration of only the necessary master/slave connections, thus providing reduced area and multiplexer delays.

The AHB Bus Fabric connects 6 masters and 9 slaves. The master ports connected to the bus matrix include ICODE, DCODE, and SYSTEM bus from Cortex-M4F, DMA Controller, and USB Controller. The slave ports connected to the bus matrix include the BOOT ROM, Flash memory, RAM0, RAM1, RAM2, RAM3, APB0, APB1 (APB0 and APB1 contain the instances of the APB peripherals in the system) and AHB Decode (decodes addresses to the various AHB peripherals in the system).

APB0 and APB1 are top-level blocks that contain the APB peripherals. The AHB Decode block maps to registers in the DMA Controller, USB Controller, AES-CRC, and Flash Controller blocks.

Figure 8 shows the connection between the various masters and slaves in the system.

Figure 8: Bus Matrix Interconnection



2.7 Register Description

See Appendix A.21.2, System Control Registers, on page 825 for a detailed description of the registers.

3 Power, Reset, and Clock Control

3.1 Overview

The 88MW300/302 power supply, power mode, and on-chip DC-DC converter, clocking, reset, and wake-up signals are managed by the Power Management Unit (PMU), which is in the Always ON (AON) power domain.

3.2 Power

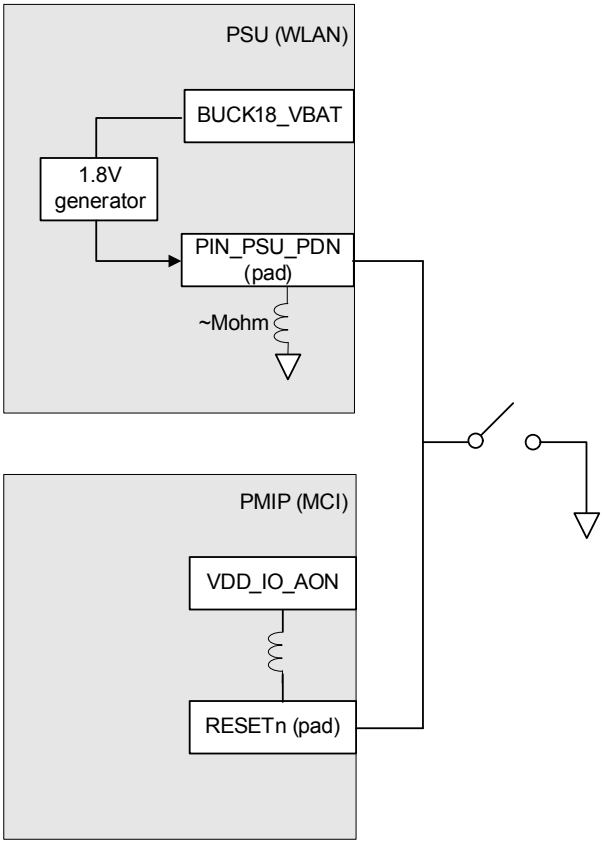
3.2.1 Power Supplies

3.2.1.1 Power Supply Blocks

Figure 9 shows the 2 power supply blocks and internal signals:

- PSU for WLAN
- PMIP for Microcontroller (MCI)

Figure 9: Power Supply Blocks



3.2.1.2 Power-up Requirements

See Table 16, Power and Ground, on page 64 for the power pins.

Figure 10 shows the device power supplies.

Figure 11 shows the device power-up sequence.

- External VBAT_IN
- Internal AVDD18/VDD11 from on-chip LDOs and BUCK

The following requirements must be met for correct power-up:

- External 3.3V is used for VDDIO_0:3, VDDIO_AON, and VBAT_IN, as needed by the platform
- VBAT_IN is used as input to LDO18, which will further input into LDO11
- BUCK18_VBAT_IN is used as input to BUCK18, which will further input into LDO11

Figure 10: Power Option

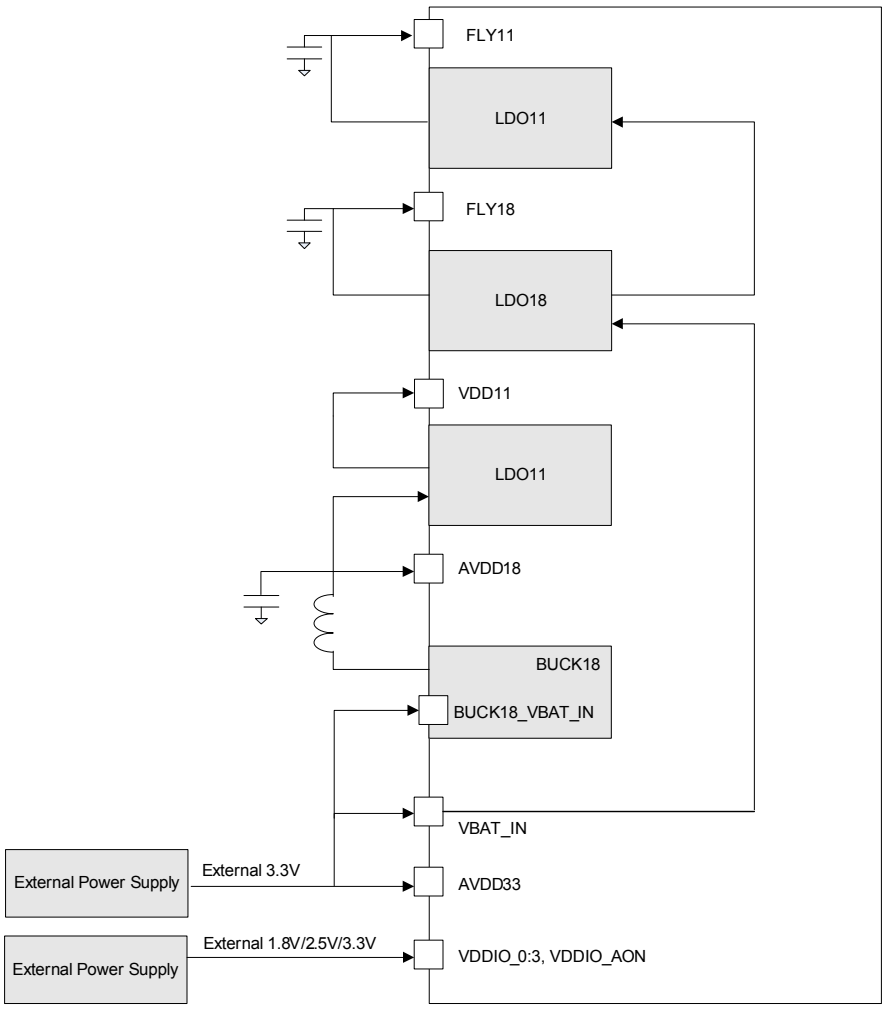
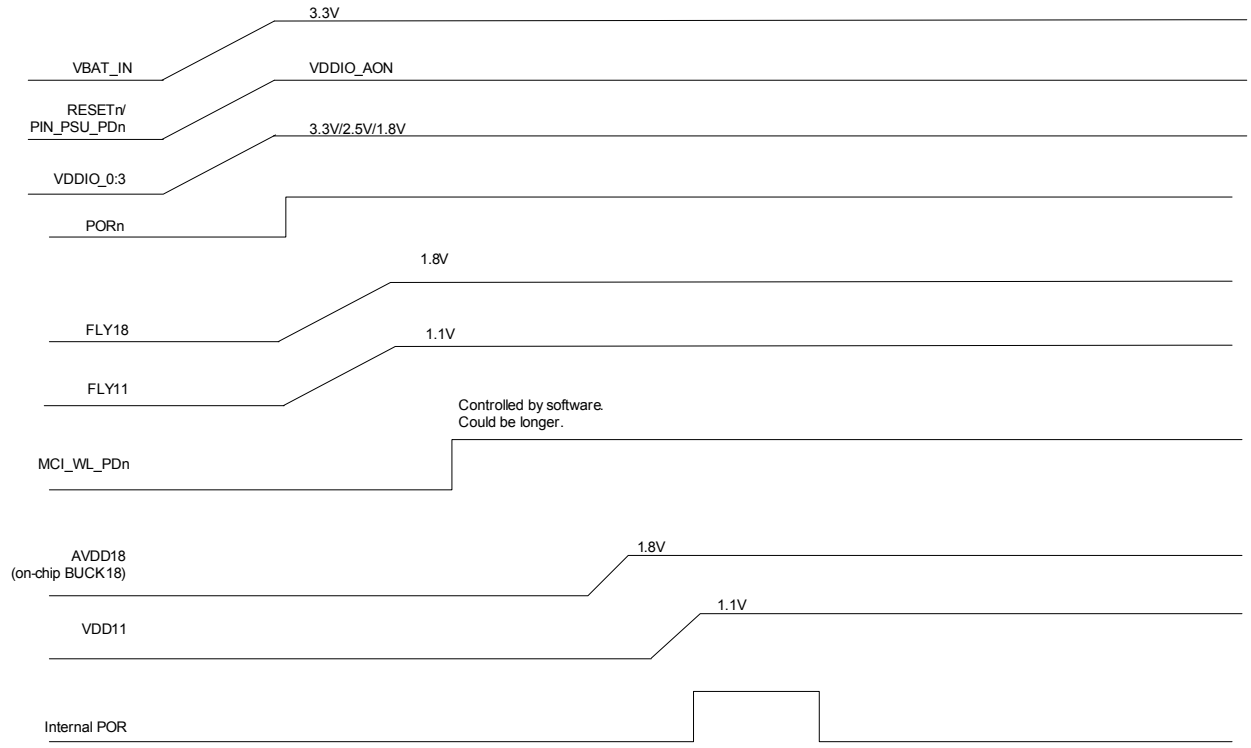


Figure 11: Power-Up Sequence



3.2.2 Power Domains

The 88MW300/302 provides several independent power domains, including:

- VDD_AON – PMU, RTC, low-power comparator, and 4K memory brown-out detection logic are in the VDD_AON power domain. They are operational in all power modes.
- VDD_MEM – 192 KB of SRAM and Flash Controller sits on this power domain. It is on in PM0, PM1, PM2, and PM3.
- VDD_MCU – Cortex-M4F, RC32M digital and ADC/DAC/ACOMP digital control logic and the remainder of the 320 KB SRAM (Table 22), all AHB and APB peripherals and PINMUX are in this power domain. It is on in PM0, PM1, and PM2 power modes.
- VDDIO_0 domain: GPIO_0 to GPIO_15
- VDDIO_1 domain: GPIO_16 to GPIO_21
- VDDIO_2 domain: GPIO_28 to GPIO_33
- VDDIO_3 domain: GPIO_34 to GPIO_49, GPIO_27

Table 22: VDD_MCU Address Memory

Flash	CODE	SRAM
0x1F00_0000 to 0x1FFF_FFFF (16 MB)	0x0010_0000 to 0x0015_7FFF (384 KB)	0x2000_0000 to 0x2001_FFFF (64 KB)

3.2.3 I/O Power Configuration

The 88MW300/302 has configurable I/O domains. All domains support independent power-off functionality.

- VDDIO_0 domain: GPIO_0 to GPIO_15
- VDDIO_1 domain: GPIO_16 to GPIO_21
- VDDIO_2 domain: GPIO_28 to GPIO_33
- VDDIO_3 domain: GPIO_34 to GPIO_49, GPIO_27
- VDDIO_AON domain: GPIO_22 to GPIO_26, RESETn

Configuration is controlled by the PMU register, IO_PAD_PWR_CFG. See Appendix A.20, PMU Address Block, on page 769 for a detailed description of the registers.

Table 23 shows the configurable options.

Table 23: I/O Power Configuration

I/O Power Domain	Corresponding GPIOs	Register Field	Value	Description
VDDIO_0	GPIO_0 to GPIO_15	PMU.IO_PAD_PWR_CFG.GPIO0_V18	0 (default)	1.8V/2.5V/3.3V
VDDIO_1	GPIO_16 to GPIO_21	PMU.IO_PAD_PWR_CFG.GPIO1_V18	0 (default)	1.8V/2.5V/3.3V
VDDIO_2	GPIO_28 to GPIO_33	PMU.IO_PAD_PWR_CFG.GPIO2_V18	0 (default)	1.8V/2.5V/3.3V
VDDIO_3	GPIO_34 to GPIO_49, GPIO_27	PMU.IO_PAD_PWR_CFG.GPIO3_V18	0 (default)	1.8V/2.5V/3.3V
VDDIO_AON	GPIO_22 to GPIO_26, RESETn	PMU.IO_PAD_PWR_CFG.GPIO_AON_V18	0 (default)	1.8V/2.5V/3.3V

Note: The real power supply to VDD_IOx_y power pin should match the corresponding configuration of the IO_PAD_PWR_CFG register.

After the 88MW300/302 is powered on, all I/O domains are turned on (controlled by IO_PAD_PWR_CFG.GPIO_[domain]_LOW_VDDB). The default I/O is applied to 3.3V (controlled by IO_PAD_PWR_CFG.GPIO_[domain]_V18). The default pad regulator works in normal mode (controlled by IO_PAD_PWR_CFG.GPIO_[domain]_PDB).

Firmware could configure the power voltage of the corresponding I/O domain at any time to apply to different devices. Also, firmware could configure the corresponding domain, where the pad regulator is located in power-down mode to save power consumption.

Firmware must power on the I/O domain first before the I/O data transfer starts. The pad value is tri-stated before the I/O is powered on. In Sleep mode, or for power consumption savings, firmware must also power off the I/O domains before entering Sleep mode. Otherwise, the pad value is unknown.

Note: No matter the I/O function, the input level of I/O pins should not exceed the corresponding I/O domain power supply.

3.2.4 AON Domain

The PMU, RTC, ultra low-power comparator, brown-out detection logic, and 4K_MEM are in the AON domain. These modules can be powered in all power modes.

The PMU module manages the different power modes, power mode transition, and wake-up from low-power mode. The 4K_MEM is 4 KB-size SRAM and located from 0x480C_0000 memory space. Even in the lowest power mode, the content of 4K_MEM can be retained so it can be used to store critical application data. A 32-bit RTC is included in the AON domain.

See Section 9, Real Time Clock (RTC), on page 183 for a detailed description.

3.2.4.1 Ultra Low-Power Comparator

The low-power comparator can operate in differential mode and in single-ended mode.

In differential mode, COMP_IN_N (muxed with GPIO_24) and COMP_IN_P (muxed with GPIO_23) are compared to each other by the comparator. In single-ended mode, the comparator compares reference voltage GPIO_23 input. The reference voltage is controlled by the PMIP_CMP_CTRL.COMP_REF_SEL bit registers. When using the comparator in differential mode, both GPIO_24 and GPIO_23 have to be in hi-Z state. For single-ended mode, GPIO_23 must be in hi-Z state. It can generate an interrupt or wake-up, which is controlled by the PMIP_CMP_CTRL register.

3.2.4.2 Brown-out Detection

The 88MW300/302 contains VBAT brown-out detection circuits. It can generate a reset when a voltage supply is below a pre-set threshold. The Cortex-M4F core and all peripherals except the PMU and low-power comparator are reset by this event. The brown-out reset event is disabled by default. To enable this reset event, first program the PMIP_BRNDET_VBAT register to set the brown-out threshold and enable brown-out circuits, then set the brown-out PMU reset enable register, PMIP_BRN_CFG.

See Section 22.7.2, Brown-Out Detection (BOD) Specifications, on page 329 for electrical specifications.

3.3 Power Modes

3.3.1 System Power Modes

To optimize power consumption for different system configurations, the device can be set to different low-power modes by the PMU. Table 24 shows the system power modes and associated MCI and WLAN subsystem power states.

Table 24: System Power Modes

Power Mode	Description	MCI Power State	WLAN Power State
MW_PM00	PM0+WPM0	active	active
MW_PM01	PM0+WPM1	active	deep-sleep
MW_PM02	PM0+WPM2	active	WLAN shut-down
MW_PM10	PM1+WPM0	idle	active
MW_PM11	PM1+WPM1	idle	deep-sleep
MW_PM12	PM1+WPM2	idle	WLAN shut-down
MW_PM20	PM2+WPM0	standby	active
MW_PM21	PM2+WPM1	standby	deep-sleep
MW_PM22	PM2+WPM2	standby	WLAN shut-down
MW_PM30	PM3+WPM0	sleep	active
MW_PM31	PM3+WPM1	sleep	deep-sleep
MW_PM32	PM3+WPM2	sleep	WLAN shut-down
MW_PM40	PM4+WPM0	deep-sleep	active
MW_PM41	PM4+WPM1	deep-sleep	deep-sleep
MW_PM42	PM4+WPM2	deep-sleep	WLAN shut-down
MW_PM52	PM5+WPM2	shut-down	WLAN shut-down

3.3.2 MCI Subsystem Power Modes

Table 25 shows the MCI subsystem power modes.

Table 25: MCI Subsystem Power Modes

Subsystem	PM0	PM1	PM2	PM3	PM4
Cortex-M4F	C0	C1	C2	C3	C3
SRAM	M0	M0	M2	M2[1]	M3
Flash	active standby	active standby	power-down	off	off
RTC	on	on	on	on	on
Peripherals	on[2]	on2	state retentive	off	off
XTAL	on/off	on/off	on/off	off	off
SFLL	on/off	on/off	off	off	off
AUPLL	on/off	on/off	off	off	off

1. All memories are in state retention mode in PM2 power state and 192 KB out of 512 KB of SRAM, plus 32 KB Flash memory will be in state retention mode in PM3 power mode. Table 26 shows the memory addresses in state retention mode in PM3.
2. When in PM0 and PM1 modes, functional clocks for peripherals can be shut off by programming the PMU.PERI_CLK_EN registers.

Table 26: Address of Memories Available in State Retention Mode

Flash Memory	CODE	SRAM
0x2002_0000 to 0x2002_7FFF (32 KB)	0x0010_0000 to 0x0012_7FFF (160 KB)	0x20000000 to 0x20007FFF (32 KB)

3.3.2.1 PM0 – Active Mode

The 88MW300/302 enters PM0 state upon the completion of the POR reset sequence and a wake-up from PM2, PM3, and PM4 states. When in PM0 state, all internal power domains and external power supplies may be fully powered and functional. Each peripheral function clocks can be gated off by programming the PMU clock enable registers.

3.3.2.2 PM1 – Idle Mode

In PM1 mode, the clock to the Cortex-M4F core is stopped. All other on-chip functions may continue operation in idle mode. The core can be quickly reactivated and resume execution by a generated interrupt.

Entering into the PM1 mode is performed by the Cortex-M4F core executing the WFI instruction with clearing the Cortex-M4F System Control Register SLEEPDEEP bit. The Cortex-M4F NVIC continues monitoring interrupts and wakes up the Cortex-M4F when an interrupt is detected.

3.3.2.3 PM2 – Standby Mode

The PM2 state offers lower power consumption by placing the Cortex-M4F core, most of the 88MW300/302 peripherals, and SRAM arrays in a low-power mode. In this mode, the core state and registers, peripheral registers, and internal SRAM values are preserved, and the state of the I/Os is kept. Flash memory is in PWDN mode. Marvell recommends disabling BOD before entering PM2.

Entering into PM2 state is performed by writing PMU PWR_MODE registers to the PM2 state (01). The sequence for entering PM2 state is as follows:

1. Set Cortex-M4F System Control Register SLEEPDEEP bit.
2. Program PMU_CLK_SRC register to switch source clock to RC32M if system source clock is not RC32M.
3. Program PMU_SFLL_CTRL0 to disable PLLs and WLAN_CTRL registers to disable all 3 reference clock requests.
4. Turn I/O domains off.
5. Set PMU_PWR_MODE registers to PM2 state.
6. Execute WFI instruction.

Exiting the PM2 state occurs when the PMU detects any wake-up that is enabled before entering the PM2 state. The sequence is as follows:

1. PMU detects a wake-up event.
2. Cortex-M4F wakes up through interrupt.
3. If corresponding bit in NVIC is set, the ISR subroutine is executed.
4. Continue the instruction that follows WFI.

3.3.2.4 PM3 – Sleep Mode

In PM3 mode, all the power supplies except the power supplies for AON domain and Memory modules are turned off. RTC can be still running with a 32 kHz clock and 4K_MEM in AON domain is in retention mode. 192 KB SRAM is in Retention mode (see Table 26, Address of Memories Available in State Retention Mode, on page 86).

During this mode, all functional clocks except the RTC clock are gated off.

Wake-up from PM3 mode occurs by any of the wake-up sources in Table 30, Low-Power Mode Wake-Up Sources, on page 91.

Marvell recommends disabling the BOD and ULP Comparator before entering PM3.

The system can be programmed to enter PM3 mode by programming the PMU.PWR_MODE.pwr_mode register to 2'b10.The sequence for entering PM3 state is as follows:

1. Set Cortex-M4F System Control Register SLEEPDEEP bit.
2. Program PMU_CLK_SRC register to switch source clock to RC32M if system source clock is not RC32M.
3. Program PMU_SFLL_CTRL0 to disable PLLs and WLAN_CTRL registers to disable all 3 reference clock requests.
4. Power off I/O domain (IO_PAD_PWR_CFG.POR_LVL_[domain]_LOW_VDDB_CORE) except AON domain to save power consumption and prevent IO pads entering unknown state, and power off the regulators of all IO domains (IO_PAD_PWR_CFG.VDD_[domain]_REG_PDB_CORE) to save power consumption as well.

   Note: Exiting the PM3 state occurs when the PMU detects any wake-ups that are enabled before entering the PM3 state. The sequence is as follows:

   a) PMU detects a wake-up event.
   b) Cortex-M4F wakes up, and a Boot from SRAM occurs.
5. Software enables RC32 or XTAL32K if it is not enabled.

6. Set PMU_PWR_MODE registers to PM3 state.
7. Execute WFI instruction.

Exiting the PM3 state occurs when the PMU detects any wake-ups that are enabled before entering the PM3 state. The sequence is as follows:

1. PMU detects wake-up event.
2. Cortex-M4F wakes up and resets all except AON domain.

3.3.2.5 PM4 – Shutoff Mode

In PM4 mode, the power is shut off to the entire chip with the exception of the AON domain. RTC can still be running with a 32 kHz clock and 4K_MEM in retention mode.

Wake-up from PM4 mode occurs by any of the wake-up sources in Table 30, Low-Power Mode Wake-Up Sources, on page 91.

Marvell suggests disabling the BOD and ULP Comparator before entering PM4.

Entering into the PM4 state is performed by writing the power mode registers to the PM4 state (2'b11). The sequence to enter the PM4 state is as follows:

1. Set Cortex-M4F System Control Register DEEPSLEEP bit.
2. Enable RC32K or XTAL32K clock if not already enabled.
3. Program PMU_CLK_SRC register to switch source clock to RC32M if system source clock is not RC32M.
4. Program PMU_SFLL_CTRL to disable PLLs and WLAN_CTRL registers to disable all 3 reference clock requests.
5. Power off all I/O domains (IO_PAD_PWR_CFG.POR_LVL_[domain]_LOW_VDDB_CORE) except AON domain to save power and prevent IO pads entering unknown state, and power off the regulators of all I/O domains (IO_PAD_PWR_CFG.VDD_[domain]_REG_PDB_CORE) to save power as well.
6. Set PMUPWR_MODE registers to the PM4 state.
7. Execute WFI instruction.

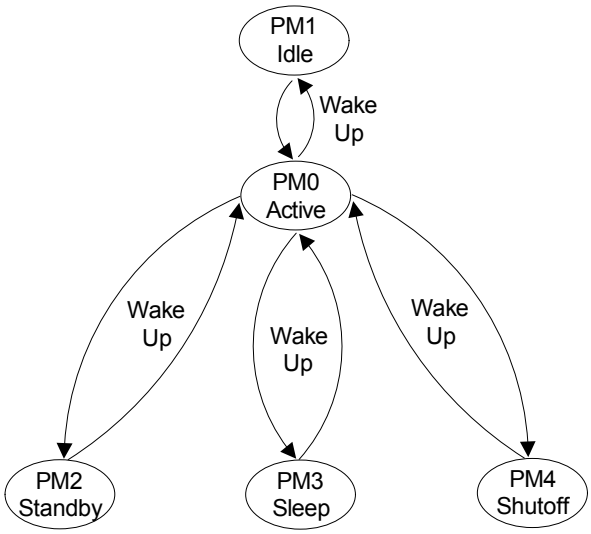
Exiting the PM4 state occurs when the PMU detects any wake-ups that are enabled before entering PM4 state. The sequence is as follows:

1. PMU detects a wake-up event.
2. Cortex-M4F wakes up and resets all except AON domain.

3.3.2.6 MCI Power Mode Transitions

Figure 12 shows the state machine for the MCI power mode state transitions.

Figure 12: Power Mode Transitions



3.3.3 WLAN Power States

3.3.3.1 WPM0 – Active

The WLAN enters WPM0 state when the WLAN feature is enabled from the WPM2 state or a wake-up from the WPM1 state. When in the WPM0 state, all internal power domains and external power supplies are fully powered and functional.

3.3.3.2 WPM1 – Deep Sleep

In WPM1 state, all power supplies, except the power supplies to the WLAN AON domains and memory modules, are turned off. Exiting WPM1 occurs when the MCI host wake-up interrupt or an internal WLAN beacon timer interrupt is received.

3.3.3.3 WPM2 – Shut Down

In WPM2 state, all power supplies are tuned off. Entering the WPM2 state is performed by writing the power mode register (PMU register from MCI). Exiting the WPM2 state occurs when this bit is de-asserted.

3.3.4 Core and SRAM Power States

In different power modes, the 88MW300/302 core and memory may be in different power states.

3.3.4.1 Cortex-M4F Core Power States

Table 27: Cortex-M4F Core Power States

Cortex-M4F State	Definition	Comments
RUN (C0)	HCLK on, FCLK on	--
IDLE (C1)	HCLK off, FCLK on	--
STDBY (C2)	HCLK off, FCLk off	State retentive mode
OFF (C3)	Power is removed	--

3.3.4.2 Memory Power States

Table 28: SRAM Memory Power States

SRAM State	Definition	Comments
RUN (M0)	CLK on	--
STDBY (M2)	PWDN	State retentive mode
OFF (M3)	Power is removed	--

Table 29: Flash Memory Power Modes

Flash State	Definition
Active	Ready for access
STDBY	Chip select is de-asserted
PWDN	Leakage reduction mode
OFF	Power is removed

3.4 Reset/Wake-up Sources

3.4.1 Wake-up from PM1 Mode

Any enabled interrupt can wake up the core from Idle mode.

See Section 2.5, External Interrupts, on page 74 for a detailed list of all interrupts.

3.4.2 Wake-up from PM2/3/4 Modes

The 88MW300/302 supports internal and external wake-up sources when exiting from Standby or Sleep modes. Table 30 shows the sources.

The PMU supports both active-high and active-low wake-up from WAKE_UP0 and WAKE_UP1. It is programmed with the PMU WAKEUP_EDGE_DETECT registers.

See Appendix A.20, PMU Address Block, on page 769.

Table 30: Low-Power Mode Wake-Up Sources

Source	PM2	PM3	PM4
RTC	yes	yes	yes
WAKE_UP0	yes	yes	yes
WAKE_UP1	yes	yes	yes
ULP_COMP	yes	yes	yes

3.4.3 Reset Controller

The 88MW300/302 has the following reset sources:

- Power-On Reset (POR) – PMU detects power supply ramping from 0 volt to VBAT voltage level. Entire SoC is reset in this event.
- 1 dedicated PIN reset – PMU detects a PIN reset. Entire SoC is reset in this event.
- Low-power mode exit reset – When PMU detects wake-up while in PM3 and PM4 modes, it resets Cortex-M4F core and all peripherals except for AON domain.
- Warm reset – Cortex-M4F core and all peripherals are reset except PMU, low-power comparator, and core debug logic:
  - Triggered when LOCKUP
  - PMIP brown-out reset – generated when voltage level of VBAT is detected as dropping below a pre-set threshold level (brown-out source must be enabled for it to function)
  - Cortex-M4F – triggered soft reset
  - WDT timeout

3.5 Clock Controller

3.5.1 Overview

The 88MW300/302 clock controller unit controls system source clock, clock frequencies for Cortex-M4F, AHB and APB bus clocks, and all peripheral function clocks. There are 4 different clock sources and 3 PLLs:

- MAINXTAL – External crystal oscillator (38.4 MHz)
- XTAL32K – External crystal oscillator 32.768 kHz
- RC32M – Internal RC32M
- RC32K – Internal RC32K
- SFLL – System PLL
- AUPLL – Audio PLL
- USB PLL

There are 2 external clock sources:

- 38.4 MHz crystal oscillator
- 32.768 kHz crystal oscillator

There are 2 internal RCs:

- 32 MHz (approximate) clock
- 32 kHz clock

There are 3 PLLs:

- SFLL
- AUPLL
- USB PLL

The SFLL generates a maximum of 200 MHz clock to support the Cortex-M4F core, AHB bus clocks, and most of the peripherals. Programmable dividers divide the 200 MHz clock to support all peripheral function clocks, as well as APB bus clocks. The AUPLL is used to generate clocks for audio and GAU (ADC/DAC/ACOMP). The USB PLL generates 480 MHz clock for the USB module.

Users can switch the clock source to internal the RC32M clock or the 38.4 MHz oscillator by programming the PMU clock source select registers for low performance applications. Dynamic source clock change is supported between RC32M and MAINXTAL or RC32M and SFLL. Switching source clocks between MAINXTAL and SFLL is not allowed, and vice versa. Users should always switch to RC32M first. All peripheral function clocks can be shut off by the peripheral clock enable registers when not used for the application.

Table 31 shows the clock sources. Table 32 shows the clock frequencies.

See Section 22.6.1, RC32K Specifications, on page 327 for electrical specifications.

3.5.2 Clock Sources

Table 31: Clock Sources

Clock Name	Frequency (MHz)	Description
MAINXTAL	38.4	External crystal
XTAL32K	32.768 k	External crystal
RC32M	32 (±50%)	On-chip RC OSC, before calibration
	32 (±1%)	On-chip RC OSC, after calibration
RC32K	32 k (±50%)	On-chip RC OSC, before calibration
	32 k (±2%)	On-chip RC OSC, after calibration
SFLL	200	Output frequency is programmable
AUPLL	Audio bit clock	Output frequency is programmable
USB PLL	480	--

Figure 13: High-Level Clocking Diagram

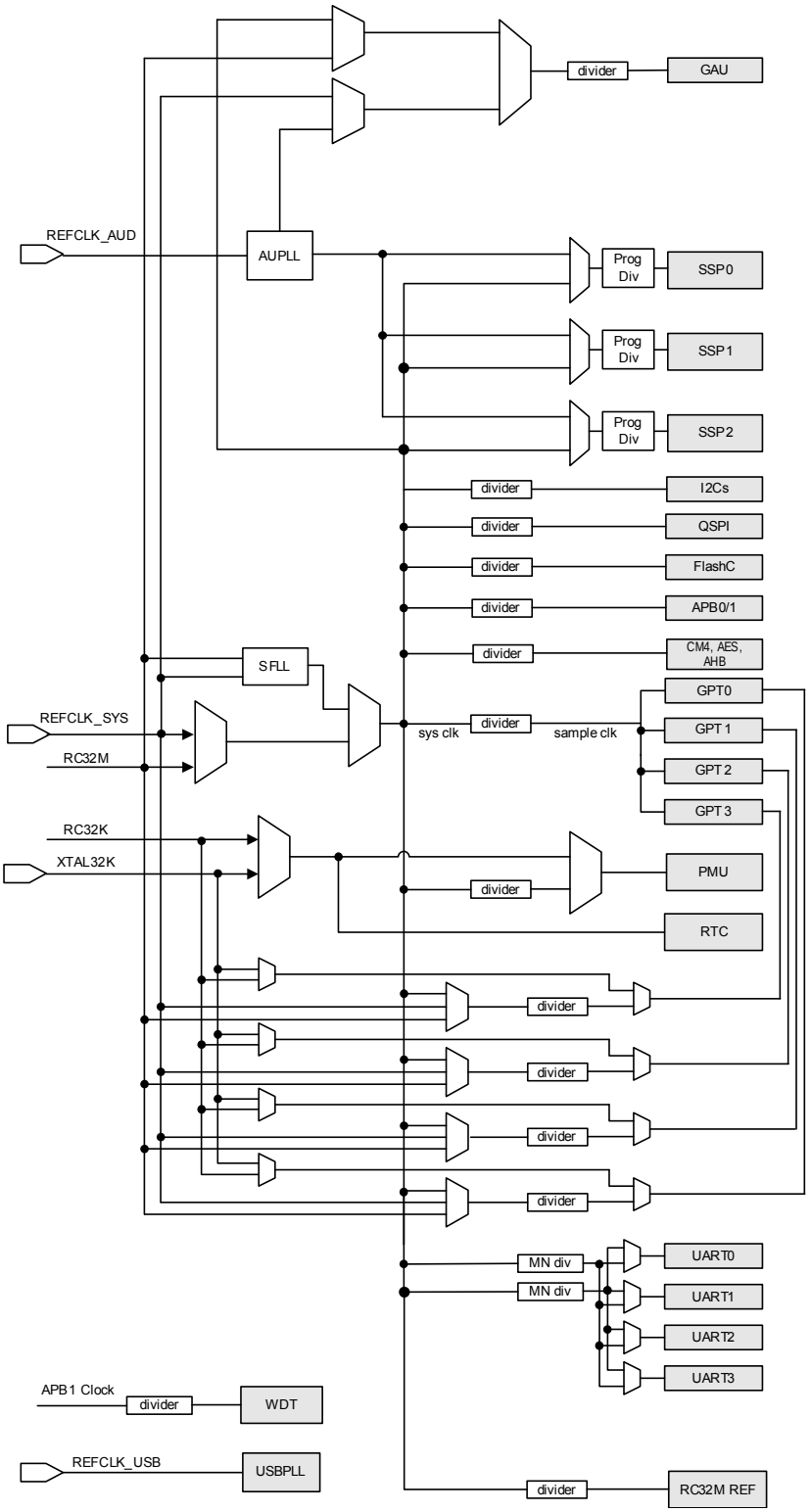


Table 32: Clock Frequency

Module	Maximum Frequency (MHz)	Clock Source	Programmable Divider
Cortex-M4F HCLK	200	SFLL	yes
Cortex-M4F FCLK	200	SFLL	yes
AHB BUS	200	SFLL	yes
APB1 BUS	50	SFLL	yes
APB0 BUS	50	SFLL	yes
Memory	200	SFLL	yes
AES/CRC	200	SFLL	yes
USB	60	USB PLL	no
SSP	25	SFLL/AUPLL	yes
SSP Audio	24.587	AUPLL	yes
UART	58.9	SFLL	yes
GPT	50	SFLL	yes
RTC	32 kHz	OSC / RC32K	no
QSPI	50	SFLL	yes
I2C	100	SFLL	yes
GAU	programmable (up to 67 MHz)	AUPLL	yes

3.5.3 SFLL

SFLL is the main source clock for the fast system clock. The output frequency can be programmed by the PMU SFLL_CTRL0 and SFLL_CTRL1 registers.

*SFLL output frequency = (reference clock frequency / REFDIV) * 2 * FBDIV / POSTDIV*

3.5.4 Cortex-M4F Core Clock and Bus Clock

The 88MW300/302 PMU clock control unit generates clocks for Coretex-M4 core, as well as AHB and APB bus clocks. All clocks are always from the same clock source. The core (HCLK, FCLK, and AHB) bus clocks are always running at the same frequency. The APB bus clock supports 1:1, 2:1, 4:1, and 8:1 divider ratios.

To select the source clock from RC32M, SFLL, and MAINXTAL program the PMU.CLK_SRC registers. Program the PMU.MCU_CORE_CLK_DIV register to divide the frequency from the source clock.

The APB bus clock divider ratio is controlled by the PMU.PERI1_CLK_DIV register.

Table 33 and Table 34 show the APB clock divider ratios for different values of the PMU.PER1_CLK_DIV register bits.

Table 33: APB0 Bus Clock Divider Ratio

PER1_CLK_DIV[17:16]	APB0 CLK Divider Ratio
00	1:1
01	2:1
10	4:1
11	8:1

Table 34: APB1 Bus Clock Divider Ratio

PER1_CLK_DIV[19:17]	APB1 CLK Divider Ratio
00	1:1
01	2:1
10	4:1
11	8:1

3.5.5 UART Clocks

Select the UART frequency by the PMU.UART_CLK_SEL register. There are 2 programmable fractional dividers that generate the preferred UART clock frequencies. The fractional divisors are changed by programming the nominator and denominator fields in the PMU.UART_FAST_CLK_DIV and PMU.UART_SLOW_CLK_DIV registers based on source clock frequency, which is selected by the PMU.CLK_SRC register to obtain the preferred UART clock frequency.

Table 35 shows the programming.

Table 35: UART Slow and Fast Clock Programming

UART_FAST_CLK_DIV, UART_SLOW_CLK_DIV	
Bit [10:0]	denominator
Bit [23:11]	numerator

The relation between source clock and output clock frequencies is as follows:

$$\frac{\text{numerator}}{\text{denominator}} = \frac{\text{source\_clock}}{\text{output\_clock}}$$

3.5.6 AUPLL for Audio Clock and GAU Clock

The 88MW300/302 has a dedicated AUPLL to generate the audio bit clock for the SSP module and to provide a low jitter GAU main clock. When the SSP works in I^2S mode, the I^2S audio clock can be from the AUPLL only. Program the FRACT field in PMU.AUPLL_CTRL0 and DIV_MCLK and DIV_FBCCLK fields in PMU.AUPLL_CTRL1 to obtain the necessary PLL VCO frequency.

Table 36 shows the VCO frequencies supported.

Table 36: VCO Frequency Select

Master Clock (MHz)	DIV_MCLK Nmclk (integer/hex	Fref (MHz)	DIV_FBCCLK Nfbc (integer/hex)	FRACT	Nfbc (effective)	Fvco (MHz)
38.4	10/0xA	3.84	35/0x23	0x08208	35.28	135.4752
38.4	10/0xA	3.84	38/0x26	0x0AAAA	38.4	147.4560
38.4	9/0x9	4.2667	31/0x1F	0xE54B	31.44	134.14
38.4	6/0x6	6.4	20/0x14	0x0	20	128

Fref = Master clock / DIV_MCLK

Nfbc = DIV_FBCCLK / (1 - FRACT / 4194304)

*Fvco = Fref * Nfbc*

Fout = Fvco / POSTDIV

Table 37: AUPLL Post Divider Programming
NOTE: OCLK = FVCO/POSTDIV

POSTDIV Divider Ratio	DIV_OCLK_ MODULO[2:0]	DIV_OCLK_ PATTERN[1:0]
1	011	00
2	101	00
4	000	00
6	001	01
8	001	00
9	001	10
12	010	01
16	010	00
18	010	10
24	100	01
36	100	10
48	110	01
72	110	10

The AUPLL is disabled after POR. It is important to set the preferred parameters before setting the AUPLL power-up bit in the PMU.AUPLL_CTRL0 register.

3.5.7 GAU Clock

The AUPLL provides the GAU main clock (used for ADC, DAC, and ACOMP) to allow for a low jitter clock for high accuracy analog applications. See Table 36, VCO Frequency Select, on page 97 and Table 37, AUPLL Post Divider Programming, on page 98 for configuration.

Users can turn off this clock by the PMU GAU Clock Gate register.

3.5.8 GPT Clock

The PMU provides the clock source for the GPT. When the CLK_SRC bit in the CLK_CNTL register of the GPT is set to 0, the GPT selects the clock source from the PMU. To enable the GPTx clock, set the GPTx_CLK_EN bit in the PERI_CLK_EN register to 0. The GPT clock can be programmed to select from the following clocks by GPTx_CLK_SEL0 (x = 0, 1, 2, 3) and GPTx_CLK_SEL1 bits in the GPTx_CTRL register of PMU module:

- System clock
- RC32M
- MAINXTAL
- XTAL32K
- RC32K

The clock can be divided if the system clock/RC32M/MAINXTAL is selected. For GPT0, GPT1 and GPT2, the clock can be divided through the GPTx_CLK_DIV bits in the GPTx_CTRL register of PMU module (x = 0, 1, 2). For GPT3, the clock can be divided through GPT3_CLK_DIV_2_0 and GPT3_CLK_DIV_5_3 in the PERI2_CLK_DIV register.

3.5.8.1 GPT Sampling Clock

When GPT is in the input function, the PMU provides the sampling clock to GPT to sample input signals. The sampling clock source is the system clock, and it can be divided by the GPT_SAMPLE_CLK_DIV bits in the PERI2_CLK_DIV register.

3.5.9 Clock Output

XTAL32K/RC32K/RC32M/AUPLL/SFLL can be output through the corresponding GPIO pins.

See Section 6.2, I/O Configuration, on page 143 and Section 6.2.1, PINMUX Alternate Functions, on page 143 for details.

See Section 22.6, Clock Specifications, on page 327 for electrical specifications.

3.6 Register Description

See Appendix A, 88MW300/302 Register Set, on page 351 for a detailed description of the registers.

THIS PAGE INTENTIONALLY LEFT BLANK

4 Boot ROM

4.1 Overview

The 88MW300/302 Boot ROM is located in memory from 0x00 to 0x7FFF.

4.2 Features

4.2.1 Multiple Boot Sources

There are several boot source options based on the boot pin (GPIO_16 and GPIO_27) settings, including:

- QSPI Flash
- UART
- USB DISK
- USB DFU

4.2.2 Secure Boot

Encrypted and/or signed images stored in QSPI Flash may be used for a secure boot. On-board One Time Programmable (OTP) memory is used to store configuration flags, encryption keys, and so on. This allows for the ability to:

- Keep JTAG always disabled
- Boot encrypted image (using AES)
- Boot signed image (using RSA)
- Bypass the boot pins to boot from QSPI Flash

4.3 Boot Source Selection

Boot ROM is automatically activated by applying a reset.

Depending on the input value of the boot pin, either the Flash or a peripheral communication interface is selected as the boot interface.

Table 38 shows the boot pin configuration options. Also see Section 4.12.1, Boot ROM GPIOs, on page 131 for more Boot ROM GPIO information.

Table 38: Boot Pin Configuration

Boot Pin1 Input Level (GPIO_16)	Boot Pin0 Input Level (GPIO_27)	Description
1	1	Load code image to SRAM from QSPI interface Flash
0	0	Load code image to SRAM from UART interface
1	0	Load code image to SRAM/Flash from USB interface

Note: If errors occur when booting from the QSPI interface (GPIO_16 = 1, GPIO_27 = 1), the Boot ROM will automatically switch the boot source to the UART interface regardless of the input level of the boot pins.

4.4 OTP Content

Table 39 shows the OTP content.

Table 39: OTP Content

Name	Purpose	Default Value	Size
security_flag	0 = JTAG enabled after boot and UART/USB slave boot without password 1 = JTAG never enabled after boot and UART/USB slave boot with password	0	1 bit
non_flash_boot	0 = UART/USB boot enabled 1 = UART/USB boot disabled	0	1 bit
encrypted_boot	0 = encrypted boot off 1 = encrypted boot on	0	1 bit
signed_boot	0 = signed boot off 1 = signed boot on	0	1 bit
aes_key / mainPassword	AES CCM* key / UART/USB slave boot password	All 0s	32 bytes
rsa_hash	SHA256 hash (of OEM's RSA public key)	All 0s	32 bytes

4.5 Data Format in AON Memory

The 88MW300/302 includes 4 KB (kilobyte) Always ON (AON) memory, which is located in 0x480C0000. This memory module belongs to the VDD_AON domain, so its data can be reserved in all power modes. The first 16 bytes of this memory area is used by Boot ROM.

Table 40 shows the data structure.

Table 40: AON Data Structure

Offset Address	Size	Field Name	Description
0x00 to 0x03	32 bits	pm3EntryAddr	Address of Entry Function for PM3 Mode Wake-up Needs to be written by software before going into PM3 state.
0x04 to 0x07	32 bits	bootMode	Select Boot Mode Written by Boot ROM.
0x08 to 0x0B	32 bits	powerMode	Store Power Mode Status Indicates which low-power mode MCU just exited. Written by Boot ROM.
0x0C to 0x0F	32 bits	errorCode	Boot ROM Error Code Storage Written by Boot ROM.

Table 41 shows the bootMode sub-field.

Table 41: Sub-Field in bootMode

Bits	Sub-Field Name	Description
31:2	n/a	Reserved
1:0	bootMode	Boot Mode 0x00 = boot from UART 0x01 = reserved 0x10 = boot from USB 0x11 = boot from QSPI Flash

Table 42 shows the powerMode sub-field.

Table 42: Sub-Field in powerMode

Bits	Sub-Field Name	Description
31:2	n/a	Reserved
1:0	powerMode	Power Mode

Table 43 shows the errorCode sub-field.

Table 43: Sub-Field in errorCode

Bits	Sub-Field Name	Description
31:20	n/a	Reserved
19	aesTimeOut	AES Timeout Error 0 = no error 1 = AES timeout error
18	loadImageFail	Load Image Fail 0 = no error 1 = Flash load image fail
17	secHeaderFail	Section Header Fail 0 = no error 1 = load section header fail
16	magicCodeErr	Magic Code Error 0 = no error 1 = magic code in bootInfo header check fail
15	MicError	MIC Error 0 = no error 1 = decryption failure (MIC mismatch at end of decryption)
14	imageLenErr	Image Length Error 0 = no error 1 = encrypted image not present in Flash (length field all 0s)
13	aesKeyCrcErr	AES Key CRC Error 0 = no error 1 = CRC mismatch for 256-bit AES key in OTP
12	aesKeyflag	AES Key Flag 0 = AES key present in OTP 1 = AES key not present in OTP
11:10	n/a	Reserved
9	sigFail	Signature Fail 0 = no error 1 = signature verification fail
8	digSigFlag	Digital Signature Flag 0 = no error 1 = digital signature not present in Flash (all 0s)
7	pubKeyErr	Public Key Error 0 = no error 1 = hash public key in Flash does not match key stored in OTP
6	pubKeyFlag	Public Key Flag 0 = no error 1 = public key not present in Flash (all 0s)

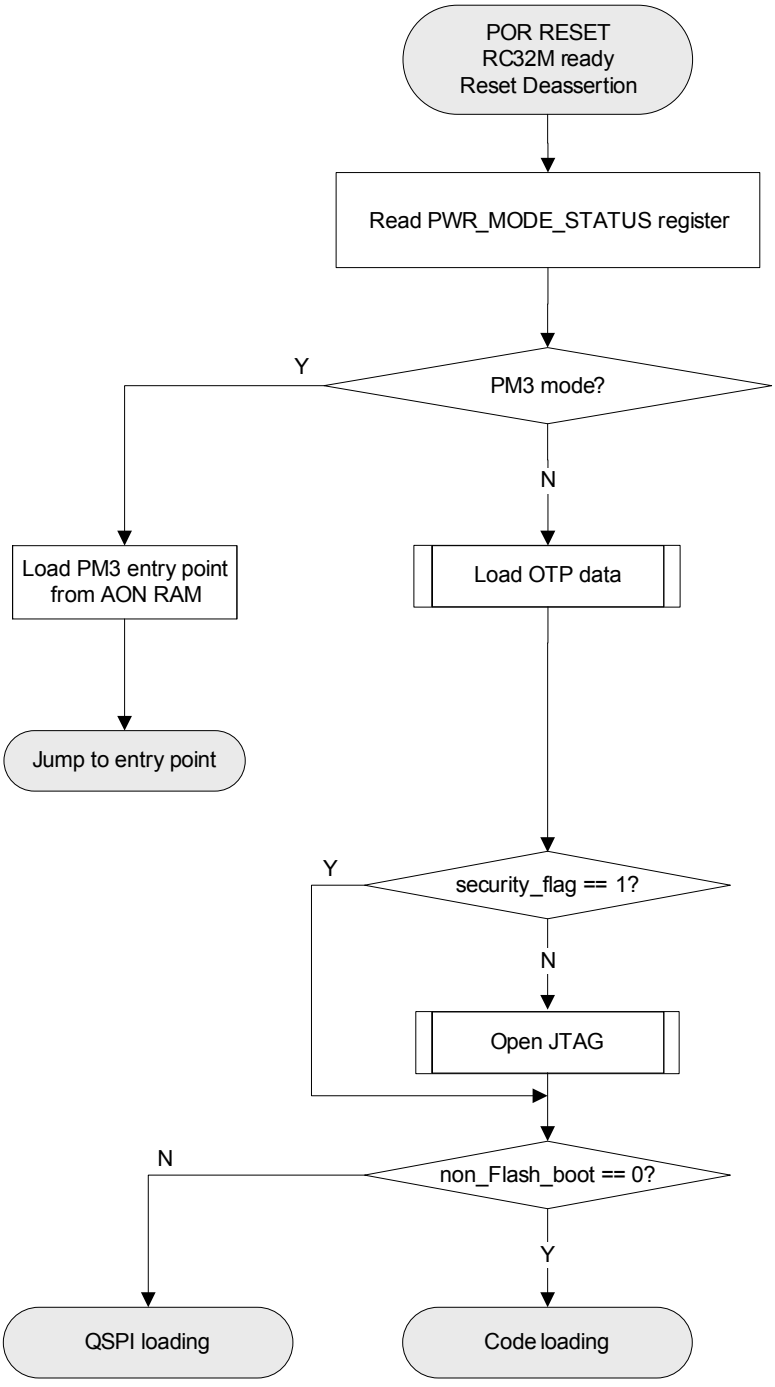
Table 43: Sub-Field in errorCode (Continued)

Bits	Sub-Field Name	Description
5	pubKeyCrcErr	Public Key CRC Error 0 = no error 1 = CRC mismatch for hash of RSA public key in OTP
4	pubKeyHashFlag	Public Key Hash Flag 0 = RSA public key hash present in OTP 1 = RSA public key hash not present in OTP
3	aesKeyErr	AES Key Error 0 = no error 1 = length of AES key error
2	pubKeyHashErr	Public Key Hash Error 0 = no error 1 = length of RSA public key hash error
1	flashAccessFail	Flash Access Fail 0 = no error 1 = Flash not accessible
0	otpAccessFail	OTP Access Fail 0 = no error 1 = OTP not accessible

4.6 Boot ROM Flow Charts

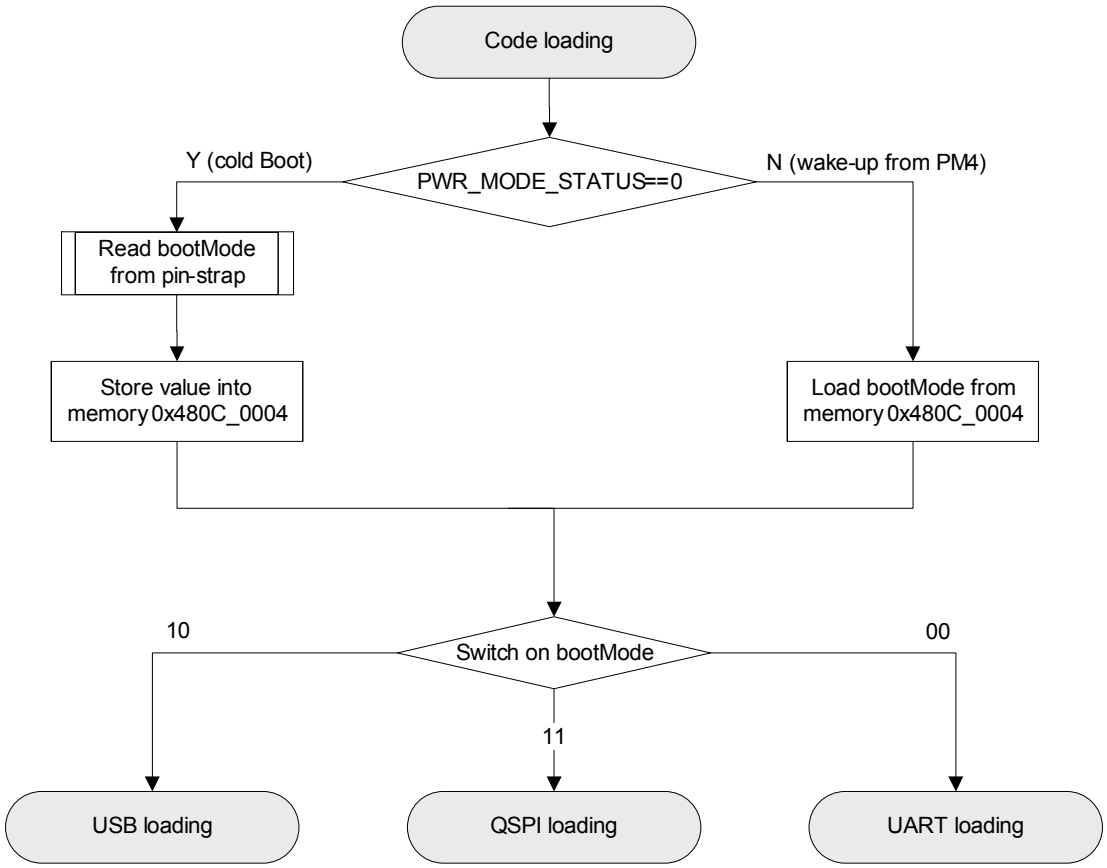
4.6.1 POR Reset

Figure 14: Boot ROM Flow, POR Reset



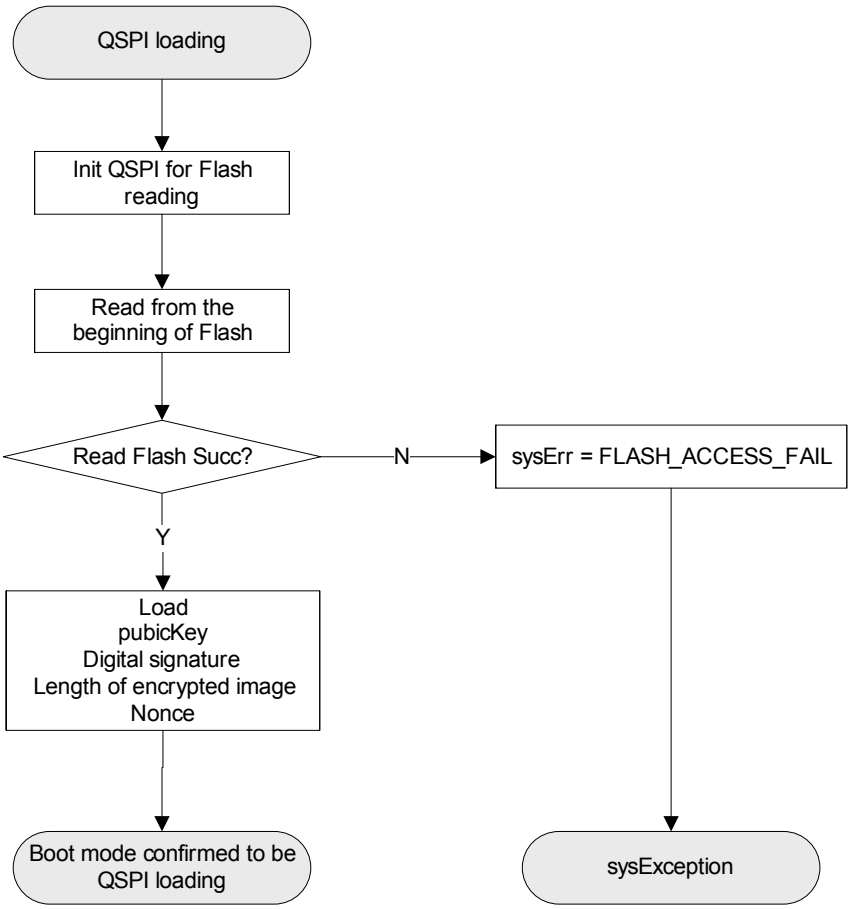
4.6.2 Code Loading

Figure 15: Boot ROM Flow, Code Loading



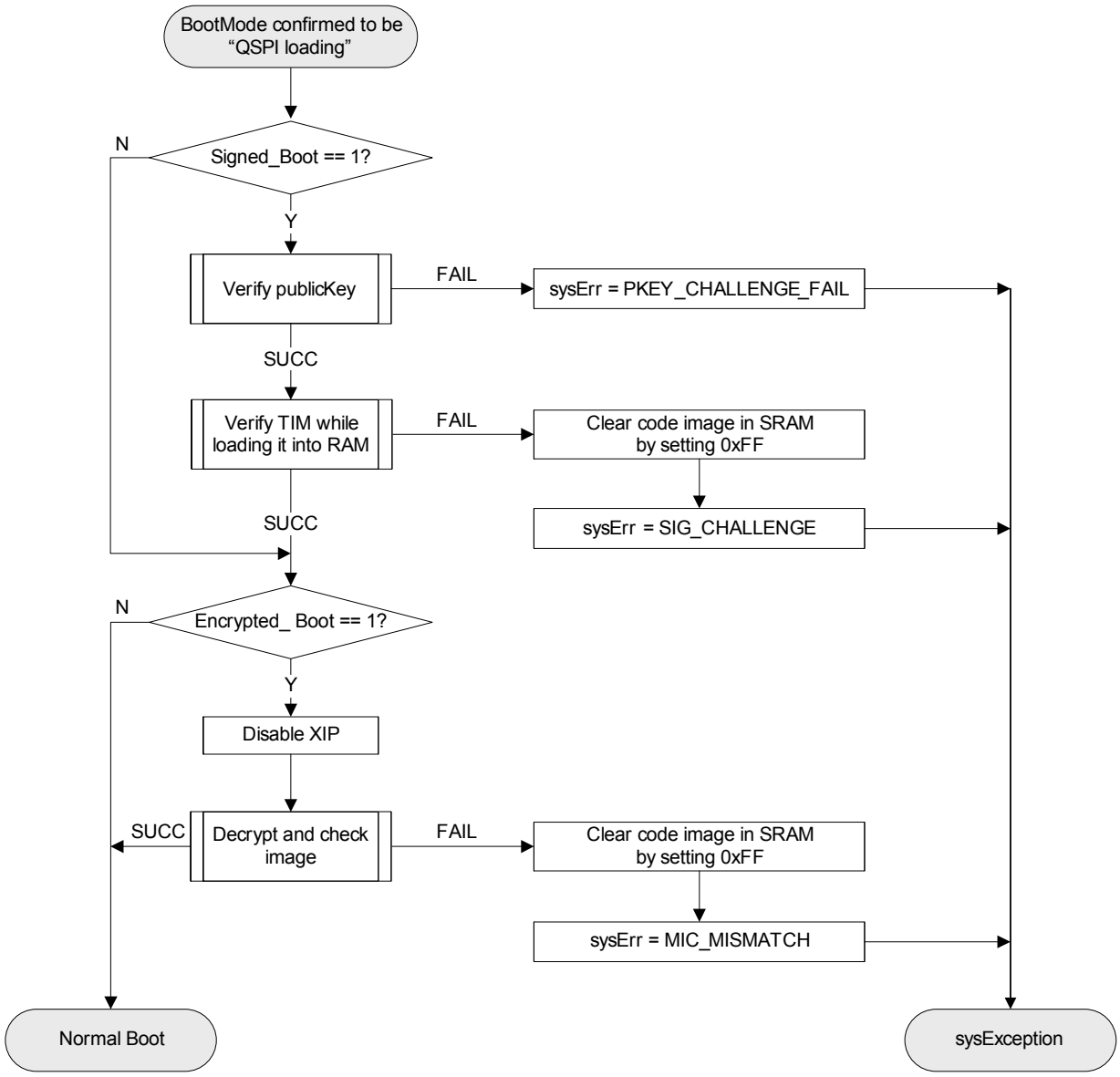
4.6.3 QSPI Loading

Figure 16: Boot ROM Flow, QSPI Loading



4.6.4 Boot Mode Confirmed to be QSPI Loading

Figure 17: Boot ROM Flow, Boot Mode Confirmed to be QSPI Loading[1]



1. TIM = BootInfo + SecHeader + CodeImage

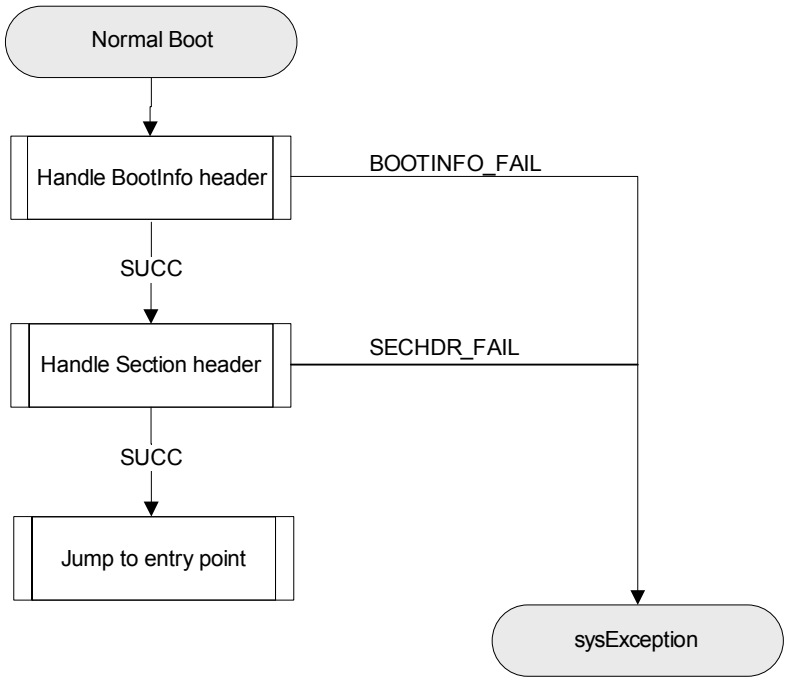
Note: The step "Decrypt and Check the image", the ciphertext is in Flash, and the decrypted plaintext is in SRAM.

The step "Verify TIM while loading it into RAM", means loading the code from Flash into SRAM. If the image is signed, it will verify the signature for TIM. Otherwise, it will copy the code without signature verification.

"Signed_Boot" and "Encrypted_Boot" are 2 independent Boot options in OTP memory.

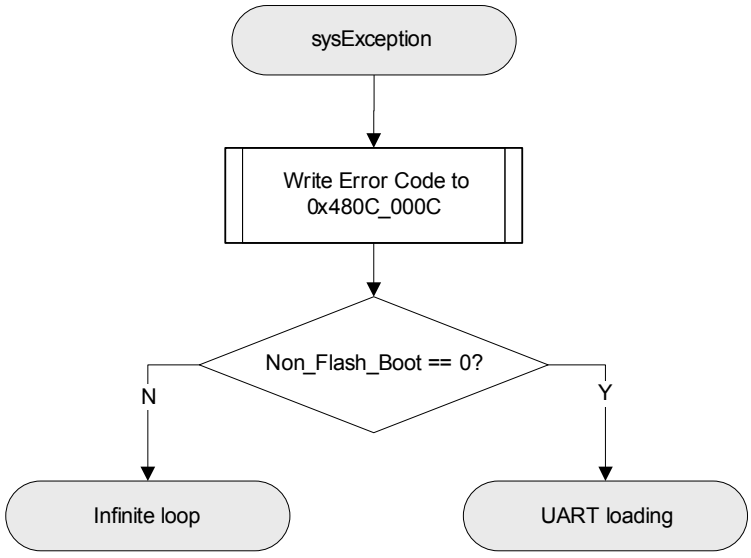
4.6.5 Normal Boot

Figure 18: Boot ROM Flow, Normal Boot



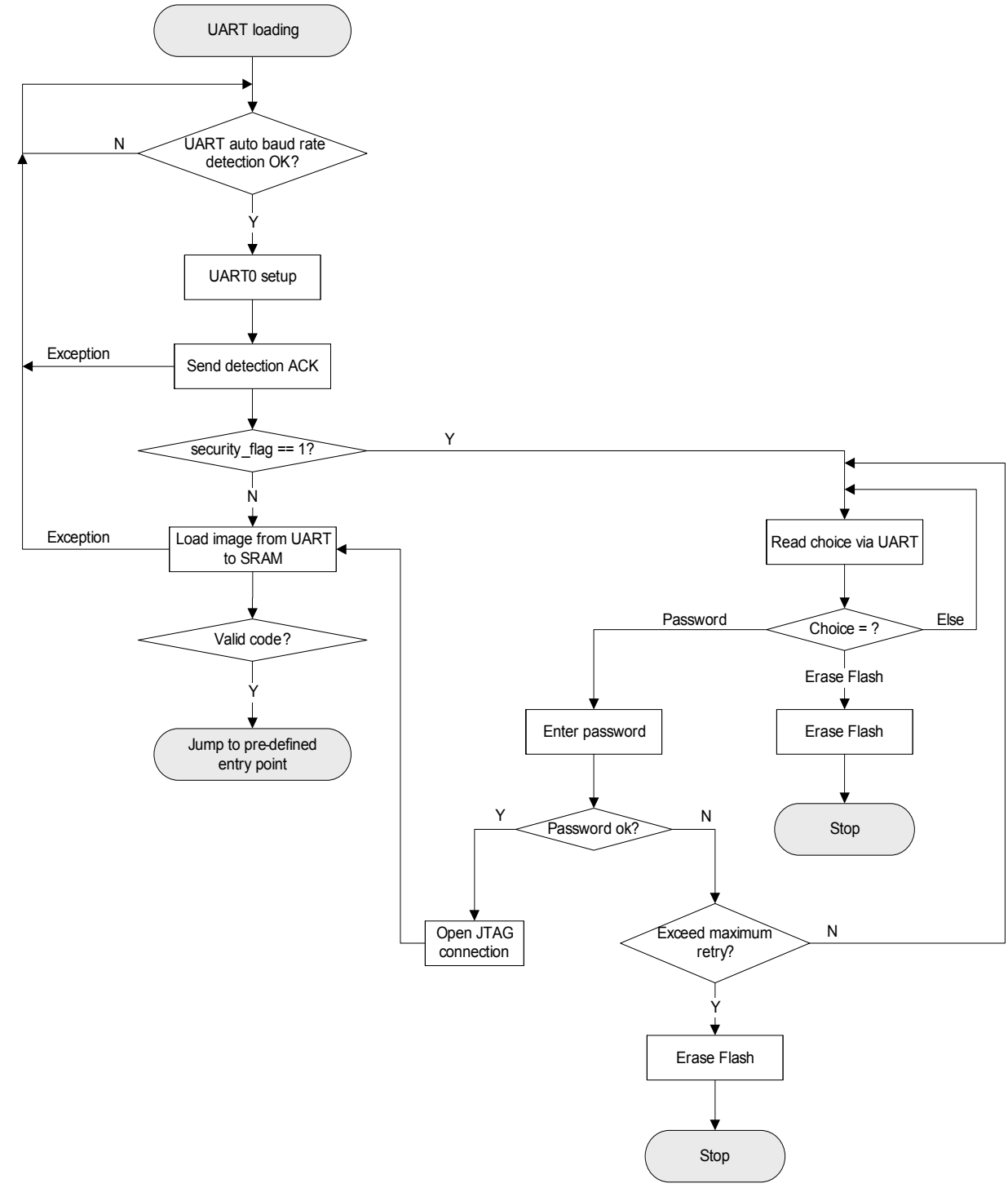
4.6.6 System Exception

Figure 19: Boot ROM Flow, System Exception



4.6.7 UART Loading

Figure 20: Boot ROM Flow, UART Loading



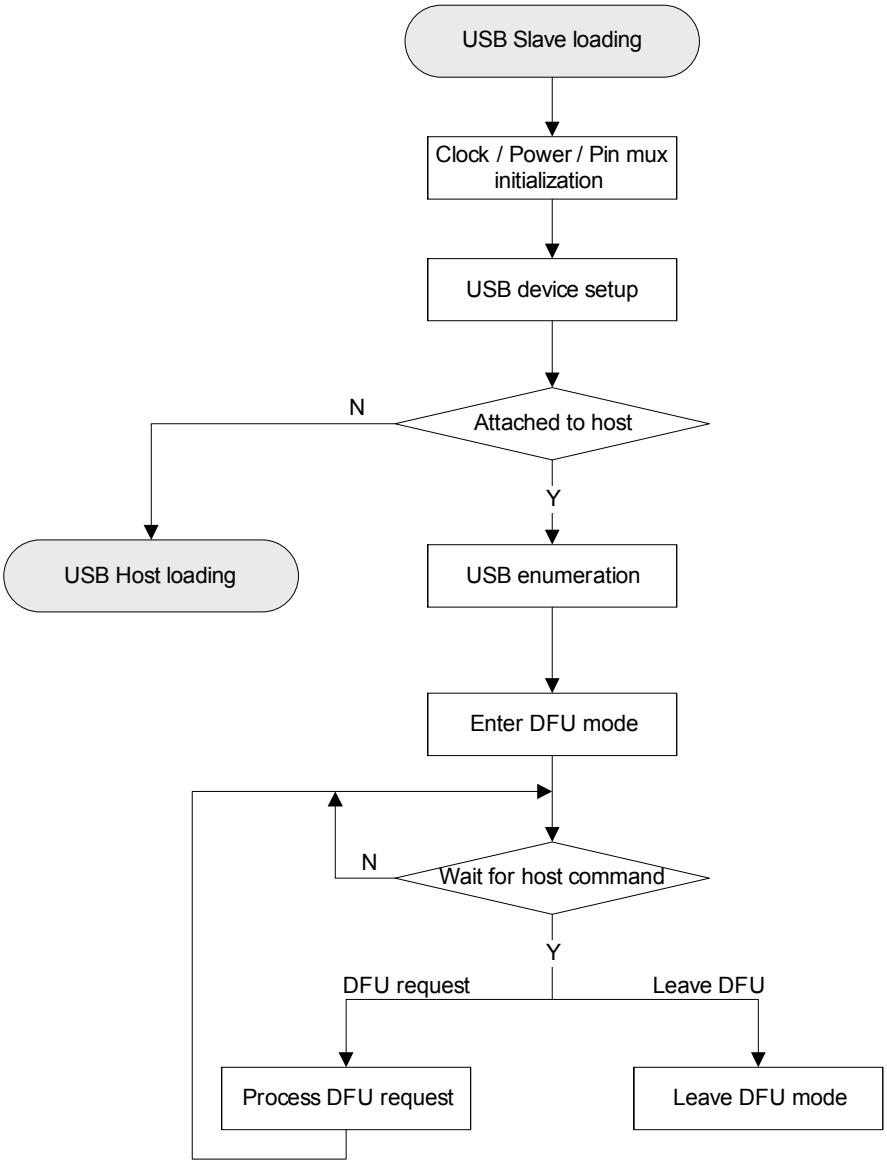
Note: The maximum value of "Exceed maximum retry" is 10, and the retry counter will reset when power ON or hardware reset.

If the password is verified as correct, the image can be loaded from the UART path. Meanwhile, JTAG is available for debug or re-Flash operation.

"Erase Flash" erases the entire Flash against hacking attempts.

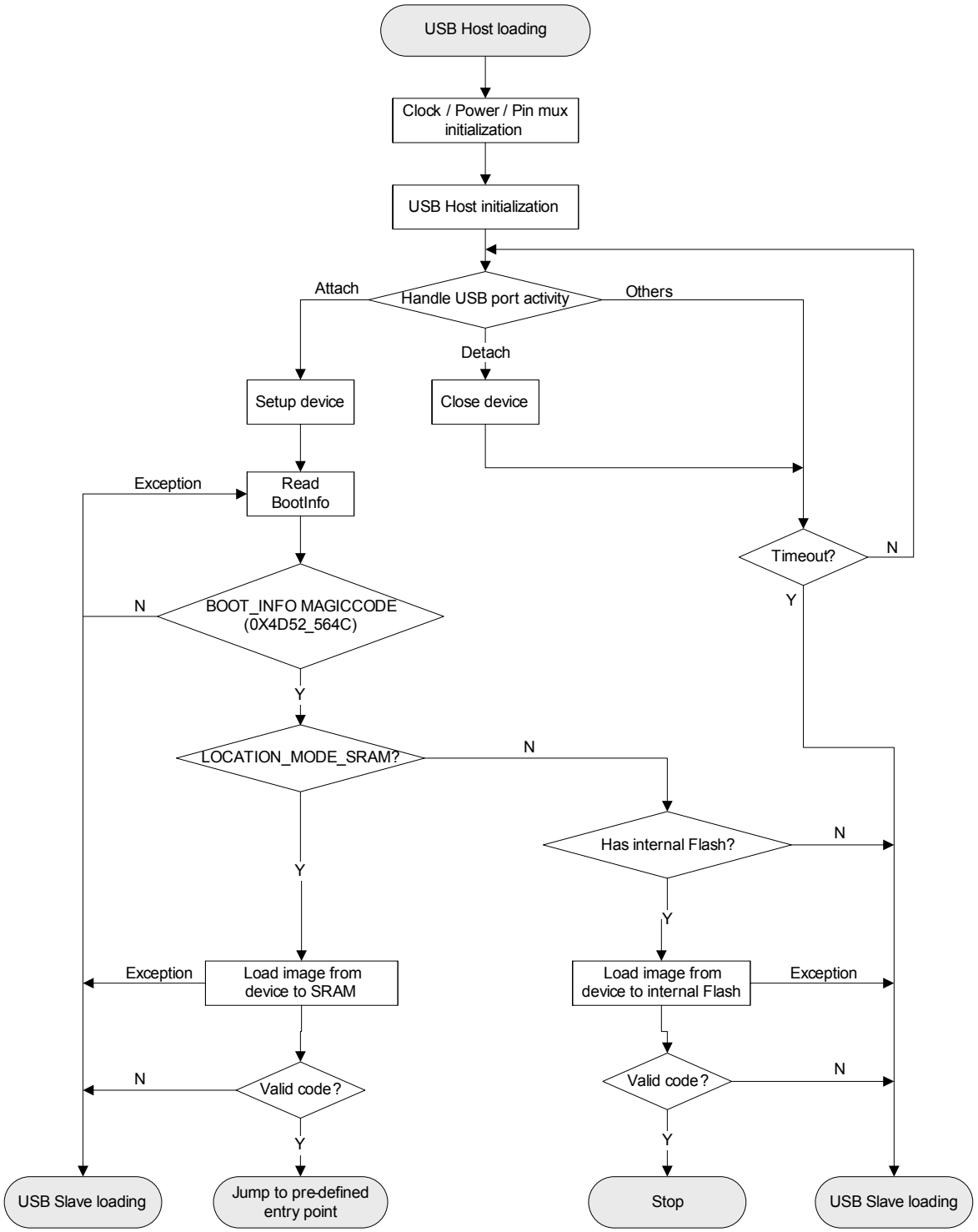
4.6.8 USB Slave Loading

Figure 21: Boot ROM Flow, USB Loading as Slave



4.6.9 USB Host Loading

Figure 22: Boot ROM Flow, USB Loading as Host



4.7 Boot from QSPI Flash

4.7.1 Code Image

The code image is in binary, which is generated by the Cortex-M4 tool chain, for example, ARM RVCT or IAR compiler and linker. A code image can be loaded into the space from:

- 0x10_0000 to 0x15_7FFF through the DBus
- 0x2000_0000 to 0x2001_FFFF through the system bus
- 32 K optional Flash_SRAM memory

It is not suggested to load all code images using Boot ROM. Instead, Boot ROM can load a boot loader into RAM and launch it to perform more sophisticated boot loading tasks.

The memory space from 0x2000_0000 to 0x2000_5657 is occupied by Boot ROM as the STACK and data sections. When Boot ROM starts upon reset, it will initialize this memory space.

Table 44 shows the memory usage.

Table 44: Boot ROM Memory Usage

Address	Usage	Occupied by Boot ROM when Wake-up from PM3	Occupied by Boot ROM when Wake-up from PM4 or on POR
0x2000_0000–0x2000_0039 0x2000_1000–0x2000_2656	data	yes	yes
0x2000_2658–0x2000_5657	heap	no	yes
0x2000_0040–0x2000_0FFF	stack	yes	yes

The user's code sections (excluding the data sections) should not cover this region. Otherwise, it may flush the Boot ROM STACK and data and cause errors.

4.7.2 Code Image Format

The format of the code image consists of:

- OEM public key—always located at the beginning of the image
- Digital signature
- Length of encrypted image
- Boot header (comprised of bootInfoHeader and SectionHeader)—includes important information employed during SoC boot
- User data—immediately follows SectionHeader; includes primary firmware code (PFC, containing OS and driver) and possible customer applications
- MIC

For QSPI boot, all the fields are valid.

For USB boot, only the boot header and user data are valid. All other fields are reserved.

Figure 23 shows a code image format.

Figure 23: Code Image Memory Mapping

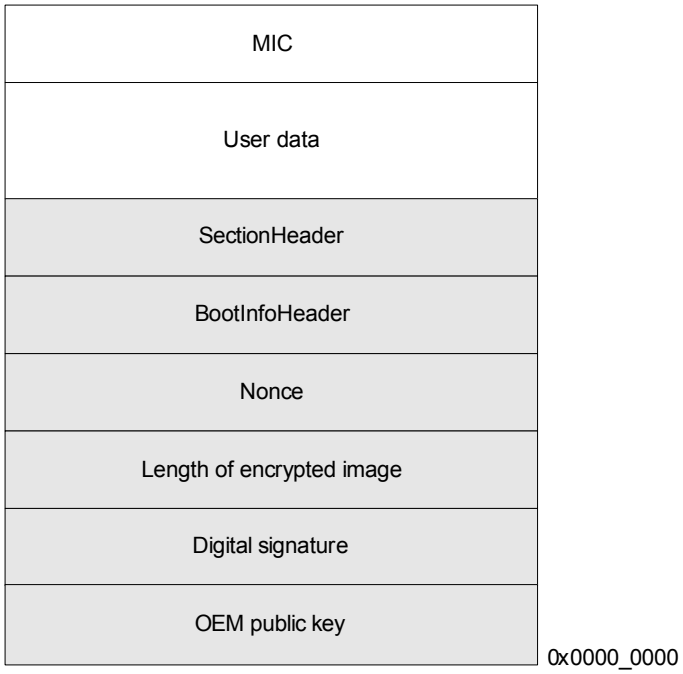


Table 45 shows a code image format. All fields are little endian.

Table 45: Code Image Fields

Section Name	Address	Size	Field Name	Description
OEM public key	0x00 to 0x125	294 bytes	publicKey	OEM Public Key for Digital Signature This field is only valid for QSPI boot when signedflag = 1.
Digital signature	0x126 to 0x225	256 bytes	Digital signature	Digital Signature for Hash of bootinfo header, sectionheader, and Code Image (includes the 4-byte CRC check field) This field is only valid for QSPI boot when signedflag = 1.
Length of encrypted image	0x226 to 0x229	4 bytes	Length of encrypted image	Length of Encrypted Image Includes the length of the bootInfo Header, section Header and Code image (includes 4-byte CRC field). This field is only valid for QSPI boot when encryptedFlag = 1.
Nonce(IV)	0x22A to 0x239	16 bytes	Nonce	Nonce of AES Decryption This field is only valid for QSPI boot when encryptedFlag = 1.

Table 45: Code Image Fields (Continued)

Section Name	Address	Size	Field Name	Description
bootInfoHeader	0x23A	32 bits	n/a	Reserved
	0x23E	32 bits	n/a	Reserved
	0x242	32 bits	commonCfg0	Common Configuration 0
	0x246	32 bits	magicCode	Magic Code 0x4D52564C (MRVL) means the code image is valid. Otherwise, Boot ROM regards this image as invalid.
	0x24A	32 bits	entryAddr	Address of Entry Function
	0x24E	32 bits	sfllCfg	Configure Frequency of SFLL
	0x252	32 bits	n/a	Reserved
	0x256	32 bits	flashcCfg0	Flash Controller Configuration0 This field is ignored if encryptedFlag = 1 or signedflag = 1 or boot from USB.
	0x25A	32 bits	flashcCfg1	Flash Controller Configuration1 This field is ignored if encryptedFlag = 1 or signedflag = 1 or boot from USB.
	0x25E	32 bits	flashcCfg2	Flash Controller Configuration2 This field is ignored if encryptedFlag = 1 or signedflag = 1.
	0x262	32 bits	flashcCfg3	Flash Controller Configuration3 Flash Address Offset value. Default value is 0x0. When flashcCfg0.flashcOffsetEn =1 then this register specifies the address offset from Flash base address that is used for all Flash memory accesses. This field is ignored if encryptedFlag = 1 or signedflag = 1 or boot from USB.
	0x266 to 0x285	32 bytes	n/a	Reserved
	0x286	32 bits	CRCCheck	CRC Check for bootInfoHeader Boot ROM will calculate a CRC value on this bootInfoHeader (excluding CRCCheck) with CRC mode CRC-16-CCITT. If the calculated CRC value matches CRCCheck, then CRC checking passes. The 16-bit CRC value is stored in the lowest 2 bytes (0x286-0x287) of this field. The highest 2 bytes (0x288-0x289) are always 0.

Table 45: Code Image Fields (Continued)

Section Name	Address	Size	Field Name	Description
Section Header	0x28A	32 bits	n/a	Reserved
	0x28E	32 bits	codeLen	Code Length
	0x292	32 bits	n/a	Reserved
	0x296	32 bits	destStartAddr	Start Address in SRAM where PFC is Loaded To
	0x29A	32 bits	n/a	Reserved
	0x29E	32 bits	bootCfg0	Boot Configuration 0
	0x2A2	32 bits	bootCfg1	Boot Configuration 1
	0x286 to 0x2B5	16 bytes	n/a	Reserved
	0x2B6	32 bits	CRCCheck	CRC Check for sectionHeader Boot ROM will calculate a CRC value on this sectionHeader (excluding CRCCheck) with CRC mode CRC-16-CCITT. If the calculated CRC value matches CRCCheck, then CRC checking passes. The 16-bit CRC value is stored in the lowest 2 bytes of this field. The highest 2 bytes are always 0.
Code image	0x2BA	Decided by code of user	Code image	Image to be Executed
MIC	After code image of user	128 bits	MIC	MIC of AES Decryption, Authentication Using AES-CCM

Table 46: Sub-Field in commonCfg0

Bits	Sub-Field Name	Default	Description
31:23	n/a	All 1's	Reserved
22:18	flashIntfPrescaler	5'b00100/ 5'b0001	Flash Interface Module (QSPI and FlashC) Clock Prescaler The default value of this field is related to the clock source. If the clock source is PLL, the default value of this field is 5'b00100. Otherwise, it is 5'b00001.
17	flashIntfPrescalerOff	1	Prescaler Select 0 = use flashIntPrescaler 1 = use default prescaler
16:10	n/a	All 1's	Reserved
9:8	clockSel	2'b11	Set System Clock Source 00 = use SFLL_200M for system clock; SFLL reference clock is MAINXTAL 01 = use SFLL_200M for system clock; SFLL reference clock is RC32M 10 = use MAINXTAL for system clock 11 = use RC32M for system clock
7:1	n/a	All 1's	Reserved
0	dmaEn	1	DMA Enable 0 = do not use DMA when loading code to SRAM 1 = enable DMA when loading code to SRAM This field is ignored if encryptedFlag = 1 or signedflag = 1 or boot from USB.

Table 47: Sub-Field in sfllCfg

Bits	Sub-Field Name	Default	Description
31:25	n/a	All 1's	Reserved
24:16	sfllRefdiv	0x60 / 0x50	Reference Divider Default value is 0x50 when reference clock is RC32M. Default value is 0x60 when reference clock is MAINXTAL.
15:7	sfllFbdiv	0xF0	Feedback Divider
6:1	n/a	All 1's	Reserved
0	sfllCfgOff	1	SFLL Configuration 0 = use configuration in sfllCfg 1 = use default SFLL configuration

Table 48: Sub-Field in flashcCfg0

Bits	Sub-Field Name	Default	Description
31	flashcCfgOff	1	Flash Controller Configuration 0 = use configuration in flashcCfg0, flashcCfg1, flashcCfg2, flashcCfg3 1 = use default Flash Controller configuration
30	flashcCacheEn	1	Enable Cache Mode Controls whether the read data from the Flash device is cached or not. 0 = read Data from Flash does not use the Flash cache 1 = read data from Flash through the Flash cache
29:24	n/a	All 1's	Reserved
23	flashcOffsetEn	0	Address Offset Enable 0 = all Flash Memory accesses do not use Address Offset 1 = using Address Offset defined in flashcCfg3 is enabled for all Flash memory accesses
22:21	flashcClkOutDly	2'b00	Add Delay on Outgoing Clock to Flash
20	flashcCapEdge	0	Serial Interface Data Capture Clock Edge Capture serial interface input data on either the rising or falling edge of the serial interface clock. This bit is used to allow more time to capture the input data.
19:18	flashcClkInDly	2'b01	Add delay on the Clock that the front end flip flops that capture read data from Flash
17:16	flashcDataDly	2'b01	Add delay on Incoming Data from Flash This needs to be used to tune the timing of the data capture inside the Flash Controller.
15	flashcClkPha	0	Serial Interface Clock Phase Selects the serial interface clock phase. 0 = data is captured on the rising edge of the serial clock when flashcClkPol =0 and on the falling edge when CLK_POL=1 1 = data is captured on the falling edge of the serial clock when flashcClkPol =0 and on the rising edge when CLK_POL=1
14	flashcClkPol	0	Serial Interface Clock Polarity Selects the serial interface clock as HIGH or LOW when inactive. 0 = serial Interface clock is LOW when inactive 1 = serial Interface clock is HIGH when inactive
13:4	n/a	all 1's	Reserved

Table 48: Sub-Field in flashcCfg0 (Continued)

Bits	Sub-Field Name	Default	Description
3:0	flashcCmdType	4'b0101	Serial Flash Command Type The Flash Controller will automatically build the necessary Instruction, followed by Address, dummy clocks etc., based on this Command Type field for Winbond devices. 0x0 = read data 0x1 = fast read 0x2 = fast read dual output 0x3 = fast read quad output 0x4 = fast read dual I/O 0x5 = fast read dual I/O with continuous read mode 0x6 = fast read quad I/O 0x7 = fast read quad I/O with continuous read mode 0x8 = word read quad I/O 0x9 = word read quad I/O with continuous read mode 0xA = octal word read quad I/O 0xB = octal word read quad I/O with continuous read mode Others = reserved

Table 49: Sub-Field in flashcCfg1

Bits	Sub-Field Name	Default	Description
31	flashcUseCfgOvrd	0	Use Configuration Override This bit when set by software, overrides the built-in hardware Flash transfer generation defined through flashCfg0.flashcCmdType. Software has control over Instruction, Address, and other configuration. Software needs to program all the necessary fields in FCCR2 to successfully complete a transfer to Flash. 0 = use flashCfg0.flashcCmdType to determine/build Flash command/transfer 1 = use flashCfg1 and flashCfg2 to determine/build Flash Command/transfer
30:29	flashcDataPin	0x0	Data Transfer Pins Number of pins used to transfer data. 0x0 = use 1 pin for data 0x1 = use 2 pins for data 0x2 = use 4 pins for data 0x3 = reserved
28	flashcAddrPin	0x0	Address Transfer Pins Number of pins used to transfer the Flash address. 0 = use 1 pin 1 = use number of pins as indicated by DATA_PIN field in this register
27	flashcByteLen	0x0	Byte Length Number of bytes in each serial transfer. 0 = 1 byte 1 = 4 bytes
26:14	n/a	All 1’s	Reserved

Table 49: Sub-Field in flashcCfg1 (Continued)

Bits	Sub-Field Name	Default	Description
13:12	flashcDummyCnt	0x0	Dummy Count Defines the number of dummy bytes that need to be sent to Flash. The data shifted out is always 0. 0x0 = 0 bytes 0x1 = 1 byte 0x2 = 2 bytes 0x3 = reserved
11:10	n/a	All 1's	Reserved
9:8	flashcRmCnt	0x0	Read Mode Count Number of Read Mode bytes (as defined in flashCfg2.flashcRdmode) that are sent out to Flash. 0x0 = 0 bytes 0x1 = 1 byte 0x2 = 2 bytes 0x3 = reserved
7	n/a	1	Reserved
6:4	flashcAddrCnt	0x0	Address Count Number of bytes of address that is sent to Flash. 0x0 = 0 bytes 0x1 = 1 byte 0x2 = 2 bytes 0x3 = 3 bytes 0x4 = 4 bytes Others = reserved
3:2	n/a	All 1's	Reserved
1:0	flashcInstrCnt	0x0	Instruction Count Number of bytes of Instruction (defined in flashCfg2.flashcInstr) to send to Flash. 0x0 = 0 bytes 0x1 = 1 byte 0x2 = 2 bytes 0x3 = reserved

Table 50: Sub-Field in flashcCfg2

Bits	Sub-Field Name	Default	Description
31:16	flashcRdmode	0x0	Flash Read Mode Determines the Read Mode information that gets sent to the Flash device after Address cycle. When flashcCfg1.flashcRmCnt =0, this register content is not sent out. When flashcCfg1.flashcRmCnt = 1, bits [7:0] of this register are sent out. When flashcCfg1.flashcRmCnt =2, bits [15:8] of this register are sent out first followed by bits [7:0].
15:0	flashcInstr	0x0	Flash Instruction The contents of this register define the instruction Op code that gets sent to the Flash device. When flashcCfg1.flashcInstrCnt =0, this register content is not sent out. When flashcCfg1.flashcInstrCnt =1, bits [7:0] of this register are sent out. When flashcCfg1.flashcInstrCnt =2, bits [15:8] of this register are sent out first followed by bits [7:0].

Table 51: Sub-Field in bootCfg0

Bits	Sub-Field Name	Default	Description
31	emptyCfg	1	emptyCfg 0 = bootCfg1 is applied 1 = bootCfg1 is ignored
30:0	n/a	All 1’s	Reserved

Table 52: Sub-Field in bootCfg1

Bits	Sub-Field Name	Default	Description
31:21	n/a	All 1’s	Reserved
20	flashcOff	1	Decide Whether to Enable Flash Controller to Support XIP This field is valid only when encryptedFlag = 0 and signedflag = 0 and boot from QSPI. 0 = enable Flash Controller and XIP is supported 1 = disable Flash Controller and code cannot be executed directly in Flash
19:16	n/a	All 1’s	Reserved
15	flashProgMode	0	Decide internal Flash Program Mode This field is valid only for USB host boot. 0 = internal Flash normal page program 1 = internal Flash quad page program
14	locationMode	0	Decide Load Code Image to SRAM or Internal Flash This field is valid only for USB host boot. 0 = load code to SRAM 1 = load code to Flash

Table 52: Sub-Field in bootCfg1 (Continued)

Bits	Sub-Field Name	Default	Description
13	LEDEnable	1	LED Enable This field is valid only for USB host boot. If locationMode = 1, this bit can decide if LED is on after copying code to internal Flash. 0 = no pulse is on GPIO_16 1 = GPIO_16 outputs a pulse after copying code to internal Flash. If connect a LED to GPIO_16, it can indicate when the code loading is finished.
12:9	n/a	All 1's	Reserved
8:6	flashReadMode	3'b000	Flash Read Mode 3'b000 = normal read mode 3'b001 = fast read mode 3'b010 = fast read dual out mode 3'b011 = fast read quad out mode Others = normal read mode
5:1	n/a	All 1's	Reserved
0	sramModeEn	0	32 KB FLASH_SRAM Mode Enable This field is valid only for the first section header in the section chain. Controls whether the 32 KB optional memory is used as Flash Cache or SRAM. This field is corresponding to the register FCCR.SRAM_MODE_EN, which is in the Flash Controller module. 0 = use the 32 KB memory as Flash Cache 1 = use the 32 KB memory as SRAM flashcCacheEn needs to be '0' when sramModeEn =1 flashcCacheEn and sramModeEn can not be set to '1' at the same time

Table 53: User Data Format

Byte	0	1	2	...	codeLen-1	codeLen	codeLen+1	codeLen+2	codeLen+3
content	User data					CRC Signature for User Data with CRC Mode CRC-16-CCITT For 16-bit CRC mode, the lowest 2 bytes are valid. The highest 2 bytes are always 0. If encrypted mode is enable, CRC is not calculated, and this field is reserved.			

4.8 Boot from UART

When UART is selected as the boot interface, code can be loaded through UART. This section shows the Boot ROM UART download protocol and UART packet format.

See Section 4.12.3, Sample of Code Loading Through UART, on page 135 for an example of loading code through UART, which is based on this download protocol.

Terms used are as follows:

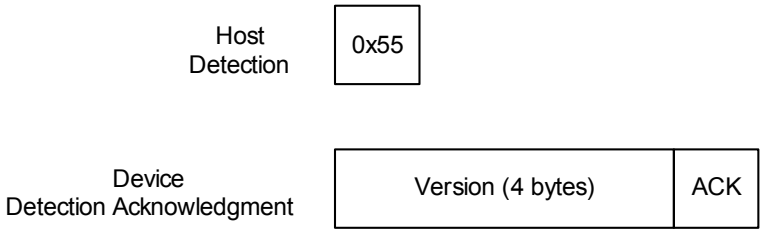
- Host—external device with UART interface
- Device—88MW300/302

The basic download process is as follows:

1. Host sends detection byte until Device acknowledges it.

Figure 24 shows the Detection and Detection Acknowledgement (ACK) packets.

Figure 24: Detection and Detection Acknowledgment Packets



Version = version number of Boot ROM

ACK:
0xA3 = detection return without password request
0xAC = detection return with password request for security mode

A Detection packet is sent by the Host. A Detection Acknowledge packet is sent by the Device.

Since the Device needs time to process the detection packet from the Host, the Host should provide an interval before sending out the next detection byte when not receiving a Device Detection Acknowledgement. The recommended interval is 10 ms or more.

There are 2 kinds of ACKs from the Device, 0xA3 and 0xAC.

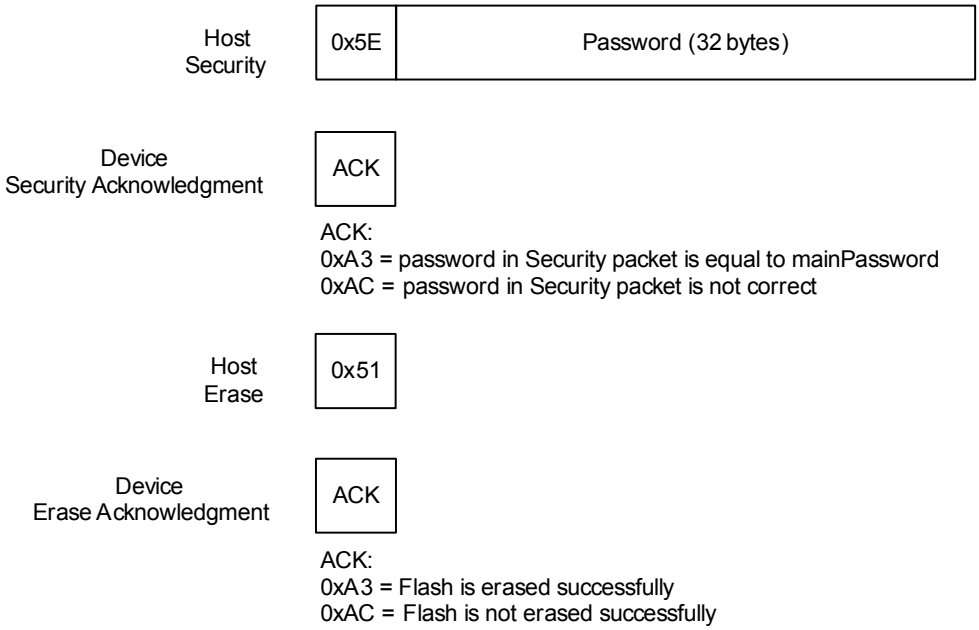
- If the security_flag in the OTP is not set, the device will return 0xA3 as acknowledgment.
- If the security_flag in the OTP is set, the device will return 0xAC indicating that the user needs to input a password for further operation.

Alternatively, the user can select to erase all Flash content.

1. If the ACK field of the Detection Acknowledgement packet is 0xAC, Host sends a Security or Erase packet. If the ACK field of the Detection Acknowledgement packet is 0xA3, go to Step3.

Figure 25 shows the Security/Erase and acknowledgement packets.

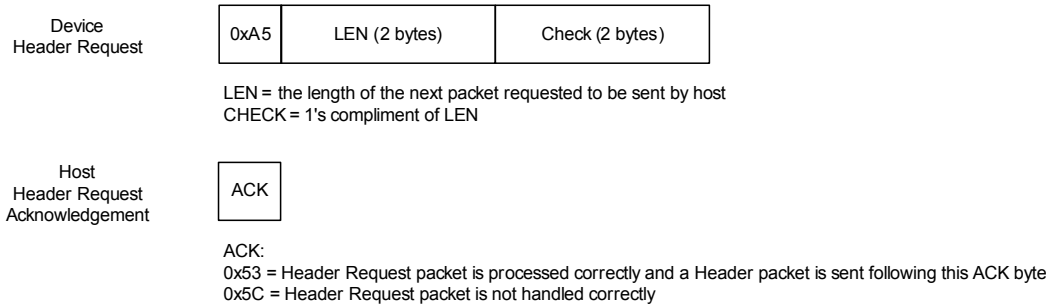
Figure 25: Security/Erase and Acknowledgment Packets



If the Host sends an Erase packet, the Device erases the Flash and sends an Erase Acknowledgement packet according to the result. If the Host sends a Security packet, the Device compares the received password with mainPassword and sends a Security Acknowledgement packet with the comparison result. If it returns 0xAC, the flow goes to Step3. Otherwise, it repeats Step2.

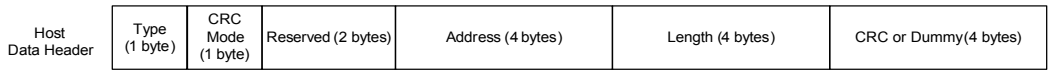
2. Device sends a Header Request packet with LEN = 16. After receiving Header Request, Host sends a Header Request Acknowledgement. See Figure 26.

Figure 26: Header Request and Acknowledgment Packet



3. Host sends a Data Header packet after the Header Request Acknowledgement. See Figure 27.

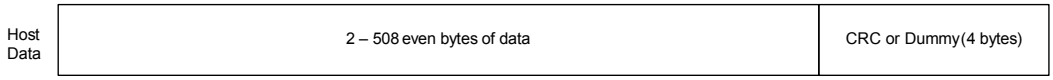
Figure 27: Data Header Packet



- Type = type of packet
  - 0x01 = Data Header packet with CRC check (this packet and next Data packet are with CRC check)
  - 0x02 = Data Header packet without CRC check (this packet and next Data packet are without CRC check)
- CRC Mode = CRC mode of this packet and the following data packet
  - 0 = CRC-16-CCITT with polynomial 0x8408
  - 1 = CRC-16-IBM with polynomial 0xA001
  - 2 = CRC-16-T10-DIF with polynomial 0xEDD1
  - 3 = CRC-32-IEEE with polynomial 0xEDB88320
  - Others = reserved
- Address = start address, where the data in the following Data packet are placed
- Length = length of the following Data packet (include the 4-byte CRC check)
- CRC =
  - If Type = 0x01, this field is the CRC signature of this packet, which is calculated on the first 12 bytes of the packet.
  - If Type = 0x02, the value of this field is ignored.

4. Device sends Header Request. The LEN field is equal to the length field of the Data Header packet received. See Figure 26, Header Request and Acknowledgment Packet, on page 126.
5. After receiving Header Request packet, Host sends an ACK packet (see Figure 26, Header Request and Acknowledgment Packet, on page 126) followed by a Data packet (see Figure 28, Data Packet, on page 127). The length is equal to the LEN field of the Header Request packet sent by Device.

Figure 28: Data Packet

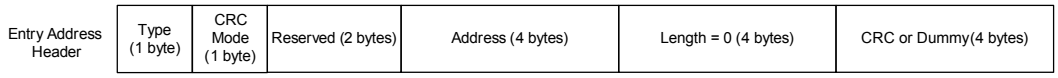


- CRC =
  - If Type field of the previous Data Header packet is 0x01, this field is the CRC signature of this packet, which is calculated on the whole packet excluding the last 4 bytes.
  - If Type = 0x02, the value of this field is ignored.

6. Repeat Step3 to Step6 until all data bytes are downloaded to the Device.
7. Device sends another Header Request with length = 16.

8. Host receives Header Request, and then sends an ACK packet (see Figure 26, Header Request and Acknowledgment Packet, on page 126), followed by an Entry Address Header packet (see Figure 29, Entry Address Header Packet, on page 128).

Figure 29: Entry Address Header Packet



- Type = the type of this packet
  - 0x04 = Entry Address Header packet with CRC check
  - 0x08 = Entry Address Header packet without CRC check
  - Others = reserved
- CRC Mode = CRC mode of this packet
  - 0 = CRC-16-CCITT with polynomial 0x8408
  - 1 = CRC-16-IBM with polynomial 0xA001
  - 2 = CRC-16-T10-DIF with polynomial 0xEDD1
  - 3 = CRC-32-IEEE with polynomial 0xEDB88320
  - Others = reserved
- Address = entry address
- Length = for Entry Address Header, this field is always 0
- CRC =
  - If Type = 0x04, this field is the CRC signature of this packet, which is calculated on the first 12 bytes of the packet.
  - If Type = 0x08, the value of this field is ignored.

9. Device sets the entry address according to the received Entry Address Header.
10. Device sends out the last Header Request with LEN = 0 (0xA5 0x00 0x00 0xFF 0xFF). This tells the Host that this is the last request from Boot ROM.
11. Host sends an ACK packet, then terminates.
12. Device jumps to the entry address.

4.9 USB Disk

This section shows the method for booting from a USB disk. In this boot method, the 88MW300/302 acts as a USB host.

4.9.1 Requirements

Requirements for booting from a USB disk include:

- USB disk (which is used as the boot media) should get enumerated as a USB Mass Storage Class device
- File system on the USB disk should be FAT32
- Image to be booted should be present in root folder
- Name of image to be booted should be boot.bin

4.9.2 Options

The image must have a header as described in Section 4.7, Boot from QSPI Flash, on page 114. The image can either be written to Flash or downloaded to SRAM and run directly from there. The following bits in the bootcfg1 section of the Section Header must be set correctly:

- flashProgMode
- locationMode

4.9.3 Procedure

The procedure for booting from a USB disk is as follows:

1. Set the boot configuration pins for booting from USB (see Section 4.3, Boot Source Selection, on page 102).
2. Copy boot.bin (with valid header) to the USB disk root folder.
3. Ensure that the file system on the USB disk is FAT32.
4. Connect the USB disk to the USB OTG port.
5. Reset the processor.
6. The processor should boot from the USB disk.

4.10 USB DFU

Table 54: DFU Request Type

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0010 0001b	DFU_DETACH	wTimeout	interface	0	none
0010 0001b	DFU_DNLOAD	wBlock	interface	length	Marvell-specific
1010 0001b	DFU_UPLOAD	wBlock	interface	length	Marvell-specific
1010 0001b	DFU_GETSTATUS	0	interface	6	status
0010 0001b	DFU_CLRSTATUS	0	interface	0	none
1010 0001b	DFU_GETSTATE	0	interface	1	status
0010 0001b	DFU_ABORT	0	interface	0	none

Table 55: DFU Requests

DFU Request	Command	wValue	wLength	Data
DFU_UPLOAD	Read Memory	0x0001 to 0xFFFF	1 to 2048	address = (wValue-1)*2048 + offset (set by set address pointer command)
DFU_DNLOAD	Write Memory	0x0001 to 0xFFFF	1 to 2048	
	Set Memory Type	0	2	0x21 + bMemoryType
	Set Address Pointer	0	5	0x22 + dwAddressPointer
	Chip Erase	0	1	0x23
	Sector Erase	0	5	0x23 + wStartSector+wSectorSize
	Deactivate Security Mode	0	5	0x24 + dwPassword
	Leave DFU Mode	0	0	none

A DFU download example sequence (of a USB Slave Boot in Boot ROM) is as follows:

1. Set Mem Type: 21 01 00 00 00 00 02 00.
   Data: 21 01 (indicates Flash Mem); 21 00 (indicates SRAM Mem)
2. Get Status: a1 03 00 00 00 00 06 00.
3. Set Address Point: 21 01 00 00 00 00 05 00.
   Data: 22 00 00 00 00 (set address pointer 0x0 in Flash)
4. Get Status: a1 03 00 00 00 00 06 00.
5. Write Mem: 21 01 01 00 00 00 XX XX (XX XX: length of image file).
   Data: image data
6. Get Status: a1 03 00 00 00 00 06 00.

4.11 Fast Boot at Wake-up from PM3 Mode

1. Wake-up from PM3 low-power mode.
2. Read a 32-bit entry point address from 0x480C_0000 – 0x480C_0003 in AON 4K_MEM.
3. Jump to the entry address directly.

4.12 Additional Boot ROM Information

4.12.1 Boot ROM GPIOs

Table 56: Boot ROM GPIOs

GPIO Name	Function	Description
GPIO_27	CON[4]	Select Boot Mode See Section 1.5, Configuration Pins, on page 65.
GPIO_16	CON[5]	Select Boot Mode See Section 1.5, Configuration Pins, on page 65.
	GPIO_16	Output a pulse after copying code to internal Flash through USB interface. It can be disabled through the bit bootCfg1.LEDEnable. See Table 52, Sub-Field in bootCfg1, on page 123.
GPIO_2	UART0_TXD	UART0 Interface
GPIO_3	UART0_RXD	
	GPT0_CH3	GPT0 Interface (UART baud rate detection)
GPIO_28	QSPI_SSn	Used to Connect an External Flash
GPIO_29	QSPI_CLK	
GPIO_30	QSPI_D0	
GPIO_31	QSPI_D1	
GPIO_32	QSPI_D2	
GPIO_33	QSPI_D3	

4.12.2 Flash Requirements for Flash Boot Mode

An off-chip SPI/QSPI serial Flash can be used for Flash boot, which must meet the following conditions:

- SPI bus operation mode, Mode 0 (CPOL = 0, CPHA = 0) or Mode 3 (CPOL=1, CPHA=1)
- Supports clock frequency of 16 MHz or higher

Figure 30: Flash Boot Mode, Timing

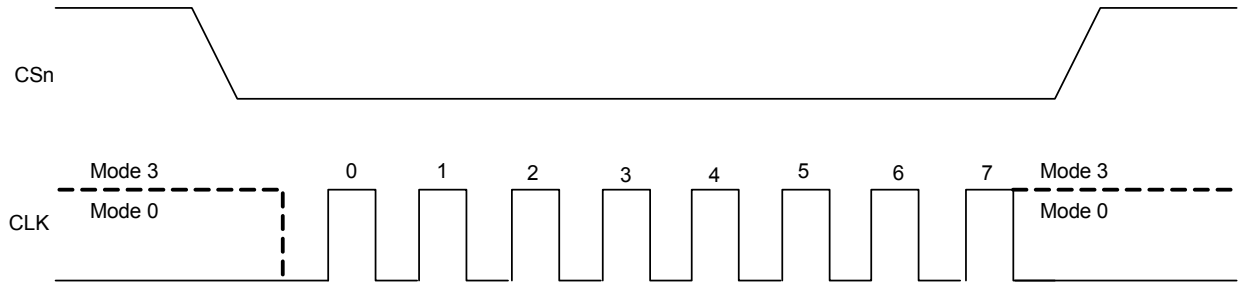


Table 57 shows the commands required to support basic Flash boot function.

Table 57: Flash Boot Mode, Basic Functions

Instruction	Byte 1 (code)	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	n-Bytes
Release from deep power-down	ABh	--	--	--	--	--	--
Read data	03h	A23 to A16	A15 to A8	A7 to A0	(D7 to D0)	(next byte)	continuous
Write enable	06h	--	--	--	--	--	--
Chip erase	C7h	--	--	--	--	--	--
Read status register	05h	(S7 to S0)	--	--	--	--	--

Table 58 shows additional instructions that may be used during boot.

Table 58: Flash Boot Mode, Additional Functions

Instruction	Byte 1 (code)	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	n-Bytes
Continuous Read Mode Reset[1]	FFh	FFh	--	--	--	--	--
Fast Read	0Bh	A23 to A16	A15 to A8	A7 to A0	dummy	(D7 to D0)	(next byte) continuous
Fast Read Dual Output	3Bh	A23 to A16	A15 to A8	A7 to A0	dummy	(D7 to D0)[2]	(next byte) continuous
Fast Read Quad Output	6Bh	A23 to A16	A15 to A8	A7 to A0	dummy	(D7 to D0)[3]	(next byte) continuous
Write Enable for Volatile Status Register	50h	--	--	--	--	--	--
Write Status Register	01h	S7 to S0	S15 to S8	--	--	--	--
Write Status Register 2	35h	(S15 to S8)	--	--	--	--	--
Page Program	02h	A23 to A16	A15 to A8	A7 to A0	D7 to D0	next byte	--
Quad Page Program	32h	A23 to A16	A15 to A8	A7 to A0	D7 to D0[4]	next byte	--
Sector Erase	20h	A23 to A16	A15 to A8	A7 to A0	--	--	--

1. This instruction is recommended when using the Dual or Quad "Continuous Read Mode" features.
2. . Dual Output Data:
IO0 = (D6, D4, D2, D0)
IO1 = (D7, D5, D3, D1)
3. Quad Output Data:
IO0 = (D4, D0, ……)
IO1 = (D5, D1, ……)
IO2 = (D6, D2, ……)
IO3 = (D7, D3, ……)
4. Quad Page Program Input Data:
IO0 = D4, D0, ……
IO1 = D5, D1, ……
IO2 = D6, D2, ……
IO3 = D7, D3, ……

Table 59 shows the supported SPI Flash for basic Flash boot function (but not limited to these). However, if the complete boot function is needed, the Winbond W25 series is recommended.

Table 59: SPI Flash for Basic Flash Boot Function

Company	Part Number
WINBOND	W25Q80BV
EON	EN25F80
ATMEL	AT25DF081
MXIC	MX25L8005
SPANSION	S25FL008A
ST	M25P80
AMIC	A25L080
GIGADEVICE	GD25Q80

4.12.3 Sample of Code Loading Through UART

This section provides a sample of code loading through UART. In this sample, security mode is enabled, and CRC check is enabled during data transfer.

1. Host: 55 // Detection packet
2. Device: 01 00 00 00 ac // Detection Acknowledgement packet
   // 01 00 00 00: the Boot ROM version number, 01 is the lowest byte
   // ac: the ACK field, it means Boot ROM is in security mode and the host is requested to send a password
3. Host: 5e 14 25 36 47 58 69 7a 8b 9c ad be cf 00 01 23 45 67 89 10 11 22 33 44 55 66 77 88 99 aa bb cc dd // Security packet
   // 5e: it indicates that this is a Security packet
   // 14 25 36 47 58 69 7a 8b 9c ad be cf 00 01 23 45 67 89 10 11 22 33 44 55 66 77 88 99 aa bb cc dd: 256 bits password, as same as AES Key, 14 is the lowest byte
4. Device: ac // Security Acknowledgement packet
   // ac: it means the password received from the host (0x58473625) is equal to bootInfo.mainPassword
5. Device: a5 10 00 ef ff // Header Request packet
   // a5: it means this is a Header Request packet
   // 10 00: the requested length. It is 0x0010
   // ef ff: the 1's complement of 10 00
6. Host: 53 // Header Request Acknowledgement packet
   // It means the host recognizes and processes the Header Request packet successfully
7. Host: 01 03 00 00 00 00 11 00 0c 00 00 00 e8 44 1b 80
   // Data Header packet
   // 01: This is a Data Header packet, and CRC check is applied to this packet and the next Data packet
   // 03: the CRC mode is CRC-32-IEEE
   // 00 00: reserved
   // 00 00 11 00: The start address, where data in the next Data packet are placed, is 0x00110000
   // 0c 00 00 00: the length of the next Data packet is 0x0000000c
   // e8 44 1b 80: CRC signature for this packet
8. Device: a5 0c 00 f3 ff // Header Request packet
   // a5: it means this is a Header request packet
   // 0c 00: the requested length. It is 0x000c
   // f3 ff: the 1's complement of 0c 00
9. Host: 53 // Header Request Acknowledgement packet
   // It means the host recognizes and processes the Header Request packet successfully

10. Host: 00 04 00 20 7d 00 11 00 46 50 02 ff
// Data packet
// 00 04 00 20 7d 00 11 00: data which need to be stored to memory
// 46 50 02 ff: CRC signature of this packet
11. Device: a5 10 00 ef ff // The function of Packet 11 - 58 is similar to Packet 5-10
// Please refer to the comments of Packet 5-10
12. Host: 53
13. Host: 01 03 00 00 08 00 11 00 14 00 00 00 2d 20 53 c6
14. Device: a5 14 00 eb ff
15. Host: 53
16. Host: 07 48 00 68 50 f0 01 00 05 49 08 60 05 48 0f 21 d7 fb b5 98
17. Device: a5 10 00 ef ff
18. Host: 53
19. Host: 01 03 00 00 18 00 11 00 14 00 00 00 06 11 e8 ba
20. Device: a5 14 00 eb ff
21. Host: 53
22. Host: 01 60 05 48 0f 21 01 60 04 48 0f 21 01 60 f8 e7 8b ec 65 e5
23. Device: a5 10 00 ef ff
24. Host: 53
25. Host: 01 03 00 00 28 00 11 00 14 00 00 00 7b 42 25 3f
26. Device: a5 14 00 eb ff
27. Host: 53
28. Host: 04 00 0a 48 0c 00 06 46 18 00 06 46 24 00 06 46 b1 8c 15 0c
29. Device: a5 10 00 ef ff
30. Host: 53
31. Host: 01 03 00 00 38 00 11 00 14 00 00 00 50 73 9e 43
32. Device: a5 14 00 eb ff
33. Host: 53
34. Host: 00 f0 09 f8 00 28 01 d0 c0 46 c0 46 00 20 ff f7 38 73 1c 72
35. Device: a5 10 00 ef ff
36. Host: 53
37. Host: 01 03 00 00 48 00 11 00 14 00 00 00 c0 e2 ce ef
38. Device: a5 14 00 eb ff
39. Host: 53
40. Host: df ff 00 f0 02 f8 01 20 70 47 80 b5 00 f0 02 f8 3f 21 74 71
41. Device: a5 10 00 ef ff
42. Host: 53
43. Host: 01 03 00 00 58 00 11 00 14 00 00 00 eb d3 75 93
44. Device: a5 14 00 eb ff
45. Host: 53
46. Host: 00 bf 00 00 07 46 38 46 00 f0 02 f8 fb e7 00 00 58 c7 59 e6
47. Device: a5 10 00 ef ff
48. Host: 53
49. Host: 01 03 00 00 68 00 11 00 14 00 00 00 96 80 b8 16
50. Device: a5 14 00 eb ff

51. Host: 53
52. Host: 80 b5 c0 46 c0 46 02 4a 11 00 18 20 ab be fb e7 9c 9a 31 11
53. Device: a5 10 00 ef ff
54. Host: 53
55. Host: 01 03 00 00 78 00 11 00 14 00 00 00 bd b1 03 6a
56. Device: a5 14 00 eb ff
57. Host: 53
58. Host: 26 00 02 00 c0 46 c0 46 c0 46 c0 46 ff f7 d8 ff b3 de e5 8f
59. Device: a5 10 00 ef ff // Header Request packet
    // a5: it means this is a Header request packet
    // 10 00: the requested length. It is 0x0010
    // ef ff: the 1's complement of 10 00
60. Host: 53 // Header Request Acknowledgement packet
    // It means the host recognizes and processes the Header Request packet successfully
61. Host: 04 03 00 00 7d 00 11 00 00 00 00 00 01 08 3b 65
    // Entry Address Header packet
    // 04: This is an Entry Address Header packet with CRC check
    // 03: CRC mode is CRC-32-IEEE
    // 00 00: reserved
    // 7d 00 11 00: entry address is 0x0011007d
    // 00 00 00 00: this field is always 0 for Entry Address Header Packet
    // 01 08 3b 65: CRC signature for this packet
62. Device: a5 00 00 ff ff // Header Request packet
    // a5: it means this is a Header request packet
    // 00 00: the requested length is 0
    // ff ff: the 1's complement of 00 00
63. Host: 53 // Header Request Acknowledgement packet
    // It means the host recognizes and processes the Header Request packet successfully

THIS PAGE INTENTIONALLY LEFT BLANK

5 Flash Controller

5.1 Overview

The 88MW300/302 Flash Controller provides a communication path between the Cortex-M4F CPU and the off-chip Flash memory. Main components of the controller include the Quad Serial interface, Flash Cache, and Flash Controller configuration registers.

5.2 Interface

The Flash Controller has an SPI interface (supports Quad-mode) that communicates with the Flash memory.

5.3 Cache

The Flash Cache is a 32 KB, 8-way set associative cache, which can cache data from a 16 MB address range. The cache is organized as 128 sets, with each set consisting of 8, 32-byte wide lines. The 32 KB is a SRAM type memory that holds CPU instructions and/or data. There is a small amount of Content Addressable Memory (CAM) memory that is used to hold information required to determine cache hit or miss.

The Flash Cache also has logic that compares the CAM information (tags) to the tag in the requested address to see if it resides in the set. In addition, the Flash Cache includes replacement logic. The Flash Cache uses a pseudo-LRU algorithm to determine which line within a set needs eviction, if needed. The pseudo least recently used algorithm will first select any cache line which is not valid, and if all 8 cache lines are valid it will select the cache line which it deems has been least recently used for eviction and replacement.

5.4 Functional Description

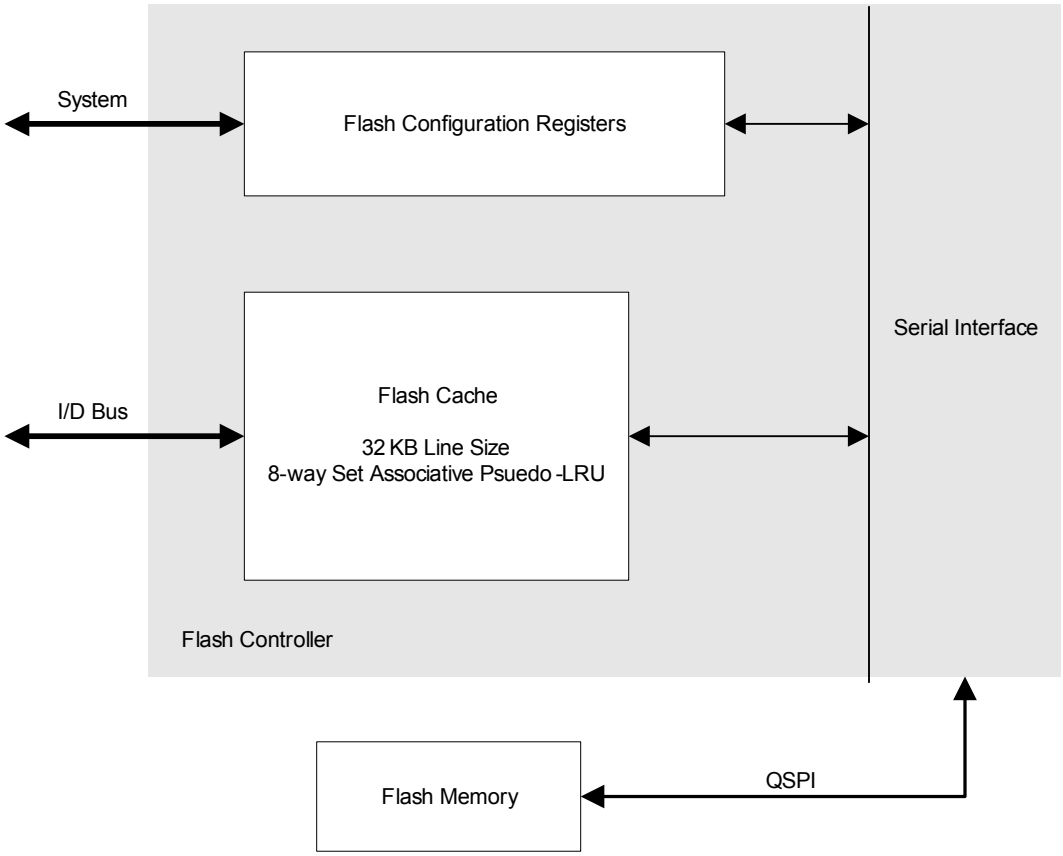
The Flash Controller provides a path for executing code (or data) residing in Flash memory. A read request or a code (or data) fetch by CM4 to Flash memory space will go through the cache if the FCCR.CACHE_MODE_EN is set to 1.

If a hit occurs, then the data is returned immediately. If a miss occurs, then the Flash Cache generates a read request for a 32-byte line to the serial interface; the interface reads 32 bytes of data from the Flash memory and returns it to the Flash Cache. The Flash Cache returns this data to CM4. The Flash Controller Configuration Register (FCCR) can be configured by software to set the FLASHC_PAD_EN to 1 before the CPU uses the Flash Controller. Software can also configure different types of Read commands supported on WinBond Flash memory prior to issuing reads to Flash memory space.

Figure 31, Flash Controller Block Diagram, on page 140 shows an overall block diagram.

5.4.1 Block Diagram

Figure 31: Flash Controller Block Diagram



5.4.2 Modes

5.4.2.1 Cache Bypass Mode

Cache bypass mode is the default mode of operation and requires that both FCCR.CACHE_MODE_EN and FCCR.SRAM_MODE_EN to be cleared to 0. In this mode, the fetched code from Flash memory bypasses the 32 KB cache.

5.4.2.2 Cache Mode

Cache mode is enabled when software sets FCCR.CACHE_MODE_EN to 1 and FCCR.SRAM_EN to 0. In this mode, the content fetched from the Flash memory will go through the 32 KB cache memory.

5.4.2.3 SRAM Mode

SRAM mode is enabled when FCCR.SRAM_MODE_EN is set to 1. The 32 KB data RAM can be utilized as SRAM in this mode. When FCCR.SRAM_MODE_EN is set to 1, also set FCCR.CACHE_MODE_EN to 0.

5.4.2.4 Configuration Override

The Flash Controller provides the capability to configure various different Command Semantics supported by Flash device vendors, specifically for Read. There are 2 ways of configuring and building the necessary Read Commands through the Flash Control Configuration register (FCCR) and Flash Controller Configuration Register 2 (FCCR).

When FCCR2.USE_CFG_OVRD = 0 (default), the user or software can configure FCCR.CMD_ TYPE to achieve the necessary Read Command semantics for the target Flash device. In this case, the Flash Controller hardware configures the required Address clock cycles, Instruction Opcodes, and Dummy clock cycles required for a specific Read command type without software intervention.

When FCCR2.USE_CFG_OVRD = 1, software has the flexibility to configure the number of Address clock cycles, Dummy clock cycles, Instruction Opcodes, Read mode bit configuration, and clock cycles. When FCCR2.USE_CFG_OVRD = 1, all the contents and fields of FCCR2 are used by the Flash Controller to determine the command semantics. Software needs to ensure that complete configuration is provided.

5.4.3 Programming Notes

5.4.3.1 Switch From Flash Controller (memory mapped) to QSPI

Procedure to switch from Flash Controller (memory mapped) to QSPI when communicating with the Flash device.

The default Read Command type of Flash Controller is FCCR.CMD_TYPE=0x5, "Fast Read Dual I/O Continuous Read Mode". While operating the Flash Controller in this Read mode, the user must Exit the Continuous Read mode in order to switch to normal Read command.

To Exit Continuous Read Mode:

1. Set FCCR.CACHE_EN=0x0.
2. Set FCCR.CMD_TYPE=0xC.
3. Poll for FCSR.CONT_RD_MD_EXIT_DONE. Wait for this bit to become 1.
4. Write FCSR.CONT_RD_MD_EXIT_DONE=0x1 to clear.
5. Set FCCR.FLASH_PAD_EN=0x0. The QSPI interface can now be used to communicate with the Flash device.

5.4.3.2 Switch Back from QSPI to Flash Controller

Procedure to switch back from QSPI to Flash Controller when communicating with the Flash device.

1. Set FCCR.FLASH_PAD_EN=0x1.
2. Set FCCR.CMD_TYPE=0x5.
3. Set FCCR.CACHE_EN=0x1, if cache mode is desired.

The execution from Flash can now be resumed.

5.4.3.3 Retain State of Flash Cache in PM3

The cache inside the Flash Controller can retain state during PM3 for faster resume. Configuration is required.

The Flash Controller is reset when exiting PM3, during which the cache gets flushed and everything in the cache becomes invalid. To prevent this from happening, and to preserve the cache state:

1. PMU -> LOW_PWR_CTRL.CACHE_LINE_FLUSH = 0x0 (prevent/override cache flush).
2. Enter PM3.
3. Exit PM3 (reset happens).
4. Set FCCR.CACHE_LINE_FLUSH = 0x0.
5. PMU -> LOW_PWR_CTRL.CACHE_LINE_FLUSH = 0x1 (restore the default).

5.4.3.4 Cache Flush

The Flash Cache is flushed automatically upon POR and exiting from Low-Power Modes (PM3). However, software must ensure that FCCR.CACHE_LINE_FLUSH is set (1) when the Flash device content is modified through Erase or Program operations using the QSPI interface. Otherwise, outdated data might be present in the cache.

5.5 Register Description

See Appendix A.4.2, Flash Controller Registers, on page 506 for a detailed description of the registers.

6 General Purpose Input Output (GPIO)

6.1 Overview

The 88MW300/302 GPIO unit provides as many as 50 GPIO pins. All ports are brought out of the device using alternate function multiplexing.

The GPIO function can be multiplexed on a multi-function I/O pin by selecting the GPIO alternate function in the pad configuration registers (PINMUX) and configuring the GPIO internal registers.

> ***Note:*** Both the GPIO internal and PINMUX registers are accessed separately through the APB interface (different base addresses). See Section 2.4, Memory Map, on page 69 for base addressing.
>
> For GPIO internal registers, see, Appendix A.11, GPIO Address Block, on page 645.
>
> For PINMUX registers, see, Appendix A.17, PINMUX Address Block, on page 747.

6.2 I/O Configuration

Many of the 88MW300/302 package pins are multiplexed. They can be configured as general-purpose I/Os or an alternate function using the Multi-Function Pin Alternate-Function Select registers. Some functions can be configured to appear on 1 of several different pins using alternate function controls.

The I/O pad can support a 50 kΩ pull-up, 50 kΩ pull-down, or tri-state configuration. The default value of the I/O function is pull-up.

> ***Note:*** See Section 6.2.2, I/O Padding Pin States, on page 160 for detailed information regarding the I/O pins on each package.

6.2.1 PINMUX Alternate Functions

Each I/O or package pin has a dedicated register to control its functionality. The function number is specified from 0 to 7 by the least significant 3 bits of each register.

Function 0 is the GPIO function for all I/O pins, except:

- TDO/TCK/TMS/TDI/TRSTn (GPIO_6 to GPIO_10)
- WAKE_UP0/WAKE_UP1 (GPIO_22/23)
- OSC32K (GPIO_24)
- XTAL32K_IN/XTAL32K_OUT (GPIO_25/26)
- QSPI (GPIO_28 to GPIO_33)

After POR, PINMUX is Function 0 by default.

If Function 0 is a GPIO function, it has a 50 kohm pull-up, by default.

When Function 0 is not a GPIO function, the following pins also have a 50 kohm pull-up, by default:

- GPIO_6 to GPIO_10
- GPIO_30 to GPIO_33

The following tables show the alternate functions for each GPIO pin.

Table 60: GPIO_0 (Offset=0x00)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP0_CLK	I/O	SSP 0 Serial Clock
2	UART0_CTSn	I	UART 0 CTSn (active low)
1	GPT0_CH0	I/O	General Purpose Timer 0, Channel 0
0	GPIO_0	I/O	General Purpose I/O 0

Table 61: GPIO_1 (Offset=0x04)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP0_FRM	I/O	SSP 0 Frame Indicator
2	UART0_RTSn	O	UART 0 RTSn (active low)
1	GPT0_CH1	I/O	General Purpose Timer 0, Channel 1
0	GPIO_1	I/O	General Purpose I/O 1

Table 62: GPIO_2 (Offset=0x08)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP0_TXD	O	SSP 0 TXD
2	UART0_TXD	O	UART 0 TXD
1	GPT0_CH2	I/O	General Purpose Timer 0, Channel 2
0	GPIO_2	I/O	General Purpose I/O 2

Table 63: GPIO_3 (Offset=0x0C)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP0_RXD	I	SSP 0 RXD
2	UART0_RXD	I	UART 0 RXD
1	GPT0_CH3	I/O	General Purpose Timer 0, Channel 3
0	GPIO_3	I/O	General Purpose I/O 3

Table 64: GPIO_4 (Offset=0x10)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	AUDIO_CLK	O	Audio Clock AUPLL Audio clock output provided by Audio PLL for external codec.
2	I2C0_SDA	I/O	I^2C 0 SDA
1	GPT0_CH4	I/O	General Purpose Timer 0, Channel 4
0	GPIO_4	I/O	General Purpose I/O 4

Table 65: GPIO_5 (Offset=0x14)

Function	Name	I/O	Description
7:3	N/A	N/A	N/A
2	I2C0_SCL	I/O	I^2C 0 SCL
1	GPT0_CH5	I/O	General Purpose Timer 0, Channel 5
0	GPIO_5	I/O	General Purpose I/O 5

Table 66: GPIO_6 (Offset=0x18)

Function	Name	I/O	Description
7:3	N/A	N/A	N/A
2	I2C1_SDA	I/O	I^2C 1 SDA
1	GPIO_6	I/O	General Purpose I/O 6
0	TDO	O	JTAG Test Data

Table 67: GPIO_7 (Offset=0x1C)

Function	Name	I/O	Description
7:5	N/A	N/A	N/A
4	I2C0_SDA	I/O	I^2C 0 SDA
3	SSP2_CLK	I/O	SSP 2 Serial Clock
2	UART2_CTSn	I	UART 2 CTSn (active low)
1	GPIO_7	I/O	General Purpose I/O 7
0	TCK	I	JTAG Test Clock SWCLK in SWD Mode

Table 68: GPIO_8 (Offset=0x20)

Function	Name	I/O	Description
7:5	N/A	N/A	N/A
4	I2C0_SCL	I/O	I^2C 0 SCL
3	SSP2_FRM	I/O	SSP 2 Frame Indicator
2	UART2_RTSn	O	UART 2 RTSn (active low)
1	GPIO_8	I/O	General Purpose I/O 8
0	TMS	I/O	JTAG Controller Select SWDIO in SWD Mode

Table 69: GPIO_9 (Offset=0x24)

Function	Name	I/O	Description
7:5	N/A	N/A	N/A
4	I2C1_SDA	I/O	I^2C 1 SDA
3	SSP2_TXD	O	SSP 2 TXD
2	UART2_TXD	O	UART 2 TXD
1	GPIO_9	I/O	General Purpose I/O 9
0	TDI	I	JTAG Controller Select

Table 70: GPIO_10 (Offset=0x28)

Function	Name	I/O	Description
7:5	N/A	N/A	N/A
4	I2C1_SCL	I/O	I^2C 1 SCL
3	SSP2_RXD	I	SSP 2 RXD
2	UART2_RXD	I	UART 2 RXD
1	GPIO_10	I/O	General Purpose I/O 10
0	TRSTn	I	JTAG Test Reset (active low)

Table 71: GPIO_11 (Offset=0x2C)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP1_CLK	I/O	SSP 1 Serial Clock
2	UART1_CTSn	I	UART 1 CTSn (active low)
1	GPT2_CH0	I/O	General Purpose Timer 2, Channel 0
0	GPIO_11	I/O	General Purpose I/O 11

Table 72: GPIO_12 (Offset=0x30)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP1_FRM	I/O	SSP 1 Frame Indicator
2	UART1_RTSn	O	UART 1 RTSn (active low)
1	GPT2_CH1	I/O	General Purpose Timer 2, Channel 1
0	GPIO_12	I/O	General Purpose I/O 12

Table 73: GPIO_13 (Offset=0x34)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP1_TXD	O	SSP 1 TXD
2	UART1_TXD	O	UART 1 TXD
1	GPT2_CH2	I/O	General Purpose Timer 2, Channel 2
0	GPIO_13	I/O	General Purpose I/O 13

Table 74: GPIO_14 (Offset=0x38)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP1_RXD	I	SSP 1 RXD
2	UART1_RXD	I	UART 1 RXD
1	GPT2_CH3	I/O	General Purpose Timer 2, Channel 3
0	GPIO_14	I/O	General Purpose I/O 14

Table 75: GPIO_15 (Offset=0x3C)

Function	Name	I/O	Description
7:2	N/A	N/A	N/A
1	GPT2_CH4	I/O	General Purpose Timer 2, Channel 4
0	GPIO_15	I/O	General Purpose I/O 15

Table 76: GPIO_16 (Offset=0x40)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	AUDIO_CLK	O	Audio Clock AUPLL Audio clock output from Audio PLL for external codec.
2	N/A	N/A	N/A
1	CON[5]	I/O	Configuration Bit See Section 1.5, Configuration Pins, on page 65.
0	GPIO_16	I/O	General Purpose I/O 16

Table 77: GPIO_17 (Offset=0x44)

Function	Name	I/O	Description
7:3	N/A	N/A	N/A
2	I2C1_SCL	I/O	I^2C 1 SCL
1	GPT3_CH0	I/O	General Purpose Timer 3, Channel 0
0	GPIO_17	I/O	General Purpose I/O 17

Table 78: GPIO_18 (Offset=0x48)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP1_CLK	I/O	SSP 1 Serial Clock
2	I2C1_SDA	I/O	I^2C 1 SDA
1	GPT3_CH1	I/O	General Purpose Timer 3, Channel 1
0	GPIO_18	I/O	General Purpose I/O 18

Table 79: GPIO_19 (Offset=0x4C)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP1_FRM	I/O	SSP 1 Frame Indicator
2	I2C1_SCL	I/O	I^2C 1 SCL
1	GPT3_CH2	I/O	General Purpose Timer 3, Channel 2
0	GPIO_19	I/O	General Purpose I/O 19

Table 80: GPIO_20 (Offset=0x50)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP1_TXD	O	SSP 1 TXD
2	I2C0_SDA	I/O	I^2C 0 SDA
1	GPT3_CH3	I/O	General Purpose Timer 3, Channel 3
0	GPIO_20	I/O	General Purpose I/O 20

Table 81: GPIO_21 (Offset=0x54)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP1_RXD	I	SSP 1 RXD
2	I2C0_SCL	I/O	I^2C 0 SCL
1	GPT3_CH4	I/O	General Purpose Timer 3, Channel 4
0	GPIO_21	I/O	General Purpose I/O 21

Table 82: GPIO_22 (Offset=0x58)

Function	Name	I/O	Description
7:2	N/A	N/A	N/A
1	GPIO_22	I/O	General Purpose I/O 22
0	WAKE_UP0	I	Wake-Up 0

Table 83: GPIO_23 (Offset=0x5C)

Function	Name	I/O	Description
7:6	N/A	N/A	N/A
5	COMP_IN_P	I	LDO18 Comparator Input, Positive
4:3	N/A	N/A	N/A
2	UART0_CTSn	I	UART 0 CTSn (active low)
1	GPIO_23	I/O	General Purpose I/O 23
0	WAKE_UP1	I	Wake-Up 1

Table 84: GPIO_24 (Offset=0x60)

Function	Name	I/O	Description
7:6	N/A	N/A	N/A
5	COMP_IN_N	I	LDO18 Comparator Input, Negative
4	N/A	N/A	N/A
3	GPT1_CH5	I/O	General Purpose Timer 1, Channel 5
2	UART0_RXD	I	UART 0 RXD
1	GPIO_24	I/O	General Purpose I/O 24
0	OSC32K	O	32 kHz Output, Choose RC or Crystal

Table 85: GPIO_25 (Offset=0x64)

Function	Name	I/O	Description
7:3	N/A	N/A	N/A
2	I2C1_SDA	I/O	I^2C 1 SDA
1	GPIO_25	I/O	General Purpose I/O 25
0	XTAL32K_IN	I	32.768 kHz Crystal Input

Table 86: GPIO_26 (Offset=0x68)

Function	Name	I/O	Description
7:3	N/A	N/A	N/A
2	I2C1_SCL	I/O	I^2C 1 SCL
1	GPIO_26	I/O	General Purpose I/O 26
0	XTAL32K_OUT	O	32.768 kHz Crystal Output

Table 87: GPIO_27 (Offset=0x6C)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	USB_DRV_VBUS	O	Drive 5V on VBUS
2	UART0_TXD	O	UART 0 TXD
1	CON[4]	I/O	Configuration Bit See Section 1.5, Configuration Pins, on page 65.
0	GPIO_27	I/O	General Purpose I/O 27

Table 88: GPIO_28 (Offset=0x70)

Function	Name	I/O	Description
7:6	N/A	N/A	N/A
5	GPT1_CH0	I/O	General Purpose Timer 1, Channel 0
4:3	N/A	N/A	N/A
2	I2C0_SDA	I/O	I^2C 0 SDA
1	GPIO_28	I/O	General Purpose I/O 28
0	QSPI_SSn		QSPI Chip Select (active low)

Table 89: GPIO_29 (Offset=0x74)

Function	Name	I/O	Description
7:6	N/A	N/A	N/A
5	GPT1_CH1	I/O	General Purpose Timer 1, Channel 1
4:3	N/A	N/A	N/A
2	I2C0_SCL	I/O	I^2C 0 SCL
1	GPIO_29	I/O	General Purpose I/O 29
0	QSPI_CLK	O	QSPI Clock

Table 90: GPIO_30 (Offset=0x78)

Function	Name	I/O	Description
7:6	N/A	N/A	N/A
5	GPT1_CH2	I/O	General Purpose Timer 1, Channel 2
4	N/A	N/A	N/A
3	SSP0_CLK	I/O	SSP 0 Serial Clock
2	UART0_CTSn	I	UART 0 CTSn (active low)
1	GPIO_30	I/O	General Purpose I/O 30
0	QSPI_D0	I/O	QSPI Data 0

Table 91: GPIO_31 (Offset=0x7C)

Function	Name	I/O	Description
7:6	N/A	N/A	N/A
5	GPT1_CH3	I/O	General Purpose Timer 1, Channel 3
4	N/A	N/A	N/A
3	SSP0_FRM	I/O	SSP 0 Frame Indicator
2	UART0_RTSn	O	UART 0 RTSn (active low)
1	GPIO_31	I/O	General Purpose I/O 31
0	QSPI_D1	I/O	QSPI Data 1

Table 92: GPIO_32 (Offset=0x80)

Function	Name	I/O	Description
7:6	N/A	N/A	N/A
5	GPT1_CH4	I/O	General Purpose Timer 1, Channel 4
4	N/A	N/A	N/A
3	SSP0_TXD	O	SSP 0 TXD
2	UART0_TXD	O	UART 0 TXD
1	GPIO_32	I/O	General Purpose I/O 32
0	QSPI_D2	I/O	QSPI Data 2

Table 93: GPIO_33 (Offset=0x84)

Function	Name	I/O	Description
7:6	N/A	N/A	N/A
5	GPT1_CH5	I/O	General Purpose Timer 1, Channel 5
4	N/A	N/A	N/A
3	SSP0_RXD	I	SSP 0 RXD
2	UART0_RXD	I	UART 0 RXD
1	GPIO_33	I/O	General Purpose I/O 33
0	QSPI_D3	I/O	QSPI Data 3

Table 94: GPIO_34 (Offset=0x88)

Function	Name	I/O	Description
7:2	N/A	N/A	N/A
1	GPT3_CH5	I/O	General Purpose Timer 3, Channel 5
0	GPIO_34	I/O	General Purpose I/O 34

Table 95: GPIO_35 (Offset=0x8C)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP1_CLK	I/O	SSP 1 Serial Clock
2	UART1_CTSn	I	UART 1 CTSn (active low)
1	GPT0_CLKIN	I	General Purpose Timer 0, Clock Input
0	GPIO_35	I/O	General Purpose I/O 35

Table 96: GPIO_36 (Offset=0x90)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP1_FRM	I/O	SSP 1 Frame Indicator
2	UART1_RTSn	O	UART 1 RTSn (active low)
1	GPT1_CLKIN	I	General Purpose Timer 1, Clock Input
0	GPIO_36	I/O	General Purpose I/O 36

Table 97: GPIO_37 (Offset=0x94)

Function	Name	I/O	Description
7:3	N/A	N/A	N/A
2	UART0_RTSn	O	UART 0 RTSn (active low)
1	GPT2_CH5	I/O	General Purpose Timer 2, Channel 5
0	GPIO_37	I/O	General Purpose I/O 37

Table 98: GPIO_38 (Offset=0x98)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP1_TXD	O	SSP 1 TXD
2	UART1_TXD	O	UART 1 TXD
1	GPT2_CLKIN	I	General Purpose Timer 2, Clock Input
0	GPIO_38	I/O	General Purpose I/O 38

Table 99: GPIO_39 (Offset=0x9C)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP1_RXD	I	SSP 1 RXD
2	UART1_RXD	I	UART 1 RXD
1	GPT3_CLKIN	I	General Purpose Timer 3, Clock Input
0	GPIO_39	I/O	General Purpose I/O 39

Table 100: GPIO_40 (Offset=0xA0)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	ACOMP1_GPIO_OUT	O	ACOMP1 GPIO Output
2	ACOMP0_GPIO_OUT	O	ACOMP0 GPIO Output
1	ADC_DAC_TRIGGER0	I	ADC/DAC External Trigger 0
0	GPIO_40	I/O	General Purpose I/O 40

Table 101: GPIO_41 (Offset=0xA4)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	ACOMP1_EDGE_PULSE	O	ACOMP Edge Pulse 1
2	ACOMP0_EDGE_PULSE	O	ACOMP Edge Pulse 0
1	ADC_DAC_TRIGGER1	I	ADC/DAC External Trigger 1
0	GPIO_41	I/O	General Purpose I/O 41

Table 102: GPIO_42 (Offset=0xA8)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP1_CLK	I/O	SSP 1 Serial Clock
2	UART1_CTSn	I	UART 1 CTSn (active low)
1	ADC0_0 / ACOMP0 / TS_INP / VOICE_P	I	ADC0 Channel 0 ACOMP0 Channel 0 ACOMP1 Channel 0 Temperature sensor remote sensing positive input Voice sensing positive input
0	GPIO_42	I/O	General Purpose I/O 42

Table 103: GPIO_43 (Offset=0xAC)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP1_FRM	I/O	SSP 1 Frame Indicator
2	UART1_RTSn	O	UART 1 RTSn (active low)
1	ADC0_1 / ACOMP1 / DACB / TS_INN / VOICE_N	I/O	ADC0 Channel 1 ACOMP0 Channel 1 ACOMP1 Channel 1 Temperature sensor remote sensing negative input Voice sensing negative input
0	GPIO_43	I/O	General Purpose I/O 43

Table 104: GPIO_44 (Offset=0xB0)

Function	Name	I/O	Description
7	RF_CNTL1_P	O	WLAN Radio Control 1
6:4	N/A	N/A	N/A
3	SSP1_TXD	O	SSP 1 TXD
2	UART1_TXD	O	UART 1 TXD
1	ADC0_2 / ACOMP2 / DACA	I/O	ADC0 Channel 2 ACOMP0 Channel 2 ACOMP1 Channel 2 DAC Channel A output
0	GPIO_44	I/O	General Purpose I/O 44

Table 105: GPIO_45 (Offset=0xB4)

Function	Name	I/O	Description
7	RF_CNTL0_N	O	WLAN Radio Control 0
6:4	N/A	N/A	N/A
3	SSP1_RXD	I	SSP 1 RXD
2	UART1_RXD	I	UART 1 RXD
1	ADC0_3 / ACOMP3 / EXT_VREF	I	ADC0 Channel 3 ACOMP0 Channel 3 ACOMP1 Channel 3 ADC or DAC external voltage reference input
0	GPIO_45	I/O	General Purpose I/O 45

Table 106: GPIO_46 (Offset=0xB8)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP2_CLK	I/O	SSP 2 Serial Clock
2	UART2_CTSn	I	UART 2 CTSn (active low)
1	ADC0_4 / ACOMP4	I	ADC0 Channel 4 ACOMP0 Channel 4 ACOMP1 Channel 4
0	GPIO_46	I/O	General Purpose I/O 46

Table 107: GPIO_47 (Offset=0xBC)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP2_FRM	I/O	SSP 2 Frame Indicator
2	UART2_RTSn	O	UART 2 RTSn (active low)
1	ADC0_5 / ACOMP5	I	ADC0 Channel 5 ACOMP0 Channel 5 ACOMP1 Channel 5
0	GPIO_47	I/O	General Purpose I/O 47

Table 108: GPIO_48 (Offset=0xC0)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP2_TXD	O	SSP 2 TXD
2	UART2_TXD	O	UART 2 TXD
1	ADC0_6 / ACOMP6	I	ADC0 Channel 6 ACOMP0 Channel 6 ACOMP1 Channel 6
0	GPIO_48	I/O	General Purpose I/O 48

Table 109: GPIO_49 (Offset=0xC4)

Function	Name	I/O	Description
7:4	N/A	N/A	N/A
3	SSP2_RXD	I	SSP 2 RXD
2	UART2_RXD	I	UART 2 RXD
1	ADC0_7 / ACOMP7	I	ADC0 Channel 7 ACOMP0 Channel 7 ACOMP1 Channel 7
0	GPIO_49	I/O	General Purpose I/O 49

6.2.2 I/O Padding Pin States

The I/O padding can be configured to pull-up, pull-down, or tri-state mode. The I/O pins are configured to default mode when selecting 1 PINMUX alternate function. When an I/O pin is set to a GPIO function and the data transfer direction is input, users can reconfigure the I/O pin to pull-up, pull-down, or tri-state mode by setting bits[15:13] and bit[3] of the corresponding I/O PINMUX Configuration register. See Appendix A.17.2, PINMUX Registers (_GPIO), on page 749.

Table 110, I/O Pin Mode Configuration shows the configuration.

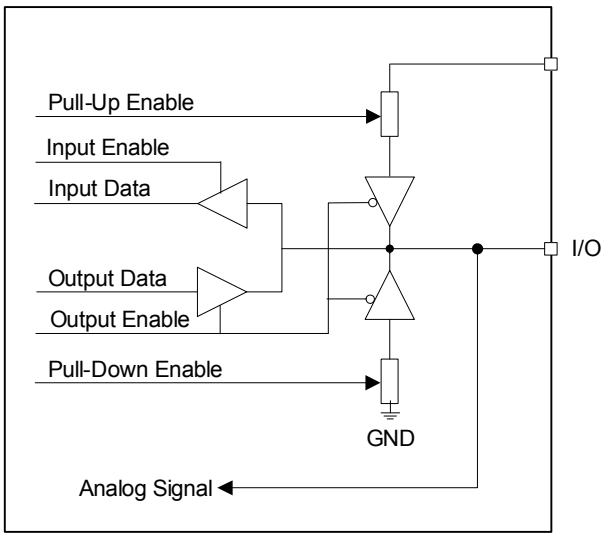
Table 110: I/O Pin Mode Configuration[1]

Bit Field of I/O PINMUX Configuration Register				Description
[15]	[14]	[13]	[3]	
0	X	X	1	Pull-up and pull-down from PINMUX alternate function
1	1	0	1	Pull-up enabled
1	0	1	1	Pull-down enabled
1	1	1	1	Not allowed
1	0	0	0	Tri-state

1. X = don't care

Figure 32 shows the I/O padding structure.

Figure 32: I/O Padding Structure

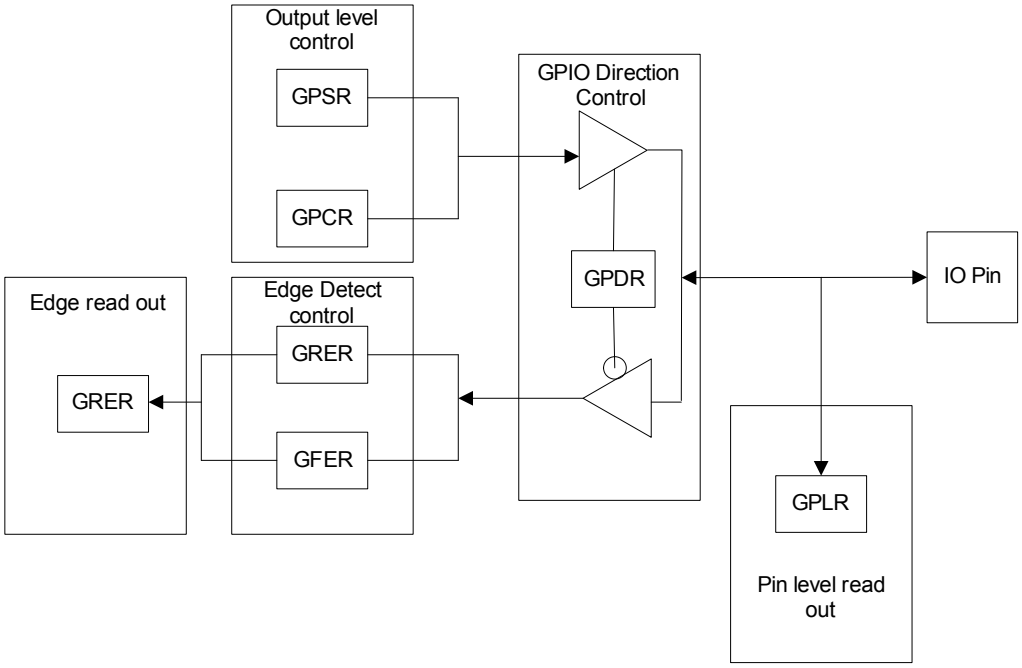


6.3 Functional Description

6.3.1 Block Diagram

Figure 33 shows an overall block diagram.

Figure 33: General Purpose I/O Block Diagram



6.3.2 GPIO Ports

The GPIO pins are mapped to port groups:

- GPIO_PORT0 – 32 pins (correspond to GPIO[31:0])
- GPIO_PORT1 – 18 pins (correspond to GPIO[49:32])

Individual GPIO pins within a port are numbered from 0 to 31 according to their bit positions within the GPIO registers.

6.3.3 I/O Control

All the GPIO pins are inputs by default. The direction of the GPIO pins is configured by programming the GPIO Pin Direction Register (GPDR (0, 1) registers) or the GPIO Set Direction Register-GSDR and GPIO Clear Direction Register-GCDR (0, 1) registers through the APB interface. When the system is in low-power mode, the input enable to the GPIOs should be disabled by setting the di_en (bit[3]) bit of the corresponding GPIO Configuration register in the PINMUX.

When configured as an input, GPIO can read the data on the external I/O pads and also serve as an interrupt. Interrupts are generated when the GPIO Rising Edge detect enable -GRER (0, 1) or GPIO Falling edge detect enable-GFER (0, 1) registers are configured. Rising and falling edges are detected using APB Clock Synchronized GPIO inputs. The status of edge detection can be read through the GPIO edge detect status-GEDR (0, 1) registers. The edge detect status is used to generate a combined interrupt from the GPIO block. Specific GPIO interrupts can also be masked by programming the APMASK register.

When configured as an output, the GPSR and GPCR (0, 1) registers are programmed to define the GPIO output port status.

The value of each GPIO port can be read through the GPIO Pin Level register-GPLR (read only) when the GPIO is configured as an input or output. This register can be read at any time to confirm the port state for the input configuration.

6.3.4 GPIO Interrupt

The GPIOs can be programmed to accept external signals as interrupt sources on any bit of the signal. The type of interrupt is programmable with either a rising or falling edge.

When the GPIO is configured as an output, it requires a few GPIO clock cycles for the value to be updated in the GPLR register. The number of cycles required for updating the value depends on the CORE and GPIO clock frequencies and is specified as follows:

- Core runs at 32M and GPIO runs at 32M: After writing to GPSR, correct GPLR value is reflected in 3 GPIO clock cycles
- Core runs at 200M and GPIO runs at 50M: After writing to GPSR, correct GPLR value is reflected in 5 GPIO clock cycles

When the GPIO is configured as an input, this register is updated with the current level of the GPIO input after 12 cycles of the 200 MHz clock in the system.

The interrupts can be masked by programming the APMASK register. The interrupt status can be read before masking (called raw status) and after masking. A single combined interrupt is generated as output from the GPIO. All individual edges detected (as recorded in the GEDR registers) have to be masked, to mask the interrupt output. If the pin direction register is reprogrammed to output, then any pending interrupts are not lost. However, no new interrupts are generated.

When an edge is detected on a port that matches the type of edge programmed in the GRER and/or GFER registers, the corresponding status bit is set in GEDR registers. GEDR register value is updated with the current edge-detect status value after 12 cycles of the 200 MHz clock in the system from the occurrence of the edge.

6.3.5 External Interrupts

The 88MW300/302 external interrupt sources are directly connect to the Cortex-M4F NVIC module, an external interrupt pin can be used simultaneously by a peripheral device. All external interrupts are active high.

See Section 2.5, External Interrupts, on page 74 for an external interrupt mapping table for peripheral interrupts and interrupts from the GPIO.

6.4 Register Description

There are a total of 42 registers in the GPIO register block. For each 32 bits forming a GPIO port, there is a set of 14 registers. For up to 50 GPIOs, 3 GPIO_ports are defined. There are 3 instances of each of the 14 registers.

See Appendix A.11.2, GPIO Registers, on page 647 for a detailed description of the registers.

THIS PAGE INTENTIONALLY LEFT BLANK

7 WLAN

7.1 Overview

The 88MW300/302 integrates a highly integrated, single-band (2.4 GHz) IEEE 802.11n 1x1 WLAN subsystem, specifically designed to support next generation, high throughput data rates.

The subsystem provides the combined functions of CPU, memory, Medium Access Controller (MAC), Direct Sequence Spread Spectrum (DSSS) and Orthogonal Frequency Division Multiplexing (OFDM) baseband modulation, direct conversion WLAN RF radio, and encryption. For security, the 802.11i security standard is supported through several protocols.

7.2 Features

- 1x1 SISO, 2.4 GHz, HT20 operation
- Antenna diversity
- CMOS and low-swing sine wave input clock
- Low power with deep sleep and standby modes
- Pre-regulated supplies
- Integrated T/R switch, PA, and LNA
- Optional 802.11n features
- One Time Programmable (OTP) memory to eliminate need for external EEPROM

7.3 WLAN MAC

- Simultaneous peer-to-peer and Infrastructure Modes
- RTS/CTS for operation under DCF
- Hardware filtering of 32 multicast addresses
- On-chip Tx and Rx FIFO for maximum throughput
- Open System and Shared Key Authentication services
- A-MPDU Rx (de-aggregation) and Tx (aggregation)
- Reduced Inter-Frame Spacing (RIFS) receive
- Management information base counters
- Radio resource measurement counters
- Quality of service queues
- Block acknowledgement extension
- Multiple-BSSID and Multiple-Station operation
- Transmit rate adaptation
- Transmit power control
- Long and short preamble generation on a frame-by-frame basis for 802.11b frames
- Marvell mobile hotspot

7.4 WLAN Baseband

- 802.11n 1x1 SISO (on-chip Marvell RF radio)
- Backward compatibility with legacy 802.11g/b technology
- PHY data rates up to 72.2 Mbps
- 20 MHz bandwidth/channel
- Modulation and Coding Scheme (MCS)—MCS 0~7
- Radio resource measurement
- Optional 802.11n SISO features:
  - 1 spatial stream STBC reception and transmission
  - Short guard interval
  - RIFS on receive path for 802.11n packets
  - 802.11n greenfield Tx/Rx
- Power save features

7.5 WLAN Radio

The 88MW300/302 direct conversion WLAN RF radio integrates all the necessary functions for transmit and receive operation. See Section 22.6, Clock Specifications, on page 327 and Section 22.12, WLAN Radio Specifications, on page 344 for associated electrical specifications. Features include:

- Integrated direct-conversion radio
- 20 MHz channel bandwidth
- Integrated T/R switch, PA, and LNA

7.5.1 WLAN Rx Path

- Direct conversion architecture eliminates need for external SAW filter
- On-chip gain selectable LNA with optimized noise figure and power consumption
- High dynamic range AGC function in receive mode

7.5.2 WLAN Tx Path

- Integrated power amplifier with power control
- Optimized Tx gain distribution for linearity and noise performance

7.5.3 WLAN Local Oscillator

- Fractional-N for multiple reference clock support
- Fine channel step

7.5.4 Channel Frequencies Supported

Table 111 shows the channel frequencies supported.

Table 111: Channel Frequencies Supported

20 MHz Channels	
Channel	**Frequency (GHz)**
1	2.412
2	2.417
3	2.422
4	2.427
5	2.432
6	2.437
7	2.442
8	2.447
9	2.452
10	2.457
11	2.462
12	2.467
13	2.472

7.6 WLAN Encryption

- WEP 64- and 128-bit encryption with hardware TKIP processing (WPA)
- AES-CCMP hardware implementation as part of 802.11i security standard (WPA2)
- Enhanced AES engine performance
- AES-Cipher-Based Message Authentication Code (CMAC) as part of the 802.11w security standard
- WLAN Authentication and Privacy Infrastructure (WAPI)

THIS PAGE INTENTIONALLY LEFT BLANK

8 Direct Memory Access (DMA) Controller

8.1 Overview

The 88MW300/302 Direct Memory Access Controller (DMAC) is an AMBA High Speed Bus (AHB) system level controller used to transfer data between peripherals and memory as well as memory to memory without CPU action.

The data stream is maintained in 32 DMA channels. Each channel is required for each source/destination pair. The DMAC has 1 AHB master interface and 1 AHB slave interface. The master interface reads the data from a source peripheral and writes the data to a destination peripheral. There are 2 AHB transfers required for each DMA data transfer.

8.2 Features

- Compliance to the AMBA specification for easy integration
- Memory-to-memory, memory-to-peripheral, and peripheral-to-memory transfers
- Priority mechanism to process active channels
- 32 DMA logic channels (each channel can support a unidirectional transfer)
- 16x32 bits physical channel space to store the data
- 64 hardware handshake interfaces support for on-chip peripherals
- Programmable data-burst size (4, 8, and 16) and programmable peripheral device data widths (byte, half-word or word)
- Up to 8191 bytes of data transfer
- AHB slave DMA programming interface (program DMAC by writing to DMA control registers over AHB slave interface)
- Separate and combined DMA interrupt requests (generate an interrupt to the processor on a DMA error or when a DMA transfer has completed). Interrupt request signals include:
  - BLOCK signals when a block transfer has completed.
  - TFR signals when a transfer has completed.
  - BUSERR signals when a bus error has occurred.
  - ADDRERR signals when a peripheral address alignment error has occurred.
- Interrupt masking (mask each individual DMA interrupt request)

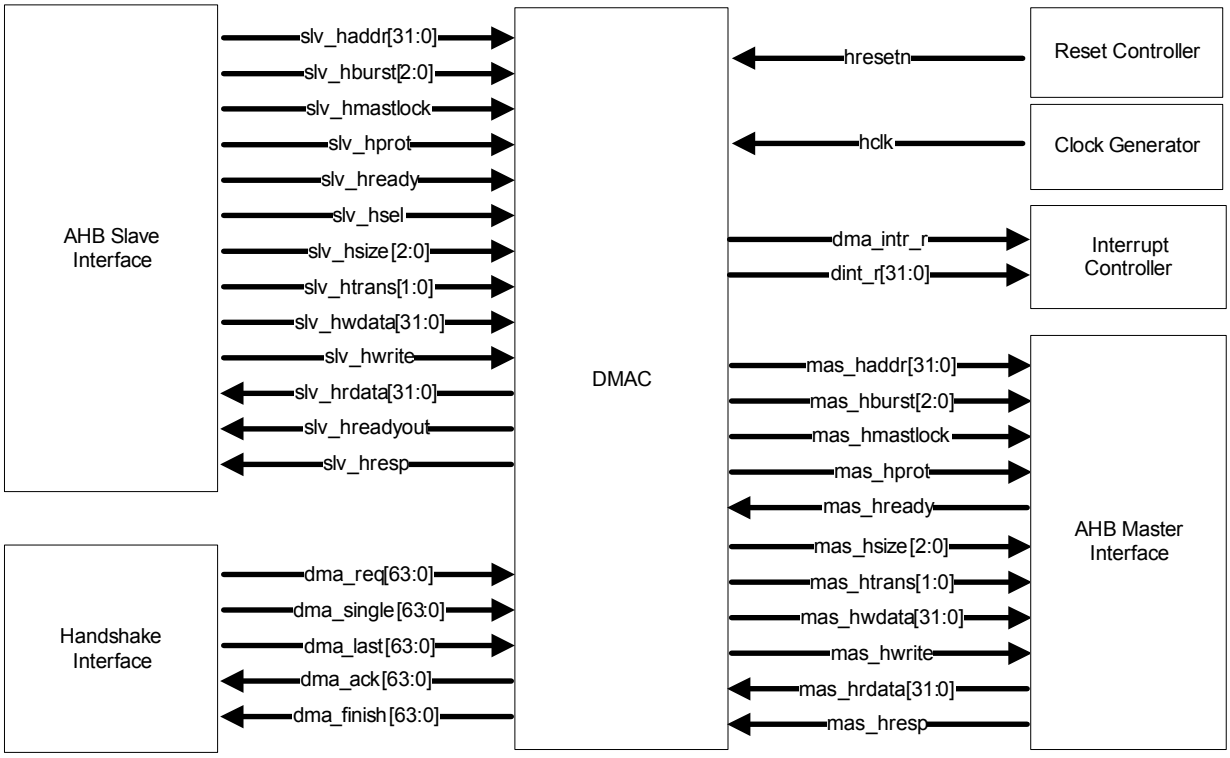
8.3 Basic Definitions

- Source peripheral – Device from which the DMAC reads data; the DMAC then stores the data in the channel FIFO.
- Destination peripheral – Device to which the DMAC writes the data from the FIFO (previously read from the source peripheral).
- Memory – Source or destination that is always "ready" for a DMA transfer and does not require a handshaking interface to interact with the DMAC.
- Channel – Read/Write data path between a source peripheral and a destination peripheral that occurs through the channel FIFO. If the source peripheral is not memory, then a source handshaking interface is asserted to the channel. If the destination peripheral is not a memory, then a destination handshaking interface is asserted to the channel. Source and destination handshaking interfaces can be assigned dynamically by programming the channel registers.
- Master interface – DMAC is a master on the AHB bus, reading data from source and writing it to the destination over the AHB bus.
- Slave interface – The AHB interface over which the DMAC is programmed.
- Handshaking interface – A set of signals that conform to a protocol and handshake between the DMAC and source or destination peripheral in order to control transferring a single or burst transaction between them. This interface is used to request, acknowledge, and control a DMAC transaction.
- Flow controller – Device that determines the length of a DMA block transfer and terminates it.
- Block – Block of DMAC data, the amount of which is the block length and is determined by the flow controller. For transaction, a block is either a burst or single transfers.
- Single transaction – Length of a single transaction is always 1 and is converted to a single AHB transfer.
- Burst transaction – Length of a burst transaction is programmed into the DMAC. The burst transaction is converted into a sequence of AHB burst transfers. The burst transaction length is under program control and normally bears some relationship to the FIFO size in the DMAC and in the source and destination peripheral.

8.4 Interface Signal Description

8.4.1 Internal Signals Diagram

Figure 34: DMAC Internal Signals



8.4.2 Clock and Reset Interface

Table 112: Clock Unit Interface Signals

Signal	Type	Source	Description
hclk	input	Clock generator	System (AHB) bus clock
hresetn	input	Reset controller	Power-on reset that resets the logic running off of hclk. Asynchronous assertion and de-assertion. Clocks are turned off during de-assertion.

8.4.3 Interface to Handshake Interface

Table 113: Handshake Interface Signals

Signal	Type	Source	Description
dma_request[63:0]	input	DMA peripheral	Burst transaction request from peripheral
dma_single[63:0]	input	DMA peripheral	Single transaction request from peripheral
dma_last[63:0]	input	DMA peripheral	Last transaction in block indicator
dma_ack[63:0]	output	DMA peripheral	Transaction complete acknowledge signal
dma_finish[63:0]	output	DMA peripheral	DMA block complete signal

8.4.4 DMA Interrupt Request Interface

Table 114: DMA Interrupt Request Interface Signals

Signal	Type	Source	Description
dma_intr_r	output	Interrupt controller	Logic OR of all individual channel interrupts
dint_r[31:0]	output	Interrupt controller	Logic OR of all types inner interrupts within each channel

8.4.5 AHB Slave Interface

Table 115: AHB Slave Interface Signals

Signal	Type	Source	Description
slv_haddr[31:0]	input	AHB master	System address bus
slv_hburst[2:0]	input	AHB master	Burst type indicates if the transfer is a single or forms part of a burst.
slv_hmastlock	input	AHB master	When high, the current transfer is part of a locked sequence.
slv_hprot	input	AHB master	The protection control signals provide additional information about a bus access and primarily intended for use by any module that wants to implement some level of protection.
slv_hready	input	AHB master	When HIGH, indicates to the master and all slaves that the previous transfer is complete.
slv_hsel	input	Decoder	Current transfer is intended for the selected slave.
slv_hsize[2:0]	input	AHB master	Size of the transfer
slv_htrans[1:0]	input	AHB master	Transfer type of the current transfer This can be: IDLE, BUSY, NONSEQUENTIAL and SEQUENTIAL.
slv_hwdata[31:0]	input	AHB master	The write data bus transfer data from the master to the slave during write operations.
slv_hwrite	input	AHB master	Transfer direction When HIGH this signal indicates a write transfer and when LOW a read transfer.

Table 115: AHB Slave Interface Signals (Continued)

Signal	Type	Source	Description
slv_hrdata	output	AHB master	The read data bus transfers data from bus slaves to bus master during read operations.
slv_hreadyout	output	AHB master	When HIGH, a transfer has finished on the bus. This signal can be driven LOW to extend a transfer.
slv_hresp	output	AHB master	The transfer response provides additional information on the status of a transfer.

8.4.6 AHB Master Interface

Table 116: AHB Master Interface Signals

Signal	Type	Source	Description
mas_haddr[31:0]	output	AHB slave	System address bus
mas_hburst[2:0]	output	AHB slave	Burst type indicates if the transfer is a single or forms part of a burst.
mas_hmastlock	output	AHB slave	When high, the current transfer is part of a locked sequence.
mas_hprot	output	AHB slave	The protection control signals provide additional information about a bus access and primarily intended for use by any module that wants to implement some level of protection.
mas_hsize[2:0]	output	AHB slave	Size of the transfer
mas_htrans[1:0]	output	AHB slave	Transfer type of the current transfer This can be: IDLE, BUSY, NONSEQUENTIAL and SEQUENTIAL.
mas_hwdata[31:0]	output	AHB slave	Write data bus transfer data from the master to the slave during write operations
mas_hwrite	output	AHB slave	Transfer direction When HIGH this signal indicates a write transfer and when LOW a read transfer.
mas_hrdata	input	AHB slave	The read data bus transfers data from bus slaves to bus master during read operations.
mas_hready	input	AHB slave	When HIGH, a transfer has finished on the bus. This signal can be driven LOW to extend a transfer.
mas_hresp	input	AHB slave	The transfer response provides additional information on the status of a transfer.

8.5 Functional Description

8.5.1 32 Channels and Priorities

8.5.1.1 DMA Channels

The DMAC uses 32 channels to manage the stream transfer. Each of the 32 DMA channels can be controlled by 4, 32-bit registers: SADRx, TADRx, CTRLAx, and CTRLBx.

Each channel is serviced in increments of that device burst size and delivered in the granularity of the device port width. The burst size and port width for each device are programmed in the channel registers and are based on the device FIFO depth and bandwidth requirements. When multiple channels are actively executing, each channel is serviced with a burst of data. After each burst of data, the DMA controller performs a context switch to another active channel. The DMAC performs context switches based on whether a channel is active, whether its device is currently requesting service, and the channel priority.

8.5.1.2 DMA Channel-Priority Scheme

The DMA channel-priority scheme helps to ensure that peripherals are serviced according to their bandwidth requirements. Assign a higher priority to peripherals with higher bandwidth requirements and a lower priority to peripherals with lower bandwidth requirements. This assignment ensures that higher-bandwidth peripherals are services more often than lower-bandwidth peripherals.

The DMA channels are divided internally into 4 sets of 8 channels each. The channels in each set use a round-robin priority. Set 0 has the highest priority, and Set 3 has the lowest. Program the modules that have the most severe latency requirements into Set 0. See Table 117.

When all 32 channels are running concurrently, Set 0 is serviced 4 out of every 8 consecutive channel-servicing instances, Set 1 is serviced twice, and Set 2 and 3 are each serviced once.

For example, if all the channels request data transfers, the sets are prioritized in the following order: Set 0, Set 1, Set 0, Set 2, Set 0, Set 1, Set 0, Set 3. After 8 channel-servicing instances, the pattern repeats. The channels in each set are given a round-robin priority.

Table 117 shows the channel priority.

Table 117: Channel Priority

Set	Channels	Priority	Number of Times Served
0	0, 1, 2, 3, 16, 17, 18, 19	Highest	4/8
1	4, 5, 6, 7, 20, 21, 22, 23	Higher than 2 and 3, lower than 0	2/8
2	8, 9, 10, 11, 24, 25, 26, 27	Higher than 3, lower than 0 and 1	1/8
3	12, 13, 14, 15, 28, 29, 30, 31	Lowest	1/8

8.5.2 Enable/Stop Channel

8.5.2.1 Enable a Channel

Program CHL_ENx = 1'b1 to enable a channel. Programming DMA_CHL_ENx = 1'b0 is not recommended.

CHL_ENx will be cleared after the channel has finished all expected transfers. To stop a channel which has not finished all transfers, see Section 8.5.2.2, Stop a Running Channel.

A channel cannot be enabled to initiate a transfer if the channel interrupt is raised. It can only be enabled again after programming the corresponding interrupt (clear it by writing 1).

8.5.2.2 Stop a Running Channel

To stop a channel that has not finished all transfers, program CHL_STOPx = 1'b1. Channelx will stop immediately if the channel is not on service, and stop after the current transfer finished if the channel is on service. Poll the corresponding CHL_ENx to check whether the stop operation is successful since the operation may not be immediate.

After CHL_ENx is in a disabled state, CHL_STOPx is automatically cleared by hardware.

Programming CHL_STOPx = 1'b0 has no effect on the channel, which is already in a disabled state.

8.5.3 Transfer Type and Flow Controller

The DMAC transfer types include:

- Memory-to-Memory
- Memory-to-Peripheral
- Peripheral-to-Memory

The transfer type is programmed in the channel CTRLAx register based on the different source and target device type.

The DMAC is always assigned as the flow controller. The block size is known before the channel is enabled and should be programmed into the CTRLAx.LEN field. Peripherals may not be used as a flow controller.

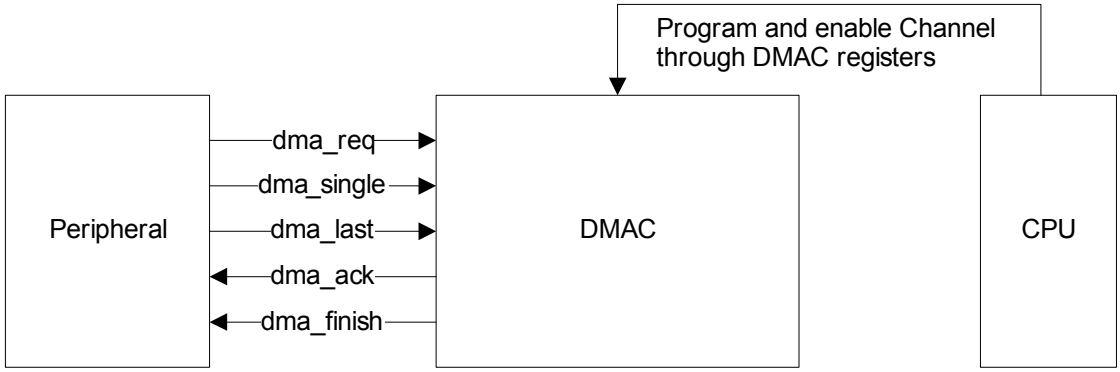
8.5.4 Hardware Handshaking Interface

Handshaking interfaces are used at the transaction level to control the flow of single or burst transactions. Hardware handshaking is supported and accomplished using a dedicated handshaking interface.

Generally, the DMAC tries to transfer the data using burst transactions and, where possible, fill or empty the channel FIFO in single burst. In the latter case, the peripheral asserts a single status flag to indicate to the DMAC that there is enough data or space to complete a single transaction from or to the source/destination peripheral.

Figure 35 shows the hardware handshaking interface between a peripheral (whether a destination or source) and the DMAC when the DMAC is the flow controller.

Figure 35: Hardware Handshaking Interface



There are some cases where a DMA block transfer cannot complete using only burst transactions. Typically this occurs when the block size is not a multiple of the burst transaction length. In these cases, the block transfer uses burst transactions up to the point where the amount of data left to complete the block is less than the amount of data in a burst transaction. At this point, the DMAC samples the "single" status flag and completes the block transfer using single transactions.

The single transaction region is the time interval where the DMAC uses single transactions to complete the block transfer. Burst transactions are exclusively used outside this region.

Table 118 shows the hardware handshaking signals.

Table 118: Hardware Handshaking Interface Signals

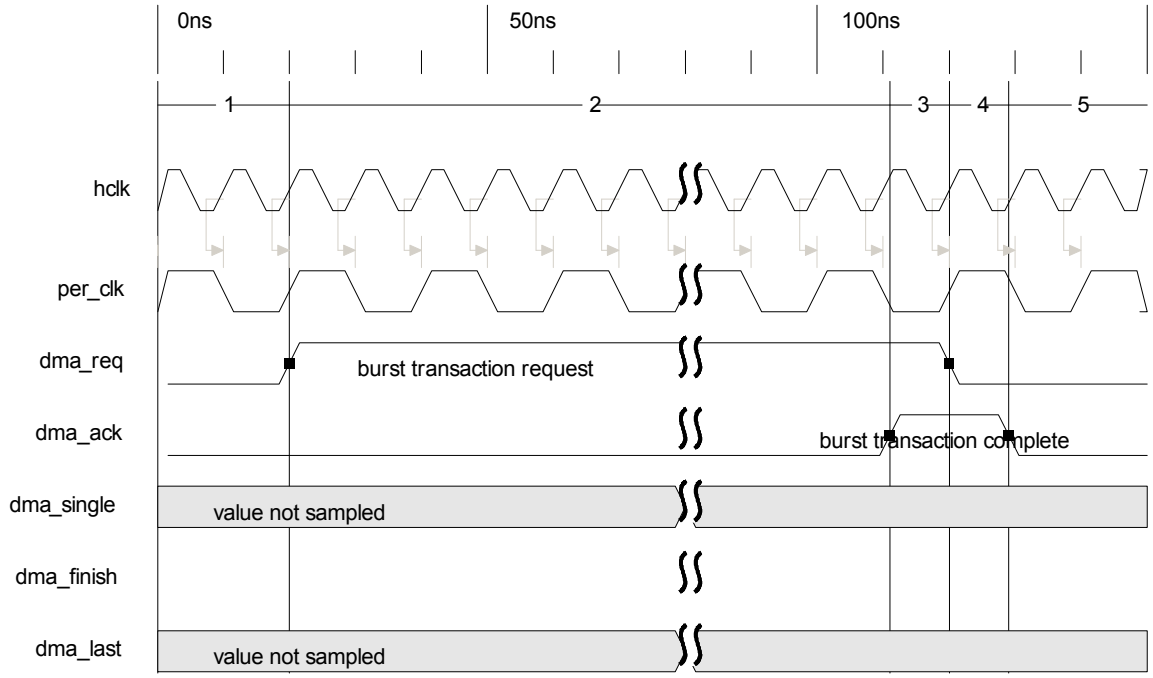
Signal	Type	Description
dma_ack	output	DMAC acknowledge signal to peripheral The dma_ack signal is asserted after the data phase of the last AHB transfer in the current transaction -single or -burst - to the peripheral that has completed. For a single transaction, dma_ack remains asserted until the peripheral de-asserts dma-single; dma_ack is de-asserted 1 hclk cycle later. For a burst transaction, dma_ack remains asserted until the peripheral de-asserts dma_req; dma_ack is de-asserted 1 hclk cycle later.
dma_finish	output	DMAC asserts dma_finish to signal block completion This has the same timing as dma_ack and forms a handshaking loop with dma_req is the last transaction in the block was a burst transaction, or with dma_single if the last transaction in the block was a single transaction.
dma_last	input	Since the peripheral is not the flow controller, dma_last is not sampled by the DMAC and this signal is ignored.
dma_req	input	Burst transaction request form peripheral The DMAC always interprets the dma_req signal as a burst transaction request, regardless of the level of the dma_single. This is a level-sensitive signal; once asserted by the peripheral, dma_req must remain asserted until the DMAC asserts dma_ack. Upon receiving the dma_ack signals from the DMAC to indicate the burst transaction is complete, the peripheral should de-asserted the burst request signal dma_req. Once dma_req is de-asserted by the peripheral, the DMAC de-asserts dma_ack. This signal is sampled by DMAC only outside the single transaction region.

Table 118: Hardware Handshaking Interface Signals (Continued)

Signal	Type	Description
dma_s	input	Single transfer status The dma_single signal is a status signal that is asserted by a destination peripheral when it can accept at least 1 destination data item; otherwise it is cleared. For a source peripheral, the dma_ single signal is again a status signal and is asserted by a source peripheral when it can transmit at least 1 source data item; otherwise it is cleared. once asserted, dma_single must remain asserted until dma_ack is asserted, at which time the peripheral should de-asserted dma_single. This signal is sampled by DMAC only in the single transaction region.

Figure 36 shows the timing diagram of a burst transaction where the peripheral clock (per_clk) is half of the hclk frequency. In this example, the peripheral is outside the single transaction region, and the DMAC does not sample dma_single.

Figure 36: Burst Transaction – hclk=2*per_clk

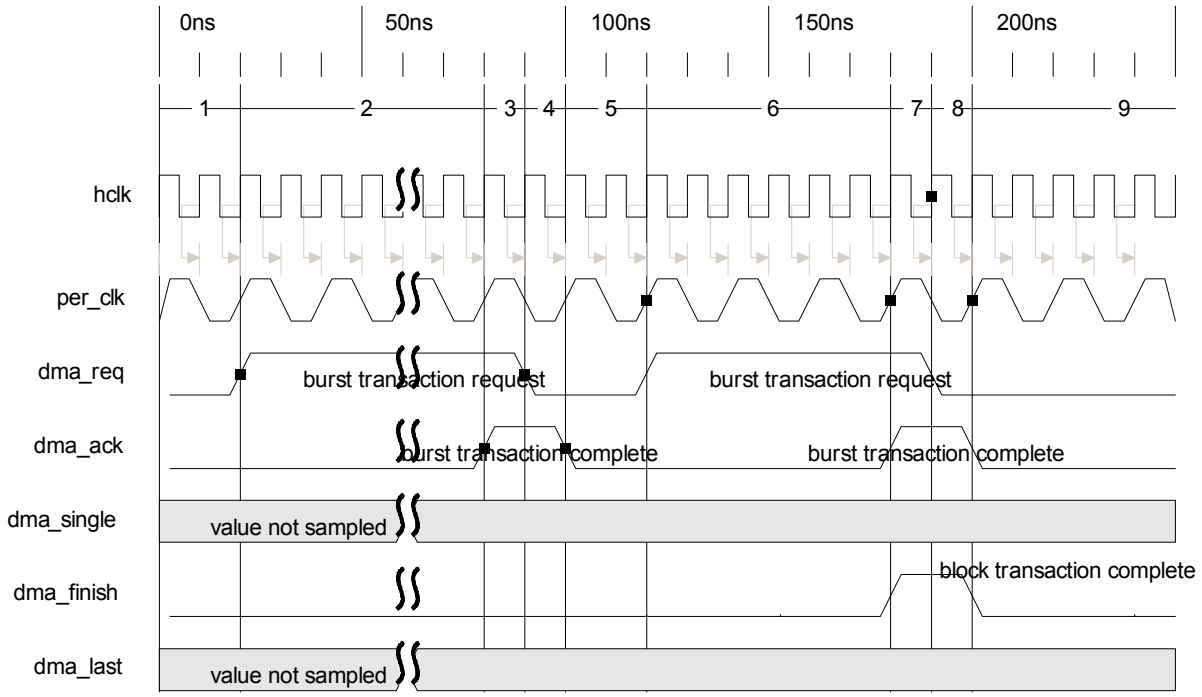


The handshaking loop is as follows:

- dma_req asserted by peripheral
- dma_ack asserted by DMAC
- dma_req de-asserted by peripheral
- dma_ack de-asserted by DMAC

Figure 37 shows 2 back-to-back burst transactions at the end of a block transaction where the hclk frequency is twice the pclk frequency. The peripheral is an APB peripheral. The second burst transaction terminates the block, and dma_finish is asserted to indicate block completion.

Figure 37: Back-to-Back Burst Transaction – hclk=2*per_clk

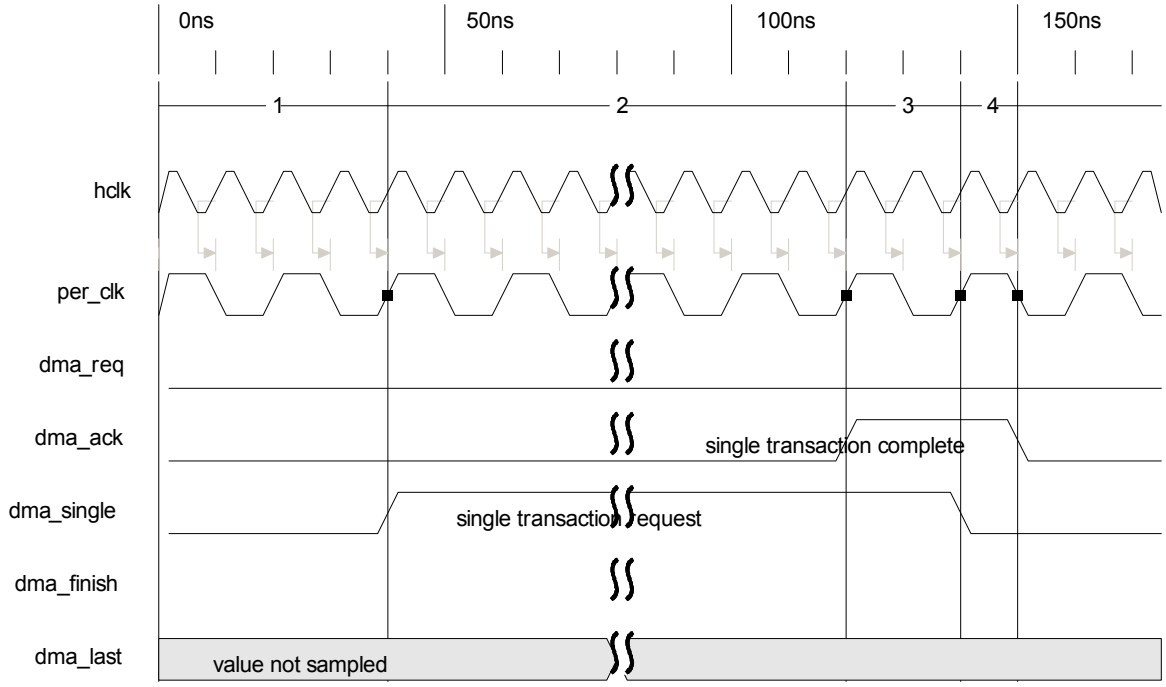


When designing the hardware handshaking interface, note that:

- Once asserted, the dma_req must remain asserted until the corresponding dma_ack signal is received, even if the condition that generates dma_req in the peripheral is False.
- The dma_req signal should be de-asserted when dma_ack is asserted, even if the condition that generates dma_req in the peripheral is true.

Figure 38 shows a single transaction that occurs in the single transaction region.

Figure 38: Single Transaction – hclk=2*per_clk

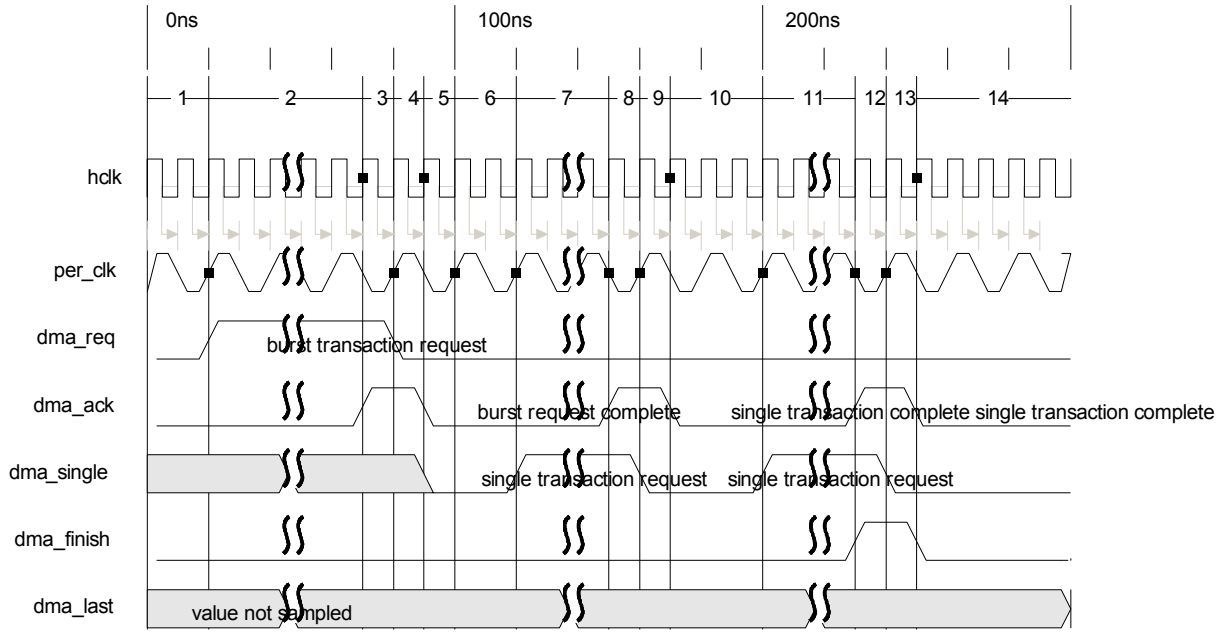


The handshaking loop is as follows:

- dma_single asserted by peripheral
- dma_ack asserted by DMAC
- dma_single de-asserted by peripheral
- dma_ack de-asserted by DMAC

Figure 39 shows a burst transaction, followed by 2 back-to-back single transactions, where the hclk frequency is twice the per_clk frequency.

Figure 39: Burst Followed by Back-to-Back Single Transactions



After the first burst transaction, the peripheral enters the single transaction region and the DMAC starts sampling dma_single. DMAC samples that dma_single is asserted and performs single transactions. The second single transaction terminates the block transfer; dma_finish is asserted to indicate block completion.

8.5.5 Interrupt Management

There are 4 types of interrupt source:

- BLOCKINT – Block Transfer Complete Interrupt
  This interrupt is generated on DMA block transfer completion to the destination peripheral
- TFRINT – DMA Transfer Complete Interrupt
  This interrupt is generated on DMA transfer completion to destination peripheral.
- BUSERROR – AHB Bus Error Interrupt
  This interrupt is generated when an error response is received from AHB slave.
- ADDRERRINT – Address Configuration Error Interrupt
  This interrupt is generated when the on-chip peripheral address is not aligned to width.

There are several groups of interrupt-related registers:

- MASK_BLOCKINT, MASK_TFRINT, MASK_BUSERRINT, MASK_ADDRERRINT
- STATUS_BLOCKINT, STATUS_TFRINT, STATUS_BUSERRINT, STATUS_ADDRERRINT
- STATUS_CHLINT

When a channel has been enabled to generated interrupts:

- Interrupt events are stored in the STATUS registers
- Contents of the STATUS registers are masked with the contents of MASK registers
- Masked interrupts are stored in the STATUS_CHLINT register
- Contents of STATUS_CHLINT are used to drive dma_int_r and dint_r
- Writing to the appropriate bit in STATUS registers clears an interrupt in STATUS registers
- Each mask and status interrupt register has a bit allocated per channel

8.5.6 Transfer Operation Example

Transfers are set up by programming registers for that channel. As shown in Figure 2 6, a single block is made up of numerous transactions -single and burst - which are in turn composed of AHB transfers. A peripheral requests a transaction through the handshaking interface to the DMAC.

The following examples show the effect of different settings on a DMA block transfer.

- Memory-to-Peripheral (assumed that peripheral 0 and channel 0 are used)
  - Polling the DMA_CHANNEL[0].CHL_EN.CHL_EN status, if channel0 in disable status, then it can initiate a new transfer, otherwise, channel0 is serving another transfer and can't use it now.
  - Program DMA_CHANNEL[0].CTRLB.PERNUM = 6'b00_0000.
  - Program DMA_CHANNEL[0].CTRLA fields:
    - DMA_CHANNEL[0].CTRLA.INCSRCADDR = 1'b1
    - DMA_CHANNEL[0].CTRLA.INCTRGADDR = 1'b0
    - DMA_CHANNEL[0].CTRLA.TRAN_TYPE = 2'b01
    - DMA_CHANNEL[0].CTRLA.TRAN_SIZE = 2'bxx
    - DMA_CHANNEL[0].CTRLA.WIDTH = 2'bxx
    - DMA_CHANNEL[0].CTRLA.LEN = 13bx_xxxx_xxxx_xxxx
  - Program SADR0 and TADR0 respectively to the memory and peripheral addresses. The peripheral address should be aligned to DMA_CHANNEL[0].CTRLA.WIDTH, otherwise, ADDRERRINT[0] will occur and the transfer will be stopped if it is not been masked.
  - Program the MASK registers to decide which type of interrupt should be visible to CPU.

- Program DMA_CHANNEL[0].CHL_EN.CHL_EN = 1'b1.
- Poll the status of DMA_CHANNEL[0].CHL_EN.CHL_EN, or wait for interrupt and check the interrupt registers.

■ Peripheral-to-Memory (It is assumed that peripheral 15 and channel 8 are used)
- Poll the DMA_CHANNEL[8].CHL_EN.CHL_EN status, if channel8 in disable status, then it can initiate a new transfer, otherwise, channel8 is serving another transfer and can't use it now.
- Program DMA_CHANNEL[8].CTRLB.PERNUM = 6'b00_1111.
- Program DMA_CHANNEL[8]CTRLA fields:
  - DMA_CHANNEL[8]CTRLA.INCSRCADDR = 1'b0
  - DMA_CHANNEL[8]CTRLA.INCTRGADDR = 1'b1
  - DMA_CHANNEL[8]CTRLA.TRAN_TYPE = 2'b10
  - DMA_CHANNEL[8]CTRLA.TRAN_SIZE = 2'bxx
  - DMA_CHANNEL[8]CTRLA.WIDTH = 2'bxx
  - DMA_CHANNEL[8]CTRLA.LEN = 13bx_xxxx_xxxx_xxxx
- Program SADR8 and TADR8 to the peripheral and memory addresses. The peripheral address should be aligned to DMA_CHANNEL[8]CTRLA.WIDTH, otherwise, ADDRERRINT[8] will occur and the transfer will be stopped if it is not been masked.
- Program the MASK registers to decide which type of interrupt should be visible to CPU.
- Program DMA_CHANNEL[8].CHL_EN.CHL_EN = 1'b1.
- Polling the status of DMA_CHANNEL[8].CHL_EN.CHL_EN, or wait for interrupt and check the interrupt registers.

■ Memory-to-Memory (It is assumed that channel 0 is used)
- Poll the DMA_CHANNEL[0].CHL_EN.CHL_EN status, if channel0 in disable status, then it can initiate a new transfer, otherwise, channel0 is serving another transfer and can't use it now.
- Program DMA_CHANNEL[0].CTRLA fields:
  - DMA_CHANNEL[0].CTRLA.INCSRCADDR = 1'b0
  - DMA_CHANNEL[0].CTRLA.INCTRGADDR = 1'b1
  - DMA_CHANNEL[0].CTRLA.TRAN_TYPE = 2'b10
  - DMA_CHANNEL[0].CTRLA.TRAN_SIZE = 2'bxx
  - DMA_CHANNEL[0].CTRLA.WIDTH = 2'bxx
  - DMA_CHANNEL[0].CTRLA.LEN = 13bx_xxxx_xxxx_xxxx
- Program SADR0 and TADR0 to the source and target memory addresses.
- Program the MASK registers to decide which type of interrupt should be visible to CPU.
- Program DMA_CHANNEL[0].CHL_EN.CHL_EN = 1'b1.
- Polling the status of DMA_CHANNEL[0].CHL_EN.CHL_EN, or wait for interrupt and check the interrupt registers.

8.6 Register Description

See Appendix A.2.2, DMAC Registers, on page 360 for a detailed description of the registers.

9 Real Time Clock (RTC)

9.1 Overview

The 88MW300/302 Real Time Clock (RTC) registers are controlled through the APB bus. The RTC is optimized for a counter in the Always ON (AON) domain.

9.2 Features

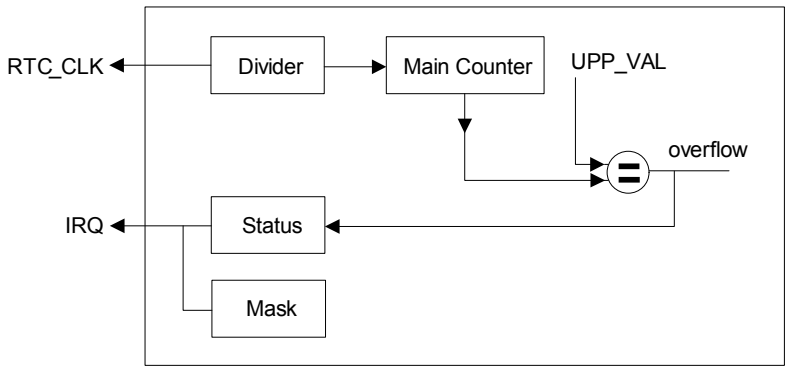
- Selectable clock source
- Programmable clock divider
- 32-bit Up counter with a programmable upper overflow boundary
- Interrupt is generated on the counter clock when it reaches the upper boundary

9.3 Functional Description

9.3.1 Block Diagram

Figure 40 shows an overall block diagram.

Figure 40: RTC Block Diagram



9.3.2 Counter Clock

The clock source of the RTC comes from the PMU. It can be set to XTAL32K or RC32K through RTC_INT_SEL bits in PERI_CLK_SRC register of PMU module. To avoid any potential issues, stopping the counter is required before changing the clock source. Reset the counter after changing the clock source.

The RTC can divide the clock simultaneously. CLK_DIV stores the clock division factor. The clock division formula is:

$$counter\_clock\_divide = counter\_clock / (2^{CLK\_DIV})$$

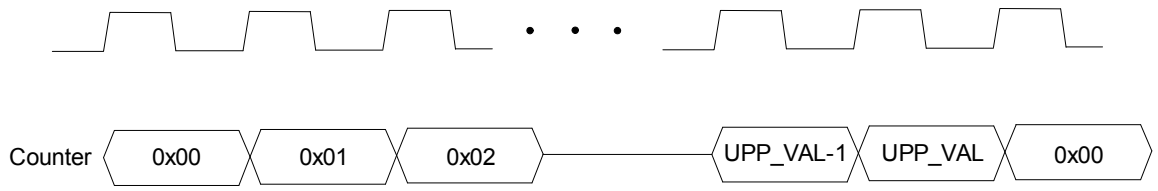
For example, if a timer clock divider register is set to 2, then the timer gets 1 tick every 4 clock ticks. The bit width of a clock divider register is 4, which makes the maximum value of CLK_DIV as 15 and the maximum division ratio as 32768:1.

9.3.3 Counting Mode

The RTC counter works in a counting-up mode. UPP_VAL defines the upper boundary of the counter; default value is 0xFFFFFFFF, the maximum counter value. The lower boundary is always 0.

The RTC counter value increments until reaching the upper boundary defined by UPP_VAL (event counter-reach-upper). In the next tick, the counter resets to 0 and begins counting up again. Upon a counter reset (write 1 to CNT_RESET), the counter resets to 0. A full cycle from 0 to UPP_VAL consists of UPP_VAL+1 counter ticks. Figure 41 shows the timing.

Figure 41: Count-up Mode Timing



9.3.4 Counter Update Mode

The counter value can be read from the APB bus through the register CNT_VAL. The update mode of CNT_VAL can be configured using CNT_UP_MOD.

Table 119 shows the configuration.

Table 119: Counter Update Mode

CNT_UPDT_MOD	Description
3	Reserved
2	Auto-update CNT_VAL is updated on every counter clock tick
1	Reserved
0	Update off

9.3.5 Interrupt

When the counter reaches the UPP_VAL, the CNT_UPP_INT bit in the INT_RAW register is set to 1. Interrupt status bits are always enabled to be set in the INT_RAW register. The interrupt status bit can be cleared by writing 1 to the corresponding bit in the INT_RAW register. Each interrupt status has a corresponding mask in the INT_MSK register. If the corresponding mask is set to 1, the interrupt status does not assert the interrupt. By default, all bits are masked. The INT register is the masked result of INT_RAW register. The interrupt is asserted if any of the bits in INT register is 1.

9.4 Programming Notes

9.4.1 Initialization

1. Before using the RTC, poll the STS_RESETN bit to be set.
2. Program various parameters
   a) Select clock source with RTC_INT_SEL bit in PERI_CLK_SRC register of PMU module.
   b) Set counter upper value in UPP_VAL.
   c) If the counter value needs to be read out, program CNT_UPDT_MOD to 0x2. Otherwise, leave it at 0x0
3. Write 1 to CNT_RESET to reset the counter. Poll CNT_RST_DONE bit to be set to determine when the counter finishes resetting. Do not access any other registers until CNT_RST_DONE is 1.
4. Write 1 to CNT_START to start the counter. Poll CNT_RUN bit to be set to determine when the counter begins to count.

9.4.2 UPP_VAL

The value written to UPP_VAL is not valid immediately. It is not effective until the counter overflows. To make the value valid immediately, write 1 to CNT_RESET.

9.5 Register Description

See Appendix A.19.2, RTC Registers, on page 761 for a detailed description of the registers.

THIS PAGE INTENTIONALLY LEFT BLANK

10 Watchdog Timer (WDT)

10.1 Overview

The 88MW300/302 Watchdog Timer (WDT) regains control in case of system failure (due to a software error) to increase application reliability. The WDT can generate a reset or an interrupt when the counter reaches a given timeout value.

10.2 Features

- WDT module gets clock from APB clock
- 32-bit down counter with the minimal timeout value of 65536
- Configurable reset or interrupt generation with the given timeout value
- Supports 8 types of reset pulse length

10.3 Functional Description

10.3.1 Counter Operation

The watchdog counter descends from a preset (timeout) value to 0. The timeout value is obtained by the formula of 2^(16+ WDT.TORR.TOP_INIT) or 2^(16+ WDT.TORR.TOP). The register bit WDT.TORR.TOP_INIT is only used to initialize timeout period for the first counter restarts, which should be written after reset and before the WDT is enabled. The register bit WDT.TORR.TOP is used to select the timeout period from which the WDT count restarts. Depending on the output response mode selected, when the counter reaches 0, either a system reset or an interrupt occurs . The output response mode is set using the WDT.CR.RMOD register bit. WDT.CR.RMOD = 0 generates a system reset and WDT.CR.RMOD = 1 first generates an interrupt. If it is not cleared before a second timeout occurs then, a system reset is generated.

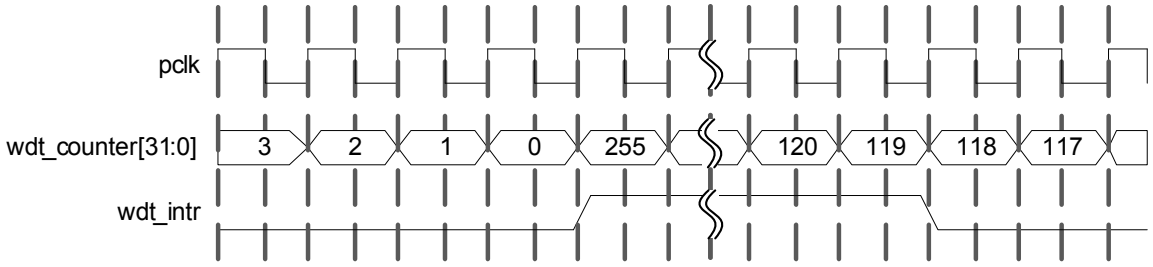
Users can restart the counter to its initial value (timeout value) by writing to the restart register WDT.CRR[7:0] at any time. The process of restarting the watchdog counter is sometimes referred to as "kicking the dog." As a safety feature to prevent accidental restarts, the value 0x76 must be written to the current counter value register (WDT.CRR).

10.3.2 Interrupt

The WDT can be programmed to generate an interrupt (and then a system reset) when a timeout occurs. When WDT.CR.RMOD is programmed to 1, the WDT generates an interrupt. If it is not cleared by the time a second timeout occurs, then it generates a system reset. If a restart occurs at the same time the watchdog counter reaches 0, an interrupt is not generated.

Figure 42 shows the timing diagram of the interrupt being generated and cleared. The interrupt is cleared by reading the WDT.EOI register in which no kicks required. The interrupt can also be cleared by a “kick” (watchdog counter restart).

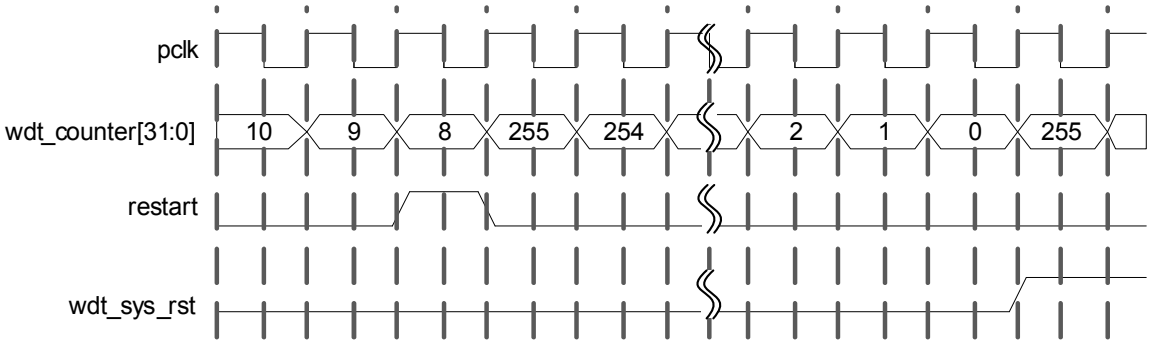
Figure 42: Interrupt Generation



10.3.3 System Reset

When bit WDT.CR.RMOD is programmed to 0, the WDT generates a system reset when a timeout occurs. Figure 43 shows the timing diagram of the WDT system reset.

Figure 43: Counter Restart and System Reset



10.3.4 Reset Pulse Length

The reset pulse length is the number of pclk cycles for which a system reset is asserted. When a system reset is generated, it remains asserted for the number of cycles specified by the reset pulse length or until the system is reset (by the Reset Controller). A counter restart has no effect on the system reset once it has been asserted. The reset pulse length is set by programmed the WDT.CR.RPL register field. The register bits can be programmed to 8 types of pulse length. The reset pulse is selected by balancing reset reliability and reset latency. A longer reset pulse provides a more reliable reset but may result in longer reset latency.

10.4 Initialization Sequence

When the counter reaches 0, depending on the output response mode selected, either system reset or an interrupt occurs.

The following sequence of operations must be followed to start the watchdog timer.

1. Configure the WDT in Generate Reset mode by setting WDT.CR.RMOD to 0.
2. Configure timeout value.
3. Set WDT.CR.WDT.EN to 1'b1.

The following sequence of operations must be followed to start the watchdog timer to generate an interrupt.

1. Configure WDT in Generate Interrupt mode by setting WDT.CR.RMOD to 1.
2. Configure timeout value.
3. Set WDT.CR.WDT.EN to 1'b1.

10.5 Register Description

See Appendix A.18.2, WDT Registers, on page 751 for a detailed description of the registers.

THIS PAGE INTENTIONALLY LEFT BLANK

11 General Purpose Timers (GPT)

11.1 Overview

The 88MW300/302 includes 4, 32-bit GPTs. Registers are controlled through the APB bus.

11.2 Features

Each GPT is a multi-purpose counter that supports the following:

- Selectable clock source
- Programmable clock divider and pre-scalar
- 32-bit up counter
- 6 independent channels with multiple modes
- Input capture for external inputs
- Edge-aligned and Center-aligned Pulse Width Modulation (PWM)
- “1-shot” mode to trigger a 1-time output change and interrupt
- Auto-trigger ADC/DAC module for PWM mode
- DMA transfer for input capture
- Interrupt generation on counter and channel events

11.3 Interface Signal Description

Table 120 shows the interface signals.

Table 120: GPT Interface Signals

Signal Name	Type	Description
GPTx_CHx	I/O	Timer Number and Channel Number
GPTx_CLKIN	I	Clock

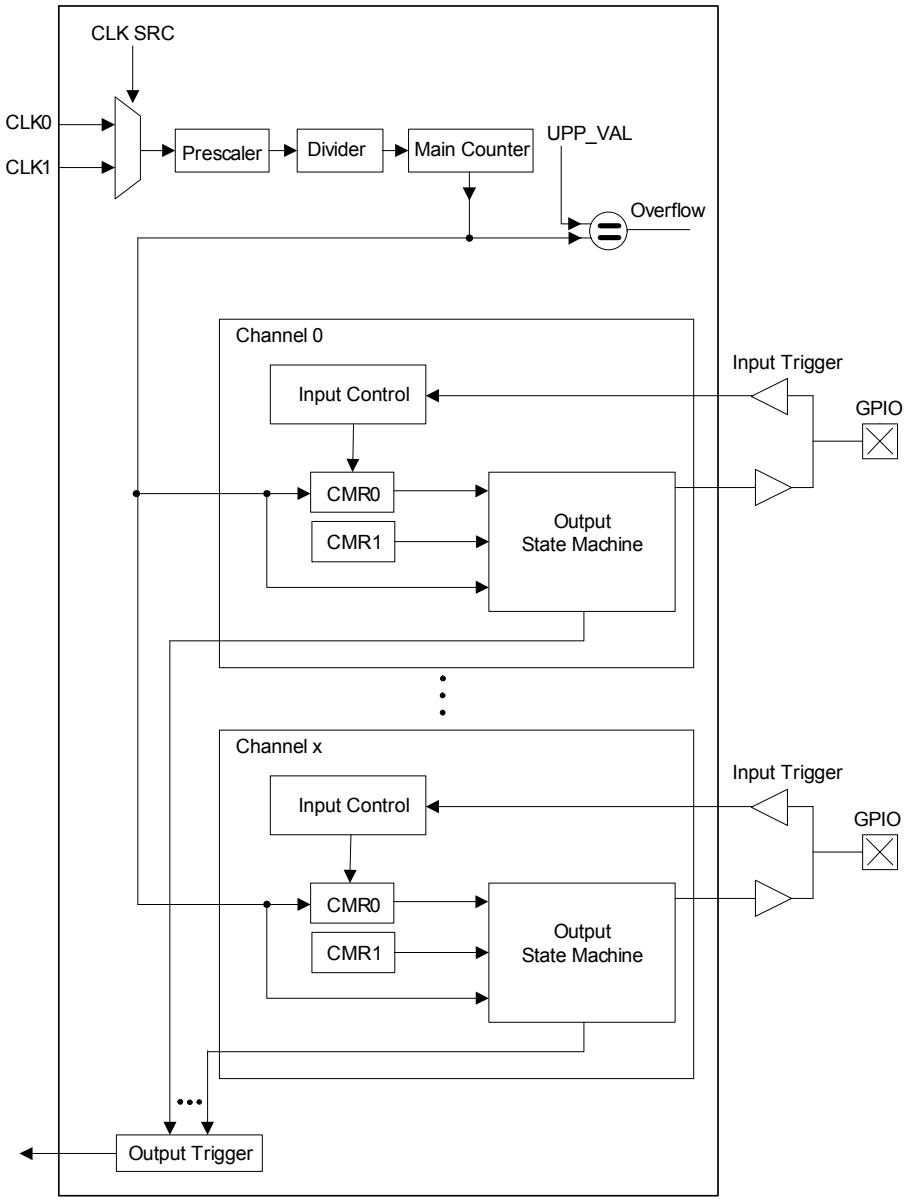
11.4 Functional Description

Each timer supports as many as 6 channels. Each channel shares the same clock source but has a separate set of registers for configuration. In this way, each channel can serve different applications independently. The register prefix CHx means that register is for the Channel *x*.

11.4.1 Block Diagram

Figure 44 shows an overall block diagram.

Figure 44: GPT Block Diagram



11.4.2 Counter

11.4.2.1 Counter Clock

Counter Clock Source

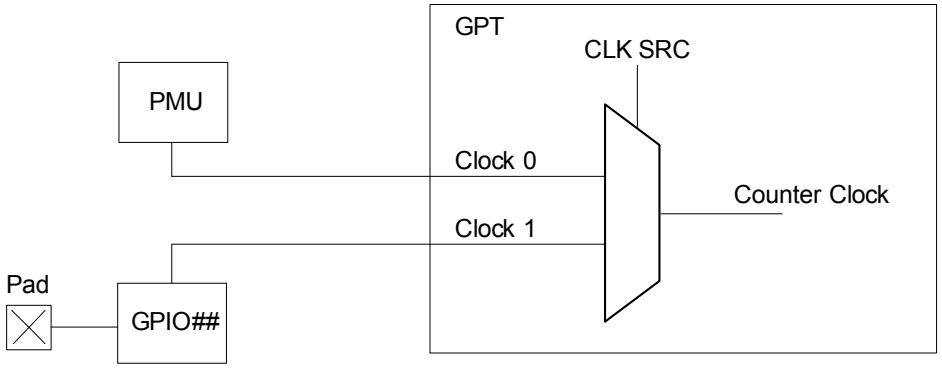
The clock source of the timer can be selected with CLK_SRC in CLK_CNTL register in the GPT. There are 2 choices available:

- Clock 0 (default) from PMU
- Clock 1 from the GPIO

Clock 0 can be chosen from multiple sources. Details regarding the sources of Clock 0 are in the PMU and Clocking registers description. When using Clock 1, the corresponding GPIO function must be programmed to the appropriate value, and the pad must be connected to a clean external clock. To avoid any potential issues, stopping the counter is necessary before changing the clock source. Reset the counter after changing the clock source.

Figure 45 shows the clock source selection.

Figure 45: Clock Source Selection



Clock Pre-Scaling and Division

The GPT can divide and pre-scale the clock simultaneously. The combination of the divider and pre-scalar allows for many possible integer ratios within the range. CLK_PRE can linearly pre-scale the counter clock using the formula:

counter_clock_prescale = counter_clock / (CLK_PRE + 1)

Each pre-scalar has 8 bits, allowing a scaling factor from 1 to 256.

After the clock pre-scaling, the resulting clock can be further divided down. CLK_DIV stores the clock division factor. The clock division formula is:

$$counter\_clock\_divide = counter\_clock\_prescale / (2^{CLK\_DIV})$$

For example, if a timer clock divider register is set to 2, then the timer gets 1 tick every 4 clock ticks. The bit width of a clock divider register is 4, which makes the maximum value of CLK_DIV as 15 and the maximum division ratio as 32768:1.

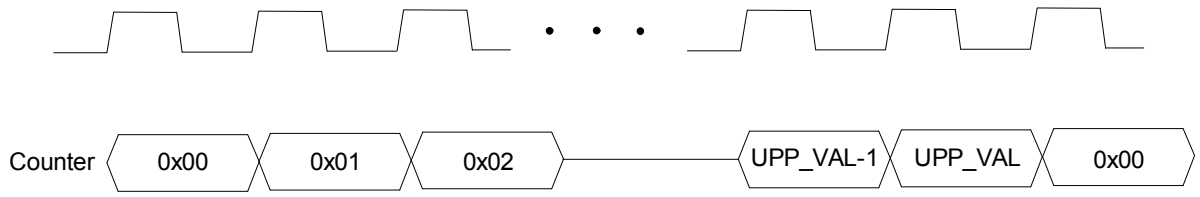
11.4.2.2 Counting Mode

The GPT always counts up. UPP_VAL defines the upper boundary of the counter. The main counter counts from 0 to UPP_VAL, overflows to 0 and continues counting. A full cycle from 0 to UPP_VAL consists of UPP_VAL+1 counter ticks. The CNT_UPP_STS status bit is set upon an overflow. Upon a count reset (write 1 to CNT_RESET), the counter resets to 0.

The value written to UPP_VAL is not valid immediately. It is not effective until the counter overflows. To make the value valid immediately, write 1 to CNT_RESET.

Figure 46 show the count-up mode timing.

Figure 46: Count-up Mode Timing



11.4.2.3 Counter Update Mode

The counter can be read from the APB bus through the register CNT_VAL. Updates to CNT_VAL are determined by CNT_UPDT_MOD. Table 121 shows the configuration.

Table 121: Counter Update Mode Configuration

CNT_UPDT_MOD	Description
3	Update Off If CNT_VAL does not need to be read, CNT_UPDT_MOD can be set to off to save power.
2	Reserved
1	Auto-update Fast Used when counter clock is at least 5 times slower than the APB clock. CNT_VAL is updated on every counter clock tick.
0	Auto-update Normal Can be used for any clock relationship between the counter clock and the APB clock. Only every 3-4 counter ticks are updated to CNT_VAL.

11.4.3 Interrupts

Table 122 shows the type of events that can generate interrupts.

Table 122: Available Interrupt Events

Event	Availability
Channel status	Yes
Channel error status	Yes
Reach UPP_VAL	Yes
DMA overflow	Yes

Registers used to control the interrupts include STS, INT, and INT_MSK. They all have corresponding bits in the same location. The status bits are in the STS register. Various events in the timer set the status bits automatically. The status bit can be cleared by writing 1 to the corresponding bit in STS.

Each status bit has a corresponding mask in INT_MSK register. If the mask bit is set to 1, the status bit is masked and does not generate an interrupt. If the mask bit is 0, then the status bit can generate an interrupt. By default, all bits are masked.

The INT register is the masked result of the STS register. If the mask bit is 1, then the corresponding bit in the INT register is 0. If the mask bit is 0, then the corresponding bit in the INT register is the same value as that in STS register.

The interrupt is asserted if any of the bits in INT register is 1.

11.4.4 Channel Operation Modes

11.4.4.1 Counter Match Register 0 and 1 (CMR0 and CMR1)

CMR0 and CMR1 are a pair of multipurpose registers for each channel. In input-capture mode, CMR0 is used to store the captured values. In all other modes, CMR0 and CMR1 are used to determine counter parameters.

The flow of updating the values of CMR0 and CMR1 is as follows:

1. Write new values to CMR0 and CMR1.
2. Write 1 to CHx_CMR_UPDT in the USER_REQ register.
3. Check the value of CHx_ERR_STS, if it is 0, it means CMR0 and CMR1 are updated successfully; if it is 1, clear CHx_ERR_STS, and repeat Steps 2-3.

11.4.4.2 No Function Mode

Set CHx_IO to 0 to configure the channel to no function. The channel does nothing and does not set the status bit. Set unused channels to this mode to save power and avoid unpredictable behaviors.

11.4.4.3 Input Capture Mode

Set CHx_IO to 1 to configure the channel to input-capture mode.

In input-capture mode, the channel waits for 1 of 2 trigger events to occur:

- An external trigger can come from a GPIO. The timer samples the edge transition using a fast sampling clock.
- Write to CHx_USER_ITRIG (x = 1, 2, 3, 4, or 5) to generate a software trigger.

CMR0 is the capture register.

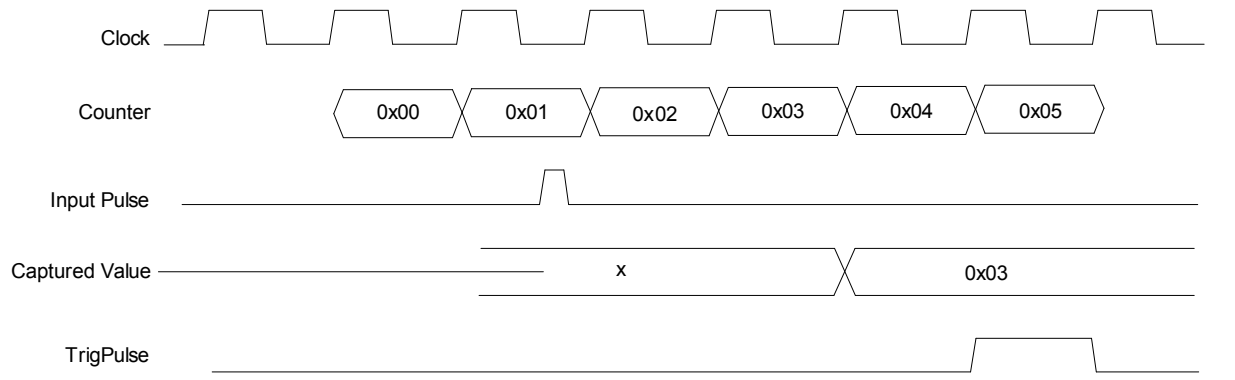
The external trigger event can be a rising or falling edge. An external trigger event is considered valid after being filtered with the settings programmed in the IC_CNTL and CHx_CNTL registers. Space external triggers sufficiently apart relative to the sampling parameters to allow sufficient time to read out the value before the next capture. Small glitches can be filtered using the input capture registers, but in general the triggers should be clean.

Each valid trigger event sets the channel status bit CHx_STS.

A valid trigger event can be generated manually by writing 1 to CHx_USER_ITRIG. This Write bypasses any sampling filters in IC_CNTL register. In this mode, during the tick where a trigger event occurs, the counter value is copied to the capture register.

Figure 47 shows the input capture mode timing.

Figure 47: Input Capture Timing



DMA

In input-capture mode, CHx_CMR0 is shared as a capture register. If the captured value is required to be stored in memory by DMA, the general-purpose timer provides hardware handshake signals to automate this process. The DMA signals follow the protocol of the DMA Controller.

To enable the DMA function:

1. Set DMAz_EN (z=0,1) in the DMA_CNTL_EN register to 0.
2. Select GPT channel x as the source by programming DMAz_CH = x.
3. Program CHx_CNTL to set channel *x* to input capture.
4. Set DMAz_EN to 1 to enable the DMA channel.

On the DMA Controller:

1. Write to the DMA_HS register in system control module to set DMA handshake mapping.
2. Set SAR to the address of the capture register (CHx_CMR0).
3. Set DAR to the memory address.
4. In the CTL register:
   a) Write to SRC_TR_WIDTH and DST_TR_WIDTH to set the transfer width to 32 bits.

b) Write to SRC_MSIZE and DEST_MSIZE to set the burst transfer length to 1 item.
c) Write to TT_FC to set the transfer type to peripheral-to-memory.
d) Write to BLOCK_TS to configure the transfer length.
e) Set SINC to maintain source address.
f) Set DINC to make destination address increase.

5. In CFG register, set HS_SEL_SRC and HS_SEL_DST to select hardware handshaking; set SRC_PER and DST_PER to assign hardware handshaking interfaces.

11.4.4.4 1-Shot Pulse Mode

Set CH_IO to 4 to configure the channel to 1-shot pulse mode. See Table 123 and Figure 48.

Table 123: 1-Shot Pulse Control Registers

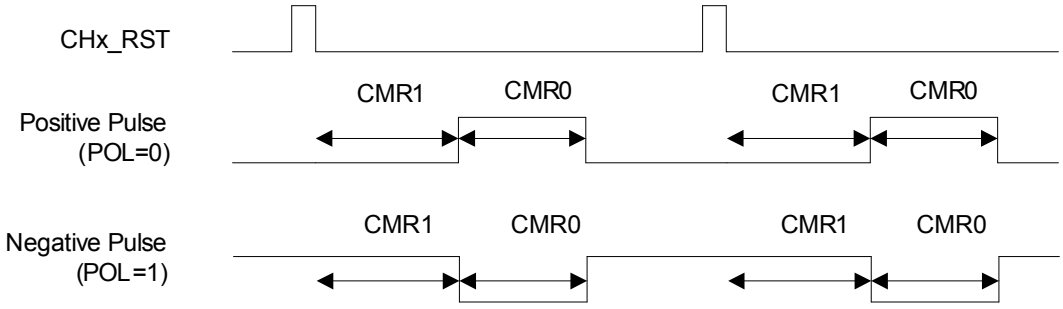
Register	Description
Positive Polarity (POL = 0)	Positive pulse
Negative Polarity (POL = 1)	Negative pulse
Duty Cycle	CMR0
Period	CMR0 + CMR1

- Generates a single pulse
- Setting CMR1 to 0 results in an instant pulse generation
- Setting CMR0 to 0 results in no pulse, but the status bit remains set at the end of period

Write 1 to CHx_RST to generate 1 pulse:

1. After the channel reset, the 1 state resets to POL.
2. Wait CMR1 cycles, then change output state to the reverse value of POL.
3. Wait CMR0 cycles, then change output state to POL and set the channel status bit.

Figure 48: 1-Shot Pulse



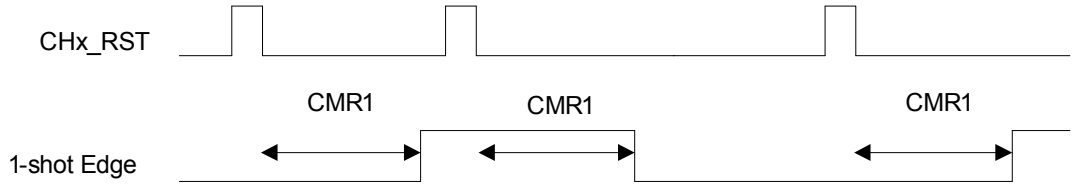
11.4.4.5 1-Shot Edge Mode

Set CH_IO to 5 to configure the channel to 1-shot edge mode. This mode generates a single edge transition. Setting CMR1 to 0 results in an instant edge transition. See Figure 49.

Write 1 to CHx_RST to generate 1 edge transition:

1. Channel reset.
2. Wait CMR1 cycles, then invert the current output state and set the channel status bit.

Figure 49: 1-Shot Edge Timing



11.4.4.6 Pulse Width Modulation (PWM) Edge-Aligned Mode

Set CH_IO to 6 to configure the channel to PWM edge-aligned mode. See Table 124 and Figure 50, PWM Edge-Aligned, on page 199.

Table 124: PWM Edge-Aligned Control Registers

Register	Description
Positive Polarity (POL = 0)	High -> Low
Negative Polarity (POL = 1)	Low -> High
Duty Cycle	CMR0
Period	CMR0 + CMR1

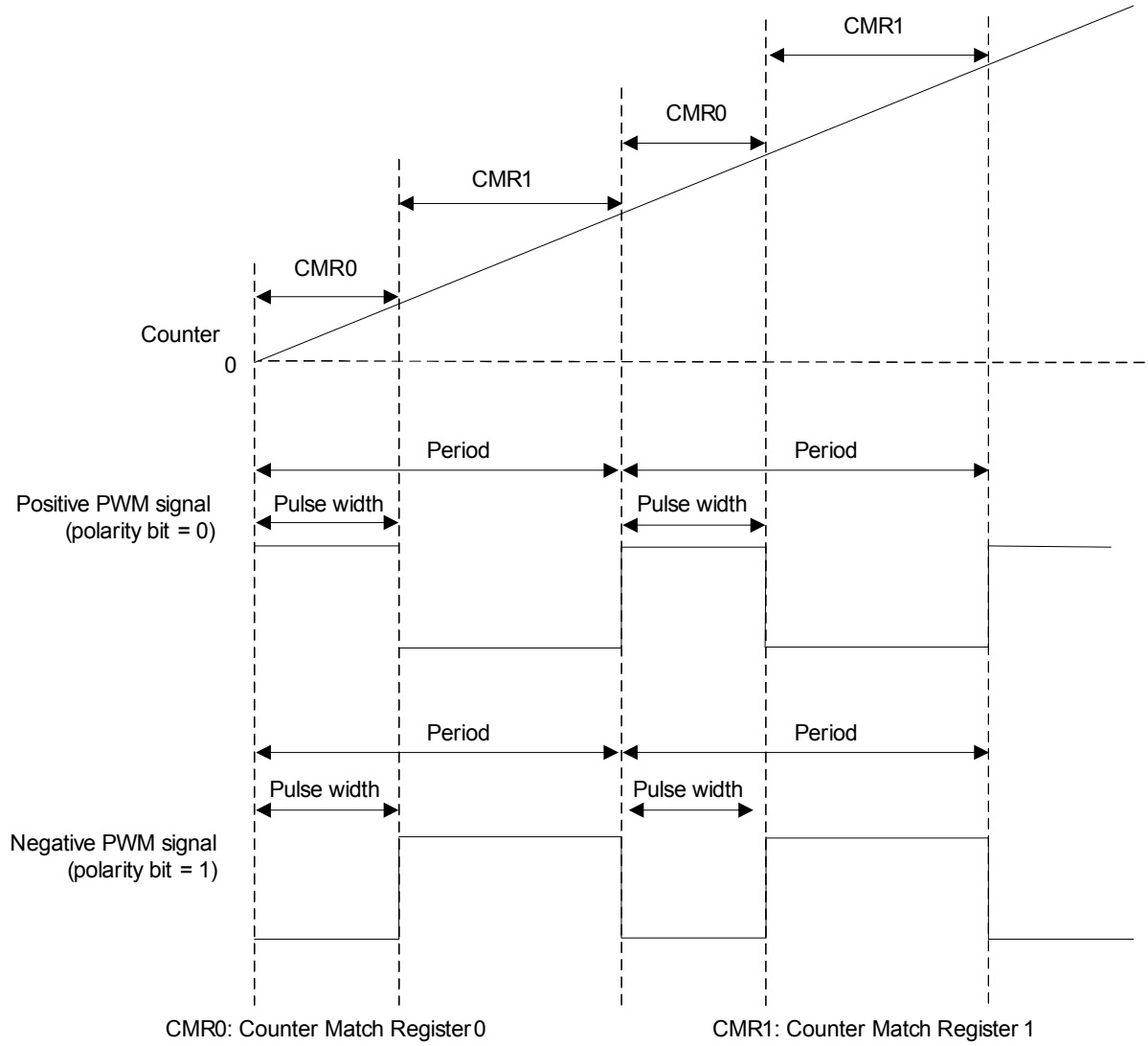
PWM edge-aligned is a periodic square waveform aligned to the starting edge of the period. To adjust the duty cycle, subtract a number from either CMR0 or CMR1 and add it to the other, thereby keeping the period the same.

Setting CMR0 to 0 results in a 0% duty cycle, and setting CMR1 to 0 results in a 100% duty cycle. Setting both CMR0 and CMR1 to 0 pauses the PWM. The output remains at the previous state and no additional interrupts are generated. To restart the PWM, set at least 1 CMR0 or CMR1 to a non-0 value, then write 1 to CHx_CMR_UPDT.

The behavior of the PWM Edge-Aligned mode is as follows:

1. Change CH_IO to 6.
2. Channel reset – Output state is first reset to POL.
3. On the next counter tick, output state changes to the reverse value of POL.
4. Wait CMR0 cycles, then set the output state to POL.
5. Wait CMR1 cycles, then set the output state to the reverse value of POL and set the channel status bit.
6. Repeat 4-5.

Figure 50: PWM Edge-Aligned



11.4.4.7 Pulse Width Modulation (PWM) Center-Aligned Mode

Set CH_IO to 7 to configure the channel to PWM center-aligned mode. See Table 125 and Figure 51, PWM Center-Aligned, on page 201.

Table 125: PWM Center-Aligned Control Registers

Register	Description
Positive Polarity (POL = 0)	Low -> High -> High -> Low
Negative Polarity (POL = 1)	High -> Low -> Low -> High
Duty Cycle	2 x CMR0
Period	2 x CMR0 + 2 x CMR1

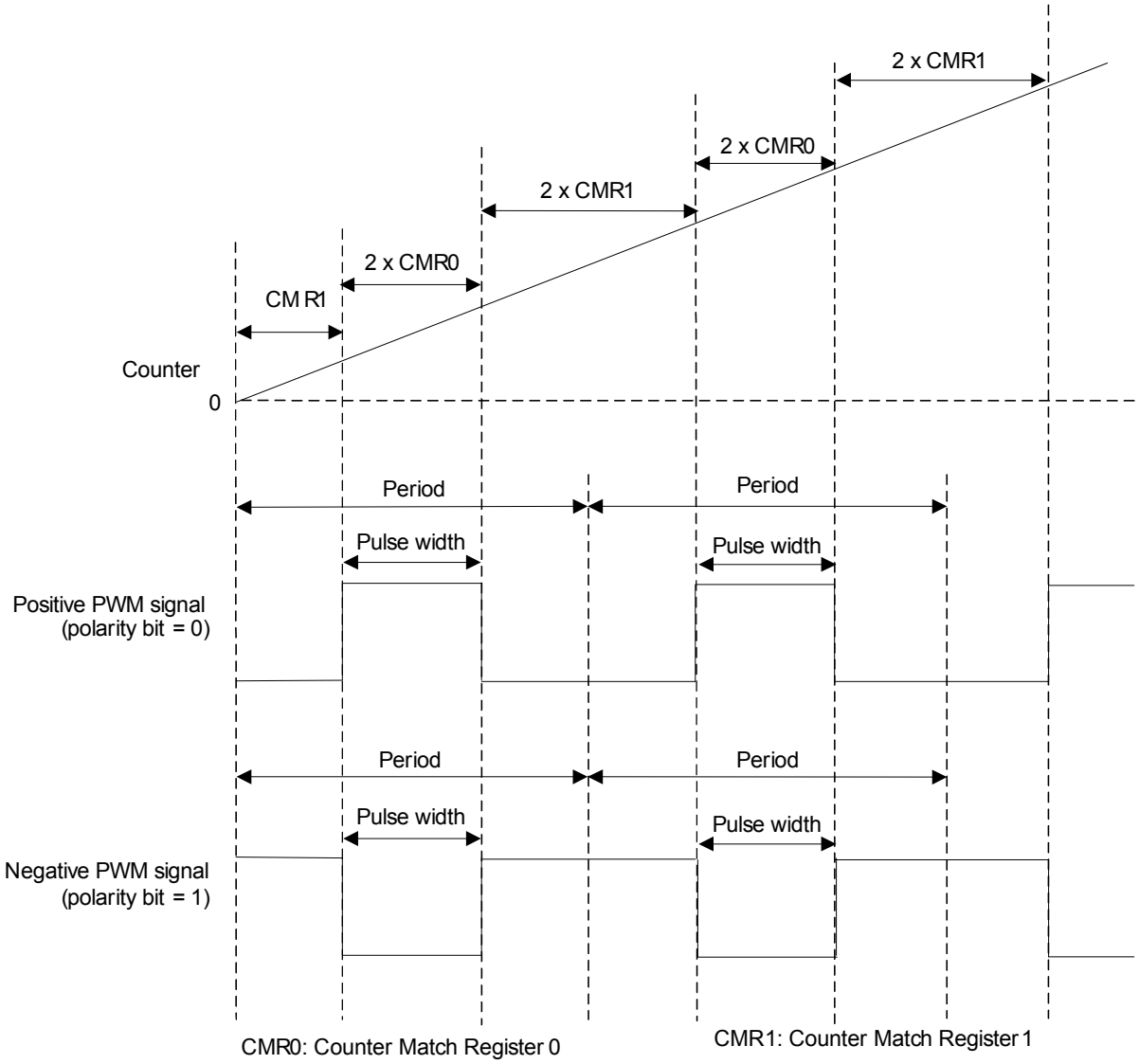
PWM center-aligned is a periodic square waveform aligned to the center of the period. To adjust the duty cycle, subtract a number from either CMR0 or CMR1 and add it to the other, thereby keeping the period the same.

Setting CMR0 to 0 results in a 0% duty cycle, and setting CMR1 to 0 results in a 100% duty cycle. Setting both CMR0 and CMR1 to 0 pauses the PWM. The output remains at the previous state and no additional interrupts are generated. To restart the PWM, set at least 1 CMR0 or CMR1 to a non-0 value.

The behavior of the PWM Center-Aligned mode is as follows (see Figure 51, PWM Center-Aligned, on page 201):

1. Change CH_IO to 7.
2. Write 1 to CHx_RST – Output state is first reset to POL.
3. Wait CMR1 cycles, then set the output state to the reverse value of POL.
4. Wait 2x CMR0 cycles, then set the output state to POL.
5. Wait CMR1 cycles, then set the channel status bit.
6. Repeat steps 3, 4, and 5.

Figure 51: PWM Center-Aligned



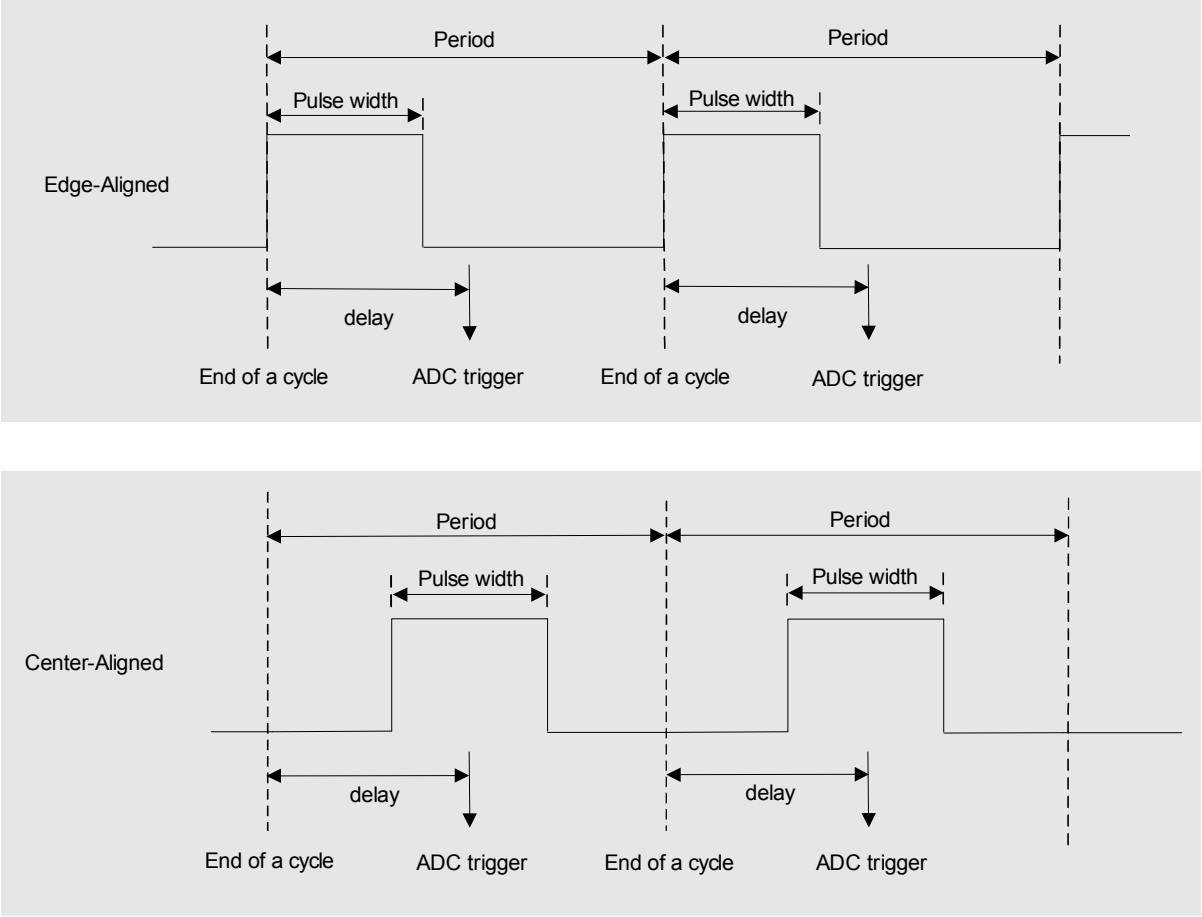
11.4.5 ADC Trigger

The ADC trigger is available only in GPT0 and GPT1.

The ADC trigger is a hardware handshake signal that periodically signals the Analog-Digital Converter (ADC) to begin a data conversion. The ADC trigger source can be selected from the 6 GPT channels using TRIG_CHSEL. bits in the TCR register. The selected GPT channel must be in a PWM mode for the ADC trigger to assert. The ADC trigger can be delayed from the end of the PWM period by programming TRIG_DLY bits in the TDR register. When TRIG_EN equals 1, the ADC trigger is enabled. See Figure 52.

- TRIG_DLY has a 4-cycle resolution which allows the delay to cover the maximum period for the PWM Center-Aligned mode
- Effect of ADC delay must be shorter than PWM period
- PWM period must be longer than ADC conversion time

Figure 52: ADC Trigger for PWM Edge-Aligned and Center-Aligned

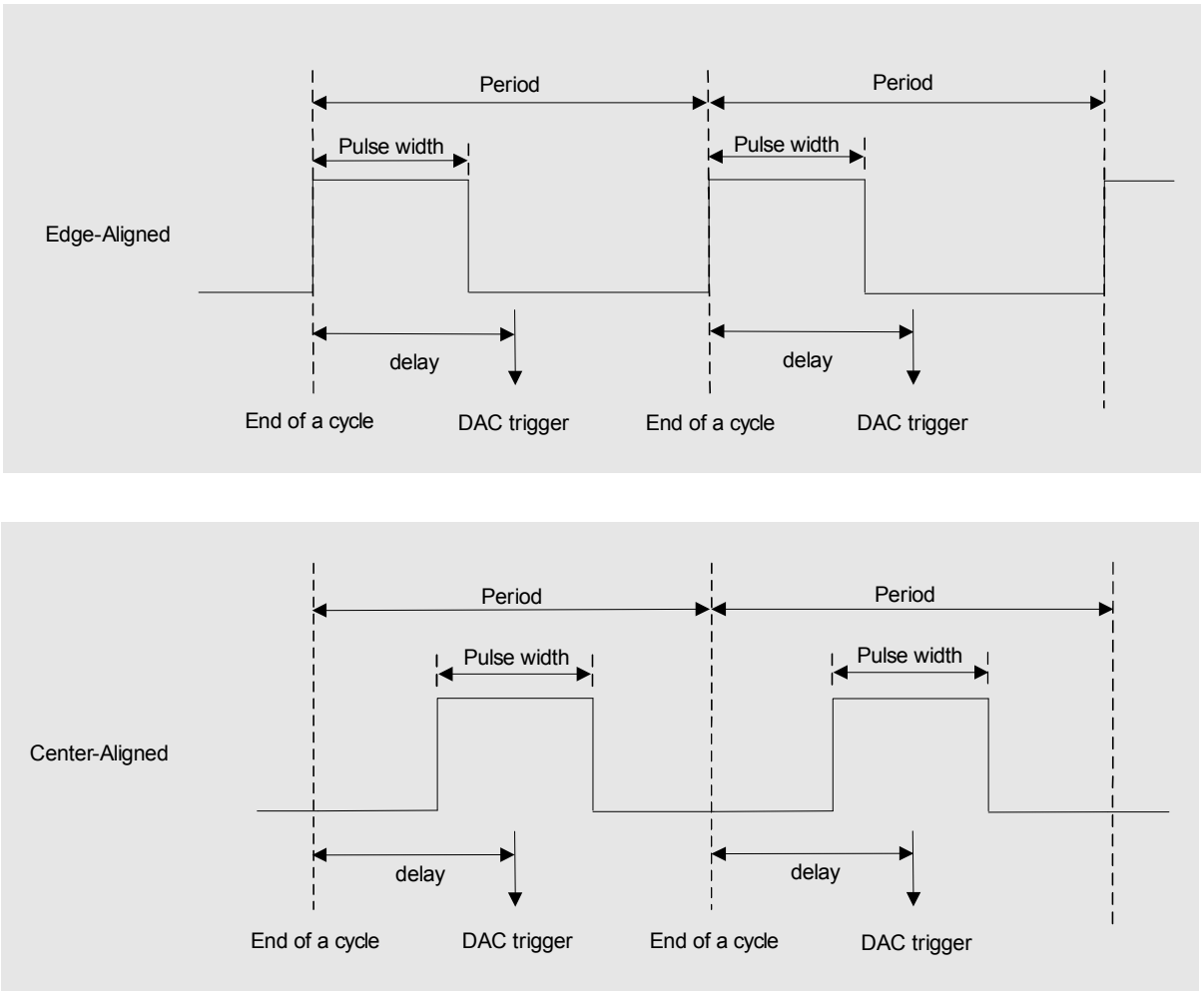


11.4.6 DAC Trigger

The DAC trigger--available only in GPT2 and GPT3--is a hardware handshake that signals the DAC to begin a conversion. The DAC trigger source can be selected from the 6 GPT channels using TRIG_CHSEL bit in the TCR register. The selected GPT channel must be in a PWM mode for the DAC trigger to assert. The DAC trigger can be delayed from the end of PWM period by programming TRIG_DLY bits in the TDR register. The DAC trigger is enabled when TRIG_EN = 1. See Figure 53.

- TRIG_DLY has a 4-cycle resolution which allows the delay to cover the maximum period for the PWM Center-Aligned mode
- TRIG_DLY delay time must be shorter than the PWM period
- PWM period must be longer than the DAC conversion time

Figure 53: DAC Trigger for PWM Edge-Aligned and Center-Aligned



11.5 Programming Notes

11.5.1 Initialization

1. Before using the GPT, poll STS_RESETN bit to be set.
2. Program various parameters:
   a) Select clock source with CLK_SRC. If CLK_SRC is set to 0, corresponding registers in PMU module should be configured for further clock source selection.
   b) Select clock prescalar and divider values in CLK_PRE & CLK_DIV.
   c) Set counter upper value in UPP_VAL.
   d) If the counter value needs to be read out, program CNT_UPDT_MOD. Otherwise, leave it at 0x0.
   e) Set CH_IO to select channel mode and configure related parameters for the channels to be used.
3. Write 1 to CNT_RESET to reset the counter. Poll CNT_RST_DONE for a 1 to determine when the counter finishes resetting. Do not access any other registers until CNT_RST_DONE is 1.
4. Write 1 to CNT_START to start the counter. Poll CNT_RUN for a 1 to determine when the counter begins to count.
5. If GPT channels are used, write 1 to CHx_ RST to enable the corresponding channel.

11.5.2 UPP_VAL

The value written to UPP_VAL is not valid immediately. It is not effective until the counter overflows. To make the value valid immediately, write 1 to CNT_RESET.

11.5.3 User Request Register

The User Request Register (USER_REQ) performs various operations on each channel. Operations are not pipelined and must be synchronized to the proper clock domain, so operations in the USER_ REQ register must not be performed in quick succession. The definition of quick succession in this case is 5 counter clock cycles.

For example, if a Write is to perform a channel reset, wait at least 5 counter clock cycles before performing the next channel reset. When the counter-clock frequency is close to the APB clock frequency, the quick succession delay can be easily waited out with just a few extra register accesses. When the counter clock frequency is much slower than the APB clock, then the safest way is to read CNT_VAL and wait 5 counter clock cycles to pass.

11.6 Register Description

See Appendix A.12.2, GPT Registers, on page 662 for a detailed description of the registers.

12 Advanced Encryption Standard (AES)

12.1 Overview

The 88MW300/302 AES engine provides fast and energy efficient hardware encryption and decryption service for the device.

12.2 Features

- Supports as many as 6 block cipher modes: ECB, CBC, CTR, CCM*, MMO, and Bypass
- Supports 128-, 192-, and 256-bit keys
- Efficient CPU/DMA access support
- Interrupt on finished AES operation, input FIFO full and output FIFO empty
- Error indication for each block cipher mode
- Separate 4*32-bit input and output FIFO

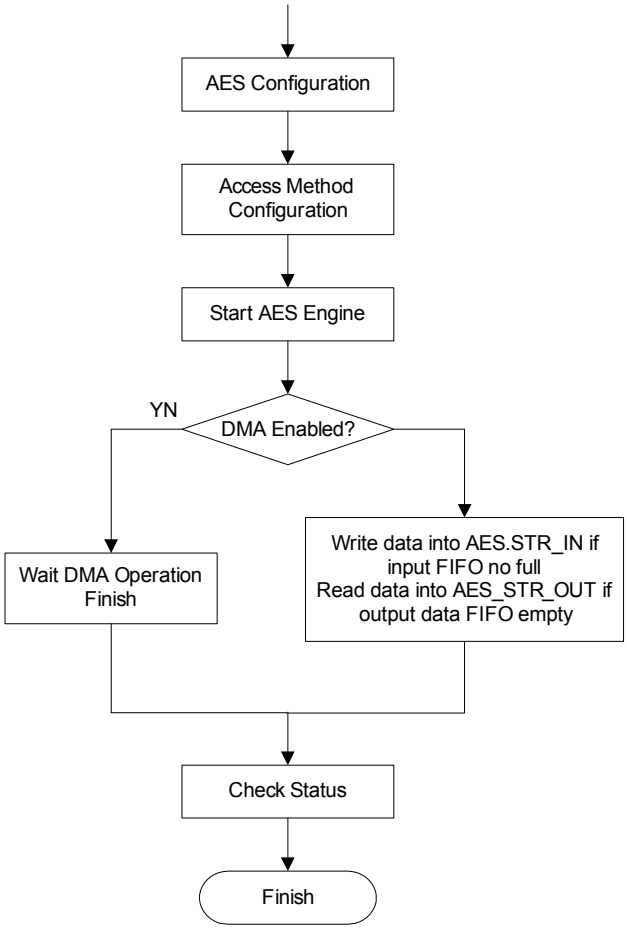
12.3 Functional Description

The AES module implements ECB, CBC, CTR, CCM*, MMO, and Bypass block cipher modes by efficient hardware.

12.3.1 AES Operational Flow

Figure 54 shows the AES operational flow.

Figure 54: AES Operational Flow



12.3.2 AES Configuration

Ensure correct configuration before starting the AES engine by following these steps:

1. Set AES engine to encrypt or decrypt by clearing/setting AES.CTRL1.DECRYPT
2. Configure AES block cipher mode by setting AES.CTRL1.MODE, 0 for ECB mode, 1 for CBC mode, 2 for CTR mode, 5 for CCM* mode, 6 for MMO mode and 7 for BYPASS mode
3. Configure AES key size. AES engine supports 3 types of key size: 128-, 192-, and 256- bit. Configure AES key size parameter by setting AES.CTRL1.KEY_SIZE
4. Fill the key according to key size. AES engine contains 8, 32-bit key registers defined as AES.KEY0, AES.KEY1, AES.KEY2, AES.KEY3, AES.KEY4, AES.KEY5, AES.KEY6 and AES.KEY7. When key size is set to 128-bit, then AES.KEY7/6/5/4 is used. When key size is set to 192-bit, then AES.KEY7/6/5/4/3/2 is used. When key size is set to 256-bit, then AES.KEY7/6/5/4/3/2/1/0 is used. However, MMO does not support 192- and 256- bit key size, and key size is ignored in Bypass mode.
5. For all modes except CCM* mode, set input data size by setting AES.MSTR_LEN. For CCM* mode, set associate data size by setting AES.ASTR_LEN, set message data size by setting AES.MSTR_LEN.
6. If AES block cipher mode is CTR mode, set CTR mode's counter modular by setting AES.CTRL1.CTR_MODE.
7. For CCM* encryption or MMO mode, If MIC/HASH is needed, set AES.CTRL1.OUT_MIC bit to 1 to append MIC/HASH at the end of output stream. If only MIC/HASH is needed, we can block the encrypted data into output FIFO (set AES.CTRL1.OUT_MSG bit to 1), and get MIC/HASH from AES.OV3/2/1/0.
8. For CCM* mode, set AES.CTRL1.OUT_HDR bit to 1 to output B0 at the beginning of the output stream, if necessary.
9. Fill with initial value according to AES block cipher mode. AES engine contains 4, 32-bit initial vector registers: AES.IV0, AES.IV1, AES.IV2, AES.IV3.
   - For ECB/MMO/BYPASS mode, there are no initial vectors that need to be configured.
   - For CTR mode,
     - Set AES.IV0= initial counter, AES.IV1= Nonce[31:0], AES.IV2=Nonce[63:32], AES.IV3=Nonce[95:64].
     - For CCM* mode, set AES.IV0=Nonce [31:0], AES.IV1=Nonce[63:32], AES.IV2=Nonce[95:64], AES.IV3.Byte0=15-Nonce[103:96], AES.IV3.Byte1 = 15 - Nonce Byte Length.

   Note: For Bypass mode, the AES engine ignores input data; it passes it along unchanged to the output.

12.3.3 Data Access Method

The AES module contains separate 4*32-bit input and output FIFO. The input and output FIFO can be accessed by DMA or CPU.

When setting AES.CTRL1.IO_SRC bit to 1 and AES.CTRL1.DMA_EN bit to 1, the input and output FIFO are accessed by DMA. There are 2 channels required: 1 for input data and the other for output data. Before starting AES engine, the DMA controller must be configured, the transfer size is the input data length and output data length. The AES operation finishes as the DMA operation finishes.

When setting AES.CTRL1.IO_SRC bit to 0 and AES.CTRL1.DMA_EN bit to 0, the input and output FIFO are accessed by CPU. Then it writes data into AES.STR_IN if the input FIFO is not full; read data from AES.STR_OUT if output FIFO is not empty. The AES operation finishes as the transfer data size reaches input and output data size.

12.3.4 Starting the AES Engine

Clear AES input and output FIFO and reset AES core before starting the AES engine. The AES input and output FIFO can be cleared by setting the AES.CTRL1.IF_CLR bit and AES.CTRL1.OF_CLR bit to 1. The AES core can be reset by setting and then clearing AES.CTRL2.CORE_RESET.

12.3.5 Interrupt Request

There are 3 interrupts for the AES engine: input FIFO full interrupt, output FIFO empty interrupt, and AES operation done interrupt. Each interrupt can be masked or cleared by setting AES.IMR/AES.IC registers.

12.3.6 Partial Code Support

The AES engine can automatically pad the input data when the input data length is not a multiple of 128 bits. The AES module supports the following padding scheme for different AES block cipher modes. Table 126 shows the scheme.

Table 126: Padding Scheme

Mode	Description
CCM*	Automatically padding 0 for both A string and M string
MMO	Automatically padding “100…00”+2 bytes length information
CBC	Cipher stealing is performed to partial codeword
CTR	Partial code word don’t affect the operation
ECB	Check partial case, assert error when partial cases detected

12.3.7 Error Status Check

Register AES.STATUS.STATUS records the error status for the AES engine when the AES operation has finished. Table 127 shows the error status for different AES block cipher modes.

Table 127: Error Status for Different AES Block Cipher Modes

Mode	Status[2]	Status[1]	Status[0]
ECB	n/a	Data is not multiple of 16 bytes	Input data size less than 16 bytes
CBC	n/a	n/a	
CTR	n/a	n/a	
CCM*	MIC mismatch during decryption	n/a	n/a
MMO	n/a	Data is more than 2^{13}-1 bytes	n/a
Bypass	n/a	n/a	n/a

12.3.8 Output Vector

The output vector provides some useful information, such as the last cipher block in CBC mode, last counter in CTR mode, MIC value in CCM* mode and HASH value in MMO mode. Register AES.OV3/2/1/0 records useful information for different AES block cipher modes.

Table 128 shows the recorded information in AES output vector for different AES block cipher mode.

Table 128: AES Output Vector

Block Cipher Mode	Output Vector
ECB	n/a
CBC	Last Cipher Block AES.OV0 = cipher[31:0] AES.OV1 = cipher[63:32] AES.OV2 = cipher[95:64] AES.OV3 = cipher[127:96]
CTR	Last Counter AES.OV0 = counter[31:0] AES.OV1 = counter[63:32] AES.OV2 = counter[95:64] AES.OV3 = counter[127:96]
CCM*	Encryption: MIC value If MIC is less than 32 byte, always MSB byte is used. Example: 8 byte MIC: AES.OV2 = MIC[31:0] AES.OV3 = MIC[63:32]
MMO	HASH Value AES.OV0 = HASH[31:0] AES.OV1 = HASH[63:32] AES.OV2 = HASH[95:64] AES.OV3 = HASH[127:96]
Bypass	n/a

12.3.9 AES Operation Pseudo Code

```
AES_Config_Type aesConfig

aesConfig.mode <- AES_MODE_CBC
aesConfig.encDecSel <- AES_MODE_ENCRYPTION
aesConfig.keySize <- keysize
aesConfig.mStrLen <- length

for i=1 to keysize do
aesConfig.key[i] = key[i]

for i=1 to 4 do
aesConfig.initVect[i] = vector[i]

while j < length or k < length do
 if AES input fifo not full  do
feed the data plain_text[j]
 j++

if AES output fifo not empty do
 read the encryption data
k++
```

12.4 References for AES Standard

[1] www.nist.gov/aes (AES development, historical site)

[2] http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf (AES standard and test vectors)

[3] http://csrc.nist.gov/groups/ST/toolkit/block_ciphers.html (overview over block ciphers, key handling and test vectors)

[4] Schneier, B.: Applied Cryptography, 2nd ed., Wiley1996

[5] Wobst, R.: Cryptology Unlocked, Wiley 2007

[6] http://en.wikipedia.org/wiki/Block_cipher_modes_of_operation

[7] http://en.wikipedia.org/wiki/CCM_mode

12.5 Register Description

See Appendix A.5, AES Address Block, on page 517 for a detailed description of the registers.

13 Cyclic Redundancy Check (CRC)

13.1 Overview

A Cyclic Redundancy Check (CRC) or polynomial code checksum is a hash function designed to detect data integrity. The 88MW300/302 CRC unit calculates a short, fixed-length binary sequence, known as the CRC code. For each block of data, CRC code and original data are sent or stored together. When a block of data is used, the same CRC calculation is processed. If the new CRC does not match the one pre-calculated earlier in the block of data, then the block contains a data error and the device may take corrective action, such as resending or requesting the block again. Otherwise the data is assumed to be error free (though, with some small probability, it may contain undetected errors; this is the fundamental nature of error-checking).

13.2 Features

A standard AHB slave interface is used to configure the module, receive the bit stream, and output the CRC result.

- Supports 32-bit parallel bit stream input, and supports up to 32-bit CRC output
- Supports up to 2^32 (4294967296) byte length to calculate CRC
- Supports the following CRC standards
  - CRC-16-CCITT, the polynomial is x^16+x^12+x^5+1
  - CRC-16-IBM, the polynomial is x^16+x^15+x^2+1
  - CRC-16-T10-DIF, the polynomial is x^16+x^15+x^11+x^9+x^8+x^7+x^5+x^4+x^2+x+1
  - CRC-32-IEEE 802.3, the polynomial is x^32+x^26+x^23+x^22+x^16+x^12+x^11+x^10+x^8+x^7+x^5+x^4+x^2+x+1
  - CRC-16-DNP, the polynomial is x^16+x^13+x^12+x^11+x^10+x^8+x^6+x^5+x^2+1

13.3 CRC Operation Flow

1. Disable CRC (CRC.CTRL.ENABLEto 0).
2. Disable the interrupt (set CRC.IMR.MASK to 1).
3. Clear all the interrupts (set CRC_ICR.CLEAR to 1).
4. Configure the stream length (set register CRC.STREAM_LEN_M1).
5. Configure CRC mode (set bit CRC.CTRL.MODE).
6. Enable the interrupt (set CRC.IMR.MASK to 0).
7. Enable CRC (set CRC.CTRL.ENABLE to 1).
8. Write stream in and waiting for interrupt to occur. (If interrupt occurred, go to 9)
9. Get CRC calculation result (Read register CRC.RESULT).
10. Complete CRC operation.

Note: The CRC input stream registers accepts a word (32-bit) at a time. If the input data is not 4 bytes aligned, pad 0's at the start of the data stream. For example, if the data stream consists of 5 bytes starting from the lower address: 0xA1, 0xA2, 0xA3, 0xA4, 0xA5.

Write the following words to the stream input register:

- 0xA1000000
- 0xA5A4A3A2

The CRC result bit order is:

16-bit CRC: x_0~x_{15} [msb->lsb]

32-bit CRCt: x_0~x_{31} [msb->lsb]

13.4 Register Description

See Appendix A.6.2, CRC Registers, on page 537 for a detailed description of the registers.

14 Universal Asynchronous Receiver Transmitter (UART)

14.1 Overview

The 88MW300/302 Universal Asynchronous Receive Transmitter (UART) supports 16550A and 16750 functions. It also includes a slow infrared Transmit encoder and Receive decoder that conforms to the Infrared Data Association (IrDA) Serial Infrared (SIR) specification.

14.2 Features

- Compliance to the AMBA specification (Rev 2.0)
- Programmable use of UART or IrDA SIR input/output
- Separate 64x8 transmit and 64x11 receive FIFO memory buffers to reduce CPU interrupts
- Supports 8-bit or 32-bit peripheral bus
- Programmable FIFO disabling for 1-byte depth
- Programmable baud rate generator
- Ability to add or delete standard asynchronous communication bits (start, stop, and parity) in the serial data
- Independently controlled transmit, receive, line status, and data-set interrupts
- Supports modern control functions: CTS and RTS
- Auto-flow capability control data I/O without generating interrupt
  - RTS (output) controlled by the UART Receive FIFO
  - CTS (input) from modern control UART transmitter
- Programmable serial interface
  - 5 to 8-bit characters
  - Even, odd, or no parity detection
  - 1 or 2 stop-bit generation
  - Baud-rate generation up to F(uart)/16 bps
  - False start-bit filter
- Line break generation and detection
- Internal diagnostic capabilities that include:
  - Loopback control for communications link fault isolation
  - Break, parity, and framing-error simulation
- Separate DMA requests for Transmit and Receive data services

14.3 Interface Signal Description

14.3.1 External Interface

Table 129 shows the interface signals from the UART to the I/O pins of the device.

Table 129: Pad Interface Signals

Signal Name	Type	Source/Destination	Description
UART_TXD	O	PAD	UART Transmit serial data
UART_RXD	I	PAD	UART Receive serial data
UART_CTSn	I	PAD	UART Clear To Send modem status (active low)
UART_RTSn	O	PAD	UART Request To Send modem status (active low)

14.3.2 Internal Interface

Table 130 shows the interface signals.

Table 130: Internal Interface Signals

Signal Name	Type	Source/Destination	Description
rst_uart_n	I	Reset controller	UART reset signal to clk_uart clock domain (active low)
clk_uart	I	Clock generator	UART reference clock
uart_int	O	Interrupt controller	UART interrupt (active high), a single combined interrupt generated as an OR function of all interrupts
dma_tx_ack	I	DMA controller	UART transmit DMA acknowledge signal (active high)
dma_tx_finish	I	DMA controller	UART transmit DMA finish signal (active high)
dma_tx_single	O	DMA controller	UART transmit DMA signal request (active high)
dma_tx_request	O	DMA controller	UART transmit DMA burst request (active high)
dma_rx_ack	I	DMA controller	UART receive DMA acknowledge signal (active high)
dma_rx_finish	I	DMA controller	UART receive DMA finish signal (active high)
dma_rx_single	O	DMA controller	UART receive DMA signal request (active high)
dma_rx_request	O	DMA controller	UART receive DMA burst request (active high)

14.3.3 AMBA APB Interface

Table 131 shows the interface signals.

Table 131: AMBA APB Interface Signals

Signal Name	Type	Source/Destination	Description
clk_apb	I	Clock generator	APB clock, used to time all bus transfers

Table 131: AMBA APB Interface Signals (Continued)

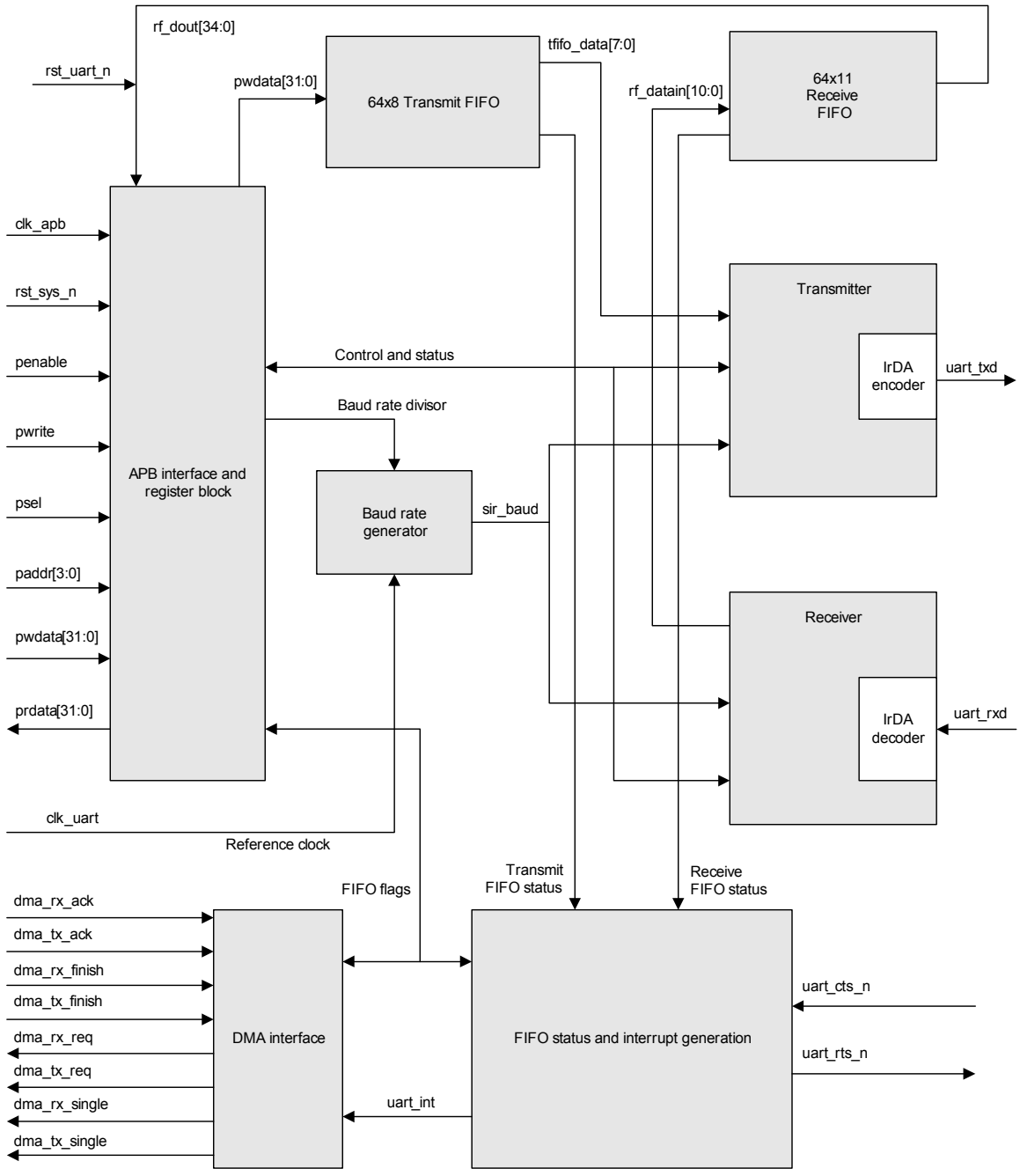
Signal Name	Type	Source/Destination	Description
rst_sys_n	I	Reset controller	Bus reset (active low)
penable	I	APB	APB enable PENABLE is asserted high for 1 cycle of clk_apb to enable a bus transfer
pwrite	I	APB	APB transfer direction signal Indicates a write access when high, read access when low.
psel	I	APB	UART and SIR select signal from decoder When set high, indicates the slave device is selected by the AMBA APB bridge, and that a data transfer is required.
paddr[3:0]	I	APB	Subset of AMBA APB address bus
pwdata[31:0]	I	APB	Subset of unidirectional AMBA APB write data bus
prdata[31:0]	O	APB	Subset of unidirectional AMBA APB read data bus

14.4 Function Description

14.4.1 Block Diagram

Figure 55 shows an overall block diagram.

Figure 55: UART Block Diagram



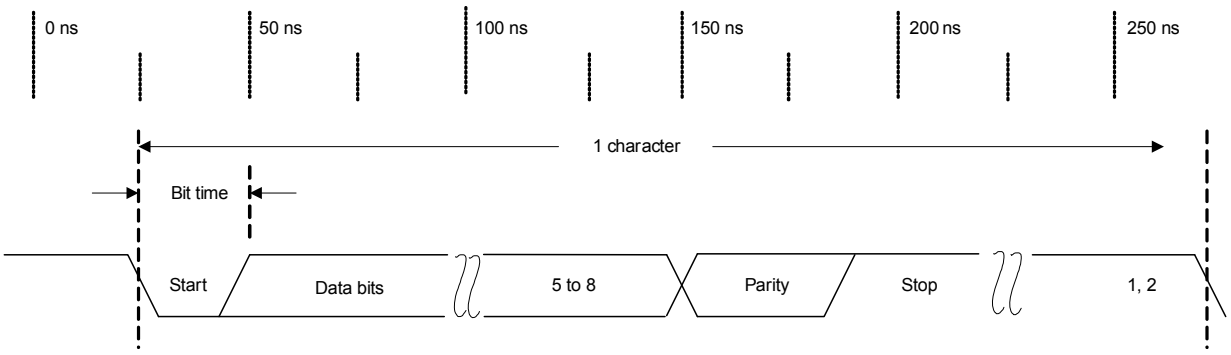
14.4.2 UART Operation

14.4.2.1 Data Format

Serial communication between the UART and a selected device is asynchronous. Therefore, additional bits (start and stop) are added to the serial data to indicate the beginning and end. Utilizing these bits allows 2 devices to be synchronized. This structure of serial data—accompanied by start and stop bits—is referred to as a character.

Figure 56 shows the data format.

Figure 56: Serial Data Format



An additional parity bit can be added to the serial character. This bit appears after the last data bit and before the stop bit(s) in the character structure in order to provide the UART with the ability to perform simple error checking on the receive data.

The UART Line Control Register (UART_LCR) is used to control the serial character characteristics. The individual bits of the data word are sent after start bit, starting with the Least Significant bit (LSb). These are followed by the optional parity bit, followed by stop bit(s), which can be 1 or 2.

- Data bits – This field can have 5 to 8 bits, which is depend on the programmed value in LCR.WLS10.
- Parity and sticky bit – 3 bits in LSR register determine the existence or value of parity value. Table 132 shows a true table for the Sticky Parity (STKYP), Even Parity Select (EPS), and Parity Enable (PEN) bits of the LSR.

Table 132: Parity Truth Table

PEN	EPS	STKYP	Parity Bit (transmitted or checked)
0	x	x	Not transmitted or checked
1	1	0	Even parity
1	0	0	Odd parity
1	0	1	1
1	1	--	0

- Stop bits – 1 or 2 stop bits can be programmed in LSR.STB. It specifies the number of stop bits transmitted and received in each character. When receiving, the receiver checks only the first stop bit.

- Set break – If LSR.SB is set to 1, a low-level is continually output on the TXD output after completing transmission of current character. Acts only on the TXD pin and has no effect on the transmit logic.

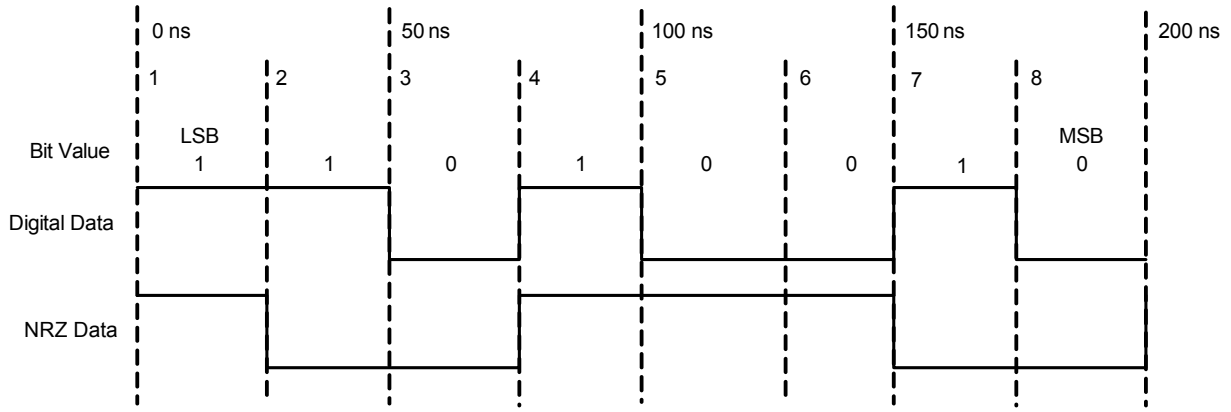
To the receiver, when a break (LSR.BI) is detected, a break interrupt (LSR.BI) is raised. A break interrupt is cleared when the CPU reads the LSR register or receiver detects a high (idle value). In FIFO mode, only 1 character equal to 0x00 is loaded into the FIFO regardless of the length of the break condition. In non-FIFO mode, 1 character equal to 0x00 is stored in RBR register. Receiver can resume when the receive sequence exit form break condition.

14.4.2.2 NRZ Coding

The UART can use NRZ coding to represent individual bit values. To enable NRZ coding, set the <NRZ Coding Enable> field in the Interrupt Enable Register. A bit value of 1 is represented by a line transition, and 0 is represented by no line transition.

Figure 57 shows the data byte 0b0100_1011 in NRZ coding. The LSb in the byte is transmitted first.

Figure 57: Example NRZ Bit Encoding – 0b0100_1011



Note: The NRZ cannot be used in infrared mode. NRZ encoding/decoding is only applied to the data bits. Start, parity, and stop bits are not involved.

14.4.2.3 Baud Rate Divider

The UART contains a programmable baud-rate generator that can take a fixed input clock and divide it down to generate the preferred baud rate. The baud rate is calculated by taking an example 14.7456 MHz fixed-input clock and dividing it by the Divisor Latch registers.

The baud-rate generator output frequency is 16 times the baud rate. There are 2, 8-bit Divisor Latch Registers that store the divisor in a 16-bit binary format.

The baud rate of the data shifted into or out of a UART is given by the formula:

BaudRate = (14.857 MHz) / (16 x Divisor)

Table 133: Baud Rate Divider

Required Baud Rate	Divisor	Actual Baud Rate @ 14.857 MHz
9600	96	9673
19200	48	19345
38400	24	38690
57600	16	58036
115200	8	116071
230400	4	232143
460800	2	464286
921600	1	928571

The divisor reset value is 0x0002. 0 is a meaningless value and is forbidden. Changing the baud rate is not permitted while actively transmitting or receiving data.

14.4.3 IrDA 1.0 SIR Operation

The SIR interface is used to support 2-way wireless communication that uses infrared transmission. The SIR interface provides a Transmit encoder and Receive decoder to support a physical link that conforms to the IrDA SIR specification.

The SIR interface does not contain the actual the actual IR LED driver or the receive amplifier. The I/O pins attached to the SIR interface (UART_RXD/UART_TXD) have only digital CMOS-level signals. SIR supports 2-way communication, but full-duplex communication is not possible because reflections from the transmit LED enter the receiver. Half-duplex is functional, and the receiver works when both transmitter and receiver SIR mode are enabled.

SIR supports frequencies up to 115.2 Kbps. The baud divisor must be 8 or more because the input clock is 14.7456 MHz.

The SIR modulation technique works with 7- or 8-bits characters with an optional parity bit. The data is preceded by 0-value start bit and ends with 1 or 2 stop bits. The role of the SIR ENDEC is to provide a digital encoded output and decoded input to the UART.

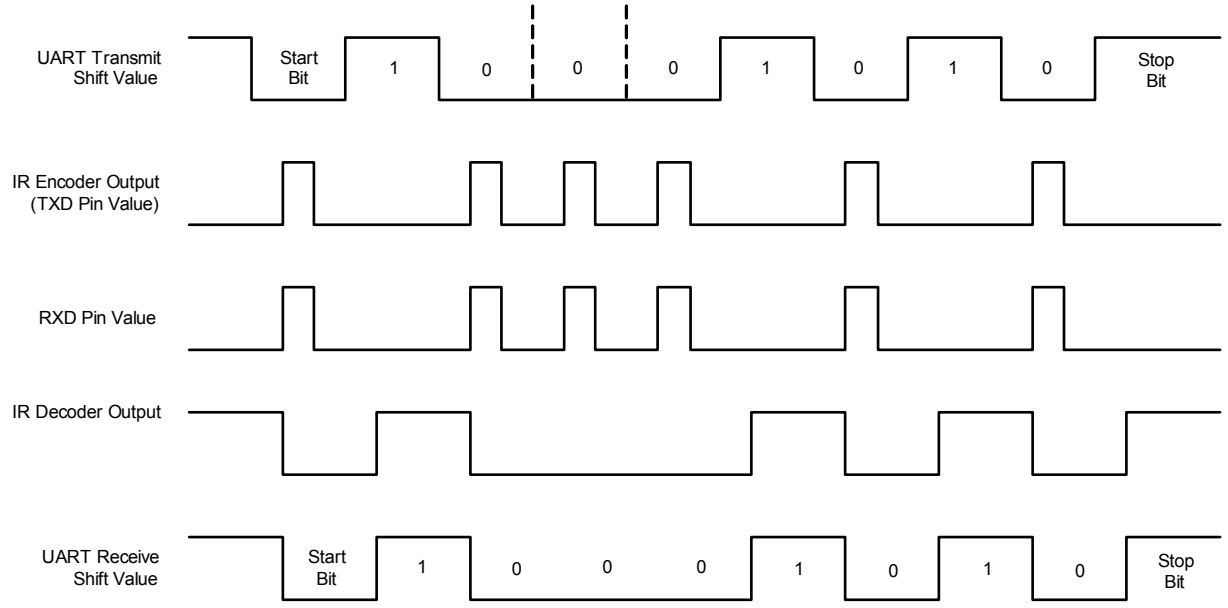
Modes of operation include:

- Normal mode
- Low-power mode

14.4.3.1 Normal Mode

In normal mode, the encoding scheme sends a pulse 3/16 of a bit wide in the middle of every 0-value bit, and sends no pulses for bits with a value of 1. These levels control the driver of an infrared transmitter, sending a pulse of light for each 0. On the reception side, the incoming light pulses energize the photo transistor base of the receiver, pulling its output LOW. The drives the RXD signal LOW. The pulse for each 0-value bit must occur, even for consecutive bits with no edge between them. Figure 58 shows an example.

Figure 58: IR Transmit and Receive Example



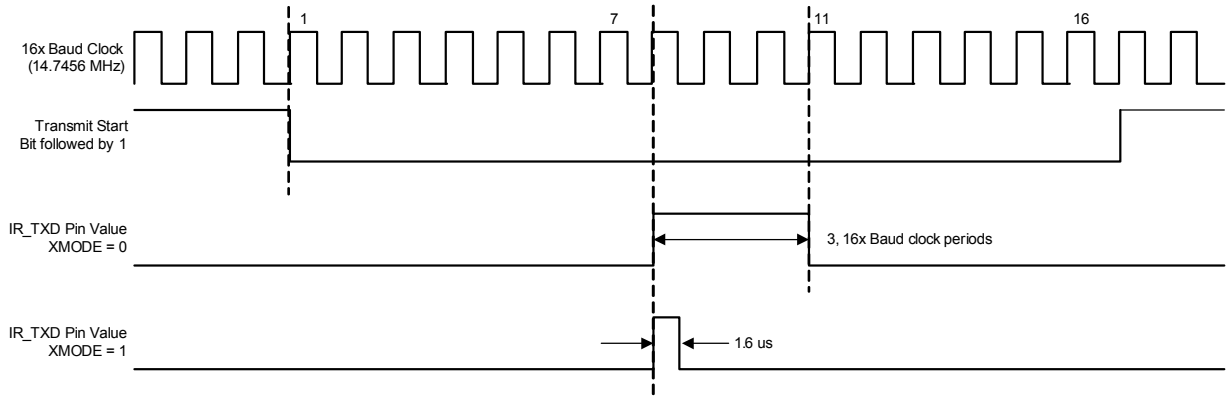
The last line is the same as the first, but it is shifted half a bit period. When the <Transmit Pulse Width Select> is clear, each 0 bit has a pulse width of 3/16 of a bit time.

14.4.3.2 Low-Power Mode

It is strongly recommended to set the <Transmit Pulse Width Select> field in the Infrared Selection Register for the transmit-pulse width. The shorter infrared pulse generated when the <Transmit Pulse Width Select> field is set reduces the LED power consumption. At 2400 bps when the UART frequency is 14.7456 MHz, the LED is normally on 78 us for each 0-bit transmitted. When the <Transmit Pulse Width Select> field is set, the LED is on for only 1.6 us.

Figure 59 shows an example.

Figure 59: XMODE Example



In generalization, we use to denote the UART functional frequency, in order to make IR correct functionality, divider equal or bigger than 8 is required. When *N* is denoted as the divider, then the pulse width under IrDA can be shown as follows:

$$\frac{1}{\frac{F_{UART}}{16 \times N}} \times \frac{3}{16}$$

When *N* = 8, the pulse width will get minimal width. In IrDA normal mode, pulse width is calculated with the actual divider. In IrDA low-power mode, pulse width is calculated with divider 8 regardless of the actual divider.

<Transmit Pulse Width Select>=1 toggles only the transmit-pulse width. It does not affect the receive-pulse width, which is always take the received sequence as generated in low-power mode.

To prevent transmitter LED reflection feedback to the receiver, disable the IR receive decoder when the IR Transmit encoder transmits data, and disable the IR Transmit encoder when the IR Receive decoder receives data. UART_SCR.RCVEIR and UART_SCR.XMITIR must not be set at the same time.

14.4.4 Clock Support

The frequency selected for the UART functional clock must accommodate the required range of band rates:

- Fuart (min) >= 16 x baud_rate (max)
- Fuart (max) <= 16 x 65535 x baud_rate (min)

For example, for a range of baud rates from 110 baud to 460800 baud the UART clock frequency must be between 7.3728 MHz to 115.34 MHz.

There is also a constraint on the ratio of clock frequencies for PCLK to UART clock. The frequency of the UART clock and the frequency of PCLK must at least meet the following constraints:

$$\frac{1}{F_{UART}} \times 16 \times \mathrm{DL}_{divisor} \times \mathrm{L}_{character} > \frac{1}{F_{apb}} \times 14$$

14.4.5 Reset

The UART is disabled on reset. To enable the UART, software must program the <UART Unit Enable> field in the Interrupt Enable Register. When the UART is enabled, the receiver waits for a frame start bit and the transmitter send data if it is available in the Transmit Holding Register. Transmit data can be written to the Transmit Holding Register before the UART unit is enabled. In FIFO mode, data is transmitted from the FIFO to the pin.

When UART unit is disabled, the transmitter or receiver finishes the current byte and stops transmitting or receiving more data. Data in the FIFO is not cleared and transmission resumes when the UART is enabled.

14.4.6 FIFO Operation

When FIFO_MODE is enabled, data written to the Transmit Holding Register by either the CPU or DMA is automatically transferred to the transmit FIFO. When reading the Receive Buffer Register by either CPU or DMA, the data in the Receive FIFO is transferred automatically from the FIFO data register.

UART has a Transmit FIFO and Receive FIFO, with each FIFO holding 64 characters of data. The FIFOs are filled or emptied by CPU or DMA transactions. The data is accessed through the TXFIFO and RXFIFO. The Processor accesses are normally triggered by an interrupt caused by an UART Interrupt Identification Register (UART_IIR). The data writes to the TXFIFO are either 8bits or 32bits wide. The processor always reads 32bits from RXFIFO, with 0 inserted in the MSb, down to the programmed data size.

The TXFIFO and RXFIFO are each accessed as 1, 32-bit location by the CPU. For data transmission, the UART port transmits the data from the TXFIFO to the external peripheral through the UART.txd interface. Data received from the external peripheral through the UART.rxd interface is converted to parallel bytes and written into the RXFIFO.

The TXFIFO and RXFIFO are differentiated by whether the access is Read or a Write transfer. Reads from the Receive Buffer Register automatically target the RXFIFO. Writes to the Transmit Holding Register automatically target the TXFIFO. From the memory-map perspective, the TXFIFO and the RXFIFO are at the same physical address.

14.4.6.1 Transmit FIFO

The transmit FIFO is an 8-bit wide, 64-location deep, FIFO memory buffer. Data written across the APB interface is stored in the FIFO until read out by the transmit logic.

TXFIFO can be accessed by both Processor and DMA bursts, setting TIL causes transmitter interrupts and DMA requests to occur when the transmitter FIFO is empty. Clearing TIL causes transmitter interrupts and DMA requests to occur when the transmitter FIFO is half empty. The trigger level must be bigger or equal to the DMA burst length programmed in the DMA registers.

The transmit FIFO can be disabled to act like a 1-byte holding register.

14.4.6.2 Receive FIFO

The receive FIFO is an 11-bit wide, 64 location deep, FIFO memory buffer. Received data and corresponding error bits are stored in the receive FIFO by the receive logic until read out across the APB interface.

When the number of bytes in the RXFIFO equals the interrupts trigger level programmed into this field and the receive-data-available interrupt is enabled (with Interrupt Enable Register), an interrupt is generated and appropriate bits are set in the Interrupt Identification Register. The receive DMA request is generated as well when trigger level is reached. The trigger level must be bigger or equal to the DMA burst size programmed in the DMA registers.

The receive FIFO can be disabled to act like a 1-byte holding registers.

14.4.6.3 32-Bit Peripheral Bus

UART supports an 8- (default) and 32-bit peripheral bus. If a 32-bit bus is preferred, the bytes are written in Little Endian format (7:0) with byte 3 (the most recent byte) starting at bit[31], byte 2 starting at bit[23], and so on.

- 8-bit mode – only the least significant byte contains valid data on the peripheral bus. The upper 24 bits are ignored.
- 32-bit mode – UART can only read or write partial words of 1, 2, 3 or 4 continuous bytes from the peripheral bus. In this mode, UART always transmit or receive byte 0 first, and then byte 1 and so on. The Receive FIFO and Transmit FIFO must be enabled when in 32-bit mode.

14.4.7 DMA Support

The UART provides an interface to connect to a DMA controller. The DMA operation of the UART is controlled using the DMA Control Register. There are 2 types of request UART can signal to DMA: burst transfer request and single transfer request.

The burst transfer and single transfer request signals are not mutually exclusive, they can both be asserted at the same time. When there is more than the watermark level in the RXFIFO, the burst transfer request and the single transfer request are asserted. When the amount of data left in the RXFIFO is less than the watermark level, the single request only is asserted. This is useful for situation where the number of characters left to be received in the stream is less than a burst.

If you disable the FIFO mode in the UART then it operates in non-FIFO mode and only the DMA single transfer mode can operate, because only 1 character can be transferred to, or from the FIFOs at any time.

When the UART is in FIFO mode, data transfers can be made by either single or burst transfers depending on the programmed watermark level and the amount of data in the FIFO. Table 3 3 and Table 3 4 list the trigger level for the transmit and receive FIFOs.

Table 134 shows trigger levels for the transmit FIFO.

Table 134: DMA Trigger Points for Transmit FIFO

Transmitter Trigger Level (TTL)	Transmit (number of empty locations)
0	32
1	64

Table 135 shows trigger levels for the receive FIFO.

Table 135: DMA Trigger Points for Receive FIFO

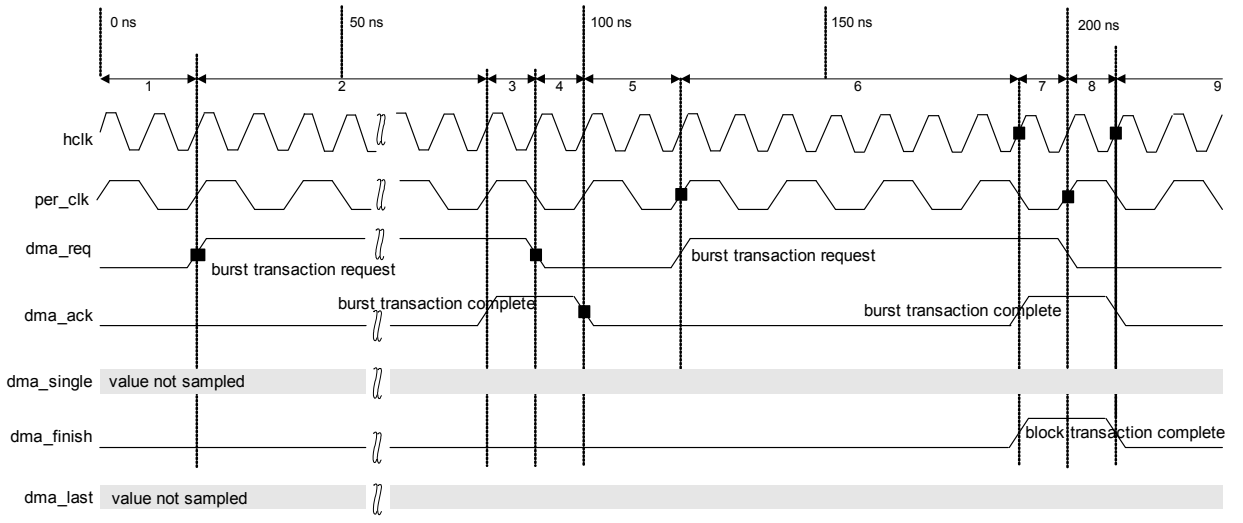
Transmitter Trigger Level (TTL)	Transmit (number of empty locations)
00	1
01	8
10	16
11	32

To prevent overflow of the TXFIFO or underflow of the RXFIFO when using the DMA, be careful when setting the FIFO trigger threshold levels by setting the DMA burst size and data width. TXFIFO overflow and RXFIFO underflow will cause data missing. The DMA burst size must be smaller or equal the trigger threshold.

Assume that DMAC is in another clock domain (hclk) different from UART clock domain (per_clk).

Figure 60 shows the timing diagram of a burst transaction where the UART clock, per_clk is half of the hclk frequency. In this example, the UART is outside the single transaction region, and therefore DMAC does not sample dma_single.

Figure 60: Burst Transaction – hclk=2*per_clk



When designing the hardware handshaking interface:

- Once asserted, the dma_req must remain asserted until the corresponding dma_ack signal is received, even if the condition that generates dma_req in the peripheral is False.
- The dma_req signal should be de-asserted when dma_ack is asserted, even if the condition that generates dma_req in the peripheral is True.

Figure 61 shows a single transaction that occurs in the single transaction region. The handshaking loop is as follows:

- dma_single asserted by peripheral
- dma_ack asserted by DMAC
- dma_single de-asserted by peripheral
- dma_ack de-asserted by DMAC

Figure 61: Signal Transaction – hclk=2*per_clk

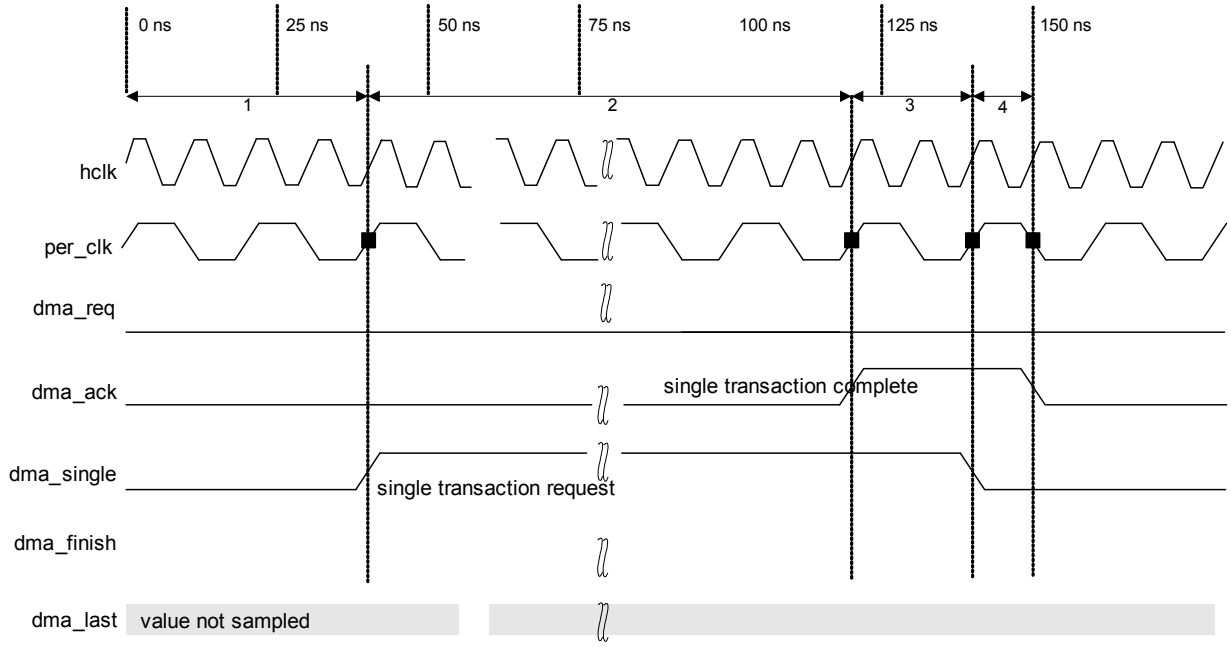
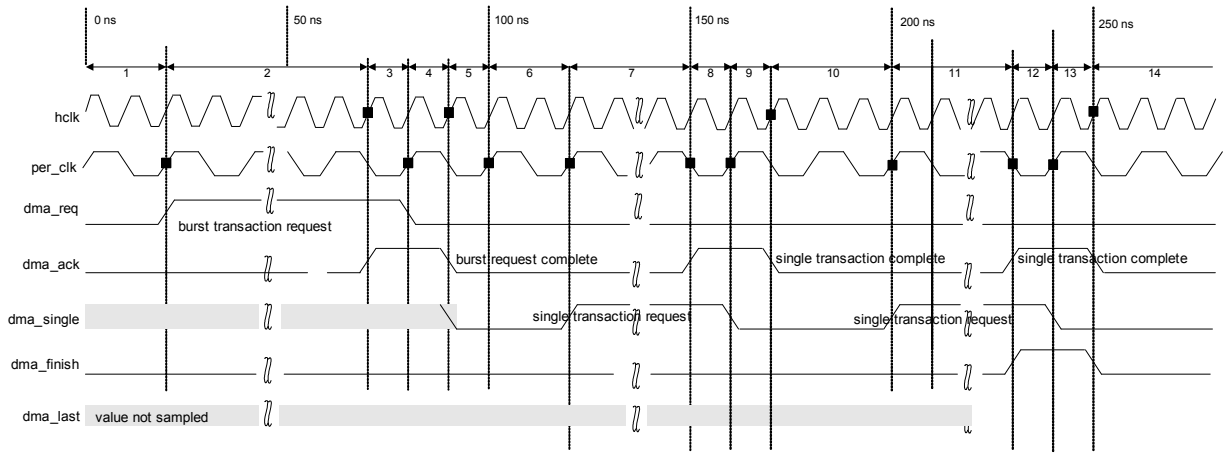


Figure 62 shows a burst transaction, followed by 2 back-to-back single transactions, where the hclk frequency is twice the per_clk frequency.

Figure 62: Burst Followed by Back-to-Back Single Transactions



After the first burst transaction, the UART enters the single transaction region and the DMAC starts sampling dma_single. DMAC samples that dma_single is asserted and performs single transactions. The second single transaction terminates the block transfer; dma_finish is asserted to indicate block completion.

14.4.8 UART Modem Operation

The UART supports the Data Terminal Equipment (DTE) mode of operation. Table 136 shows the signals.

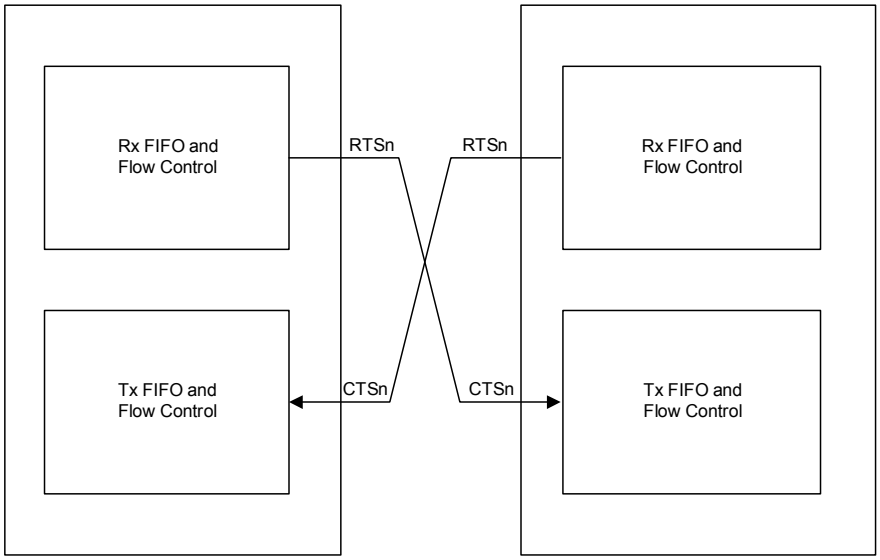
Table 136: Modem I/O Signals in DTE Mode

Signal Name	Type	Source/Destination	Description
UART_CTSn	I	PAD	UART Clear To Send modem status (active low)
UART_RTSn	O	PAD	UART Request To Send modem status (active low)

14.4.9 UART Hardware Flow Control

The hardware flow control feature is fully selectable, and enables you to control the serial data flow by using the uart_cts_n input and uart_rts_n output signals.

Figure 63: Hardware Flow Control



Auto-flow mode can be used in 2 ways: full auto-flow, automating both CTSn and RTSn; and half auto-flow, automating only CTSn. Table 137 shows the bits to enable RTS and CTS flow control both simultaneously and independently.

Table 137: Control Bits to Enable Hardware Flow Control

UART_MCR Register Bit		Description
AFE	RTS	
0	x	Both RTS and CTS flow control disabled
1	0	Only CTS flow control enabled
1	1	Both RTS and CTS flow control enabled

14.4.9.1 RTS Flow Control

The RTS flow control is linked to the programmable receive FIFO watermark levels. When RTS flow control is enabled, the uart_rts_n is asserted until the receive FIFO is filled up to the watermark level. When the receive FIFO watermark level is reached, the uart_rts_n signal is deasserted, indicating that there is no more room to receive any more data. The transmission of data is expected to cease after the current character has been transmitted.

The uart_rts_n signal is reasserted when data has been read out of the receive FIFO so that it is filled to less than the watermark level. If RTS flow control is disabled and the UART is still enabled, then data is still enabled, then data is received until the receive FIFO is full, or no more data is transmitted to it.

14.4.9.2 CTS Flow Control

If CTS flow control is enabled, then the transmitter checks the uart_cts_n signal before transmitting the next byte. If the uart_cts_n signal is asserted, it transmits the byte otherwise transmission does not occur.

The data continues to be transmitted while uart_cts_n is asserted, and the transmit FIFO is not empty. If the transmit FIFO is empty and the uart_cts_n signal is asserted no data is transmitted.

If the uart_cts_n signal is deasserted and CTS flow control is enabled, then the current character transmission is completed before stopping. If CTS flow control is disabled and the UART is enabled, then the data continues to be transmitted until the transmit FIFO is empty.

When auto-flow control is enabled in IrDA mode, since IrDA can't work in full-duplex mode, transmitter won't work when Both RTS and CTS flow control enabled.

14.4.10 Auto Baud Rate Detection

UART supports auto-baud-rate detection. When enabled, the UART counts the number of clock cycle within the start-bit pulse (a special sequencer with the LSb '1' is required). This number is then written into the Auto-Baud Counter Register and is used to calculate the baud rate. When the Auto-Baud Count Register is written, an auto-baud-lock interrupt is generated (if enabled), and the UART automatically programs the Divisor Latch Registers with the appropriate baud rate if ABR.ABUP is '1'. If preferred, the user can read the Auto-Baud Count Register and use this information to program the Divisor Latch Registers with a baud rate calculated by CPU, in this condition, ABR.ABUP need to '0'. After the baud rate has been programmed, the user verifies that the predetermined characters are being received correctly.

Software can use either of 2 methods for auto-baud calculation: table-based and formula-based.

- Formula method – any baud rate can be programmed by the UART. This method works well for higher baud rates, but it could fail below 28.8 Kbps if the remote transmitter's actual baud rate differs by more than 1 percent of its target.
- Table method – is more immune to such errors, because the table rejects uncommon baud rates and rounds to the common 1s. The table method allows any baud rate defined by the formula in (1) above 28.8 Kbps. Below 28.8 Kbps, the only baud rates that can be programmed by the UART are 19200, 14400, 9600, 4800, 1200, and 300 baud.

When the baud rate is detected, the auto-baud circuitry disables itself by clearing the <ABE> field in the Auto-Baud Count Register. To re-enable auto-baud detection, set the <ABE> field again.

Changing the baud rate is not permitted when actively transmitting or receiving data. Auto-baud-rate detection is not supported in IrDA mode.

14.4.11 Interrupts

There are 7 maskable interrupts generated in the UART. These interrupts are combined to produce 1 interrupt output that is the OR of the individual outputs:

- Receive timeout interrupt
- Modem state change interrupt, that can be caused by delta clear to send
- Receive line status interrupt, that can be caused by:
  - Overrun error
  - Parity error
  - Framing error
  - Break interrupt
- Receive FIFO error interrupt
- Transmit data request interrupt
- Receive data available interrupt
- Auto-baud-lock interrupt
- UART_INT, this is an OR function of the 7 individual masked interrupt

Enable or disable individual interrupts by changing the mask bit in the Interrupt Enable Register (UART_IER) and Auto-Baud Control Register (UART_ABR). Setting the appropriate mask bit HIGH enables the interrupt.

The status of the individual interrupt sources can be read either from the Interrupt Identification Register (UART_IIR), Line Status Register (UART_LSR) and Modem Status Register (UART_MSR).

14.4.11.1 Receive Timeout Interrupt

The receive timeout interrupt is asserted when the trailing bytes are programmed to remove by the processor (UART_FCR.TRAIL) and receive FIFO is not empty, and no more data is received during a 32-bit period. The receive timeout interrupt is cleared either when the FIFO becomes empty through reading all the data (or by reading the holding register).

14.4.11.2 Modem State Change Interrupt

The modem status interrupt is asserted if any of the modem status signals change. It is cleared by writing a 0 to corresponding bit(s) in the Modem Control Register, depending on the modem status signals that generated the interrupt.

14.4.11.3 Receive Line Status Interrupt

The receive line status interrupt is asserted when an error occurs in the reception of data by the UART. The interrupt can be caused by a number of different error conditions:

- Overrun
- Framing
- Parity
- Break

Determine the cause of the interrupt by reading the Interrupt Identification Register (UART_IIR) and Line Status Register (UART_LSR).

14.4.11.4 Receive FIFO Error Interrupt

The receive FIFO error interrupt is asserted when an error occurs in the Receive FIFO, this can be either an invalid data received from pin RXD or try to read data from an error address in the FIFO.

14.4.11.5 Transmit Data Request Interrupt

The transmit interrupt changes state when any of the following events occurs:

- If the FIFO mode is enabled and the transmit FIFO is equal to or lower than the programmed trigger level then the transmit interrupt is asserted HIGH. The transmit interrupt is cleared by writing data to the transmit FIFO until it becomes greater than the trigger level.
- If the FIFO mode is disabled (have a depth of 1 location), and there is no data present in the transmitter single location, the transmit interrupt is asserted HIGH. It is cleared by performing a single write to the transmit FIFO.

14.4.11.6 Receive Data Available Interrupt

The receive interrupt changes state when any of the following events occurs:

- If the FIFO mode is enabled and the receive FIFO reaches the programmed trigger level. When this happens, the receive interrupt is asserted HIGH. The receive interrupt is cleared by reading data from the receive FIFO until it becomes less than the trigger level.
- If the FIFO mode is disabled (have a depth of 1 location) and data is received thereby filling the location, the receive interrupt is asserted HIGH. The receive interrupt is cleared by performing a single read of the receive FIFO.

14.4.11.7 Auto Baud Lock Interrupt

The Auto-Baud-Lock interrupt changes state when Divisor Latch Register programmed by auto-baud circuitry in Auto-baud-rate detect mode.

14.4.11.8 UART_INT

The masked interrupts are also combined into a single output; that is an OR function of the individual masked sources. Connect this output to a system interrupt controller.

The combined UART interrupt is asserted if any of the individual interrupts are asserted and enabled and enabled.

UART_INT is masked by OUT2 Signal Control (UART_MCR.OUT2) which connects the UART interrupt, only when either OUT2 or loopback mode enable is set 1, the UART_INT can be asserted.

14.5 Register Description

See Appendix A.10.2, UART Registers, on page 630 for a detailed description of the registers.

15 Inter-Integrated Circuit (I^2C)

15.1 Overview

The 88MW300/302 I^2C bus interface complies with the common I^2C protocol and can operate in standard mode (with data rates up to 100 Kbps), fast mode (with data rates up to 400 Kbps) and high-speed mode (with data rates up to 2 Mbps). Additionally, high-speed mode devices and fast mode devices are downward compatible. It also supports DMA capability.

The device supports 2, I^2C interfaces, I2C0 and I2C1, both of which are identical in function.

15.2 Features

- 2 I^2C serial interfaces consisting of a serial data line (SDA) and serial clock (SCL)
- 3 speeds include:
  - Standard mode (up to 100 Kbps)
  - Fast mode (up to 400 Kbps)
  - High-speed mode (2 Mbps)
- Clock synchronization
- Master or Slave I^2C operation, multi-master, multi-slave operation, and arbitration support
- 7- or 10-bit addressing and General Call
- 7- or 10-bit combined format transfers
- Bulk transmit mode in slave
- 16 * 32 bits deep transmit and receive buffers, respectively
- Interrupt operation
- DMA function support

15.3 Interface Signal Description

Table 138 shows the interface signals.

Table 138: I^2C Interface Signals

Signal Name	Type	Description
SDA	I/O	Data signal line (Serial Data)
SCL	I/O	Clock signal line (Serial Clock)

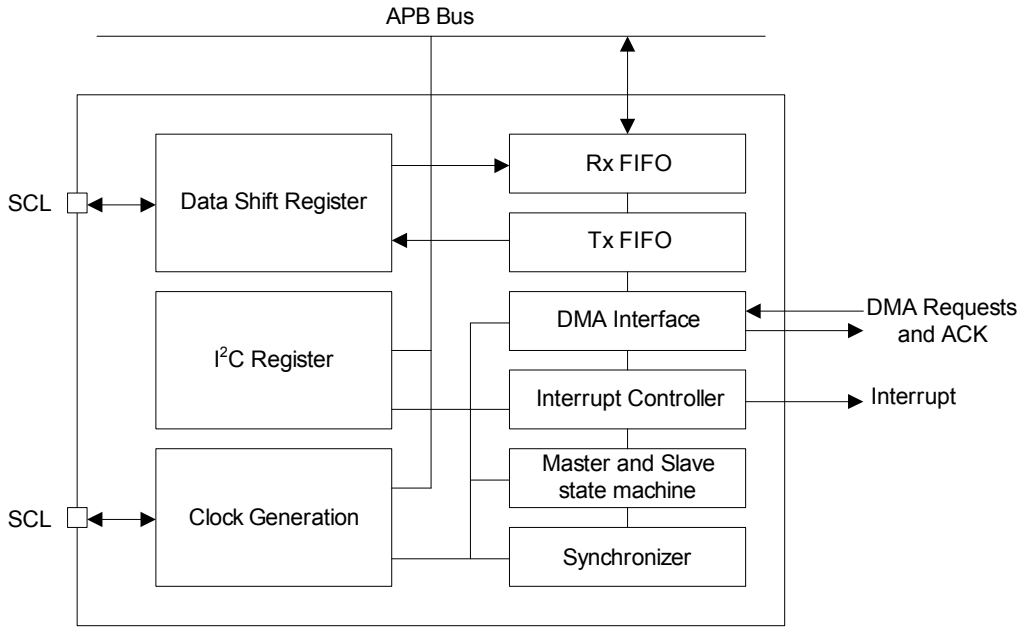
15.4 Functional Description

15.4.1 Block Diagram

The I^{2}C consists of an APB slave interface, an I^{2}C interface, and FIFO logic to maintain coherency between the interfaces.

Figure 64 shows a simplified block diagram.

Figure 64: I2C Block Diagram



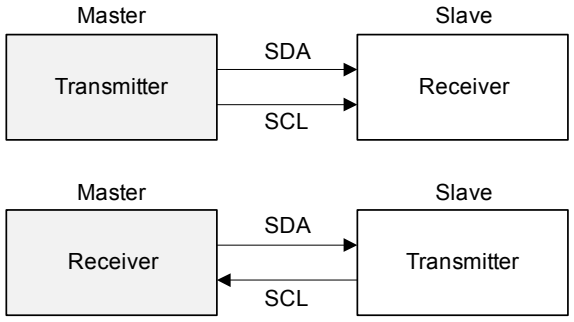
15.4.2 I²C Bus Terminology

Table 139 shows bus terminology.

Table 139: I²C Bus Terminology

I²C Device	Description
Transmitter	Sends Data Over I²C Bus Transmitter can either be a device that initiates the data transmission to the bus (a master-transmitter) or responds to a request from the master to send data to the bus (a slave-transmitter).
Receiver	Receives Data Over I²C Bus A receiver can either be a device that receives data on its own request (a master-receiver) or in response to a request from the master (a slave-receiver).
Master	Component that initiates a transfer (START command), generated the clock (SCL) signal and terminates the transfer (STOP command). A master can be either a transmitter or a receiver.
Slave	Device addressed by the master. A slave can be either a receiver or transmitter (see Figure 65, Master/Slave and Transmitter/Receiver Relationship, on page 233).
Multi-master	Ability for more than 1 master to co-exist on the bus at the same time without collision or data loss.
Arbitration	Predefined procedure that authorizes only 1 master at a time to take control of the bus. Refer to "Multiple Master Arbitration" for more information.
Synchronization	Predefined procedure that synchronizes the clock signals provided by 2 or more masters. For more information about this feature, refer to “Clock Synchronization.”
Bus Transfer Terminology ***The following terms are specific to data transfers that occur to/from the I²C bus.***	
START (RESTART)	Data transfer begins with a START or RESTART condition. The level of the SDA data line changes from high to low, while the SCL clock line remains high. When this occurs, the bus becomes busy. **NOTE:** START and RESTART conditions are functionally identical.
STOP	Data transfer is terminated by a STOP condition that occurs when the level on the SDA data line passes from the low state to the high state, while the SCL clock line remains high. When the data transfer has been terminated, the bus is free or idle once again. The bus stays busy if a RESTART is generated instead of a STOP condition.

Figure 65: Master/Slave and Transmitter/Receiver Relationship



15.4.3 I²C Behavior

The I²C can be controlled by software to be either:

- I²C master only, communicating with other I²C slaves
- I²C slave only, communicating with 1 or more I²C master

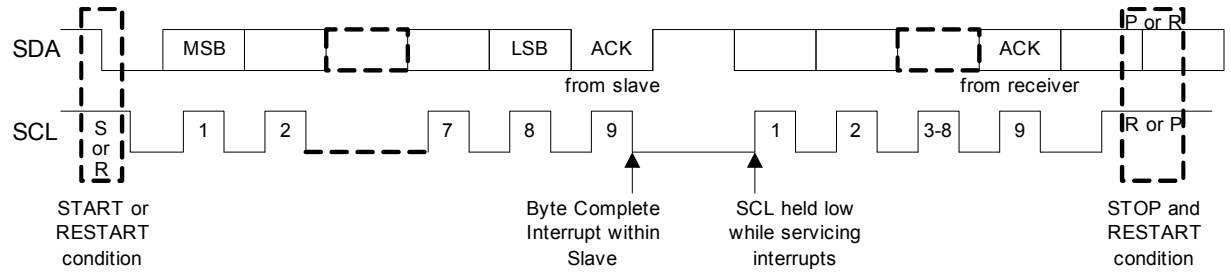
The master is responsible for generating the clock and controlling the transfer of data. The slave is responsible for either transmitting or receiving data to/from the master. The acknowledgement of data is sent by the device that is receiving data, which can be either a master or a slave. The I²C protocol also allows multiple masters to reside on the I²C bus and uses an arbitration procedure to determine bus ownership.

Each slave has a unique address that is determined by the system designer. When a master wants to communicate with a slave, the master transmits a START/RESTART condition that is then followed by the slave's address and a control bit (R/W) to determine if the master wants to transmit data or receive data from the slave. The slave then sends an acknowledge (ACK) pulse after the address.

If the master (master-transmitter) is writing to the slave (slave-receiver), the receiver gets 1 byte of data. This transaction continues until the master terminates the transmission with a STOP condition. If the master is reading from a slave (master-receiver), the slave transmits (slave-transmitter) a byte of data to the master, and the master then acknowledges the transaction with the ACK pulse. This transaction continues until the master terminates the transmission by not acknowledging (Negative Acknowledgement (NACK)) the transaction after the last byte is received, and then the master issues a STOP condition or addresses another slave after issuing a RESTART condition.

Figure 66 shows the data transfer.

Figure 66: Data Transfer on the I²C Bus



The I²C is a synchronous serial interface. The SDA line is a bidirectional signal and changes only while the SCL line is low, except for STOP, START, and RESTART conditions. The output drivers are open-drain or open-collector to perform wire-AND functions on the bus. The maximum number of devices on the bus is limited by only the maximum capacitance specification of 400 pF. Data is transmitted in byte packages.

> ***Note:*** Placing data into the FIFO generates a START, and emptying the FIFO generates a STOP. For more information, see Section 15.4.3.1, START and STOP Generation, on page 235.

15.4.3.1 START and STOP Generation

When operating as a I^2C master, placing data into the Transmit FIFO causes the I^2C to generate a START condition on the I^2C bus. Allowing the Transmit FIFO to empty causes the I^2C to generate a STOP condition on the I^2C bus.

When operating as a slave, the I^2C does not generate START and STOP conditions, as per the protocol. However, if a read request is made to the I^2C, it holds the SCL line low until read data has been supplied to it. This action stalls the I^2C bus until read data is provided to the slave I^2C, or the I^2C slave is disabled by writing a 0 to ENABLE in register I2C.ENABLE.

15.4.3.2 Combined Formats

The I^2C I supports mixed read and write combined format transactions in both 7-bit and 10-bit addressing modes.

The I^2C does not support mixed address and mixed address format—that is, a 7-bit address transaction followed by a 10-bit address transaction or vice versa—combined format transactions.

To initiate combined format transfers, set the register CON.RESTART_EN to 1. With this value set and operating as a master, when the I^2C completes a I^2C transfer, it checks the Transmit FIFO and executes the next transfer. If the direction of this transfer differs from the previous transfer, the combined format is used to issue the transfer. If the Transmit FIFO is empty when the current I^2C transfer completes, a STOP is issued and the next transfer is issued following a START condition.

15.4.4 I²C Protocols

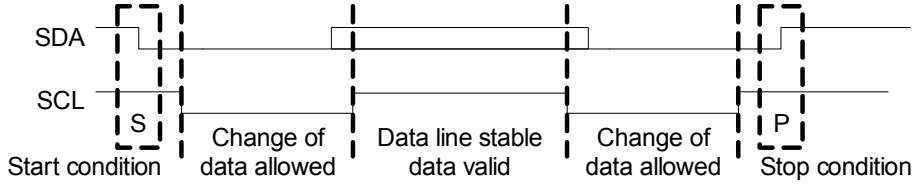
15.4.4.1 START and STOP Conditions

When the bus is idle, both the SCL and SDA signals are pulled high through external pullup resistors on the bus. When the master wants to start a transmission on the bus, the master issues a START condition. This is defined to be a high-to-low transition of the SDA signal while SCL is 1. When the master must terminate the transmission, it issues a STOP condition, which is defined to be a low-to-high transition of the SDA line while SCL is 1.

When data is being transmitted on the bus, the SDA line must be stable when SCL is 1.

Figure 67 shows the timing of the START and STOP conditions.

Figure 67: START and STOP Condition



Note: The signal transitions for the START/STOP conditions reflect those observed at the output signals of the Master driving the I²C bus. Use caution when observing the SDA/SCL signals at the input signals of the Slave(s), because unequal line delays may result in an incorrect SDA/SCL timing relationship.

15.4.4.2 Addressing Slave Protocol

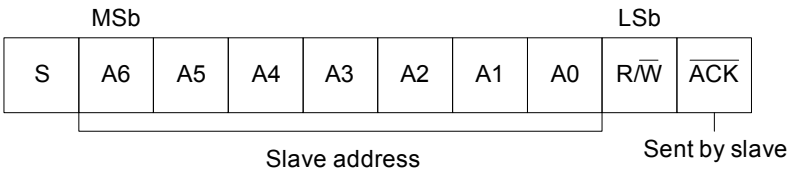
There are 2 address formats: the 7-bit and 10-bit address formats.

15.4.4.2.1 7-Bit Address Format

During the 7-bit address format, the first 7 bits (bits[7:1]) of the first byte set the slave address and the Least Significant bit (LSb) (bit[0]) is the R/W bit (as shown in Figure 68). When bit 0 (R/W) is set to 0, the master writes to the slave. When bit 0 (R/W) is set to 1, the master reads from the slave.

Figure 68 shows the 7-bit address format.

Figure 68: 7-Bit Address Format



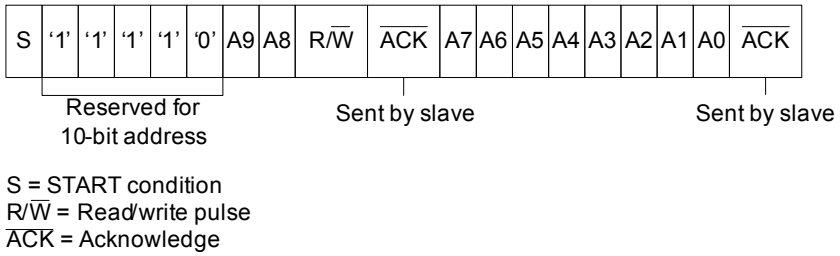
S = START condition
R/$\overline{W}$ = Read/write pulse
$\overline{ACK}$ = Acknowledge

15.4.4.2.2 10-Bit Address Format

During 10-bit addressing, 2 bytes are transferred to set the 10-bit address. The transfer of the first byte contains the following bit definition. The first 5 bits (bits[7:3]) notify the slaves that this is a 10-bit transfer followed by the next 2 bits (bits[2:1]), which set the slaves address bits 9:8, and the LSb (bit 0) is the /W bit. The second byte transferred sets bits 7:0 of the slave address.

Figure 69 shows the 10-bit address format.

Figure 69: 10-Bit Address Format



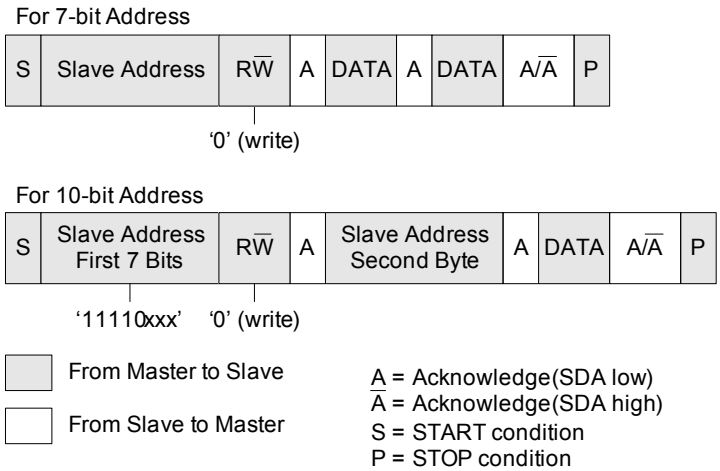
15.4.4.3 Transmitting and Receiving Protocol

The master can initiate data transmission and reception to/from the bus, acting as either a master-transmitter or master-receiver. A slave responds to requests from the master to either transmit data or receive data to/from the bus, acting as either a slave-transmitter or slave-receiver, respectively.

15.4.4.3.1 Master-Transmitter and Slave-Receiver

All data is transmitted in byte format, with no limit on the number of bytes transferred per data transfer. After the master sends the address and R/W bit or the master transmits a byte of data to the slave, the slave-receiver must respond with the acknowledge signal (ACK). When a slave-receiver does not respond with an ACK pulse, the master aborts the transfer by issuing a STOP condition. The slave must leave the SDA line high so that the master can abort the transfer.

Figure 70: Master-Transmitter Protocol

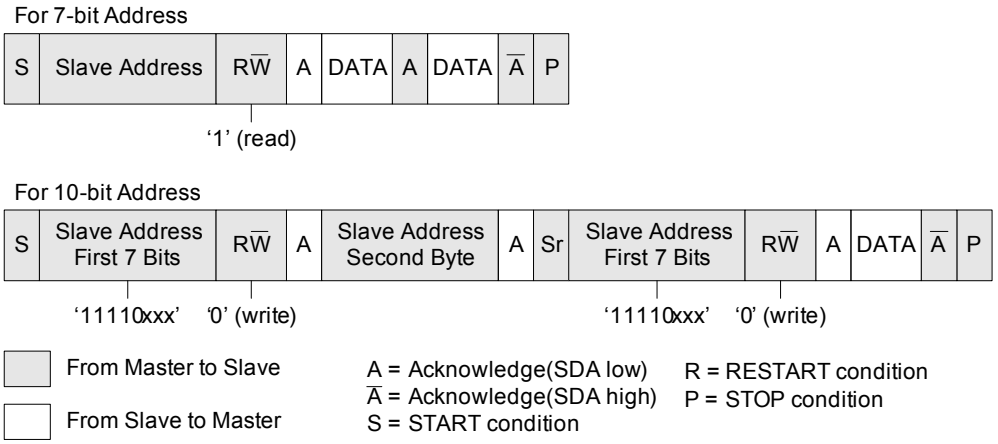


15.4.4.3.2 Master-Receiver and Slave-Transmitter

If the master is receiving data as shown in Figure 71, then the master responds to the slave-transmitter with an acknowledge pulse after a byte of data has been received, except for the last byte. This method is how the master-receiver notifies the slave-transmitter that this is the last byte. The slave-transmitter relinquishes the SDA line after detecting the Negative Acknowledgment (NACK) so that the master can issue a STOP condition.

When a master refuses to relinquish the bus with a STOP condition, the master can issue a RESTART condition. This is identical to a START condition except it occurs after the ACK pulse. Operating in master mode, the I^2C can then communicate with the same slave using a transfer of a different direction. For a description of the combined format transactions that the I^2C supports, refer to “Combined Formats.”

Figure 71: Master-Receive Protocol



15.4.5 Multiple Master Arbitration

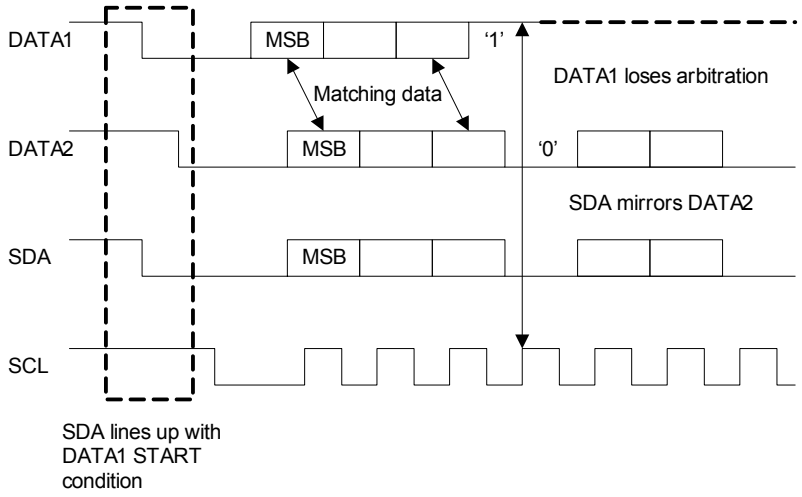
The I^2C bus protocol allows multiple masters to reside on the same bus. If there are 2 masters on the same I^2C bus, there is an arbitration procedure if both try to take control of the bus at the same time by generating a START condition at the same time. Once a master (for example, a microcontroller) has control of the bus, no other master can take control until the first master sends a STOP condition and places the bus in an idle state.

Arbitration occurs on the SDA line, while the SCL line is 1. The master, which transmits a 1 while the other master transmits 0, loses arbitration and turns off its data output stage. The master that lost arbitration can continue to generate clocks until the end of the byte transfer. If both masters are addressing the same slave device, the arbitration could go into the data phase.

The I^2C stops generating SCL when it has detected it has lost arbitration to another master.

Figure 72 shows the timing of when 2 masters are arbitrating on the bus.

Figure 72: Multiple Master Arbitration



Arbitration is not allowed between the following conditions:

- RESTART condition and a data bit
- STOP condition and a data bit
- RESTART condition and a STOP condition

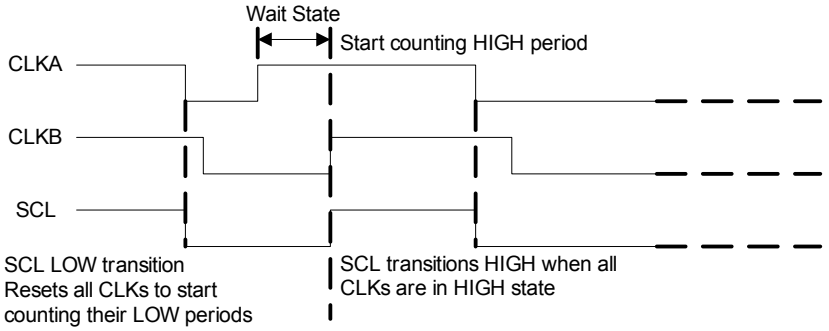
Slaves are not involved in the arbitration process.

15.4.6 Clock Synchronization

When 2 or more masters try to transfer information on the bus at the same time, they must arbitrate and synchronize the SCL clock. All masters generate their own clock to transfer messages. Data is valid only during the high period of SCL clock. Clock synchronization is performed using the wired-AND connection to the SCL signal. When the master transitions the SCL clock to 0, the master starts counting the low time of the SCL clock and transitions the SCL clock signal to 1 at the beginning of the next clock period. However, if another master is holding the SCL line to 0, then the master goes into a HIGH wait state until the SCL clock line transitions to 1.

All masters then count off their high time. The master with the shortest high time transitions the SCL line to 0. The masters then counts out their low time. The count with the longest low time forces the other master into a HIGH wait state. Therefore, a synchronized SCL clock is generated. Optionally, slaves may hold the SCL line low to slow down the timing on the I^2C bus.

Figure 73: Multi-Master Clock Synchronization



15.4.7 Operation Modes

This section provides information on operation modes.

Note: The I²C should only be set to operate as an I²C Master, or I²C Slave, but not both simultaneously. This is achieved by ensuring that Bit[6] (SLAVE_DISABLE) and Bit[0] (MASTER_MODE) of the IIC_CON register are never set to 0 and 1, respectively.

15.4.7.1 Slave Mode Operation

Initial Configuration

To use the I²C as a slave, perform the following steps:

1. Disable the I²C by writing a 0 to Bit 0 of the I2C _ENABLE register.
2. Write to the I2C _SAR register (bits 9:0) to set the slave address. This is the address to which the I²C responds.
3. Write to the IIC_CON register to specify which type of addressing is supported (7-bit or 10-bit by setting Bit 3). Enable the I²C in slave-only mode by writing a 0 into bit 6 (SLAVE_DISABLE) and a 0 to Bit 0 (MASTER_MODE).

   Note: Slaves and masters do not have to be programmed with the same type of addressing 7- or 10-bit address. For instance, a slave can be programmed with 7-bit addressing and a master with 10-bit addressing, and vice versa.
4. Enable the TWSI by writing a 1 to Bit 0 of the IIC_ENABLE register.

Slave-Transmitter Operation for a Single Byte

When another I²C master device on the bus addresses the I²C and requests data, the I²C acts as a slave-transmitter and the following steps occur:

1. The other I²C master device initiates a I²C transfer with an address that matches the slave address in the I2C_SAR register of the I²C.
2. The I²C acknowledges the sent address and recognizes the direction of the transfer to indicate that it is acting as a slave-transmitter.
3. The I²C asserts the RD_REQ interrupt (Bit 5 of the I2C_RAW_INTR_STAT register) and holds the SCL line low. It is in a wait state until software responds. If the RD_REQ interrupt has been masked, due to I2C_INTR_MASK [5] register (M_RD_REQ bit field) being set to 0, then Marvell recommends that a hardware and/or software timing routine be used to instruct the CPU to perform periodic reads of the I2C_RAW_INTR_STAT register.
   a) Reads that indicate I2C_RAW_INTR_STAT [5] (R_RD_REQ bit field) being set to 1 must be treated as the equivalent of the RD_REQ interrupt being asserted.
   b) Software must then act to satisfy the I²C transfer.
   c) The timing interval used should be in the order of 10 times the fastest SCL clock period the I²C can handle. For example, for 400 Kbps, the timing interval is 25 μs.

   Note: The value of 10 is recommended here because this is approximately the amount of time required for a single byte of data transferred on the I²C bus.
4. If there is any data remaining in the TX FIFO before receiving the read request, then the I²C asserts a TX_ABRT interrupt (Bit 6 of the I2C_RAW_INTR_STAT register) to flush the old data from the TX FIFO.

   Note: Because the I²C TX FIFO is forced into a flushed/reset state whenever a TX_ABRT event occurs, software must release the I²C from this state by reading the I2C_CLR_TX_ABRT register before attempting to write into the TX FIFO. See register I2C_RAW_INTR_STAT for more details.

If the TX_ABRT interrupt has been masked, due to of I2C_INTR_MASK [6] register (M_TX_ABRT bit field) being set to 0, it is recommended that re-using the timing routine (described in the previous step), or a similar one be used to read the I2C_RAW_INTR_STAT register.

a) Reads that indicate Bit 6 (R_TX_ABRT) being set to 1 must be treated as the equivalent of the TX_ABRT interrupt being asserted.
b) There is no further action required from software.
c) The timing interval used should be similar to that described in the previous step for the I2C_RAW_INTR_STAT [5] register.

5. Software writes to the IIC_DATA_CMD register with the data to be written (by writing a 0 in Bit 8).
6. Software must clear the RD_REQ and TX_ABRT interrupts (bits 5 and 6, respectively) of the I2C_RAW_INTR_STAT register before proceeding. If the RD_REQ and/or TX_ABRT interrupts have been masked, then clearing of the I2C_RAW_INTR_STAT register will have already been performed when either the R_RD_REQ or R_TX_ABRT bit has been read as 1.
7. The I^2C releases the SCL and transmits the byte.
8. The master may hold the I^2C bus by issuing a RESTART condition or release the bus by issuing a STOP condition.

Slave-Receiver Operation for a Single Byte

When another I^2C master device on the bus addresses the I^2C and is sending data, the I^2C acts as a slave-receiver and the following steps occur:

1. The other I^2C master device initiates a I^2C transfer with an address that matches the I^2C slave address in the I2C_SAR register.
2. The I^2C acknowledges the sent address and recognizes the direction of the transfer to indicate that the I^2C is acting as a slave-receiver.
3. I^2C receives the transmitted byte and places it in the receive buffer.

   Note: If the RX FIFO is completely filled with data when a byte is pushed, then an overflow occurs and the I2C continues with subsequent I2C transfers. Because a NACK is not generated, software must recognize the overflow when indicated by the I2C (by the R_RX_OVER bit in the IIC_INTR_STAT register) and take appropriate actions to recover from lost data. Therefore, there is a real-time constraint on software to service the RX FIFO before the latter overflow as there is no way to reapply pressure to the remote transmitting master. Users must select a deep enough RX FIFO depth to satisfy the interrupt service interval of their system.

4. I^2C asserts the RX_FULL interrupt (I2C_RAW_INTR_STAT [2] register). If the RX_FULL interrupt has been masked, due to setting I2C_INTR_MASK [2] register to 0 or setting I2C_TX_TL to a value larger than 0, then Marvell recommends that a timing routine (see Slave-Transmitter Operation for a Single Byte, on page 240) be implemented for periodic reads of the I2C_STATUS register. Reads of the I2C_STATUS register, with Bit 3 (RFNE) set at 1, must then be treated by software as the equivalent of the RX_FULL interrupt being asserted.
5. Software may read the byte from the IIC_DATA_CMD register (bits 7:0).
6. The other master device may hold the I^2C bus by issuing a RESTART condition or release the bus by issuing a STOP condition.

Slave-Transfer Operation for Bulk Transfers

In the standard I^2C protocol, all transactions are single byte transactions and the programmer responds to a remote master read request by writing 1 byte into the slave TX FIFO. When a slave (slave-transmitter) is issued with a read request (RD_REQ) from the remote master (master-receiver), at a minimum there should be at least 1 entry placed into the slave-transmitter TX FIFO. I^2C is designed to handle more data in the TX FIFO so that subsequent read requests can take that data without raising an interrupt to get more data. Ultimately, this eliminates the possibility of significant latencies being incurred between raising the interrupt for data each time had there been a restriction of having only 1 entry placed in the TX FIFO.

This mode only occurs when I^2C is acting as a slave-transmitter. If the remote master acknowledges the data sent by the slave-transmitter and there is no data in the slave TX FIFO, the I^2C holds the I^2C SCL line low while it raises the read request interrupt (RD_REQ) and waits for data to be written into the TX FIFO before it can be sent to the remote master.

If the RD_REQ interrupt is masked, due to Bit 5 (M_RD_REQ) of the I2C_INTR_STAT register being set to 0, then it is recommended that a timing routine be used to activate periodic reads of the I2C_RAW_INTR_STAT register. Reads of I2C_RAW_INTR_STAT that return Bit 5 (R_RD_REQ) set to 1 must be treated as the equivalent of the RD_REQ interrupt referred to in this section. This timing routine is similar to that described in Slave-Transmitter Operation for a Single Byte, on page 240.

The RD_REQ interrupt is raised upon a read request, and like interrupts, must be cleared when exiting the interrupt service handling routine (ISR). The ISR allows users to either write 1 byte or more than 1 byte into the TX FIFO. During the transmission of these bytes to the master, if the master acknowledges the last byte, then the slave must raise the RD_REQ again because the master is requesting for more data.

If the programmer knows in advance that the remote master is requesting a packet of *n* bytes, then when another master addresses I^2C and requests data, the TX FIFO could be written with *n* number bytes and the remote master receives it as a continuous stream of data. For example, the I^2C slave continues to send data to the remote master as long as the remote master is acknowledging the data sent and there is data available in the TX FIFO. There is no need to hold the SCL line low or to issue RD_REQ again.

If the remote master is to receive *n* bytes from the I^2C but the programmer wrote a number of bytes larger than *n* to the TX FIFO, then when the slave finishes sending the requested *n* bytes, it clears the TX FIFO and ignores any excess bytes.

The I^2C generates a transmit abort (TX_ABRT) event to indicate the clearing of the TX FIFO in this example. At the time an ACK/NACK is expected, if a NACK is received, then the remote master has all the data it wants. At this time, a flag is raised within the slave state machine to clear the leftover data in the TX FIFO. This flag is transferred to the processor bus clock domain where the FIFO exists and the contents of the TX FIFO are cleared at that time.

15.4.7.2 Master Mode Operation

Initial Configuration

Perform the following steps to use the I^2C as a master:

1. Disable the I^2C by writing 0 to the I2C_ENABLE register.
2. Write to the I2C_CON register to set the maximum speed mode supported for slave operation (bits 2:1) and to specify whether the I^2C starts its transfers in 7/10 bit addressing mode when the device is a slave (Bit 3).
3. Write to the I2C_TAR register the address of the I^2C device to be addressed. It also indicates whether a General Call or a START BYTE command is going to be performed by I^2C. The required speed of the I^2C master-initiated transfers, either 7-bit or 10-bit addressing, is controlled by the BIT Offset address10_MASTER bit field (bit 12).
4. Enable the I^2C by writing a 1 in the I2C_ENABLE register.
5. Now write the transfer direction and data to be sent to the I2C_DATA_CMD register. If the I2C_DATA_CMD register is written before the I^2C is enabled, the data and commands are lost as the buffers are kept cleared when I^2C is not enabled.

   Note: For multiple I^2C transfers, perform additional writes to the TX FIFO such that the TX FIFO does not become empty during the I^2C transaction. If the TX FIFO is completely emptied at any stage, then further writes to the TX FIFO result in an independent I^2C transaction.

Dynamic TAR or BIT Offset address10_MASTER Update

The I^2C supports dynamic updating of the TAR (bits 9:0) and BIT Offset address10_MASTER (Bit 12) bit fields of the IIC_TAR register. Users can dynamically write to the I2C_TAR register provided all of the following conditions are met:

- I^2C is not enabled (I2C_ENABLE [0] =0); OR I^2C is enabled (I2C_ENABLE [0] =1)
- I^2C is NOT engaged in any Master (tx, rx) operation (I2C_STATUS [5] =0)
- I^2C is enabled to operate in Master Mode (I2C_CON [0] =1)
- No entries in the TX FIFO (I2C_STATUS [2] =1)

Master Transmit and Master Receive

The TWSI supports switching back and forth between reading and writing dynamically. To transmit data, write the data to be written to the lower byte of the I^2C Rx/Tx Data Buffer and Command Register (I2C_DATA_CMD). The *CMD* bit [8] should be written to 0 for I^2C write operations. Subsequently, a read command may be issued by writing “don’t cares” to the lower byte of the I2C_DATA_CMD register, and a 1 should be written to the *CMD* bit. The I^2C master continues to initiate transfers as long as there are commands present in the transmit FIFO. If the transmit FIFO becomes empty, the I^2C inserts a STOP condition after completing the current transfers.

15.4.8 I2C.CLK Frequency Configuration

When the I2C is configured as a master, the *CNT registers must be set before any I2C bus transaction can occur to ensure proper I/O timing.

The *CNT registers are:

- I2C.SS_SCL_HCNT
- I2C.SS_SCL_LCNT
- I2C.FS_SCL_HCNT
- I2C.FS_SCL_LCNT

15.4.8.1 Calculating High and Low Counts

This section shows how to calculate SCL high and low counts for each speed mode in the I2C. In this TWSI module ic_clk is 16 MHz. The equation to calculate the proper number of ic_clk signals required for setting the proper SCL clocks high and low times is as follows:

```
IIC_xCNT = ( ROUNDUP ( MIN_SCL_xxxtime * OSCFREQ, 0 ) )
            ROUNDUP is an explicit Excel function call that is used to convert a
real number to its equivalent integer number.
MIN_SCL_HIGHtime =  Minimum High Period
MIN_SCL_HIGHtime =  4000 ns for 100 Kbps
           600 ns for 400 Kbps
           60 ns for 3.4 Mbs, bus loading = 100 pF
           160 ns for 3.4 Mbs, bus loading = 400 pF
MIN_SCL_LOWtime =  Minimum Low Period
MIN_SCL_LOWtime =  4700 ns for 100 Kbps
           1300 ns for 400 Kbps
           120 ns for 2.4 Mbps, bus loading = 100 pF
           320 ns for 2.4 Mbps, bus loading = 400 pF
           OSCFREQ = ic_clk Clock Frequency (Hz)
For example:
OSCFREQ = 100 MHz
I2Cmode = fast, 40 Kbps
MIN_SCL_HIGHtime = 600 ns.
MIN_SCL_LOWtime = 1300 ns.
IIC_xCNT = (ROUNDUP(MIN_SCL_HIGH_LOWtime*OSCFREQ,0))
IIC_HCNT = (ROUNDUP(600 ns * 100 MHz,0))
IIC_HCNTSCL PERIOD = 60
IIC_LCNT = (ROUNDUP(1300 ns * 100 MHz,0))
    IIC_LCNTSCL PERIOD = 130
    Actual MIN_SCL_HIGHtime = 60*(1/100 MHz) = 600 ns
    Actual MIN_SCL_LOWtime = 130*(1/100 MHz) = 1300 ns
```

15.4.9 DMA Controller Interface

The I^2C has a built-in DMA capability to request and control transfers. To enable the DMA Controller interface on the I^2C, write the DMA Control Register (DMAC.I2C_DMA_CR). Writing a 1 into the TDMAE bit field of DMAC.I2C_DMA_CR register enables the I^2C transmit handshaking interface. Writing a 1 into the RDMAE bit field of the DMAC.I2C_DMA_CR register enables the I^2C receive handshaking interface.

See Section 8, Direct Memory Access (DMA) Controller, on page 169 for more information.

15.5 Register Description

See Appendix A.7.2, I2C Registers, on page 545 for a detailed description of the registers.

THIS PAGE INTENTIONALLY LEFT BLANK

16 Synchronous Serial Protocol (SSP)

16.1 Overview

An SSP port is a synchronous serial controller that can be connected to a variety of external Analog-to-Digital converters (ADC), audio and telecommunication codecs, and many other devices that use serial protocols for data transfer.

The 88MW300/302 supports 3 SSP interfaces: SSP0, SSP1, and SSP2. The SSP ports are configurable to operate in Master mode (the attached peripheral functions as a slave) or Slave mode (the attached peripheral functions as a master).

The SSP ports support serial bit rates up to 25 Mbps. Serial data sample size can be set to 8, 16, 18, or 32 bits in length. A FIFO is provided for Transmit data, and a second independent FIFO is provided for Receive data. Both FIFOs are 16 x 32 bits wide or both are 32 x 16 bits wide. The FIFOs can be loaded or emptied by the Cortex-M4F or by DMA burst transfers.

16.2 Features

- Directly supports Texas Instruments* (TI) Synchronous Serial Protocol (SSP), and Motorola* Serial Peripheral Interface (SPI)
- I^2S protocol is supported by programming the PSP
  - I^2S Philips standard
  - Most Significant bit (MSb)-justified standard (left justified)
  - Master or Slave mode operation
  - Data transfer up to 25 Mbps
  - Programmable data frame size: 8, 16, 18, or 32 bits
  - Separate FIFO for transmit and receive with 16 x 32 or 32 x 16 bit length
  - Receive-without-Transmit operation
  - Network mode with as many as 8 time slots for Programmable Serial Protocol (PSP) formats
- Independent transmit/receive in any, all, or none of the time slots
- Supports DMA transfer

16.3 Interface Signal Description

Table 140 shows the interface signals.

Table 140: SSP Interface Signals[1]

Signal Name	Type	Description
SSPx_RXD	I	Synchronous Serial Protocol Receive Data Serial data in. Sample length is selected by the <Extended Data Size Select> and <Data Size Select> fields in the SSP Control Register 0,
SSPx_TXD	O	Synchronous Serial Protocol Transmit Data Serial data out. Sample length is selected by the <Extended Data Size Select> and <Data Size Select> fields in the SSP Control Register 0,
SSPx_CLK	I/O	Synchronous Serial Protocol Serial Clock Controls the timing of a serial transfer. SSPx_CLK can be generated internally (master mode) or taken from an external source (slave mode)
SSPx_FRM	I/O	Synchronous Serial Protocol Serial Frame Indicator Indicates the beginning and the end of a serialized data sample. The SSPx_FRM can be generated internally (master mode) or taken from an external source (slave mode).

1. See Section 22.11.1, SSP Timing and Specifications, on page 340 for electrical specifications.

16.4 Functional Description

Serial data is transferred between the Cortex-M4F Processor and an external peripheral through FIFOs in the SSPx port. Data transfers between an SSPx port and memory are initiated by the core or by DMA bursts. Separate data paths for transmitting and receiving permit simultaneous transaction in both directions, depending on the protocols chosen. The core and DMA bursts transaction can transfer data between:

- Memory to the FIFO Data Register for the TXFIFO
- FIFO Data Register to memory for the RXFIFO

Data written to the FIFO Data Register by either the core or DMA is automatically transferred to the Transmit FIFO. When reading the FIFO Data Register by either the core or DMA, the data in the Receive FIFO is transferred automatically from the FIFO data register.

16.4.1 FIFO Operation

There are 2 separate and independent FIFOs available for transmitting (TXFIFO to peripheral) and receiving (RXFIFO from peripheral) serial data. The FIFOs are filled or emptied by the Cortex-M4F core or DMA bursts. The data is accessed through the TXFIFO and RXFIFO. The Cortex-M4F accesses are normally triggered by an interrupt caused by an SSP Status Register event (see the Appendix in this manual) and must always be 32 bits wide. The Cortex-M4F writes to the TXFIFO are 32-bits wide, but bits beyond the programmed FIFO data size are ignored. The Cortex-M4F Reads from the RXFIFO are also 32 bits wide with 0's inserted in the MSb's, down to the programmed data size.

The TXFIFO and RXFIFO can also be accessed by DMA bursts, which must be 8, 16, or 32 bytes in length, and must transfer 1 FIFO entry per access. When the SSP_SSCR0[EDSS] bit is set, the SSPx port must be configured as a 32-bit peripheral. The DMA burst transaction width must be programmed to at least the same data size programmed into this SSP_SSCR0[EDSS] and SSP_SSCR0[DSS] fields.

The TXFIFO and RXFIFO are each accessed as 1, 32-bit location by the Cortex-M4F core. For data transmission, the SSPx port transmits the data from the TXFIFO to the external peripheral through the SSPx_TXD interface. Data received from the external peripheral through the SSPx_RXD interface is converted to parallel words and written into the RXFIFO.

An interrupt or DMA service request is generated if a programmable FIFO trigger threshold exceeded which signals the Cortex-M4F or DMA to empty the RXFIFO or refill the TXFIFO.

The TXFIFO and RXFIFO are differentiated by whether the access is a Read or a Write transfer. Reads from the Data Register automatically target the RXFIFO. Writes to the FIFO Data Register automatically target the TXFIFO. From a memory-map perspective, the TXFIFO and the RXFIFO are at the same address. Each FIFO is 16 rows deep x 32 bits wide for a total of 16 data samples. Each sample can be 8, 16, 18, or 32 bits in length.

16.4.1.1 Parallel Data Formats for FIFO Storage

Data in the FIFOs is either stored with 1, 32-bit value per data sample (in non-packed or data size greater than 16 bits) or in a 16-bit value in packed mode when the data is 8 or 16 bits. Within each 32- or 16-bit field, the stored data sample is right-justified, with the LSb of the word in Bit 0. In the Receive FIFO, unused bits are packed as 0's above the MSb. In the Transmit FIFO, unused "don't-care" bits are above the MSb. For example, DMA accesses do not have to write to the unused bit locations. Logic in the SSP automatically formats data in the Transmit FIFO so that the sample is properly transmitted on SSPx_TXD in the selected frame format.

16.4.1.2 FIFO Operation in Packed Mode

When the TXFIFO and RXFIFO are operating in packed mode, each FIFO is 32 rows deep x 16-bits wide for a total of 32 data samples. For packed mode, each sample can be 8 or 16 bits in length. When the data is serialized and transmitted, Bits 15 to 0 are transmitted first, followed by Bits 31 to 16. When the TXFIFO and RXFIFO are operating in packed mode, they may best be thought of as a single entry of 32 bits holding 2, 8- or 16-bit samples. Thus, the Cortex-M4F Processor or the DMA should write and read 32 bits of data at a time where each Write or Read transfers 2 samples. The entire FIFO width (32 bits) must be read/written in this mode. The SSPx port does not support writing 2 separate 16-bit samples in this mode. The SSP FIFO thresholds align in 32-bit data size.

16.4.2 Using Programmed I/O Data Transfers

FIFO filling and emptying can be performed by the Cortex-M4F Processor in response to an interrupt from the FIFO logic. Each FIFO has a programmable FIFO trigger threshold that triggers an interrupt. When the number of entries in the RXFIFO exceeds the RXFIFO Trigger Threshold (SSP_SSCR1[RFT]) field in the SSP Control Register 1, an interrupt is generated (if enabled) that signals Cortex-M4F Processor to empty the RXFIFO. When the number of entries in the TXFIFO is less than or equal to the TXFIFO Trigger Threshold (SSP_SSCR1[TFT]) field in the SSP Control Register 1 plus 1, an interrupt is generated (if enabled) that signals the Cortex-M4F Processor to refill the TXFIFO.

The SSP Status Register can be polled to determine how many samples are in a FIFO and whether the FIFO is full or empty. Software is responsible for ensuring that the proper RXFIFO Trigger Threshold and TXFIFO Trigger Threshold values are chosen to prevent Receive FIFO Overrun and Transmit FIFO Underrun (in the SSP Status Register) error conditions.

16.4.3 Using DMA Data Transfers

The DMA controller can also be programmed to transfer data to and from the FIFOs. The SSP Serial Port uses the following handshaking signals to interface with the DMA controller:

- dma_tx_req
- dma_tx_single
- dma_tx_ack
- dma_rx_req
- dma_rx_single
- dma_rx_ack

As a block flow control device, the DMA Controller is programmed by the processor with the number of data items (block size) to be transmitted or received by the SSP. The block will be broken into a number of transactions, each initiated by a request from the SSP. The DMA Controller must also be programmed with the number of data items to be transferred for each DMA request, which is named burst size. When the block size is a multiple of the burst size, the DMA block transfer consists of a series of burst transactions. When the block size programmed into the DMA Controller is not a multiple of the burst size, a series of burst transactions followed by single transactions are needed to complete the block transfer.

16.4.3.1 Transmit and Receive FIFO Trigger Threshold

During SSP serial transfers, transmit FIFO requests are generated whenever the number of entries in the transmit FIFO is less than or equal to the transmit FIFO trigger threshold, which can be configured in TXFIFO threshold (SSCR1[TFT]) field. The DMA responds by writing a burst of data to the transmit FIFO buffer. The configuration of DMA burst size and SSCR1[TFT] should be correctly set to prevent the underflow and overflow of Transmit FIFO.

Programming DMA burst size to a value greater than the transmit FIFO threshold that triggers the DMA request may cause overflow when there is not enough space in the SSP transmit FIFO to service the destination burst request. Therefore, the DMA burst size should not be greater than the trigger threshold. The optimal operation is setting the DMA burst size the same as the transmit FIFO trigger threshold (SSCR1[TFT] value +1).

When the DMA burst size is equal to the empty space in the Transmit FIFO, the difference of transmit FIFO threshold will also lead to different situation. If the trigger threshold is low, the number of burst transactions of the DMA is lower than the high trigger threshold situation, which means has a better bus utilization. However, the probability of the FIFO underflow is high because the DMA may not have had enough time to service the DMA request before the transmit FIFO becomes empty. Software should choose a trigger threshold to balance the number of transactions per block and the probability of an underflow.

During SSP serial transfers, Receive FIFO requests are generated whenever the number of entries in the Receive FIFO is above the Receive FIFO trigger threshold, which can be configured in RXFIFO threshold (SSCR1[RFT]) field. The DMA responds by reading a burst of data from the Receive FIFO buffer. The configuration of DMA burst size and SSCR1[RFT] should be correctly to prevent the underflow and overflow of Transmit FIFO.

Programming DMA burst size to a value greater than the Receive FIFO threshold that triggers the DMA request may cause underflow when there is not enough data in the SSP Receive FIFO to service the destination burst request. Therefore, the DMA burst size should not be greater than the trigger threshold. The optimal operation is setting the DMA burst size the same as the Receive FIFO trigger threshold (SSCR1[RFT] value +1). Similar to choosing the transmit FIFO threshold, software should also choose a Receive FIFO trigger threshold to balance the number of transactions per block and the probability of an Receive FIFO overflow.

When not using packed mode, the SSP stores 1 data sample per FIFO location where each FIFO has 16 locations. When using packed mode, the SSP stores 2 data samples per FIFO location where each FIFO has 16 locations.

16.4.3.2 Handshaking Interface Details

The request signals for source and destination (dma_tx_req and dma_rx_req) are activated when their corresponding FIFOs reach the trigger levels discussed prior. The DMA uses rising-edge detection of the dma_tx_req /dma_rx_req signal to identify a request on the channel from SSP. Upon reception of the dma_tx_ack/dma_rx_ack signal from the DMA to indicate the burst transaction is complete, the SSP de-asserts the burst request signals. Then, the dma_tx_ack/dma_rx_ack will be de-asserted by the DMA. When the SSP samples that dma_tx_ack/dma_rx_ack is de-asserted, it can re-assert the dma_tx_req/dma_rx_req if their corresponding FIFOs exceed their trigger levels (back-to-back burst transaction). Otherwise, the DMA request remains de-asserted.

Figure 74 shows a timing diagram of a burst transaction where pclk = hclk.

Figure 74: Burst Transaction, hclk = pclk

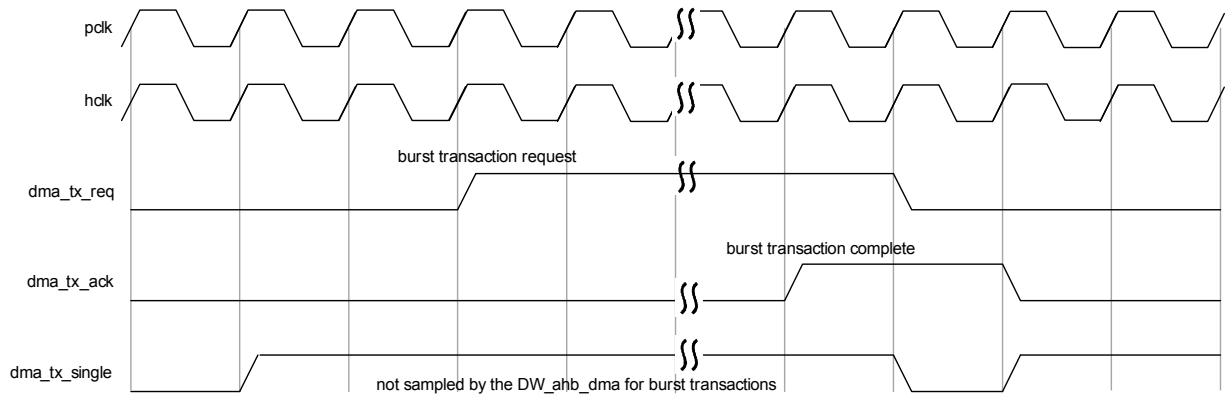
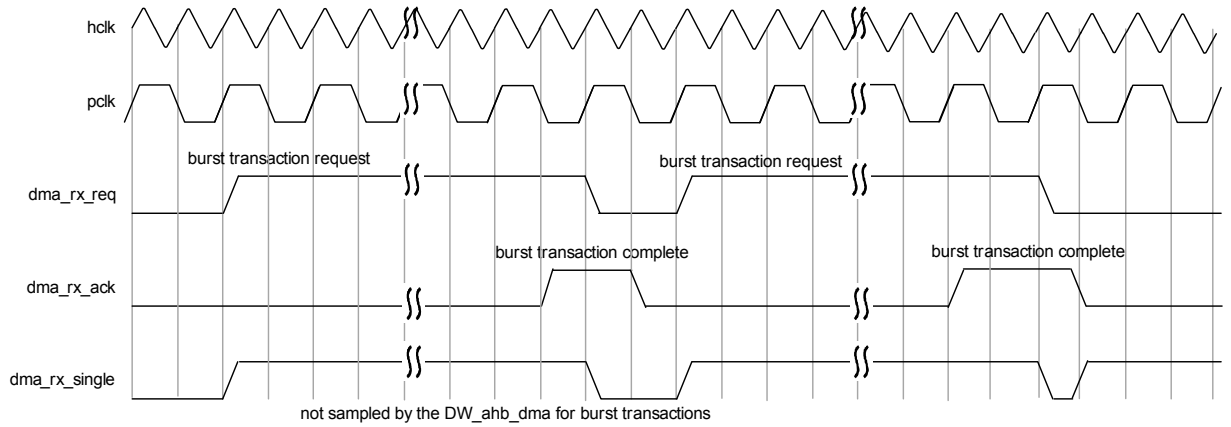


Figure 75 shows 2 back-to-back burst transactions where the hclk frequency is twice the pclk frequency.

Figure 75: Burst Transaction, hclk = 2*pclk



The handshaking loop is as follows:

- dma_tx_req/dma_rx_req asserted by SSP
- dma_tx_ack/dma_rx_ack asserted by DMAC
- dma_tx_req/dma_rx_req de-asserted by SSP
- dma_tx_ack/dma_rx_ack de-asserted by DMAC
- dma_tx_req/dma_rx_req re-asserted by SSP, if back-to-back transaction is required

The burst transaction request signals (dma_tx_req and dma_rx_req) are generated in the SSP off pclk and sampled in the DMA by hclk. The acknowledge signals (dma_tx_ack and dma_rx_ack) are generated in the DMA off hclk and sampled in the SSP of pclk. The handshaking mechanism between the DMA and the SSP supports quasi-synchronous clocks. That is, hclk and pclk must be phase-aligned, and the hclk frequency must be a multiple of the pclk frequency.

The dma_tx_single signal is asserted when there is at least 1 free entry in the transmit FIFO and is cleared when the transmit FIFO is full or the dma_tx_ack signal is active. The dma_tx_single signal will be re-asserted when the dma_tx_ack signal is removed if the condition for setting still holds true.

The dma_rx_single signal is asserted when there is at least 1 valid data entry in the receive FIFO and is cleared when the receive FIFO is empty or the dma_rx_ack signal is active. The dma_rx_ single signal will be re-asserted when the dma_rx_ack signal is removed if the condition for setting still holds true.

These signals are needed by the DMA only for the case that the block size programmed in the DMA is not a multiple of the burst size. For example, for a Block size of 15 and burst size of 4:

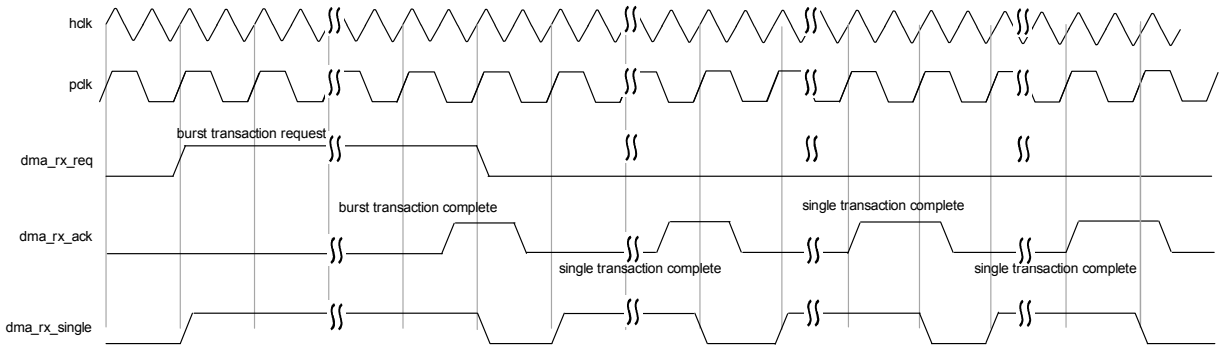
The first 12 data items are transferred as already described using 3 burst transactions. When the last 3 data frames enter the receive FIFO, the dma_rx_req signal is not activated because the FIFO level is below the trigger level 4. The DMA samples dma_rx_single and completes the DMA block transfer using 3 single transactions. The block transfer is made up of 3 burst transactions followed by 3 single transactions.

The handshaking loop is as follows:

- dma_tx_single/dma_rx_single asserted by SSP
- dma_tx_ack/dma_rx_ack asserted by DMAC
- dma_tx_single/dma_rx_single de-asserted by SSP
- dma_tx_ack/dma_rx_ack de-asserted by DMAC

Figure 76 shows a burst transaction, followed by 3 back-to-back single transactions, where the hclk frequency is twice the pclk frequency.

Figure 76: Burst Transaction, +3 Back-to-Back Singles **C hclk = 2*pclk



The single transaction request signals (dma_tx_single and dma_rx_single) are generated in the SSP on the pclk edge and sampled in DMA on hclk. The acknowledge signals (dma_tx_ack and dma_rx_ack) are generated in the DMA on the hclk edge hclk and sampled in the SSP on pclk. The handshaking mechanism between the DMA and the SSP supports quasi-synchronous clocks. That is, hclk and pclk must be phase-aligned, and the hclk frequency must be a multiple of pclk frequency.

16.4.4 SSP Interrupts

The SSP can generate 5 interrupts, each enabled individually by configuring their Interrupt Mask. See Table 141.

Table 141: SSP Interrupts

Interrupt	Description
Bit Count Error interrupt (SSSR[BCE])	In slave mode, if the real sample size mismatch to the configuration of SSCR0[EDSS] and SSCR0[DSS], and when the SSCR1[EBCEI] (Enable Bit Count Error interrupt) is set, this interrupt is generated. Only support SSP and PSP formats.
Transmit FIFO Underrun	This interrupt is generated when a read operation occurs to the empty TXFIFO when the SSCR0[TIM] (Transmit FIFO Underrun Interrupt Mask) is set to 0 (enabled).
Receive FIFO Overrun	This interrupt is generated when a write operation occurs to the full RXFIFO when the SSCR0[RIM] (Receive FIFO Overrun Interrupt Mask) is set to 0 (enabled).
Transmit FIFO Service Request	This interrupt is generated when the number of entries in the transmit FIFO reaches or is below the transmit FIFO threshold (configured in SSCR1[TFT]) and if the SSCR1[TIE] (Transmit FIFO Interrupt Enable) is set to 1 (enabled).
Receive FIFO Service Request	This interrupt is generated when the number of entries in the receive FIFO exceeds the receive FIFO threshold (configured in SSCR1[RFT]) and if the SSCR1[RIE] (Receive FIFO Interrupt Enable) is set to 1 (enabled).

The SSP also has the read-write Interrupt test registers for testing purposes, which can assert interrupts directly by writing the register bits. For details, see Appendix A.9.2, SSP Registers, on page 614.

16.4.5 Data Formats

This section describes the types of formats used to transfer serial data between the Cortex-M4F core and external peripherals.

16.4.5.1 Serial Data Formats for Transfer to/from Peripherals

There are 4 interface signals for each SSPx port transfer data between the Cortex-M4F core and external peripherals. Although serial-data formats exist, each has the same basic structure, and in all cases, the interface signals used are:

- SSPx_CLK – Bit rate at which serial data is driven onto and sampled from the port
- SSPx_FRM – Boundaries of a basic data “unit”, comprised of multiple serial bits
- SSPx_TXD – Serial datapath for transmitted data from the SSPx port to the peripheral
- SSPx_RXD – Serial datapath for received data from peripheral to the SSPx port

A data frame can contain 8, 16, 18, or 32 bits (see SSP_SSCR0[EDSS] and SSP_SSCR0[DSS] fields in the Appendix A.9, SSP Address Block, on page 613. Serial data is transmitted with the MSb first. The formats directly supported are the Motorola SPI and Texas Instruments SSP. The I^2S protocol is supported by programming the PSP format.

The SSPx_FRM function and use varies between each format:

- SPI format – SSPx_FRM functions as a chip select to enable the external device (target of the transfer) and is held active-low during the data transfer. During continuous transfers, the SSPx_FRM signal can be either held low or pulsed depending upon the value of the Motorola* SPI SSPx_CLK phase setting, SSP_SSCR1[SPH], field in the SSP Control Register 1. Master and Slave modes are supported. SPI is a full-duplex format.
- SSP format – SSPx_FRM is pulsed high for 1 (serial) data period at the start of each frame. Master and Slave modes are supported. SSP is a full-duplex format.
- PSP format (I^2S) – SSPx_FRM is programmable in direction, delay, polarity, and width. Master and Slave modes are supported. PSP can be programmed to be either full- or half-duplex format.

The SSPx_CLK function and use varies between each format:

- SPI format – Programmers choose which edge of SSPx_CLK to use for switching Transmit data and for sampling Receive data. In addition, moving the phase of SSPx_CLK can be user-initiated, shifting its active state 1/2 cycle earlier or later at the start and end of a frame. Master and Slave modes are supported, and in both, the SSPx_CLK only toggles during active transfers (does not run continuously).
- SSP format – Data sources switch Transmit data on the rising edge of SSPx_CLK and sample Receive data on the falling edge. Master and Slave modes are supported. When driven by the SSPx port, the SSPx_CLK only toggles during active transfers (not continuously) unless the SSP_SSCR1[SCFR], SSP_SSCR1[ECRA], or SSP_SSCR1[ECRB] functions are used.

When the SSPx_CLK is driven by another device, it is allowed to be either continuous or only driven during transfers.

- PSP format (I^2S) – Programmers choose which edge of SSPx_CLK to use for switching Transmit data and for sampling Receive data. In addition, programmers can control the Idle state for SSPx_CLK and the number of active clocks that precede and follow the data transmission. Master and Slave modes are supported. When driven by the SSPx port, the SSPx_CLK toggles only during active transfers, not continuously, unless the SSP_SSCR1[SCFR], SSP_SSCR1[ECRA], or SSP_SSCR1[ECRB] functions are used. When the SSPx_CLK is driven by another device, it is allowed to be either continuous or driven only during transfers, but certain restrictions on PSP parameters apply (see Programmable Serial Protocol (PSP) Format).

Normally, if the serial clock (SSPx_CLK) is driven by the SSPx port, it toggles only while an active data transfer is underway. However, there are several conditions that may cause the clock to run continuously. If the Receive-without-Transmit mode is enabled by setting the Receive Without Transmit SSP_SSCR1[RWOT], field and the frame format is not Microwire then the SSPx_CLK toggles regardless of whether Transmit data exists within the Transmit FIFO. The SSPx_CLK also toggles continuously if the SSPx port is in Network mode, or if the SSP_SSCR1[ECRA] or SSP_SSCR1[ECRB] bits are enabled. At other times, SSPx_CLK is held in an inactive or idle state, as defined by the specified protocol under which it operates.

16.4.5.2 TI-SSP Format Details

When outgoing data in the SSP controller is ready to transmit, SSPx_FRM asserts for 1 clock period. On the following clock, data to be transmitted is driven on SSPx_TXD 1 bit at a time, with the MSb first. For Receive data, the peripheral similarly drives data on the SSPx_RXD pin. Word length can be 8, 16, 18, or 32 bits. All output transitions occur on the rising edge of SSPx_CLK while data sampling occurs on the falling edge. The SSPx_TXD signal either retains the value of the last bit sent (bit 0) or goes to a high impedance state at the end of the transfer. If the SSPx port is disabled or reset, SSPx_TXD is forced to 0 (unless the TXD Tri-State Enable, SSP_SSCR1[TTE], bit is set, in which case it goes into a high impedance state).

Figure 77, Texas Instruments Synchronous Serial Frame Protocol (Single Transfers), on page 255 shows the TI Synchronous Serial Protocol for a single transmitted frame.

Figure 78, Texas Instruments Synchronous Serial Frame Protocol (Multiple Transfers), on page 256 shows the TI Synchronous Serial Protocol when back-to-back frames are transmitted.

Once the Transmit FIFO contains data, SSPx_FRM is pulsed high for 1 SSPx_CLK period and the value to be transmitted is transferred from the Transmit FIFO to the Transmit Logic Serial Shift register. On the next rising edge of SSPx_CLK, the most significant bit of the 8 to 32-bit data frame is shifted to the SSPx_TXD pin. Likewise, the MSb of the received data is shifted onto the SSPx_RXD pin by the off-chip serial slave device. Both the SSP port and the off-chip serial slave device then latch each data bit into the serial shifter on the falling edge of each SSPx_CLK. The received data is transferred from the serial shifter to the Receive FIFO on the first rising edge of SSPx_CLK after the last bit has been latched.

For back-to-back transfers, the start of 1 frame immediately follows the completion of the previous. The MSb of 1 transfer immediately follows the LSb of the preceding with no “dead” time between them.

When the enhanced SSPx port is a master to the frame synch (SSPx_FRM) and a slave to the clock (SSPx_CLK), then at least 3 extra clocks (SSPx_CLK) are needed at the beginning and end of each block of transfers to synchronize control signals from the ARM peripheral bus (APB) clock domain into the SSP clock domain (a block of transfers is a group of back-to-back continuous transfers).

> ***Note:*** When configured as either master or slave to SSPx_CLK or SSPx_FRM, the SSP port continues to drive SSPx_TXD until the last bit of data is sent (the LSb) or the SSPx_TXD line becomes high impedance. If SSP_SSCR0[SSE] is cleared, the SSPx_TXD line goes low. SSP_SSPSP[EDTS] has no effect when in SSP mode. SSPx_RXD is undefined before the MSb is sent and after the LSb is sent. SSPx_RXD must not float.

Figure 77: Texas Instruments Synchronous Serial Frame Protocol (Single Transfers)

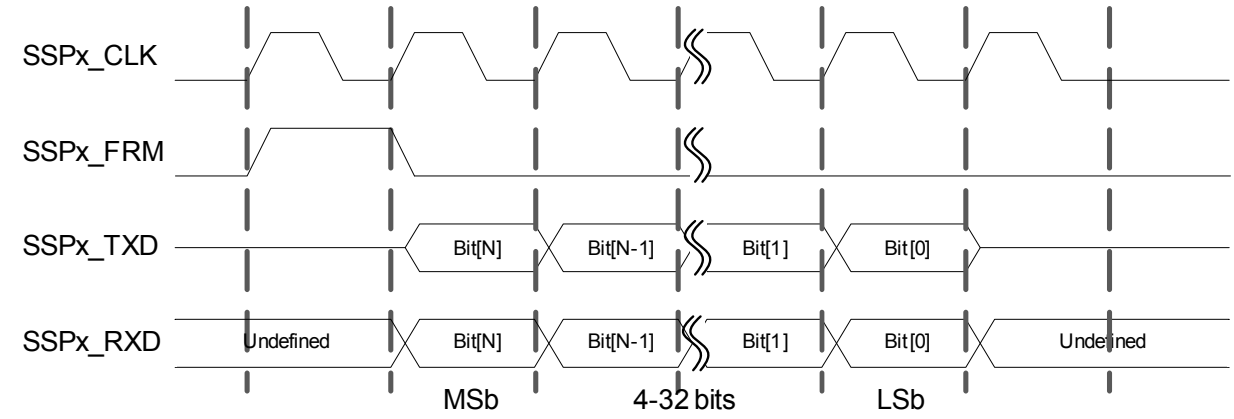
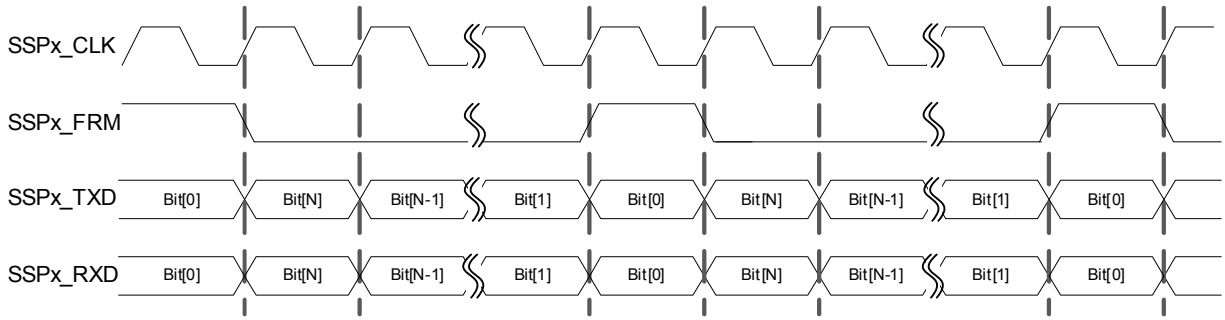


Figure 78: Texas Instruments Synchronous Serial Frame Protocol (Multiple Transfers)



16.4.5.3 Motorola SPI Format Details

The SPI format has 4 possible sub-modes depending on the SSPx_CLK edges selected for driving data and sampling received data and on the selection of the phase mode of SSPx_CLK (see Section 16.4.5.3.1, Serial Clock Phase (SPH), on page 257, for a complete description of each sub-mode).

When the SSP port is disabled or in idle mode, SSPx_CLK and SSPx_TXD are low and SSPx_FRM is high. When transmit data is ready to be sent, SSPx_FRM goes low (1 clock period before the first rising edge of SSPx_CLK) and stays low for the remainder of the frame. The most significant bit of the serial data is driven onto SSPx_TXD 1/2 cycle later. Halfway into the first bit period, SSPx_CLK asserts high and continues toggling for the remaining data bits. Data transitions on the falling edge of SSPx_CLK and is sampled on the rising edge of SSPx_CLK. 8, 16, 18, or 32 bits can be transferred per frame.

With the assertion of SSPx_FRM, Receive data is driven simultaneously from the peripheral on SSPx_RXD, MSb first. Data transitions on SSPx_CLK falling edges and is sampled by the controller on SSPx_CLK rising edges. At the end of the frame, SSPx_FRM is de-asserted high 1 clock period (1/2 clock cycle after the last falling edge of SSPx_CLK) after the last bit has been latched at its destination and the completed incoming word is shifted into the “incoming” FIFO. The peripheral can drive SSPx_RXD to a high-impedance state after sending the last bit of the frame. SSPx_TXD retains the last value transmitted when the controller goes into Idle mode, unless the enhanced SSPx port is disabled or reset (which forces SSPx_TXD to 0).

For back-to-back transfers, start and completion are like those of a single transfer, but SSPx_FRM does not de-assert between words. Both transmitter and receiver are configured for the word length and internally track the start and end of frames. There are no “dead” bits; the LSb of 1 frame is followed immediately by the MSb of the next.

When in Motorola SPI format, the enhanced SSPx port can be either a master or a slave device, but the clock and frame direction must be the same. For example, the SSP Serial Bit Rate Clock Direction, SSP_SSCR1[SCLKDIR], and the SSP Frame Direction, SSP_SSCR1[SFRMDIR], fields must either both be set or cleared.

When in Motorola SPI format, if the SSP port is the master and SSP_SSPSP[ETDS] is cleared, the end-of-transfer data state for SSPx_TXD is low. If the SSP port is the master and SSP_ SSPSP[ETDS] is set, the end-of-transfer data state for SSPx_TXD remains at the last bit transmitted (LSb). If the SSP port is the slave, then the SSP_SSPSP[ETDS] is undefined.

SSPx_RXD is undefined before the frame is active and after the LSb is received. SSPRXD must not float. When the SSP port is configured as a master and SSP_SSCR1[TTE] is set, SSP_ SSPSP[ETDS] is ignored and SSPx_TXD becomes high impedance between active transfers).

> ***Note:*** The input clock to the SSPx port must not be active when SSPx_FRM is de-asserted. When the SSP port is slave to clock and frame, SSP_SSCR1[SCFR] must be set.

16.4.5.3.1 Serial Clock Phase (SPH)

The phase relationship between SSPx_CLK and SSPx_FRM when the Motorola SPI protocol is selected is controlled by SSP_SSCR1[SPH].

The combination of the SSP_SSCR1[SPO] and SSP_SSCR1[SPH] settings determine when SSPx_CLK is active during the assertion of SSPx_FRM and which SSPx_CLK edge transmits and receives data on SSPx_TXD and SSPx_RXD.

When SPH is cleared, SSPx_CLK remains in its inactive (idle) state (as determined by SSP_SSCR1[SPO]) for 1 full cycle after SSPx_FRM is asserted low at the beginning of a frame. SSPx_CLK continues to toggle for the rest of the frame. It is then held in its inactive state for 1/2 of an SSPx_CLK period before SSPx_FRM is de-asserted high at the end of the frame. When SPH is set, SSPx_CLK remains in its inactive or idle state (as determined by SSP_SSCR1[SPO]) for 1/2 cycle after SSPx_FRM is asserted low at the beginning of a frame. SSPx_CLK continues to toggle for the remainder of the frame and is then held in its inactive state for 1 full SSPx_CLK period before SSPx_FRM is de-asserted high at the end of the frame. When programming SSP_SSCR1[SPO] and SSP_SSCR1[SPH] to the same value (both set or both cleared), transmit data is driven on the falling edge of SSPx_CLK and receive data is latched on the rising edge of SSPx_CLK. When programming SSP_SSCR1[SPO] and SSP_SSCR1[SPH] to opposite values (1 set and the other cleared), transmit data is driven on the rising edge of SSPx_CLK and receive data is latched on the falling edge of SSPx_CLK.

See Figure 79, Figure 80, Figure 81, and Figure 82.

Figure 79: Motorola SPI Frame Protocol (Single Transfers)

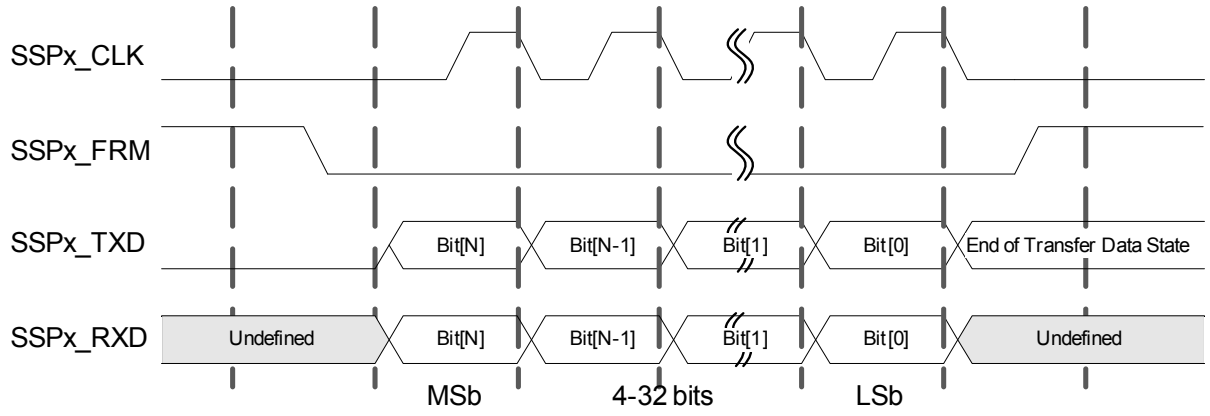


Figure 80: Motorola SPI Frame Protocol (Multiple Transfers)

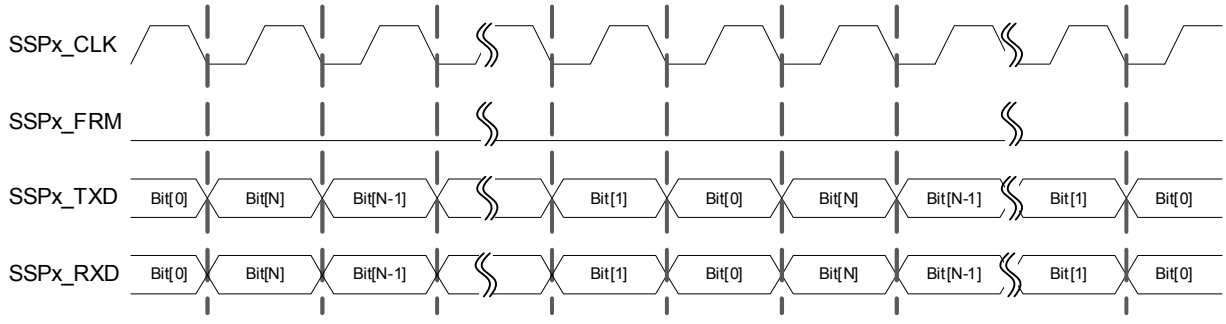
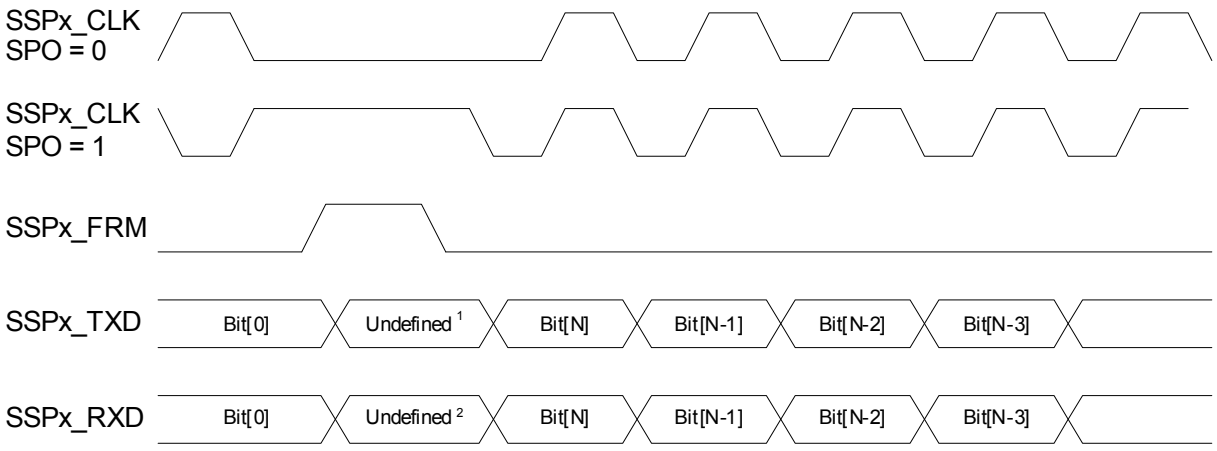
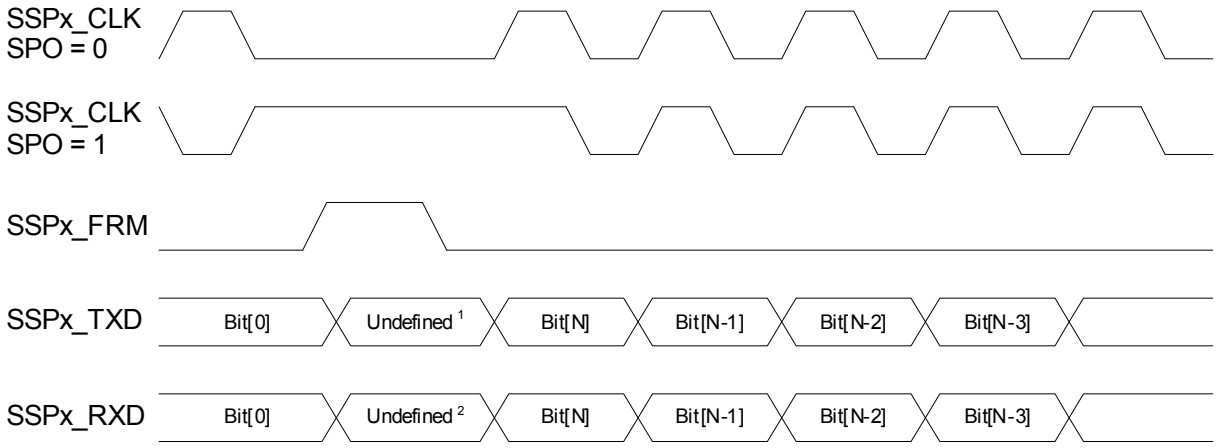


Figure 81: Motorola SPI Frame Protocols for SPO and SPH Programming (SPH Set)



Notes:
SSPx_TXD will be tri-stated at this point if TTE bit is set
SSPx_RXD should not float
SPH = 0

Figure 82: Motorola SPI Frame Protocols for SPO and SPH Programming (SPH Cleared)



Notes:
SSPx_TXD will be tri-stated at this point if TTE bit is set
SSPx_RXD should not float
SPH = 1

16.4.6 Programmable Serial Protocol (PSP) Format

The PSP format defines programmable parameters that determine the transfer timings between data samples. There are 4 serial clock modes defined in the Serial Bit-rate Clock Mode, SSP_SSPSP[SCMODE], field in the SSP Programmable Serial Protocol Register. These modes select the SSPx_CLK rising and falling edges for driving data, sampling received data, and the SSPx_CLK idle state.

Table 142, Programmable Protocol Parameters, on page 260 shows the Idle and Disable modes of the SSPx_TXD, SSPx_CLK, and SSPx_FRM interface signals are programmable using the following fields in the SSP Programmable Serial Protocol Register:

- End Of Transfer Data State (SSP_SSPSP[ETDS])
- Serial Frame Polarity (SSP_SSPSP[SFRMP])
- Serial Bit-rate Clock Mode (SSP_SSPSP[SCMODE])

When Transmit data is ready, SSPx_CLK remains in its Idle state for the number of serial clock (SSPx_CLK) periods programmed into the Start Delay (SSP_SSPSP[STRTDLY]) field in the SSP Programmable Serial Protocol Register.

SSPx_CLK then starts toggling. SSPx_TXD remains in the idle state for the number of serial clock periods programmed into the Dummy Start (SSP_SSPSP[DMYSTRT]) field in the SSP Programmable Serial Protocol Register. SSPx_FRM is asserted after the number of half serial clock periods programmed into the Serial Frame Delay (SSP_SSPSP[SFRMDLY]) field. SSPx_FRM remains asserted for the number of serial clock periods programmed into the Serial Frame Width (SSP_SSPSP[SFRMWDTH]) field in the SSP Programmable Serial Protocol Register, then SSPx_FRM de-asserts.

Serial data of 8, 16, 18, or 32 bits can be transferred per frame by setting the SSP_SSCR0[EDSS] and SSP_SSCR0[DSS] fields to the preferred data size select. Once the last bit (LSb) is transferred, SSPx_CLK continues toggling for the number of serial clock periods programmed into the Dummy Stop (SSP_SSPSP[DMYSTOP]) field. Depending on the value programmed into the End Of Transfer Data State (SSP_SSPSP[EDTS]) field when the SSPx port goes into Idle mode, SSPx_TXD either retains the last bit-value transmitted or is forced to 0 unless the SSPx port is disabled or reset, which forces SSPx_TXD to 0.

With the assertion of SSPx_FRM, Receive data is driven simultaneously from the peripheral onto SSPx_RXD, MSb first. Data transitions on the SSPx_CLK edge based on the serial-clock mode that is selected (SSP_SSPSP[SCMODE]) and is sampled by the SSPx port on the opposite clock edge. When the SSPx port is a master to SSPx_FRM and a slave to SSPx_CLK, at least 3 extra SSPSCLKs are needed at the beginning and end of each block of transfers to synchronize control signals from the APB clock domain into the SSP clock domain (a block of transfers is a group of back-to-back continuous transfers).

In general, because of the programmable nature of the PSP protocol, this protocol can be used to achieve a variety of serial protocols. For example: some DigRF protocol timing can be achieved by programming these values:

- Start Delay (SPP_SSPSP[STRTDLY]) = 0
- Dummy Start (SPP_SSPSP[DMYSTRT]) = 0
- Dummy Stop (SPP_SSPSP[DMYSTOP]) = 0
- Serial Frame Delay (SPP_SSPSP[SFRMDLY]) = 0

The SSPx port should be configured as a clock slave (SSP Serial Bit Rate Clock Direction (SSP_SSCR1[SCLKDIR]) = 1) and frame master (SSP Frame Direction (SSP_SSCR1[SFRMDIR]) = 0). Also, the Frame Sync Relative Timing Bit (SSP_SSPSP[FSRT]) field in the SSP Programmable Serial Protocol Register must be set for continuous transfers, and the Serial Frame Width (SSP_SSPSP[SFRMWDTH]) field should be equal to the data sample size.

Table 142: Programmable Protocol Parameters

Symbol	Definition	Range	Units
--	Serial clock mode (SSP_SSPSP[SCMODE])	(Drive, Sample, SSPx_CLK Idle) 0x0 = fall, rise, low 0x1 = rise, fall, low 0x2 = rise, fall, high 0x3 = fall, rise, high	--
--	Serial frame polarity (SSP_SSPSP[SFRMP])	High or low	--
T1	Start delay (SSP_SSPSP[STRTDLY])	0 to 7	clock period
T2	Dummy start (SSP_SSPSP[EDMYSTRT] + SSP_SSPSP[DMYSTRT])	0 to 15	clock period
T3	Data size (SSP_SSCR0[EDSS] and SSP_SSCR0[DSS])	4 to 32	clock period
T4	Dummy stop (SSP_SSPSP[EDMYSTOP] + SSP_SSPSP[DMYSTOP])	0 to 31	clock period
T5	SSPSFRM delay (SSP_SSPSP[SFRMDLY])	0 to 127	half-clock period
T6	SSPSFRM width (SSP_SSPSP[SFRMWDTH])	1 to 63	clock period
--	End of transfer data state (SSP_SSPSP[ETDS])	Low or Bit[0]	--

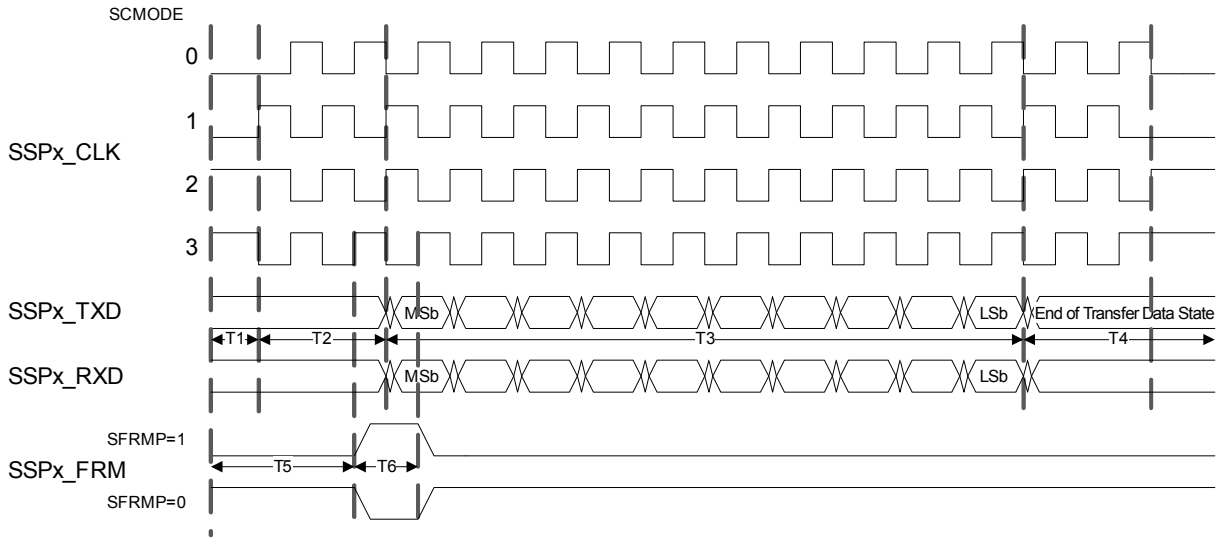
The SSPx_FRM delay (T5) must not extend beyond the end of T4. The SSPx_FRM width (T6) must be asserted for at least 1 SSPx_CLK period and should be de-asserted before the end of T4 (for example, in terms of time, not bit values, to ensure that SSPx_FRM is asserted for at least 2 edges of SSPx_CLK).

(T5 + T6) <= (T1 + T2 + T3 + T4), 1<= T6 < (T2 + T3 + T4), and (T5 + T6) >= (T1 + 1)

Program T1 to 0b0 when SSPx_CLK is enabled by any of the SSP_SSCR1[SCFR], SSP_SSCR1[ECRA], or SSP_SSCR1[ECRB] fields in the SSP Control Register 1. While the PSP can be programmed to generate the assertion of SSPx_FRM during the middle of the data transfer (for example, after the MSb has been sent), the SSPx port is unable to Receive data in frame-Slave mode (SSP_SSPSP[SFRMDIR] is set, if the assertion of the frame is not before the MSb is sent (for example, T5 <= T2 if the SSP_SSCR1[SFRMDIR] bit is set). Transmit data transitions from the end-of-transfer-data state (SSP_SSPSP[ETDS]) to the next MSb data value upon assertion of the internal version of SSPx_FRM. Program the SSP_SSPSP[STRTDLY] field to 0x00 whenever SSPx_CLK or SSPx_FRM is configured as an input (for example, SSP_SSCR1[SCLKDIR] and SSP_SSCR1[SFRMDIR] are cleared.

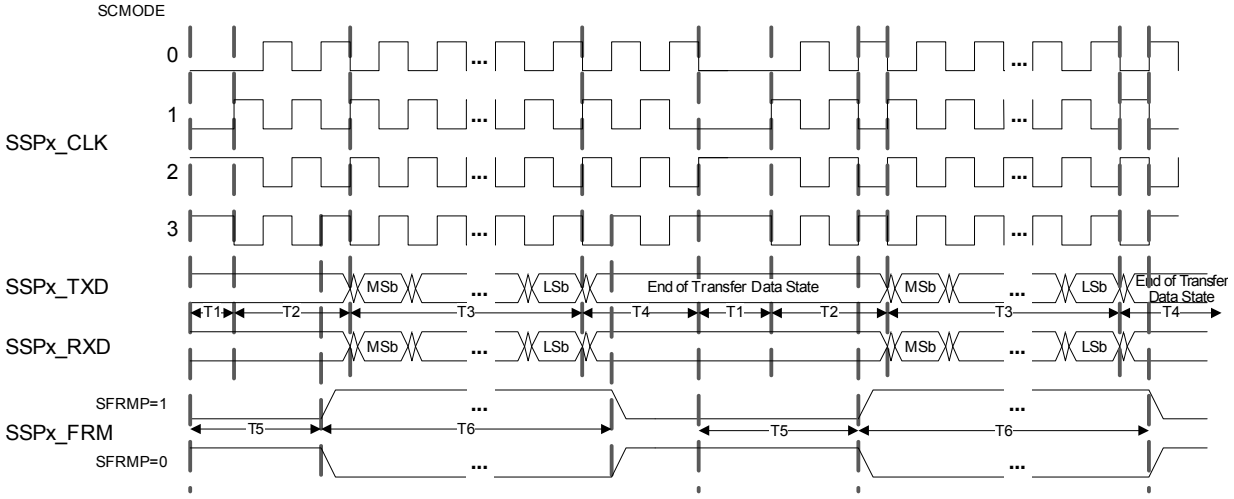
See Figure 83, Programmable Serial Protocol Format, on page 261 and Figure 84, Programmable Protocol Format (Consecutive Transfers), on page 261.

Figure 83: Programmable Serial Protocol Format



Note: If SSPx port is the master of SSPx_CLK (output) and SSPSP_x[ETDS=0], the End of Transfer data state (ETDS) for the SSPx_TXD line is 0. If the SSP is the master of the clock, and the SSPSP[ETDS] bit is set, then the SSPx_TXD line remains at the last bit transmitted (LSB).
If the SSPx port is a slave to SSPx_CLK (input), and modes 1 or 3 are used, then the ETDS can only change from the LSB if more SSPx_CLKs are sent to the SSPx port.

Figure 84: Programmable Protocol Format (Consecutive Transfers)



16.4.6.1 High Impedance on SSPx_TXD

The SSP supports placing the SSPx_TXD into high impedance during idle times instead of driving SSPx_TXD as controlled by the TXD Tri-State Enable (SSP_SSCR1[TTE]) and TXD Tri-State Enable On Last Phase (SSP_SSCR1[TTELP]) fields in the SSP Control Register 1. The SSP_SSCR1[TTE] enables a high-impedance state on SSPx_TXD. The SSP_SSCR1[TTELP] determines on which SSPx_CLK phase SSPx_TXD becomes high impedance.

See Figure 85, Figure 86, Figure 87, Figure 88, and Figure 89.

Figure 85: TI SSP with SSP_SSCR1[TTE] = 1 and SSP_SSCR1[TTELP]] = 0

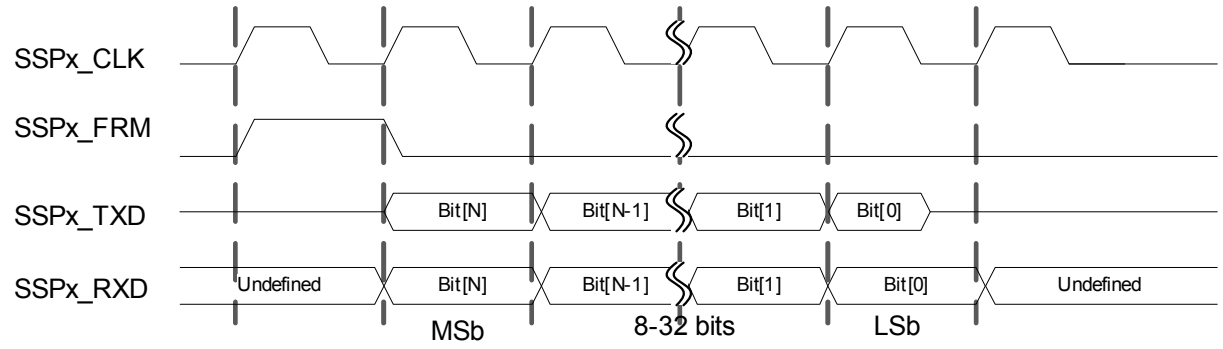


Figure 86: TI SSP with SSP_SSCR1[TTE] = 1 and SSP_SSCR1[TTELP] = 1

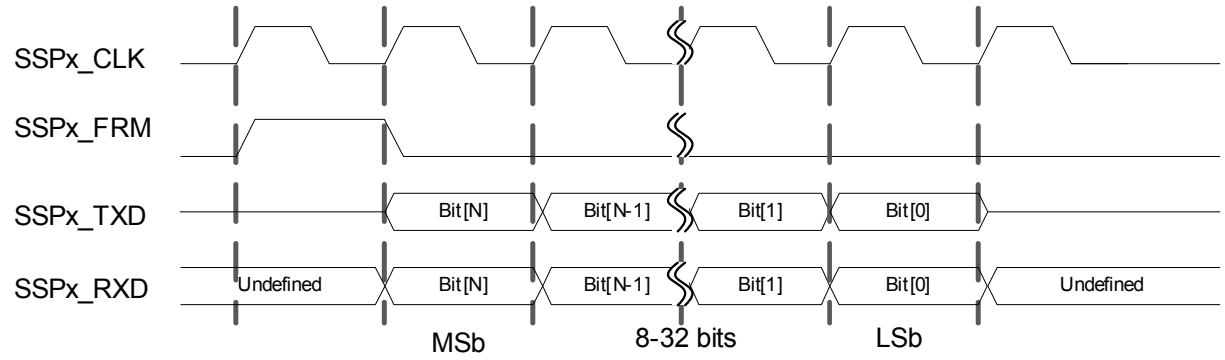


Figure 87: Motorola* SPI with <TXD Tri-State Enable> = 1 and <TXD Tri-State Enable On Last Phase> = 0

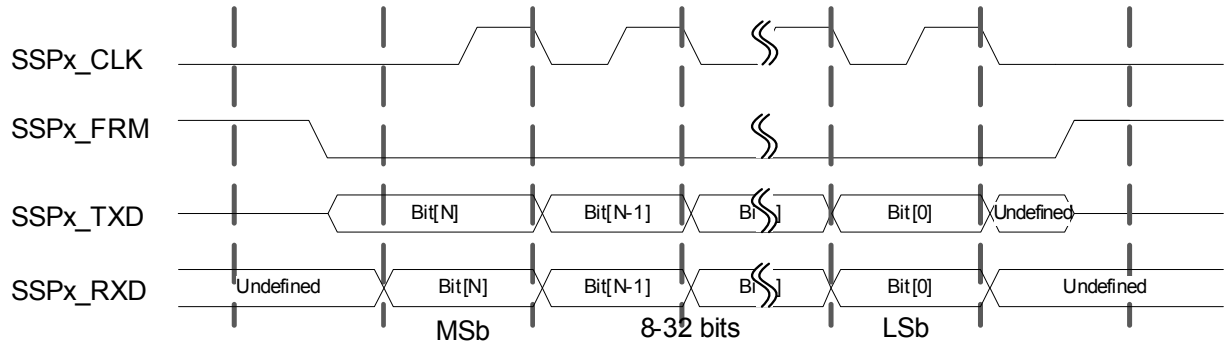


Figure 88: PSP Format with SSP_SSCR1[TTE] = 1, SSP_SSCR1[TTELP] = 0, and SSP_ SSCR1[SFRMDIR] = 1

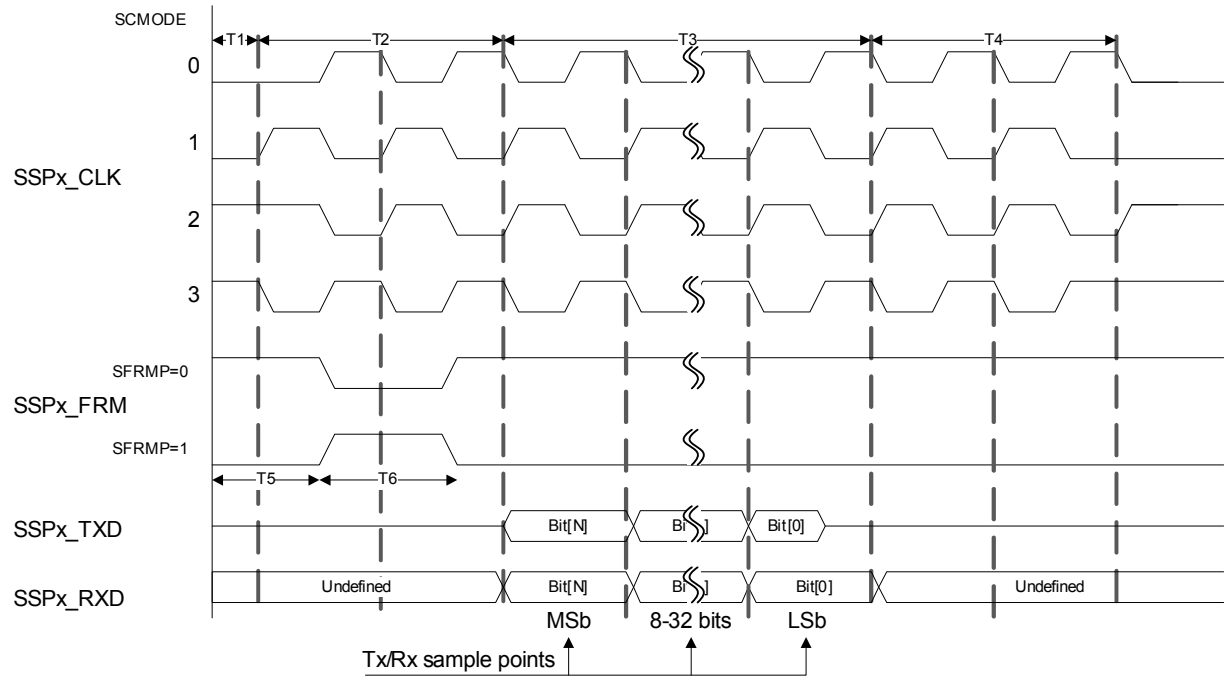
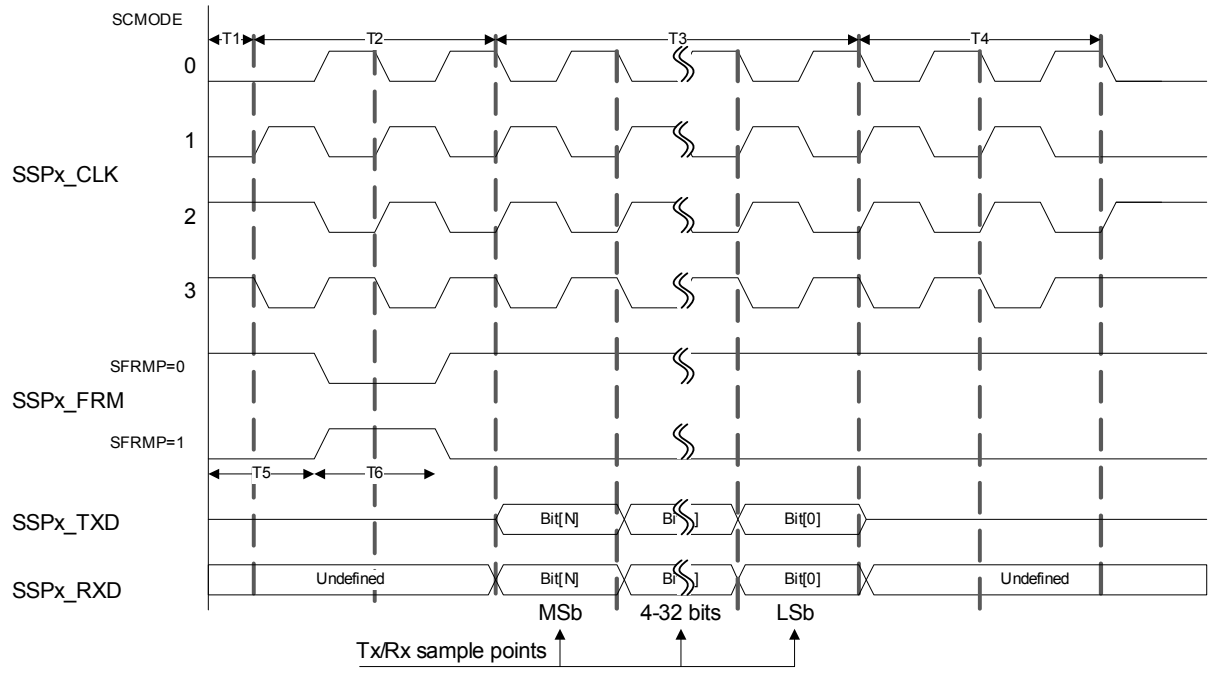


Figure 89: PSP Format with SSP_SSCR1[TTE] = 1, and either SSP_SSCR1[TTELP] = 1, or SSP_SSCR1[SFRMDIR] = 0



16.4.7 Network Mode

The SSP_SSCR0[MOD] bit selects between Normal and Network modes. Normal mode (MOD 0x0) is used when using the Texas Instruments* Synchronous Serial Protocol (SSP), and the Motorola* Serial Peripheral Interface (SPI). Network mode (MOD = 0x1) is used for emulating the I2S protocol. Software should set MOD only when using the PSP format. If the SSPx port is a master of the clock and SSP_SSCR1[SCLKDIR] is cleared, then setting MOD causes the SSPx_CLK to run continuously.

When in Network mode, only 1 SSPx_FRM is sent (master mode) or received (slave mode) for the number of time slots programmed into the SSP_SSCR0[FRDC] field. When beginning in Network mode, while the SSPx port is a master to the SSPx_FRM interface signal, the first SSPx_FRM signal does not occur until after data is in the TXFIFO. After assertion of the first SSPx_FRM signal, if the SSP is a master to SSPx_FRM, subsequent SSPx_FRM signals continue to assert regardless of whether data resides in the TXFIFO. Therefore, the transmit underrun bit, SSP_SSSR[TUR], is set to 0b1 if there is no data in the TXFIFO and the SSPx port is programmed to drive SSPx_TXD data in the current time slot, even if the SSPx port is master to SSPx_FRM. When using PSP format in Network mode, the parameters SFRMDLY, STRTDLY, DMYSTOP, DMYSTRT must all be 0b0. The other parameters SFRMP, SCMODE, FSRT, SFRMWDTH are programmable.

When the SSPx port is a master to the SSPx_FRM signal and a need arises to exit from Network mode, software should:

1. Clear the SSP_SSCR0[MOD] bit. SSP_SSCR0[SSE] does not need to change.
2. Wait until SSP_SSTSS[NMBSY] is cleared.
3. Disable the SSPx port by clearing SSP_SSCR0[SSE].
4. Before exiting Network mode, verify the TXFIFO is empty (SSP_SSSR[TFL]=0b0000 andSSP_SSSR[TNF]=0b1).

If data remains in the TXFIFO after the Network mode is exited, a non-Network mode frame will be sent.

Due to synchronization delay between the internal bus and the SSPx port clock domain, 1 extra frame may be transmitted after software clears the SSP_SSCR0[MOD] bit. The SSPx port continues to drive SSPx_CLK (if SSP_SSCR1[SCLKDIR] is cleared) and SSPx_FRM (if SSP_ SSCR1[SFRMDIR] is cleared) until the end of the last valid time slot.

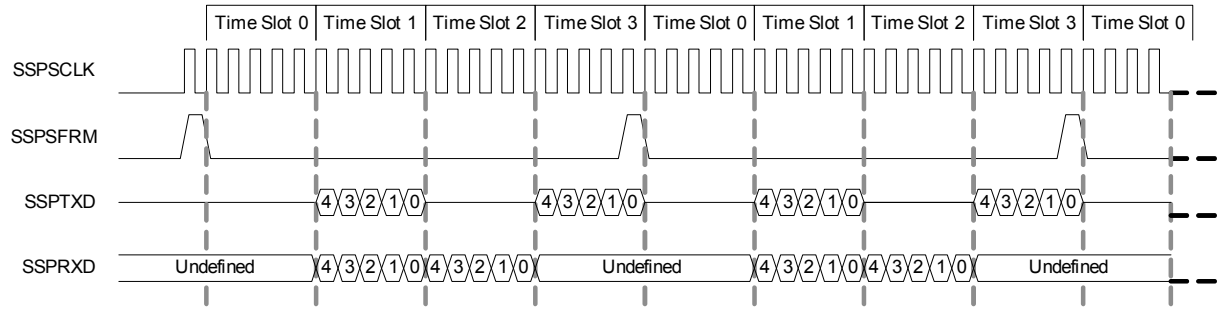
If the SSPx port is a slave to both SSPx_CLK (SSP_SSCR1[SCLKDIR] set) and SSPx_FRM (SSP_ SSCR1[SFMRDIR] set), the SSP_SSTSS[NMBSY] bit remains asserted until the SSP_ SSCR0[MOD] bit is cleared or until 1 SSPx_CLK after the end of the last valid time slot.

> ***Note:*** When operating in Slave mode (SSP_SSCR1[SCLKDIR] = SSP_SSCR1[SFRMDIR] =1) the external codec must provide the correct number of bits as determined by the frame width (SSP_SSPSP[SFRMWDTH]) and number of time slots (SSP_SSCR0[FRDC]). When the number of bits read in during a Frame cycle do not match the number expected, a Bit Count Error will occur (SSP_SSSR[BCE]) and the contents in the FIFO cannot be guaranteed. Software must be used to handle any error conditions when they occur.
>
> ***Note:*** In the next example, a data format of 5 is used. However, a data format of 5 is not a supported data size for the SSP controller. A data size of 5 was used only to reduce the size of the diagram.

16.4.7.1 Network Mode Registers

The register bits and fields that must be programmed for the Network Mode are SSP_SSCR0[MOD], SSP_SSCR0[FRDC], SSP_SSPSP[FSRT], SSP_SSPSP[SFRMWDTH], SSP_SSPSP[SCMODE], SSP_SSPSP[SFRMP], and SSP_SSPSP[SFRMDLY]. The definitions of each of these bits and fields is located in the registers document. See Figure 90.

Figure 90: Network Mode (Example Using 4 Time Slots)



Note: This example has 5 bits of data per sample. The TxD 3-state enable bit and the TxD3-state enable on last phase bit are both 0x1. The SSP is a master of SSPSCLK and SSPSFRM. The SSP has been programmed for 4 time slots (SSCR0_x[FRDC=0x11]), the Tx time Slot Active register is programmed to0x0000_000A (SSTSA[TTSA] = 00001010), and the Rx Time Slot Active register has been set to0x0000_0006 (SSRSA[RSTA] = 00000110).

Note: A data format of 5 is not a supported data size for the SSP controller A data size of 5 is used only to reduce the size of the diagram.

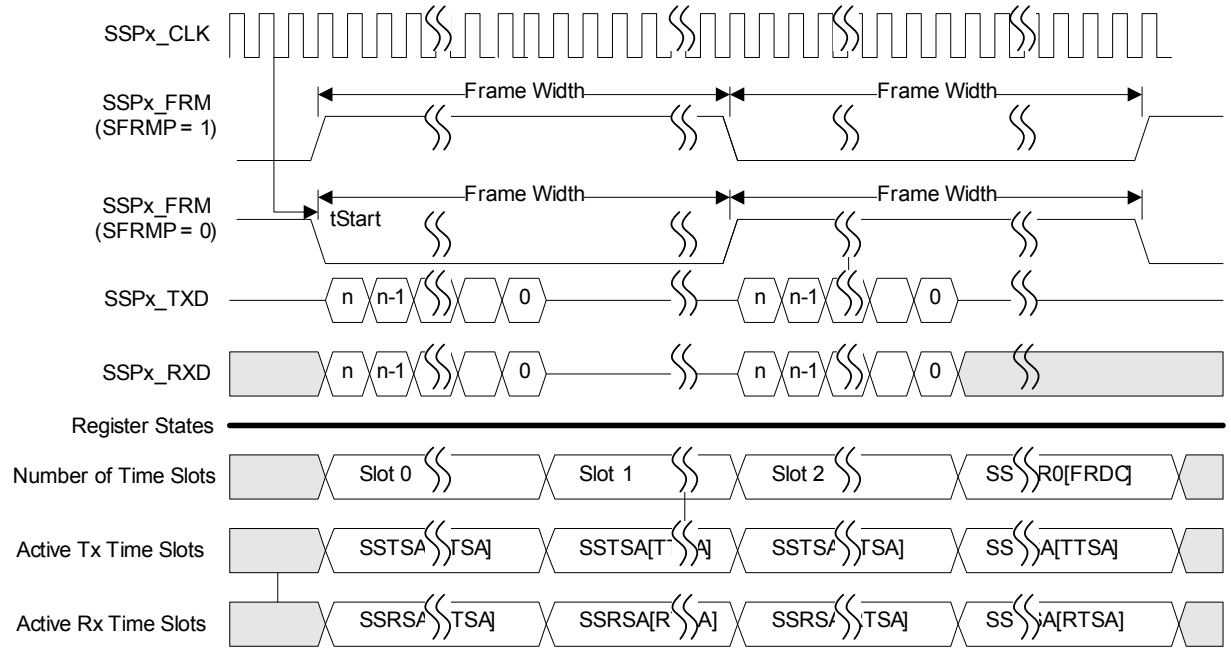
16.4.8 I²S Emulation Using SSP

Network Mode (SSCR0[MOD] = 1) is used along with the Programmable Serial Protocol (PSP) format to emulate I²S mode.

Figure 91 shows a frame cycle where 4 time slots are being used and time slots 0 and 3 are being accessed.

The total number of time slots to use is defined by the SSCR0[FRDC] register. The slots that data is read into is defined by the RTSA field in the SSTRA register, and the slots that data is transmitted out on is defined by the TTSA field in the SSTSA register. The number of bits used for each time slot is defined by the Data Size Select bits (EDSS, DSS) in the SSCR0 register. When using master mode (SSCR1[SFRMDIR] = 0x0) the Serial Frame Width (SSPSP[SFRMWDTH]) is used to determine the number of SSPx_CLK for the left and right channels.

Figure 91: Network Mode and PSP Frame Format

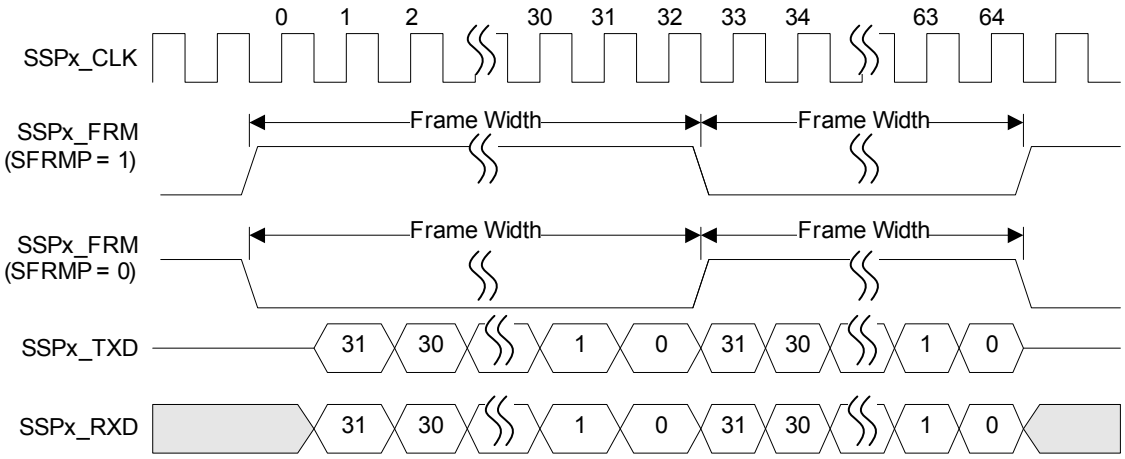


16.4.8.1 "Normal" Mode

The following bit fields must be configured for normal I^{2}S mode (see Figure 92):

- SSP_SSCR0[EDSS] = 0b1 (32-bit data)
- SSP_SSCR0[DSS] = 0b1111 (32-bit data)
- SSP_SSCR0[FRF] = 0b11 (PSP format)
- SSP_SSCR0[FRDC] = 0b01 (Number of Time Slots+1)
- SSP_SSCR1[SCLKDIR] = 0b0 (SSP port is master of SSPx_CLK)
- SSP_SSCR1[SFRMDIR] = 0b0 (SSP port is master of SSPx_FRM)
- SSP_SSPSP[SFRMWDTH] = 0b100000 (Frame Width – 32 SSPx_CLK cycles)
- SSP_SSPSP[SFRMP] = 0b0 (Frame Polarity – Low)
- SSP_SSPSP[FSRT] = 0b1 (Frame Sync Timing – Delay audio data 1 SSPx_CLK cycle after SSPx_FRM transition)
- SSP_SSPSP[SCMODE] = 0x0 (Data driven on falling edge and sampled on rising)
- SSP_SSPSP[DMYSTOP] = 0b00 (Extended Dummy Stop)
- SSP_SSPSP[EDMYSTOP] = 0b000 (Extended Dummy Stop)
- SSP_SSTSA[TTSA] = 0x3 (Transmit Active Time Slots)
- SSP_SSRSA[RTSA] = 0x3(Receive Active Time Slots)

Figure 92: Normal I^{2}S Format

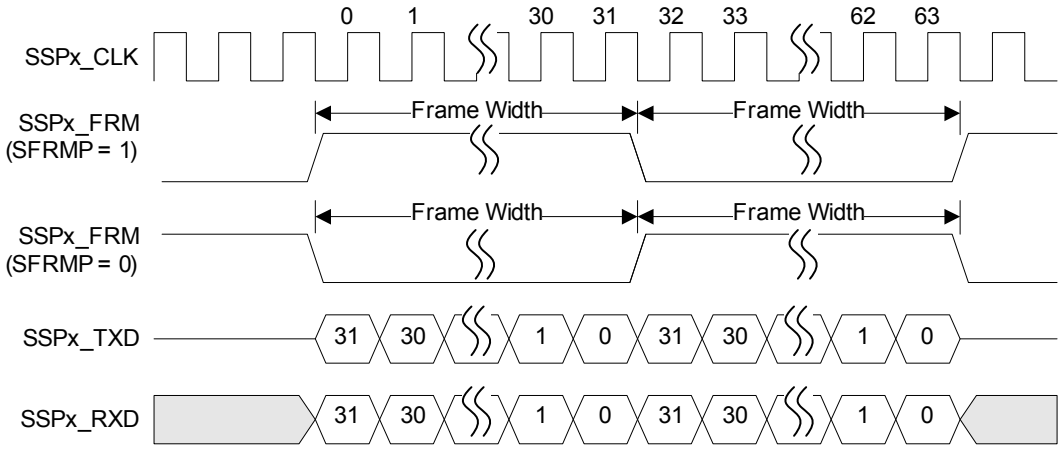


16.4.8.2 "MSb-justified" Mode

The following bit fields must be configured for MSb-Justified I^{2}S mode (see Figure 93):

- SSP_SSCR0[EDSS] = 0b1 (32-bit data)
- SSP_SSCR0[DSS] = 0b1111 (32-bit data)
- SSP_SSCR0[FRF] = 0b11 (PSP format)
- SSP_SSCR0[FRDC] = 0b01 (Number of Time Slots+1)
- SSP_SSCR1[SCLKDIR] = 0b0 (SSP port is master of SSPx_CLK)
- SSP_SSCR1[SFRMDIR] = 0b0 (SSP port is master of SSPx_FRM)
- SSP_SSPSP[SFRMWDTH] = 0b100000 (Frame Width – 32 SSPx_CLK cycles)
- SSP_SSPSP[SFRMP] = 0b0 (Frame Polarity – Low)
- SSP_SSPSP[FSRT] = 0b0 (Frame Sync Timing – Audio data aligned with SSPx_FRM)
- SSP_SSPSP[SCMODE] = 0x0 (Data driven on falling edge and sampled on rising)
- SSP_SSPSP[EDMYSTOP] = 0b000 (Extended Dummy Stop)
- SSP_SSTSA[TTSA] = 0x3 (Transmit Active Time Slots)
- SSP_SSRSA[RTSA] = 0x3 (Receive Active Time Slots)

Figure 93: MSb-Justified I^{2}S Format



16.5 Register Description

See Appendix A.9.2, SSP Registers, on page 614 for a detailed description of the registers.

17 USB OTG Interface Controller (USBC)

17.1 Overview

The 88MW300/302 USB Interface includes a Universal Serial Bus (USB) On-the-Go (OTG) capable dual-role host/device controller that is compliant with the USB 2.0 specification.

17.2 Features

- Full USB OTG functionality with integrated transceiver, allowing support for an Enhanced Host Controller Interface (EHCI) host or a device
- Supports High-Speed/Full-Speed/Low-Speed USB 2.0 Host/Device/OTG modes
- Up to 16 configurable bi-directional endpoints for device mode
  - I/O Transfer types supported – Control, Interrupt, Bulk, or Isochronous
  - Endpoint 0 – Dedicated for control endpoint
- Control signals for external power supply and detection of voltages for OTG signaling
- Capability to respond as self- or bus-powered device and control to allow charging from bus
- 2 KB TxFIFOs for each endpoint, which can hold the largest USB 2.0 packet
- 2 KB shared Rx buffer for all incoming data

17.3 Interface Signal Description

Table 143 shows the interface signals.

Note: After Power-On Reset (POR), if USB is in host mode, USB_DP/USB_DM will be SE0. If USB is in device mode, USB_DP/USB_DM will be High-z.

Table 143: USB OTG Controller Interface Signals

Signal Name	Type	Description
USB_OTG_P (USB_DP)	I/O	USB D+
USB_OTG_N (USB_DM)	I/O	USB D-
USB_VBUS	I/O	VBUS Selection Input in device mode; unused in host mode. Output mode VDD 2.1V to 5.25V.
USB_DRV_VBUS (GPIO_27)	O	Drive 5V on VBUS 0 = do not drive VBUS 1 = drive 5V on VBUS The USB_DRV_VBUS port is connected to the SoC pad to drive an external power management chip to provide power for USB_VBUS.
ID_PIN (USB_ID)	I	Pin Identification 0 = A-device 1 = B-device

Table 144 shows the Host Controller signals.

Table 144: USB Host Controller Interface Signals

Signal Name	Type	Description
USBH_P (USB_DP)	I/O	USB D+
USBH_N (USB_DM)	I/O	USB D-

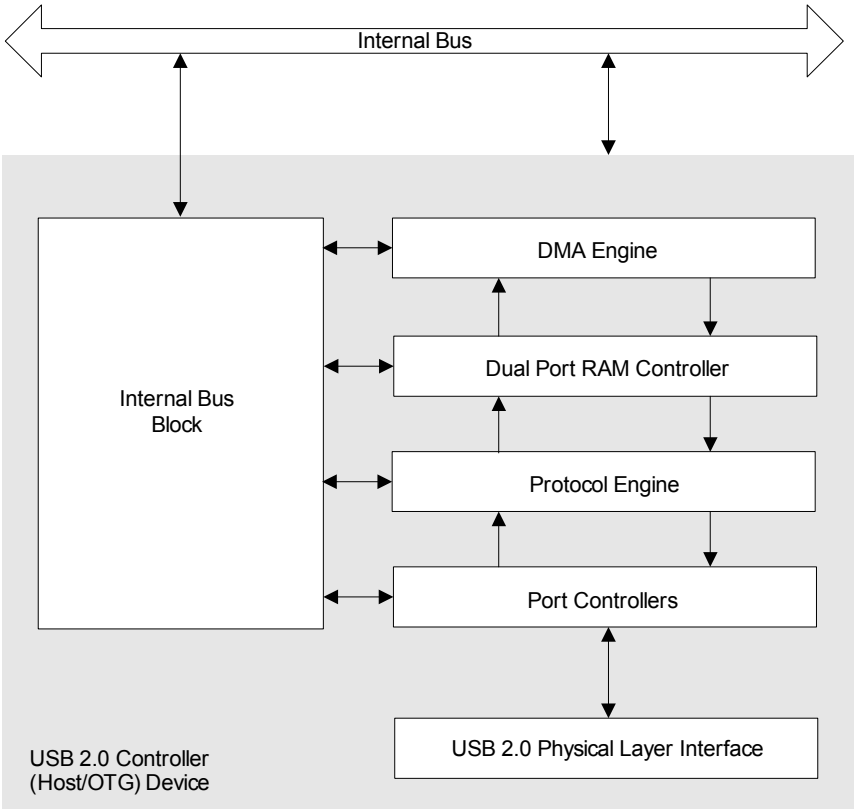
17.4 Internal Bus Interface

The internal bus contains all the control and status registers that allow a processor to interface to the USB core. These registers allow a microprocessor to control the configuration of the core, ascertain the capabilities of the core, and control the core in operation for both host and device modes.

17.4.1 Block Diagram

Figure 94 shows an overall block diagram.

Figure 94: USB Controller Block Diagram



17.4.2 DMA Engine

The DMA Engine Block presents a bus initiator (master) interface to the internal bus. It is responsible for moving all of the data to be transferred over the USB between the USB core and buffers in the system memory.

The DMA controller must access both the control information and packet data from the system memory. The control information is contained in the link list-based queue structures. The DMA controller has state machines that can parse all of the data structures defined in this controller specification.

17.4.3 Dual Port RAM Controller

The Dual Port RAM controller, which is used for context information, builds configurable FIFOs between the Protocol Engine block and the DMA controller. These FIFOs decouple the system processor memory bus request from the extremely tight timing required by the USB itself. The use of the FIFO buffers differs between host and device mode operation. In Host mode, a single data channel is maintained in each direction through the dual-port memory. In Device mode, multiple FIFO channels are maintained for each of the active endpoints in the system.

17.4.4 Protocol Engine

The Protocol Engine parses all the USB tokens and generates the response packets. It is responsible for checking for errors, checking field generation, formatting all handshaking, ping, and data response packets on the bus, and for generating any signals based on a USB-based time frame. In Host mode, the Protocol Engine also generates all of the token packets that are required by the USB protocol. The Protocol Engine contains several sub-functions:

- The token state machines track all of the tokens on the bus and filter the traffic based on the address and endpoint information in the token. In Host mode, these state machines also generate the tokens required for data transfer and bus control.
- The CRC5 and CRC16 CRC generator/checker circuits check and generate the CRC check fields for the token and data packets.
- The data and handshake state machines generate any responses required on the USB and move the packet data through the dual-memory FIFOs to the DMA controller block.
- The Interval timers provide timing strobes that identify important bus timing events: bus timeout interval, microframe interval, start of frame interval, and bus reset, resume, and suspend intervals.
- Reports all transfer status to the DMA engine.

17.4.5 Port Controller

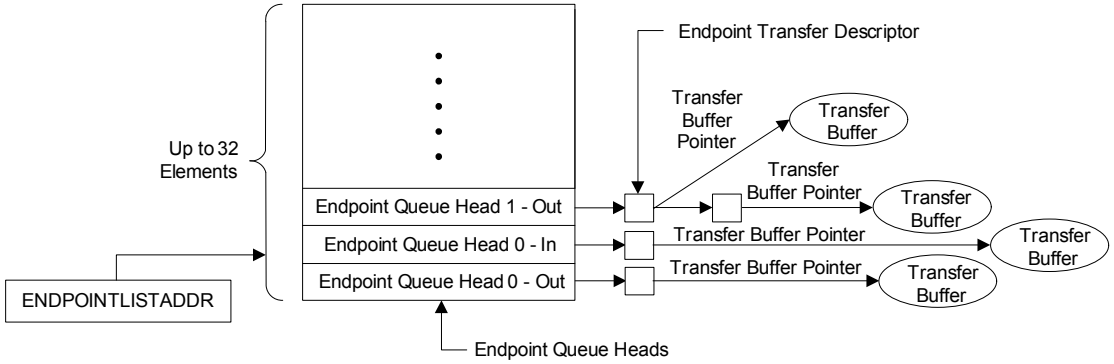
The Port controller block interfaces to the UTMI/UTMI+ compatible transceiver macrocell. The primary function of the Port controller block is to isolate the remainder of the USB core from the transceiver and to move all of the transceiver signaling into the primary clock domain of the USB core. This process allows the core to run synchronously with the system processor and its associated resources.

17.5 Functional Description

The USB OTG Controller is a fully-compliant USB peripheral device that can also assume the role of a USB host. The OTG state machines determine the role of the device based on the connector signals and then initializes the device in the appropriate mode of operation (host or peripheral) based upon its method of connection. After connecting, the devices can negotiate using the OTG protocols to assume the role of host or peripheral depending on the task to be accomplished. The attached peripherals share USB bandwidth through a host-scheduled, token-based protocol.

Figure 95 shows the endpoint queue head data structure.

Figure 95: End Point Queue Head Organization



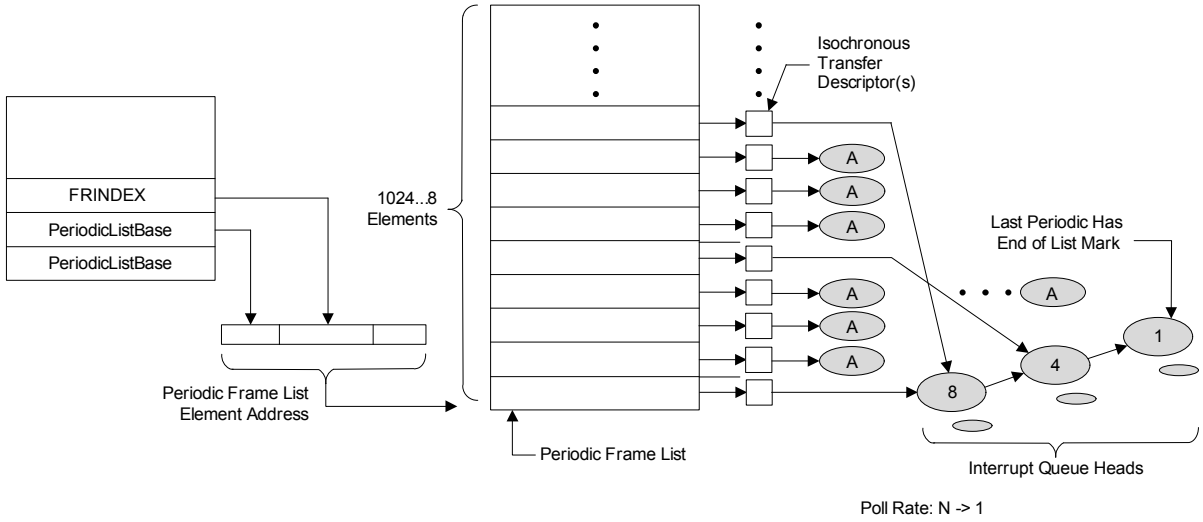
17.5.1 Host Data Structure

The Host data structures are used to communicate control, status, and data between software and the Host Controller.

The Periodic Frame List is an array of pointers for the periodic schedule. A sliding window on the Periodic Frame List is used. The Asynchronous Transfer List is where all the control and bulk transfers are managed.

Figure 96 shows the structure.

Figure 96: Periodic Schedule Organization



17.6 USB Controller Operation

17.6.1 FIFO Operation in Device Mode

17.6.1.1 Streaming Mode

Device Mode, ISO IN—Streaming mode

- The controller starts fetching data after the software primes the endpoint, and fills the TX FIFO to the available space.
- When the respective IN arrives from the Host, the controller sends the data, fetching more data from system memory to TX FIFO as space becomes available.

Device Mode, ISO OUT—Streaming mode

- When ISO OUT arrives and data is sent from Host, the device controller begins storing it to RX buffer.
- The device controller starts sending data from RX FIFO to system memory as soon as a burst size of data is available.
- After all the ISO OUT data is received, and while the RX FIFO still has data inside waiting to be transferred to system memory, the device controller can receive other packets.
- If only 1 RX FIFO position is free, or full, the device “BTO” the OUT/DATA from Host.
- If there is more than 1 position free, the device accepts OUT/DATA from Host but if the FIFO is not read to system memory, an overflow occurs anyway, and the packet is NAKed.

Device Mode, Bulk IN—Streaming mode

- When the Host sends the IN, if the device has not primed the EP yet, it “NAKs” the IN.
- After EP is primed, DMA engine in the device controller starts fetching data to the TX buffer.
- When the protocol engine part of the controller is primed (after synchronization), the device controller responds to an IN with data.
- Device controller continues to fetch data as space is available in TX buffer (at least 1 burst worth of data of free space available).
- If during a packet the system bus is unable to buffer all the data for that packet on time, the packet is cut short due to underrun.
- During a transfer, the device controller “NAKs” an IN if the TX buffer is empty, or more specifically, if no data was loaded for the next packet.

Device Mode, Bulk OUT—Streaming Mode

- When the Bulk OUT arrives and data is sent from Host, the device controller begins storing it to the RX buffer.
- The device controller starts sending data from RX FIFO to system memory as soon as a burst worth of data is available.
- After all the Bulk OUT data is received and the RX FIFO still has data inside waiting to be transferred to system memory, the device controller can receive other packets.
- If only 1 RX FIFO position is free, or full, the device “BTOs” the OUT/DATA from Host.
- If there is more than 1 position free, the device accepts OUT/DATA from the Host; if the FIFO is not read to system memory, an overflow occurs anyway, and the packet NAKed.
- This behavior is the same as for ISO OUT.

17.6.1.2 Additional Notes on TX FIFO Buffering – IN Endpoints

Initial Priming

- When priming has started on DMA side, the controller loads the leading data into the TX buffer, and only then completes the priming operation (PE is primed). This pre-buffering is performed for the entire first packet, or until the TX FIFO is full.

Buffering after First Packet

- After the first packet, the second packet in a dTD is sent as long as at least 1 byte was loaded to the FIFO for the second packet.
- Once FIFO is loaded with the first packet, the entire dTD (all the packets in 1 dTD) is sent for every IN from Host, and the system bus must continue back-filling the TX FIFO to keep up with data being sent for the several packets.

BTO or NAK from Host to Data Packet

- If a packet is NAKed or BTOed by Host, device flushes TX buffer and removes the priming state.
- It then returns to repeat the buffer operation.
- If an IN arrives from the host in the meantime, it is NAKed.
- Once the packet is fully loaded inside the TX buffer again, or the TX buffer is full, the prime state in the PE is set to active.
- Only then does the device controller respond to the Host IN token with the data packet.

Underrun of Device TX FIFO

- If a TX FIFO underrun occurs, device clears the prime, flushes TX buffer, and then reloads all of failing packet to the TXFIFO.
- At the same time, the device “NAKs” an IN from the Host.
- Only when entire packet is in FIFO again or FIFO is full does the Device respond to the Host IN with a data packet.

Buffering Between dTDs

- When the Host sends an ACK to the last data packet of the first dTD, the device disables the primed state.
- This last data packet can either be maximum packet sized, a short packet, or a 0-length packet, depending on the transfer total length and the ZLT bit in the Queue head. Either is ACKed by the Host, and the priming is disabled after that.
- The device then retires the dTD back to system memory and loads the new dTD.
- In the meantime, any IN packet from the Host is NAKed.
- After loading the new dTD, the device starts buffering the first packet into the TX buffer.
- Once the packet is fully loaded inside the TX buffer again, or the TX buffer is full, the prime state in the PE is set to active.
- Only then does the device controller responds to the Host IN token with the data packet.
- This behavior is similar to the initial priming of the first dTD.

17.6.1.3 Non-Streaming Mode

Device Mode, ISO IN—Non-Streaming Mode

- Same as Streaming mode

Device Mode, ISO OUT—Non-Streaming Mode

- Same as Streaming mode

Device Mode, Bulk IN—Non-Streaming Mode

- When the Host sends the IN, if the device has not yet primed the EP; it “NAKs” the IN.
- After EP is primed, the DMA engine in device controller starts fetching data to the TX buffer.
- The device controller continues fetching data as space is available in the TX buffer.
- The device controller responds only to the first IN with data when the entire packet is fetched to the TX FIFO or the TX FIFO is full. In the meantime, it responds to the IN with NAK.
- The following packets are sent only if next packet is fully in FIFO, or if the FIFO is full again.

Device Mode, Bulk OUT—Non-Streaming Mode

- When Bulk OUT arrives and data is sent from Host, the device controller begins storing it to the RX buffer.
- The device controller starts sending data from RX FIFO to system memory as soon as a burst size of data is available.
- The device controller responds with NYET to this first packet of a transfer, so the Host sends PINGs after that.
- The device controller sends NAKs to the PINGs as long as data is still in the RX FIFO (RX FIFO not yet empty).
- Only when the RX FIFO is empty again does the device controller ACK the PING, and the Host sends the next IN.

17.6.1.4 FIFO Operation in Host Mode

Host Mode, ISO IN/OUT—Streaming Mode

- Taking the example of using Streaming mode: It is a 760-byte transfer, using a max packet size of 370 bytes, MULT=3.

For OUT Direction

- Assuming a watermark value of 512 bytes (larger than the 370-byte packet), the controller uses the packet size as watermark.
- Behavior is the same as in Non-Streaming mode.
- Host controller starts fetching the data to TX FIFO after the SOF is sent.
- In Host ISO OUT, the TXFIFOTHRES watermark applies to every packet in a transfer.
- Assuming a packet size if 1024 bytes, TX FIFO with a size of 1024 bytes, and a TX watermark of 512 bytes as well: The DMA writes the Packet Start TAG to the TX FIFO, and then proceeds to fill with data. As the Start TAG is read just after it is written, 512 bytes are again available on the FIFO. The DMA then fills the TX FIFO to the 512 bytes watermark.
- Only then does the controller send the OUT token and start the packet. As the controller reads the first bytes from the TX FIFO, it makes space available for the missing End Packet TAG, which then fits without any problems.
- If the packet size is 1024 for a transfer size of 3072, MULT=3, and using a TX watermark of 512 bytes, the behavior is the same as in the previous example, except that when the first packet starts being sent, the controller backfills the TX FIFO with the remaining data for the packet, as soon as 1 burst of data space is available. The behavior is the same for each of the MULT=3 packets.
- Using an ISO packet size larger than the TX FIFO size is not a problem, as long as the system bus has enough bandwidth to backfill the TX buffer after the packet started being sent.
- For a packet size = 1024 bytes, TX watermark set to 512 and TX FIFO size is 1024, the controller fills the TX FIFO to 512 bytes before sending the OUT token, and then continues backfilling the FIFO.

For IN Direction

- Behavior is the same as in Non-Streaming mode in the sense that the Host Controller is working with 1 packet at a time.
- Controller starts sending data from RX FIFO to system memory as soon as 1 burst of data is available.
- Controller always waits for all the packet data to be stored in system memory, and only then proceeds to the next packet; that is, it sends only the IN token for the next packet when the RX FIFO is empty.
- If the packet size is 1024 for a transfer size of 3072, MULT=3, the behavior is the same, each packet is taken care of as described above.

Host Mode, ISO IN/OUT—Non-Streaming Mode

The available slot for HS ISO within a micro-frame is up to 80%, which means that the buffering could extend until that time is exhausted with the resulting side effect of wasted bandwidth. It does work, but can become a low efficiency method.

Using Non-Streaming mode, 760 byte transfer, using a maximum packet size of 370 bytes, MULT=3.

For OUT Direction

- After the SOF is sent, the host controller begins fetching the data from memory to the TX FIFO.
- Only after the first 370 bytes are fetched (a full packet), it sends the first OUT token to the line.
- During the first OUT/DATA, the controller continues fetching data for the remaining packets (370+20 bytes).
- Controller continues fetching data (within the limit of the buffer size) until all data is fetched and all packets are sent;
- Non-Streaming mode is the same behavior as Streaming mode with the TX Fill level set to the same size as the TX FIFO.
- If the packet size is 1024 for a transfer size of 3072, MULT=3, the behavior is the same as in the previous example, except that when the first packet starts being sent, the controller backfills the TX FIFO with the remaining data for the packet, as soon as 1 burst of data space is available. The behavior is the same for each of the MULT=3 packets.
- For packet size = 1024 bytes, TX watermark set to 512 and TX FIFO size is 2048, the controller fills the TX FIFO to 1024 bytes before sending the OUT token.

For IN Direction

- After the SOF is sent, the Host Controller starts fetching the iTD from system memory. As soon as the iTD has been read, it issues the IN token if the RX buffer is empty.
- While data is being received from the device and being stored to the RX buffer, the controller writes data to system memory as soon as 1 burst worth of data is available.
- Only after all data has been stored from the RX buffer to system memory does the Host Controller issue the second IN, and the same for the third IN.
- When sending the IN, if the RX buffer is not empty at the calculated time, the host delays issuing IN token until the buffer is empty of packet data.
- If the packet size is 1024 for a transfer size of 3072, MULT=3, the behavior is the same, each packet is taken care of as described above.

Host Mode, Bulk OUT—Streaming Mode

For every packet in a transfer, the Host Controller pre-fetches the data until the TX FIFO is filled up to the level specified by the TXFIFOTHRES Register. Only then is the OUT token sent while the controller continues fetching more data to the TX FIFO. This is the same TXFIFOTHRES behavior as in ISO OUT.

TXFIFOTHRES is set in number of bursts; that is, if TXFIFOTHRES=2, the actual watermark level is, for example, 2xINCR8, or 2x (INCR of VUSB_HS_TX_BURST length)

- If the TX FIFO is full, the Host Controller re-fetches data for the next packet as soon as there is 1 burst worth of free space available on the TX FIFO, regardless of the current packet being sent. So as soon as the first bytes are transmitted for the current packet, the Host Controller starts fetching for the next. This method is valid in Streaming and Non-Streaming modes.

Host Mode, Bulk IN—Streaming Mode

Host issues IN token when the RX FIFO is empty which is valid for all packets in a transfer (qTD).

- The Host Controller starts sending data from RX FIFO to system memory as soon as there is 1 burst worth of data available on the RX FIFO (example: 1x INCR8, or 1x(INCR of VUSB_HS_ TX_BURST length)), regardless of the current packet being received. This is valid in Streaming and Non-Streaming modes.

Host Mode, Bulk OUT—Non-Streaming Mode

Host issues OUT token when entire data packet is in the TX FIFO, or the FIFO is full.

- The Host Controller starts fetching data for the next packet as soon as there is 1 burst worth of free space available on the TX FIFO, regardless of the current packet being sent. So as soon as the first bytes are transmitted for the current packet, the Host Controller starts fetching for the next. This is valid in Streaming and Non-Streaming modes.

Host Mode, Bulk IN—Non-Streaming Mode

Host issues IN token when the RX FIFO is empty.

The Host Controller starts sending data to system memory as soon as there is 1 burst worth of data available on the RX FIFO, regardless of the current packet being received. This is valid in Streaming and Non-Streaming modes.

17.6.2 Clock Control and Enables

The USB OTG and Host Controllers have separate clock enables for their system and USB clocks.

The USB Clock/Reset Control Register (APULL_CTRL0) is used to enable the AXI interface to the USB OTG and Host controllers.

17.6.3 Programming Guidelines

Both USB controllers receive their clocks from their respective USB UTMI physical layer interfaces.

To use the USB EHCI host controller or the USB EHCI OTG controller their physical layer interfaces must be initialized prior to any USB controller operation such as register write access.

17.7 Register Description

See Appendix A.3, USBC Address Block, on page 413 for a detailed description of the registers.

18 Quad Serial Peripheral Interface (QSPI) Controller

18.1 Overview

The 88MW300/302 includes a QSPI Controller. The QSPI Controller is muxed with the Flash Controller to connect to an external serial Flash device.

The QSPI Controller is a synchronous serial peripheral that can be connected to a variety of slave devices that communicate using SPI protocol for data transfer. The QSPI Controller always operates as a master and supports standard single bit, and high performance dual/quad output SPI as well as dual/quad I/O SPI. The QSPI Controller has an extremely flexible architecture where the command type, instruction encode, amount of data to be transferred and other parameters are all configurable through memory mapped registers.

18.2 Features

- Supports Standard SPI protocol with single bit Data In and Data Out
- Supports dual/quad output operations
- Supports dual/quad I/O operations
- Supports DMA and non-DMA modes for data transfer
- Separate FIFO for transmit and receive with the length of 8*32 bit
- Support for interrupts for a variety of events and conditions related to FIFOs
- 200 Mbps maximum serial data rate in quad mode with 50 MHz functional clock

18.3 Interface Signal Description

Table 145 shows the interface signals.

Table 145: QSPI Interface Signals[1]

Signal Name	Type	Width/Bit	Description
GPIO_29/QSPI_CLK	O	1	QSPI bit clock
GPIO_28/QSPI_SSn	O	1	QSPI chip select, active low
GPIO_30/QSPI_D0	I/O	1	Data I/O 0
GPIO_31/QSPI_D1	I/O	1	Data I/O 1
GPIO_32/QSPI_D2	I/O	1	Data I/O 2
GPIO_33/QSPI_D3	I/O	1	Data I/O 3

1. See Section 22.11.2, QSPI Timing and Specifications, on page 341 for electrical specifications.

Note: GPIO pins are multiplexed with QSPI pins. Therefore, software must configure the appropriate PINMUX registers to use them as QSPI pins. See Section 6.2.1, PINMUX Alternate Functions, on page 143 and Appendix A.17, PINMUX Address Block, on page 747.

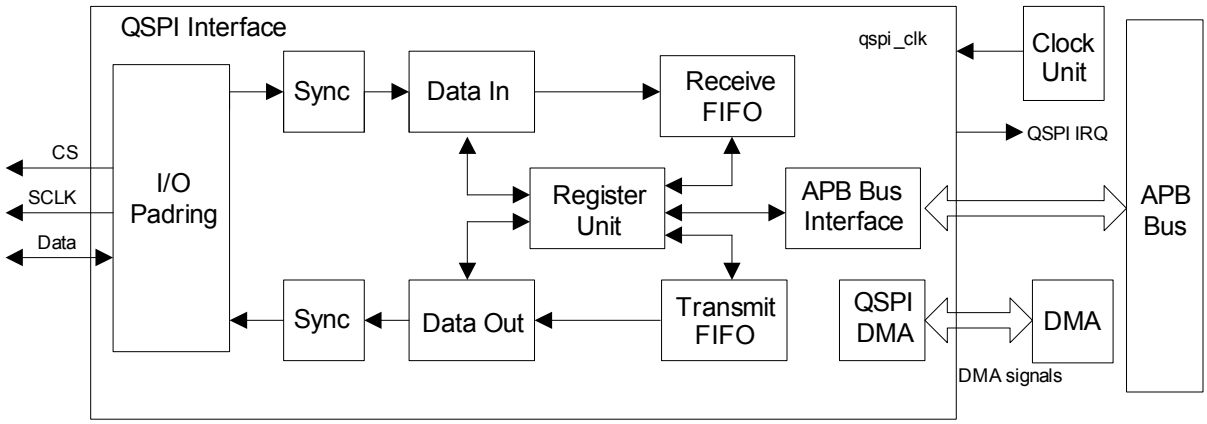
18.4 Functional Description

Serial data is transferred between the 88MW300/302 processor and serial peripheral through the FIFOs in the QSPI Controller. QSPI always operates as a master providing the Serial bit clock and Chip-Select or Frame-Sync. The Controller supports both the DMA and non-DMA modes of transferring data.

18.4.1 Block Diagram

Figure 97 shows an overall diagram.

Figure 97: QSPI Controller Block Diagram



18.4.2 Basic Operation

QSPI always operates as a master and can be configured to generate read or write transactions to the attached slave device. There are 2 varieties of frames that can be generated by the QSPI (Read frame and Write frame). A Read frame basically consists of instruction encode, address of the location to read from and data itself. A Write frame consists of a similar structure and consists of again instruction, address followed by outputting data.

Some attached slaves that need a few extra cycles for data setup for high performance operation can be supported by configuring the QSPI to generate “dummy clocks.”

For Write transactions, the required fields are:

- Instr.INSTR – determines the instruction encode or the command
- HdrCnt.INSTR_CNT – determines the number of clocks or bytes required for instruction encode
- HdrCnt.ADDR_CNT – determines number of bytes or clocks of address
- Addr.ADDR – determines the address inside the slave
- HdrCnt.DUMMY_CNT – determines the number of extra or dummy clocks required by a slave

For Read transaction, the required fields are:

- Instr.INSTR – determines the instruction encode or the command
- HdrCnt.INSTR_CNT – determines the number of clocks or bytes required for instruction encode
- HdrCnt.ADDR_CNT – determines number of bytes or clocks of address.
- Addr.ADDR – determines the address inside the slave
- HdrCnt.DUMMY_CNT – determines the number of extra or dummy clocks required by a slave
- DInCnt.DATA_IN_CNT – number of bytes of data to be transferred

QSPI Single, Dual or Quad mode operation is configurable by programming the Conf.DATA_PIN and Conf.ADDR_PIN fields.

Different values on Conf.DATA_PIN signify:

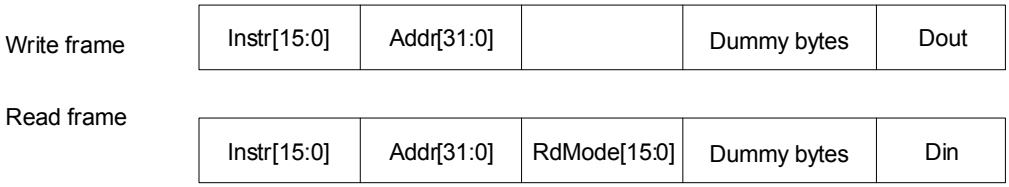
- 00 = use 1 serial interface pin (use in single mode)
- 01 = use 2 serial interface pins (use in dual mode)
- 10 = use 4 serial interface pins (use in quad mode)

Different values on Conf.ADDR_PIN signify:

- 0 = use 1 serial interface pin
- 1 = use the number of pins as indicated in Conf.DATA_PIN ()

18.4.3 Serial Flash Data Format

Figure 98: Frame of Data Format for Serial Flash Access



- Instr – Serial Interface Instruction

  After Conf.XFER_START is set to 1, the content of the Instr register is shifted out to the serial interface. HdrCnt.INSTR_CNT determines how the content is shifted out.

  - When HdrCnt.INSTR_CNT = 0, the content of this register is not shifted out to the serial interface
  - When HdrCnt.INSTR_CNT = 1, bits [7:0] are shifted out
  - When HdrCnt.INSTR_CNT = 2, bits [15:8] are shifted out first, followed by bits [7:0]
- Addr – Serial Interface Address

  After the contents of the Instr register is shifted out, the content of the Addr register is shifted out to the serial interface. HdrCnt.ADDR_CNT determines how the content of the Addr register is shifted out on the serial interface.

  - When HdrCnt.ADDR_CNT = 0, the content of this register is not shifted out to the serial interface
  - When HdrCnt.ADDR_CNT = 1, bits [7:0] are shifted out
  - When HdrCnt.ADDR_CNT = 2, bits [15:8] are shifted out first, followed by bits [7:0]
  - When HdrCnt.ADDR_CNT = 3, bits [23:16] are shifted out first, followed by bits [15:8], then bits [7:0]
  - When HdrCnt.ADDR_CNT = 4, bits [31:24] are shifted out first, followed by bits [23:16], then bits [15:8] and finally bits [7:0]
- RdMode – Serial Interface Read Mode

  After the contents of the Addr register is shifted out, the content of the RdMode register is shifted out to the serial interface.HdrCnt.RM_CNT determines how the contents of the RdMode register are shifted out.

  - When HdrCnt.RM_CNT = 0, the content of this register is not shifted out to the serial interface
  - When HdrCnt.RM_CNT = 1, bits [7:0] are shifted out
  - When HdrCnt.RM_CNT = 2, bits [15:8] are shifted out first, followed by bits [7:0]

- Dummy_byte bytes to shift out to the serial interface after the contents of RdMode register is shifted out. The number of dummy bytes shifted out is determined by HdrCnt.DUMMY_CNT. Different values on HdrCnt.DUMMY_CNT signify
- 00: 0 byte
- 01: 1 byte
- 10: 2 bytes
- 11: 3 bytes

- DOut – Serial Interface Data Out

  Data written to the DOutregister is stored in the 8X32 bit Write FIFO. After the contents of the Instruction register (Instr), the Address register (Addr), the Read Mode register (RdMode) and Dummy value are transferred out to the serial interface, the data in the Write FIFO is shifted out. The serial interface clock stops when a Write FIFO empty condition occurs, i.e. Cntl.WFIFO_EMPTY = 1. The clock restarts when Write FIFO is not empty, i.e Cntl.WFIFO_EMPTY = 0. Conf.BYTE_LEN determines the number of bytes shifted out on the serial interface.

  - When Conf.BYTE_LEN = 0, only the first byte, for example, bits[7:0] of the Write FIFO is shifted out with bit 7 shifted out first and bit 0 shifted out last.
  - When Conf.BYTE_LEN = 1, all 4 bytes from each the Write FIFO are shifted out with bits [7:0] are shifted out (bit 7 shifted out first and bit 0 shifted out last), followed by bits [15:8] (bit 15 shifted out first and bit 8 shifted out last), then bits [23:16] (bit 23 shifted out first and bit 16 shifted out last) and finally bits [31:24] (bit 31 shifted out first and bit 24 shifted out last).

  Note: To avoid a Write FIFO overflow condition (Cntl.WFIFO_OVRFLW = 1), check if Cntl.WFIFO_FULL = 0 before writing to the DOut register.

- DIn – Serial Interface Data In

  For read transfers, Conf.RW_EN = 0, data from the serial interface input pins are shifted in and stored in a 8X32 bit Read FIFO. The contents of the Read FIFO are read from this register. The serial interface clock stops when a Read FIFO full condition occurs, i.e Cntl.RFIFO_FULL= 1. The clock restarts when Read FIFO is not full, that is, Cntl.RFIFO_FULL= 0.

  - When Conf.BYTE_LEN = 0, data is shifted into bits [7:0] of the Read FIFO
  - When Conf.BYTE_LEN = 1, data is shifted into bits [7:0] first, followed by bits [15:8], then bits [23:16] and finally bits [31:24]

  Note: To avoid a Read FIFO underflow condition, Conf.RFIFO_UNDRFLW = 1, check if Conf.RFIFO_EMPTY=0 before reading the DIn register.

DMA transfer is supported in QSPI functions. The specific bits in register QSPI.CONF2 must be set for the DMA transfer.

Figure 99 to Figure 102 show the data flow for Read and Write transactions using non-DMA and DMA operation of the QSPI Controller.

Figure 99: Non-DMA Mode Read Flow

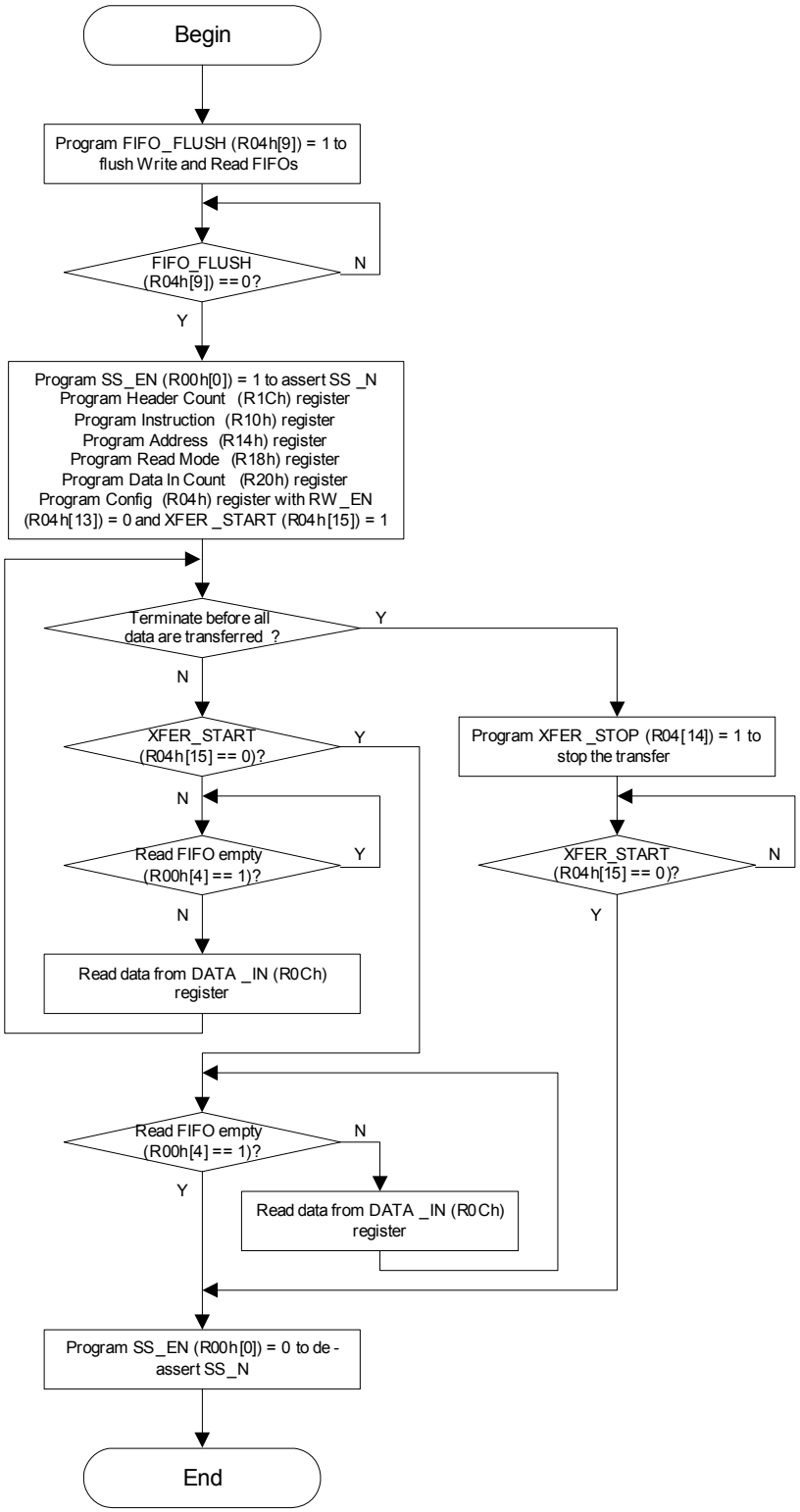


Figure 100:Non-DMA Mode Write Flow

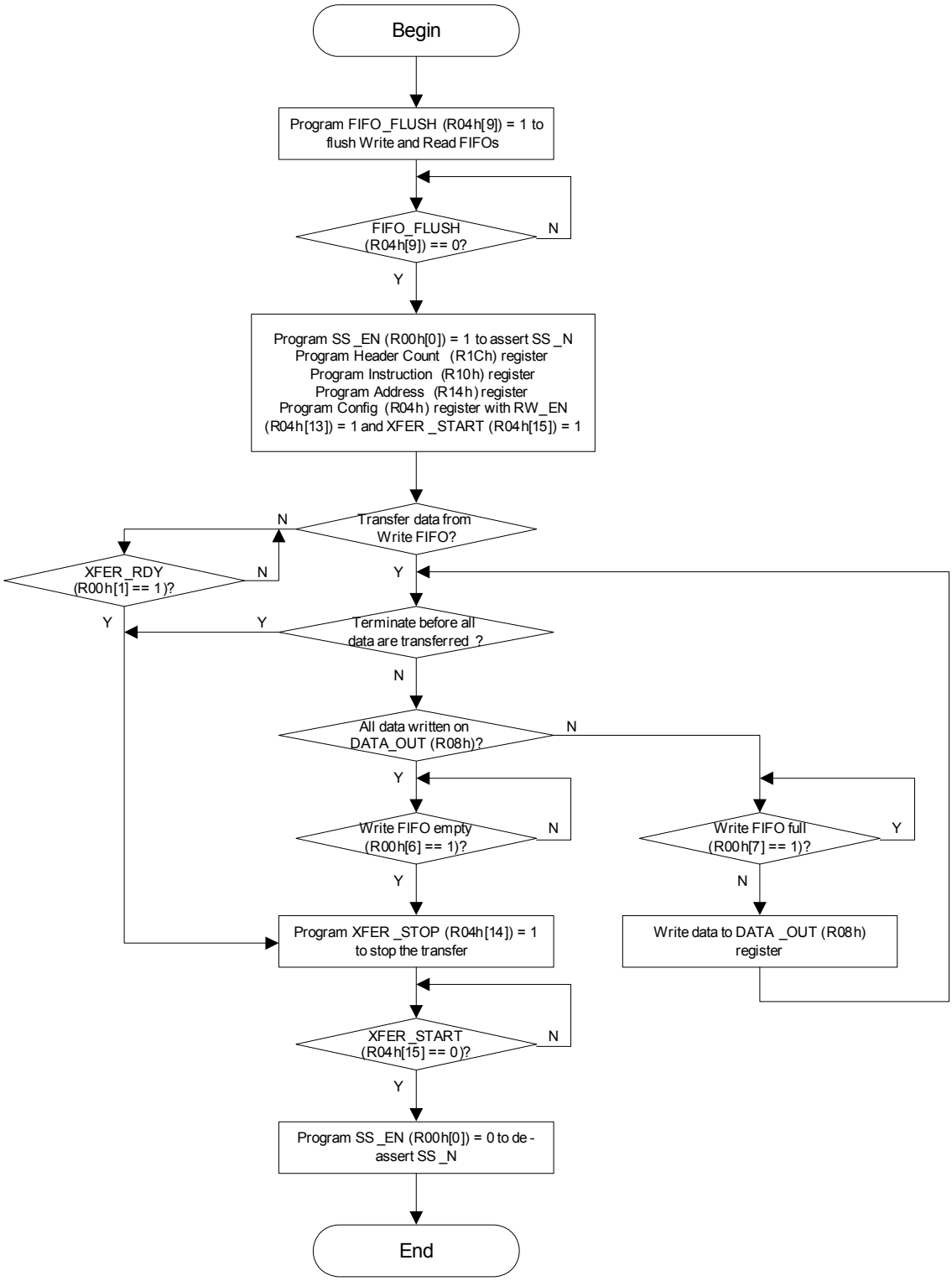


Figure 101:DMA Mode Read Flow

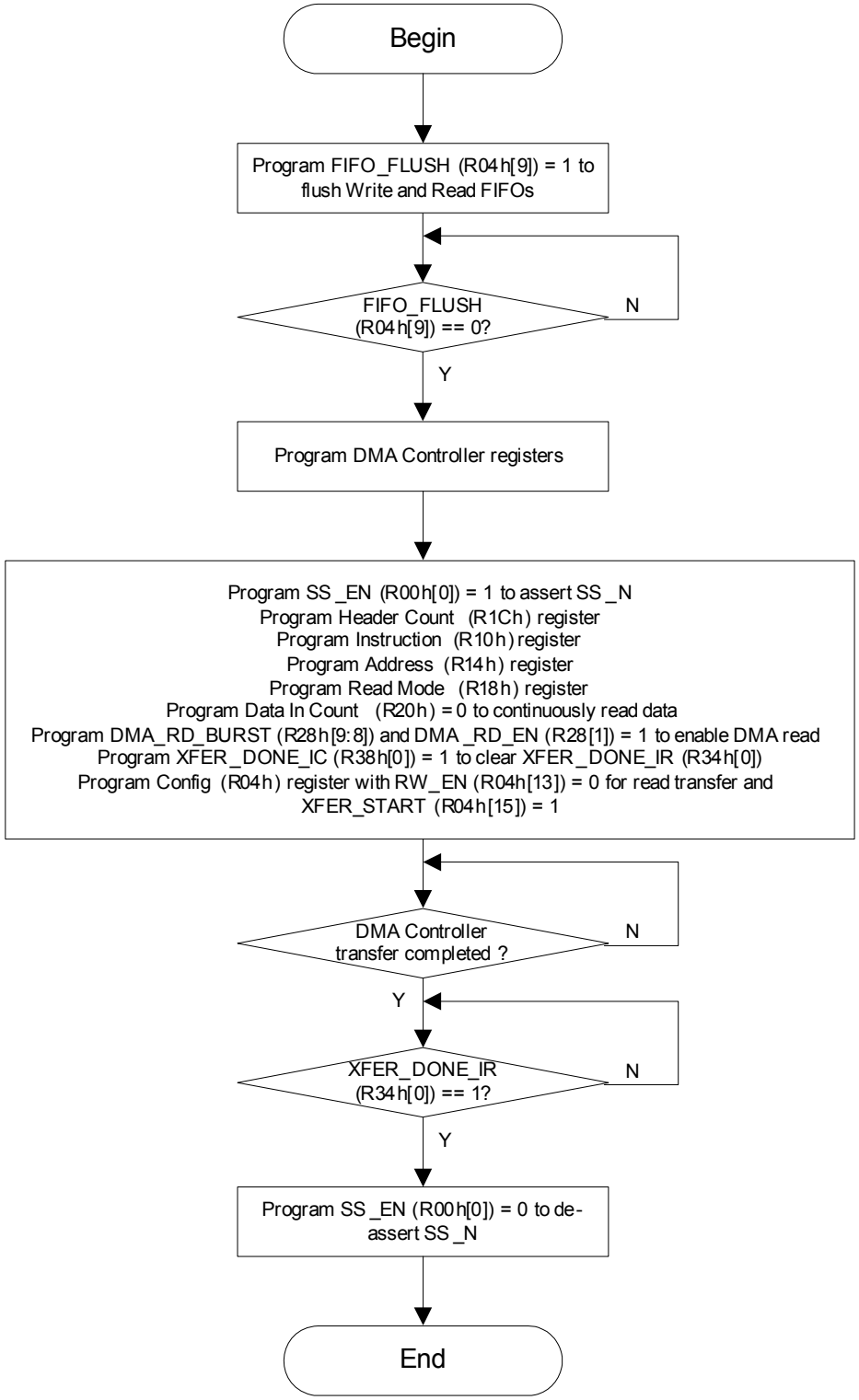
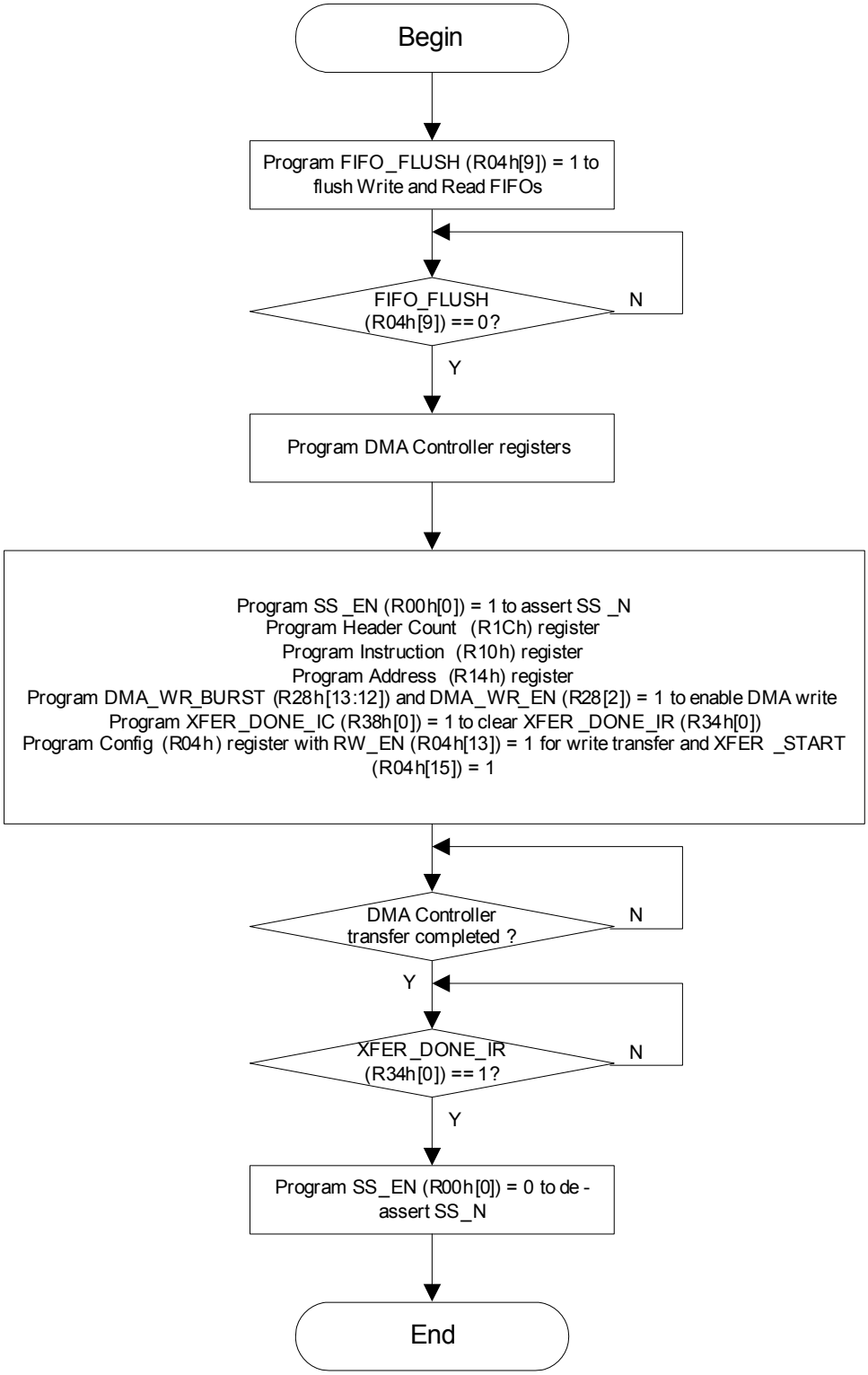


Figure 102:DMA Mode Write Flow



18.5 Register Description

See Appendix A.8.2, QSPI Registers, on page 594 for a detailed description of the registers.

THIS PAGE INTENTIONALLY LEFT BLANK

19 Analog Digital Converter (ADC)

19.1 Overview

The 88MW300/302 ADC is a 2-step converter with up to 16-bit resolution. The ADC includes an Analog Multiplexer (AMUX) and a Programmable Gain Amplifier (PGA) with as many as 8 individually configurable channels and a reference voltage regulator. The conversion results can be written to memory through the DMA. Several modes of operation are available.

19.2 Features

- Selectable throughput rates and resolution (12 to 16 bits)
- Throughput rate as fast as 2 MHz
- Single-ended and differential conversions from 8 external and 6 internal sources
- ADC gain setting support: 0.5x, 1x, 2x
- Additional PGA setting support: 4x, 8x, 16x, 32x
- Selectable reference voltage (Vref)
  - Internal reference 1.2V (Vref_12)
  - Vref_18
  - External reference (do not exceed 1.8V)
- Input voltage ranges (differential)
  - -Vref/PGA to + Vref/PGA
  - Do not exceed VDDIO_3 voltage level
  - Do not exceed VDD_IOx_y voltage level
- Offset and gain auto calibration
- Embedded temperature sensor with internal or external diode options
- DAC dual inputs
- Sequences with scan length up to 16
- Sequential conversion composed of any channel in any order
- 1-shot or continuous mode
- Scan average of 1, 2, 4, 8, 16
- Interrupt generation and/or DMA request
- Internal GPT trigger on ADC conversion
- Battery measurement capability

19.3 Interface Signal Description

The external interface of the ADC module is mainly given by the power and analog pads. The ADC clock is 64 MHz or 67.07 MHz and is provided by the AUPLL.

The analog unit uses an ADC clock divided-down clock (divided by 1 to 32). Access to the configuration and data registers of the ADC is provided through an APB. The digital unit internal to the ADC runs with the APB bus clock.

See Table 36, VCO Frequency Select, on page 97/Table 37, AUPLL Post Divider Programming, on page 98 for FVCO and clock divider settings.

- For normal ADC mode/temperature sensor mode, set FVCO to 128 MHz and the post divider value to 2 for 64 MHz operation.
- For voice applications, set FVCO to 134.14 MHz and the post divider value to 2 for 67.07 MHz operation.

Table 146 shows the interface signals.

Table 146: ADC Interface Signals[1]

Signal Name	Type	Source/ Destination	Description
ADC0_CH[7:0]	A, I	GPIO to ADC	External Analog Inputs from GPIO ADC0_CH[7]: GPIO_49 ADC0_CH[6]: GPIO_48 ADC0_CH[5]: GPIO_47 ADC0_CH[4]: GPIO_46 ADC0_CH[3]: GPIO_45/External voltage reference (EXT_VREF) ADC0_CH[2]: GPIO_44/DACA ADC0_CH[1]: GPIO_43/External diode negative (TS_INN)/DACB/VOICE_N ADC0_CH[0]: GPIO_42/External diode positive (TS_INP)/VOICE_P

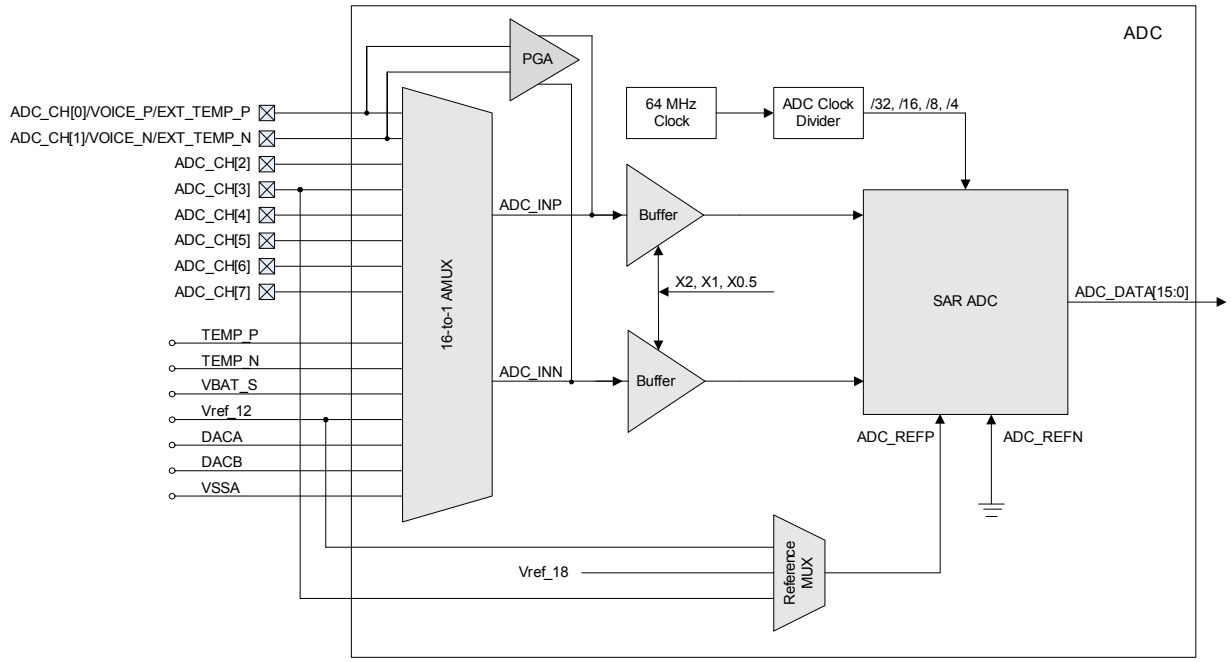
1. See Section 22.8, ADC Specifications, on page 330 for electrical specifications.

19.4 Functional Description

19.4.1 Block Diagram

Figure 103 shows the block diagram.

Figure 103:ADC Block Diagram



19.4.2 ADC On-Off Control and Conversion Trigger

The ADC can be directly reset by system reset or the ADC_REG_CMD.SOFT_RST bit.

The ADC is powered up by setting the ADC_REG_GENERAL.GLOBAL_EN bit to 1'b1, and it is fully powered down when this bit is set to 1'b0.

After the ADC is powered up, the data conversion can be activated by writing a 1'b1 to the ADC_REG_CMD.CONV_START bit if the ADC is in software control mode (ADC_REG_CONFIG.TRIGGER_EN = 1’b0).

The actual conversion starts after the ADC wakes up (T_{WARM}) from power-down mode where T_{WARM} is 32 us by default. The conversions can be stopped by writing a 1’b0 to the ADC_REG_CMD.CONV_START bit.

The ADC_REG_STATUS.ACT bit is set to high when the ADC is actively converting.

19.4.3 ADC Input

The ADC module can support as many as 8 external inputs. The actual input channels depend on the package types. Refer to the PINMUX function for details.

There are 8 external inputs that can be selected as 8 different single-ended inputs or 4 differential inputs. In addition, it is possible to select 6 single-ended internal inputs. The available selections are given in the ADC_REG_ANA.SINGLEDIFF and scan sequence registers ADC_REG_SCN1/2.

Table 147 shows the various ADC input configurations.

Table 147: ADC Input Configurations

NOTE: AMUX_SEL refers to ADC_REG_SCN1/2.SCAN_CHAN_X.

SINGLEDIFF/ AMUX_SEL[3:0]	ADC Positive Input	ADC Negative Input	Description
0/0000	ADC_CH[0]	VSSA	External single-ended or Temperature sensor (external diode)
0/0001	ADC_CH[1]	VSSA	External single-ended
0/0010	ADC_CH[2]	VSSA	External single-ended
0/0011	ADC_CH[3]	VSSA	External single-ended
0/0100	ADC_CH[4]	VSSA	External single-ended
0/0101	ADC_CH[5]	VSSA	External single-ended
0/0110	ADC_CH[6]	VSSA	External single-ended
0/0111	ADC_CH[7]	VSSA	External single-ended
0/1000	VBAT_S	VSSA	Internal single-ended (nominal 1/3 VBAT)
0/1001	Vref_12	VSSA	Internal single-ended (internal reference 1.2V)
0/1010	DACA	VSSA	Internal single-ended (DACA internal output)
0/1011	DACB	VSSA	Internal single-ended (DACB internal output)
0/1100	VSSA	VSSA	Internal single-ended (internal analog ground)
0/1101-0/1110	Reserved	Reserved	Reserved
0/1111	TEMP_P	VSSA	Temperature Sensor (internal diode) • Internal Temperature Sensor ADC_REG_ANA.TSEXT_SEL=0 • External Temperature Sensor ADC_REG_ANA.TSEXT_SEL=1
1/0000	ADC_CH[0]	ADC_CH[1]	External differential or Temperature sensor (external diode)
1/0001	ADC_CH[2]	ADC_CH[3]	External differential
1/0010	ADC_CH[4]	ADC_CH[5]	External differential
1/0011	ADC_CH[6]	ADC_CH[7]	External differential
1/0100	DACA	DACB	Internal differential

Table 147: ADC Input Configurations (Continued)
NOTE: AMUX_SEL refers to ADC_REG_SCN1/2.SCAN_CHAN_X.

SINGLEDIFF/ AMUX_SEL[3:0]	ADC Positive Input	ADC Negative Input	Description
1/1111	TEMP_P	TEMP_N	Temperature sensor (internal diode) • Internal Temperature Sensor ADC_REG_ANA.TSEXT_SEL=0 • External Temperature Sensor ADC_REG_ANA.TSEXT_SEL=1
1/0101-1/1110	Reserved	Reserved	Reserved

19.4.4 Input Range

When configured to single-ended input, the voltage sampled is the difference between the input channel and VSSA:

ΔV (differential voltage) = VIN_CH (input channel) – VSSA

When configured to differential input, the voltage sampled is the difference between the odd and even channels:

ΔV (differential voltage) = VIN_EVEN (even channel) – VIN_ODD (odd channel)

The input voltage for each external channel must be positive and cannot exceed the VDDIO_3 voltage level and VDD_IOx_y voltage level. For 16-bit and 16-bit audio settings, the input signal should be limited to within 99.7% of the ADC voltage conversion range to avoid overflow in offset/gain calibration.

Table 148 shows the detailed input range.

Table 148: Detailed Input Range

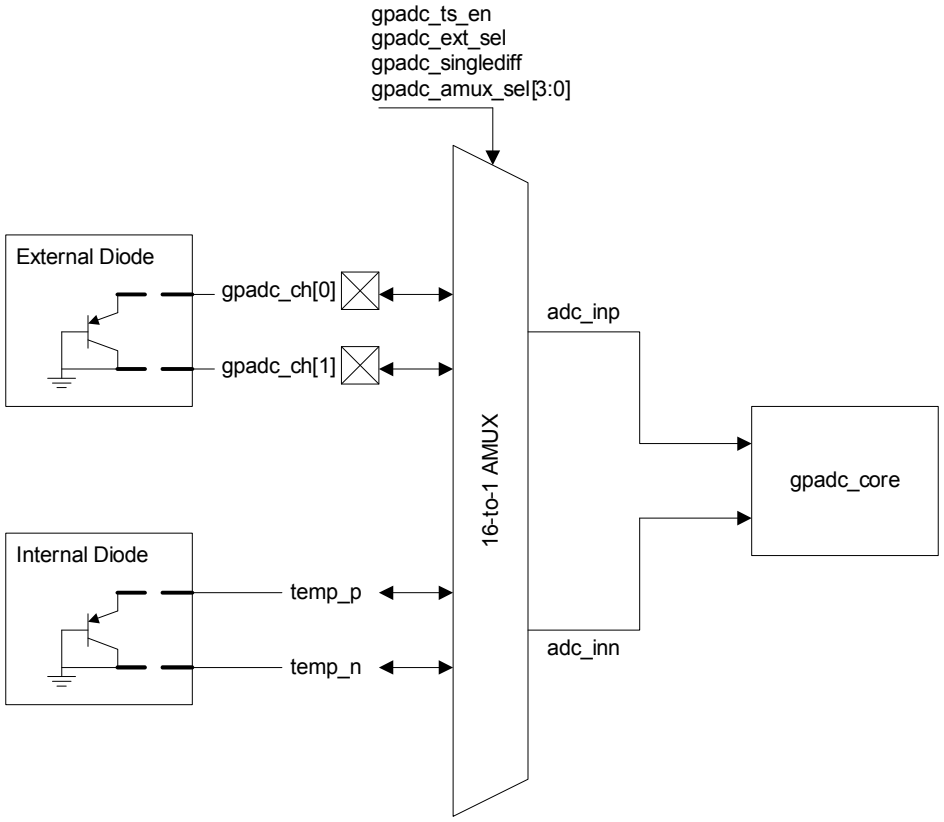
Single-ended or Differential Input (ADC_REG_ANA.SINGLE_DIFF bit)	ADC Gain Setting (ADC_REG_ANA.INBUF_GAIN[1:0] bits)	Input Range (ΔV)
single-ended	0.5	0 to 2*Vref
single-ended	1	0 to Vref
single-ended	2	0 to 0.5*Vref
differential	0.5	-2*Vref to 2*Vref
differential	1	-Vref to Vref
differential	2	-0.5*Vref to 0.5*Vref

19.4.5 Temperature Measurement

The on-chip temperature sensor is used either to provide an absolute measurement of the device temperature or to detect changes in the ambient temperature (see Figure 104). The emitter and the collector/base of a PNP can be selected as 2 differential inputs of the ADC for temperature measurement. To do so, set the ADC_REG_ANA.TS_EN bit to 1'b1 and ADC_REG_SCN channels to 4'b1111.

Users have the option to measure the off-chip temperature by connecting an external diode (for example, 2N3906) onto GPIO input pins (ADC_CH[0] – ADC_CH[1]) with ADC_REG_ANA.TSEXT_SEL = 1'b1, or to monitor the on-chip temperature by using an embedded PNP with ADC_REG_ANA.TSEXT_SEL = 1'b0.

Figure 104:ADC Temperature Sensor Mode with External Diode



By selecting internal voltage reference 1.2V (Vref_12), 16-bit audio ADC accuracy and by measuring the internal temperature sensor, the temperature is calculated according to the following formula:

Tmeas (in C) = ADC_REG_RESULT.DATA[15:0]/TS_GAIN - TS_OFFSET

where:

- ADC_REG_RESULT.DATA is denoted as signed 16 bits
- TS_OFFSET and TS_GAIN are by default equal to:
  - For internal sensor: TS_OFFSET = 305; TS_GAIN = 6.295
  - For external sensor: TS_OFFSET = 282; TS_GAIN = 6.39

19.4.6 ADC Reference Voltage

ADC_REG_ANA.VREF_SEL is used to select the reference voltage. Change the reference voltage only when no conversion is running.

The positive reference voltage for analog-to-digital conversions is selectable as either the internal reference 1.2V (Vref_12), Vref_18, or external reference applied to the GPIO pin (GPIO_45). The external reference should not exceed 1.8V.

19.4.7 ADC Throughput and Resolution

When the ADC clock is 64 MHz, through programming ADC_REG_GENERAL.CLK_DIV_RATIO[5:0] and ADC_REG_ANA.RES_SEL[1:0], the ADC throughput and resolution is listed in ADC Conversion Time and Throughput Rate Lookup Table. Table 149 shows the Lookup table.

Table 149: ADC Conversion Time and Throughput Rate Lookup Table

CLK_DIV_RATIO[5:0]	RES_SEL[1:0]	64 MHz Main Clock		Significant Bit
		1-Shot Latency (µs)	Throughput Rate (ksps)	
N (divide-by-N, N=1 to 32)	00	$N*(0.5+T_{WARM})$	1000/(N*0.5)	12
	01	$N*(5.5+T_{WARM})$	1000/(N*5.5)	14
	10	$N*(17.5+T_{WARM})$	1000/(N*17.5)	16
	11	$N*(65.5+T_{WARM})$	1000/(N*65.5)	16
00000 (divide-by-1)	00	$0.5+T_{WARM}$	2000	12
	01	$5.5+T_{WARM}$	181.8	14
	10	$17.5+T_{WARM}$	57.1	16
	11	$65.5+T_{WARM}$	15.27	16

19.4.8 ADC Conversion Results

The digital conversion result is represented in 2's complement form (see Table 150, Table 151, Table 152, and Table 152). The digital conversion result is available in ADC_REG_RESULT.DATA[15:0] when ADC_REG_ISR.RDY is set to 1'b1.

Table 150: ADC Conversion Result Format (OSR[1:0]=2'b11 or =2'b10)

ΔV /Vref	Results	
	Binary	Hexidecimal
1	0111 1111 1111 1111	7FFF
0.5	0100 0000 0000 0000	4000
1/32768	0000 0000 0000 0001	0001
0	0000 0000 0000 0000	0000
-1/32768	1111 1111 1111 1111	FFFF
-0.5	1100 0000 0000 0000	C000
-1	1000 0000 0000 0000	8000

Table 151: ADC Conversion Result Format (OSR[1:0]=2'b01)

ΔV /Vref	Results	
	Binary	Hexidecimal
1	0001 1111 1111 1111	1FFF
0.5	0001 0000 0000 0000	1000
1/8192	0000 0000 0000 0001	0001
0	0000 0000 0000 0000	0000
-1/8192	1111 1111 1111 1111	FFFF
-0.5	1111 0000 0000 0000	F000
-1	1110 0000 0000 0000	E000

Table 152: ADC Conversion Result Format (OSR[1:0]=2'b00)

ΔV /Vref	Results	
	Binary	Hexidecimal
1	0000 0111 1111 1111	07FF
0.5	0000 0100 0000 0000	0400
1/2048	0000 0000 0000 0001	0001
0	0000 0000 0000 0000	0000
-1/2048	1111 1111 1111 1111	FFFF
-0.5	1111 1011 0000 0000	FC00
-1	1111 1000 0000 0000	F800

19.4.9 ADC Interrupts

The ADC outputs a single active high-level interrupt signal when any of the following exceptions is pending:

- Sequence conversion has completed and corresponding 16-bit final data in ADC_REG_RESULT.DATA[15:0] is ready for reading
- Overflow occurred in the offset calibration process
- Overflow occurred in the gain calibration process
- Source data negative side saturation occurred
- Source data positive side saturation occurred
- FIFO overrun occurred
- FIFO underrun occurred

ADC_REG_ISR, the interrupt status register, is read only. Every non-reserved bit in this register represents 1 exception. A bit read as 1 means the corresponding exception is pending; the masked exception is not captured in this register.

ADC_REG_IMR, the interrupt mask register, is readable and writable. This register is used to mask off unused exceptions. When a bit is set to 1, the corresponding exception is masked off and it does not trigger the interrupt signal. By default, all the non-reserved bits should be 1.

ADC_REG_IRSR, the interrupt raw status register, is read-only. All the pending exceptions are captured in this register regardless of the values in the interrupt mask register.

ADC_REG_ICR, the interrupt clear register, is write only. Write a 1 to a bit to clear the corresponding pending exception.

19.4.10 ADC Calibration

Sampling of internal connections VSSA or Vref_12 allows for self/system offset and gain calibration of the ADC to correct error due to process and temperature variations where absolute accuracy is important. This calibration must be done individually for each reference used and each ADC decimation rate.

Automatic calibration measurement is activated by the following steps (the sequence cannot be reversed):

1. Stop ADC conversion by setting ADC_REG_CMD.CONV_START=0.
2. Enable automatic calibration by setting ADC_REG_GENERAL.ADC_CAL_EN=1.
3. Start ADC conversion by setting ADC_REG_CMD.CONV_START=1.

After the automatic calibration, ADC_REG_GENERAL.ADC_CAL_EN is cleared by hardware. Measured data is stored in ADC_REG_OFFSET_CAL.OFFSET_CAL and ADC_REG_GAIN_CAL.GAIN_CAL. Setting ADC_REG_CONFIG.CAL_DATA_RST=1 resets the offset to 0 (ADC_REG_OFFSET_CAL = 0) and gain factor to 0 (ADC_REG_GAIN_CAL = 0).

Instead of using the automatic offset/gain calibration, the user can set the calibration coefficients by setting ADC_REG_CONFIG.CAL_DATA_SEL=1. Calibration data is then defined by entering data into ADC_REG_OFFSET_CAL.OFFSET_CAL_USR and ADC_REG_GAIN_CAL.GAIN_CAL_USR.

Table 153 shows the equations used to calculated the gain and offset correction values.

Table 153: Equations for Gain and Offset Correction

Calibration Type	Calibration Correction Value (INBUF_GAIN = 1)
Self Offset, input buffer disabled. Vref = Vref_12	$N_{(VSSA-VSSA)}$
Self Gain, input buffer disabled. Vref = Vref_12	1-$N_{(1.2-VSSA)}$ /0x7FFF
System Gain, input buffer disabled. Vref = external 1.25V	1-$N_{(1.2-VSSA)}$ /0x7AE1

Equation Notes:

- All N are 16-bit 2's complement numbers.
- $N_{(VSSA-VSSA)}$ is a differential sampling of 0 with input positive and negative sides connected to VSSA when offset calibration mode and it yields a 2's complement value close to 0.
- $N_{(1.2-VSSA)}$ is a sampling of Vref_12 and it yields a 2's complement value close to 0x7FFF in OSR[1:0] = 2'b11 mode.
- $N_{(1.2-VSSA)}$ is a sampling of Vref_12 with external accurate voltage reference (1.25V from ADC_CH[3]) and yields a 2's complement value close to 0x7AE1 in OSR[1:0] = 2'b11 mode.

19.4.11 DMA Request

With ADC_REG_DMAR.DMA_EN to 1'b1, the ADC sends out a DMA request every time the converted data in FIFO achieves the number of ADC_REG_DMAR.FIFO_THL. This request is automatically cleared when the corresponding result register is read and a DMA acknowledge signal sent from DMA has been received (optional).

19.4.12 Battery Monitor

The internal power supply monitor allows the battery voltage to be measured by programming ADC_REG_ANA.SINGLEDIFF to 1'b0 and ADC_REG_ANA.AMUX_SEL[3:0] to 4'b1000. This monitoring is achieved with a potential divider that reduces the voltage by a factor of 0.33 (nominal), allowing it to fall inside the ADC input range. After the battery voltage measurement is finished, ADC_REG_ANA.AMUX_SEL[3:0] should be assigned another value other than 4'b1000 to disable the resistor chain that performs the voltage reduction, thereby avoiding a continuous drain on the power supply.

19.4.13 External Trigger from GPT

It is also possible to trigger conversions from an event. Once the ADC is powered up by setting the ADC_REG_GENERAL.GLOBAL_EN bit to 1, the data conversion can be activated by external trigger source (GPT0 for ADC0) by writing 1 to the ADC_REG_CONFIG.TRIGGER_EN bit to enable the Event Trigger mode.

19.5 Register Description

See Appendix A.14.2, ADC Registers, on page 694 for a detailed description of the registers.

THIS PAGE INTENTIONALLY LEFT BLANK

20 Digital Analog Converter (DAC)

20.1 Overview

The 88MW300/302 integrates a register string-based DAC with true 10-bit resolution. It includes 2 channels. Each channel can output a single-ended signal or combine both channels to output a differential signal.

20.2 Features

- 10-bit resolution
- Throughput rate as fast as 2μs (500 kHz)
- Capable of directly driving a piezo speaker with 5 kΩ load
- Flexible waveform generator (sinusoidal, triangle, noise, etc.) at various frequency range
- Selectable output mode: single-ended or differential
- Internal or external reference voltage
- Interrupt generation and/or DMA request
- 3 selectable output ranges
- Supports event trigger from GPT or GPIO

20.3 Interface Signal Description

Table 154 shows the interface signals.

Table 154: DAC Interface Signals[1]

Pin Name	Type	Default Value	Source/ Destination	Description
EXT_VREF	A, I	n/a	GPIO to DAC	External voltage reference from GPIO_45
DACA	A, O	n/a	DAC to GPIO	DAC channel A output to GPIO_44
DACB	A, O	n/a	DAC to GPIO	DAC channel A output to GPIO_43

1. See Section 22.9, DAC Specifications, on page 337 for electrical specifications.

20.4 Functional Description

20.4.1 Configuration

The 88MW300/302 DAC can support 2 independent single-ended channels or 1 differential channel, as set by the DAC.BCTRL.B_WAVE register bits.

In single-ended mode, each channel output can be sent to the pad by setting DAC.xCTRL.x_IO_EN register bit (x can be A or B). Each channel can be powered on by the DAC.xCTRL.x_EN register bit (x can be A or B).

Table 155 shows the output voltage calculation.

Table 155: Output Voltage Calculation Formula

A_Range[1:0]	REF_SEL	Output
00	0	0.16+(0.64*input code/1023)
01/10	0	0.19+(1.01*input code/1023)
11	0	0.18+(1.42*input code/1023)
00	1	0.08*Vref_ext+(0.32*Vref_ext*input code/1023)
01/10	1	0.095*Vref_ext+(0.505*Vref_ext*input code/1023)
11	1	0.09*Vref_ext+(0.71*Vref_ext*input code/1023)

- DAC.ACTRL.A_RANGE impacts the output range of both channel A and B simultaneously
- DAC can be operated in differential mode by setting DAC.BCTRL.B_WAVE to 2'b11
- Each DAC channel can be powered on by setting the corresponding DAC.xCTRL.x_EN bit to 1 (x can be A or B)
- DAC conversion rate is selectable by setting the DAC.CLK.CLK_CTRL bits. See Section 3.5.6, AUPLL for Audio Clock and GAU Clock, on page 97 for AUPLL setting for 64 MHz OCLK.

Table 156 shows the clock divisor.

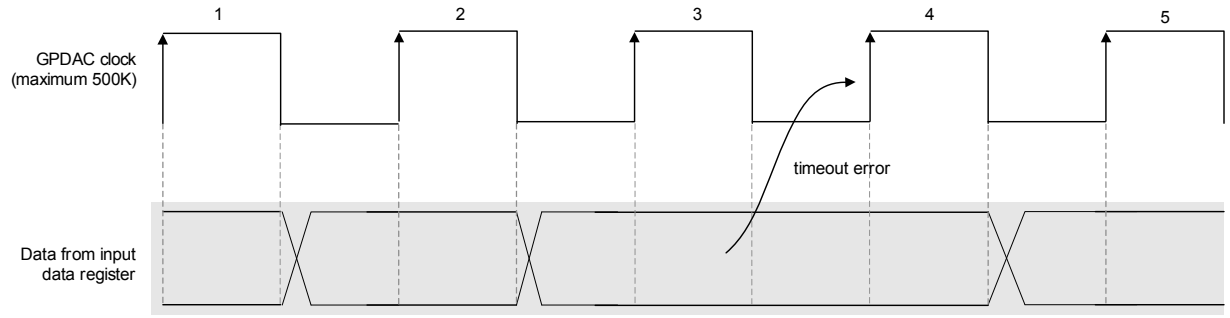
Table 156: Clock Divisor

gpdac_clk_ctrl (DAC.CLK.CLK_CTRL)	DAC Conversion Rate
00	62.5 kHz
01	125 kHz
10	250 kHz
11	500 kHz

20.4.2 Synchronous Mode

Each DAC channel can operate in synchronous mode by setting the DAC.xCTRL.x_MODE register bit (x can be A or B). In this mode, each DAC channel has a timing requirement for input data refresh speed. Figure 105 shows the timing.

Figure 105:Synchronous Mode



20.4.3 Asynchronous Mode

In this mode, the DAC works in passive mode, and does not have a time-sequence requirement for input data refreshing.

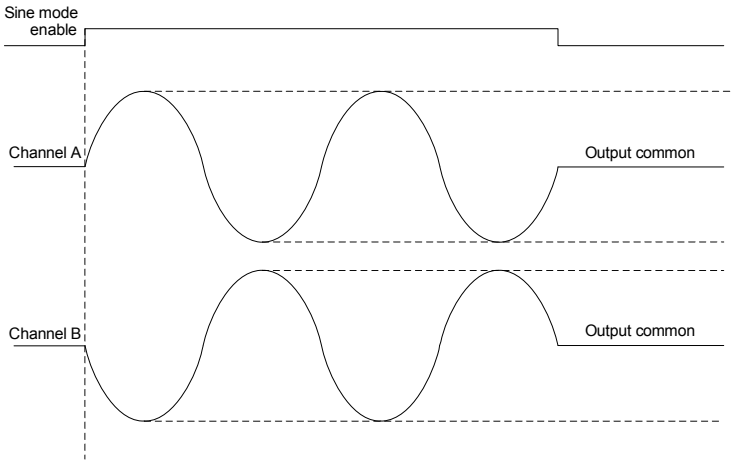
20.4.4 Sinusoidal Waveform Generation

This function is enabled by setting DAC.ACTRL.A_WAVE[1:0] register bits to 2'b10. The amplitude of the sine signal is controlled by the DAC.ACTRL.A_RANGE[1:0] register bits. Each period, starting at 0 degree consists of 16 samples and the frequency is given by the equation:

$$f_{sine} = f_{clk} / 16$$

The sine wave is output on Channel A. In differential mode, the sine wave is output on both channels (if 2 channels have been enabled), but inverted. See Figure 106.

Figure 106:Sinusoidal Waveform Generation



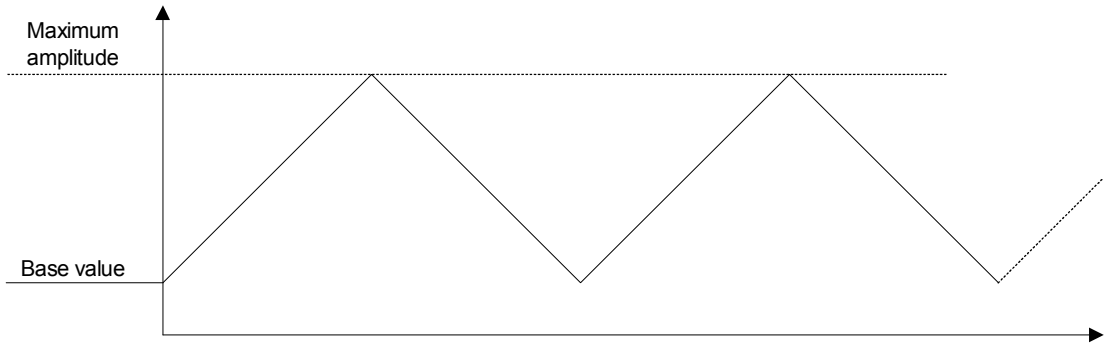
20.4.5 Triangle Waveform Generation

20.4.5.1 Up and Down Mode

- Amplitude of triangle wave configured through the DAC.ACTRL.A_RANGE[1:0] register bits
- Triangle counter increases 3 clock cycles after each trigger event
- Triangle counter increases while less than maximum amplitude set by DAC.ACTRL.A_TRIA_MAMP_SEL[3:0] register field
- Increment step is defined by the DAC.ACTRL.A_TRIA_STEP_SEL[1:0] register
- Once configured amplitude is reached, counter is decremented down to base value defined by the DAC.ADATA register

Figure 107 shows the timing.

Figure 107:Full Triangle Generation Mode

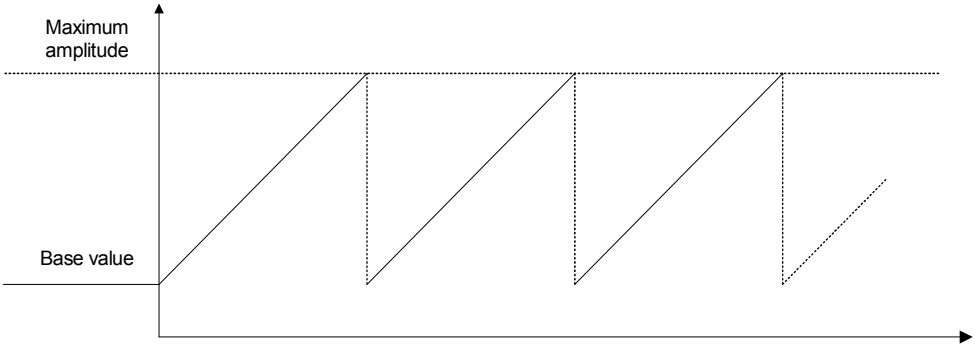


20.4.5.2 Up Mode

This mode is set by the DAC.ACTRL.TRIA_HALF bit in the DAC.ACTRL register. The remaining control information with respect to setting the amplitude, maximum amplitude, increment step are configured using the DAC.ACTRL.A_RANGE[1:0], DAC.ACTRL.A_ TRIA_MAMP_SEL[3:0] and DAC.ACTRL. A_TRIA_STEP_SEL[1:0] register fields. See Section 20.4.5.1, Up and Down Mode for a detailed description of how the register fields are used. The difference from the Up and Down mode is that once the configured amplitude is reached, the counter is directly down to the base value.

Figure 108 shows the timing.

Figure 108:Half Triangle Generation Mode



20.4.6 Noise Generation

The DAC can generate pseudo noise.

20.4.7 DMA Request

Each DAC channel supports DMA data transfer. A DAC DMA request is generated each time when previous data conversion is complete and new data is requested to be loaded to DAC.xDATA.x_DATA[9:0] while the DAC.xCTRL.x_DEN register field (x can be A or B) is set to 1. If the DAC.xCTRL.x_DEN register field is set for both channel A and B, 2 DMA requests are generated.

20.4.8 Event Trigger from GPT or GPIO

Events from GPT or GPIO can trigger the reload of new data from DAC.xData.x_DATA (x can be A or B) to DAC for conversion. The DAC event trigger mode is activated by writing DAC.xCTRL.x_EN bit to 1 and DAC.xCTRL.x_TRIG_EN bit to 1 (x can be A or B).

Each DAC channel accepts up to 4 trigger sources: GPT2, GPT3, GPIO_41, and GPIO_40. DAC.xCTRL.x_TRIG_SEL[1:0] defines the appropriate trigger.

GPT match interrupt can generate the trigger event.

In addition, the transition edge of selected external GPIO source can trigger the reload of new data. Rising edge, falling edge, or both edges can be selected by DAC.xCTRL.x_TRIG_TPY[1:0].

When there is no event occurred, the DAC continuously converts previous 10-bit data and hold the analog conversion result at the output.

When a trigger event is generated, a new 10- bit data block is loaded in the DAC and a new analog conversion result is presented at the output.

20.5 Registers Description

See Appendix A.15.2, DAC Registers, on page 717 for a detailed description of the registers.

THIS PAGE INTENTIONALLY LEFT BLANK

21 Analog Comparator (ACOMP)

21.1 Overview

The 88MW300/302 has 2 analog identical comparators, ACOMP0 and ACOMP1, which are designed to have true rail-to-rail inputs and operate over the full voltage range of the power supply VDDIO_3. The comparator outputs are latched and can be used as interrupts.

21.1.1 Features

- 8 selectable external positive inputs
- 8 selectable external negative inputs
- 2 selectable internal positive inputs
  - DACA output
  - DACB output
- 5 selectable internal negative inputs
  - DACA output
  - DACB output
  - VDDIO_3, VDDIO_3*0.75, VDDIO_3*0.5, VDDIO_3*0.25
  - Internal reference 1.2V (Vref_12)
  - VSSA
- Selectable positive and negative hysteresis between 0 and 70 mV, with 10 mV step
- Selectable response time as fast as 130 ns
- Interrupt generation on selectable edges (rising edge and/or falling edge) or levels.
- Extremely low-power mode
- Configurable output when inactive
- Comparator output on GPIOs through alternate functionality, output inversion available

21.2 Interface Signal Description

The external interface signals of the ACOMP0/1 include power supply and analog inputs.

Table 157 shows the interface signals.

Table 157: ACOMP Module Interface Signals[1]

Pin Name	Type	Default Value	Source/ Destination	Description
ACOMP0_IN [7:0]	A, I	n/a	GPIO to ACOMP	External Analog Inputs from GPIO ACOMP_CH[0]: GPIO_42 ACOMP_CH[1]: GPIO_43 ACOMP_CH[2]: GPIO_44 ACOMP_CH[3]: GPIO_45 ACOMP_CH[4]: GPIO_46 ACOMP_CH[5]: GPIO_47 ACOMP_CH[6]: GPIO_48 ACOMP_CH[7]: GPIO_49
ACOMP0_GPIO_OUT	D, O	n/a	ACOMP to GPIO	ACOMP0 synchronized or asynchronized comparison output to GPIO_40
ACOMP1_GPIO_OUT	D, O	n/a	ACOMP to GPIO	ACOMP1 synchronized or asynchronized comparison output to GPIO_40
ACOMP0_EDGE_PULSE	D, O	n/a	ACOMP to GPIO	ACOMP0 edge detector output to GPIO_41
ACOMP1_EDGE_PULSE	D, O	n/a	ACOMP to GPIO	ACOMP1 edge detector output to GPIO_41

1. See Section 22.10, ACOMP Specifications, on page 339 for electrical specifications.

21.3 Functional Description

21.3.1 ACOMP0/1 Control Signals

The ACOMP.CTRL[x].POS_SEL[3:0] field and the ACOMP.CTRL[x].NEG_SEL[3:0] field select which signals are connected to the 2 inputs of the comparator.

21.3.1.1 Warm-up Time

When the ACOMP.CTRL[x].EN bit is set from low to high, ACOMPx is turned on and compares the 2 analog inputs. The warm-up time of the comparator after turn-on is programmable in the ACOMP.CTRL[x].WARMTIME[1:0] field. When it is warmed up, the ACOMP.STATUS[x].ACT bit is set high.

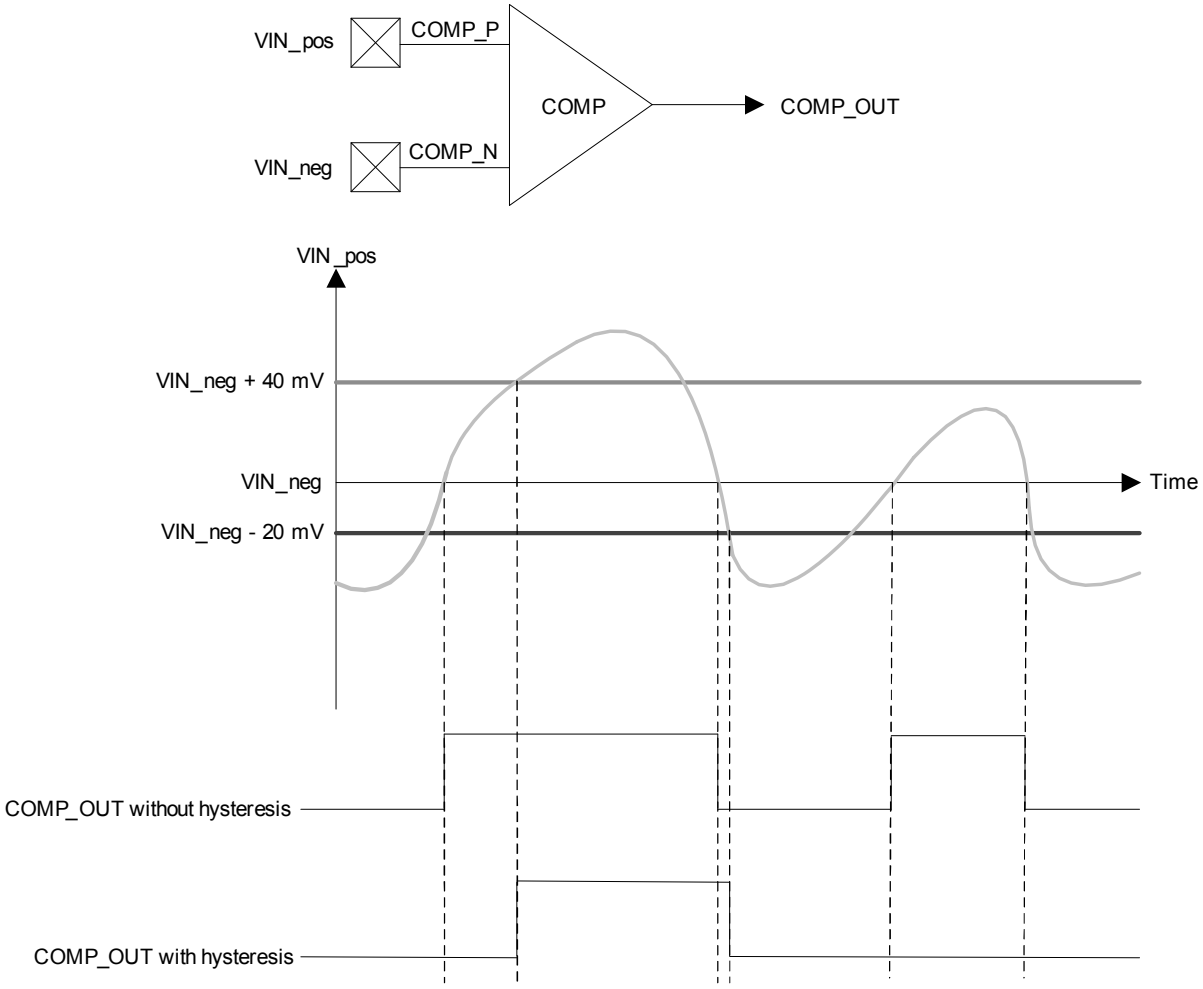
21.3.1.2 Response Time

When the voltage of input signals changes and triggers a polarity flip at the comparator output, the delay from input to output is the response time of the comparator. The response time is also programmable through the ACOMP.CTRL[x].BIAS_PROG[1:0] field.

21.3.1.3 Hysteresis

The programmable hysteresis in ACOMP0/1 can be used to filter input fluctuations due to noise, and only changes that are big enough to reach the hysteresis threshold trigger an output change, as shown in Figure 109.

Figure 109:Comparator Hysteresis



The hysteresis voltage levels for the positive input and the negative input are set in the ACOMP.CTRL[x].HYST_SELP[2:0] and ACOMP.CTRL[x].HYST_SELN[2:0] field.

21.3.2 Comparator Output

The outputs from ACOMP0/1 are available in ACOMP.STATUS[x].OUT, or as alternate functions to the GPIO pins. Set the ACOMP.ROUTE[x].PE bit to 1 to enable output to pin.

21.3.2.1 Asynchronous Comparison Output at Register

When comparator is enabled by ACOMP.CTRL[x].EN, real time comparator output is available in ACOMP.STATUS[x].OUT.

When comparator is disabled, comparator output in ACOMP.STATUS[x].OUT can be set by ACOMP.CTRL[x].INACT_VAL.

21.3.2.2 Synchronous/Asynchronous Comparison Output at GPIO

The comparator output at GPIO can be programmed to be synchronized with the main clock of the comparator, or asynchronized by setting ACOMP.ROUTE[x].OUTSEL.

When powered down, the output values can be set through register bit ACOMP.CTRL[x].INACT_ VAL.

21.3.2.3 Comparison Output Inversion

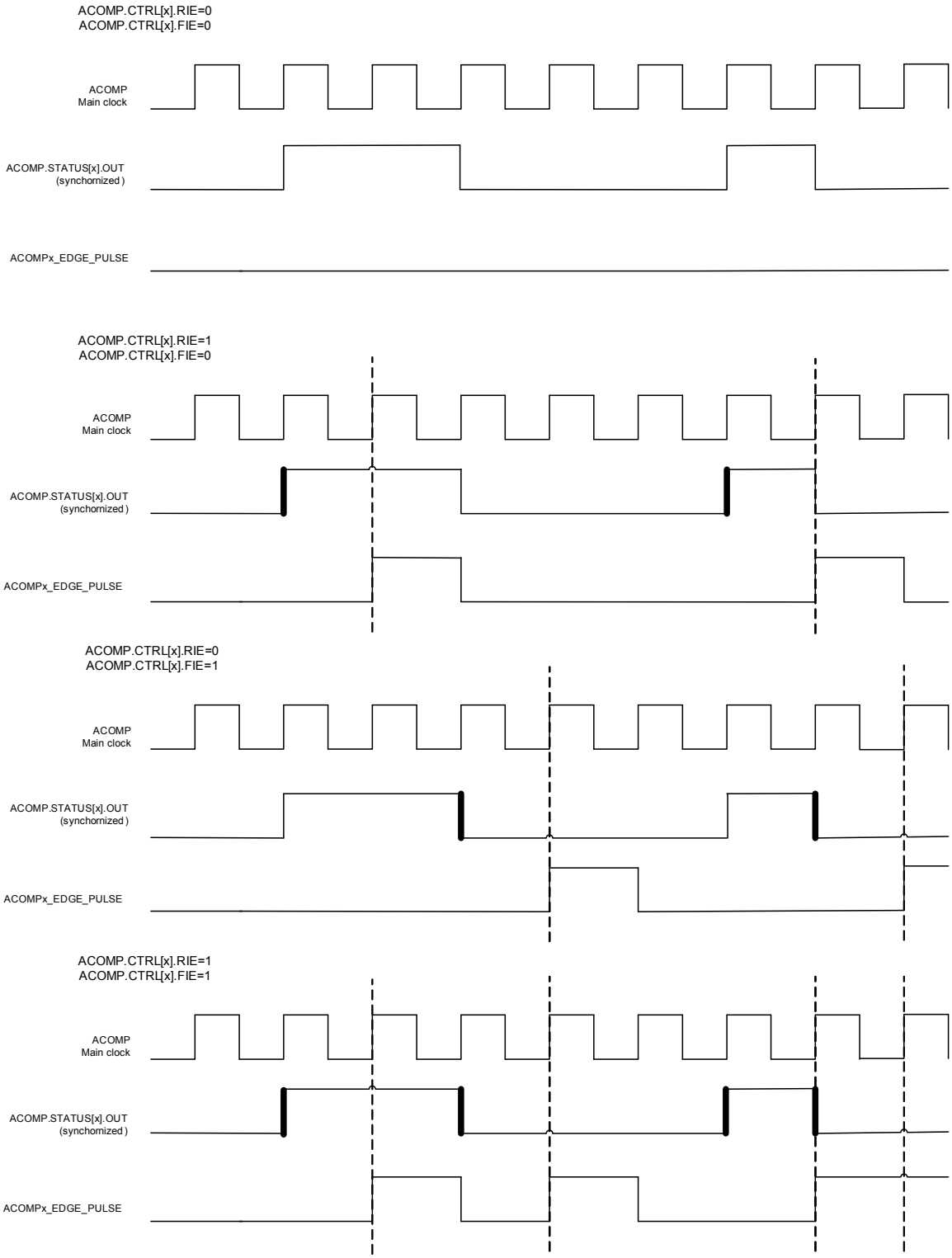
The comparison output to GPIO can be inverted by the ACOMP.CTRL[x].GPIOINV bit.

21.3.3 Comparator Output Edge Detection

The comparator output edge detection can be enabled by selecting ACOMP.CTRL[x]. RIE (rising edge) or ACOMP.CTRL[x]. FIE (falling edge). The edge detection results are routed to GPIOs.

Figure 110, Comparator Output Edge Detection, on page 311 shows the timing.

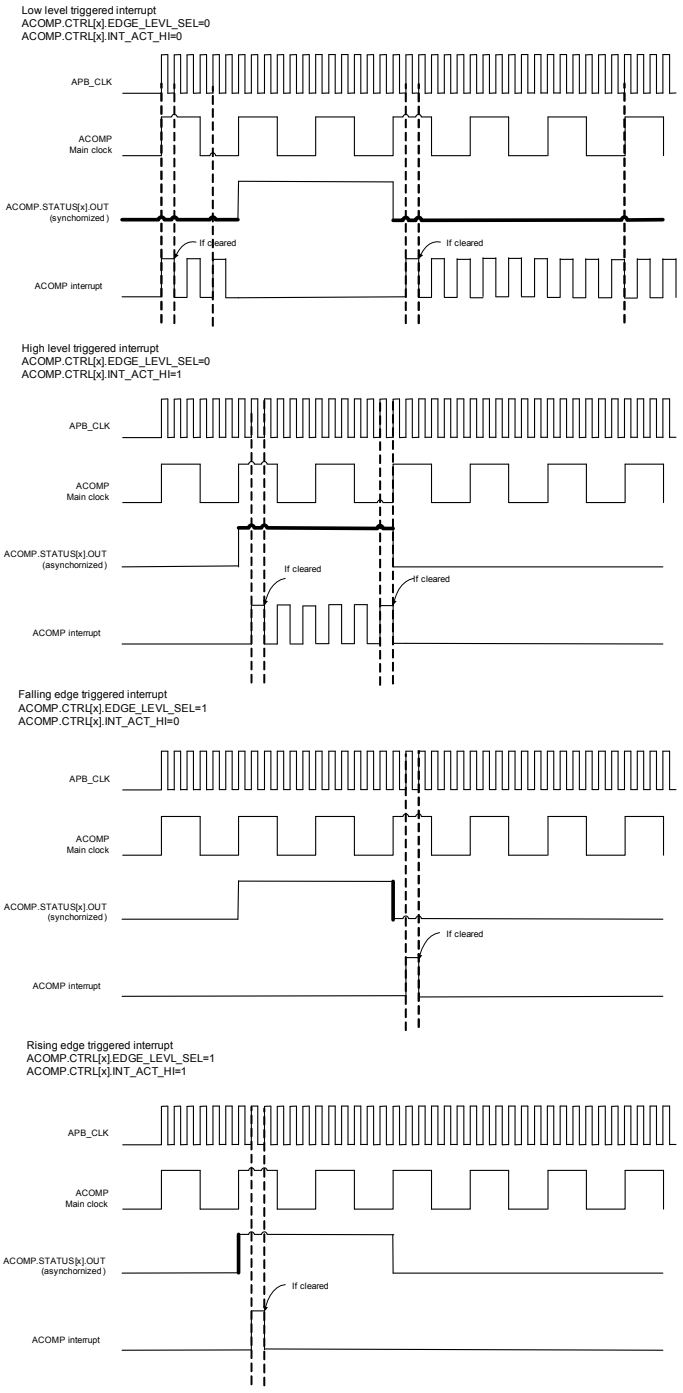
Figure 110:Comparator Output Edge Detection



21.3.4 Interrupt

An interrupt is generated upon detection of level or edge changes of ACOMP0/1 comparison results. Interrupt trigger type and active mode can be selected by ACOMP.CTRL[x].EDGE_LEVL_SEL and ACOMP.CTRL[x].INT_ACT_HI. Figure 111 shows the timing.

Figure 111:Interrupt



21.4 Register Description

See Appendix A.16, ACOMP Address Block, on page 729 for a detailed description of the registers.

THIS PAGE INTENTIONALLY LEFT BLANK

22 Electrical Specifications

22.1 Absolute Maximum Ratings

The absolute maximum ratings define limitations for electrical and thermal stresses. These limits prevent permanent damage to the device. These ratings are not operating ranges. Operation at absolute maximum ratings is not guaranteed.

Table 158: Absolute Maximum Ratings

Symbol	Parameter	Min	Typ	Max	Units
VDD11	Power supply voltage with respect to VSS	--	1.1	1.26	V
VDDIO_0 VDDIO_3	Power supply voltage with respect to VSS VDDIO_0, 3 must be powered on at the same time or after VBAT_IN is powered on and powered off before or at the same time VBAT_IN is powered off.	--	1.8	1.98	V
		--	2.5	2.75	V
		--	3.3	3.63	V
VDDIO_1 VDDIO_2 VDDIO_AON	Power supply voltage with respect to VSS VDDIO_1, 2, AON must be powered on at the same time or after VBAT_IN is powered on and powered off before or at the same time VBAT_IN is powered off.	--	1.8	2.16	V
		--	2.5	3.0	V
		--	3.3	4.0	V
GPIO V_{IL}	GPIO input low voltage	-0.4	--	--	V
GPIO V_{IH}	GPIO input high voltage	--	--	$VDDIO_{group}$[1]+0.4	V
VTR_VDD33	Power supply voltage with respect to VSS	--	3.3	4.0	V
USB_AVDD33	Power supply voltage with respect to VSS	--	3.3	4.0	V
USB_VBUS	Power supply voltage with respect to VSS	--	5	5.5	V
AVDD33	Power supply voltage with respect to VSS	--	3.3	4.0	V
AVDD18	Power supply voltage with respect to VSS	--	1.8	1.98	V
VBAT_IN	Power supply voltage with respect to VSS	--	3.3	3.63	V
BUCK18_VBAT_IN	Power supply voltage with respect to VSS	--	3.3	4.3	V
$T_{STORAGE}$	Storage temperature	-55	--	+125	°C

1. $VDDIO_{group}$ is the VDDIO power for the individual GPIO pin.

22.2 Recommended Operating Conditions

Table 159: Recommended Operating Conditions

Symbol	Parameter	Condition	Min	Typ	Max	Units
VDD11	1.1V core power supply	--	--	1.1	1.21	V
VDDIO_0 VDDIO_1 VDDIO_2 VDDIO_3[1] VDDIO_AON	1.8V digital I/O power supply	--	1.62	1.8	1.98	V
	2.5V digital I/O power supply	--	2.25	2.5	2.75	V
	3.3V digital I/O power supply	--	2.97	3.3	3.63	V
VTR_VDD33	3.3V OTP analog power supply or floating for Read OTP only operation	--	2.97	3.3	3.63	V
USB_AVDD33	3.3V USB analog power supply	--	2.97	3.3	3.63	V
USB_VBUS	5V analog power supply—high-power port 5 unit loads. 1 unit load = 100 mA.	--	4.75	5.0	5.25	V
	5V analog power supply—low-power port 1 unit load only.	--	4.40	5.0	5.25	V
AVDD33	3.3V analog power supply	--	2.97	3.3	3.63	V
AVDD18	1.8V analog power supply	--	1.71	1.8	1.89	V
VBAT_IN	LDO18 power supply	--	1.84	3.3	3.6	V
BUCK18_ VBAT_IN	BUCK18 power supply	--	2.4	3.3	4.3	V
T_A	Ambient operating temperature	Extended	-30	--	85	°C
		Industrial	-40	--	85	°C
T_J	Maximum junction temperature	--	--	--	125	°C

1. When the VDDIO_3 domain pad is used as GAU, in a typical 1.8V condition, the minimum is -5% (1.71V).

22.3 Digital Pad Ratings

22.3.1 I/O Static Ratings

Table 160: I/O Static Ratings, 1.8V VDDIO

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{IL}	Input low voltage	--	-0.4	--	VDDIO*30%	V
V_{IH}	Input high voltage	--	VDDIO*70%	--	VDDIO+0.4	V
V_{HYS}	Input hysteresis	--	150	--	--	mV
I_{OL} @0.4V	Output drive low	V(PAD) = 0.4V	4	6	12[1 2 3]	mA
I_{OH}@VDDIO-0.5V	Output drive high	V(PAD) = VDDIO - 0.5V	3	4.5	12[1 2 3]	mA
Input capacitance	--	--	--	--	5	pF
Input leakage 1	--	VDDIO is ON, 0<V(PAD)<VDDIO	--	--	5	μA

1. . Maximum current is based on best case conditions. Not all parts can achieve this.
2. The absolute maximum DC current is 20 mA for IOL / IOH, per pad. The user should not exceed this.
3. Each VDDIO_0/_1/_2/_3/_AON supply domain can only have maximum absolute DC current of 40 mA for driving low and maximum absolute DC current 40 mA for driving high.

Table 161: I/O Static Ratings, 2.5V VDDIO

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{IL}	Input low voltage	--	-0.4	--	VDDIO*30%	V
V_{IH}	Input high voltage	--	VDDIO*70%	--	VDDIO+0.4	V
V_{HYS}	Input hysteresis	--	150	--	--	mV
I_{OL} @0.4V	Output drive low	V(PAD) = 0.4V	4	6	12[1 2 3]	mA
I_{OH}@VDDIO-0.5V	Output drive high	V(PAD) = VDDIO - 0.5V	3	4.5	12[1 2 3]	mA
Input capacitance	--	--	--	--	5	pF
Input leakage 1	--	VDDIO is ON, 0<V(PAD)<VDDIO	--	--	5	µA

1. . Maximum current is based on best case conditions. Not all parts can achieve this.
2. The absolute maximum DC current is 20 mA for IOL / IOH, per pad. The user should not exceed this.
3. Each VDDIO_0/_1/_2/_3/_AON supply domain can only have maximum absolute DC current of 40 mA for driving low and maximum absolute DC current 40 mA for driving high.

Table 162: I/O Static Ratings, 3.3V VDDIO

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{IL}	Input low voltage	--	-0.4	--	VDDIO*30%	V
V_{IH}	Input high voltage	--	VDDIO*70%	--	VDDIO+0.4	V
V_{HYS}	Input hysteresis	--	150	--	--	mV
I_{OL} @0.4V	Output drive low	V(PAD) = 0.4V	4	6	12[1 2 3]	mA
I_{OH}@VDDIO-0.5V	Output drive high	V(PAD) = VDDIO - 0.5V	3	4.5	12[1 2 3]	mA
Input capacitance	--	--	--	--	5	pF
Input leakage 1	--	VDDIO is ON, 0<V(PAD)<VDDIO	--	--	5	µA

1. . Maximum current is based on best case conditions. Not all parts can achieve this.
2. The absolute maximum DC current is 20 mA for IOL / IOH, per pad. The user should not exceed this.
3. Each VDDIO_0/_1/_2/_3/_AON supply domain can only have maximum absolute DC current of 40 mA for driving low and maximum absolute DC current 40 mA for driving high.

22.3.2 Current Consumption

22.3.2.1 WLAN Subsystem

Table 163: WLAN Tx/Rx Current Consumption

Parameter	3.3V (Typ)	1.8V (Typ)	1.1V (Typ)	Units
Transmit				
Tx_802.11b 11 Mbps 20 MHz @4 dBm	59.0	89.7	27.6	mA
Tx_802.11b 11 Mbps 20 MHz @18 dBm	183.3	91.2	28.2	mA
Tx_802.11g 54 Mbps 20 MHz @4 dBm	61.1	89.7	29.5	mA
Tx_802.11g 54 Mbps 20 MHz @14 dBm	150.9	93.9	29.5	mA
Tx_802.11n MCS7 20 MHz @4 dBm	60.2	75.3	29.7	mA
Tx_802.11n MCS7 20 MHz @14 dBm	154.0	93.2	29.8	mA
Tx_802.11n MCS7 20 MHz @4 dBm AMPDU on	60.7	87.6	29.4	mA
Tx_802.11n MCS7 20 MHz @14 dBm AMPDU on	154.7	93.8	29.9	mA
Receive				
Rx_802.11b 11 Mbps 20 MHz	0.0	42.5	29.3	mA
Rx_802.11g 54 Mbps 20 MHz	0.0	43.8	33.4	mA
Rx_802.11n MCS7 20 MHz	0.0	45.2	35.3	mA
NOTE: The above current measurement is at room temperature condition. • 1.1V current to VDD11 pins. • 1.8V current to AVDD18 pins. • 3.3V current to AVDD33 pin.				

Table 164: WLAN Estimated Tx PA Power vs. Current Consumption

Parameter	Condition	Min	Typ	Max	Units
18.5 dBm	3.3V	--	--	270	mA
18 dBm		--	--	250	
16 dBm		--	--	220	
15 dBm		--	--	200	
14 dBm		--	--	190	
10 dBm		--	--	150	
8 dBm		--	--	135	
5 dBm		--	--	120	
4 dBm		--	--	115	
0 dBm		--	--	100	
-5 dBm		--	--	85	
-10 dBm		--	--	75	
-20 dBm		--	--	70	

22.3.2.2 MCI Subsystem

Table 165: MCI VBAT Consumption

Symbol	Parameter	Condition	Min	Typ	Max	Units
PM0	Active (Power Mode = 00)	PLL = on All peripherals clock = off	--	15.8	--	mA
		PLL = on Typical peripherals clock = on	--	25	--	mA
		PLL = on All peripherals clock = on	--	28.3	--	mA
		RC32M = on All peripherals clock = off	--	3.62	--	mA
		RC32M = on Typical peripherals clock = on	--	4.77	--	mA
		RC32M = on All peripherals clock = on	--	5.3	--	mA
PM1	Idle (Power Mode = 00)	RC32M = on All peripherals clock = off	--	3.59	--	mA
		RC32M = on Typical peripherals clock = on	--	4.74	--	mA
		RC32M = on All peripherals clock = on	--	5.27	--	mA
PM2	Standby (Power Mode = 01)	RC32M = on All peripherals clock = off, except RC32K	--	282	--	µA
PM3	Sleep (Power Mode = 10)	RC32K = on SRAM = retention mode RTC = on ULP comparator = off Brown-out detection = off	12	70	91	µA
PM4	Deep sleep (Power Mode = 11)	AON domain = on SRAM = retention mode RTC = on ULP comparator = off Brown-out detection = off	9.2	45	58	µA
NOTE: The above current measurement is at room temperature condition.						

22.4 Regulators

Table 166: LDO11 Specifications

Symbol	Parameter	Condition	Min	Typ	Max	Units
LDO11_VIN	Input voltage	--	1.62	1.8	1.98	V
LDO11_VOUT	Output voltage	--	1.045	1.1	1.155	V
$LDO11\_VOUT_{ss}$	Control step size	--	--	--	25	mV
$LDO11_{max}$	Output load current	--	--	--	50	mA
$LDO11_{icons}$	Current consumption (sleep mode current)	--	--	10	--	µA
C_{load}	Load capacitance	--	0.7	1	10	µF

Table 167: BUCK18 Specifications

Symbol	Parameter	Condition	Min	Typ	Max	Units
BUCK18_VBAT_IN	Input voltage	--	2.4	3.3	4.3	V
BUCK18_VOUT	Output voltage	1.8V	1.71	1.8	1.89	V
$BUCK18\_VOUT_{ss}$	Control step size	--	--	20	--	mV
$BUCK18_{max}$	Output load current	--	--	--	180	mA
$BUCK18_{icons}$	Current consumption (sleep mode current)	--	--	10	--	µA
C_{load}	Load capacitance	low-ESR capacitor	--	10	--	µF
L_{load}	Load inductance	--	--	2.2	--	µH

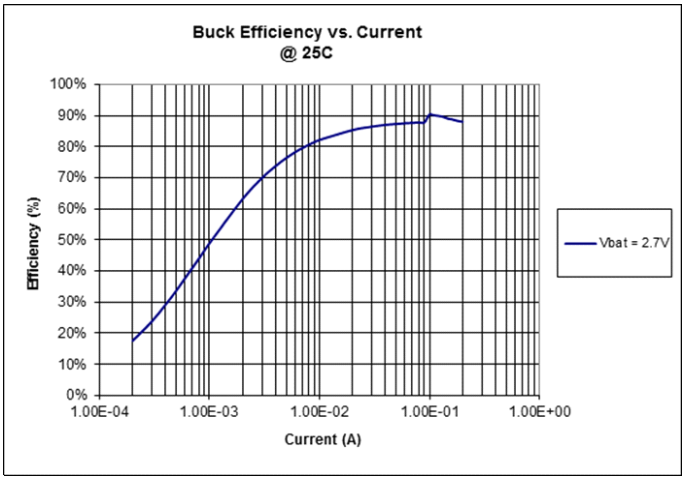
Table 168: Efficiency vs. BUCK18_VBAT_IN @ 147 mA Output[1]

VBAT (V)	IBAT (mA)	VOUT (V)	Efficiency (%)
4.8	66	1.762	81.8
4.5	70	1.762	82.3
4.2	74	1.761	83.3
3.9	79	1.761	84.0
3.6	85	1.761	84.7
3.3	92	1.760	85.3
3.0	100	1.760	86.3
2.7	110	1.759	87.2
2.4	122	1.758	88.4

1. This information is provided as a reference to typical characteristics and should be used for estimation purposes only.

Figure 112 shows the efficiency vs. current for BUCK18_VBAT_IN = 2.7V.

Figure 112:BUCK18_VBAT_IN = 2.7V Efficiency vs. Current[1]



1. This information is provided as a reference to typical characteristics and should be used for estimation purposes only.

NOTE: LDO11[1] Specifications

Symbol	Parameter	Condition	Min	Typ	Max	Units
LDO11_VOUT	Output voltage	VBAT = 3.3V	1.05	1.1	1.20	V
$LDO11_{max}$	Output load current	--	--	--	100	mA
C_{load}	Load capacitance	--	1	--	--	µF

1. Recommend X7R or X5R ceramic capacitors with low ESR.

NOTE: LDO18 Specifications

Symbol	Parameter	Condition	Min	Typ	Max	Units
LDO18_VIN (VBAT_IN)	Input voltage	--	1.84	3.3	3.6	V
LDO18_VOUT	Output voltage	VBAT = 3.3V	1.72	1.80	1.88	V
$LDO18_{max}$	Output load current	--	--	--	150	mA
C_{load}	Load capacitance	--	1	--	--	µF

22.5 Package Thermal Conditions

22.5.1 68-Pin QFN

Table 169: Thermal Conditions—68-pin QFN

Symbol	Parameter	Condition	Typ	Units
θ_{JA}	Thermal resistance[1] Junction to ambient of package. $\theta_{JA} = (T_J - T_A)/ P$ P = total power dissipation	JEDEC 3 x 4.5 inch, 4-layer PCB no air flow	24.2	°C/W
		JEDEC 3 x 4.5 inch, 4-layer PCB 1 meter/sec air flow	21.5	°C/W
		JEDEC 3 x 4.5 inch, 4-layer PCB 2 meter/sec air flow	20.5	°C/W
		JEDEC 3 x 4.5 inch, 4-layer PCB 3 meter/sec air flow	19.9	°C/W
ψ_{JT}	Thermal characteristic parameter[1] Junction to top-center of package. $\psi_{JT} = (T_J - T_{TOP})/P$ T_{TOP} = temperature on top-center of package	JEDEC 3 x 4.5 inch, 4-layer PCB no air flow	0.28	°C/W
ψ_{JB}	Thermal characteristic parameter[1] Junction to bottom surface, center of PCB. $\psi_{JT} = (T_J - T_B)/P$ T_B = surface temperature of PCB	JEDEC 3 x 4.5 inch, 4-layer PCB no air flow	14.8	°C/W
θ_{JC}	Thermal resistance[1] Junction to case of the package $\theta_{JC} = (T_J - T_C)/ P_{Top}$ T_C = temperature on top-center of package P_{TOP} = power dissipation from top of package	JEDEC 3 x 4.5 inch, 4-layer PCB no air flow	9.7	°C/W
θ_{JB}	Thermal resistance[1] Junction to board of package $\theta_{JB} = (T_J - T_B)/ P_{bottom}$ P_{BOTTOM} = power dissipation from bottom of package to PCB surface	JEDEC 3 x 4.5 inch, 4-layer PCB no air flow	15.0	°C/W

1. Refer to white paper on AN-63 Thermal Calculations for more information.

22.5.2 88-Pin QFN

Table 170: Thermal Conditions—88-pin QFN

Symbol	Parameter	Condition	Typ	Units
θ_{JA}	Thermal resistance[1] Junction to ambient of package. $\theta_{JA} = (T_J - T_A)/ P$ P = total power dissipation	JEDEC 3 x 4.5 inch, 4-layer PCB no air flow	25.8	°C/W
		JEDEC 3 x 4.5 inch, 4-layer PCB 1 meter/sec air flow	22.9	°C/W
		JEDEC 3 x 4.5 inch, 4-layer PCB 2 meter/sec air flow	21.8	°C/W
		JEDEC 3 x 4.5 inch, 4-layer PCB 3 meter/sec air flow	21.2	°C/W
ψ_{JT}	Thermal characteristic parameter[1] Junction to top-center of package. $\psi_{JT} = (T_J - T_{TOP})/P$ T_{TOP} = temperature on top-center of package	JEDEC 3 x 4.5 inch, 4-layer PCB no air flow	0.3	°C/W
ψ_{JB}	Thermal characteristic parameter[1] Junction to bottom surface, center of PCB. $\psi_{JT} = (T_J - T_B)/P$ T_B = surface temperature of PCB	JEDEC 3 x 4.5 inch, 4-layer PCB no air flow	15.0	°C/W
θ_{JC}	Thermal resistance[1] Junction to case of the package $\theta_{JC} = (T_J - T_C)/ P_{Top}$ T_C = temperature on top-center of package P_{TOP} = power dissipation from top of package	JEDEC 3 x 4.5 inch, 4-layer PCB no air flow	10.0	°C/W
θ_{JB}	Thermal resistance[1] Junction to board of package $\theta_{JB} = (T_J - T_B)/ P_{bottom}$ P_{BOTTOM} = power dissipation from bottom of package to PCB surface	JEDEC 3 x 4.5 inch, 4-layer PCB no air flow	15.2	°C/W

1. Refer to white paper on AN-63 Thermal Calculations for more information.

22.6 Clock Specifications

22.6.1 RC32K Specifications

Table 171: RC32K[1] Specifications

Parameter	Condition	Min	Typ	Max	Units
Frequency before calibration	--	18.6	31.8	39.8	kHz
Startup time	--	--	0.9	--	ms
After-calibration frequency accuracy	Use 32.768 kHz crystal as reference clock	32.3	32.7	33.1	kHz
Temperature tolerance	--	--	65	--	ppm/C
Duty cycle	--	40	50	60	%

1. -40 to 85^{o}C, VBAT = 3.6V with default setting unless otherwise specified.

22.6.2 Single-Ended Clock Input Modes Specifications

Table 172: CMOS Mode[1] Specifications

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{IH}	Input high voltage	--	AVDD18 - 0.5	AVDD18	1.98	V
V_{IL}	Input low voltage	--	0	0	0.4	V

1. Typical input capacitance is approximately 2 pF and input resistance is >20 kΩ.

Table 173: Phase Noise—2.4 GHz Operation Specifications

Parameter	Test Conditions	Min	Typ	Max	Units
Fref = 38.4 MHz	Offset = 1 kHz	--	--	-123	dBc/Hz
	Offset = 10 kHz	--	--	-134	dBc/Hz
	Offset = 100 kHz	--	--	-140	dBc/Hz
	Offset > 1 MHz	--	--	-140	dBc/Hz

22.6.3 Crystal Specifications

22.6.3.1 38.4 MHz

Table 174: Crystal Specifications (38.4 MHz)

Parameter	Condition	Typical	Units
Fundamental Frequencies	--	38.4	MHz
Frequency Tolerance	Over operating temperature	< ±10	ppm
	Over process at 25ºC	< ±10	ppm
SMD and AT Cut Height	--	<1.2	mm
Load Capacitor	--	10	pF
Maximum Series Resistance	--	60	Ω
Resonance Mode	--	A1, Fundamental	--

22.6.3.2 32.768 kHz

Table 175: Crystal Specifications (32.768 kHz)

Parameter	Condition	Min	Typ	Max	Units
Crystal frequency	--	--	32.768	--	kHz
Frequency accuracy tolerance	--	-40	--	40	ppm
Startup time	--		--	600	ms
Duty cycle tolerance	--	--	50	--	%
Crystal load capacitance	--	--	12.5	--	pF
Crystal shunt capacitance	--	--	--	7	pF
Equivalent Series Resistance (ESR)	--	--	--	100	kΩ

22.7 Power and Brown-Out Detection Specifications

22.7.1 Power-On Reset (POR) Specifications

Table 176: POR Specifications

Symbol	Parameter	Condition	Min	Typ	Max	Units
--	Power-on reset threshold (rising edge)	--	--	1.25	--	V

22.7.2 Brown-Out Detection (BOD) Specifications

Table 177: VBAT BOD Timing Data

Symbol	Parameter	Min	Typ	Max	Units
Vtrig (BOD) – Vbat Brown-out trigger level	BRNTRIG_VBAT_CNTL = 0x0	--	1.6	--	V
	BRNTRIG_VBAT_CNTL = 0x1	--	1.7	--	V
	BRNTRIG_VBAT_CNTL = 0x2	--	1.8	--	V
	BRNTRIG_VBAT_CNTL = 0x3	--	2.7	--	V
	BRNTRIG_VBAT_CNTL = 0x4 (default)	--	2.6	--	V
	BRNTRIG_VBAT_CNTL = 0x5	--	2.8	--	V
	BRNTRIG_VBAT_CNTL = 0x6	--	2.9	--	V
	BRNTRIG_VBAT_CNTL = 0x7	--	3.0	--	V
Vhys (BOD) – Vbat Brown-out hysteresis	BRNHYST_VBAT_CNTL = 0x0	--	0	--	mV
	BRNHYST_VBAT_CNTL = 0x1	--	41	--	mV
	BRNHYST_VBAT_CNTL = 0x2	--	66	--	mV
	BRNHYST_VBAT_CNTL = 0x3	--	85	--	mV
Ton (BOD) – Vbat Brown-out detector turn-on time	--	--	20	--	µs

NOTE: $V_{fall(BOD)}$ = BOD falling edge. $V_{fall(BOD)} = V_{trig(BOD)} – V_{hys(BOD)}$.
NOTE: $V_{rise(BOD)}$ – BOD rising edge. $V_{rise(BOD)} = V_{trig(BOD)} – V_{trig(BOD)}$.

22.8 ADC Specifications

22.8.1 ADC

Table 178: ADC Specifications

NOTE: Typical values: T_A = 25°C, VDDIO_3 = 3.3V (for 1.8V <= VDD <= 3.6V voltage range), unless otherwise noted.
These are design guidelines and are based on bench characterization results and/or design simulation.
Minimum and maximum values: Unless otherwise specified, the minimum and maximum values are guaranteed in the worst conditions of ambient temperature, supply voltage, and frequencies by tests on bench.
Based on characterization, the minimum and maximum values refer to sample tests and represent the mean value ±3 times the standard deviation.

Symbol	Parameter	Condition	Min	Typ	Max	Units
--	ADC main clock • See Table 36, VCO Frequency Select, on page 97 (128 MHz). • See Table 37, AUPLL Post Divider Programming, on page 98 (ratio = 2).	FVCO=128 MHz post divider ratio = 2	--	64	--	MHz
Reference Voltage						
--	Internal reference voltage	--	1.20	1.22	1.23	V
--	External reference voltage	--	0.6	--	1.8	V
Analog Inputs						
--	Input voltage	Input buffer disabled	0	--	VDDIO_3	V
		Input buffer enabled	0.2	--	VDDIO_3 - 0.2	V
--	ADC voltage conversion range • For 16-bit and 16-bit audio setting, input signal should be limited within 99.7% of the ADC voltage conversion range. • Vref stands for the voltage reference of the ADC. Could be an internal 1.22V, analog 1.8V power supply(1.8V), or an external voltage (<1.8V).	--	-2*vref or -VDDIO_3	--	2*vref or VDDIO_3	V
C_{ADC}	Internal sampling and hold capacitance	--	--	1000	--	fF
R_{ADC}	Internal sampling switch resistance	--	1	--	--	kΩ
R_{MUX}	Input multiplexer impedance	--	1	--	--	kΩ

Table 178: ADC Specifications (Continued)

NOTE: Typical values: T_A = 25°C, VDDIO_3 = 3.3V (for 1.8V <= VDD <= 3.6V voltage range), unless otherwise noted.
These are design guidelines and are based on bench characterization results and/or design simulation.
Minimum and maximum values: Unless otherwise specified, the minimum and maximum values are guaranteed in the worst conditions of ambient temperature, supply voltage, and frequencies by tests on bench.
Based on characterization, the minimum and maximum values refer to sample tests and represent the mean value ±3 times the standard deviation.

Symbol	Parameter	Condition	Min	Typ	Max	Units
R_S	External input resistance R_S maximum formula: $R_S < \frac{1/f_{ADC}}{4 \times \ln(2^{15}) \times C_{ADC}} - (R_{ADC} + R_{MUX})$	2 MHz ADC operating clock without input buffer 16-bit settling accuracy	--	--	8	kΩ
		2 MHz ADC operating clock with input buffer 16-bit settling accuracy	--	--	46	kΩ
--	Input frequency range Defined as input signal -3 dB bandwidth through analog path.	With input buffer	--	31	--	MHz
		Without input buffer	--	130	--	MHz
Conversion Rate (ADC main clock frequency is 64 MHz)						
--	ADC sampling clock frequency	Fast mode	--	2	--	MHz
		Low-power mode	--	0.2	2	MHz
--	Conversion time in ADC clocks	12-bit setting	--	1	--	clock cycles
		14-bit setting	--	11	--	clock cycles
		16-bit setting	--	35	--	clock cycles
		16-bit audio setting	--	131	--	clock cycles
--	1-shot latency (It is recommended to perform a calibration after each power-up.)	16-bit audio setting @ 2 MHz	65.5 + T_{WARM}	--	--	µs
		16-bit setting @ 2 MHz	17.5 + T_{WARM}	--	--	µs
		14-bit setting @ 2 MHz	5.5 + T_{WARM}	--	--	µs
		12-bit setting @ 2 MHz	0.5 + T_{WARM}	--	--	µs

Table 178: ADC Specifications (Continued)

NOTE: Typical values: T_A = 25°C, VDDIO_3 = 3.3V (for 1.8V <= VDD <= 3.6V voltage range), unless otherwise noted.
These are design guidelines and are based on bench characterization results and/or design simulation.
Minimum and maximum values: Unless otherwise specified, the minimum and maximum values are guaranteed in the worst conditions of ambient temperature, supply voltage, and frequencies by tests on bench.
Based on characterization, the minimum and maximum values refer to sample tests and represent the mean value ±3 times the standard deviation.

Symbol	Parameter	Condition	Min	Typ	Max	Units
--	Data rate	16-bit audio setting @ 2 MHz	--	15.27	--	kHz
		16-bit setting @ 2 MHz	--	57.14	--	kHz
		14-bit setting @ 2 MHz	--	181.81	--	kHz
		12-bit setting @ 2 MHz	--	2000	--	kHz
DC Accuracy						
--	Resolution	Single-ended	--	11	15	bits
		Differential	--	12	16	bits
	Offset error	Before calibration 16-bit setting	--	±30	--	LSb
		After calibration 16-bit setting	--	±4	--	LSb
Dynamic Performance						
SNDR	Signal-to-Noise and-Distortion Ratio • With internal 1.22V reference, signal amplitude is 1.15Vp. • 1 kHz sine-wave differential input, 0 dB of full scale, 2 MHz ADC clock.	12-bit setting	--	64	--	dB
		14-bit setting	--	70	--	dB
		16-bit setting	--	76	--	dB
		16-bit audio setting	--	82	--	dB
--	Dynamic Range • With internal 1.22V reference, signal amplitude is 1mVp. • 1 kHz sine-wave differential input, -60 dB of full scale, 4 MHz ADC clock.	12-bit setting	--	66	--	dB
		14-bit setting	--	72	--	dB
		16-bit setting	--	78	--	dB
		16-bit audio setting	--	84	--	dB
Warm-up Time						

Table 178: ADC Specifications (Continued)

NOTE: Typical values: T_A = 25°C, VDDIO_3 = 3.3V (for 1.8V <= VDD <= 3.6V voltage range), unless otherwise noted.
These are design guidelines and are based on bench characterization results and/or design simulation.
Minimum and maximum values: Unless otherwise specified, the minimum and maximum values are guaranteed in the worst conditions of ambient temperature, supply voltage, and frequencies by tests on bench.
Based on characterization, the minimum and maximum values refer to sample tests and represent the mean value ±3 times the standard deviation.

Symbol	Parameter	Condition	Min	Typ	Max	Units
T_{WARM}	Warm-up time of ADC and internal reference generator (T_{WARM}) • Total warm-up time (T_{WARM}) depends on the current energy mode and 'gpadc_timebase' setting. • Programmable ADC main clock frequency = 64 MHz	--	1	--	32	µs

22.8.2 Temperature Sensor

Table 179: ADC Temperature Sensor Specifications

NOTE: Typical values: T_A = 25°C, VDDIO_3 = 3.3V (for 1.8V <= VDD <= 3.6V voltage range), unless otherwise noted.
These are design guidelines and are based on bench characterization results and/or design simulation.
On-chip temperature can be measured by using the ADC with an internal voltage reference and 16-bit resolution setting.
Minimum and maximum values: Unless otherwise specified, the minimum and maximum values are guaranteed in the worst conditions of ambient temperature, supply voltage, and frequencies by tests on bench.
Based on characterization, the minimum and maximum values refer to sample tests and represent the mean value ±3 times the standard deviation.

Parameter	Condition	Min	Typ	Max	Units
ADC main clock • See Table 36, VCO Frequency Select, on page 97 (128 MHz). • See Table 37, AUPLL Post Divider Programming, on page 98 (ratio = 2). • See Table 517, ADC General Register (adc_reg_general), on page 695 (clk_div_ratio value = 4).	FVCO = 128 MHz post divider ratio = 2 clk_div_ratio value = 4	--	16	--	MHz
Internal reference voltage	--	1.20	1.22	1.23	V
ADC operating clock frequency	--	--	500	--	kHz
Conversion time in ADC operating clocks	16-bit setting	--	131	--	clock cycles
Data rate	ADC main clock frequency is 16 MHz.	--	3.82	--	kHz
Measurement latency Total warm-up time (T_{WARM}) depends on the current energy mode and 'gpadc_timebase' setting.	ADC main clock frequency is 16 MHz.	--	262+ T_{WARM}	--	µs
Resolution	1LSb=1.2V/2^15	--	0.15	--	°C/LSb
Measurement accuracy	Initial accuracy w/o calibration	--	±5	--	°C
	1-temperature calibration	--	±3	--	°C

22.8.3 Battery Voltage Monitor

Table 180: ADC Battery Voltage Monitor Specifications

NOTE: Typical values: T_A = 25°C, VDDIO_3 = 3.3V (for 1.8V <= VDD <= 3.6V voltage range), unless otherwise noted.
These are design guidelines and are based on bench characterization results and/or design simulation.
Minimum and maximum values: Unless otherwise specified, the minimum and maximum values are guaranteed in the worst conditions of ambient temperature, supply voltage, and frequencies by tests on bench.
Based on characterization, the minimum and maximum values refer to sample tests and represent the mean value ±3 times the standard deviation.

Parameter	Condition	Min	Typ	Max	Units
ADC main clock • See Table 36, VCO Frequency Select, on page 97 (FVCO = 128 MHz). • See Table 37, AUPLL Post Divider Programming, on page 98 (ratio = 2).	FVCO = 128 MHz post divider ratio = 2	--	64	--	MHz
Internal reference voltage	--	1.20	1.22	1.23	V
Data rate (ADC main clock frequency is 64 MHz.)	14-bit setting 2 MHz ADC operating clock	--	181.81	--	kHz
Measurement latency Total warm-up time (T_{WARM}) depends on the current energy mode and 'gpadc_timebase' setting.	14-bit setting 2 MHz ADC operating clock	--	5.5+ T_{WARM}	--	µs
Measurement accuracy Based on characterization, not tested in production.	Initial accuracy w/o calibration Internal Vref T = -40°C to 125°C	-2	--	2	%

22.8.4 Audio Mode

Table 181: ADC Audio Mode Specifications

NOTE: T_A = 25°C, VDDIO_3 = 3.3V, AVDD18 = 1.8V, unless otherwise noted.

Parameter	Condition	Min	Typ	Max	Units
ADC main clock • See Table 36, VCO Frequency Select, on page 97 (134.14 MHz). • See Table 37, AUPLL Post Divider Programming, on page 98 (ratio = 2).	FVCO = 134.14 MHz post divider ratio = 2	--	67.07	--	MHz
ADC operating clock frequency	--	--	2.1	--	MHz
Conversion time in ADC operating clocks	16-bit audio setting	--	131	--	clock cycles
Data rate	2.1 MHz ADC operating clock	--	16	--	kHz
PGA Input					
Analog input voltage	Any pin (in analog input mode)	0	--	AVDD18	V

Table 181: ADC Audio Mode Specifications (Continued)
NOTE: T_A = 25°C, VDDIO_3 = 3.3V, AVDD18 = 1.8V, unless otherwise noted.

Parameter	Condition	Min	Typ	Max	Units
Maximum differential input signal swing	--	--	1.2 / PGA_Gain	--	V
Common mode input range	For DC couple input	--	0.9	--	V
Differential input impedance	PGA_GAIN = 4x	--	80	--	kΩ
	PGA_GAIN = 8x	--	40	--	kΩ
	PGA_GAIN = 16x	--	20	--	kΩ
	PGA_GAIN = 32x	--	10	--	kΩ
Analog source resistance The analog signal source resistance should be kept as minimum as possible for PGA gain accuracy.	--	--	100	--	Ω
PGA Performance					
PGA gain	ADC.AUDIO.PGA_GAIN = 3'b000	--	4	--	--
	ADC.AUDIO.PGA_GAIN = 3'b001	--	8	--	--
	ADC.AUDIO.PGA_GAIN = 3'b010	--	16	--	--
	ADC.AUDIO.PGA_GAIN = 3'b011	--	32	--	--
ADC gain The PGA must work together with the input buffer for sufficient driving capability. If the input buffer is bypass when PGA is on, distortion may occur. The total ADC gain is the multiplication of PGA gain and the input buffer gain (0.5/1/2).	--	PGA_Gain * Input buffer_Gain			--
Input signal bandwidth	--	--	--	1	MHz
Gain switching settling time	--	--	--	10	µs

22.9 DAC Specifications

Table 182: DAC Specifications

NOTE: Typical values: T_A = 25°C, VDDIO_3 = 3.3V (for 1.8V <= VDD <= 3.6V voltage range), unless otherwise noted.
These are design guidelines and are based on bench characterization results and/or design simulation.
Minimum and maximum values: Unless otherwise specified, the minimum and maximum values are guaranteed in the worst conditions of ambient temperature, supply voltage, and frequencies by tests on bench.
Based on characterization, the minimum and maximum values refer to sample tests and represent the mean value ±3 times the standard deviation.

Parameter	Condition	Min	Typ	Max	Units
DAC main clock • See Table 36, VCO Frequency Select, on page 97 (FVCO = 128 MHz). • See Table 37, AUPLL Post Divider Programming, on page 98 (ratio = 2).	FVCO = 128 MHz post divider ratio = 2	--	64	--	MHz
Reference Voltage (Vref)					
Internal reference voltage	gpdac_a_range[1:0]=11	--	1.42	--	V
	gpdac_a_range[1:0]=10/01	--	1.01	--	V
	gpdac_a_range[1:0]=00	--	0.64	--	V
External reference voltage	gpdac_a_range[1:0]=11	--	Vref_ext*0.71	--	V
	gpdac_a_range[1:0]=10/01	--	Vref_ext*0.505	--	V
	gpdac_a_range[1:0]=00	--	Vref_ext*0.32	--	V
Externally supplied reference voltage (Vref_ext)	Ref_sel=1	1.5	--	2	V
Conversion Range					
Analog output range	Single ended See Table 155, Output Voltage Calculation Formula, on page 302.	0.12	--	1.6	V
Output Load					
Resistive load (minimum resistive load between DAC output and VSS)	single-ended	5	--	--	kΩ
	differential	5	--	--	kΩ
Capacitive load (maximum capacitive load at DAC output)	--	--	--	50	pF
Conversion Rate					
Conversion rate	clk_ctrl[1:0]=2'b11	--	500	--	kHz
	clk_ctrl[1:0]=2'b10	--	250	--	kHz
	clk_ctrl[1:0]=2'b01	--	125	--	kHz
	clk_ctrl[1:0]=2'b00 (default)	--	62.5	--	kHz

Table 182: DAC Specifications (Continued)

NOTE: Typical values: T_A = 25°C, VDDIO_3 = 3.3V (for 1.8V <= VDD <= 3.6V voltage range), unless otherwise noted.
These are design guidelines and are based on bench characterization results and/or design simulation.
Minimum and maximum values: Unless otherwise specified, the minimum and maximum values are guaranteed in the worst conditions of ambient temperature, supply voltage, and frequencies by tests on bench.
Based on characterization, the minimum and maximum values refer to sample tests and represent the mean value ±3 times the standard deviation.

Parameter	Condition	Min	Typ	Max	Units
DC Accuracy					
Resolution	single-ended	--	--	10	bits
	differential	--	--	10	bits
Differential nonlinearity (RMS)	Guaranteed monotonic, internal 1.2V reference	--	±0.5	--	LSb[1]
Integral nonlinearity (best-fit method)	Internal reference	--	±2	--	LSb
Offset error	a_range=2'b11, internal vref (See Table 155, Output Voltage Calculation Formula, on page 302.) (difference between measured value at code 0x0 and the ideal value (0.18V))	--	20	--	mV
Gain error (after offset removal)	a_range=2'b11, internal vref	--	2	--	%
PSRR	Power supply rejection ratio (to AVDD18) (static DC measurement)	--	60	--	dB

1. 1 LSb= Vref/1024

22.10 ACOMP Specifications

Table 183: ACOMP Specifications

NOTE: Typical values: T_A = 25°C, VDDIO_3 = 3.3V (for 1.8V <= VDD <= 3.6V voltage range), unless otherwise noted.
These are design guidelines and are based on bench characterization results and/or design simulation.
Minimum and maximum values: Unless otherwise specified, the minimum and maximum values are guaranteed in the worst conditions of ambient temperature, supply voltage, and frequencies by tests on bench.
Based on characterization, the minimum and maximum values refer to sample tests and represent the mean value ±3 times the standard deviation.

Symbol	Parameter	Condition	Min	Typ	Max	Units
Analog Input						
--	Analog input voltage	Any pin (in analog input mode)	0	--	VDDIO_3	V
--	Common mode input range	--	0	--	VDDIO_3	V
Reference Voltage						
--	Internal reference voltage	--	1.20	1.22	1.23	V
Analog Response Time (no digital delay) **NOTE:** Digital delay can be up to a maximum of 2, 19.6 MHz clock periods). **NOTE:** Vcm is the common-mode voltage on COMP_P and COMP_N.						
--	Fast response mode MODE 3, Vcm = 1.5V	Overdrive (COMP_P – COMP_N = ±100 mV	--	130	--	ns
--	Medium response mode MODE 2, Vcm = 1.5V	Overdrive (COMP_P – COMP_N = ±100 mV	--	190	--	ns
--	Slow response mode MODE 1, Vcm = 1.5V	Overdrive (COMP_P – COMP_N = ±100 mV	--	450	--	ns
DC Offset						
--	Offset voltage	--	--	±14	--	mV
--	Hysteresis	Programmed in 7 steps and 0	--	10	--	mV
--			--	20	--	mV
--			--	30	--	mV
--			--	40	--	mV
--			--	50	--	mV
--			--	60	--	mV
--			--	70	--	mV
Warm-up Time						
T_{WARM}	Warm-up time of ACOMP and internal reference generator	VDDIO_3 = 3.3V	--	0.5	1.0	µs
		VDDIO_3 = 1.8V	--	1.0	1.3	µs

22.11 AC Specifications

22.11.1 SSP Timing and Specifications

Figure 113:SSP Serial Frame Format Timing Diagram

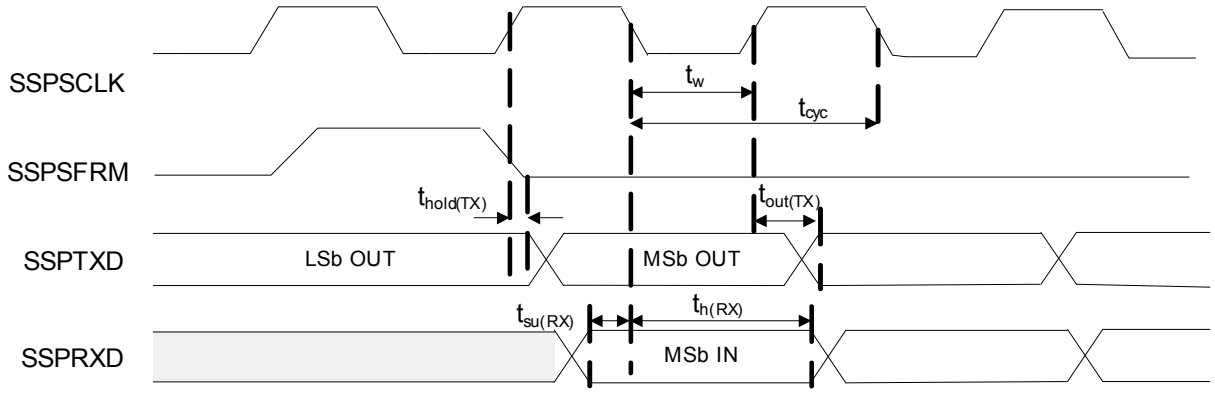
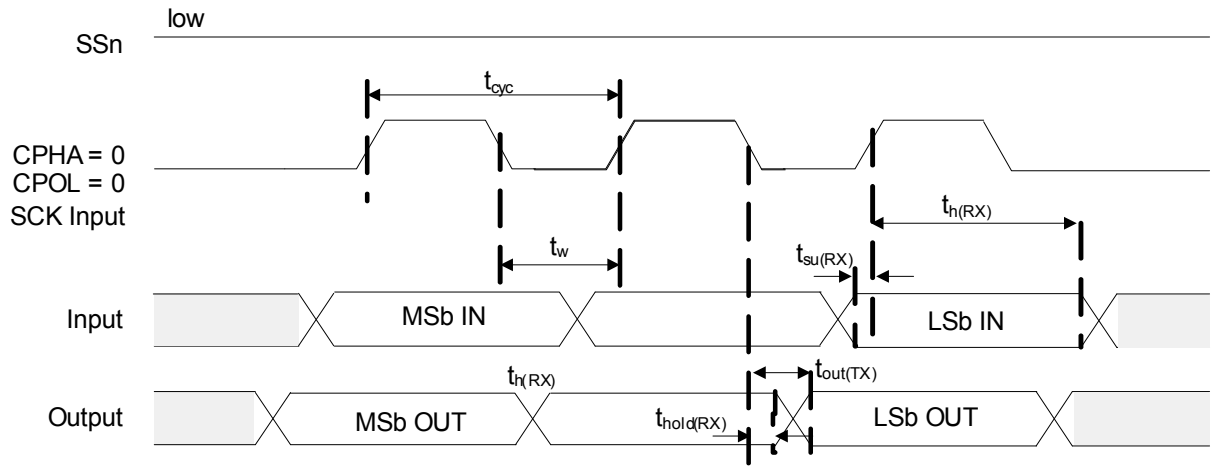


Table 184: SSP Timing Data

Symbol	Parameter	Condition	Min	Typ	Max	Units
$t_{out(TX)}$	TX delay time	master	--	--	3	ns
		slave	--	--	11.5	ns
$t_{hold(TX)}$	TX hold time	master	0	--	--	ns
		slave	0	--	--	ns
$t_{su(RX)}$	Setup time RX valid before clock low	master	12	--	--	ns
		slave	4	--	--	ns
$t_{h(RX)}$	Hold time, RX data valid after clock low	master	0	--	--	ns
		slave	2	--	--	ns
t_{cyc}	Serial bit clock cycle time	master	40	--	--	ns
		slave	40	--	--	ns
t_w	Serial clock high/low time	master	$\frac{T_{cyc}}{2} - 0.5$	--	--	ns
		slave	$\frac{T_{cyc}}{2} - 0.5$	--	--	ns

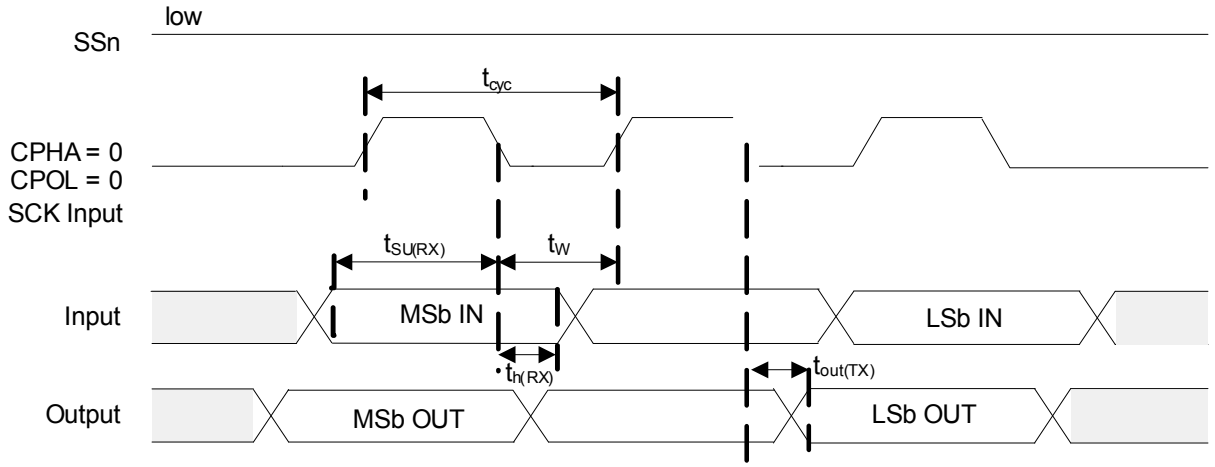
22.11.2 QSPI Timing and Specifications

Figure 114:QSPI Timing Diagram (Normal)



Shaded areas are not valid.

Figure 115:QSPI Timing Diagram (Use Second Clock Capture Edge)



Shaded areas are not valid.

Note: When the [CLK_CAPT_EDGE] = 1'b1, the interface clock frequency can be up to 50 MHz. t_{cyc} = 20 ns. Used only in mode[0,0) and mode(1,0).

Table 185: QSPI Timing Data

NOTE: Capacitive load C=5 pF. Minimum annotated_transition = 0.5 ns. Maximum annotated_transition=1.5 ns.

Symbol	Parameter	Condition	Min	Typ	Max	Units
t_{cyc}	QSPI clock cycle time	master	40	--	--	ns
t_w	Clock high and low time	master	$\frac{T_{cyc}}{2} - 0.5$	--	--	ns
$t_{su(RX)}$	Data input setup time	master	8	--	--	ns
$t_{h(RX)}$	Data input hold time	master	0	--	--	ns
$t_{out(TX)}$	Data output delay time	master	--	--	7	ns
$t_{hold(TX)}$	TX hold time	master	0	--	--	ns

22.11.3 USB Timing and Specifications

Figure 116:USB Timing Diagram

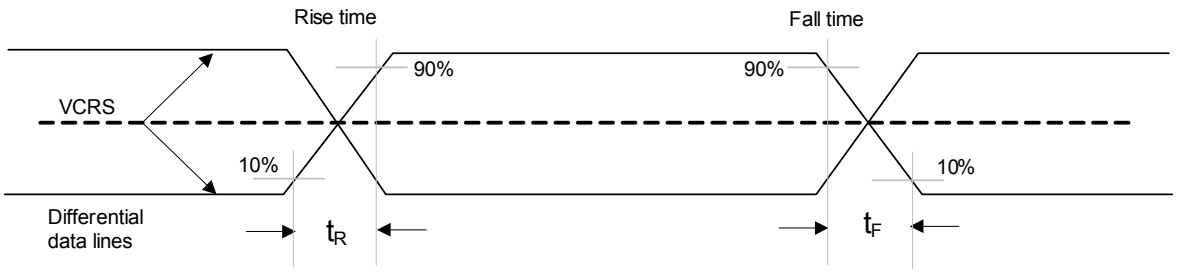


Table 186: USB Timing Data

Symbol	Parameter	Condition	Min	Typ	Max	Units
Supply Voltage						
VBUS	High-power pin	--	4.75	--	5.25	V
VBUS	Low-power pin	--	4.40	--	5.25	V
Input Level for High Speed						
V_{hssq}	High-speed squelch detection threshold	--	100	--	150	mV
V_{hsdsc}	High-speed disconnect detection threshold	--	525	--	625	mV
V_{hscm}	High-speed data signaling common mode voltage range	--	-50	--	500	mV
Input Level for Low/Full Speed						
V_{ih}	High (driven)	Measured at A or B connector	2	--	--	V

Table 186: USB Timing Data (Continued)

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{ihz}	High (floating)	Measured at A or B connector	2.7	--	3.6	V
V_{il}	Low	Measured at A or B connector	--	--	0.8	V
Output Level for High Speed						
V_{hsol}	High-speed idle level	--	-10.0	--	10.0	mV
V_{hsoh}	High-speed data signaling high	--	360	--	440	mV
V_{hsol}	High-speed data signaling low	--	-10.0	--	10.0	mV
V_{chirpj}	Chirp J level	--	700	--	1100	mV
V_{chirpk}	Chirp K level	--	-900	--	-500	mV
Output Level for Low/Full Speed						
V_{il}	Low	Measured with RL of 1.425k to 3.6V	0.0	--	0.3	V
V_{ih}	High (driven)	Measured with RL of 14.25k to ground	2.8	--	3.6	V
V_{crs}	Output signal crossover voltage	Excluding first transition from Idle state	1.3	--	2.0	V
High-Speed Source Electrical Characteristics						
T_{hsr}	Rise time	--	500	--	--	ps
T_{hsf}	Fall time	--	500	--	--	ps
Full-Speed Source Electrical Characteristics						
T_{fr}	Rise time	--	4	--	20	ns
T_{ff}	Fall time	--	4	--	20	ns
T_{frfm}	Rise and fall time matching	Tfr/Tff	90	--	111.11	%
Low-Speed Source Electrical Characteristics						
T_{lr}	Rise time	--	75	--	300	ns
T_{lf}	Fall time	--	75	--	300	ns
T_{lrfm}	Rise and fall time matching	Tlr/Tlf	80	--	125	%

22.11.4 RESETn Pin Specification

Table 187: RESETn Pin Specification

Symbol	Parameter	Condition	Min	Typ	Max	Units
--	Minimum reset pulse width on RESETn pin[1]	--	80	--	--	µs

1. From design, not production.

22.12 WLAN Radio Specifications

The 88MW300/302 WLAN radio interface pin is powered from the AVDD33 voltage supply.

22.12.1 Receive Mode Specifications

Table 188: LNA and Rx RF Mixer Specifications—802.11n/g/b

Parameter	Condition	Min	Typ	Max	Units
RF frequency range	2.4 GHz—IEEE 802.11n/g/b	2400	--	2500	MHz
Rx input IP3 at RF high gain (In-Band)	Rx input IP3 when LNA in high gain mode (24 dB) at chip input	--	-15	--	dBm
Maximum input level (802.11b, 1 Mbps)	--	--	--	0	dBm
Maximum input level (802.11g, 54 Mbps)	--	--	--	-3	dBm
Maximum input level (802.11n, MCS7)	--	--	--	-6	dBm
Maximum Receive Sensitivity					
802.11b, 1 Mbps	--	--	-98	-97	dBm
802.11b, 2 Mbps	--	--	-95	-94	dBm
802.11b, 5.5 Mbps	--	--	-93	-92	dBm
802.11b, 11 Mbps	--	--	-90	-88	dBm
802.11g, 6 Mbps	--	--	-92	-91	dBm
802.11g, 9 Mbps	--	--	-91	-90	dBm
802.11g, 12 Mbps	--	--	-89	-88	dBm
802.11g, 18 Mbps	--	--	-87	-86	dBm
802.11g, 24 Mbps	--	--	-84	-83	dBm
802.11g, 36 Mbps	--	--	-81	-80	dBm
802.11g, 48 Mbps	--	--	-76	-75	dBm
802.11g, 54 Mbps	--	--	-75	-74	dBm
802.11n, MCS0	--	--	-91	-90	dBm
802.11n, MCS1	--	--	-89	-88	dBm

Table 188: LNA and Rx RF Mixer Specifications—802.11n/g/b (Continued)

Parameter	Condition	Min	Typ	Max	Units
802.11n, MCS2	--	--	-87	-86	dBm
802.11n, MCS3	--	--	-84	-83	dBm
802.11n, MCS4	--	--	-81	-80	dBm
802.11n, MCS5	--	--	-76	-75	dBm
802.11n, MCS6	--	--	-75	-74	dBm
802.11n, MCS7	--	--	-73	-72	dBm

22.12.2 Transmit Mode Specifications

Table 189: Tx Mode Specifications—802.11n/g/b

Parameter	Condition	Min	Typ	Max	Units
RF frequency range	2.4 GHz—IEEE 802.11n/g/b	2400	--	2500	MHz
Tx output saturation power at chip output	2.4 GHz—IEEE 802.11n/g/b	--	26	--	dBm
Tx carrier suppression (CW)	Carrier suppression at chip output	--	-36	--	dB
Tx I/Q suppression with IQ calibration	I/Q suppression at chip output	--	-45	--	dBc
Tx power with mask and EVM compliance to IEEE limits—normal power mode					
802.11b, 1 Mbps	--	17	19	--	dBm
802.11g, 6 Mbps	--	17	19	--	dBm
802.11g, 54 Mbps	--	16	18	--	dBm
802.11n, MCS0	--	17	19	--	dBm
802.11n, MCS7	--	16	18	--	dBm
Tx power with mask and EVM compliance to IEEE limits—low power mode					
802.11b, 1 Mbps	--	10	11	--	dBm
802.11g, 6 Mbps	--	10	11	--	dBm
802.11g, 54 Mbps	--	10	11	--	dBm
802.11n, MCS0	--	10	11	--	dBm
802.11n, MCS7	--	10	11	--	dBm

22.12.3 Local Oscillator Specifications

Table 190: Local Oscillator Specifications

Parameter	Condition	Min	Typ	Max	Units
Phase noise	Measured at 2.438 GHz at 100 kHz offset	--	-103	--	dBc/Hz
Integrated RMS phase noise at RF output (from 1 kHz to 10 MHz)	Reference clock frequency = 38.4 MHz (2.4 GHz)	--	0.6	--	degrees
Frequency resolution	--	0.02	--	--	kHz

23 Part Order Numbering/Package Marking

23.1 Part Order Numbering

Figure 117 shows the part order numbering scheme for the device. Refer to Marvell Field Application Engineers (FAEs) or representatives for further information when ordering parts.

Figure 117:Sample Part Number

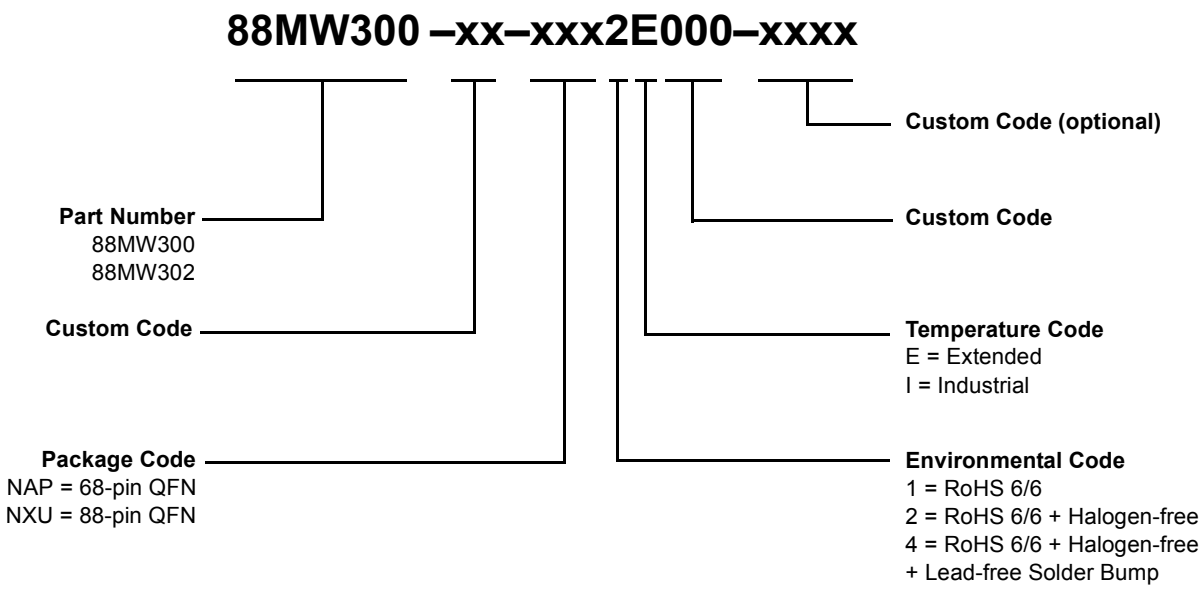


Table 191: Part Order Options[1]

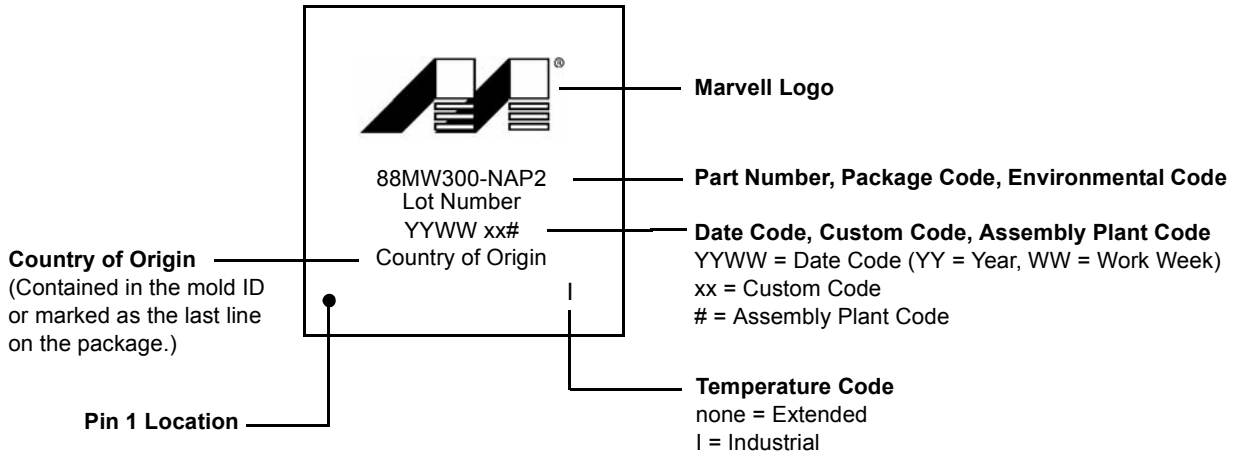
Package Type	Part Order Number
88MW300	
68-pin QFN (tray)	88MW300-xx-NAP2E000
68-pin QFN (tape-and-reel)	88MW300-xx-NAP2E000-P123
68-pin QFN (tray)	88MW300-xx-NAP2I000
68-pin QFN (tape-and-reel)	88MW300-xx-NAP2I000-P123
88MW302	
88-pin QFN (tray)	88MW302-xx-NXU2E000
88-pin QFN (tape-and-reel)	88MW302-xx-NXU2E000-P123
88-pin QFN (tray)	88MW302-xx-NXU2I000
88-pin QFN (tape-and-reel)	88MW302-xx-NXU2I000-P123

1. See Table 159, Recommended Operating Conditions, on page 316 for supported temperature ranges.

23.2 Package Marking

Figure 118 shows a sample package marking and pin 1 location for the device.

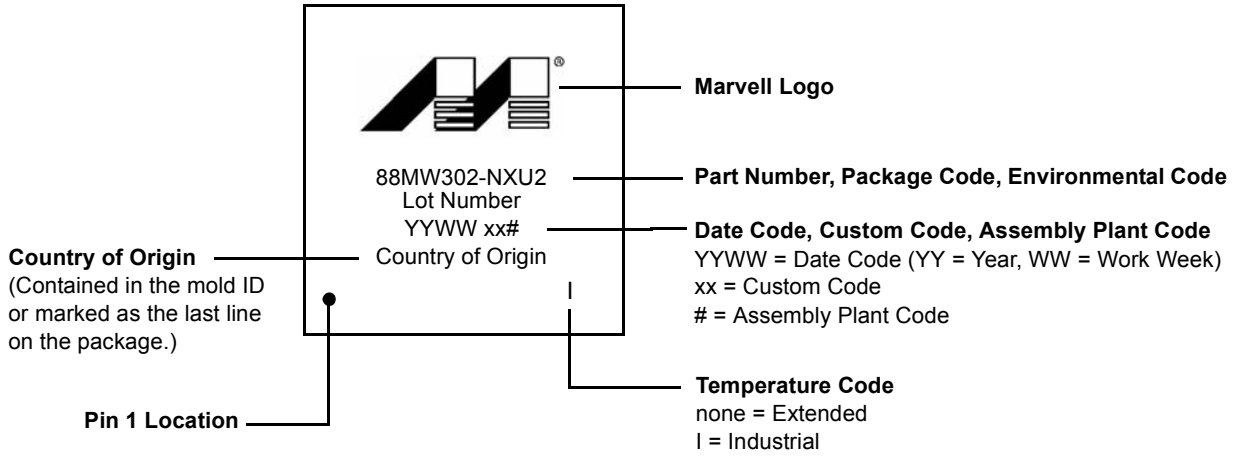
Figure 118:Package Marking and Pin 1 Location—88MW300



Note: The above drawing is not drawn to scale. Location of markings is approximate.

Figure 119 shows a sample package marking and pin 1 location for the 88MW302 device.

Figure 119:Package Marking and Pin 1 Location—88MW302



Note: The above drawing is not drawn to scale. Location of markings is approximate.

88MW300/302

Registers

THIS PAGE INTENTIONALLY LEFT BLANK

A 88MW300/302 Register Set

A.1 Overall Memory Map

Table 192: Overall Memory Map

Register Block	Register Name	Base Address
DMA	DMAC Address Block	0x4400_0000
USBC	USBC Address Block	0x4400_1000
FlashC	Flash Controller Address Block	0x4400_3000
AES	AES Address Block	0x4400_4000
CRC	CRC Address Block	0x4400_5000
I2C0	I2C Address Block	0x4600_0000
QSPI	QSPI Address Block	0x4601_0000
SSP0	SSP Address Block	0x4602_0000
UART0	UART Address Block	0x4604_0000
GPIO	GPIO Address Block	0x4606_0000
GPT0	GPT Address Block	0x4607_0000
GPT1	GPT Address Block	0x4608_0000
--	Reserved	0x4609_0000
RC32	RC32 Address Block	0x460A_0000
ADC	ADC Address Block	0x460B_0000
DAC	DAC Address Block	0x460B_0200
ACOMP	ACOMP Address Block	0x460B_0400
UART1	UART Address Block	0x460C_0000
SSP1	SSP Address Block	0x460D_0000
SSP2	SSP Address Block	0x4800_0000
Pin Mux	PINMUX Address Block	0x4801_0000
UART2	UART Address Block	0x4802_0000
WDT	WDT Address Block	0x4804_0000
I2C1	I2C Address Block	0x4805_0000
Reserved	Reserved	0x4806_0000
GPT2	GPT Address Block	0x4807_0000
GPT3	GPT Address Block	0x4808_0000
RTC	RTC Address Block	0x4809_0000
PMU	PMU Address Block	0x480A_0000
SYS_CTL	System Control Address Block	0x480B_0000
4k_MEM	4k_MEM	0x480C_0000

THIS PAGE INTENTIONALLY LEFT BLANK

A.2 DMAC Address Block

A.2.1 DMAC Register Map

Table 193: DMAC Register Map

Offset	Name	HW Rst	Description	Details
0x000	MASK_BLOCKINT	0x0000_0000	DMA Channel Block Transfer Interrupt Mask Register	Page: 360
0x004	STATUS_BLOCKINT	0x0000_0000	DMA Channel Block Transfer Interrupt Register	Page: 365
0x008	MASK_TFRINT	0x0000_0000	DMA Channel Transfer Completion Interrupt Mask Register	Page: 369
0x00C	STATUS_TFRINT	0x0000_0000	DMA Channel Transfer Completion Interrupt Register	Page: 374
0x010	MASK_BUSERRINT	0x0000_0000	DMA Channel Bus Error Interrupt Mask Register	Page: 378
0x014	STATUS_BUSERRINT	0x0000_0000	DMA Channel Bus Error Interrupt Mask Register	Page: 382
0x018	MASK_ADDRERRINT	0x0000_0000	DMA Channel Source/target Address Alignment Error Interrupt Mask Register	Page: 387
0x01C	STATUS_ADDRERRINT	0x0000_0000	DMA Channel Source/target Address Alignment Error Interrupt Register	Page: 393
0x020	STATUS_CHLINT	0x0000_0000	DMA Channel Interrupt Register	Page: 397
0x080	HPROT	0x0000_0003	The Protection Control Signals Register	Page: 400
0x100	SADR0	0x0000_0000	DMA Source Address Register	Page: 400
0x104	TADR0	0x0000_0000	DMA Target Address Register	Page: 401
0x108	CTRLA0	0x0000_0000	DMA Control Register A	Page: 401
0x10C	CTRLB0	0x0000_0000	DMA Control Register B	Page: 403
0x110	CHL_EN0	0x0000_0000	DMA Channel Enable Register	Page: 403
0x114	CHL_STOP0	0x0000_0000	DMA Channel Stop Register	Page: 404
0x130	SADR1	0x0000_0000	DMA Source Address Register	Page: 400
0x134	TADR1	0x0000_0000	DMA Target Address Register	Page: 401
0x138	CTRLA1	0x0000_0000	DMA Control Register A	Page: 401
0x13C	CTRLB1	0x0000_0000	DMA Control Register B	Page: 403
0x140	CHL_EN1	0x0000_0000	DMA Channel Enable Register	Page: 403
0x144	CHL_STOP1	0x0000_0000	DMA Channel Stop Register	Page: 404
0x160	SADR2	0x0000_0000	DMA Source Address Register	Page: 400
0x164	TADR2	0x0000_0000	DMA Target Address Register	Page: 401

Table 193: DMAC Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x168	CTRLA2	0x0000_0000	DMA Control Register A	Page: 401
0x16C	CTRLB2	0x0000_0000	DMA Control Register B	Page: 403
0x170	CHL_EN2	0x0000_0000	DMA Channel Enable Register	Page: 403
0x174	CHL_STOP2	0x0000_0000	DMA Channel Stop Register	Page: 404
0x190	SADR3	0x0000_0000	DMA Source Address Register	Page: 400
0x194	TADR3	0x0000_0000	DMA Target Address Register	Page: 401
0x198	CTRLA3	0x0000_0000	DMA Control Register A	Page: 401
0x19C	CTRLB3	0x0000_0000	DMA Control Register B	Page: 403
0x1A0	CHL_EN3	0x0000_0000	DMA Channel Enable Register	Page: 403
0x1A4	CHL_STOP3	0x0000_0000	DMA Channel Stop Register	Page: 404
0x1C0	SADR4	0x0000_0000	DMA Source Address Register	Page: 400
0x1C4	TADR4	0x0000_0000	DMA Target Address Register	Page: 401
0x1C8	CTRLA4	0x0000_0000	DMA Control Register A	Page: 401
0x1CC	CTRLB4	0x0000_0000	DMA Control Register B	Page: 403
0x1D0	CHL_EN4	0x0000_0000	DMA Channel Enable Register	Page: 403
0x1D4	CHL_STOP4	0x0000_0000	DMA Channel Stop Register	Page: 404
0x1F0	SADR5	0x0000_0000	DMA Source Address Register	Page: 400
0x1F4	TADR5	0x0000_0000	DMA Target Address Register	Page: 401
0x1F8	CTRLA5	0x0000_0000	DMA Control Register A	Page: 401
0x1FC	CTRLB5	0x0000_0000	DMA Control Register B	Page: 403
0x200	CHL_EN5	0x0000_0000	DMA Channel Enable Register	Page: 403
0x204	CHL_STOP5	0x0000_0000	DMA Channel Stop Register	Page: 404
0x220	SADR6	0x0000_0000	DMA Source Address Register	Page: 400
0x224	TADR6	0x0000_0000	DMA Target Address Register	Page: 401
0x228	CTRLA6	0x0000_0000	DMA Control Register A	Page: 401
0x22C	CTRLB6	0x0000_0000	DMA Control Register B	Page: 403
0x230	CHL_EN6	0x0000_0000	DMA Channel Enable Register	Page: 403
0x234	CHL_STOP6	0x0000_0000	DMA Channel Stop Register	Page: 404
0x250	SADR7	0x0000_0000	DMA Source Address Register	Page: 400
0x254	TADR7	0x0000_0000	DMA Target Address Register	Page: 401
0x258	CTRLA7	0x0000_0000	DMA Control Register A	Page: 401

Table 193: DMAC Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x25C	CTRLB7	0x0000_0000	DMA Control Register B	Page: 403
0x260	CHL_EN7	0x0000_0000	DMA Channel Enable Register	Page: 403
0x264	CHL_STOP7	0x0000_0000	DMA Channel Stop Register	Page: 404
0x280	SADR8	0x0000_0000	DMA Source Address Register	Page: 400
0x284	TADR8	0x0000_0000	DMA Target Address Register	Page: 401
0x288	CTRLA8	0x0000_0000	DMA Control Register A	Page: 401
0x28C	CTRLB8	0x0000_0000	DMA Control Register B	Page: 403
0x290	CHL_EN8	0x0000_0000	DMA Channel Enable Register	Page: 403
0x294	CHL_STOP8	0x0000_0000	DMA Channel Stop Register	Page: 404
0x2B0	SADR9	0x0000_0000	DMA Source Address Register	Page: 400
0x2B4	TADR9	0x0000_0000	DMA Target Address Register	Page: 401
0x2B8	CTRLA9	0x0000_0000	DMA Control Register A	Page: 401
0x2BC	CTRLB9	0x0000_0000	DMA Control Register B	Page: 403
0x2C0	CHL_EN9	0x0000_0000	DMA Channel Enable Register	Page: 403
0x2C4	CHL_STOP9	0x0000_0000	DMA Channel Stop Register	Page: 404
0x2E0	SADR10	0x0000_0000	DMA Source Address Register	Page: 400
0x2E4	TADR10	0x0000_0000	DMA Target Address Register	Page: 401
0x2E8	CTRLA10	0x0000_0000	DMA Control Register A	Page: 401
0x2EC	CTRLB10	0x0000_0000	DMA Control Register B	Page: 403
0x2F0	CHL_EN10	0x0000_0000	DMA Channel Enable Register	Page: 403
0x2F4	CHL_STOP10	0x0000_0000	DMA Channel Stop Register	Page: 404
0x310	SADR11	0x0000_0000	DMA Source Address Register	Page: 400
0x314	TADR11	0x0000_0000	DMA Target Address Register	Page: 401
0x318	CTRLA11	0x0000_0000	DMA Control Register A	Page: 401
0x31C	CTRLB11	0x0000_0000	DMA Control Register B	Page: 403
0x320	CHL_EN11	0x0000_0000	DMA Channel Enable Register	Page: 403
0x324	CHL_STOP11	0x0000_0000	DMA Channel Stop Register	Page: 404
0x340	SADR12	0x0000_0000	DMA Source Address Register	Page: 400
0x344	TADR12	0x0000_0000	DMA Target Address Register	Page: 401
0x348	CTRLA12	0x0000_0000	DMA Control Register A	Page: 401
0x34C	CTRLB12	0x0000_0000	DMA Control Register B	Page: 403

Table 193: DMAC Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x350	CHL_EN12	0x0000_0000	DMA Channel Enable Register	Page: 403
0x354	CHL_STOP12	0x0000_0000	DMA Channel Stop Register	Page: 404
0x370	SADR13	0x0000_0000	DMA Source Address Register	Page: 400
0x374	TADR13	0x0000_0000	DMA Target Address Register	Page: 401
0x378	CTRLA13	0x0000_0000	DMA Control Register A	Page: 401
0x37C	CTRLB13	0x0000_0000	DMA Control Register B	Page: 403
0x380	CHL_EN13	0x0000_0000	DMA Channel Enable Register	Page: 403
0x384	CHL_STOP13	0x0000_0000	DMA Channel Stop Register	Page: 404
0x3A0	SADR14	0x0000_0000	DMA Source Address Register	Page: 400
0x3A4	TADR14	0x0000_0000	DMA Target Address Register	Page: 401
0x3A8	CTRLA14	0x0000_0000	DMA Control Register A	Page: 401
0x3AC	CTRLB14	0x0000_0000	DMA Control Register B	Page: 403
0x3B0	CHL_EN14	0x0000_0000	DMA Channel Enable Register	Page: 403
0x3B4	CHL_STOP14	0x0000_0000	DMA Channel Stop Register	Page: 404
0x3D0	SADR15	0x0000_0000	DMA Source Address Register	Page: 400
0x3D4	TADR15	0x0000_0000	DMA Target Address Register	Page: 401
0x3D8	CTRLA15	0x0000_0000	DMA Control Register A	Page: 401
0x3DC	CTRLB15	0x0000_0000	DMA Control Register B	Page: 403
0x3E0	CHL_EN15	0x0000_0000	DMA Channel Enable Register	Page: 403
0x3E4	CHL_STOP15	0x0000_0000	DMA Channel Stop Register	Page: 404
0x400	SADR16	0x0000_0000	Dma Source Address Register	Page: 400
0x404	TADR16	0x0000_0000	DMA Target Address Register	Page: 401
0x408	CTRLA16	0x0000_0000	DMA Control Register A	Page: 401
0x40C	CTRLB16	0x0000_0000	DMA Control Register B	Page: 403
0x410	CHL_EN16	0x0000_0000	DMA Channel Enable Register	Page: 403
0x414	CHL_STOP16	0x0000_0000	DMA Channel Stop Register	Page: 404
0x430	SADR17	0x0000_0000	DMA Source Address Register	Page: 400
0x434	TADR17	0x0000_0000	DMA Target Address Register	Page: 401
0x438	CTRLA17	0x0000_0000	DMA Control Register A	Page: 401
0x43C	CTRLB17	0x0000_0000	DMA Control Register B	Page: 403
0x440	CHL_EN17	0x0000_0000	DMA Channel Enable Register	Page: 403

Table 193: DMAC Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x444	CHL_STOP17	0x0000_0000	DMA Channel Stop Register	Page: 404
0x460	SADR18	0x0000_0000	DMA Source Address Register	Page: 400
0x464	TADR18	0x0000_0000	DMA Target Address Register	Page: 401
0x468	CTRLA18	0x0000_0000	DMA Control Register A	Page: 401
0x46C	CTRLB18	0x0000_0000	DMA Control Register B	Page: 403
0x470	CHL_EN18	0x0000_0000	DMA Channel Enable Register	Page: 403
0x474	CHL_STOP18	0x0000_0000	DMA Channel Stop Register	Page: 404
0x490	SADR19	0x0000_0000	DMA Source Address Register	Page: 400
0x494	TADR19	0x0000_0000	DMA Target Address Register	Page: 401
0x498	CTRLA19	0x0000_0000	DMA Control Register A	Page: 401
0x49C	CTRLB19	0x0000_0000	DMA Control Register B	Page: 403
0x4A0	CHL_EN19	0x0000_0000	DMA Channel Enable Register	Page: 403
0x4A4	CHL_STOP19	0x0000_0000	DMA Channel Stop Register	Page: 404
0x4C0	SADR20	0x0000_0000	DMA Source Address Register	Page: 400
0x4C4	TADR20	0x0000_0000	DMA Target Address Register	Page: 401
0x4C8	CTRLA20	0x0000_0000	DMA Control Register A	Page: 401
0x4CC	CTRLB20	0x0000_0000	DMA Control Register B	Page: 403
0x4D0	CHL_EN20	0x0000_0000	DMA Channel Enable Register	Page: 403
0x4D4	CHL_STOP20	0x0000_0000	DMA Channel Stop Register	Page: 404
0x4F0	SADR21	0x0000_0000	DMA Source Address Register	Page: 400
0x4F4	TADR21	0x0000_0000	DMA Target Address Register	Page: 401
0x4F8	CTRLA21	0x0000_0000	DMA Control Register A	Page: 401
0x4FC	CTRLB21	0x0000_0000	DMA Control Register B	Page: 403
0x500	CHL_EN21	0x0000_0000	DMA Channel Enable Register	Page: 403
0x504	CHL_STOP21	0x0000_0000	DMA Channel Stop Register	Page: 404
0x520	SADR22	0x0000_0000	DMA Source Address Register	Page: 400
0x524	TADR22	0x0000_0000	DMA Target Address Register	Page: 401
0x528	CTRLA22	0x0000_0000	DMA Control Register A	Page: 401
0x52C	CTRLB22	0x0000_0000	DMA Control Register B	Page: 403
0x530	CHL_EN22	0x0000_0000	DMA Channel Enable Register	Page: 403
0x534	CHL_STOP22	0x0000_0000	DMA Channel Stop Register	Page: 404

Table 193: DMAC Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x550	SADR23	0x0000_0000	DMA Source Address Register	Page: 400
0x554	TADR23	0x0000_0000	DMA Target Address Register	Page: 401
0x558	CTRLA23	0x0000_0000	DMA Control Register A	Page: 401
0x55C	CTRLB23	0x0000_0000	DMA Control Register B	Page: 403
0x560	CHL_EN23	0x0000_0000	DMA Channel Enable Register	Page: 403
0x564	CHL_STOP23	0x0000_0000	DMA Channel Stop Register	Page: 404
0x580	SADR24	0x0000_0000	DMA Source Address Register	Page: 400
0x584	TADR24	0x0000_0000	DMA Target Address Register	Page: 401
0x588	CTRLA24	0x0000_0000	DMA Control Register A	Page: 401
0x58C	CTRLB24	0x0000_0000	DMA Control Register B	Page: 403
0x590	CHL_EN24	0x0000_0000	DMA Channel Enable Register	Page: 403
0x594	CHL_STOP24	0x0000_0000	DMA Channel Stop Register	Page: 404
0x5B0	SADR25	0x0000_0000	DMA Source Address Register	Page: 400
0x5B4	TADR25	0x0000_0000	DMA Target Address Register	Page: 401
0x5B8	CTRLA25	0x0000_0000	DMA Control Register A	Page: 401
0x5BC	CTRLB25	0x0000_0000	DMA Control Register B	Page: 403
0x5C0	CHL_EN25	0x0000_0000	DMA Channel Enable Register	Page: 403
0x5C4	CHL_STOP25	0x0000_0000	DMA Channel Stop Register	Page: 404
0x5E0	SADR26	0x0000_0000	DMA Source Address Register	Page: 400
0x5E4	TADR26	0x0000_0000	DMA Target Address Register	Page: 401
0x5E8	CTRLA26	0x0000_0000	DMA Control Register A	Page: 401
0x5EC	CTRLB26	0x0000_0000	DMA Control Register B	Page: 403
0x5F0	CHL_EN26	0x0000_0000	DMA Channel Enable Register	Page: 403
0x5F4	CHL_STOP26	0x0000_0000	DMA Channel Stop Register	Page: 404
0x610	SADR27	0x0000_0000	DMA Source Address Register	Page: 400
0x614	TADR27	0x0000_0000	DMA Target Address Register	Page: 401
0x618	CTRLA27	0x0000_0000	DMA Control Register A	Page: 401
0x61C	CTRLB27	0x0000_0000	DMA Control Register B	Page: 403
0x620	CHL_EN27	0x0000_0000	DMA Channel Enable Register	Page: 403
0x624	CHL_STOP27	0x0000_0000	DMA Channel Stop Register	Page: 404
0x640	SADR28	0x0000_0000	DMA Source Address Register	Page: 400

Table 193: DMAC Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x644	TADR28	0x0000_0000	DMA Target Address Register	Page: 401
0x648	CTRLA28	0x0000_0000	DMA Control Register A	Page: 401
0x64C	CTRLB28	0x0000_0000	DMA Control Register B	Page: 403
0x650	CHL_EN28	0x0000_0000	DMA Channel Enable Register	Page: 403
0x654	CHL_STOP28	0x0000_0000	DMA Channel Stop Register	Page: 404
0x670	SADR29	0x0000_0000	DMA Source Address Register	Page: 400
0x674	TADR29	0x0000_0000	DMA Target Address Register	Page: 401
0x678	CTRLA29	0x0000_0000	DMA Control Register A	Page: 401
0x67C	CTRLB29	0x0000_0000	DMA Control Register B	Page: 403
0x680	CHL_EN29	0x0000_0000	DMA Channel Enable Register	Page: 403
0x684	CHL_STOP29	0x0000_0000	DMA Channel Stop Register	Page: 404
0x6A0	SADR30	0x0000_0000	DMA Source Address Register	Page: 400
0x6A4	TADR30	0x0000_0000	DMA Target Address Register	Page: 401
0x6A8	CTRLA30	0x0000_0000	DMA Control Register A	Page: 401
0x6AC	CTRLB30	0x0000_0000	DMA Control Register B	Page: 403
0x6B0	CHL_EN30	0x0000_0000	DMA Channel Enable Register	Page: 403
0x6B4	CHL_STOP30	0x0000_0000	DMA Channel Stop Register	Page: 404
0x6D0	SADR31	0x0000_0000	DMA Source Address Register	Page: 400
0x6D4	TADR31	0x0000_0000	DMA Target Address Register	Page: 401
0x6D8	CTRLA31	0x0000_0000	DMA Control Register A	Page: 401
0x6DC	CTRLB31	0x0000_0000	DMA Control Register B	Page: 403
0x6E0	CHL_EN31	0x0000_0000	DMA Channel Enable Register	Page: 403
0x6E4	CHL_STOP31	0x0000_0000	DMA Channel Stop Register	Page: 404
0x800	ACK_DELAY	0x0000_000C	DMA Ack Delay Cycle For Single Transfer In M2p Transfer Type Register	Page: 404
0x900	ERR_INFO0	0x0000_0000	DMA Error Information Register 0	Page: 405
0x904	ERR_INFO1	0x0000_0000	DMA Error Information Register 1	Page: 405
0x908	DIAGNOSE_INFO0	0x0000_0000	DMA Diagnose Information Register 0	Page: 406
0x90C	DIAGNOSE_INFO1	0x0000_0000	DMA Diagnose Information Register 1	Page: 406
0x910	DIAGNOSE_INFO2	0x0000_0000	DMA Diagnose Information Register 2	Page: 407
0x914	DIAGNOSE_INFO3	0x0000_0000	DMA Diagnose Information Register 3	Page: 407

Table 193: DMAC Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x918	DIAGNOSE_INFO4	0x0000_0000	DMA Diagnose Information Register 4	Page: 408
0x91C	DIAGNOSE_INFO5	0x0000_0000	DMA Diagnose Information Register 5	Page: 409
0x920	DIAGNOSE_INFO6	0x0000_0000	DMA Diagnose Information Register 6	Page: 410
0x924	DIAGNOSE_INFO7	0x0000_0000	DMA Diagnose Information Register 7	Page: 411

A.2.2 DMAC Registers

A.2.2.1 DMA Channel BLOCK TRANSFER INTERRUPT MASK Register (MASK_BLOCKINT)

Instance Name	Offset
MASK_BLOCKINT	0x000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	mask_blockint31	mask_blockint30	mask_blockint29	mask_blockint28	mask_blockint27	mask_blockint26	mask_blockint25	mask_blockint24	mask_blockint23	mask_blockint22	mask_blockint21	mask_blockint20	mask_blockint19	mask_blockint18	mask_blockint17	mask_blockint16	mask_blockint15	mask_blockint14	mask_blockint13	mask_blockint12	mask_blockint11	mask_blockint10	mask_blockint9	mask_blockint8	mask_blockint7	mask_blockint6	mask_blockint5	mask_blockint4	mask_blockint3	mask_blockint2	mask_blockint1	mask_blockint0
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 194: DMA Channel BLOCK TRANSFER INTERRUPT MASK Register (MASK_BLOCKINT)

Bits	Field	Type/ HW Rst	Description
31	mask_blockint31	R/W 0x0	DMA Channel 31 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT31 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
30	mask_blockint30	R/W 0x0	DMA Channel 30 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT30 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
29	mask_blockint29	R/W 0x0	DMA Channel 29 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT29 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt

Table 194: DMA Channel BLOCK TRANSFER INTERRUPT MASK Register (MASK_BLOCKINT)

Bits	Field	Type/ HW Rst	Description
28	mask_blockint28	R/W 0x0	DMA Channel 28 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT28 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
27	mask_blockint27	R/W 0x0	DMA Channel 27 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT27 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
26	mask_blockint26	R/W 0x0	DMA Channel 26 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT26 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
25	mask_blockint25	R/W 0x0	DMA Channel 25 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT25 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
24	mask_blockint24	R/W 0x0	DMA Channel 24 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT24 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
23	mask_blockint23	R/W 0x0	DMA Channel 23 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT23 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
22	mask_blockint22	R/W 0x0	DMA Channel 22 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT22 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
21	mask_blockint21	R/W 0x0	DMA Channel 21 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT21 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt

Table 194: DMA Channel BLOCK TRANSFER INTERRUPT MASK Register (MASK_BLOCKINT)

Bits	Field	Type/ HW Rst	Description
20	mask_blockint20	R/W 0x0	DMA Channel 20 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT20 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
19	mask_blockint19	R/W 0x0	DMA Channel 19 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT19 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
18	mask_blockint18	R/W 0x0	DMA Channel 18 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT18 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
17	mask_blockint17	R/W 0x0	DMA Channel 17 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT17 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
16	mask_blockint16	R/W 0x0	DMA Channel 16 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT16 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
15	mask_blockint15	R/W 0x0	DMA Channel 15 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT15 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
14	mask_blockint14	R/W 0x0	DMA Channel 14 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT14 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
13	mask_blockint13	R/W 0x0	DMA Channel 13 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT13 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt

Table 194: DMA Channel BLOCK TRANSFER INTERRUPT MASK Register (MASK_BLOCKINT)

Bits	Field	Type/ HW Rst	Description
12	mask_blockint12	R/W 0x0	DMA Channel 12 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT12 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
11	mask_blockint11	R/W 0x0	DMA Channel 11 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT11 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
10	mask_blockint10	R/W 0x0	DMA Channel 10 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT10 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
9	mask_blockint9	R/W 0x0	DMA Channel 9 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT9 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
8	mask_blockint8	R/W 0x0	DMA Channel 8 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT8 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
7	mask_blockint7	R/W 0x0	DMA Channel 7 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT7 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
6	mask_blockint6	R/W 0x0	DMA Channel 6 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT6 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
5	mask_blockint5	R/W 0x0	DMA Channel 5 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT5 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt

Table 194: DMA Channel BLOCK TRANSFER INTERRUPT MASK Register (MASK_BLOCKINT)

Bits	Field	Type/ HW Rst	Description
4	mask_blockint4	R/W 0x0	DMA Channel 4 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT4 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
3	mask_blockint3	R/W 0x0	DMA Channel 3 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT3 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
2	mask_blockint2	R/W 0x0	DMA Channel 2 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT2 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
1	mask_blockint1	R/W 0x0	DMA Channel 1 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT1 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt
0	mask_blockint0	R/W 0x0	DMA Channel 0 Block Transfer Interrupt Mask Bit This bit enables the interrupt when STATUS_BLOCKINT0 bit in STATUS_BLOCKINT is set. 0x0 = mask the corresponding block interrupt 0x1 = unmask the corresponding block interrupt

A.2.2.2 DMA Channel BLOCK TRANSFER INTERRUPT Register (STATUS_ BLOCKINT)

Instance Name	Offset
STATUS_BLOCKINT	0x004

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	status_blockint31	status_blockint30	status_blockint29	status_blockint28	status_blockint27	status_blockint26	status_blockint25	status_blockint24	status_blockint23	status_blockint22	status_blockint21	status_blockint20	status_blockint19	status_blockint18	status_blockint17	status_blockint16	status_blockint15	status_blockint14	status_blockint13	status_blockint12	status_blockint11	status_blockint10	status_blockint9	status_blockint8	status_blockint7	status_blockint6	status_blockint5	status_blockint4	status_blockint3	status_blockint2	status_blockint1	status_blockint0
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 195: DMA Channel BLOCK TRANSFER INTERRUPT Register (STATUS_BLOCKINT)

Bits	Field	Type/ HW Rst	Description
31	status_blockint31	R/W1CLR 0x0	DMA Channel 31 Block Transfer Interrupt Bit This interrupt is generated on channel 31 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
30	status_blockint30	R/W1CLR 0x0	DMA Channel 30 Block Transfer Interrupt Bit This interrupt is generated on channel 30 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
29	status_blockint29	R/W1CLR 0x0	DMA Channel 29 Block Transfer Interrupt Bit This interrupt is generated on channel 29 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
28	status_blockint28	R/W1CLR 0x0	DMA Channel 28 Block Transfer Interrupt Bit This interrupt is generated on channel 28 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
27	status_blockint27	R/W1CLR 0x0	DMA Channel 27 Block Transfer Interrupt Bit This interrupt is generated on channel 27 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
26	status_blockint26	R/W1CLR 0x0	DMA Channel 26 Block Transfer Interrupt Bit This interrupt is generated on channel 26 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed

Table 195: DMA Channel BLOCK TRANSFER INTERRUPT Register (STATUS_BLOCKINT)

Bits	Field	Type/ HW Rst	Description
25	status_blockint25	R/W1CLR 0x0	DMA Channel 25 Block Transfer Interrupt Bit This interrupt is generated on channel 25 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
24	status_blockint24	R/W1CLR 0x0	DMA Channel 24 Block Transfer Interrupt Bit This interrupt is generated on channel 24 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
23	status_blockint23	R/W1CLR 0x0	DMA Channel 23 Block Transfer Interrupt Bit This interrupt is generated on channel 23 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
22	status_blockint22	R/W1CLR 0x0	DMA Channel 22 Block Transfer Interrupt Bit This interrupt is generated on channel 22 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
21	status_blockint21	R/W1CLR 0x0	DMA Channel 21 Block Transfer Interrupt Bit This interrupt is generated on channel 21 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
20	status_blockint20	R/W1CLR 0x0	DMA Channel 20 Block Transfer Interrupt Bit This interrupt is generated on channel 20 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
19	status_blockint19	R/W1CLR 0x0	DMA Channel 19 Block Transfer Interrupt Bit This interrupt is generated on channel 19 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
18	status_blockint18	R/W1CLR 0x0	DMA Channel 18 Block Transfer Interrupt Bit This interrupt is generated on channel 18 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed

Table 195: DMA Channel BLOCK TRANSFER INTERRUPT Register (STATUS_BLOCKINT)

Bits	Field	Type/ HW Rst	Description
17	status_blockint17	R/W1CLR 0x0	DMA Channel 17 Block Transfer Interrupt Bit This interrupt is generated on channel 17 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
16	status_blockint16	R/W1CLR 0x0	DMA Channel 16 Block Transfer Interrupt Bit This interrupt is generated on channel 16 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
15	status_blockint15	R/W1CLR 0x0	DMA Channel 15 Block Transfer Interrupt Bit This interrupt is generated on channel 15 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
14	status_blockint14	R/W1CLR 0x0	DMA Channel 14 Block Transfer Interrupt Bit This interrupt is generated on channel 14 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
13	status_blockint13	R/W1CLR 0x0	DMA Channel 13 Block Transfer Interrupt Bit This interrupt is generated on channel 13 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
12	status_blockint12	R/W1CLR 0x0	DMA Channel 12 Block Transfer Interrupt Bit This interrupt is generated on channel 12 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
11	status_blockint11	R/W1CLR 0x0	DMA Channel 11 Block Transfer Interrupt Bit This interrupt is generated on channel 11 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
10	status_blockint10	R/W1CLR 0x0	DMA Channel 10 Block Transfer Interrupt Bit This interrupt is generated on channel 10 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed

Table 195: DMA Channel BLOCK TRANSFER INTERRUPT Register (STATUS_BLOCKINT)

Bits	Field	Type/ HW Rst	Description
9	status_blockint9	R/W1CLR 0x0	DMA Channel 9 Block Transfer Interrupt Bit This interrupt is generated on channel 9 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
8	status_blockint8	R/W1CLR 0x0	DMA Channel 8 Block Transfer Interrupt Bit This interrupt is generated on channel 8 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
7	status_blockint7	R/W1CLR 0x0	DMA Channel 7 Block Transfer Interrupt Bit This interrupt is generated on channel 7 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
6	status_blockint6	R/W1CLR 0x0	DMA Channel 6 Block Transfer Interrupt Bit This interrupt is generated on channel 6 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
5	status_blockint5	R/W1CLR 0x0	DMA Channel 5 Block Transfer Interrupt Bit This interrupt is generated on channel 5 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
4	status_blockint4	R/W1CLR 0x0	DMA Channel 4 Block Transfer Interrupt Bit This interrupt is generated on channel 4 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
3	status_blockint3	R/W1CLR 0x0	DMA Channel 3 Block Transfer Interrupt Bit This interrupt is generated on channel 3 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
2	status_blockint2	R/W1CLR 0x0	DMA Channel 2 Block Transfer Interrupt Bit This interrupt is generated on channel 2 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed

Table 195: DMA Channel BLOCK TRANSFER INTERRUPT Register (STATUS_BLOCKINT)

Bits	Field	Type/ HW Rst	Description
1	status_blockint1	R/W1CLR 0x0	DMA Channel 1 Block Transfer Interrupt Bit This interrupt is generated on channel 1 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed
0	status_blockint0	R/W1CLR 0x0	DMA Channel 0 Block Transfer Interrupt Bit This interrupt is generated on channel 0 DMA block burst/single transfer completion. 0x0 = DMA block burst/single transfer is not completed 0x1 = DMA block burst/single transfer is completed

A.2.2.3 DMA Channel Transfer Completion Interrupt Mask Register (MASK_ TFRINT)

Instance Name	Offset
MASK_TFRINT	0x008

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	mask_tfrint31	mask_tfrint30	mask_tfrint29	mask_tfrint28	mask_tfrint27	mask_tfrint26	mask_tfrint25	mask_tfrint24	mask_tfrint23	mask_tfrint22	mask_tfrint21	mask_tfrint20	mask_tfrint19	mask_tfrint18	mask_tfrint17	mask_tfrint16	mask_tfrint15	mask_tfrint14	mask_tfrint13	mask_tfrint12	mask_tfrint11	mask_tfrint10	mask_tfrint9	mask_tfrint8	mask_tfrint7	mask_tfrint6	mask_tfrint5	mask_tfrint4	mask_tfrint3	mask_tfrint2	mask_tfrint1	mask_tfrint0
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 196: DMA Channel Transfer Completion Interrupt Mask Register (MASK_TFRINT)

Bits	Field	Type/ HW Rst	Description
31	mask_tfrint31	R/W 0x0	DMA Channel 31 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT31 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
30	mask_tfrint30	R/W 0x0	DMA Channel 30 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT30 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
29	mask_tfrint29	R/W 0x0	DMA Channel 29 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT29 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt

Table 196: DMA Channel Transfer Completion Interrupt Mask Register (MASK_TFRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
28	mask_tfrint28	R/W 0x0	DMA Channel 28 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT28 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
27	mask_tfrint27	R/W 0x0	DMA Channel 27 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT27 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
26	mask_tfrint26	R/W 0x0	DMA Channel 26 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT26 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
25	mask_tfrint25	R/W 0x0	DMA Channel 25 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT25 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
24	mask_tfrint24	R/W 0x0	DMA Channel 24 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT24 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
23	mask_tfrint23	R/W 0x0	DMA Channel 23 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT23 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
22	mask_tfrint22	R/W 0x0	DMA Channel 22 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT22 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
21	mask_tfrint21	R/W 0x0	DMA Channel 21 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT21 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt

Table 196: DMA Channel Transfer Completion Interrupt Mask Register (MASK_TFRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
20	mask_tfrint20	R/W 0x0	DMA Channel 20 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT20 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
19	mask_tfrint19	R/W 0x0	DMA Channel 19 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT19 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
18	mask_tfrint18	R/W 0x0	DMA Channel 18 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT18 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
17	mask_tfrint17	R/W 0x0	DMA Channel 17 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT17 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
16	mask_tfrint16	R/W 0x0	DMA Channel 16 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT16 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
15	mask_tfrint15	R/W 0x0	DMA Channel 15 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT15 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
14	mask_tfrint14	R/W 0x0	DMA Channel 14 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT14 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
13	mask_tfrint13	R/W 0x0	DMA Channel 13 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT13 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt

Table 196: DMA Channel Transfer Completion Interrupt Mask Register (MASK_TFRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
12	mask_tfrint12	R/W 0x0	DMA Channel 12 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT12 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
11	mask_tfrint11	R/W 0x0	DMA Channel 11 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT11 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
10	mask_tfrint10	R/W 0x0	DMA Channel 10 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT10 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
9	mask_tfrint9	R/W 0x0	DMA Channel 9 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT9 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
8	mask_tfrint8	R/W 0x0	DMA Channel 8 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT8 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
7	mask_tfrint7	R/W 0x0	DMA Channel 7 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT7 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
6	mask_tfrint6	R/W 0x0	DMA Channel 6 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT6 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
5	mask_tfrint5	R/W 0x0	DMA Channel 5 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT5 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt

Table 196: DMA Channel Transfer Completion Interrupt Mask Register (MASK_TFRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
4	mask_tfrint4	R/W 0x0	DMA Channel 4 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT4 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
3	mask_tfrint3	R/W 0x0	DMA Channel 3 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT3 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
2	mask_tfrint2	R/W 0x0	DMA Channel 2 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT2 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
1	mask_tfrint1	R/W 0x0	DMA Channel 1 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT1 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt
0	mask_tfrint0	R/W 0x0	DMA Channel 0 Transfer Completion Interrupt Mask Bit This bit enables the interrupt when STATUS_TFRINT0 bit in STATUS_TFRINT is set. 0x0 = mask the corresponding transfer completion interrupt 0x1 = unmask the corresponding transfer completion interrupt

A.2.2.4 DMA Channel Transfer Completion Interrupt Register (STATUS_TFRINT)

Instance Name	Offset
STATUS_TFRINT	0x00C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	status_tfrint31	status_tfrint30	status_tfrint29	status_tfrint28	status_tfrint27	status_tfrint26	status_tfrint25	status_tfrint24	status_tfrint23	status_tfrint22	status_tfrint21	status_tfrint20	status_tfrint19	status_tfrint18	status_tfrint17	status_tfrint16	status_tfrint15	status_tfrint14	status_tfrint13	status_tfrint12	status_tfrint11	status_tfrint10	status_tfrint9	status_tfrint8	status_tfrint7	status_tfrint6	status_tfrint5	status_tfrint4	status_tfrint3	status_tfrint2	status_tfrint1	status_tfrint0
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 197: DMA Channel Transfer Completion Interrupt Register (STATUS_TFRINT)

Bits	Field	Type/ HW Rst	Description
31	status_tfrint31	R/W1CLR 0x0	DMA Channel 31 Transfer Completion Interrupt This interrupt is generated on DMA channel 31 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
30	status_tfrint30	R/W1CLR 0x0	DMA Channel 30 Transfer Completion Interrupt This interrupt is generated on DMA channel 30 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
29	status_tfrint29	R/W1CLR 0x0	DMA Channel 29 Transfer Completion Interrupt This interrupt is generated on DMA channel 29 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
28	status_tfrint28	R/W1CLR 0x0	DMA Channel 28 Transfer Completion Interrupt This interrupt is generated on DMA channel 28 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
27	status_tfrint27	R/W1CLR 0x0	DMA Channel 27 Transfer Completion Interrupt This interrupt is generated on DMA channel 27 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
26	status_tfrint26	R/W1CLR 0x0	DMA Channel 26 Transfer Completion Interrupt This interrupt is generated on DMA channel 26 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
25	status_tfrint25	R/W1CLR 0x0	DMA Channel 25 Transfer Completion Interrupt This interrupt is generated on DMA channel 25 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed

Table 197: DMA Channel Transfer Completion Interrupt Register (STATUS_TFRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
24	status_tfrint24	R/W1CLR 0x0	DMA Channel 24 Transfer Completion Interrupt This interrupt is generated on DMA channel 24 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
23	status_tfrint23	R/W1CLR 0x0	DMA Channel 23 Transfer Completion Interrupt This interrupt is generated on DMA channel 23 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
22	status_tfrint22	R/W1CLR 0x0	DMA Channel 22 Transfer Completion Interrupt This interrupt is generated on DMA channel 22 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
21	status_tfrint21	R/W1CLR 0x0	DMA Channel 21 Transfer Completion Interrupt This interrupt is generated on DMA channel 21 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
20	status_tfrint20	R/W1CLR 0x0	DMA Channel 20 Transfer Completion Interrupt This interrupt is generated on DMA channel 20 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
19	status_tfrint19	R/W1CLR 0x0	DMA Channel 19 Transfer Completion Interrupt This interrupt is generated on DMA channel 19 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
18	status_tfrint18	R/W1CLR 0x0	DMA Channel 18 Transfer Completion Interrupt This interrupt is generated on DMA channel 18 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
17	status_tfrint17	R/W1CLR 0x0	DMA Channel 17 Transfer Completion Interrupt This interrupt is generated on DMA channel 17 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
16	status_tfrint16	R/W1CLR 0x0	DMA Channel 16 Transfer Completion Interrupt This interrupt is generated on DMA channel 16 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
15	status_tfrint15	R/W1CLR 0x0	DMA Channel 15 Transfer Completion Interrupt This interrupt is generated on DMA channel 15 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed

Table 197: DMA Channel Transfer Completion Interrupt Register (STATUS_TFRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
14	status_tfrint14	R/W1CLR 0x0	DMA Channel 14 Transfer Completion Interrupt This interrupt is generated on DMA channel 14 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
13	status_tfrint13	R/W1CLR 0x0	DMA Channel 13 Transfer Completion Interrupt This interrupt is generated on DMA channel 13 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
12	status_tfrint12	R/W1CLR 0x0	DMA Channel 12 Transfer Completion Interrupt This interrupt is generated on DMA channel 12 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
11	status_tfrint11	R/W1CLR 0x0	DMA Channel 11 Transfer Completion Interrupt This interrupt is generated on DMA channel 11 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
10	status_tfrint10	R/W1CLR 0x0	DMA Channel 10 Transfer Completion Interrupt This interrupt is generated on DMA channel 10 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
9	status_tfrint9	R/W1CLR 0x0	DMA Channel 9 Transfer Completion Interrupt This interrupt is generated on DMA channel 9 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
8	status_tfrint8	R/W1CLR 0x0	DMA Channel 8 Transfer Completion Interrupt This interrupt is generated on DMA channel 8 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
7	status_tfrint7	R/W1CLR 0x0	DMA Channel 7 Transfer Completion Interrupt This interrupt is generated on DMA channel 7 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
6	status_tfrint6	R/W1CLR 0x0	DMA Channel 6 Transfer Completion Interrupt This interrupt is generated on DMA channel 6 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
5	status_tfrint5	R/W1CLR 0x0	DMA Channel 5 Transfer Completion Interrupt This interrupt is generated on DMA channel 5 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed

Table 197: DMA Channel Transfer Completion Interrupt Register (STATUS_TFRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
4	status_tfrint4	R/W1CLR 0x0	DMA Channel 4 Transfer Completion Interrupt This interrupt is generated on DMA channel 4 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
3	status_tfrint3	R/W1CLR 0x0	DMA Channel 3 Transfer Completion Interrupt This interrupt is generated on DMA channel 3 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
2	status_tfrint2	R/W1CLR 0x0	DMA Channel 2 Transfer Completion Interrupt This interrupt is generated on DMA channel 2 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
1	status_tfrint1	R/W1CLR 0x0	DMA Channel 1 Transfer Completion Interrupt This interrupt is generated on DMA channel 1 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed
0	status_tfrint0	R/W1CLR 0x0	DMA Channel 0 Transfer Completion Interrupt This interrupt is generated on DMA channel 0 transfer completion. 0x0 = transfer is not completed 0x1 = transfer is completed

A.2.2.5 DMA Channel Bus Error Interrupt Mask Register (MASK_BUSERRINT)

Instance Name	Offset
MASK_BUSERRINT	0x010

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	mask_buserrint31	mask_buserrint30	mask_buserrint29	mask_buserrint28	mask_buserrint27	mask_buserrint26	mask_buserrint25	mask_buserrint24	mask_buserrint23	mask_buserrint22	mask_buserrint21	mask_buserrint20	mask_buserrint19	mask_buserrint18	mask_buserrint17	mask_buserrint16	mask_buserrint15	mask_buserrint14	mask_buserrint13	mask_buserrint12	mask_buserrint11	mask_buserrint10	mask_buserrint9	mask_buserrint8	mask_buserrint7	mask_buserrint6	mask_buserrint5	mask_buserrint4	mask_buserrint3	mask_buserrint2	mask_buserrint1	mask_buserrint0
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 198: DMA Channel Bus Error Interrupt Mask Register (MASK_BUSERRINT)

Bits	Field	Type/ HW Rst	Description
31	mask_buserrint31	R/W 0x0	DMA Channel 31 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT31 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
30	mask_buserrint30	R/W 0x0	DMA Channel 30 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT30 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
29	mask_buserrint29	R/W 0x0	DMA Channel 29 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT29 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
28	mask_buserrint28	R/W 0x0	DMA Channel 28 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT28 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
27	mask_buserrint27	R/W 0x0	DMA Channel 27 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT27 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
26	mask_buserrint26	R/W 0x0	DMA Channel 26 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT26 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt

Table 198: DMA Channel Bus Error Interrupt Mask Register (MASK_BUSERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
25	mask_buserrint25	R/W 0x0	DMA Channel 25 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT25 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
24	mask_buserrint24	R/W 0x0	DMA Channel 24 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT24 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
23	mask_buserrint23	R/W 0x0	DMA Channel 23 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT23 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
22	mask_buserrint22	R/W 0x0	DMA Channel 22 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT22 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
21	mask_buserrint21	R/W 0x0	DMA Channel 21 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT21 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
20	mask_buserrint20	R/W 0x0	DMA Channel 20 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT20 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
19	mask_buserrint19	R/W 0x0	DMA Channel 19 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT19 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
18	mask_buserrint18	R/W 0x0	DMA Channel 18 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT18 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt

Table 198: DMA Channel Bus Error Interrupt Mask Register (MASK_BUSERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
17	mask_buserrint17	R/W 0x0	DMA Channel 17 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT17 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
16	mask_buserrint16	R/W 0x0	DMA Channel 16 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT16 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
15	mask_buserrint15	R/W 0x0	DMA Channel 15 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT15 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
14	mask_buserrint14	R/W 0x0	DMA Channel 14 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT14 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
13	mask_buserrint13	R/W 0x0	DMA Channel 13 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT13 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
12	mask_buserrint12	R/W 0x0	DMA Channel 12 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT12 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
11	mask_buserrint11	R/W 0x0	DMA Channel 11 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT11 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
10	mask_buserrint10	R/W 0x0	DMA Channel 10 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT10 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt

Table 198: DMA Channel Bus Error Interrupt Mask Register (MASK_BUSERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
9	mask_buserrint9	R/W 0x0	DMA Channel 9 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT9 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
8	mask_buserrint8	R/W 0x0	DMA Channel 8 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT8 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
7	mask_buserrint7	R/W 0x0	DMA Channel 7 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT7 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
6	mask_buserrint6	R/W 0x0	DMA Channel 6 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT6 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
5	mask_buserrint5	R/W 0x0	DMA Channel 5 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT5 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
4	mask_buserrint4	R/W 0x0	DMA Channel 4 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT4 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
3	mask_buserrint3	R/W 0x0	DMA Channel 3 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT3 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
2	mask_buserrint2	R/W 0x0	DMA Channel 2 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT2 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt

Table 198: DMA Channel Bus Error Interrupt Mask Register (MASK_BUSERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
1	mask_buserrint1	R/W 0x0	DMA Channel 1 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT1 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt
0	mask_buserrint0	R/W 0x0	DMA Channel 0 Bus Error Interrupt Mask Bit This bit enables the interrupt when STATUS_BUSERRINT0 bit in STATUS_ERRINT is set. 0x0 = mask the corresponding bus error interrupt 0x1 = unmask the corresponding bus error interrupt

A.2.2.6 DMA Channel Bus Error Interrupt Mask Register (STATUS_ BUSERRINT)

Instance Name	Offset
STATUS_BUSERRINT	0x014

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	status_buserrint31	status_buserrint30	status_buserrint29	status_buserrint28	status_buserrint27	status_buserrint26	status_buserrint25	status_buserrint24	status_buserrint23	status_buserrint22	status_buserrint21	status_buserrint20	status_buserrint19	status_buserrint18	status_buserrint17	status_buserrint16	status_buserrint15	status_buserrint14	status_buserrint13	status_buserrint12	status_buserrint11	status_buserrint10	status_buserrint9	status_buserrint8	status_buserrint7	status_buserrint6	status_buserrint5	status_buserrint4	status_buserrint3	status_buserrint2	status_buserrint1	status_buserrint0
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 199: DMA Channel Bus Error Interrupt Mask Register (STATUS_BUSERRINT)

Bits	Field	Type/ HW Rst	Description
31	status_buserrint31	R/W1CLR 0x0	DMA Channel 31 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel31 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
30	status_buserrint30	R/W1CLR 0x0	DMA Channel 30 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel30 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated

Table 199: DMA Channel Bus Error Interrupt Mask Register (STATUS_BUSERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
29	status_buserrint29	R/W1CLR 0x0	DMA Channel 29 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel29 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
28	status_buserrint28	R/W1CLR 0x0	DMA Channel 28 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel28 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
27	status_buserrint27	R/W1CLR 0x0	DMA Channel 27 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel27 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
26	status_buserrint26	R/W1CLR 0x0	DMA Channel 26 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel26 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
25	status_buserrint25	R/W1CLR 0x0	DMA Channel 25 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel25 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
24	status_buserrint24	R/W1CLR 0x0	DMA Channel 24 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel24 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
23	status_buserrint23	R/W1CLR 0x0	DMA Channel 23 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel23 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated

Table 199: DMA Channel Bus Error Interrupt Mask Register (STATUS_BUSERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
22	status_buserrint22	R/W1CLR 0x0	DMA Channel 22 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel22 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
21	status_buserrint21	R/W1CLR 0x0	DMA Channel 21 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel21 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
20	status_buserrint20	R/W1CLR 0x0	DMA Channel 20 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel20 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
19	status_buserrint19	R/W1CLR 0x0	DMA Channel 19 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel19 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
18	status_buserrint18	R/W1CLR 0x0	DMA Channel 18 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel18 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
17	status_buserrint17	R/W1CLR 0x0	DMA Channel 17 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel17 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
16	status_buserrint16	R/W1CLR 0x0	DMA Channel 16 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel16 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated

Table 199: DMA Channel Bus Error Interrupt Mask Register (STATUS_BUSERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
15	status_buserrint15	R/W1CLR 0x0	DMA Channel 15 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel15 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
14	status_buserrint14	R/W1CLR 0x0	DMA Channel 14 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel14 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
13	status_buserrint13	R/W1CLR 0x0	DMA Channel 13 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel13 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
12	status_buserrint12	R/W1CLR 0x0	DMA Channel 12 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel12 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
11	status_buserrint11	R/W1CLR 0x0	DMA Channel 11 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel11 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
10	status_buserrint10	R/W1CLR 0x0	DMA Channel 10 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel10 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
9	status_buserrint9	R/W1CLR 0x0	DMA Channel 9 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel9 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated

Table 199: DMA Channel Bus Error Interrupt Mask Register (STATUS_BUSERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
8	status_buserrint8	R/W1CLR 0x0	DMA Channel 8 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel8 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
7	status_buserrint7	R/W1CLR 0x0	DMA Channel 7 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel7 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
6	status_buserrint6	R/W1CLR 0x0	DMA Channel 6 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel6 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
5	status_buserrint5	R/W1CLR 0x0	DMA Channel 5 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel5 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
4	status_buserrint4	R/W1CLR 0x0	DMA Channel 4 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel4 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
3	status_buserrint3	R/W1CLR 0x0	DMA Channel 3 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel3 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
2	status_buserrint2	R/W1CLR 0x0	DMA Channel 2 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel2 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated

Table 199: DMA Channel Bus Error Interrupt Mask Register (STATUS_BUSERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
1	status_buserrint1	R/W1CLR 0x0	DMA Channel 1 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel1 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated
0	status_buserrint0	R/W1CLR 0x0	DMA Channel 0 Bus Error Interrupt Bit This interrupt is generated when an ERROR response is received from AHB slave on the HRESP bus during a DMA channel0 transfer. In addition, the channel is disabled. 0x0 = no bus error interrupt is generated 0x1 = bus error interrupt is generated

A.2.2.7 DMA Channel Source/target Address Alignment Error Interrupt Mask Register (MASK_ADDRERRINT)

Instance Name	Offset
MASK_ADDRERRINT	0x018

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	mask_addrerrint31	mask_addrerrint30	mask_addrerrint29	mask_addrerrint28	mask_addrerrint27	mask_addrerrint26	mask_addrerrint25	mask_addrerrint24	mask_addrerrint23	mask_addrerrint22	mask_addrerrint21	mask_addrerrint20	mask_addrerrint19	mask_addrerrint18	mask_addrerrint17	mask_addrerrint16	mask_addrerrint15	mask_addrerrint14	mask_addrerrint13	mask_addrerrint12	mask_addrerrint11	mask_addrerrint10	mask_addrerrint9	mask_addrerrint8	mask_addrerrint7	mask_addrerrint6	mask_addrerrint5	mask_addrerrint4	mask_addrerrint3	mask_addrerrint2	mask_addrerrint1	mask_addrerrint0
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 200: DMA Channel Source/target Address Alignment Error Interrupt Mask Register (MASK_ ADDRERRINT)

Bits	Field	Type/ HW Rst	Description
31	mask_addrerrint31	R/W 0x0	DMA Channel 31 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR31 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt

Table 200: DMA Channel Source/target Address Alignment Error Interrupt Mask Register (MASK_ADDRERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
30	mask_addrerrint30	R/W 0x0	DMA Channel 30 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR30 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
29	mask_addrerrint29	R/W 0x0	DMA Channel 29 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR29 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
28	mask_addrerrint28	R/W 0x0	DMA Channel 28 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR28 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
27	mask_addrerrint27	R/W 0x0	DMA Channel 27 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR27 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
26	mask_addrerrint26	R/W 0x0	DMA Channel 26 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR26 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
25	mask_addrerrint25	R/W 0x0	DMA Channel 25 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR25 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
24	mask_addrerrint24	R/W 0x0	DMA Channel 24 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR24 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt

Table 200: DMA Channel Source/target Address Alignment Error Interrupt Mask Register (MASK_ADDRERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
23	mask_addrerrint23	R/W 0x0	DMA Channel 23 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR23 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
22	mask_addrerrint22	R/W 0x0	DMA Channel 22 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR22 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
21	mask_addrerrint21	R/W 0x0	DMA Channel 21 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR21 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
20	mask_addrerrint20	R/W 0x0	DMA Channel 20 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR20 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
19	mask_addrerrint19	R/W 0x0	DMA Channel 19 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR19 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
18	mask_addrerrint18	R/W 0x0	DMA Channel 18 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR18 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
17	mask_addrerrint17	R/W 0x0	DMA Channel 17 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR17 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt

Table 200: DMA Channel Source/target Address Alignment Error Interrupt Mask Register (MASK_ADDRERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
16	mask_addrerrint16	R/W 0x0	DMA Channel 16 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR16 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
15	mask_addrerrint15	R/W 0x0	DMA Channel 15 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR15 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
14	mask_addrerrint14	R/W 0x0	DMA Channel 14 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR14 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
13	mask_addrerrint13	R/W 0x0	DMA Channel 13 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR13 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
12	mask_addrerrint12	R/W 0x0	DMA Channel 12 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR12 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
11	mask_addrerrint11	R/W 0x0	DMA Channel 11 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR11 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
10	mask_addrerrint10	R/W 0x0	DMA Channel 10 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR10 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt

Table 200: DMA Channel Source/target Address Alignment Error Interrupt Mask Register (MASK_ADDRERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
9	mask_addrerrint9	R/W 0x0	DMA Channel 9 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR9 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
8	mask_addrerrint8	R/W 0x0	DMA Channel 8 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR8 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
7	mask_addrerrint7	R/W 0x0	DMA Channel 7 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR7 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
6	mask_addrerrint6	R/W 0x0	DMA Channel 6 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR6 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
5	mask_addrerrint5	R/W 0x0	DMA Channel 5 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR5 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
4	mask_addrerrint4	R/W 0x0	DMA Channel 4 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR4 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
3	mask_addrerrint3	R/W 0x0	DMA Channel 3 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR3 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt

Table 200: DMA Channel Source/target Address Alignment Error Interrupt Mask Register (MASK_ADDRERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
2	mask_addrerrint2	R/W 0x0	DMA Channel 2 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR2 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
1	mask_addrerrint1	R/W 0x0	DMA Channel 1 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR1 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt
0	mask_addrerrint0	R/W 0x0	DMA Channel 0 Source/target Address Alignment Error Interrupt Mask Bit This bit enables the interrupt when STATUS_ADDRERR0 bit in STATUS_ADDRERR is set. 0x0 = mask the corresponding address error interrupt 0x1 = unmask the corresponding address error interrupt

A.2.2.8 DMA Channel Source/target Address Alignment Error Interrupt Register (STATUS_ADDRERRINT)

Instance Name	Offset
STATUS_ADDRERRINT	0x01C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	status_addrerrint31	status_addrerrint30	status_addrerrint29	status_addrerrint28	status_addrerrint27	status_addrerrint26	status_addrerrint25	status_addrerrint24	status_addrerrint23	status_addrerrint22	status_addrerrint21	status_addrerrint20	status_addrerrint19	status_addrerrint18	status_addrerrint17	status_addrerrint16	status_addrerrint15	status_addrerrint14	status_addrerrint13	status_addrerrint12	status_addrerrint11	status_addrerrint10	status_addrerrint9	status_addrerrint8	status_addrerrint7	status_addrerrint6	status_addrerrint5	status_addrerrint4	status_addrerrint3	status_addrerrint2	status_addrerrint1	status_addrerrint0
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 201: DMA Channel Source/target Address Alignment Error Interrupt Register (STATUS_ ADDRERRINT)

Bits	Field	Type/ HW Rst	Description
31	status_addrerrint31	R/W1CLR 0x0	DMA Channel 31 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 31 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
30	status_addrerrint30	R/W1CLR 0x0	DMA Channel 30 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 30 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
29	status_addrerrint29	R/W1CLR 0x0	DMA Channel 29 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 29 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
28	status_addrerrint28	R/W1CLR 0x0	DMA Channel 28 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 28 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
27	status_addrerrint27	R/W1CLR 0x0	DMA Channel 27 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 27 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated

Table 201: DMA Channel Source/target Address Alignment Error Interrupt Register (STATUS_ADDRERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
26	status_addrerrint26	R/W1CLR 0x0	DMA Channel 26 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 26 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
25	status_addrerrint25	R/W1CLR 0x0	DMA Channel 25 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 25 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
24	status_addrerrint24	R/W1CLR 0x0	DMA Channel 24 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 24 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
23	status_addrerrint23	R/W1CLR 0x0	DMA Channel 23 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 23 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
22	status_addrerrint22	R/W1CLR 0x0	DMA Channel 22 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 22 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
21	status_addrerrint21	R/W1CLR 0x0	DMA Channel 21 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 21 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
20	status_addrerrint20	R/W1CLR 0x0	DMA Channel 20 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 20 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
19	status_addrerrint19	R/W1CLR 0x0	DMA Channel 19 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 19 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated

Table 201: DMA Channel Source/target Address Alignment Error Interrupt Register (STATUS_ ADDRERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
18	status_addrerrint18	R/W1CLR 0x0	DMA Channel 18 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 18 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
17	status_addrerrint17	R/W1CLR 0x0	DMA Channel 17 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 17 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
16	status_addrerrint16	R/W1CLR 0x0	DMA Channel 16 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 16 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
15	status_addrerrint15	R/W1CLR 0x0	DMA Channel 15 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 15 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
14	status_addrerrint14	R/W1CLR 0x0	DMA Channel 14 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 14 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
13	status_addrerrint13	R/W1CLR 0x0	DMA Channel 13 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 13 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
12	status_addrerrint12	R/W1CLR 0x0	DMA Channel 12 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 12 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
11	status_addrerrint11	R/W1CLR 0x0	DMA Channel 11 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 11 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated

Table 201: DMA Channel Source/target Address Alignment Error Interrupt Register (STATUS_ADDRERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
10	status_addrerrint10	R/W1CLR 0x0	DMA Channel 10 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 10 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
9	status_addrerrint9	R/W1CLR 0x0	DMA Channel 9 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 9 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
8	status_addrerrint8	R/W1CLR 0x0	DMA Channel 8 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 8 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
7	status_addrerrint7	R/W1CLR 0x0	DMA Channel 7 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 7 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
6	status_addrerrint6	R/W1CLR 0x0	DMA Channel 6 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 6 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
5	status_addrerrint5	R/W1CLR 0x0	DMA Channel 5 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 5 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
4	status_addrerrint4	R/W1CLR 0x0	DMA Channel 4 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 4 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
3	status_addrerrint3	R/W1CLR 0x0	DMA Channel 3 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 3 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated

Table 201: DMA Channel Source/target Address Alignment Error Interrupt Register (STATUS_ ADDRERRINT) (Continued)

Bits	Field	Type/ HW Rst	Description
2	status_addrerrint2	R/W1CLR 0x0	DMA Channel 2 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 2 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
1	status_addrerrint1	R/W1CLR 0x0	DMA Channel 1 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 1 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated
0	status_addrerrint0	R/W1CLR 0x0	DMA Channel 0 Source/target Address Alignment Error Interrupt Bit This interrupt is generated when channel 0 source or target address is not aligned to corresponding cntl.width. 0x0 = no address error interrupt is generated 0x1 = address error interrupt is generated

A.2.2.9 DMA CHANNEL INTERRUPT REGISTER (STATUS_CHLINT)

Instance Name	Offset
STATUS_CHLINT	0x020

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	status_chlint31	status_chlint30	status_chlint29	status_chlint28	status_chlint27	status_chlint26	status_chlint25	status_chlint24	status_chlint23	status_chlint22	status_chlint21	status_chlint20	status_chlint19	status_chlint18	status_chlint17	status_chlint16	status_chlint15	status_chlint14	status_chlint13	status_chlint12	status_chlint11	status_chlint10	status_chlint9	status_chlint8	status_chlint7	status_chlint6	status_chlint5	status_chlint4	status_chlint3	status_chlint2	status_chlint1	status_chlint0
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 202: DMA CHANNEL INTERRUPT REGISTER (STATUS_CHLINT)

Bits	Field	Type/ HW Rst	Description
31	status_chlint31	R 0x0	DMA Channel 31 Interrupt OR of the content of the respective unmapped interrupt of channel 31.
30	status_chlint30	R 0x0	DMA Channel 30 Interrupt OR of the content of the respective unmapped interrupt of channel 30.
29	status_chlint29	R 0x0	DMA Channel 29 Interrupt OR of the content of the respective unmapped interrupt of channel 29.

Table 202: DMA CHANNEL INTERRUPT REGISTER (STATUS_CHLINT) (Continued)

Bits	Field	Type/ HW Rst	Description
28	status_chlint28	R 0x0	DMA Channel 28 Interrupt OR of the content of the respective unmapped interrupt of channel 28.
27	status_chlint27	R 0x0	DMA Channel 27 Interrupt OR of the content of the respective unmapped interrupt of channel 27.
26	status_chlint26	R 0x0	DMA Channel 26 Interrupt OR of the content of the respective unmapped interrupt of channel 26.
25	status_chlint25	R 0x0	DMA Channel 25 Interrupt OR of the content of the respective unmapped interrupt of channel 25.
24	status_chlint24	R 0x0	DMA Channel 24 Interrupt OR of the content of the respective unmapped interrupt of channel 24.
23	status_chlint23	R 0x0	DMA Channel 23 Interrupt OR of the content of the respective unmapped interrupt of channel 23.
22	status_chlint22	R 0x0	DMA Channel 22 Interrupt OR of the content of the respective unmapped interrupt of channel 22
21	status_chlint21	R 0x0	DMA Channel 21 Interrupt OR of the content of the respective unmapped interrupt of channel 21.
20	status_chlint20	R 0x0	DMA Channel 20 Interrupt OR of the content of the respective unmapped interrupt of channel 20.
19	status_chlint19	R 0x0	DMA Channel 19 Interrupt OR of the content of the respective unmapped interrupt of channel 19.
18	status_chlint18	R 0x0	DMA Channel 18 Interrupt OR of the content of the respective unmapped interrupt of channel 18.
17	status_chlint17	R 0x0	DMA Channel 17 Interrupt OR of the content of the respective unmapped interrupt of channel 17.
16	status_chlint16	R 0x0	DMA Channel 16 Interrupt OR of the content of the respective unmapped interrupt of channel 16.
15	status_chlint15	R 0x0	DMA Channel 15 Interrupt OR of the content of the respective unmapped interrupt of channel 15.

Table 202: DMA CHANNEL INTERRUPT REGISTER (STATUS_CHLINT) (Continued)

Bits	Field	Type/ HW Rst	Description
14	status_chlint14	R 0x0	DMA Channel 14 Interrupt OR of the content of the respective unmapped interrupt of channel 14.
13	status_chlint13	R 0x0	DMA Channel 13 Interrupt OR of the content of the respective unmapped interrupt of channel 13.
12	status_chlint12	R 0x0	DMA Channel 12 Interrupt OR of the content of the respective unmapped interrupt of channel 12.
11	status_chlint11	R 0x0	DMA Channel 11 Interrupt OR of the content of the respective unmapped interrupt of channel 11.
10	status_chlint10	R 0x0	DMA Channel 10 Interrupt OR of the content of the respective unmapped interrupt of channel 10.
9	status_chlint9	R 0x0	DMA Channel 9 Interrupt OR of the content of the respective unmapped interrupt of channel 9.
8	status_chlint8	R 0x0	DMA Channel 8 Interrupt OR of the content of the respective unmapped interrupt of channel 8.
7	status_chlint7	R 0x0	DMA Channel 7 Interrupt OR of the content of the respective unmapped interrupt of channel 7.
6	status_chlint6	R 0x0	DMA Channel 6 Interrupt OR of the content of the respective unmapped interrupt of channel 6.
5	status_chlint5	R 0x0	DMA Channel 5 Interrupt OR of the content of the respective unmapped interrupt of channel 5.
4	status_chlint4	R 0x0	DMA Channel 4 Interrupt OR of the content of the respective unmapped interrupt of channel 4.
3	status_chlint3	R 0x0	DMA Channel 3 Interrupt OR of the content of the respective unmapped interrupt of channel 3.
2	status_chlint2	R 0x0	DMA Channel 2 Interrupt OR of the content of the respective unmapped interrupt of channel 2.
1	status_chlint1	R 0x0	DMA Channel 1 Interrupt OR of the content of the respective unmapped interrupt of channel 1.
0	status_chlint0	R 0x0	DMA Channel 0 Interrupt OR of the content of the respective unmapped interrupt of channel 0.

A.2.2.10 THE PROTECTION CONTROL SIGNALS Register (HPROT)

Instance Name	Offset
HPROT	0x080

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

Table 203: THE PROTECTION CONTROL SIGNALS Register (HPROT)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3:0	hprot	R/W 0x3	Protection Control Signals

A.2.2.11 DMA SOURCE ADDRESS Register (SADR)

Instance Name	Offset	Count
SADR0	0x100	32

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	srcaddr																														srcaddr0	
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 204: DMA SOURCE ADDRESS Register (SADR)

Bits	Field	Type/ HW Rst	Description
31:2	srcaddr	R/W 0x0	SOURCE ADDRESS
1:0	srcaddr0	R/W 0x0	SRCADDR0 SRCADDR[1:0] should align to the cmd.width, that is SRCADDR[1:0] should be 2'b00 when the CMDx.width is configured as word, SRCADDR[0] should be 1'b0 when CMDx.width is configured as half-word, if the address is not aligned to the width, the error interrupt will be generated.

A.2.2.12 DMA TARGET ADDRESS Register (TADR)

Instance Name	Offset	Count
TADR0	0x104	32

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	trgaddr																														trgaddr0	
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 205: DMA TARGET ADDRESS Register (TADR)

Bits	Field	Type/ HW Rst	Description
31:2	trgaddr	R/W 0x0	TARGET ADDRESS
1:0	trgaddr0	R/W 0x0	TRGADDR0 TRGADDR[1:0] should align to the cmd.width, that is TRGADDR[1:0] should be 2'b00 when the CMDx.width is configured as word, TRGADDR[0] should be 1'b0 when CMDx.width is configured as half-word, if the address is not aligned to the width, the error interrupt will be generated.

A.2.2.13 DMA CONTROL Register A (CTRLA)

Instance Name	Offset	Count
CTRLA0	0x108	32

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved	incsrcaddr	inctrgaddr	Reserved										tran_type		tran_size		width		len													
HW Rst	?	0	0	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 206: DMA CONTROL Register A (CTRLA)

Bits	Field	Type/ HW Rst	Description
31	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

Table 206: DMA CONTROL Register A (CTRLA) (Continued)

Bits	Field	Type/ HW Rst	Description
30	incsrcaddr	R/W 0x0	Source Address Increment If the source address is an internal peripheral FIFO address or external I/O address, the address is not incremented on each successive access. In these cases, DCMDx.INCSRCADDR must be cleared. 0x0 = do not increment source address 0x1 = stop the running channel
29	inctrgaddr	R/W 0x0	Target Address Increment If the target address is an internal peripheral FIFO address or external I/O address, the address is not incremented on each successive access. In these cases, DCMDx.INCTRGADDR must be cleared. 0x0 = do not increment target address 0x1 = increment target address
28:19	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
18:17	tran_type	R/W 0x0	Source to Target Transfer Type TRAN TYPE, indicate the source and target type. 0x0 = M2M 0x1 = M2P 0x2 = P2M 0x3 = reserved
16:15	tran_size	R/W 0x0	Size Maximum burst transaction length, number of data items, each of cmd.width. 0x0 = 1 0x1 = 4 0x2 = 8 0x3 = 16
14:13	width	R/W 0x0	Width Width of the on-chip peripheral when the source or target is on-chip peripheral, width of the memory when the transfer is M2M type. 0x0 = reserved for on-chip peripheral-related transactions (1 byte) 0x1 = 1 byte 0x2 = half-word (2 bytes) 0x3 = word (4 bytes)
12:0	len	R/W 0x0	Length of the Transfer in Bytes The maximum transfer length is (8K-1) bytes. If the transfer is memory-to-memory type, the length of the transfer may be any value up to a maximum of (8K-1) bytes. If the transfer involves any of the on-chip peripherals, the length must be an inter multiple of the peripheral sample width DCMDx.WIDTH.

A.2.2.14 DMA CONTROL Register B (CTRLB)

Instance Name	Offset	Count
CTRLB0	0x10C	32

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

Table 207: DMA CONTROL Register B (CTRLB)

Bits	Field	Type/ HW Rst	Description
31:6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
5:0	pernum	R/W 0x0	Peripheral Number Indicates the valid peripheral request number. Do not map the same peripheral requests to 2 or more active channel since it produces unpredictable results.

A.2.2.15 DMA CHANNEL ENABLE Register (CHL_EN)

Instance Name	Offset	Count
CHL_EN0	0x110	32

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

Table 208: DMA CHANNEL ENABLE Register (CHL_EN)

Bits	Field	Type/ HW Rst	Description
31	chl_en	R/W 0x0	Enable/Disable the Channel This bit allows software to start the channel. To stop an enabled channel, see the chl_stop bit. This bit is reset as soon as it is cleared and when the channel stops normally. After the channel stops normally, STATUS_TFRINTR is set. 0x0 = disable the channel 0x1 = enable the channel
30:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.2.2.16 DMA CHANNEL STOP Register (CHL_STOP)

Instance Name	Offset	Count
CHL_STOP0	0x114	32

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

Table 209: DMA CHANNEL STOP Register (CHL_STOP)

Bits	Field	Type/ HW Rst	Description
31	chl_stop	R/W 0x0	Stop the Running Channel This bit allows software to stop the channel when the channel is enabled and has not finished all the transfer yet. When the channel is already in disable state, the stop bit is meaningless and the write can be successful. 0x0 = no impact on the channel 0x1 = stop the running channel
30:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.2.2.17 DMA ACK DELAY CYCLE for Single Transfer in M2P Transfer Type Register (ACK_DELAY)

Instance Name	Offset
ACK_DELAY	0x800

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

Table 210: DMA ACK DELAY CYCLE for Single Transfer in M2P Transfer Type Register (ACK_ DELAY)

Bits	Field	Type/ HW Rst	Description
31:10	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
9:0	ack_delay_num	R/W 0xC	DMA ACK Delay Cycle for Single Write Transaction to Peripheral This field indicates the delay cycles for a single write transaction to peripheral.

A.2.2.18 DMA ERROR INFORMATION REGISTER 0 (ERR_INFO0)

Instance Name	Offset
ERR_INFO0	0x900

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

Table 211: DMA ERROR INFORMATION REGISTER 0 (ERR_INFO0)

Bits	Field	Type/ HW Rst	Description
31:0	err_addr	R 0x0	ADDRESS INFORMATION RELATED ERROR

A.2.2.19 DMA ERROR INFORMATION REGISTER 1 (ERR_INFO1)

Instance Name	Offset
ERR_INFO1	0x904

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

Table 212: DMA ERROR INFORMATION REGISTER 1 (ERR_INFO1)

Bits	Field	Type/ HW Rst	Description
31:27	err_chlnum	R 0x0	Channel Id Information Related Error
26:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.2.2.20 DMA DIAGNOSE INFORMATION REGISTER 0 (DIAGNOSE_INFO0)

Instance Name	Offset
DIAGNOSE_INFO0	0x908

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

Table 213: DMA DIAGNOSE INFORMATION REGISTER 0 (DIAGNOSE_INFO0)

Bits	Field	Type/ HW Rst	Description
31:0	diagnose_addr	R 0x0	Address Information Related Diagnose

A.2.2.21 DMA DIAGNOSE INFORMATION REGISTER 1 (DIAGNOSE_INFO1)

Instance Name	Offset
DIAGNOSE_INFO1	0x90C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	diagnose_req_chl_data	diagnose_rest_len													Reserved													diagnose_req_chl_data_chlnum				
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0

Table 214: DMA DIAGNOSE INFORMATION REGISTER 1 (DIAGNOSE_INFO1)

Bits	Field	Type/ HW Rst	Description
31	diagnose_req_chl_data	R 0x0	Indicate Whether There is a Valid Request
30:18	diagnose_rest_len	R 0x0	Indicate the Remaining Data Length of the Selected Channel
17:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4:0	diagnose_req_chl_data_chlnum	R 0x0	Indicate Which Channel is in Service

A.2.2.22 DMA DIAGNOSE INFORMATION REGISTER 2 (DIAGNOSE_INFO2)

Instance Name	Offset
DIAGNOSE_INFO2	0x910

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

Table 215: DMA DIAGNOSE INFORMATION REGISTER 2 (DIAGNOSE_INFO2)

Bits	Field	Type/ HW Rst	Description
31:28	diagnose_chl_state	R 0x0	Indicate the Channel State
27:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.2.2.23 DMA DIAGNOSE INFORMATION REGISTER 3 (DIAGNOSE_INFO3)

Instance Name	Offset
DIAGNOSE_INFO3	0x914

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	diagnose_src_addr																														diagnose_src_addr0	
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 216: DMA DIAGNOSE INFORMATION REGISTER 3 (DIAGNOSE_INFO3)

Bits	Field	Type/ HW Rst	Description
31:2	diagnose_src_addr	R 0x0	Indicate the Source Address of the Selected Channel
1:0	diagnose_src_addr0	R 0x0	Indicate The Source Address 0 of the Selected Channel

A.2.2.24 DMA DIAGNOSE INFORMATION REGISTER 4 (DIAGNOSE_INFO4)

Instance Name	Offset
DIAGNOSE_INFO4	0x918

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	diagnose_trg_addr																														diagnose_trg_addr0	
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 217: DMA DIAGNOSE INFORMATION REGISTER 4 (DIAGNOSE_INFO4)

Bits	Field	Type/ HW Rst	Description
31:2	diagnose_trg_addr	R 0x0	Indicate the Target Address of the Selected Channel
1:0	diagnose_trg_addr0	R 0x0	Indicate the Target Address 0 of the Selected Channel

A.2.2.25 DMA DIAGNOSE INFORMATION REGISTER 5 (DIAGNOSE_INFO5)

Instance Name	Offset
DIAGNOSE_INFO5	0x91C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	diagnose_chl_en	diagnose_incrsrcaddr	diagnose_incrtrgaddr	diagnose_ctrl_len													diagnose_ctrl_transize		diagnose_ctrl_trantype		diagnose_ctrl_width		Reserved									
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	?	?	?	?	?	?	?	?	?	?

Table 218: DMA DIAGNOSE INFORMATION REGISTER 5 (DIAGNOSE_INFO5)

Bits	Field	Type/ HW Rst	Description
31	diagnose_chl_en	R 0x0	Indicate the Channel Enable State of the Selected Channel
30	diagnose_incrsrcaddr	R 0x0	Indicate Whether to Increase the Src Address of The Selected Channel
29	diagnose_incrtrgaddr	R 0x0	Indicate Whether to Increase the Trg Address of The Selected Channel
28:16	diagnose_ctrl_len	R 0x0	Indicate the Length Information of the Selected Channel
15:14	diagnose_ctrl_transize	R 0x0	Indicate the Transfer Size Information of the Selected Channel
13:12	diagnose_ctrl_trantype	R 0x0	Indicate the Transfer Type Information of the Selected Channel
11:10	diagnose_ctrl_width	R 0x0	Indicate the Width Information of the Selected Channel
9:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.2.2.26 DMA DIAGNOSE INFORMATION REGISTER 6 (DIAGNOSE_INFO6)

Instance Name	Offset
DIAGNOSE_INFO6	0x920

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	diagnose_mas_read	diagnose_mas_write	diagnose_ahb_size		diagnose_ahb_burst		diagnose_slicecnt							diagnose_split_word	diagnose_split_halfword	Reserved																	
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Table 219: DMA DIAGNOSE INFORMATION REGISTER 6 (DIAGNOSE_INFO6)

Bits	Field	Type/ HW Rst	Description
31	diagnose_mas_read	R 0x0	Indicate Some Output Info From Main Datapath
30	diagnose_mas_write	R 0x0	Indicate Some Output Info From Main Datapath
29:28	diagnose_ahb_size	R 0x0	Indicate Some Output Info From Main Datapath
27:26	diagnose_ahb_burst	R 0x0	Indicate Some Output Info From Main Datapath
25:19	diagnose_slicecnt	R 0x0	Indicate Some Output Info From Main Datapath
18	diagnose_split_word	R 0x0	Indicate Some Output Info From Main Datapath
17	diagnose_split_halfword	R 0x0	Indicate Some Output Info From Main Datapath
16:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.2.2.27 DMA DIAGNOSE INFORMATION REGISTER 7 (DIAGNOSE_INFO7)

Instance Name	Offset
DIAGNOSE_INFO7	0x924

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	diagnose_ahb_burst			diagnose_ahb_trans		diagnose_ahb_size			diagnose_ahb_hready	diagnose_ahb_hresp	Reserved																					
HW Rst	0	0	0	0	0	0	0	0	0	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Table 220: DMA DIAGNOSE INFORMATION REGISTER 7 (DIAGNOSE_INFO7)

Bits	Field	Type/ HW Rst	Description
31:29	diagnose_ahb_burst	R 0x0	Indicate Some AHB Master Info
28:27	diagnose_ahb_trans	R 0x0	Indicate Some AHB Master Info
26:24	diagnose_ahb_size	R 0x0	Indicate Some AHB Master Info
23	diagnose_ahb_hready	R 0x0	Indicate Some AHB Master Info
22	diagnose_ahb_hresp	R 0x0	Indicate Some AHB Master Info
21:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

THIS PAGE INTENTIONALLY LEFT BLANK

A.3 USBC Address Block

A.3.1 USBC Register Map

Table 221: USBC Register Map

Offset	Name	HW Rst	Description	Details
0x000	ID	0xE401_3A05	ID Register	Page: 416
0x004	HWGENERAL	0x0000_0000	HW General Register	Page: 417
0x008	HWHOST	0x0000_0000	HW Host Register	Page: 418
0x00C	HWDEVICE	0x0000_0002	HW Device Register	Page: 418
0x010	HWTXBUF	0x0000_0000	HW TXBUF Register	Page: 419
0x014	HWRXBUF	0x0000_0000	HW RXBUF Register	Page: 419
0x018	HWTXBUF0	0x0000_7777	HW TXBUF0 Register	Page: 420
0x01C	HWTXBUF1	0x0000_7777	HW TXBUF1 Register	Page: 420
0x080	GPTIMER0LD	0x0000_0000	GPTIMER0LD Register	Page: 421
0x084	GPTIMER0CTRL	0x0000_0000	GPTIMER0CTRL	Page: 421
0x088	GPTTIMER1LD	0x0000_0000	GPTTIMER1LD Register	Page: 422
0x08C	GPTIMER1CTRL	0x0000_0000	GP Timer1 Control Register	Page: 422
0x090	SBUSCFG	0x0000_0000	SBUS Config Register	Page: 423
0x100	CAPLENGTH	0x0100_0040	Cap Length Regiser	Page: 423
0x104	HCSPARAMS	0x0001_0000	HCS Params Register	Page: 424
0x108	HCCPARAMS	0x0000_0006	HCC Params Register	Page: 425
0x120	DCIVERSION	0x0000_0001	DCI Version Register	Page: 426
0x124	DCCPARAMS	0x8000_0000	DCC Params Register	Page: 426
0x128	DevLPMCSR	0x0001_F000	DevLPMCSR Register	Page: 427
0x140	USBCMD	0x0008_0B00	USB Command Register	Page: 429
0x144	USBSTS	0x0000_1000	USB STS Register	Page: 430
0x148	USBINTR	0x0000_0000	USB Interrupt Register	Page: 432
0x14C	FRINDEX	0x0000_0000	FR Index Register	Page: 433
0x154	PERIODICLISTBASE	0x0000_0000	Periodic List Base Register	Page: 434
0x158	ASYNCLISTADDR	0x0000_0000	Async List Address Register	Page: 434
0x15C	TTCTRL Register	0x0000_0000	TT Control Register	Page: 435
0x160	BURSTSIZE	0x0000_0000	Burst Size Register	Page: 435

Table 221: USBC Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x164	TXFILLTUNING	0x0002_0000	TX Fill Tuning Register	Page: 436
0x168	TXTTFILLTUNING	0x0000_0000	TX TT Fill Tuning Register	Page: 437
0x16C	IC_USB	0x0000_0000	IC USB Register	Page: 437
0x170	ULPI_VIEWPORT	0x0000_0000	ULPI Viewport Register	Page: 439
0x178	ENDPTNAK	0x0000_0000	Endpoint NAK Register	Page: 440
0x17C	ENDPTNAKEN	0x0000_0000	Endpoint NAKEN Register	Page: 440
0x184	PORTSC1	0x0C00_0000	PORTSC1 Register	Page: 441
0x188	PORTSC2	0x0C00_0000	PORTSC2 Register	Page: 442
0x18C	PORTSC3	0x0C00_0000	PORTSC3 Register	Page: 444
0x190	PORTSC4	0x0C00_0000	PORTSC4 Register	Page: 446
0x194	PORTSC5	0x0C00_0000	PORTSC5 Register	Page: 447
0x198	PORTSC6	0x0C00_0000	PORTSC6 Register	Page: 449
0x19C	PORTSC7	0x0C00_0000	PORTSC7 Register	Page: 451
0x1A0	PORTSC8	0x0C00_0000	PORTSC8 Register	Page: 452
0x1A4	OTGSC	0x0000_0020	OTGSC Register	Page: 454
0x1A8	USBMODE	0x0000_0002	USB Mode Register	Page: 456
0x1AC	ENDPTSETUPSTAT	0x0000_0000	Endpoint Setup Stat Register	Page: 457
0x1B0	ENDPTPRIME	0x0000_0000	Endpoint Prime Register	Page: 457
0x1B4	ENDPTFLUSH	0x0000_0000	Endpoint Flush Register	Page: 458
0x1B8	ENDPTSTAT	0x0000_0000	Endpoint Stat Register	Page: 458
0x1BC	ENDPTCOMPLETE	0x0000_0000	Endpoint Complete Register	Page: 459
0x1C0	ENDPTCTRL0	0x0081_0080	Endpoint Control 0 Register	Page: 459
0x1C4	ENDPTCTRL1	0x0000_0000	Endpoint Control 1 Register	Page: 460
0x1C8	ENDPTCTRL2	0x0000_0000	Endpoint Control 2 Register	Page: 462
0x1CC	ENDPTCTRL3	0x0000_0000	Endpoint Control 3 Register	Page: 463
0x1D0	ENDPTCTRL4	0x0000_0000	Endpoint Control 4 Register	Page: 464
0x1D4	ENDPTCTRL5	0x0000_0000	Endpoint Control 5 Register	Page: 466
0x1D8	ENDPTCTRL6	0x0000_0000	Endpoint Control 6 Register	Page: 467
0x1DC	ENDPTCTRL7	0x0000_0000	Endpoint Control 7 Register	Page: 468
0x1E0	ENDPTCTRL8	0x0000_0000	Endpoint Control 8 Register	Page: 470
0x1E4	ENDPTCTRL9	0x0000_0000	Endpoint Control 9 Register	Page: 471

Table 221: USBC Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x1E8	ENDPTCTRL10	0x0000_0000	Endpoint Control 10 Register	Page: 472
0x1EC	ENDPTCTRL11	0x0000_0000	Endpoint Control 11 Register	Page: 474
0x1F0	ENDPTCTRL12	0x0000_0000	Endpoint Control 12 Register	Page: 475
0x1F4	ENDPTCTRL13	0x0000_0000	Endpoint Control 13 Register	Page: 476
0x1F8	ENDPTCTRL14	0x0000_0000	Endpoint Control 14 Register	Page: 478
0x1FC	ENDPTCTRL15	0x0000_0000	Endpoint Control 15 Register	Page: 479
0x200	PHY_ID	0x0000_9411	PHY ID Register	Page: 480
0x204	PLL_Control_0	0x0000_5A78	PLL Control 0 Register	Page: 481
0x208	PLL_Control_1	0x0000_0231	PLL Control 1 Register	Page: 481
0x20C	Reserved_Addr3	0x0000_0000	Reserved_Addr3 Register	Page: 482
0x210	Tx_Channel_Contrl_0	0x0000_0488	TX Channel Control 0 Register	Page: 483
0x214	Tx_Channel_Contrl_1	0x0000_05B0	TX Channel Control 1 Register	Page: 484
0x218	Tx_Channel_Contrl_2	0x0000_02FF	TX Channel Control 2 Register	Page: 485
0x21C	Reserved_Addr7	0x0000_0000	Reserved_Addr7 Register	Page: 486
0x220	Rx_Channel_Contrl_0	0x0000_AA71	RX Channel Control 0 Register	Page: 486
0x224	Rx_Channel_Contrl_1	0x0000_3892	RX Channel Control 1 Register	Page: 487
0x228	Rx_Channel_Contrl_2	0x0000_0125	RX Channel Control 2 Register	Page: 488
0x230	Ana_Contrl_0	0x0000_0110	ANA Control 0 Register	Page: 489
0x234	Ana_Contrl_1	0x0000_1680	ANA Control 1 Register	Page: 490
0x238	Reserved_Addr_C	0x0000_0000	Reserved_Addr_C Register	Page: 491
0x23C	Digital_Control_0	0x0000_2586	Digital Control 0 Register	Page: 491
0x240	Digital_Control_1	0x0000_E400	Digital Control 1 Register	Page: 493
0x244	Digital_Control_2	0x0000_0F13	Digital Control 2 Register	Page: 494
0x248	Reserved_Addr_12H	0x0000_0000	Reserved_Addr_12H Register	Page: 495
0x24C	Test_Contrl_and_Status_0	0x0000_0000	Test Control and Status 0 Register	Page: 496
0x250	Test_Contrl_and_Status_1	0x0000_0060	Test Control and Status 1 Register	Page: 497
0x254	Reserved_Addr_15H	0x0000_0000	Reserved_Addr_15H Register	Page: 498
0x258	PHY_REG_CHGDTC_CONTRL	0x0000_0000	PHY REG CHGDTC Control Register	Page: 498
0x25C	PHY_REG_OTG_CONTROL	0x0000_0000	PHY REG OTG Control Register	Page: 499
0x260	usb2_phy_mon0	0x0000_0000	USB2 PHY Monitor 0 Register	Page: 500
0x264	PHY_REG_CHGDTC_CONTRL_1	0x0000_0000	PHY REG CHGDTC Control 1 Register	Page: 500

Table 221: USBC Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x268	Reserved_Addr_1aH	0x0000_0000	Reserved_Addr_1AH Register	Page: 501
0x26C	Reserved_Addr_1bH	0x0000_0000	Reserved_Addr_1BH Register	Page: 502
0x270	Reserved_Addr_1cH	0x0000_0000	Reserved_Addr_1CH Register	Page: 502
0x274	Reserved_Addr_1dH	0x0000_0000	Reserved_Addr_1DH Register	Page: 503
0x278	Internal_CID	0x0000_9411	Internal CID Register	Page: 503
0x27C	usb2_icid_reg1	0x0000_0080	USB2 ICID Register 1	Page: 504

A.3.2 USBC Registers

A.3.2.1 ID Register (ID)

Instance Name	Offset
ID	0x000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	civersion			version				revision				tag					Reserved		nid							Reserved		id					
HW Rst	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	1	?	?	1	1	1	0	1	0	?	?	0	0	0	1	0	1

Table 222: ID Register (ID)

Bits	Field	Type/ HW Rst	Description
31:29	civersion	R 0x7	CIVERSION
28:25	version	R 0x2	VERSION
24:21	revision	R 0x0	REVISION
20:16	tag	R 0x1	TAG
15:14	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
13:8	nid	R 0x3A	NID
7:6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
5:0	id	R 0x5	ID

A.3.2.2 HW General Register (HWGENERAL)

Instance Name	Offset
HWGENERAL	0x004

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_12																				sm		phym				phyw		bwt	clkc		rt
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 223: HW General Register (HWGENERAL)

Bits	Field	Type/ HW Rst	Description
31:12	reserved_12	R 0x0	Reserved_12
11:10	sm	R 0x0	SM
9:6	phym	R 0x0	PHYM
5:4	phyw	R 0x0	PHYW
3	bwt	R 0x0	BWT
2:1	clkc	R 0x0	CLKC
0	rt	R 0x0	RT

A.3.2.3 HW Host Register (HWHOST)

Instance Name	Offset
HWHOST	0x008

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	ttper								ttasy								reserved_4												nport			hc
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 224: HW Host Register (HWHOST)

Bits	Field	Type/ HW Rst	Description
31:24	ttper	R 0x0	TTPER
23:16	ttasy	R 0x0	TTASY
15:4	reserved_4	R 0x0	Reserved_4
3:1	nport	R 0x0	NPORT
0	hc	R 0x0	HC

A.3.2.4 HW Device Register (HWDEVICE)

Instance Name	Offset
HWDEVICE	0x00C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_6																										devep					dc
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Table 225: HW Device Register (HWDEVICE)

Bits	Field	Type/ HW Rst	Description
31:6	reserved_6	R 0x0	Reserved_6
5:1	devep	R 0x1	DEVEP
0	dc	R 0x0	DC

A.3.2.5 HW TXBUF Register (HWTXBUF)

Instance Name	Offset
HWTXBUF	0x010

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved	reserved_24							txchanadd								txadd								txburst							
HW Rst	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 226: HW TXBUF Register (HWTXBUF)

Bits	Field	Type/ HW Rst	Description
31	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
30:24	reserved_24	R 0x0	Reserved_24
23:16	txchanadd	R 0x0	TXCHANADD
15:8	txadd	R 0x0	TXADD
7:0	txburst	R 0x0	TXBURST

A.3.2.6 HW RXBUF Register (HWRXBUF)

Instance Name	Offset
HWRXBUF	0x014

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_16																rxadd								rxburst							
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 227: HW RXBUF Register (HWRXBUF)

Bits	Field	Type/ HW Rst	Description
31:16	reserved_16	R 0x0	Reserved_16
15:8	rxadd	R 0x0	RXADD
7:0	rxburst	R 0x0	RX Burst

A.3.2.7 HW TXBUF0 Register (HWTXBUF0)

Instance Name	Offset
HWTXBUF0	0x018

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

Table 228: HW TXBUF0 Register (HWTXBUF0)

Bits	Field	Type/ HW Rst	Description
31:0	txburst	R/W 0x7777	TX Burst

A.3.2.8 HW TXBUF1 Register (HWTXBUF1)

Instance Name	Offset
HWTXBUF1	0x01C

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

Table 229: HW TXBUF1 Register (HWTXBUF1)

Bits	Field	Type/ HW Rst	Description
31:0	txburst	R/W 0x7777	TX Burst

A.3.2.9 GPTIMER0LD Register (GPTIMER0LD)

Instance Name	Offset
GPTIMER0LD Register	0x080

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								gptld																							
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 230: GPTIMER0LD Register (GPTIMER0LD)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R/W 0x0	Reserved_24
23:0	gptld	R/W 0x0	GPTLD

A.3.2.10 GPTIMER0CTRL (GPTIMER0CTRL)

Instance Name	Offset
GPTIMER0CTRL	0x084

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	gptrun	gptrst	reserved_25					gptmode	gptcnt																							
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 231: GPTIMER0CTRL (GPTIMER0CTRL)

Bits	Field	Type/ HW Rst	Description
31	gptrun	R/W 0x0	GPTRUN
30	gptrst	R 0x0	GPTRST
29:25	reserved_25	R 0x0	Reserved_25
24	gptmode	R/W 0x0	GPTMODE
23:0	gptcnt	R 0x0	GPTCNT

A.3.2.11 GPTTIMER1LD Register (GPTTIMER1LD)

Instance Name	Offset
GPTTIMER1LD	0x088

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								gptld																							
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 232: GPTTIMER1LD Register (GPTTIMER1LD)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R/W 0x0	Reserved_24
23:0	gptld	R/W 0x0	GPTLD

A.3.2.12 GP Timer1 Control Register (GPTIMER1CTRL)

Instance Name	Offset
GPTIMER1CTRL	0x08C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	gptrun	gptrst	reserved_25					gptmode	gptcnt																							
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 233: GP Timer1 Control Register (GPTIMER1CTRL)

Bits	Field	Type/ HW Rst	Description
31	gptrun	R/W 0x0	GPTRUN
30	gptrst	R 0x0	GPTRST
29:25	reserved_25	R 0x0	Reserved_25
24	gptmode	R/W 0x0	GPTMODE
23:0	gptcnt	R 0x0	GPTCNT

A.3.2.13 SBUS Config Register (SBUSCFG)

Instance Name	Offset
SBUSCFG	0x090

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_3																														ahbbrst	
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 234: SBUS Config Register (SBUSCFG)

Bits	Field	Type/ HW Rst	Description
31:3	reserved_3	R 0x0	Reserved_3
2:0	ahbbrst	R/W 0x0	AHB BRST

A.3.2.14 Cap Length Regiser (CAPLENGTH)

Instance Name	Offset
CAPLENGTH	0x100

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

Table 235: Cap Length Regiser (CAPLENGTH)

Bits	Field	Type/ HW Rst	Description
31:16	hciversion	R 0x100	HCU Version
15:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7:0	caplength	R 0x40	Cap Lengh

A.3.2.15 HCS Params Register (HCSPARAMS)

Instance Name	Offset
HCSPARAMS	0x104

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_28				n_tt				n_ptt				reserved_17			n_tt	n_ptt				n_pcc				reserved_5			ppc	n_ports			
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 236: HCS Params Register (HCSPARAMS)

Bits	Field	Type/ HW Rst	Description
31:28	reserved_28	R 0x0	Reserved_28
27:24	n_tt	R 0x0	N_TT
23:20	n_ptt	R 0x0	N_PTT
19:17	reserved_17	R 0x0	Reserved_17
16	n_tt	R 0x1	N_TT
15:12	n_ptt	R 0x0	N_PTT
11:8	n_pcc	R 0x0	N_PCC
7:5	reserved_5	R 0x0	Reserved_5
4	ppc	R 0x0	PPC
3:0	n_ports	R 0x0	N_PORTS

A.3.2.16 HCC Params Register (HCCPARAMS)

Instance Name	Offset
HCCPARAMS Register	0x108

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_16																eecp								ist				reserved_3	asp	pfl	adc
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Table 237: HCC Params Register (HCCPARAMS)

Bits	Field	Type/ HW Rst	Description
31:16	reserved_16	R 0x0	Reserved_16
15:8	eecp	R 0x0	EECP
7:4	ist	R 0x0	IST
3	reserved_3	R 0x0	Reserved_3
2	asp	R 0x1	ASP
1	pfl	R 0x1	PFL
0	adc	R 0x0	ADC

A.3.2.17 DCI Version Register (DCIVERSION)

Instance Name	Offset
DCIVERSION	0x120

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

Table 238: DCI Version Register (DCIVERSION)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	dciversion	R 0x1	DCI Version

A.3.2.18 DCC Params Register (DCCPARAMS)

Instance Name	Offset
DCCPARAMS Register	0x124

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	lpm_en	reserved_9																						hc	dc	reserved_5		den				
HW Rst	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 239: DCC Params Register (DCCPARAMS)

Bits	Field	Type/ HW Rst	Description
31	lpm_en	R 0x1	LPM_EN
30:9	reserved_9	R 0x0	Reserved_9
8	hc	R 0x0	HC
7	dc	R 0x0	DC
6:5	reserved_5	R 0x0	Reserved_5

Table 239: DCC Params Register (DCCPARAMS) (Continued)

Bits	Field	Type/ HW Rst	Description
4:0	den	R 0x0	DEN

A.3.2.19 DevLPMCSR Register (DevLPMCSR)

Instance Name	Offset
DevLPMCSR	0x128

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	lpm_rsp		lpm_phcd_only	brmtwake	linkstate				hird				reserved_18		lpm_any_ep	hst_rsm_en	lpm_on	always_log	min_slp_en	stall_ok	ack_ok	ie_l1state	l1state	rwake_en	ie_lpmerr	ie_lpmack	ie_lpmpkt	ie_l1rsm	int_lpmerr	int_lpmack	int_lpmpkt	int_l1rsm
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Table 240: DevLPMCSR Register (DevLPMCSR)

Bits	Field	Type/ HW Rst	Description
31:30	lpm_rsp	R 0x0	LPM RSP Register
29	lpm_phcd_only	R/W 0x0	LPM_PHCD_only
28	brmtwake	R 0x0	BRMTWAKE
27:24	linkstate	R 0x0	LINKSTATE
23:20	hird	R 0x0	HIRD
19:18	reserved_18	R 0x0	Reserved_18
17	lpm_any_ep	R/W 0x0	LPM_ANY_EP
16	hst_rsm_en	R/W 0x1	HST_RSM_EN
15	lpm_on	R/W 0x1	LPM_ON
14	always_log	R/W 0x1	ALWAYS_LOG

Table 240: DevLPMCSR Register (DevLPMCSR) (Continued)

Bits	Field	Type/ HW Rst	Description
13	min_slp_en	R/W 0x1	MIN_SLP_EN
12	stall_ok	R/W 0x1	STALL_OK
11	ack_ok	R/W 0x0	ACK_OK
10	ie_l1state	R/W 0x0	IE_L1STATE
9	l1state	R/W 0x0	L1STATE
8	rwake_en	R/W 0x0	RWAKE_EN
7	ie_lpmerr	R/W 0x0	IE_LPMERR
6	ie_lpmack	R/W 0x0	IE_LPMACK
5	ie_lpmpkt	R/W 0x0	IE_LPMPKT
4	ie_l1rsm	R/W 0x0	IE_L1RSM
3	int_lpmerr	R/W 0x0	INT_LPMERR
2	int_lpmack	R/W 0x0	INT_LPMACK
1	int_lpmpkt	R/W 0x0	INT_LPMPKT
0	int_l1rsm	R/W 0x0	INT_L1RSM

A.3.2.20 USB Command Register (USBCMD)

Instance Name	Offset
USBCMD	0x140

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								itc								fs2	atdtw	sutw	reserved_12	aspe	reserved_10	asp1	asp0	lr	iaa	ase	pse	fs1	fs0	rst	rs
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0

Table 241: USB Command Register (USBCMD)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23:16	itc	R/W 0x8	ITC
15	fs2	R/W 0x0	HOST Only
14	atdtw	R/W 0x0	ATDTW
13	sutw	R/W 0x0	SUTW
12	reserved_12	R 0x0	Reserved_12
11	aspe	R/W 0x1	ASPE
10	reserved_10	R 0x0	Reserved_10
9	asp1	R/W 0x1	ASP1
8	asp0	R/W 0x1	ASP0
7	lr	R 0x0	LR
6	iaa	R/W 0x0	HOST Only
5	ase	R/W 0x0	HOST Only
4	pse	R/W 0x0	HOST Only

Table 241: USB Command Register (USBCMD) (Continued)

Bits	Field	Type/ HW Rst	Description
3	fs1	R/W 0x0	HOST Only
2	fs0	R/W 0x0	HOST Only
1	rst	R/W 0x0	RST
0	rs	R/W 0x0	RS

A.3.2.21 USB STS Register (USBSTS)

Instance Name	Offset
USBSTS	0x144

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_26						ti1	ti0	reserved_20				upi	uai	reserved_17	naki	as	ps	rcl	hch	reserved_11	ulpii	reserved_9	sli	sri	uri	aai	sei	fri	pci	uei	ui
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Table 242: USB STS Register (USBSTS)

Bits	Field	Type/ HW Rst	Description
31:26	reserved_26	R 0x0	Reserved_26
25	ti1	R/W 0x0	TI1 (rwc)
24	ti0	R/W 0x0	TI0 (rwc)
23:20	reserved_20	R/W 0x0	Reserved_20
19	upi	R/W 0x0	UPI (rwc)
18	uai	R/W 0x0	UAI (rwc)
17	reserved_17	R 0x0	Reserved_17

Table 242: USB STS Register (USBSTS) (Continued)

Bits	Field	Type/ HW Rst	Description
16	naki	R 0x0	NAKI
15	as	R 0x0	AS
14	ps	R 0x0	PS
13	rcl	R 0x0	RCL
12	hch	R 0x1	HCH
11	reserved_11	R 0x0	Reserved_11
10	ulpii	R/W 0x0	ULPII (rwc)
9	reserved_9	R 0x0	Reserved_9
8	sli	R/W 0x0	SLI (rwc)
7	sri	R/W 0x0	SRI (rwc)
6	uri	R/W 0x0	URI (rwc)
5	aai	R/W 0x0	AAI (rwc)
4	sei	R/W 0x0	SEI (rwc)
3	fri	R/W 0x0	FRI (rwc)
2	pci	R/W 0x0	PCI (rwc)
1	uei	R/W 0x0	UEI (rwc)
0	ui	R/W 0x0	UI (rwc)

A.3.2.22 USB Interrupt Register (USBINTR)

Instance Name	Offset
USBINTR	0x148

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_26						tie1	tie0	reserved_20				upe	uae	reserved_17	nake	reserved_15	reserved_14	reserved_13	reserved_12	reserved_11	ulpe	reserved_9	sle	sre	ure	aae	see	fre	pce	uee	ue
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 243: USB Interrupt Register (USBINTR)

Bits	Field	Type/ HW Rst	Description
31:26	reserved_26	R 0x0	Reserved_26
25	tie1	R/W 0x0	TIE1
24	tie0	R/W 0x0	TIE0
23:20	reserved_20	R 0x0	Reserved_20
19	upe	R/W 0x0	UPE (not Used in Device mode)
18	uae	R/W 0x0	UAE (not Used in Device mode)
17	reserved_17	R 0x0	Reserved_17
16	nake	R 0x0	NAKE
15	reserved_15	R 0x0	Not Define in DUT, AS
14	reserved_14	R 0x0	Not Define in DUT, PS
13	reserved_13	R 0x0	Not Define in DUT,RCL
12	reserved_12	R/W 0x0	Reserved_12
11	reserved_11	R/W 0x0	Reserved_11
10	ulpe	R/W 0x0	ULPE Only used VUSB_HS_PHY_ULPI =1.

Table 243: USB Interrupt Register (USBINTR) (Continued)

Bits	Field	Type/ HW Rst	Description
9	reserved_9	R/W 0x0	Reserved_9
8	sle	R/W 0x0	SLE
7	sre	R/W 0x0	SRE
6	ure	R/W 0x0	URE
5	aae	R/W 0x0	AAE (HOST only)
4	see	R/W 0x0	SEE
3	fre	R/W 0x0	FRE (HOST only)
2	pce	R/W 0x0	PCE
1	uee	R/W 0x0	UEE (rwc)
0	ue	R/W 0x0	UE

A.3.2.23 FR Index Register (FRINDEX)

Instance Name	Offset
FRINDEX	0x14C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_14																		frindex													
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 244: FR Index Register (FRINDEX)

Bits	Field	Type/ HW Rst	Description
31:14	reserved_14	R 0x0	Reserved_14
13:0	frindex	R/W 0x0	FRINDEX Device RO; Host R/W.

A.3.2.24 Periodic List Base Register (PERIODICLISTBASE)

Instance Name	Offset
PERIODICLISTBASE	0x154

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	usbadr							usbadra	reserved_0																							
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 245: Periodic List Base Register (PERIODICLISTBASE)

Bits	Field	Type/ HW Rst	Description
31:25	usbadr	R/W 0x0	USBADR
24	usbadra	R/W 0x0	USBADRA
23:0	reserved_0	R 0x0	Reserved_0

A.3.2.25 Async List Address Register (ASYNCLISTADDR)

Instance Name	Offset
ASYNCLISTADDR	0x158

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	epbase																					reserved_0										
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 246: Async List Address Register (ASYNCLISTADDR)

Bits	Field	Type/ HW Rst	Description
31:11	epbase	R/W 0x0	EPBASE
10:0	reserved_0	R 0x0	Reserved_0

A.3.2.26 TT Control Register (TTCTRL Register)

Instance Name	Offset
TTCTRL Register	0x15C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_31	ttha							reserved_2																						ttac	ttas
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 247: TT Control Register (TTCTRL Register)

Bits	Field	Type/ HW Rst	Description
31	reserved_31	R 0x0	Reserved_31
30:24	ttha	R/W 0x0	TTHA
23:2	reserved_2	R 0x0	Reserved_2
1	ttac	R/W 0x0	TTAC
0	ttas	R 0x0	TTAS

A.3.2.27 Burst Size Register (BURSTSIZE)

Instance Name	Offset
BURSTSIZE	0x160

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_16																txpburst								rxpburst							
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 248: Burst Size Register (BURSTSIZE)

Bits	Field	Type/ HW Rst	Description
31:16	reserved_16	R 0x0	Reserved_16
15:8	txpburst	R/W 0x0	TXP Burst

Table 248: Burst Size Register (BURSTSIZE) (Continued)

Bits	Field	Type/ HW Rst	Description
7:0	rxpburst	R/W 0x0	RXP Burst

A.3.2.28 TX Fill Tuning Register (TXFILLTUNING)

Instance Name	Offset
TXFILLTUNING	0x164

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_22										txfifothres						reserved_13			txschhealth					reserved_7	txschoh						
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 249: TX Fill Tuning Register (TXFILLTUNING)

Bits	Field	Type/ HW Rst	Description
31:22	reserved_22	R 0x0	Reserved_22
21:16	txfifothres	R/W 0x2	TX FIFO Threshold Only use in HOST & MPH mode.
15:13	reserved_13	R 0x0	Reserved_13
12:8	txschhealth	R/W 0x0	TXSCHHEALTH Only use in HOST & MPH mode, rwc.
7	reserved_7	R 0x0	Reserved_7
6:0	txschoh	R/W 0x0	TXSCHOH Only use in HOST & MPH mode.

A.3.2.29 TX TT Fill Tuning Register (TXTTFILLTUNING)

Instance Name	Offset
TXTTFILLTUNING	0x168

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_13																			txttschhealtj					reserved_5			txttschoh				
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 250: TX TT Fill Tuning Register (TXTTFILLTUNING)

Bits	Field	Type/ HW Rst	Description
31:13	reserved_13	R 0x0	Reserved_13
12:8	txttschhealtj	R/W 0x0	TXTTSCHHEALTJ Only use in HOST & MPH mode, rwc.
7:5	reserved_5	R 0x0	Reserved_5
4:0	txttschoh	R/W 0x0	TXTTSCHOH Only use in HOST & MPH mode.

A.3.2.30 IC USB Register (IC_USB)

Instance Name	Offset
IC_USB	0x16C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	ic8	ic_vdd8			ic7	ic_vdd7			ic6	ic_vdd6			ic5	ic_vdd5			ic4	ic_vdd4			ic3	ic_vdd3			ic2	ic_vdd2			ic1	ic_vdd1		
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 251: IC USB Register (IC_USB)

Bits	Field	Type/ HW Rst	Description
31	ic8	R/W 0x0	Available in MPH & VUSB_HS_PHY_IC_USB =1
30:28	ic_vdd8	R/W 0x0	Available in MPH & VUSB_HS_PHY_IC_USB =1
27	ic7	R/W 0x0	Available in MPH & VUSB_HS_PHY_IC_USB =1
26:24	ic_vdd7	R/W 0x0	Available in MPH & VUSB_HS_PHY_IC_USB =1

Table 251: IC USB Register (IC_USB) (Continued)

Bits	Field	Type/ HW Rst	Description
23	ic6	R/W 0x0	Available in MPH & VUSB_HS_PHY_IC_USB =1
22:20	ic_vdd6	R/W 0x0	Available in MPH & VUSB_HS_PHY_IC_USB =1
19	ic5	R/W 0x0	Available in MPH & VUSB_HS_PHY_IC_USB =1
18:16	ic_vdd5	R/W 0x0	Available in MPH & VUSB_HS_PHY_IC_USB =1
15	ic4	R/W 0x0	Available in MPH & VUSB_HS_PHY_IC_USB =1
14:12	ic_vdd4	R/W 0x0	Available in MPH & VUSB_HS_PHY_IC_USB =1
11	ic3	R/W 0x0	Available in MPH & VUSB_HS_PHY_IC_USB =1
10:8	ic_vdd3	R/W 0x0	Available in MPH & VUSB_HS_PHY_IC_USB =1
7	ic2	R/W 0x0	Available in MPH & VUSB_HS_PHY_IC_USB =1
6:4	ic_vdd2	R/W 0x0	Available in MPH & VUSB_HS_PHY_IC_USB =1
3	ic1	R/W 0x0	Available in MPH & VUSB_HS_PHY_IC_USB =1
2:0	ic_vdd1	R/W 0x0	Available in MPH & VUSB_HS_PHY_IC_USB =1

A.3.2.31 ULPI Viewport Register (ULPI_VIEWPORT)

Instance Name	Offset
ULPI_VIEWPORT	0x170

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	ulpiwu	ulpirun	ulpirw	reserved_28	ulpiss	ulpiport			ulpiaddr								ulpidatrd								ulpidatwr							
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 252: ULPI Viewport Register (ULPI_VIEWPORT)

Bits	Field	Type/ HW Rst	Description
31	ulpiwu	R/W 0x0	Not Available
30	ulpirun	R/W 0x0	Not Available
29	ulpirw	R/W 0x0	Not Available
28	reserved_28	R/W 0x0	Not Available
27	ulpiss	R/W 0x0	Not Available
26:24	ulpiport	R/W 0x0	Not Available
23:16	ulpiaddr	R/W 0x0	Not Available
15:8	ulpidatrd	R 0x0	Not Available
7:0	ulpidatwr	R/W 0x0	Not Available

A.3.2.32 Endpoint NAK Register (ENDPTNAK)

Instance Name	Offset
ENDPTNAK	0x178

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

Table 253: Endpoint NAK Register (ENDPTNAK)

Bits	Field	Type/ HW Rst	Description
31:16	eptn	R/W 0x0	EPTN (rwc)
15:0	eprn	R/W 0x0	EPRN (rwc)

A.3.2.33 Endpoint NAKEN (ENDPTNAKEN)

Instance Name	Offset
ENDPTNAKEN	0x17C

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

Table 254: Endpoint NAKEN Register (ENDPTNAKEN)

Bits	Field	Type/ HW Rst	Description
31:16	eptne	R/W 0x0	EPTNE Only 3 PHY max.
15:0	eprne	R/W 0x0	EPRNE Only 3 PHY max.

A.3.2.34 PORTSC1 Register (PORTSC1)

Instance Name	Offset
PORTSC1	0x184

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	pts		sts	ptw	pspd		pts2	pfsc	phcd	wkoc	wkds	wkcn	ptc				pic		po	pp	ls		hsp	pr	susp	fpr	occ	oca	pec	pe	csc	ccs
HW Rst	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 255: PORTSC1 Register (PORTSC1)

Bits	Field	Type/ HW Rst	Description
31:30	pts	R/W 0x0	PTS
29	sts	R/W 0x0	STS
28	ptw	R/W 0x0	PTW
27:26	pspd	R 0x3	PSPD
25	pts2	R/W 0x0	PTS2
24	pfsc	R/W 0x0	PFSC
23	phcd	R/W 0x0	PHCD
22	wkoc	R/W 0x0	WKOC
21	wkds	R/W 0x0	WKDS
20	wkcn	R/W 0x0	WKCN
19:16	ptc	R/W 0x0	PTC
15:14	pic	R/W 0x0	PIC
13	po	R 0x0	PO
12	pp	R/W 0x0	PP
11:10	ls	R 0x0	LS

Table 255: PORTSC1 Register (PORTSC1) (Continued)

Bits	Field	Type/ HW Rst	Description
9	hsp	R 0x0	HSP
8	pr	R/W 0x0	PR
7	susp	R/W 0x0	SUSP
6	fpr	R/W 0x0	FPR
5	occ	R/W 0x0	OCC
4	oca	R 0x0	OCA
3	pec	R 0x0	PEC (rwc)
2	pe	R 0x0	PE (rwc)
1	csc	R 0x0	CSC (rwc)
0	ccs	R 0x0	CCS

A.3.2.35 PORTSC2 Register (PORTSC2)

Instance Name	Offset
PORTSC2	0x188

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	pts		sts	ptw	pspd		pts2	pfsc	phcd	wkoc	wkds	wkcn	ptc				pic		po	pp	ls		hsp	pr	susp	fpr	occ	oca	pec	pe	csc	ccs
HW Rst	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 256: PORTSC2 Register (PORTSC2)

Bits	Field	Type/ HW Rst	Description
31:30	pts	R/W 0x0	PTS
29	sts	R/W 0x0	STS

Table 256: PORTSC2 Register (PORTSC2) (Continued)

Bits	Field	Type/ HW Rst	Description
28	ptw	R/W 0x0	PTW
27:26	pspd	R 0x3	PSPD
25	pts2	R/W 0x0	PTS2
24	pfsc	R/W 0x0	PFSC
23	phcd	R/W 0x0	PHCD
22	wkoc	R/W 0x0	WKOC
21	wkds	R/W 0x0	WKDS
20	wkcn	R/W 0x0	WKCN
19:16	ptc	R/W 0x0	PTC
15:14	pic	R/W 0x0	PIC
13	po	R 0x0	PO
12	pp	R/W 0x0	PP
11:10	ls	R 0x0	LS
9	hsp	R 0x0	HSP
8	pr	R/W 0x0	PR
7	susp	R/W 0x0	SUSP
6	fpr	R/W 0x0	FPR
5	occ	R/W 0x0	OCC
4	oca	R 0x0	OCA

Table 256: PORTSC2 Register (PORTSC2) (Continued)

Bits	Field	Type/ HW Rst	Description
3	pec	R 0x0	PEC (rwc)
2	pe	R 0x0	PE (rwc)
1	csc	R 0x0	CSC (rwc)
0	ccs	R 0x0	CCS

A.3.2.36 PORTSC3 Register (PORTSC3)

Instance Name	Offset
PORTSC3	0x18C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	pts		sts	ptw	pspd		pts2	pfsc	phcd	wkoc	wkds	wkcn	ptc				pic		po	pp	ls		hsp	pr	susp	fpr	occ	oca	pec	pe	csc	ccs
HW Rst	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 257: PORTSC3 Register (PORTSC3)

Bits	Field	Type/ HW Rst	Description
31:30	pts	R/W 0x0	PTS
29	sts	R/W 0x0	STS
28	ptw	R/W 0x0	PTW
27:26	pspd	R 0x3	PSPD
25	pts2	R/W 0x0	PTS2
24	pfsc	R/W 0x0	PFSC
23	phcd	R/W 0x0	PHCD
22	wkoc	R/W 0x0	WKOC

Table 257: PORTSC3 Register (PORTSC3) (Continued)

Bits	Field	Type/ HW Rst	Description
21	wkds	R/W 0x0	WKDS
20	wkcn	R/W 0x0	WKCN
19:16	ptc	R/W 0x0	PTC
15:14	pic	R/W 0x0	PIC
13	po	R 0x0	PO
12	pp	R/W 0x0	PP
11:10	ls	R 0x0	LS
9	hsp	R 0x0	HSP
8	pr	R/W 0x0	PR
7	susp	R/W 0x0	SUSP
6	fpr	R/W 0x0	FPR
5	occ	R/W 0x0	OCC
4	oca	R 0x0	OCA
3	pec	R 0x0	PEC (rwc)
2	pe	R 0x0	PE (rwc)
1	csc	R 0x0	CSC (rwc)
0	ccs	R 0x0	CCS

A.3.2.37 PORTSC4 Register (PORTSC4)

Instance Name	Offset
PORTSC4	0x190

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	pts		sts	ptw	pspd		pts2	pfsc	phcd	wkoc	wkds	wkcn	ptc				pic		po	pp	ls		hsp	pr	susp	fpr	occ	oca	pec	pe	csc	ccs
HW Rst	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 258: PORTSC4 Register (PORTSC4)

Bits	Field	Type/ HW Rst	Description
31:30	pts	R/W 0x0	PTS
29	sts	R/W 0x0	STS
28	ptw	R/W 0x0	PTW
27:26	pspd	R 0x3	PSPD
25	pts2	R/W 0x0	PTS2
24	pfsc	R/W 0x0	PFSC
23	phcd	R/W 0x0	PHCD
22	wkoc	R/W 0x0	WKOC
21	wkds	R/W 0x0	WKDS
20	wkcn	R/W 0x0	WKCN
19:16	ptc	R/W 0x0	PTC
15:14	pic	R/W 0x0	PIC
13	po	R 0x0	PO
12	pp	R/W 0x0	PP
11:10	ls	R 0x0	LS

Table 258: PORTSC4 Register (PORTSC4) (Continued)

Bits	Field	Type/ HW Rst	Description
9	hsp	R 0x0	HSP
8	pr	R/W 0x0	PR
7	susp	R/W 0x0	SUSP
6	fpr	R/W 0x0	FPR
5	occ	R/W 0x0	OCC
4	oca	R 0x0	OCA
3	pec	R 0x0	PEC (rwc)
2	pe	R 0x0	PE (rwc)
1	csc	R 0x0	CSC (rwc)
0	ccs	R 0x0	CCS

A.3.2.38 PORTSC5 Register (PORTSC5)

Instance Name	Offset
PORTSC5	0x194

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	pts		sts	ptw	pspd		pts2	pfsc	phcd	wkoc	wkds	wkcn	ptc				pic		po	pp	ls		hsp	pr	susp	fpr	occ	oca	pec	pe	csc	ccs
HW Rst	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 259: PORTSC5 Register (PORTSC5)

Bits	Field	Type/ HW Rst	Description
31:30	pts	R/W 0x0	PTS
29	sts	R/W 0x0	STS

Table 259: PORTSC5 Register (PORTSC5) (Continued)

Bits	Field	Type/ HW Rst	Description
28	ptw	R/W 0x0	PTW
27:26	pspd	R 0x3	PSPD
25	pts2	R/W 0x0	PTS2
24	pfsc	R/W 0x0	PFSC
23	phcd	R/W 0x0	PHCD
22	wkoc	R/W 0x0	WKOC
21	wkds	R/W 0x0	WKDS
20	wkcn	R/W 0x0	WKCN
19:16	ptc	R/W 0x0	PTC
15:14	pic	R/W 0x0	PIC
13	po	R 0x0	PO
12	pp	R/W 0x0	PP
11:10	ls	R 0x0	LS
9	hsp	R 0x0	HSP
8	pr	R/W 0x0	PR
7	susp	R/W 0x0	SUSP
6	fpr	R/W 0x0	FPR
5	occ	R/W 0x0	OCC
4	oca	R 0x0	OCA

Table 259: PORTSC5 Register (PORTSC5) (Continued)

Bits	Field	Type/ HW Rst	Description
3	pec	R 0x0	PEC (rwc)
2	pe	R 0x0	PE (rwc)
1	csc	R 0x0	CSC (rwc)
0	ccs	R 0x0	CCS

A.3.2.39 PORTSC6 Register (PORTSC6)

Instance Name	Offset
PORTSC6	0x198

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	pts		sts	ptw	pspd		pts2	pfsc	phcd	wkoc	wkds	wkcn	ptc				pic		po	pp	ls		hsp	pr	susp	fpr	occ	oca	pec	pe	csc	ccs
HW Rst	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 260: PORTSC6 Register (PORTSC6)

Bits	Field	Type/ HW Rst	Description
31:30	pts	R/W 0x0	PTS
29	sts	R/W 0x0	STS
28	ptw	R/W 0x0	PTW
27:26	pspd	R 0x3	PSPD
25	pts2	R/W 0x0	PTS2
24	pfsc	R/W 0x0	PFSC
23	phcd	R/W 0x0	PHCD
22	wkoc	R/W 0x0	WKOC

Table 260: PORTSC6 Register (PORTSC6) (Continued)

Bits	Field	Type/ HW Rst	Description
21	wkds	R/W 0x0	WKDS
20	wkcn	R/W 0x0	WKCN
19:16	ptc	R/W 0x0	PTC
15:14	pic	R/W 0x0	PIC
13	po	R 0x0	PO
12	pp	R/W 0x0	PP
11:10	ls	R 0x0	LS
9	hsp	R 0x0	HSP
8	pr	R/W 0x0	PR
7	susp	R/W 0x0	SUSP
6	fpr	R/W 0x0	FPR
5	occ	R/W 0x0	OCC
4	oca	R 0x0	OCA
3	pec	R 0x0	PEC (rwc)
2	pe	R 0x0	PE (rwc)
1	csc	R 0x0	CSC (rwc)
0	ccs	R 0x0	CCS

A.3.2.40 PORTSC7 Register (PORTSC7)

Instance Name	Offset
PORTSC7	0x19C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	pts		sts	ptw	pspd		pts2	pfsc	phcd	wkoc	wkds	wkcn	ptc				pic		po	pp	ls		hsp	pr	susp	fpr	occ	oca	pec	pe	csc	ccs
HW Rst	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 261: PORTSC7 Register (PORTSC7)

Bits	Field	Type/ HW Rst	Description
31:30	pts	R/W 0x0	PTS
29	sts	R/W 0x0	STS
28	ptw	R/W 0x0	PTW
27:26	pspd	R 0x3	PSPD
25	pts2	R/W 0x0	PTS2
24	pfsc	R/W 0x0	PFSC
23	phcd	R/W 0x0	PHCD
22	wkoc	R/W 0x0	WKOC
21	wkds	R/W 0x0	WKDS
20	wkcn	R/W 0x0	WKCN
19:16	ptc	R/W 0x0	PTC
15:14	pic	R/W 0x0	PIC
13	po	R 0x0	PO
12	pp	R/W 0x0	PP
11:10	ls	R 0x0	LS

Table 261: PORTSC7 Register (PORTSC7) (Continued)

Bits	Field	Type/ HW Rst	Description
9	hsp	R 0x0	HSP
8	pr	R/W 0x0	PR
7	susp	R/W 0x0	SUSP
6	fpr	R/W 0x0	FPR
5	occ	R/W 0x0	OCC
4	oca	R 0x0	OCA
3	pec	R 0x0	PEC (rwc)
2	pe	R 0x0	PE (rwc)
1	csc	R 0x0	CSC (rwc)
0	ccs	R 0x0	CCS

A.3.2.41 PORTSC8 Register (PORTSC8)

Instance Name	Offset
PORTSC8	0x1A0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	pts		sts	ptw	pspd		pts2	pfsc	phcd	wkoc	wkds	wkcn	ptc				pic		po	pp	ls		hsp	pr	susp	fpr	occ	oca	pec	pe	csc	ccs
HW Rst	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 262: PORTSC8 Register (PORTSC8)

Bits	Field	Type/ HW Rst	Description
31:30	pts	R/W 0x0	PTS
29	sts	R/W 0x0	STS

Table 262: PORTSC8 Register (PORTSC8) (Continued)

Bits	Field	Type/ HW Rst	Description
28	ptw	R/W 0x0	PTW
27:26	pspd	R 0x3	PSPD
25	pts2	R/W 0x0	PTS2
24	pfsc	R/W 0x0	PFSC
23	phcd	R/W 0x0	PHCD
22	wkoc	R/W 0x0	WKOC
21	wkds	R/W 0x0	WKDS
20	wkcn	R/W 0x0	WKCN
19:16	ptc	R/W 0x0	PTC
15:14	pic	R/W 0x0	PIC
13	po	R 0x0	PO
12	pp	R/W 0x0	PP
11:10	ls	R 0x0	LS
9	hsp	R 0x0	HSP
8	pr	R/W 0x0	PR
7	susp	R/W 0x0	SUSP
6	fpr	R/W 0x0	FPR
5	occ	R/W 0x0	OCC
4	oca	R 0x0	OCA

Table 262: PORTSC8 Register (PORTSC8) (Continued)

Bits	Field	Type/ HW Rst	Description
3	pec	R 0x0	PEC (rwc)
2	pe	R 0x0	PE (rwc)
1	csc	R 0x0	CSC (rwc)
0	ccs	R 0x0	CCS

A.3.2.42 OTGSC Register (OTGSC)

Instance Name	Offset
OTGSC	0x1A4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_31	dpie	otgsc_1mse	bseie	bsvie	asvie	avvie	idie	reserved_23	dpis	otgsc_1mss	bseis	bsvis	asvis	avvis	idis	reserved_15	dps	otgsc_1mst	bse	bsv	asv	avv	id	haba	hadp	idpu	dp	ot	haar	vc	vd
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Table 263: OTGSC Register (OTGSC)

Bits	Field	Type/ HW Rst	Description
31	reserved_31	R 0x0	OTG Not Enable
30	dpie	R/W 0x0	OTG Not Enable
29	otgsc_1mse	R/W 0x0	OTG Not Enable
28	bseie	R/W 0x0	OTG Not Enable
27	bsvie	R/W 0x0	OTG Not Enable
26	asvie	R/W 0x0	OTG Not Enable
25	avvie	R/W 0x0	OTG Not Enable

Table 263: OTGSC Register (OTGSC) (Continued)

Bits	Field	Type/ HW Rst	Description
24	idie	R/W 0x0	OTG Not Enable
23	reserved_23	R 0x0	OTG Not Enable
22	dpis	R 0x0	DPIS (rwc)
21	otgsc_1mss	R 0x0	OTGSC_1msS (rwc)
20	bseis	R 0x0	BSEIS (rwc)
19	bsvis	R 0x0	BSVIS (rwc)
18	asvis	R 0x0	ASVIS (rwc)
17	avvis	R 0x0	AVVIS (rwc)
16	idis	R 0x0	IDIS (rwc)
15	reserved_15	R 0x0	OTG Not Enable
14	dps	R 0x0	OTG Not Enable
13	otgsc_1mst	R 0x0	OTG Not Enable
12	bse	R 0x0	OTG Not Enable
11	bsv	R 0x0	OTG Not Enable
10	asv	R 0x0	OTG Not Enable
9	avv	R 0x0	OTG Not Enable
8	id	R 0x0	OTG Not Enable
7	haba	R/W 0x0	OTG Not Enable
6	hadp	R/W 0x0	OTG Not Enable

Table 263: OTGSC Register (OTGSC) (Continued)

Bits	Field	Type/ HW Rst	Description
5	idpu	R/W 0x1	OTG Not Enable
4	dp	R/W 0x0	OTG Not Enable
3	ot	R/W 0x0	OTG Not Enable
2	haar	R/W 0x0	OTG Not Enable
1	vc	R/W 0x0	OTG Not Enable
0	vd	R/W 0x0	OTG Not Enable

A.3.2.43 USB Mode Register (USBMODE)

Instance Name	Offset
USBMODE	0x1A8

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_16																srt	reserved_6									vbps	sdis	slom	es	cm	
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Table 264: USB Mode Register (USBMODE)

Bits	Field	Type/ HW Rst	Description
31:16	reserved_16	R 0x0	Reserved_16
15	srt	R/W 0x0	SRT
14:6	reserved_6	R 0x0	Reserved_6
5	vbps	R/W 0x0	VBPS Only used in Host.
4	sdis	R/W 0x0	SDIS
3	slom	R/W 0x0	SLOM

Table 264: USB Mode Register (USBMODE) (Continued)

Bits	Field	Type/ HW Rst	Description
2	es	R/W 0x0	ES
1:0	cm	R/W 0x2	Fix Device Mode

A.3.2.44 Endpoint Setup Stat Register (ENDPTSETUPSTAT)

Instance Name	Offset
ENDPTSETUPSTAT	0x1AC

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_16																endptsetupstat															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 265: Endpoint Setup Stat Register (ENDPTSETUPSTAT)

Bits	Field	Type/ HW Rst	Description
31:16	reserved_16	R 0x0	Reserved_16
15:0	endptsetupstat	R 0x0	Endpoint Setup Stat (rwc)

A.3.2.45 Endpoint Prime Register (ENDPTPRIME)

Instance Name	Offset
ENDPTPRIME	0x1B0

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

Table 266: Endpoint Prime Register (ENDPTPRIME)

Bits	Field	Type/ HW Rst	Description
31:16	petb	R 0x0	PETB (rws)

Table 266: Endpoint Prime Register (ENDPTPRIME) (Continued)

Bits	Field	Type/ HW Rst	Description
15:0	perb	R 0x0	PERB (rws)

A.3.2.46 Endpoint Flush Register (ENDPTFLUSH)

Instance Name	Offset
ENDPTFLUSH	0x1B4

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

Table 267: Endpoint Flush Register (ENDPTFLUSH)

Bits	Field	Type/ HW Rst	Description
31:16	fetb	R 0x0	FETB (rws)
15:0	ferb	R 0x0	FERB (rws)

A.3.2.47 Endpoint Stat Register (ENDPTSTAT)

Instance Name	Offset
ENDPTSTAT	0x1B8

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

Table 268: Endpoint Stat Register (ENDPTSTAT)

Bits	Field	Type/ HW Rst	Description
31:16	etbr	R 0x0	ETBR
15:0	erbr	R 0x0	ERBR

A.3.2.48 Endpoint Complete Register (ENDPTCOMPLETE)

Instance Name	Offset
ENDPTCOMPLETE	0x1BC

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

Table 269: Endpoint Complete Register (ENDPTCOMPLETE)

Bits	Field	Type/ HW Rst	Description
31:16	etce	R 0x0	ETCE (rwc)
15:0	erce	R 0x0	ERCE (rwc)

A.3.2.49 Endpoint Control 0 Register (ENDPTCTRL0)

Instance Name	Offset
ENDPTCTRL0	0x1C0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								txe	reserved_20			txt		reserved_17	txs	reserved_8								rxe	reserved_4			rxt		reserved_1	rxs
HW Rst	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Table 270: Endpoint Control 0 Register (ENDPTCTRL0)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23	txe	R 0x1	TXE
22:20	reserved_20	R 0x0	Reserved_20
19:18	txt	R 0x0	TXT
17	reserved_17	R 0x0	Reserved_17

Table 270: Endpoint Control 0 Register (ENDPTCTRL0) (Continued)

Bits	Field	Type/ HW Rst	Description
16	txs	R/W 0x1	TXS
15:8	reserved_8	R 0x0	Reserved_8
7	rxe	R 0x1	RXE
6:4	reserved_4	R 0x0	Reserved_4
3:2	rxt	R 0x0	RXT
1	reserved_1	R 0x0	Reserved_1
0	rxs	R/W 0x0	RXS

A.3.2.50 Endpoint Control 1 Register (ENDPTCTRL1)

Instance Name	Offset
ENDPTCTRL1	0x1C4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								txe	txr	txi	reserved_20	txt		txd	txs	reserved_8								rxe	rxr	rxi	reserved_4	rxt		rxd	rxs
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 271: Endpoint Control 1 Register (ENDPTCTRL1)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23	txe	R/W 0x0	TXE
22	txr	R 0x0	TXR (ws)
21	txi	R/W 0x0	TXI

Table 271: Endpoint Control 1 Register (ENDPTCTRL1) (Continued)

Bits	Field	Type/ HW Rst	Description
20	reserved_20	R 0x0	Reserved_20
19:18	txt	R/W 0x0	TXT
17	txd	R/W 0x0	TXD
16	txs	R/W 0x0	TXS
15:8	reserved_8	R 0x0	Reserved_8
7	rxe	R/W 0x0	RXE
6	rxr	R 0x0	RXR (ws)
5	rxi	R/W 0x0	RXI
4	reserved_4	R 0x0	Reserved_4
3:2	rxt	R/W 0x0	RXT
1	rxd	R/W 0x0	RXD
0	rxs	R/W 0x0	RXS

A.3.2.51 Endpoint Control 2 Register (ENDPTCTRL2)

Instance Name	Offset
ENDPTCTRL2	0x1C8

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								txe	txr	txi	reserved_20	txt		txd	txs	reserved_8								rxe	rxr	rxi	reserved_4	rxt		rxd	rxs
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 272: Endpoint Control 2 Register (ENDPTCTRL2)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23	txe	R/W 0x0	TXE
22	txr	R 0x0	TXR (ws)
21	txi	R/W 0x0	TXI
20	reserved_20	R 0x0	Reserved_20
19:18	txt	R/W 0x0	TXT
17	txd	R/W 0x0	TXD
16	txs	R/W 0x0	TXS
15:8	reserved_8	R 0x0	Reserved_8
7	rxe	R/W 0x0	RXE
6	rxr	R 0x0	RXR (ws)
5	rxi	R/W 0x0	RXI
4	reserved_4	R 0x0	Reserved_4
3:2	rxt	R/W 0x0	RXT

Table 272: Endpoint Control 2 Register (ENDPTCTRL2) (Continued)

Bits	Field	Type/ HW Rst	Description
1	rxd	R/W 0x0	RXD
0	rxs	R/W 0x0	RXS

A.3.2.52 Endpoint Control 3 Register (ENDPTCTRL3)

Instance Name	Offset
ENDPTCTRL3	0x1CC

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								txe	txr	txi	reserved_20	txt		txd	txs	reserved_8								rxe	rxr	rxi	reserved_4	rxt		rxd	rxs
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 273: Endpoint Control 3 Register (ENDPTCTRL3)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23	txe	R/W 0x0	TXE
22	txr	R 0x0	TXR (ws)
21	txi	R/W 0x0	TXI
20	reserved_20	R 0x0	Reserved_20
19:18	txt	R/W 0x0	TXT
17	txd	R/W 0x0	TXD
16	txs	R/W 0x0	TXS
15:8	reserved_8	R 0x0	Reserved_8

Table 273: Endpoint Control 3 Register (ENDPTCTRL3) (Continued)

Bits	Field	Type/ HW Rst	Description
7	rxe	R/W 0x0	RXE
6	rxr	R 0x0	RXR (ws)
5	rxi	R/W 0x0	RXI
4	reserved_4	R 0x0	Reserved_4
3:2	rxt	R/W 0x0	RXT
1	rxd	R/W 0x0	RXD
0	rxs	R/W 0x0	RXS

A.3.2.53 Endpoint Control 4 Register (ENDPTCTRL4)

Instance Name	Offset
ENDPTCTRL4	0x1D0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								txe	txr	txi	reserved_20	txt		txd	txs	reserved_8								rxe	rxr	rxi	reserved_4	rxt		rxd	rxs
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 274: Endpoint Control 4 Register (ENDPTCTRL4)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23	txe	R/W 0x0	TXE
22	txr	R 0x0	TXR (ws)
21	txi	R/W 0x0	TXI

Table 274: Endpoint Control 4 Register (ENDPTCTRL4) (Continued)

Bits	Field	Type/ HW Rst	Description
20	reserved_20	R 0x0	Reserved_20
19:18	txt	R/W 0x0	TXT
17	txd	R/W 0x0	TXD
16	txs	R/W 0x0	TXS
15:8	reserved_8	R 0x0	Reserved_8
7	rxe	R/W 0x0	RXE
6	rxr	R 0x0	RXR (ws)
5	rxi	R/W 0x0	RXI
4	reserved_4	R 0x0	Reserved_4
3:2	rxt	R/W 0x0	RXT
1	rxd	R/W 0x0	RXD
0	rxs	R/W 0x0	RXS

A.3.2.54 Endpoint Control 5 Register (ENDPTCTRL5)

Instance Name	Offset
ENDPTCTRL5	0x1D4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								txe	txr	txi	reserved_20	txt		txd	txs	reserved_8								rxe	rxr	rxi	reserved_4	rxt		rxd	rxs
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 275: Endpoint Control 5 Register (ENDPTCTRL5)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23	txe	R/W 0x0	TXE
22	txr	R 0x0	TXR (ws)
21	txi	R/W 0x0	TXI
20	reserved_20	R 0x0	Reserved_20
19:18	txt	R/W 0x0	TXT
17	txd	R/W 0x0	TXD
16	txs	R/W 0x0	TXS
15:8	reserved_8	R 0x0	Reserved_8
7	rxe	R/W 0x0	RXE
6	rxr	R 0x0	RXR (ws)
5	rxi	R/W 0x0	RXI
4	reserved_4	R 0x0	Reserved_4
3:2	rxt	R/W 0x0	RXT

Table 275: Endpoint Control 5 Register (ENDPTCTRL5) (Continued)

Bits	Field	Type/ HW Rst	Description
1	rxd	R/W 0x0	RXD
0	rxs	R/W 0x0	RXS

A.3.2.55 Endpoint Control 6 Register (ENDPTCTRL6)

Instance Name	Offset
ENDPTCTRL6	0x1D8

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								txe	txr	txi	reserved_20	txt		txd	txs	reserved_8								rxe	rxr	rxi	reserved_4	rxt		rxd	rxs
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 276: Endpoint Control 6 Register (ENDPTCTRL6)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23	txe	R/W 0x0	TXE
22	txr	R 0x0	TXR (ws)
21	txi	R/W 0x0	TXI
20	reserved_20	R 0x0	Reserved_20
19:18	txt	R/W 0x0	TXT
17	txd	R/W 0x0	TXD
16	txs	R/W 0x0	TXS
15:8	reserved_8	R 0x0	Reserved_8

Table 276: Endpoint Control 6 Register (ENDPTCTRL6) (Continued)

Bits	Field	Type/ HW Rst	Description
7	rxe	R/W 0x0	RXE
6	rxr	R 0x0	RXR (ws)
5	rxi	R/W 0x0	RXI
4	reserved_4	R 0x0	Reserved_4
3:2	rxt	R/W 0x0	RXT
1	rxd	R/W 0x0	RXD
0	rxs	R/W 0x0	RXS

A.3.2.56 Endpoint Control 7 Register (ENDPTCTRL7)

Instance Name	Offset
ENDPTCTRL7	0x1DC

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								txe	txr	txi	reserved_20	txt		txd	txs	reserved_8								rxe	rxr	rxi	reserved_4	rxt		rxd	rxs
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 277: Endpoint Control 7 Register (ENDPTCTRL7)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23	txe	R/W 0x0	TXE
22	txr	R 0x0	TXR (ws)
21	txi	R/W 0x0	TXI

Table 277: Endpoint Control 7 Register (ENDPTCTRL7) (Continued)

Bits	Field	Type/ HW Rst	Description
20	reserved_20	R 0x0	Reserved_20
19:18	txt	R/W 0x0	TXT
17	txd	R/W 0x0	TXD
16	txs	R/W 0x0	TXS
15:8	reserved_8	R 0x0	Reserved_8
7	rxe	R/W 0x0	RXE
6	rxr	R 0x0	RXR (ws)
5	rxi	R/W 0x0	RXI
4	reserved_4	R 0x0	Reserved_4
3:2	rxt	R/W 0x0	RXT
1	rxd	R/W 0x0	RXD
0	rxs	R/W 0x0	RXS

A.3.2.57 Endpoint Control 8 Register (ENDPTCTRL8)

Instance Name	Offset
ENDPTCTRL8	0x1E0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								txe	txr	txi	reserved_20	txt		txd	txs	reserved_8								rxe	rxr	rxi	reserved_4	rxt		rxd	rxs
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 278: Endpoint Control 8 Register (ENDPTCTRL8)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23	txe	R/W 0x0	TXE
22	txr	R 0x0	TXR (ws)
21	txi	R/W 0x0	TXI
20	reserved_20	R 0x0	Reserved_20
19:18	txt	R/W 0x0	TXT
17	txd	R/W 0x0	TXD
16	txs	R/W 0x0	TXS
15:8	reserved_8	R 0x0	Reserved_8
7	rxe	R/W 0x0	RXE
6	rxr	R 0x0	RXR (ws)
5	rxi	R/W 0x0	RXI
4	reserved_4	R 0x0	Reserved_4
3:2	rxt	R/W 0x0	RXT

Table 278: Endpoint Control 8 Register (ENDPTCTRL8) (Continued)

Bits	Field	Type/ HW Rst	Description
1	rxd	R/W 0x0	RXD
0	rxs	R/W 0x0	RXS

A.3.2.58 Endpoint Control 9 Register (ENDPTCTRL9)

Instance Name	Offset
ENDPTCTRL9	0x1E4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								txe	txr	txi	reserved_20	txt		txd	txs	reserved_8								rxe	rxr	rxi	reserved_4	rxt		rxd	rxs
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 279: Endpoint Control 9 Register (ENDPTCTRL9)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23	txe	R/W 0x0	TXE
22	txr	R 0x0	TXR (ws)
21	txi	R/W 0x0	TXI
20	reserved_20	R 0x0	Reserved_20
19:18	txt	R/W 0x0	TXT
17	txd	R/W 0x0	TXD
16	txs	R/W 0x0	TXS
15:8	reserved_8	R 0x0	Reserved_8

Table 279: Endpoint Control 9 Register (ENDPTCTRL9) (Continued)

Bits	Field	Type/ HW Rst	Description
7	rxe	R/W 0x0	RXE
6	rxr	R 0x0	RXR (ws)
5	rxi	R/W 0x0	RXI
4	reserved_4	R 0x0	Reserved_4
3:2	rxt	R/W 0x0	RXT
1	rxd	R/W 0x0	RXD
0	rxs	R/W 0x0	RXS

A.3.2.59 Endpoint Control 10 Register (ENDPTCTRL10)

Instance Name	Offset
ENDPTCTRL10	0x1E8

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								txe	txr	txi	reserved_20	txt		txd	txs	reserved_8								rxe	rxr	rxi	reserved_4	rxt		rxd	rxs
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 280: Endpoint Control 10 Register (ENDPTCTRL10)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23	txe	R/W 0x0	TXE
22	txr	R 0x0	TXR (ws)
21	txi	R/W 0x0	TXI

Table 280: Endpoint Control 10 Register (ENDPTCTRL10) (Continued)

Bits	Field	Type/ HW Rst	Description
20	reserved_20	R 0x0	Reserved_20
19:18	txt	R/W 0x0	TXT
17	txd	R/W 0x0	TXD
16	txs	R/W 0x0	TXS
15:8	reserved_8	R 0x0	Reserved_8
7	rxe	R/W 0x0	RXE
6	rxr	R 0x0	RXR (ws)
5	rxi	R/W 0x0	RXI
4	reserved_4	R 0x0	Reserved_4
3:2	rxt	R/W 0x0	RXT
1	rxd	R/W 0x0	RXD
0	rxs	R/W 0x0	RXS

A.3.2.60 Endpoint Control 11 Register (ENDPTCTRL11)

Instance Name	Offset
ENDPTCTRL11	0x1EC

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								txe	txr	txi	reserved_20	txt		txd	txs	reserved_8								rxe	rxr	rxi	reserved_4	rxt		rxd	rxs
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 281: Endpoint Control 11 Register (ENDPTCTRL11)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23	txe	R/W 0x0	TXE
22	txr	R 0x0	TXR (ws)
21	txi	R/W 0x0	TXI
20	reserved_20	R 0x0	Reserved_20
19:18	txt	R/W 0x0	TXT
17	txd	R/W 0x0	TXD
16	txs	R/W 0x0	TXS
15:8	reserved_8	R 0x0	Reserved_8
7	rxe	R/W 0x0	RXE
6	rxr	R 0x0	RXR (ws)
5	rxi	R/W 0x0	RXI
4	reserved_4	R 0x0	Reserved_4
3:2	rxt	R/W 0x0	RXT

Table 281: Endpoint Control 11 Register (ENDPTCTRL11) (Continued)

Bits	Field	Type/ HW Rst	Description
1	rxd	R/W 0x0	RXD
0	rxs	R/W 0x0	RXS

A.3.2.61 Endpoint Control 12 Register (ENDPTCTRL12)

Instance Name	Offset
ENDPTCTRL12	0x1F0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								txe	txr	txi	reserved_20	txt		txd	txs	reserved_8								rxe	rxr	rxi	reserved_4	rxt		rxd	rxs
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 282: Endpoint Control 12 Register (ENDPTCTRL12)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23	txe	R/W 0x0	TXE
22	txr	R 0x0	TXR (ws)
21	txi	R/W 0x0	TXI
20	reserved_20	R 0x0	Reserved_20
19:18	txt	R/W 0x0	TXT
17	txd	R/W 0x0	TXD
16	txs	R/W 0x0	TXS
15:8	reserved_8	R 0x0	Reserved_8

Table 282: Endpoint Control 12 Register (ENDPTCTRL12) (Continued)

Bits	Field	Type/ HW Rst	Description
7	rxe	R/W 0x0	RXE
6	rxr	R 0x0	RXR (ws)
5	rxi	R/W 0x0	RXI
4	reserved_4	R 0x0	Reserved_4
3:2	rxt	R/W 0x0	RXT
1	rxd	R/W 0x0	RXD
0	rxs	R/W 0x0	RXS

A.3.2.62 Endpoint Control 13 Register (ENDPTCTRL13)

Instance Name	Offset
ENDPTCTRL13	0x1F4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								txe	txr	txi	reserved_20	txt		txd	txs	reserved_8								rxe	rxr	rxi	reserved_4	rxt		rxd	rxs
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 283: Endpoint Control 13 Register (ENDPTCTRL13)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23	txe	R/W 0x0	TXE
22	txr	R 0x0	TXR (ws)
21	txi	R/W 0x0	TXI

Table 283: Endpoint Control 13 Register (ENDPTCTRL13) (Continued)

Bits	Field	Type/ HW Rst	Description
20	reserved_20	R 0x0	Reserved_20
19:18	txt	R/W 0x0	TXT
17	txd	R/W 0x0	TXD
16	txs	R/W 0x0	TXS
15:8	reserved_8	R 0x0	Reserved_8
7	rxe	R/W 0x0	RXE
6	rxr	R 0x0	RXR (ws)
5	rxi	R/W 0x0	RXI
4	reserved_4	R 0x0	Reserved_4
3:2	rxt	R/W 0x0	RXT
1	rxd	R/W 0x0	RXD
0	rxs	R/W 0x0	RXS

A.3.2.63 Endpoint Control 14 Register (ENDPTCTRL14)

Instance Name	Offset
ENDPTCTRL14	0x1F8

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								txe	txr	txi	reserved_20	txt		txd	txs	reserved_8								rxe	rxr	rxi	reserved_4	rxt		rxd	rxs
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 284: Endpoint Control 14 Register (ENDPTCTRL14)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23	txe	R/W 0x0	TXE
22	txr	R 0x0	TXR (ws)
21	txi	R/W 0x0	TXI
20	reserved_20	R 0x0	Reserved_20
19:18	txt	R/W 0x0	TXT
17	txd	R/W 0x0	TXD
16	txs	R/W 0x0	TXS
15:8	reserved_8	R 0x0	Reserved_8
7	rxe	R/W 0x0	RXE
6	rxr	R 0x0	RXR (ws)
5	rxi	R/W 0x0	RXI
4	reserved_4	R 0x0	Reserved_4
3:2	rxt	R/W 0x0	RXT

Table 284: Endpoint Control 14 Register (ENDPTCTRL14) (Continued)

Bits	Field	Type/ HW Rst	Description
1	rxd	R/W 0x0	RXD
0	rxs	R/W 0x0	RXS

A.3.2.64 Endpoint Control 15 Register (ENDPTCTRL15)

Instance Name	Offset
ENDPTCTRL15	0x1FC

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved_24								txe	txr	txi	reserved_20	txt		txd	txs	reserved_8								rxe	rxr	rxi	reserved_4	rxt		rxd	rxs
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 285: Endpoint Control 15 Register (ENDPTCTRL15)

Bits	Field	Type/ HW Rst	Description
31:24	reserved_24	R 0x0	Reserved_24
23	txe	R/W 0x0	TXE
22	txr	R 0x0	TXR (ws)
21	txi	R/W 0x0	TXI
20	reserved_20	R 0x0	Reserved_20
19:18	txt	R/W 0x0	TXT
17	txd	R/W 0x0	TXD
16	txs	R/W 0x0	TXS
15:8	reserved_8	R 0x0	Reserved_8

Table 285: Endpoint Control 15 Register (ENDPTCTRL15) (Continued)

Bits	Field	Type/ HW Rst	Description
7	rxe	R/W 0x0	RXE
6	rxr	R 0x0	RXR (ws)
5	rxi	R/W 0x0	RXI
4	reserved_4	R 0x0	Reserved_4
3:2	rxt	R/W 0x0	RXT
1	rxd	R/W 0x0	RXD
0	rxs	R/W 0x0	RXS

A.3.2.65 PHY ID Register (PHY_ID)

Instance Name	Offset
PHY_ID	0x200

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																cid1								cid0								
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	1

Table 286: PHY ID Register (PHY_ID)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:8	cid1	R 0x94	CID1
7:0	cid0	R 0x11	CID0

A.3.2.66 PLL Control 0 Register (PLL_Control_0)

Instance Name	Offset
PLL_Control_0	0x204

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																pllvdd18		refdiv					fbdiv									
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	1	0	1	1	0	1	0	0	1	1	1	1	0	0	0

Table 287: PLL Control 0 Register (PLL_Control_0)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:14	pllvdd18	R/W 0x1	PLLVDD18
13:9	refdiv	R/W 0xD	REFDIV
8:0	fbdiv	R/W 0x78	FBDIV

A.3.2.67 PLL Control 1 Register (PLL_Control_1)

Instance Name	Offset
PLL_Control_1	0x208

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																pll_ready	pll_contrl_by_pin	pu_pll	pll_lock_bypass	dll_reset_blk	icp			kvco_ext	kvco			clk_blk_en	vcocal_start	pllcal12	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	1

Table 288: PLL Control 1 Register (PLL_Control_1)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15	pll_ready	R 0x0	PLL_READY

Table 288: PLL Control 1 Register (PLL_Control_1) (Continued)

Bits	Field	Type/ HW Rst	Description
14	pll_contrl_by_pin	R/W 0x0	pll_contrl_by_pin
13	pu_pll	R/W 0x0	pu_pll
12	pll_lock_bypass	R/W 0x0	PLL_LOCK_BYPASS
11	dll_reset_blk	R/W 0x0	DLL_RESET_BLK
10:8	icp	R/W 0x2	ICP
7	kvco_ext	R/W 0x0	KVCO_EXT
6:4	kvco	R/W 0x3	KVCO
3	clk_blk_en	R/W 0x0	CLK_BLK_EN
2	vcocal_start	R/W 0x0	VCOCAL_START
1:0	pllcal12	R/W 0x1	PLLCAL12

A.3.2.68 Reserved_Addr3 Register (Reserved_Addr3)

Instance Name	Offset
Reserved_Addr3	0x20C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																reserved_bit_15_0																
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 289: Reserved_Addr3 Register (Reserved_Addr3)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	reserved_bit_15_0	R 0x0	Reserved_Bit_15_0

A.3.2.69 TX Channel Control 0 Register (Tx_Channel_Contrl_0)

Instance Name	Offset
Tx_Channel_Contrl_0	0x210

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																nd	txdata_block_en	rcal_start	ext_hs_rcal_en	ext_fs_rcal_en	impcal_vth			ext_hs_rcal				ext_fs_rcal			
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0

Table 290: TX Channel Control 0 Register (Tx_Channel_Contrl_0)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15	nd	R 0x0	ND
14	txdata_block_en	R/W 0x0	TXDATA_BLOCK_EN
13	rcal_start	R/W 0x0	RCAL_START
12	ext_hs_rcal_en	R/W 0x0	EXT_HS_RCAL_EN
11	ext_fs_rcal_en	R/W 0x0	EXT_FS_RCAL_EN
10:8	impcal_vth	R/W 0x4	IMPCAL_VTH
7:4	ext_hs_rcal	R/W 0x8	EXT_HS_RCAL
3:0	ext_fs_rcal	R/W 0x8	EXT_FS_RCAL

A.3.2.70 TX Channel Control 1 Register (Tx_Channel_Contrl_1)

Instance Name	Offset
Tx_Channel_Contrl_1	0x214

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																nd				txvdd15		txvdd12		lowvdd_en	amp			ck60_phsel			
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	1	0	1	1	0	1	1	0	0	0	0

Table 291: TX Channel Control 1 Register (Tx_Channel_Contrl_1)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:12	nd	R 0x0	ND
11:10	txvdd15	R/W 0x1	TXVDD15
9:8	txvdd12	R/W 0x1	TXVDD12
7	lowvdd_en	R/W 0x1	LOWVDD_EN
6:4	amp	R/W 0x3	AMP
3:0	ck60_phsel	R/W 0x0	CK60_PHSEL

A.3.2.71 TX Channel Control 2 Register (Tx_Channel_Contrl_2)

Instance Name	Offset
Tx_Channel_Contrl_2	0x218

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																nd				drv_slewrate		imp_cal_dly		fsdrv_en				hsdrv_en			
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1

Table 292: TX Channel Control 2 Register (Tx_Channel_Contrl_2)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:12	nd	R 0x0	ND
11:10	drv_slewrate	R/W 0x0	DRV_SLEWRATE
9:8	imp_cal_dly	R/W 0x2	IMP_CAL_DLY
7:4	fsdrv_en	R/W 0xF	FSDRV_EN
3:0	hsdrv_en	R/W 0xF	HSDRV_EN

A.3.2.72 Reserved_Addr7 Register (Reserved_Addr7)

Instance Name	Offset
Reserved_Addr7	0x21C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																reserved_bit_15_0																
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 293: Reserved_Addr7 Register (Reserved_Addr7)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	reserved_bit_15_0	R 0x0	Reserved_Bit_15_0

A.3.2.73 RX Channel Control 0 Register (Rx_Channel_Contrl_0)

Instance Name	Offset
Rx_Channel_Contrl_0	0x220

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																phase_freeze_dly	usq_length	acq_length		sq_length		discon_thresh		sq_thresh				lpf_coef		intpi	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	0	1	0	1	0	1	0	0	1	1	1	0	0	0	1

Table 294: RX Channel Control 0 Register (Rx_Channel_Contrl_0)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15	phase_freeze_dly	R/W 0x1	PHASE_FREEZE_DLY
14	usq_length	R/W 0x0	USQ_LENGTH
13:12	acq_length	R/W 0x2	ACQ_LENGTH
11:10	sq_length	R/W 0x2	SQ_LENGTH

Table 294: RX Channel Control 0 Register (Rx_Channel_Contrl_0) (Continued)

Bits	Field	Type/ HW Rst	Description
9:8	discon_thresh	R/W 0x2	DISCON_THRESH
7:4	sq_thresh	R/W 0x7	SQ_THRESH
3:2	lpf_coef	R/W 0x0	LPF_COEF
1:0	intpi	R/W 0x1	INTPI

A.3.2.74 RX Channel Control 1 Register (Rx_Channel_Contrl_1)

Instance Name	Offset
Rx_Channel_Contrl_1	0x224

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																nd		early_vos_on_en	rxdata_block_en	edge_det_en	cap_sel			rxdata_block_length		edge_det_sel		cdr_coef_sel	cdr_fastlock_en	s2to3_dly_sel	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	1	1	1	0	0	0	1	0	0	1	0	0	1	0

Table 295: RX Channel Control 1 Register (Rx_Channel_Contrl_1)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:14	nd	R 0x0	ND
13	early_vos_on_en	R/W 0x1	EARLY_VOS_ON_EN
12	rxdata_block_en	R/W 0x1	RXDATA_BLOCK_EN
11	edge_det_en	R/W 0x1	EDGE_DET_EN
10:8	cap_sel	R/W 0x0	CAP_SEL

Table 295: RX Channel Control 1 Register (Rx_Channel_Contrl_1) (Continued)

Bits	Field	Type/ HW Rst	Description
7:6	rxdata_block_length	R/W 0x2	RXDATA_BLOCK_LENGTH
5:4	edge_det_sel	R/W 0x1	EDGE_DET_SEL
3	cdr_coef_sel	R/W 0x0	CDR_COEF_SEL
2	cdr_fastlock_en	R/W 0x0	CDR_FASTLOCK_EN
1:0	s2to3_dly_sel	R/W 0x2	S2TO3_DLY_SEL

A.3.2.75 RX Channel Control 2 Register (Rx_Channel_Contrl_2)

Instance Name	Offset
Rx_Channel_Contrl_2	0x228

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																nd							usq_filter	sq_cm_sel	sampler_ctrl	sq_buffer_en	sq_always_on	rxvdd18		rxvdd12	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	1

Table 296: RX Channel Control 2 Register (Rx_Channel_Contrl_2)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:9	nd	R 0x0	ND
8	usq_filter	R/W 0x1	USQ_FILTER
7	sq_cm_sel	R/W 0x0	SQ_CM_SEL
6	sampler_ctrl	R/W 0x0	SAMPLER_CTRL
5	sq_buffer_en	R/W 0x1	SQ_BUFFER_EN

Table 296: RX Channel Control 2 Register (Rx_Channel_Contrl_2) (Continued)

Bits	Field	Type/ HW Rst	Description
4	sq_always_on	R/W 0x0	SQ_ALWAYS_ON
3:2	rxvdd18	R/W 0x1	RXVDD18
1:0	rxvdd12	R/W 0x1	RXVDD12

A.3.2.76 ANA Control 0 Register (Ana_Contrl_0)

Instance Name	Offset
Ana_Contrl_0	0x230

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																nd						bg_vsel		dig_sel		topvdd18		vdd_usb2_dig_top_sel	iptat_sel		
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0

Table 297: ANA Control 0 Register (Ana_Contrl_0)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:10	nd	R 0x0	ND
9:8	bg_vsel	R/W 0x1	BG_VSEL
7:6	dig_sel	R/W 0x0	DIG_SEL
5:4	topvdd18	R/W 0x1	TOPVDD18
3	vdd_usb2_dig_top_sel	R/W 0x0	VDD_USB2_DIG_TOP_SEL
2:0	iptat_sel	R/W 0x0	IPTAT_SEL

A.3.2.77 ANA Control 1 Register (Ana_Contrl_1)

Instance Name	Offset
Ana_Contrl_1	0x234

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																nd	pu_ana	ana_contrl_by_pin	sel_lpfr	v2i_ext	v2i			r_rotate_sel	stress_test_mode	testmon_ana						
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	1	0	1	1	0	1	0	0	0	0	0	0	0

Table 298: ANA Control 1 Register (Ana_Contrl_1)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15	nd	R 0x0	ND
14	pu_ana	R/W 0x0	PU_ANA
13	ana_contrl_by_pin	R/W 0x0	ANA_CONTrL_BY_PIN
12	sel_lpfr	R/W 0x1	SEL_LPFR
11	v2i_ext	R/W 0x0	V2I_EXT
10:8	v2i	R/W 0x6	V2I
7	r_rotate_sel	R/W 0x1	R_ROTATE_SEL
6	stress_test_mode	R/W 0x0	STRESS_TEST_MODE
5:0	testmon_ana	R/W 0x0	TESTMON_ANA

A.3.2.78 Reserved_Addr_C Register (Reserved_Addr_C)

Instance Name	Offset
Reserved_Addr_C	0x238

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																reserved_bit_15_0																
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 299: Reserved_Addr_C Register (Reserved_Addr_C)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	reserved_bit_15_0	R/W 0x0	Reserved_Bit_15_0

A.3.2.79 Digital Control 0 Register (Digital_Control_0)

Instance Name	Offset
Digital_Control_0	0x23C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																fifo_uf	fifo_ov	fs_eop_mode	host_discon_sel1	host_discon_sel0	force_end_en	early_tx_en	syncdet_window_en	clk_suspend_en	hs_dribble_en	sync_num		fifo_fill_num			
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	1	0	0	1	0	1	1	0	0	0	0	1	1	0

Table 300: Digital Control 0 Register (Digital_Control_0)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15	fifo_uf	R 0x0	FIFO_UF
14	fifo_ov	R 0x0	FIFO_OV
13	fs_eop_mode	R/W 0x1	FS_EOP_MODE

Table 300: Digital Control 0 Register (Digital_Control_0) (Continued)

Bits	Field	Type/ HW Rst	Description
12	host_discon_sel1	R/W 0x0	HOST_DISCON_SEL1
11	host_discon_sel0	R/W 0x0	HOST_DISCON_SEL0
10	force_end_en	R/W 0x1	FORCE_END_EN
9	early_tx_en	R/W 0x0	EARLY_TX_EN
8	syncdet_window_en	R/W 0x1	SYNCDET_WINDOW_EN
7	clk_suspend_en	R/W 0x1	CLK_SUSPEND_EN
6	hs_dribble_en	R/W 0x0	HS_DRIBBLE_EN
5:4	sync_num	R/W 0x0	SYNC_NUM
3:0	fifo_fill_num	R/W 0x6	FIFO_FILL_NUM

A.3.2.80 Digital Control 1 Register (Digital_Control_1)

Instance Name	Offset
Digital_Control_1	0x240

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																fs_rx_error_mode2	fs_rx_error_mode1	fs_rx_error_mode	clk_out_sel	ext_tx_clk_sel	arc_dpdm_mode	dp_pulldown	dm_pulldown	sync_ignore_sq	sq_rst_rx	mon_sel						
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0

Table 301: Digital Control 1 Register (Digital_Control_1)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15	fs_rx_error_mode2	R/W 0x1	FS_RX_ERROR_MODE2
14	fs_rx_error_mode1	R/W 0x1	FS_RX_ERROR_MODE1
13	fs_rx_error_mode	R/W 0x1	FS_RX_ERROR_MODE
12	clk_out_sel	R/W 0x0	CLK_OUT_SEL
11	ext_tx_clk_sel	R/W 0x0	EXT_TX_CLK_SEL
10	arc_dpdm_mode	R/W 0x1	ARC_DPDM_MODE
9	dp_pulldown	R/W 0x0	DP_PULLDOWN
8	dm_pulldown	R/W 0x0	DM_PULLDOWN
7	sync_ignore_sq	R/W 0x0	SYNC_IGNORE_SQ
6	sq_rst_rx	R/W 0x0	SQ_RST_RX
5:0	mon_sel	R/W 0x0	MON_SEL

A.3.2.81 Digital Control 2 Register (Digital_Control_2)

Instance Name	Offset
Digital_Control_2	0x244

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																nd_15_13			pad_strength						nd		long_eop	novbus_dpdm00	disable_el16	align_fs_outen	hs_hdl_sync	fs_hdl_opmd
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	1	1	1	1	0	0	0	1	0	0	1	1

Table 302: Digital Control 2 Register (Digital_Control_2)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:13	nd_15_13	R 0x0	ND_15_13
12:8	pad_strength	R/W 0xF	PAD_STRENGTH
7:6	nd	R 0x0	ND
5	long_eop	R/W 0x0	LONG_EOP
4	novbus_dpdm00	R/W 0x1	novbus_dpdm00
3	disable_el16	R/W 0x0	DISABLE_EL16
2	align_fs_outen	R/W 0x0	ALIGN_FS_OUTEN
1	hs_hdl_sync	R/W 0x1	HS_HDL_SYNC
0	fs_hdl_opmd	R/W 0x1	FS_HDL_OPMD

A.3.2.82 Reserved_Addr_12H Register (Reserved_Addr_12H)

Instance Name	Offset
Reserved_Addr_12H	0x248

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																reserved_bit_15_0															
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 303: Reserved_Addr_12H Register (Reserved_Addr_12H)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	reserved_bit_15_0	R 0x0	Reserved_Bit_15_0

A.3.2.83 Test Control and Status 0 Register (Test_Contrl_and_Status_0)

Instance Name	Offset
Test_Contrl_and_Status_0	0x24C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																test_dig_lpbk	test_ana_lpbk	test_length[1:0]		test_bypass	test_mode[2:0]			test_tx_pattern							
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 304: Test Control and Status 0 Register (Test_Contrl_and_Status_0)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15	test_dig_lpbk	R/W 0x0	TEST_DIG_LPBK
14	test_ana_lpbk	R/W 0x0	TEST_ANA_LPBK
13:12	test_length[1:0]	R/W 0x0	TEST_LENGTH[1:0]
11	test_bypass	R/W 0x0	TEST_BYPASS
10:8	test_mode[2:0]	R/W 0x0	TEST_MODE[2:0]
7:0	test_tx_pattern	R/W 0x0	TEST_TX_PATTERN

A.3.2.84 Test Control and Status 1 Register (Test_Contrl_and_Status_1)

Instance Name	Offset
Test_Contrl_and_Status_1	0x250

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																test_done	test_flag	test_en	test_reset	nd	test_skip[2:0]			test_utmi_sel	test_suspendm	test_tx_bitstuff_en	test_term_select	test_op_mode		test_xcvr_select	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0

Table 305: Test Control and Status 1 Register (Test_Contrl_and_Status_1)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15	test_done	R 0x0	TEST_DONE
14	test_flag	R 0x0	TEST_FLAG
13	test_en	R/W 0x0	TEST_EN
12	test_reset	R/W 0x0	TEST_RESET
11	nd	R/W 0x0	ND
10:8	test_skip[2:0]	R/W 0x0	TEST_SKIP[2:0]
7	test_utmi_sel	R/W 0x0	TEST_UTMI_SEL
6	test_suspendm	R/W 0x1	TEST_SUSPENDM
5	test_tx_bitstuff_en	R/W 0x1	TEST_TX_BITSTUFF_EN
4	test_term_select	R/W 0x0	TEST_TERM_SELECT
3:2	test_op_mode	R/W 0x0	TEST_OP_MODE
1:0	test_xcvr_select	R/W 0x0	TEST_XCVR_SELECT

A.3.2.85 Reserved_Addr_15H Register (Reserved_Addr_15H)

Instance Name	Offset
Reserved_Addr_15H	0x254

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																reserved_bit_15_0																
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 306: Reserved_Addr_15H Register (Reserved_Addr_15H)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	reserved_bit_15_0	R 0x0	Reserved_Bit_15_0

A.3.2.86 PHY REG CHGDTC Control Register (PHY_REG_CHGDTC_CONTRL)

Instance Name	Offset
PHY_REG_CHGDTC_CONTRL	0x258

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																nd												enable_switch	pu_chrg_dtc	testmon_chrgdtc	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 307: PHY REG CHGDTC Control Register (PHY_REG_CHGDTC_CONTRL)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:4	nd	R 0x0	ND
3	enable_switch	R/W 0x0	ENABLE_SWITCH
2	pu_chrg_dtc	R/W 0x0	PU_CHRG_DTC
1:0	testmon_chrgdtc	R/W 0x0	TESTMON_CHRGDTC

A.3.2.87 PHY REG OTG Control Register (PHY_REG_OTG_CONTROL)

Instance Name	Offset
PHY_REG_OTG_CONTROL	0x25C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																nd											otg_control_by_pin	pu_otg	testmon_otg[2:0]		
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 308: PHY REG OTG Control Register (PHY_REG_OTG_CONTROL)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:5	nd	R 0x0	ND
4	otg_control_by_pin	R/W 0x0	OTG_CONTROL_BY_PIN
3	pu_otg	R/W 0x0	PU_OTG
2:0	testmon_otg[2:0]	R/W 0x0	TESTMON_OTG[2:0]

A.3.2.88 USB2 PHY Monitor 0 Register (usb2_phy_mon0)

Instance Name	Offset
usb2_phy_mon0	0x260

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

Table 309: USB2 PHY Monitor 0 Register (usb2_phy_mon0)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	phy_mon	R/W 0x0	PHY_MON

A.3.2.89 PHY REG CHGDTC Control 1 Register (PHY_REG_CHGDTC_ CONTRL_1)

Instance Name	Offset
PHY_REG_CHGDTC_CONTRL_1	0x264

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																dp_dm_swap_ctrl	reserved_14	pllvdd12		vsrc_charge		vdat_charge		enable_switch_dp	enable_switch_dm	cdp_dm_auto_switch	pd_en	dcp_en	cdp_en	reserved_0	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 310: PHY REG CHGDTC Control 1 Register (PHY_REG_CHGDTC_CONTRL_1)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15	dp_dm_swap_ctrl	R/W 0x0	DP_DM_SWAP_CTRL
14	reserved_14	R 0x0	reserved_14
13:12	pllvdd12	R/W 0x0	PLLVDD12

Table 310: PHY REG CHGDTC Control 1 Register (PHY_REG_CHGDTC_CONTRL_1) (Continued)

Bits	Field	Type/ HW Rst	Description
11:10	vsrc_charge	R/W 0x0	VSRC_CHARGE
9:8	vdat_charge	R/W 0x0	VDAT_CHARGE
7	enable_switch_dp	R/W 0x0	ENABLE_SWITCH_DP
6	enable_switch_dm	R/W 0x0	ENABLE_SWITCH_DM
5	cdp_dm_auto_switch	R/W 0x0	CDP_DM_AUTO_SWITCH
4	pd_en	R/W 0x0	PD_EN
3	dcp_en	R/W 0x0	DCP_EN
2	cdp_en	R/W 0x0	CDP_EN
1:0	reserved_0	R 0x0	Reserved_0

A.3.2.90 Reserved_Addr_1AH Register (Reserved_Addr_1aH)

Instance Name	Offset
Reserved_Addr_1aH	0x268

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																reserved_bit_15_0															
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 311: Reserved_Addr_1AH Register (Reserved_Addr_1aH)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	reserved_bit_15_0	R 0x0	Reserved_Bit_15_0

A.3.2.91 Reserved_Addr_1BH Register (Reserved_Addr_1bH)

Instance Name	Offset
Reserved_Addr_1bH	0x26C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																reserved_bit_15_0																
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 312: Reserved_Addr_1BH Register (Reserved_Addr_1bH)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	reserved_bit_15_0	R 0x0	Reserved_Bit_15_0

A.3.2.92 Reserved_Addr_1CH Register (Reserved_Addr_1cH)

Instance Name	Offset
Reserved_Addr_1cH	0x270

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																reserved_bit_15_0																
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 313: Reserved_Addr_1CH Register (Reserved_Addr_1cH)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	reserved_bit_15_0	R 0x0	Reserved_Bit_15_0

A.3.2.93 Reserved_Addr_1DH Register (Reserved_Addr_1dH)

Instance Name	Offset
Reserved_Addr_1dH	0x274

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																reserved_bit_15_0															
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 314: Reserved_Addr_1DH Register (Reserved_Addr_1dH)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	reserved_bit_15_0	R 0x0	Reserved_Bit_15_0

A.3.2.94 Internal CID Register (Internal_CID)

Instance Name	Offset
Internal_CID	0x278

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																icid0								icid1							
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	1

Table 315: Internal CID Register (Internal_CID)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:8	icid0	R 0x94	ICID0
7:0	icid1	R 0x11	ICID1

A.3.2.95 USB2 ICID Register 1 (usb2_icid_reg1)

Instance Name	Offset
usb2_icid_reg1	0x27C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																icid2								phy_otg	phy_chg_dtc	phy_hsic	phy_ulpi	dig_regulator	nd		phy_multiport
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Table 316: USB2 ICID Register 1 (usb2_icid_reg1)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:8	icid2	R 0x0	ICID2
7	phy_otg	R 0x1	PHY_OTG
6	phy_chg_dtc	R 0x0	PHY_CHG_DTC
5	phy_hsic	R 0x0	PHY_HSIC
4	phy_ulpi	R 0x0	PHY_ULPI
3	dig_regulator	R 0x0	DIG_REGULATOR
2:1	nd	R 0x0	ND
0	phy_multiport	R 0x0	PHY_MULTIPORT

A.4 Flash Controller Address Block

A.4.1 Flash Controller Register Map

Table 317: Flash Controller Register Map

Offset	Name	HW Rst	Description	Details
0x00	FCCR	0x2000_0205	Flash Controller Configuration Register	Page: 506
0x04	FCTR	0x0000_0005	Flash Controller Timing Register	Page: 507
0x08	FCSR	0x0000_0000	Flash Controller Status Register	Page: 508
0x0C	FCACR	0x0000_0000	Flash Controller Auxiliary Configuration Register	Page: 509
0x10	FCHCR	0x0000_0000	Flash Controller Hit Count Register	Page: 510
0x14	FCMCR	0x0000_0000	Flash Controller Miss Count Register	Page: 510
0x18	FAOFFR	0x0000_0000	Flash Address Offset Register	Page: 511
0x1C	FADDMAT	0x0000_0000	Flash Address Match Register	Page: 511
0x20	FWAITR	0x0000_0001	Flash Wait Register	Page: 512
0x24	FCCR2	0x0000_0000	Flash Controller Configuration Register2	Page: 512
0x28	FINSTR	0x0000_0000	Flash Instruction Register	Page: 514
0x2C	FRMR	0x0000_0000	Flash Read Mode Register	Page: 515

A.4.2 Flash Controller Registers

A.4.2.1 Flash Controller Configuration Register (FCCR)

Instance Name	Offset
FCCR	0x00

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	flashc_pad_en	cache_en	cache_line_flush	sram_mode_en	Reserved												clk_pha	clk_pol	Reserved	clk_prescale					Reserved				cmd_type			
HW Rst	0	0	1	0	?	?	?	?	?	?	?	?	?	?	?	?	0	0	?	0	0	0	1	0	?	?	?	?	0	1	0	1

Table 318: Flash Controller Configuration Register (FCCR)

Bits	Field	Type/ HW Rst	Description
31	flashc_pad_en	R/W 0x0	Flashc Pad Enable Controls the mux between AHB Flashc & APB QSPI. 0x0 = APB QSPI connects to the Flash device 0x1 = Flashc connects to the Flash device
30	cache_en	R/W 0x0	Flash Cache Enable
29	cache_line_flush	R/W 0x1	Cache Line Flush
28	sram_mode_en	R/W 0x0	SRAM Mode Enable
27:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15	clk_pha	R/W 0x0	Serial Interface Clock Phase Selects the serial interface clock phase. 0x0 = data is captured on the rising edge of the serial clock when CLK_POL=0 and on the falling edge when CLK_POL=1 0x1 = data is captured on the falling edge of the serial clock when CLK_POL=0 and on the rising edge when CLK_POL=1
14	clk_pol	R/W 0x0	Serial Interface Clock Polarity Selects the Serial Interface Clock as HIGH or LOW when inactive. 0x0 = serial interface clock is LOW when inactive 0x1 = serial interface clock is HIGH when inactive
13	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
12:8	clk_prescale	R/W 0x2	Serial Interface Clock Prescaler (from Base SPI clock)

Table 318: Flash Controller Configuration Register (FCCR) (Continued)

Bits	Field	Type/ HW Rst	Description
7:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3:0	cmd_type	R/W 0x5	Serial Flash Command Typeclocks (based on This Command Type Field for Winbond devices) The Flash Controller will automatically build the necessary Instruction, followed by Address, followed by dummy.

A.4.2.2 Flash Controller Timing Register (FCTR)

Instance Name	Offset
FCTR	0x04

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

Table 319: Flash Controller Timing Register (FCTR)

Bits	Field	Type/ HW Rst	Description
31:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4	clk_capt_edge	R/W 0x1	Serial Interface Capture Clock Edge Capture serial interface input data on either the rising or falling edge of the serial interface clock. This bit is used to allow more time to capture the input data. 0x0 = • When CLK_POL (R04 [8]) = 0 and CLK_PHA (R04 [7]) = 0, capture input data on rising edge of the serial interface clock • When CLK_POL (R04 [8]) = 0 and CLK_PHA (R04 [7]) = 1, capture input data on falling edge of the serial interface clock • When CLK_POL (R04 [8]) = 1 and CLK_PHA (R04 [7]) = 0, capture input data on falling edge of the serial interface clock • When CLK_POL (R04 [8]) = 1 and CLK_PHA (R04 [7]) = 1, capture input data on rising edge of the serial interface clock 0x1 = • When CLK_POL (R04 [8]) = 0 and CLK_PHA (R04 [7]) = 0, capture input data on falling edge of the serial interface clock • When CLK_POL (R04 [8]) = 0 and CLK_PHA (R04 [7]) = 1, capture input data on rising edge of the serial interface clock When CLK_PHA (R04 [7]) = 1, this bit must be set to 0.

Table 319: Flash Controller Timing Register (FCTR) (Continued)

Bits	Field	Type/ HW Rst	Description
3:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.4.2.3 Flash Controller Status Register (FCSR)

Instance Name	Offset
FCSR	0x08

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

Table 320: Flash Controller Status Register (FCSR)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	cont_rd_md_exit_done	R/W1CLR 0x0	Continuous Read Mode Exit Status Software needs to check this bit after issuing Continuous Read Mode Exit command (CMD_TYPE = 4'hC OR 4'hD). This is a sticky bit, and software needs to write one to clear. 0x0 = continuous read mode exit not complete 0x1 = continuous read mode exit complete

A.4.2.4 Flash Controller Auxiliary Configuration Register (FCACR)

Instance Name	Offset
FCACR	0x0C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																												offset_en	addr_match_en	miss_cnt_en	hit_cnt_en
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0

Table 321: Flash Controller Auxiliary Configuration Register (FCACR)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3	offset_en	R/W 0x0	Address Offset Enable 0x0 = all Flash memory accesses do not use address offset register 0x1 = using address offset defined in FAOFFR is enabled for all Flash memory accesses
2	addr_match_en	R/W 0x0	Address Match Enable Using in conjunction with MISS_CNT_EN bit of this register. 0x0 = disable counting of misses to a specific address 0x1 = enable counting of misses to a specific address defined in FADDMAT
1	miss_cnt_en	R/W 0x0	Miss Count Enable 0x0 = disable the counting of misses 0x1 = enable the counting of number of misses to the cache
0	hit_cnt_en	R/W 0x0	Miss Count Enable 0x0 = disable the counting of hits 0x1 = enable the counting of number of hits to the cache

A.4.2.5 Flash Controller Hit Count Register (FCHCR)

Instance Name	Offset
FCHCR	0x10

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

Table 322: Flash Controller Hit Count Register (FCHCR)

Bits	Field	Type/ HW Rst	Description
31:0	hit_count	R/WCLR 0x0	Hit Counter When FCACR.HIT_CNT_EN=1 this counter tracks the number of hits that have occurred to the Flash Cache. Software can read this register to understand the hits. This register is cleared when software does a write to this register.

A.4.2.6 Flash Controller Miss Count Register (FCMCR)

Instance Name	Offset
FCMCR	0x14

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

Table 323: Flash Controller Miss Count Register (FCMCR)

Bits	Field	Type/ HW Rst	Description
31:0	miss_count	R/WCLR 0x0	Hit Counter When FCACR.MISS_CNT_EN=1 this counter tracks the number of hits that have occurred to the Flash Cache. If FCACR.ADDR_ MATCH_EN=1 AND FCACR.MISS_CNT_EN=1 then misses to the specific address configured into FADDMAT register is captured into this counter. Software can read this register to understand the misses. This register is cleared when software does a write to this register.

A.4.2.7 Flash Address Offset Register (FAOFFR)

Instance Name	Offset
FAOFFR	0x18

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

Table 324: Flash Address Offset Register (FAOFFR)

Bits	Field	Type/ HW Rst	Description
31:0	offset_val	R/W 0x0	Flash Address Offset Value When FCACR.OFFSET_EN=1 then this register specifies the address offset from Flash base address that is used for all Flash memory accesses.

A.4.2.8 Flash Address Match Register (FADDMAT)

Instance Name	Offset
FADDMAT	0x1C

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

Table 325: Flash Address Match Register (FADDMAT)

Bits	Field	Type/ HW Rst	Description
31:0	addr	R/W 0x0	Flash Memory Address to Compare and Match When FCACR.ADDR_MATCH_EN=1 and FCACR.MISS_CNT_EN=1 then all misses to the address in this register are captured into the FCMCR.

A.4.2.9 Flash Wait Register (FWAITR)

Instance Name	Offset
FWAITR	0x20

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

Table 326: Flash Wait Register (FWAITR)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	zwait	R/W 0x1	Zero Wait Enable the 2 bus cycles read for a cache-hit. Otherwise, 3 bus cycles are needed. This feature is not supported when FlashC is running faster than 100 MHz.

A.4.2.10 Flash Controller Configuration Register2 (FCCR2)

Instance Name	Offset
FCCR2	0x24

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	use_cfg_ovrd	data_pin		addr_pin	byte_len	Reserved													dummy_cnt		Reserved		rm_cnt		Reserved	addr_cnt			Reserved		instr_cnt	
HW Rst	0	0	0	0	0	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	?	?	0	0	?	0	0	0	?	?	0	0

Table 327: Flash Controller Configurationb Register2 (FCCR2)

Bits	Field	Type/ HW Rst	Description
31	use_cfg_ovrd	R/W 0x0	Use Configuration Override This bit, when set by software, overrides the built-in hardware Flash transfer generation defined through FCCR.CMD_TYPE. Software has control over Instruction, Address, and other configuration. Software needs to program all the necessary fields in FCCR2 to successfully complete a transfer to Flash. 0x0 = use FCCR.CMD_TYPE to determine/build Flash command/transfer 0x1 = use configuration FCCR2 to determine/build Flash command/transfer

Table 327: Flash Controller Configurationb Register2 (FCCR2) (Continued)

Bits	Field	Type/ HW Rst	Description
30:29	data_pin	R/W 0x0	Data Transfer Pins Number of pins used to transfer data. 0x0 = use 1 pin for data 0x1 = use 2 pins for data 0x2 = use 4 pins for data 0x3 = reserved
28	addr_pin	R/W 0x0	Address Transfer Pins Number of pins used to transfer the Flash address. 0x0 = use one pin 0x1 = use number of pins as indicated by DATA_PIN field in this register
27	byte_len	R/W 0x0	Byte Length Number of bytes in each serial transfer. 0x0 = 1 byte 0x1 = 4 bytes
26:14	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
13:12	dummy_cnt	R/W 0x0	Dummy Count Defines the number of dummy bytes that need to be sent to Flash. The data shifted out is always 0. 0x0 = 0 bytes 0x1 = 1 byte 0x2 = 2 bytes 0x3 = reserved
11:10	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
9:8	rm_cnt	R/W 0x0	Read Mode Count Number of Read Mode bytes (as defined in FRMR) that are sent out to Flash. 0x0 = 0 bytes 0x1 = 1 byte 0x2 = 2 bytes 0x3 = reserved
7	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
6:4	addr_cnt	R/W 0x0	Address Count Number of bytes of address that is sent to Flash. 0x0 = 0 bytes 0x1 = 1 byte 0x2 = 2 bytes 0x3 = 3 bytes 0x4 = 4 bytes 0x5 to 0x7 = reserved

Table 327: Flash Controller Configurationb Register2 (FCCR2) (Continued)

Bits	Field	Type/ HW Rst	Description
3:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1:0	instr_cnt	R/W 0x0	Instruction Count Number of bytes of Instruction (defined in FINSTR) to send to Flash. 0x0 = 0 bytes 0x1 = 1 byte 0x2 = 2 bytes 0x3 = reserved

A.4.2.11 Flash Instruction Register (FINSTR)

Instance Name	Offset
FINSTR	0x28

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

Table 328: Flash Instruction Register (FINSTR)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	instr	R/W 0x0	Flash Instruction The contents of this register define the instruction Op code that gets sent to the Flash device. When FCCR2.INSTR_CNT=0, this register content is not sent out. When FCCR2.INSTR_CNT=1, bits [7:0] of this register are sent out. When FCCR.INSTR_CNT=2, bits [15:8] of this register are sent out first followed by bits [7:0].

A.4.2.12 Flash Read Mode Register (FRMR)

Instance Name	Offset
FRMR	0x2C

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

Table 329: Flash Read Mode Register (FRMR)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	rdmode	R/W 0x0	Flash Read Mode The contents of this register determine the Read Mode information that gets sent to the Flash device after Address cycle. When FCCR2.RM_CNT=0, this register content is not sent out. When FCCR2.RM_CNT=1, bits [7:0] of this register are sent out. When FCCR.RM_CNT=2, bits [15:8] of this register are sent out first followed by bits [7:0].

THIS PAGE INTENTIONALLY LEFT BLANK

A.5 AES Address Block

A.5.1 AES Register Map

Table 330: AES Register Map

Offset	Name	HW Rst	Description	Details
0x00	ctrl1	0x0000_4010	AES Control Register 1	Page: 518
0x04	ctrl2	0x0000_0000	AES Control Register 2	Page: 520
0x08	status	0x0000_0081	AES Status Register	Page: 521
0x0C	astr_len	0x0000_0000	AES Astr Length Register	Page: 522
0x10	mstr_len	0x0000_0000	AES Mstr Length Register	Page: 523
0x14	str_in	0x0000_0000	AES Stream Input Register	Page: 523
0x18	iv0	0x0000_0000	AES Input Vector Register 0	Page: 524
0x1C	iv1	0x0000_0000	AES Input Vector Register 1	Page: 524
0x20	iv2	0x0000_0000	AES Input Vector Register 2	Page: 525
0x24	iv3	0x0000_0000	AES Input Vector Register 3	Page: 525
0x28	key0	0x0000_0000	AES Key 0 Register	Page: 526
0x2C	key1	0x0000_0000	AES Key 1 Register	Page: 526
0x30	key2	0x0000_0000	AES Key 2 Register	Page: 527
0x34	key3	0x0000_0000	AES Key 3 Register	Page: 527
0x38	key4	0x0000_0000	AES Key 4 Register	Page: 528
0x3C	key5	0x0000_0000	AES Key 5 Register	Page: 528
0x40	key6	0x0000_0000	AES Key 6 Register	Page: 529
0x44	key7	0x0000_0000	AES Key 7 Register	Page: 529
0x48	str_out	0x0000_0000	AES Stream Output Port Register	Page: 530
0x4C	ov0	0x0000_0000	AES Output Vector 0 Register	Page: 530
0x50	ov1	0x0000_0000	AES Output Vector 1 Register	Page: 531
0x54	ov2	0x0000_0000	AES Output Vector 2 Register	Page: 531
0x58	ov3	0x0000_0000	AES Output Vector 3 Register	Page: 532
0x5C	isr	0x0000_0000	AES Interrupt Status Register	Page: 532
0x60	imr	0x0000_0007	AES Interrupt Mask Register	Page: 533

Table 330: AES Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x64	irsr	0x0000_0000	AES Interrupt Raw Status Register	Page: 534
0x68	icr	0x0000_0000	AES Interrupt Clear Register	Page: 535
0x8C	rev_id	0x0000_0012	AES Revision Register	Page: 535

A.5.2 AES Registers

A.5.2.1 AES Control Register 1 (ctrl1)

Instance Name	Offset
ctrl1	0x00

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved					cts_mode	ctr_mod							mode			decrypt	out_mic	mic_len		key_size		dma_en	io_src	pri1	pri0	out_hdr	out_msg	of_clr	if_clr	lock0	start
HW Rst	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Table 331: AES Control Register 1 (ctrl1)

Bits	Field	Type/ HW Rst	Description
31:27	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
26	cts_mode	R/W 0x0	Cipher Stealing Mode of CBC 0x0 = marvell mode 0x1 = NIST-CS2 mode
25:19	ctr_mod	R/W 0x0	CTR Mode's Counter Modular modular = 2^128: [7'h0-7'hF] modular = 2^ctr_mod: others
18:16	mode	R/W 0x0	AES Running Mode 0x0 = ECB 0x1 = CBC 0x2 = CTR 0x3 = RESERVED 0x4 = RESERVED 0x5 = CCM* 0x6 = MMO 0x7 = BYPASS
15	decrypt	R/W 0x0	Decrypt Operation (ignored in MMO and BYPASS mode) 0x0 = encryption 0x1 = decryption

Table 331: AES Control Register 1 (ctrl1) (Continued)

Bits	Field	Type/ HW Rst	Description
14	out_mic	R/W 0x1	Append MIC/HASH at the End of Output Stream in CCM* Mode decryption/MMO mode. 0x0 = not append MIC/HASH at the end of output stream in CCM* mode decryption or MMO mode 0x1 = append MIC/HASH at the end of output stream in CCM* mode decryption or MMO mode
13:12	mic_len	R/W 0x0	Length of MIC Field 0x0 = 0 bytes 0x1 = 4 bytes 0x2 = 8 bytes 0x3 = 16 bytes
11:10	key_size	R/W 0x0	Key Size Parameter 0x0 = 16 bytes 0x1 = 32 bytes 0x2 = 24 bytes 0x3 = reserved
9	dma_en	R/W 0x0	Enable DMA 0x0 = disable DMA 0x1 = enable DMA
8	io_src	R/W 0x0	AES Data Input Source 0x0 = i/O through register 0x1 = i/O through DMA
7	pri1	R/W 0x0	AES Priority on Hardware (BH-MAC) Side 0x0 = low priority 0x1 = high priority
6	pri0	R/W 0x0	AES Priority on MCU (software) Side 0x0 = low priority 0x1 = high priority
5	out_hdr	R/W 0x0	Output B0 and l(a) in CCM* Mode 0x0 = do not output B0 and l(a) at the beginning of output stream 0x1 = output B0 and l(a) at the beginning of output stream
4	out_msg	R/W 0x1	Output Stream to Output FIFO 0x0 = block output stream from output FIFO 0x1 = forward output stream to output FIFO
3	of_clr	R/W 0x0	Clear Output FIFO
2	if_clr	R/W 0x0	Clear Input FIFO

Table 331: AES Control Register 1 (ctrl1) (Continued)

Bits	Field	Type/ HW Rst	Description
1	lock0	R/W 0x0	Lock AES on MCU Side 0x0 = write 0 to release AES 0x1 = write 1 to lock AES on behalf of MCU
0	start	R/W 0x0	Start AES

A.5.2.2 AES Control Register 2 (ctrl2)

Instance Name	Offset
ctrl2	0x04

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

Table 332: AES Control Register 2 (ctrl2)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	auto_reset_en	R/W 0x0	Enable Automatic Reset After Lock Successfully 0x0 = no automatic reset 0x1 = automatic reset after lock successfully
0	aes_reset	R/W 0x0	Reset AES 0x0 = un-reset AES 0x1 = reset AES

A.5.2.3 AES Status Register (status)

Instance Name	Offset
status	0x08

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved											rsvd_vld	of_depth			if_depth			status			Reserved			of_empty	of_rdy	Reserved	if_full	lock1	rsvd1	rsvd0	done
HW Rst	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	?	?	?	1	0	?	0	0	0	0	1

Table 333: AES Status Register (status)

Bits	Field	Type/ HW Rst	Description
31:21	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
20	rsvd_vld	R 0x0	RSVD Valid 0x0 = rsvd0 and rsvd1 not valid 0x1 = rsvd0 and rsvd1 are valid
19:17	of_depth	R 0x0	Output FIFO Depth
16:14	if_depth	R 0x0	Input FIFO Depth
13:11	status	R 0x0	AES Operation Error Status aesw_status[2]:MIC Mismatch during CCM* Decryption aesw_status[1]:Data is not multiple of 16 bytes in ECB mode or Data is more than 2^13-1 bytes in MMO mode aesw_status[0]:Input stream size less than 16 byte in ECB,CBC and CTR mode 0x0 = no operation error 0x1 = input stream size less than 16 byte in ECB,CBC and CTR mode 0x2 = data is not multiple of 16 bytes in ECB mode or data is more than 2^13-1 bytes in MMO mode 0x3 = data is not multiple of 16 bytes and less than 16 byte in ECB mode 0x4 = MIC mismatch during CCM* decryption others = not valid
10:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7	of_empty	R 0x1	Output FIFO Empty 0x0 = output FIFO is not empty 0x1 = output FIFO is empty
6	of_rdy	R 0x0	Output FIFO Is Ready to Read 0x0 = output FIFO is not ready to read 0x1 = output FIFO is ready to read

Table 333: AES Status Register (status) (Continued)

Bits	Field	Type/ HW Rst	Description
5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4	if_full	R 0x0	Input FIFO Full 0x0 = input FIFO is not full 0x1 = input FIFO is full
3	lock1	R 0x0	Lock AES on Hardware (BH-MAC) Side 0x0 = hardware unlocks AES 0x1 = hardware requests to lock AES
2	rsvd1	R 0x0	AES Is Locked by Hardware (BH-MAC) 0x0 = AES is not locked by hardware 0x1 = AES is locked by hardware
1	rsvd0	R 0x0	AES Is Locked by MCU 0x0 = AES is not locked by MCU 0x1 = AES is locked by MCU
0	done	R 0x1	AES Operation Done 0x0 = AES operation has not done yet 0x1 = AES operation done

A.5.2.4 AES Astr Length Register (astr_len)

Instance Name	Offset
astr_len	0x0C

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

Table 334: AES Astr Length Register (astr_len)

Bits	Field	Type/ HW Rst	Description
31:0	astr_len	R/W 0x0	Size of Associate String

A.5.2.5 AES Mstr Length Register (mstr_len)

Instance Name	Offset
mstr_len	0x10

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

Table 335: AES Mstr Length Register (mstr_len)

Bits	Field	Type/ HW Rst	Description
31:0	mstr_len	R/W 0x0	Size of Message String

A.5.2.6 AES Stream Input Register (str_in)

Instance Name	Offset
str_in	0x14

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

Table 336: AES Stream Input Register (str_in)

Bits	Field	Type/ HW Rst	Description
31:0	str_in	W 0x0	Input Message Word

A.5.2.7 AES Input Vector Register 0 (iv0)

Instance Name	Offset
iv0	0x18

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	iv0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 337: AES Input Vector Register 0 (iv0)

Bits	Field	Type/ HW Rst	Description
31:0	iv0	R/W 0x0	Byte 0-3 of Initial Vector

A.5.2.8 AES Input Vector Register 1 (iv1)

Instance Name	Offset
iv1	0x1C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	iv1																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 338: AES Input Vector Register 1 (iv1)

Bits	Field	Type/ HW Rst	Description
31:0	iv1	R/W 0x0	Byte 4-7 of Initial Vector

A.5.2.9 AES Input Vector Register 2 (iv2)

Instance Name	Offset
iv2	0x20

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	iv2																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 339: AES Input Vector Register 2 (iv2)

Bits	Field	Type/ HW Rst	Description
31:0	iv2	R/W 0x0	Byte 8-11 of Initial Vector

A.5.2.10 AES Input Vector Register 3 (iv3)

Instance Name	Offset
iv3	0x24

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	iv3																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 340: AES Input Vector Register 3 (iv3)

Bits	Field	Type/ HW Rst	Description
31:0	iv3	R/W 0x0	Byte 12-15 of Initial Vector

A.5.2.11 AES Key 0 Register (key0)

Instance Name	Offset
key0	0x28

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	key0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 341: AES Key 0 Register (key0)

Bits	Field	Type/ HW Rst	Description
31:0	key0	R/W 0x0	Byte 0-3 of Key

A.5.2.12 AES Key 1 Register (key1)

Instance Name	Offset
key1	0x2C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	key1																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 342: AES Key 1 Register (key1)

Bits	Field	Type/ HW Rst	Description
31:0	key1	R/W 0x0	Byte 4-7 of Key

A.5.2.13 AES Key 2 Register (key2)

Instance Name	Offset
key2	0x30

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	key2																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 343: AES Key 2 Register (key2)

Bits	Field	Type/ HW Rst	Description
31:0	key2	R/W 0x0	Byte 8-11 of Key

A.5.2.14 AES Key 3 Register (key3)

Instance Name	Offset
key3	0x34

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	key3																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 344: AES Key 3 Register (key3)

Bits	Field	Type/ HW Rst	Description
31:0	key3	R/W 0x0	Byte 12-15 of Key

A.5.2.15 AES Key 4 Register (key4)

Instance Name	Offset
key4	0x38

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	key4																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 345: AES Key 4 Register (key4)

Bits	Field	Type/ HW Rst	Description
31:0	key4	R/W 0x0	Byte 16-19 of Key

A.5.2.16 AES Key 5 Register (key5)

Instance Name	Offset
key5	0x3C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	key5																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 346: AES Key 5 Register (key5)

Bits	Field	Type/ HW Rst	Description
31:0	key5	R/W 0x0	Byte 20-23 of Key

A.5.2.17 AES Key 6 Register (key6)

Instance Name	Offset
key6	0x40

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	key6																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 347: AES Key 6 Register (key6)

Bits	Field	Type/ HW Rst	Description
31:0	key6	R/W 0x0	Byte 24-27 of Key

A.5.2.18 AES Key 7 Register (key7)

Instance Name	Offset
key7	0x44

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	key7																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 348: AES Key 7 Register (key7)

Bits	Field	Type/ HW Rst	Description
31:0	key7	R/W 0x0	Byte 28-31 of Key

A.5.2.19 AES Stream Output Port Register (str_out)

Instance Name	Offset
str_out	0x48

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

Table 349: AES Stream Output Port Register (str_out)

Bits	Field	Type/ HW Rst	Description
31:0	str_out	R 0x0	Output Message Word

A.5.2.20 AES Output Vector 0 Register (ov0)

Instance Name	Offset
ov0	0x4C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	ov0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 350: AES Output Vector 0 Register (ov0)

Bits	Field	Type/ HW Rst	Description
31:0	ov0	R 0x0	Byte 0-3 of Output Vector

A.5.2.21 AES Output Vector 1 Register (ov1)

Instance Name	Offset
ov1	0x50

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	ov1																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 351: AES Output Vector 1 Register (ov1)

Bits	Field	Type/ HW Rst	Description
31:0	ov1	R 0x0	Byte 4-7 of Output Vector

A.5.2.22 AES Output Vector 2 Register (ov2)

Instance Name	Offset
ov2	0x54

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	ov2																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 352: AES Output Vector 2 Register (ov2)

Bits	Field	Type/ HW Rst	Description
31:0	ov2	R 0x0	Byte 8-11 of Output Vector

A.5.2.23 AES Output Vector 3 Register (ov3)

Instance Name	Offset
ov3	0x58

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	ov3																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 353: AES Output Vector 3 Register (ov3)

Bits	Field	Type/ HW Rst	Description
31:0	ov3	R 0x0	Byte 12-15 of Output Vector

A.5.2.24 AES Interrupt Status Register (isr)

Instance Name	Offset
isr	0x5C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																													status[2]	status[1]	status[0]
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0

Table 354: AES Interrupt Status Register (isr)

Bits	Field	Type/ HW Rst	Description
31:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2	status[2]	R 0x0	Status of AES Output FIFO Empty Interrupt 0x0 = AES output FIFO empty interrupt not occurred 0x1 = AES output FIFO empty interrupt occurred
1	status[1]	R 0x0	Status of AES Input FIFO Full Interrupt 0x0 = AES input FIFO full interrupt not occurred 0x1 = AES input FIFO full interrupt occurred
0	status[0]	R 0x0	Status of AES Output FIFO Empty Interrupt 0x0 = AES operation done interrupt not occurred 0x1 = AES operation done interrupt occurred

A.5.2.25 AES Interrupt Mask Register (imr)

Instance Name	Offset
imr	0x60

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																													mask[2]	mask[1]	mask[0]
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	1	1

Table 355: AES Interrupt Mask Register (imr)

Bits	Field	Type/ HW Rst	Description
31:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2	mask[2]	R/W 0x1	Mask of AES Output FIFO Empty Interrupt 0x0 = enable AES output FIFO empty interrupt 0x1 = disable AES output FIFO empty interrupt
1	mask[1]	R/W 0x1	Mask of AES Input FIFO Full Interrupt 0x0 = enable AES input FIFO full interrupt 0x1 = disable AES input FIFO full interrupt
0	mask[0]	R/W 0x1	mask[0]: Mask of AES Operation Done Interrupt 0x0 = enable AES operation done interrupt 0x1 = disable AES operation done interrupt

A.5.2.26 AES Interrupt Raw Status Register (irsr)

Instance Name	Offset
irsr	0x64

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																													status_raw[2]	status_raw[1]	status_raw[0]
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0

Table 356: AES Interrupt Raw Status Register (irsr)

Bits	Field	Type/ HW Rst	Description
31:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2	status_raw[2]	R 0x0	AES Output FIFO Empty Interrupt Raw Status Regardless of Mask 0x0 = AES output FIFO empty interrupt not occurred 0x1 = AES output FIFO empty interrupt
1	status_raw[1]	R 0x0	AES Input FIFO Full Interrupt Raw Status Regardless of Mask 0x0 = AES no input FIFO full interrupt not occurred 0x1 = AES no input FIFO full interrupt occurred
0	status_raw[0]	R 0x0	AES Operation Done Interrupt Raw Status Regardless of Mask 0x0 = AES operation done interrupt not occurred 0x1 = AES operation done interrupt occurred

A.5.2.27 AES Interrupt Clear Register (icr)

Instance Name	Offset
icr	0x68

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																													clear[2]	clear[1]	clear[0]
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0

Table 357: AES Interrupt Clear Register (icr)

Bits	Field	Type/ HW Rst	Description
31:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2	clear[2]	R/W 0x0	Clearance of AES Output FIFO Empty Interrupt Status and Raw Status
1	clear[1]	R/W 0x0	Clearance of AES Input FIFO Full Interrupt Status and Raw Status
0	clear[0]	R/W 0x0	Clearance of AES Operation Done Interrupt Status and Raw Status

A.5.2.28 AES Revision Register (rev_id)

Instance Name	Offset
rev_id	0x8C

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

Table 358: AES Revision Register (rev_id)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7:4	major_rev_id	R 0x1	Major Revision ID
3:0	minor_rev_id	R 0x2	Minor Revision ID

THIS PAGE INTENTIONALLY LEFT BLANK

A.6 CRC Address Block

A.6.1 CRC Register Map

Table 359: CRC Register Map

Offset	Name	HW Rst	Description	Details
0x00	isr	0x0000_0000	Interrupt Status Register	Page: 537
0x04	irsr	0x0000_0000	Interrupt Raw Status Register	Page: 538
0x08	icr	0x0000_0000	Interrupt Clear Register	Page: 538
0x0C	imr	0x0000_0001	Interrupt Mask Register	Page: 539
0x10	ctrl	0x0000_0000	CRC Module Control Register	Page: 540
0x14	stream_len_m1	0x0000_0000	Stream Length Minus 1 Register	Page: 541
0x18	stream_in	0x0000_0000	Stream Input Register	Page: 541
0x1C	result	0x0000_0000	CRC Calculation Result Register	Page: 542
0x3C	rev_id	0x0000_0012	CRC Revision ID Register	Page: 542

A.6.2 CRC Registers

A.6.2.1 Interrupt Status Register (isr)

Instance Name	Offset
isr	0x00

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

Table 360: Interrupt Status Register (isr)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	status	R 0x0	CRC Calculation Interrupt Status After Mask status[0]: CRC done 0x0 = interrupt is not occurred 0x1 = interrupt is occurred

A.6.2.2 Interrupt Raw Status Register (irsr)

Instance Name	Offset
irsr	0x04

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																															status_raw[0]
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0

Table 361: Interrupt Raw Status Register (irsr)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	status_raw[0]	R 0x0	Raw Status of IRQ Regardless of Mask

A.6.2.3 Interrupt Clear Register (icr)

Instance Name	Offset
icr	0x08

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

Table 362: Interrupt Clear Register (icr)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	clear	W 0x0	Clearance of status[0] and status_raw[0]

A.6.2.4 Interrupt Mask Register (imr)

Instance Name	Offset
imr	0x0C

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

Table 363: Interrupt Mask Register (imr)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	mask	R/W 0x1	Mask of Interrupt 0x0 = enable generation of IRQ and corresponding status[0] 0x1 = disable generation of IRQ and corresponding status[0]

A.6.2.5 CRC Module Control Register (ctrl)

Instance Name	Offset
ctrl	0x10

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

Table 364: CRC Module Control Register (ctrl)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3:1	mode	R/W 0x0	CRC Mode Select 0x0 = x**16+x**12+x**5+1 (CRC-16-CCITT, CRC-CCITT) 0x1 = x**16+x**15+x**2+1 (CRC-16, CRC-16-IBM, CRC-16-ANSI) 0x2 = x**16+x**15+x**11+x**9+x**8+x**7+x**5+x**4+x**2+x+1 (CRC-16-T10-DIF) 0x3 = x**32+x**26+x**23+x**22+x**16+x**12+x**11+x**10+x**8+ x**7+x**5+x**4+x**2+x+1 (CRC-32-IEEE802.3) 0x4 = x**16+x**13+x**12+x**11+x**10+x**8+x**6+x**5+x**2+1(CRC-16-DNP) others = reserved
0	enable	R/W 0x0	CRC Calculate Enable 0x0 = disable CRC calculation 0x1 = enable CRC calculation (automatically cleared when CRC calculation is finished)

A.6.2.6 Stream Length Minus 1 Register (stream_len_m1)

Instance Name	Offset
stream_len_m1	0x14

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	length_m1																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 365: Stream Length Minus 1 Register (stream_len_m1)

Bits	Field	Type/ HW Rst	Description
31:0	length_m1	R/W 0x0	Input Stream Length Minus 1 (units of bytes) Example: 0x0000_0000 = input stream length of 1 byte ... 0x0000_00FF = input stream length of 256 bytes

A.6.2.7 Stream Input Register (stream_in)

Instance Name	Offset
stream_in	0x18

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

Table 366: Stream Input Register (stream_in)

Bits	Field	Type/ HW Rst	Description
31:0	data	R/W 0x0	Stream Input Data

A.6.2.8 CRC Calculation Result (result)

Instance Name	Offset
result	0x1C

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

Table 367: CRC Calculation Result (result)

Bits	Field	Type/ HW Rst	Description
31:0	data	R 0x0	CRC Calculation Result

A.6.2.9 CRC Revision ID Register (rev_id)

Instance Name	Offset
rev_id	0x3C

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

Table 368: CRC Revision ID Register (rev_id)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7:4	major_rev_id	R 0x1	Major Revision ID
3:0	minor_rev_id	R 0x2	Minor Revision ID

A.7 I²C Address Block

A.7.1 I²C Register Map

Table 369: I²C Register Map

Offset	Name	HW Rst	Description	Details
0x00	IC_CON	0x0000_007F	I2C Control Register	Page: 545
0x04	IC_TAR	0x0000_1055	I2C Target Address Register	Page: 548
0x08	IC_SAR	0x0000_0055	I2C Slave Address Register	Page: 549
0x0C	IC_HS_MADDR	0x0000_0001	I2C High Speed Master Mode Code Address Register	Page: 550
0x10	IC_DATA_CMD	0x0000_0000	I2C Rx/Tx Data Buffer and Command Register	Page: 551
0x14	IC_SS_SCL_HCNT	0x0000_01F4	Standard Speed I2C Clock SCL High Count Register	Page: 552
0x18	IC_SS_SCL_LCNT	0x0000_024C	Standard Speed I2C Clock SCL Low Count Register	Page: 553
0x1C	IC_FS_SCL_HCNT	0x0000_004B	Fast Speed I2C Clock SCL High Count Register	Page: 554
0x20	IC_FS_SCL_LCNT	0x0000_00A3	Fast Speed I2C Clock SCL Low Count Register	Page: 555
0x24	IC_HS_SCL_HCNT	0x0000_0008	High Speed I2C Clock SCL High Count Register	Page: 556
0x28	IC_HS_SCL_LCNT	0x0000_0014	High Speed I2C Clock SCL Low Count Register	Page: 557
0x2C	IC_INTR_STAT	0x0000_0000	I2C Interrupt Status Register	Page: 558
0x30	IC_INTR_MASK	0x0000_08FF	I2C Interrupt Mask Register	Page: 560
0x34	IC_RAW_INTR_STAT	0x0000_0000	I2C Raw Interrupt Status Register	Page: 562
0x38	IC_RX_TL	0x0000_0000	I2C Receive FIFO Threshold Register	Page: 565
0x3C	IC_TX_TL	0x0000_0000	I2C Transmit FIFO Threshold Register	Page: 566
0x40	IC_CLR_INTR	0x0000_0000	Clear Combined and Individual Interrupt Register	Page: 567
0x44	IC_CLR_RX_UNDER	0x0000_0000	Clear RX_UNDER Interrupt Register	Page: 567
0x48	IC_CLR_RX_OVER	0x0000_0000	Clear RX_OVER Interrupt Register	Page: 568
0x4C	IC_CLR_TX_OVER	0x0000_0000	Clear TX_OVER Interrupt Register	Page: 568
0x50	IC_CLR_RD_REQ	0x0000_0000	Clear RD_REQ Interrupt Register	Page: 569
0x54	IC_CLR_TX_ABRT	0x0000_0000	Clear TX_ABRT Interrupt Register	Page: 569
0x58	IC_CLR_RX_DONE	0x0000_0000	Clear RX_DONE Interrupt Register	Page: 570
0x5C	IC_CLR_ACTIVITY	0x0000_0000	Clear ACTIVITY Interrupt Register	Page: 570
0x60	IC_CLR_STOP_DET	0x0000_0000	Clear STOP_DET Interrupt Register	Page: 571

Table 369: I²C Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x64	IC_CLR_START_DET	0x0000_0000	Clear START_DET Interrupt Register	Page: 571
0x68	IC_CLR_GEN_CALL	0x0000_0000	Clear GEN_CALL Interrupt Register	Page: 572
0x6C	IC_ENABLE	0x0000_0000	I2C Enable Register	Page: 572
0x70	IC_STATUS	0x0000_0006	I2C Status Register	Page: 573
0x74	IC_TXFLR	0x0000_0000	I2C Transmit FIFO Level Register	Page: 575
0x78	IC_RXFLR	0x0000_0000	I2C Receive FIFO Level Register	Page: 576
0x7C	IC_SDA_HOLD	0x0000_0001	I2C SDA Hold Register	Page: 577
0x80	IC_TX_ABRT_SOURCE	0x0000_0000	I2C Transmit Abort Source Register	Page: 578
0x84	IC_SLV_DATA_NACK_ ONLY	0x0000_0000	Generate Slave Data NACK Register	Page: 580
0x88	IC_DMA_CR	0x0000_0000	DMA Control Register	Page: 581
0x8C	IC_DMA_TDLR	0x0000_0000	DMA Transmit Data Level Register	Page: 582
0x90	IC_DMA_RDLR	0x0000_0000	I2C Receive Data Level Register	Page: 582
0x94	IC_SDA_SETUP	0x0000_0064	I2C SDA Setup Register	Page: 583
0x98	IC_ACK_GENERAL_ CALL	0x0000_0001	I2C ACK General Call Register	Page: 584
0x9C	IC_ENABLE_STATUS	0x0000_0000	I2C Enable Status Register	Page: 585
0xA0	IC_FS_SPKLEN	0x0000_0006	I2C SS and FS Spike Suppression Limit Register	Page: 587
0xA4	IC_HS_SPKLEN	0x0000_0002	I2C HS Spike Suppression Limit Register	Page: 588
0xF4	IC_COMP_PARAM_1	0x000F_0FEE	Component Parameter Register 1	Page: 589
0xF8	IC_COMP_VERSION	0x3131_352A	I2C Component Version Register	Page: 591
0xFC	IC_COMP_TYPE	0x4457_0140	I2C Component Type Register	Page: 591

A.7.2 I²C Registers

A.7.2.1 IC_CON Register

If configuration parameter I2C_DYNAMIC_TAR_UPDATE = 0, all bits are Read/Write. If I2C_ DYNAMIC_TAR_UPDATE = 1, bit 4 is Read-only. This register can be written only when the I²C is disabled, which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect.

Instance Name	Offset
IC_CON	0x00

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																									ic_slave_disable	ic_restart_en	ic_10bitaddr_master_rd_only	ic_10bitaddr_slave	speed		master_mode
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	1	1	1	1	1	1

Table 370: I2C Control Register (IC_CON)

Bits	Field	Type/ HW Rst	Description
31:7	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
6	ic_slave_disable	R/W 0x1	This bit controls whether I2C has its slave disabled, which means once the presetn signal is applied, then this bit takes on the value of the configuration parameter IC_SLAVE_DISABLE. You have the choice of having the slave enabled or disabled after reset is applied, which means software does not have to configure the slave. By default, the slave is always enabled (in reset state as well). If you need to disable it after reset, set this bit to 1. If this bit is set (slave is disabled), I²C functions only as a master and does not perform any action that requires a slave. • Reset value: IC_SLAVE_DISABLE configuration parameter • Software should ensure that if this bit is written with 0, then bit 0 should also be written with a 0. 0x0 = slave is enabled 0x1 = slave is disabled

Table 370: I2C Control Register (IC_CON) (Continued)

Bits	Field	Type/ HW Rst	Description
5	ic_restart_en	R/W 0x1	Determines whether RESTART conditions may be sent when acting as a master. Some older slaves do not support handling RESTART conditions; however, RESTART conditions are used in several I^2C operations. When RESTART is disabled, the master is prohibited from performing the following functions: • Change direction within a transfer (split) • Send a START BYTE • High-speed mode operation • Combined format transfers in 7-bit addressing modes • Read operation with a 10-bit address • Send multiple bytes per transfer By replacing RESTART condition followed by a STOP and a subsequent START condition, split operations are broken down into multiple I^2C transfers. If the above operations are performed, it will result in setting bit 6 (TX_ABRT) of the IC_RAW_INTR_STAT register. • Reset value: IC_RESTART_EN configuration parameter 0x0 = disable 0x1 = enable
4	ic_10bitaddr_master_rd_only	R 0x1	If the I2C_DYNAMIC_TAR_UPDATE configuration parameter is set to 'No' (0), this bit is named IC_10BITADDR_MASTER and controls whether the I^2C starts its transfers in 7- or 10-bit addressing mode when acting as a master. If I2C_DYNAMIC_TAR_UPDATE is set to 'Yes' (1), the function of this bit is handled by bit 12 of IC_TAR register, and becomes a read-only copy called IC_10BITADDR_MASTER_rd_only. • Dependencies: If I2C_DYNAMIC_TAR_UPDATE = 1, then this bit is read-only. If I2C_DYNAMIC_TAR_UPDATE = 0, then this bit can be read or write. • Reset value: IC_10BITADDR_MASTER configuration parameter 0x0 = 7-bit addressing 0x1 = 10-bit addressing
3	ic_10bitaddr_slave	R/W 0x1	When acting as a slave, this bit controls whether the I^2C responds to 7- or 10-bit addresses. • Reset value: IC_10BITADDR_SLAVE configuration parameter 0x0 = 7-bit addressing (I^2C ignores transactions that involve 10-bit addressing; for 7-bit addressing, only the lower 7 bits of the IC_SAR register are compared) 0x1 = 10-bit addressing (I^2C responds to only 10-bit addressing transfers that match the full 10 bits of the IC_SAR register)

Table 370: I2C Control Register (IC_CON) (Continued)

Bits	Field	Type/ HW Rst	Description
2:1	speed	R/W 0x3	These bits control at which speed the I^2C operates; its setting is relevant only if one is operating the I^2C in master mode. Hardware protects against illegal values being programmed by software. This register should be programmed only with a value in the range of 1 to IC_MAX_SPEED_MODE; otherwise, hardware updates this register with the value of IC_MAX_SPEED_MODE. • Reset value: IC_MAX_SPEED_MODE configuration 0x1 = standard mode (100 Kbps) 0x2 = fast mode (400 Kbps) 0x3 = high speed mode (3.4 Mbps)
0	master_mode	R/W 0x1	This bit controls whether the I^2C master is enabled. • Reset value: IC_MASTER_MODE configuration parameter • Software should ensure that if this bit is written with 1 then bit 6 should also be written with a 1. 0x0 = master disabled 0x1 = master enabled

A.7.2.2 IC_TAR Register

12 bits or 13 bits; 13 bits only when I2C_DYNAMIC_TAR_UPDATE = 1

If the configuration parameter I2C_DYNAMIC_TAR_UPDATE is set to 'No' (0), this register is 12 bits wide, and bits 15:12 are reserved. This register can be written to only when IC_ENABLE is set to 0. However, if I2C_DYNAMIC_TAR_UPDATE = 1, then the register becomes 13 bits wide. All bits can be dynamically updated as long as any set of the following conditions are true:

- I^2C is NOT enabled (IC_ENABLE is set to 0); or
- I^2C is enabled (IC_ENABLE=1); AND I^2C is NOT engaged in any Master (tx, rx) operation (IC_STATUS[5]=0); AND I^2C is enabled to operate in Master mode (IC_CON[0]=1); AND there are NO entries in the TX FIFO (IC_STATUS[2]=1)

Instance Name	Offset
IC_TAR	0x04

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																			ic_10bitaddr_master	special	gc_or_start	ic_tar									
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	0	0	0	0	0	1	0	1	0	1	0	1

Table 371: I2C Target Address Register (IC_TAR)

Bits	Field	Type/ HW Rst	Description
31:13	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
12	ic_10bitaddr_master	R/W 0x1	This bit controls whether the I^2C starts its transfers in 7- or 10-bit addressing mode when acting as a master. • Dependencies: This bit exists in this register only if the I2C_DYNAMIC_TAR_UPDATE configuration parameter is set to 'Yes' (1). • Reset value: IC_10BITADDR_MASTER configuration parameter 0x0 = 7-bit addressing 0x1 = 10-bit addressing
11	special	R/W 0x0	This bit indicates whether software performs a General Call or START BYTE command. 0x0 = ignore bit 10 GC_OR_START and use IC_TAR normally 0x1 = perform special I2C command as specified in GC_OR_START bit

Table 371: I2C Target Address Register (IC_TAR) (Continued)

Bits	Field	Type/ HW Rst	Description
10	gc_or_start	R/W 0x0	If bit 11 (SPECIAL) is set to 1, then this bit indicates whether a General Call or START byte command is to be performed by the I^2C. 0x0 = general call address after issuing a general call, only writes may be performed. attempting to issue a read command results in setting bit 6 (TX_ABRT) of the IC_RAW_INTR_STAT register. the I^2C remains in general call mode until the SPECIAL bit value (bit 11) is cleared. 0x1 = START BYTE
9:0	ic_tar	R/W 0x55	This is the target address for any master transaction. When transmitting a General Call, these bits are ignored. To generate a START BYTE, the CPU needs to write only once into these bits. • Reset value: IC_DEFAULT_TAR_SLAVE_ADDR configuration parameter If the IC_TAR and IC_SAR are the same, loopback exists but the FIFOs are shared between master and slave, so full loopback is not feasible. Only one direction loopback mode is supported (simplex), not duplex. A master cannot transmit to itself; it can transmit to only a slave.

A.7.2.3 IC_SAR Register

Instance Name	Offset
IC_SAR	0x08

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

Table 372: I2C Slave Address Register (IC_SAR)

Bits	Field	Type/ HW Rst	Description
31:10	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
9:0	ic_sar	R/W 0x55	The IC_SAR holds the slave address when the I2C is operating as a slave. For 7-bit addressing, only IC_SAR[6:0] is used. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. Note • The default values cannot be any of the reserved address locations: that is, 0x00 to 0x07, or 0x78 to 0x7f. The correct operation of the device is not guaranteed if you program the IC_SAR or IC_TAR to a reserved value. • Reset value: IC_DEFAULT_SLAVE_ADDR configuration parameter

A.7.2.4 IC_HS_MADDR Register

Instance Name	Offset
IC_HS_MADDR	0x0C

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

Table 373: I2C High Speed Master Mode Code Address Register (IC_HS_MADDR)

Bits	Field	Type/ HW Rst	Description
31:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2:0	ic_hs_mar	R/W 0x1	This bit field holds the value of the I2C HS mode master code. HS-mode master codes are reserved 8-bit codes (00001xxx) that are not used for slave addressing or other purposes. Each master has its unique master code; up to eight high-speed mode masters can be present on the same I2C bus system. Valid values are from 0 to 7. This register goes away and becomes read-only returning 0s if the IC_MAX_SPEED_MODE configuration parameter is set to either Standard (1) or Fast (2). This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. • Reset value: IC_HS_MASTER_CODE configuration parameter

A.7.2.5 IC_DATA_CMD Register

- This is the register the CPU writes to when filling the TX FIFO and the CPU reads from when retrieving bytes from RX FIFO.
- Size: 9 bits (writes) 8 bits (reads)
- With 9 bits required for writes, the I^2C requires 16-bit data on the APB bus transfers when writing into the transmit FIFO. 8-bit transfers remain for reads from the receive FIFO.

Instance Name	Offset
IC_DATA_CMD	0x10

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

Table 374: I2C Rx/Tx Data Buffer and Command Register (IC_DATA_CMD)

Bits	Field	Type/ HW Rst	Description
31:9	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
8	cmd	R/W 0x0	This bit controls whether a read or a write is performed. This bit does not control the direction when the I^2C acts as a slave. It controls only the direction when it acts as a master. When a command is entered in the TX FIFO, this bit distinguishes the write and read commands. In slave-receiver mode, this bit is a 'don't care' because writes to this register are not required. In slave-transmitter mode, a '0' indicates that CPU data is to be transmitted and as DAT or IC_DATA_CMD[7:0]. When programming this bit, you should remember the following: attempting to perform a read operation after a General Call command has been sent results in a TX_ABRT interrupt (bit 6 of the IC_RAW_INTR_STAT register), unless bit 11 (SPECIAL) in the IC_TAR register has been cleared. If a 1 is written to this bit after receiving a RD_REQ interrupt, then a TX_ABRT interrupt occurs. • It is possible that while attempting a master I2C read transfer on I^2C, a RD_REQ interrupt may have occurred simultaneously due to a remote I2C master addressing I^2C. In this type of scenario, I^2C ignores the IC_DATA_CMD write, generates a TX_ABRT interrupt, and waits to service the RD_REQ interrupt. 0x0 = write 0x1 = read
7:0	dat	R/W 0x0	This register contains the data to be transmitted or received on the I2C bus. If you are writing to this register and want to perform a read, bits 7:0 (DAT) are ignored by the I^2C. However, when this register is read, these bits return the value of data received on the I^2C interface.

A.7.2.6 IC_SS_SCL_HCNT Register

Instance Name	Offset
IC_SS_SCL_HCNT	0x14

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

Table 375: Name: Standard Speed I2C Clock SCL High Count Register (IC_SS_SCL_HCNT)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	ic_ss_scl_hcnt	R/W 0x1F4	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock high-period count for standard speed. The table below shows some sample IC_SS_SCL_HCNT calculations. These values apply only if the ic_clk is set to the given frequency in the table. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set. For designs with APB_DATA_WIDTH = 8, the order of programming is important to ensure the correct operation of the I^2C. The lower byte must be programmed first. Then the upper byte is programmed. When the configuration parameter IC_HC_COUNT_VALUES is set to 1, this register is read only. • This register must not be programmed to a value higher than 65525, because the I^2C uses a 16-bit counter to flag an I2C bus idle condition when this counter reaches a value of IC_SS_SCL_HCNT + 10. • Reset value: IC_SS_SCL_HIGH_COUNT configuration parameter

A.7.2.7 IC_SS_SCL_LCNT Register

Instance Name	Offset
IC_SS_SCL_LCNT	0x18

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

Table 376: Name: Standard Speed I2C Clock SCL Low Count Register (IC_SS_SCL_LCNT)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	ic_ss_scl_lcnt	R/W 0x24C	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock low period count for standard speed. The table below shows some sample IC_SS_SCL_LCNT calculations. These values apply only if the ic_clk is set to the given frequency in the table. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. The minimum valid value is 8; hardware prevents values less than this being written, and if attempted, results in 8 being set. For designs with APB_DATA_WIDTH = 8, the order of programming is important to ensure the correct operation of I^2C. The lower byte must be programmed first, and then the upper byte is programmed. When the configuration parameter IC_HC_COUNT_VALUES is set to 1, this register is read only. • Reset value: IC_SS_SCL_LOW_COUNT configuration parameter

A.7.2.8 IC_FS_SCL_HCNT Register

Instance Name	Offset
IC_FS_SCL_HCNT	0x1C

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

Table 377: Name: Fast Speed I2C Clock SCL High Count Register (IC_FS_SCL_HCNT)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	ic_fs_scl_hcnt	R/W 0x4B	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock high-period count for fast speed. It is used in high-speed mode to send the Master Code and START BYTE or General CALL. The table below shows some sample IC_FS_SCL_HCNT calculations. These values apply only if the ic_clk is set to the given frequency in the table. This register goes away and becomes read-only returning 0s if IC_MAX_SPEED_MODE = standard. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set. For designs with APB_DATA_WIDTH == 8 the order of programming is important to ensure the correct operation of the I^2C. The lower byte must be programmed first. Then the upper byte is programmed. When the configuration parameter IC_HC_COUNT_VALUES is set to 1, this register is read only. • Reset value: IC_FS_SCL_HIGH_COUNT configuration parameter

A.7.2.9 IC_FS_SCL_LCNT Register

Instance Name	Offset
IC_FS_SCL_LCNT	0x20

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

Table 378: Fast Speed I2C Clock SCL Low Count Register (IC_FS_SCL_LCNT)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	ic_fs_scl_lcnt	R/W 0xA3	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock low period count for fast speed. It is used in high-speed mode to send the Master Code and START BYTE or General CALL. The table below shows some sample IC_FS_SCL_LCNT calculations. These values apply only if the ic_clk is set to the given frequency in the table. This register goes away and becomes read-only returning 0s if IC_MAX_SPEED_MODE = standard. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. The minimum valid value is 8; hardware prevents values less than this being written, and if attempted results in 8 being set. For designs with APB_DATA_WIDTH = 8 the order of programming is important to ensure the correct operation of the I^2C. The lower byte must be programmed first. Then the upper byte is programmed. If the value is less than 8 then the count value gets changed to 8. When the configuration parameter IC_HC_COUNT_VALUES is set to 1, this register is read only. • Reset value: IC_FS_SCL_LOW_COUNT configuration parameter

A.7.2.10 IC_HS_SCL_HCNT Register

Instance Name	Offset
IC_HS_SCL_HCNT	0x24

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

Table 379: High Speed I2C Clock SCL High Count Register (IC_HS_SCL_HCNT)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	ic_hs_scl_hcnt	R/W 0x8	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock high period count for high speed. The table below shows some sample IC_HS_SCL_HCNT calculations. These values apply only if the ic_clk is set to the given frequency in the table. The SCL High time depends on the loading of the bus. For 100pF loading, the SCL High time is 60ns; for 400pF loading, the SCL High time is 120ns. This register goes away and becomes read-only returning 0s if IC_MAX_SPEED_MODE != high. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set. For designs with APB_DATA_WIDTH = 8 the order of programming is important to ensure the correct operation of the I^2C. The lower byte must be programmed first. Then the upper byte is programmed. When the configuration parameter IC_HC_COUNT_VALUES is set to 1, this register is read only. • Reset value: IC_HS_SCL_HIGH_COUNT configuration parameter

A.7.2.11 IC_HS_SCL_LCNT Register

Instance Name	Offset
IC_HS_SCL_LCNT	0x28

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

Table 380: High Speed I2C Clock SCL Low Count Register (IC_HS_SCL_LCNT)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	ic_hs_scl_lcnt	R/W 0x14	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock low period count for high speed. The table below shows some sample IC_HS_SCL_LCNT calculations. These values apply only if the ic_clk is set to the given frequency in the table. The SCL low time depends on the loading of the bus. For 100pF loading, the SCL low time is 160ns; for 400pF loading, the SCL low time is 320ns. This register goes away and becomes read-only returning 0s if IC_MAX_SPEED_MODE != high. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. The minimum valid value is 8; hardware prevents values less than this being written, and if attempted results in 8 being set. For designs with APB_DATA_WIDTH == 8 the order of programming is important to ensure the correct operation of the I^2C. The lower byte must be programmed first. Then the upper byte is programmed. If the value is less than 8 then the count value gets changed to 8. When the configuration parameter IC_HC_COUNT_VALUES is set to 1, this register is read only. • Reset value: IC_HS_SCL_LOW_COUNT configuration parameter

A.7.2.12 IC_INTR_STAT Register

Each bit in this register has a corresponding mask bit in the IC_INTR_MASK register. These bits are cleared by reading the matching interrupt clear register. The unmasked raw versions of these bits are available in the IC_RAW_INTR_STAT register.

Instance Name	Offset
IC_INTR_STAT	0x2C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																				r_gen_call	r_start_det	r_stop_det	r_activity	r_rx_done	r_tx_abrt	r_rd_req	r_tx_empty	r_tx_over	r_rx_full	r_rx_over	r_rx_under
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0

Table 381: I2C Interrupt Status Register (IC_INTR_STAT)

Bits	Field	Type/ HW Rst	Description
31:12	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
11	r_gen_call	R 0x0	Set only when a General Call address is received and it is acknowledged. It stays set until it is cleared either by disabling the I^2C or when the CPU reads bit 0 of the IC_CLR_GEN_CALL register. The I^2C stores the received data in the Rx buffer.
10	r_start_det	R 0x0	Indicates whether a START or RESTART condition has occurred on the I2C interface regardless of whether I^2C is operating in slave or master mode.
9	r_stop_det	R 0x0	Indicates whether a STOP condition has occurred on the I2C interface regardless of whether the I^2C is operating in slave or master mode.
8	r_activity	R 0x0	This bit captures I^2C activity and stays set until it is cleared. There are 4 ways to clear it: • Disabling the I^2C • Reading the IC_CLR_ACTIVITY register • Reading the IC_CLR_INTR register • System reset Once this bit is set, it stays set unless one of the 4 methods is used to clear it. Even if the I^2C module is idle, this bit remains set until cleared, indicating that there was activity on the bus.
7	r_rx_done	R 0x0	When the I^2C is acting as a slave-transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done.

Table 381: I2C Interrupt Status Register (IC_INTR_STAT) (Continued)

Bits	Field	Type/ HW Rst	Description
6	r_tx_abrt	R 0x0	This bit indicates if the I^2C, as an I^2C transmitter, is unable to complete the intended actions on the contents of the transmit FIFO. This situation can occur both as an I^2C master or an I^2C slave, and is referred to as a 'transmit abort'. When this bit is set to 1, the IC_TX_ABRT_SOURCE register indicates the reason why the transmit abort takes places. • The I^2C flushes/resets/empties the TX FIFO whenever this bit is set. The TX FIFO remains in this flushed state until the register IC_CLR_TX_ABRT is read. Once this read is performed, the TX FIFO is then ready to accept more data bytes from the APB interface.
5	r_rd_req	R 0x0	This bit is set to 1 when the I^2C is acting as a slave and another I2C master is attempting to read data from I^2C. The I^2C holds the I2C bus in a wait state (SCL=0) until this interrupt is serviced, which means that the slave has been addressed by a remote master that is asking for data to be transferred. The processor must respond to this interrupt and then write the requested data to the IC_DATA_CMD register. This bit is set to 0 just after the processor reads the IC_CLR_RD_REQ register.
4	r_tx_empty	R 0x0	This bit is set to 1 when the transmit buffer is at or below the threshold value set in the IC_TX_TL register. It is automatically cleared by hardware when the buffer level goes above the threshold. When the IC_ENABLE bit 0 is 0, the TX FIFO is flushed and held in reset. There the TX FIFO looks like it has no data within it, so this bit is set to 1, provided there is activity in the master or slave state machines. When there is no longer activity, then with ic_en=0, this bit is set to 0.
3	r_tx_over	R 0x0	Set during transmit if the transmit buffer is filled to IC_TX_BUFFER_DEPTH and the processor attempts to issue another I2C command by writing to the IC_DATA_CMD register. When the module is disabled, this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.
2	r_rx_full	R 0x0	Set when the receive buffer reaches or goes above the RX_TL threshold in the IC_RX_TL register. It is automatically cleared by hardware when buffer level goes below the threshold. If the module is disabled (IC_ENABLE[0]=0), the RX FIFO is flushed and held in reset; therefore the RX FIFO is not full. So this bit is cleared once the IC_ENABLE bit 0 is programmed with a 0, regardless of the activity that continues.
1	r_rx_over	R 0x0	Set if the receive buffer is completely filled to IC_RX_BUFFER_DEPTH and an additional byte is received from an external I2C device. The I^2C acknowledges this, but any data bytes received after the FIFO is full are lost. If the module is disabled (IC_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.

Table 381: I2C Interrupt Status Register (IC_INTR_STAT) (Continued)

Bits	Field	Type/ HW Rst	Description
0	r_rx_under	R 0x0	Set if the processor attempts to read the receive buffer when it is empty by reading from the IC_DATA_CMD register. If the module is disabled (IC_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.

A.7.2.13 IC_INTR_MASK Register

These bits mask their corresponding interrupt status bits. They are active high; a value of 0 prevents a bit from generating an interrupt.

Instance Name	Offset
IC_INTR_MASK	0x30

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																				m_gen_call	m_start_det	m_stop_det	m_activity	m_rx_done	m_tx_abrt	m_rd_req	m_tx_empty	m_tx_over	m_rx_full	m_rx_over	m_rx_under
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	0	0	0	1	1	1	1	1	1	1	1

Table 382: I2C Interrupt Mask Register (IC_INTR_MASK)

Bits	Field	Type/ HW Rst	Description
31:12	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
11	m_gen_call	R/W 0x1	Set only when a General Call address is received and it is acknowledged. It stays set until it is cleared either by disabling I^2C or when the CPU reads bit 0 of the IC_CLR_GEN_CALL register. I^2C stores the received data in the Rx buffer.
10	m_start_det	R/W 0x0	Indicates whether a START or RESTART condition has occurred on the I2C interface regardless of whether I^2C is operating in slave or master mode.
9	m_stop_det	R/W 0x0	Indicates whether a STOP condition has occurred on the I2C interface regardless of whether I^2C is operating in slave or master mode.

Table 382: I2C Interrupt Mask Register (IC_INTR_MASK) (Continued)

Bits	Field	Type/ HW Rst	Description
8	m_activity	R/W 0x0	This bit captures I^2C activity and stays set until it is cleared. There are 4 ways to clear it: • Disabling the I^2C • Reading the IC_CLR_ACTIVITY register • Reading the IC_CLR_INTR register • System reset Once this bit is set, it stays set unless one of the 4 methods is used to clear it. Even if the I^2C module is idle, this bit remains set until cleared, indicating that there was activity on the bus.
7	m_rx_done	R/W 0x1	When the I^2C is acting as a slave-transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done.
6	m_tx_abrt	R/W 0x1	This bit indicates if the I^2C, as an I2C transmitter, is unable to complete the intended actions on the contents of the transmit FIFO. This situation can occur both as an I2C master or an I2C slave, and is referred to as a 'transmit abort'. When this bit is set to 1, the IC_TX_ABRT_SOURCE register indicates the reason why the transmit abort takes places. • The I^2C flushes/resets/empties the TX FIFO whenever this bit is set. The TX FIFO remains in this flushed state until the register IC_CLR_TX_ABRT is read. Once this read is performed, the TX FIFO is then ready to accept more data bytes from the APB interface.
5	m_rd_req	R/W 0x1	This bit is set to 1 when I^2C is acting as a slave and another I2C master is attempting to read data from I^2C. The I^2C holds the I2C bus in a wait state (SCL=0) until this interrupt is serviced, which means that the slave has been addressed by a remote master that is asking for data to be transferred. The processor must respond to this interrupt and then write the requested data to the IC_DATA_CMD register. This bit is set to 0 just after the processor reads the IC_CLR_RD_REQ register.
4	m_tx_empty	R/W 0x1	This bit is set to 1 when the transmit buffer is at or below the threshold value set in the IC_TX_TL register. It is automatically cleared by hardware when the buffer level goes above the threshold. When the IC_ENABLE bit 0 is 0, the TX FIFO is flushed and held in reset. There the TX FIFO looks like it has no data within it, so this bit is set to 1, provided there is activity in the master or slave state machines. When there is no longer activity, then with ic_en=0, this bit is set to 0.
3	m_tx_over	R/W 0x1	Set during transmit if the transmit buffer is filled to IC_TX_BUFFER_DEPTH and the processor attempts to issue another I2C command by writing to the IC_DATA_CMD register. When the module is disabled, this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.

Table 382: I2C Interrupt Mask Register (IC_INTR_MASK) (Continued)

Bits	Field	Type/ HW Rst	Description
2	m_rx_full	R/W 0x1	Set when the receive buffer reaches or goes above the RX_TL threshold in the IC_RX_TL register. It is automatically cleared by hardware when buffer level goes below the threshold. If the module is disabled (IC_ENABLE[0]=0), the RX FIFO is flushed and held in reset; therefore the RX FIFO is not full. So this bit is cleared once the IC_ENABLE bit 0 is programmed with a 0, regardless of the activity that continues.
1	m_rx_over	R/W 0x1	Set if the receive buffer is completely filled to IC_RX_BUFFER_ DEPTH and an additional byte is received from an external I2C device. The I^2C acknowledges this, but any data bytes received after the FIFO is full are lost. If the module is disabled (IC_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.
0	m_rx_under	R/W 0x1	Set if the processor attempts to read the receive buffer when it is empty by reading from the IC_DATA_CMD register. If the module is disabled (IC_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.

A.7.2.14 IC_RAW_INTR_STAT Register

Unlike the IC_INTR_STAT register, these bits are not masked so they always show the true status of the I^2C.

Instance Name	Offset
IC_RAW_INTR_STAT	0x34

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																				gen_call	start_det	stop_det	activity	rx_done	tx_abrt	rd_req	tx_empty	tx_over	rx_full	rx_over	rx_under
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0

Table 383: I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT)

Bits	Field	Type/ HW Rst	Description
31:12	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
11	gen_call	R 0x0	Set only when a General Call address is received and it is acknowledged. It stays set until it is cleared either by disabling I^2C or when the CPU reads bit 0 of the IC_CLR_GEN_CALL register. I^2C stores the received data in the Rx buffer.

Table 383: I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT) (Continued)

Bits	Field	Type/ HW Rst	Description
10	start_det	R 0x0	Indicates whether a START or RESTART condition has occurred on the I2C interface regardless of whether I^2C is operating in slave or master mode.
9	stop_det	R 0x0	Indicates whether a STOP condition has occurred on the I2C interface regardless of whether I^2C is operating in slave or master mode.
8	activity	R 0x0	This bit captures I^2C activity and stays set until it is cleared. There are 4 ways to clear it: • Disabling the I^2C • Reading the IC_CLR_ACTIVITY register • Reading the IC_CLR_INTR register • System reset Once this bit is set, it stays set unless one of the 4 methods is used to clear it. Even if the I^2C module is idle, this bit remains set until cleared, indicating that there was activity on the bus.
7	rx_done	R 0x0	When the I^2C is acting as a slave-transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done.
6	tx_abrt	R 0x0	This bit indicates if I^2C, as an I2C transmitter, is unable to complete the intended actions on the contents of the transmit FIFO. This situation can occur both as an I2C master or an I2C slave, and is referred to as a 'transmit abort'. When this bit is set to 1, the IC_TX_ABRT_SOURCE register indicates the reason why the transmit abort takes places. • The I^2C flushes/resets/empties the TX FIFO whenever this bit is set. The TX FIFO remains in this flushed state until the register IC_CLR_TX_ABRT is read. Once this read is performed, the TX FIFO is then ready to accept more data bytes from the APB interface.
5	rd_req	R 0x0	This bit is set to 1 when I^2C is acting as a slave and another I2C master is attempting to read data from I^2C. The I^2C holds the I2C bus in a wait state (SCL=0) until this interrupt is serviced, which means that the slave has been addressed by a remote master that is asking for data to be transferred. The processor must respond to this interrupt and then write the requested data to the IC_DATA_CMD register. This bit is set to 0 just after the processor reads the IC_CLR_RD_REQ register.
4	tx_empty	R 0x0	This bit is set to 1 when the transmit buffer is at or below the threshold value set in the IC_TX_TL register. It is automatically cleared by hardware when the buffer level goes above the threshold. When the IC_ENABLE bit 0 is 0, the TX FIFO is flushed and held in reset. There the TX FIFO looks like it has no data within it, so this bit is set to 1, provided there is activity in the master or slave state machines. When there is no longer activity, then with ic_en=0, this bit is set to 0.

Table 383: I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT) (Continued)

Bits	Field	Type/ HW Rst	Description
3	tx_over	R 0x0	Set during transmit if the transmit buffer is filled to IC_TX_BUFFER_ DEPTH and the processor attempts to issue another I2C command by writing to the IC_DATA_CMD register. When the module is disabled, this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.
2	rx_full	R 0x0	Set when the receive buffer reaches or goes above the RX_TL threshold in the IC_RX_TL register. It is automatically cleared by hardware when buffer level goes below the threshold. If the module is disabled (IC_ENABLE[0]=0), the RX FIFO is flushed and held in reset; therefore the RX FIFO is not full. So this bit is cleared once the IC_ENABLE bit 0 is programmed with a 0, regardless of the activity that continues.
1	rx_over	R 0x0	Set if the receive buffer is completely filled to IC_RX_BUFFER_ DEPTH and an additional byte is received from an external I2C device. The I^2C acknowledges this, but any data bytes received after the FIFO is full are lost. If the module is disabled (IC_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.
0	rx_under	R 0x0	Set if the processor attempts to read the receive buffer when it is empty by reading from the IC_DATA_CMD register. If the module is disabled (IC_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.

A.7.2.15 IC_RX_TL Register

Instance Name	Offset
IC_RX_TL	0x38

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

Table 384: I2C Receive FIFO Threshold Register (IC_RX_TL)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7:0	rx_tl	R/W 0x0	Receive FIFO Threshold Level Controls the level of entries (or above) that triggers the RX_FULL interrupt (bit 2 in IC_RAW_INTR_STAT register). The valid range is 0-255, with the additional restriction that hardware does not allow this value to be set to a value larger than the depth of the buffer. If an attempt is made to do that, the actual value set will be the maximum depth of the buffer. A value of 0 sets the threshold for 1 entry, and a value of 255 sets the threshold for 256 entries. • Reset value: IC_RX_TL configuration parameter

A.7.2.16 IC_TX_TL Register

Instance Name	Offset
IC_TX_TL	0x3C

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

Table 385: I2C Transmit FIFO Threshold Register (IC_TX_TL)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7:0	tx_tl	R/W 0x0	Transmit FIFO Threshold Level Controls the level of entries (or below) that trigger the TX_EMPTY interrupt (bit 4 in IC_RAW_INTR_STAT register). The valid range is 0-255, with the additional restriction that it may not be set to value larger than the depth of the buffer. If an attempt is made to do that, the actual value set will be the maximum depth of the buffer. A value of 0 sets the threshold for 0 entries, and a value of 255 sets the threshold for 255 entries. • Reset value: IC_TX_TL configuration parameter

A.7.2.17 IC_CLR_INTR Register

Instance Name	Offset
IC_CLR_INTR	0x40

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

Table 386: Clear Combined and Individual Interrupt Register (IC_CLR_INTR)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	clr_intr	R 0x0	Read this register to clear the combined interrupt, all individual interrupts, and the IC_TX_ABRT_SOURCE register. This bit does not clear hardware clearable interrupts but software clearable interrupts. Refer to Bit 9 of the IC_TX_ABRT_SOURCE register for an exception to clearing IC_TX_ABRT_SOURCE.

A.7.2.18 IC_CLR_RX_UNDER Register

Instance Name	Offset
IC_CLR_RX_UNDER	0x44

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

Table 387: Clear RX_UNDER Interrupt Register (IC_CLR_RX_UNDER)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	clr_rx_under	R 0x0	Read this register to clear the RX_UNDER interrupt (bit 0) of the IC_RAW_INTR_STAT register.

A.7.2.19 IC_CLR_RX_OVER Register

Instance Name	Offset
IC_CLR_RX_OVER	0x48

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

Table 388: Clear RX_OVER Interrupt Register (IC_CLR_RX_OVER)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	clr_rx_over	R 0x0	Read this register to clear the RX_OVER interrupt (bit 1) of the IC_RAW_INTR_STAT register.

A.7.2.20 IC_CLR_TX_OVER Register

Instance Name	Offset
IC_CLR_TX_OVER	0x4C

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

Table 389: Clear TX_OVER Interrupt Register (IC_CLR_TX_OVER)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	clr_tx_over	R 0x0	Read this register to clear the TX_OVER interrupt (bit 3) of the IC_RAW_INTR_STAT register.

A.7.2.21 IC_CLR_RD_REQ Register

Instance Name	Offset
IC_CLR_RD_REQ	0x50

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

Table 390: Clear RD_REQ Interrupt Register (IC_CLR_RD_REQ)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	clr_rd_req	R 0x0	Read this register to clear the RD_REQ interrupt (bit 5) of the IC_RAW_INTR_STAT register.

A.7.2.22 IC_CLR_TX_ABRT Register

Instance Name	Offset
IC_CLR_TX_ABRT	0x54

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

Table 391: Clear TX_ABRT Interrupt Register (IC_CLR_TX_ABRT)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	clr_tx_abrt	R 0x0	Read this register to clear the TX_ABRT interrupt (bit 6) of the IC_RAW_INTR_STAT register, and the IC_TX_ABRT_SOURCE register. This also releases the TX FIFO from the flushed/reset state, allowing more writes to the TX FIFO. Refer to Bit 9 of the IC_TX_ABRT_SOURCE register for an exception to clearing IC_TX_ABRT_SOURCE.

A.7.2.23 IC_CLR_RX_DONE Register

Instance Name	Offset
IC_CLR_RX_DONE	0x58

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

Table 392: Clear RX_DONE Interrupt Register (IC_CLR_RX_DONE)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	clr_rx_done	R 0x0	Read this register to clear the RX_DONE interrupt (bit 7) of the IC_RAW_INTR_STAT register.

A.7.2.24 IC_CLR_ACTIVITY Register

Instance Name	Offset
IC_CLR_ACTIVITY	0x5C

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

Table 393: Clear ACTIVITY Interrupt Register (IC_CLR_ACTIVITY)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	clr_activity	R 0x0	Reading this register clears the ACTIVITY interrupt if the I2C is not active anymore. If the I2C module is still active on the bus, the ACTIVITY interrupt bit continues to be set. It is automatically cleared by hardware if the module is disabled and if there is no further activity on the bus. The value read from this register to get status of the ACTIVITY interrupt (bit 8) of the IC_RAW_INTR_STAT register.

A.7.2.25 IC_CLR_STOP_DET Register

Instance Name	Offset
IC_CLR_STOP_DET	0x60

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

Table 394: Clear STOP_DET Interrupt Register (IC_CLR_STOP_DET)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	clr_stop_det	R 0x0	Read this register to clear the STOP_DET interrupt (bit 9) of the IC_ RAW_INTR_STAT register.

A.7.2.26 IC_CLR_START_DET Register

Instance Name	Offset
IC_CLR_START_DET	0x64

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

Table 395: Clear START_DET Interrupt Register (IC_CLR_START_DET)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	clr_start_det	R 0x0	Read this register to clear the START_DET interrupt (bit 10) of the IC_RAW_INTR_STAT register.

A.7.2.27 IC_CLR_GEN_CALL Register

Instance Name	Offset
IC_CLR_GEN_CALL	0x68

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

Table 396: Clear GEN_CALL Interrupt Register (IC_CLR_GEN_CALL)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	clr_gen_call	R 0x0	Read this register to clear the GEN_CALL interrupt (bit 11) of IC_ RAW_INTR_STAT register.

A.7.2.28 IC_ENABLE Register

Instance Name	Offset
IC_ENABLE	0x6C

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

Table 397: I2C Enable Register (IC_ENABLE)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

Table 397: I2C Enable Register (IC_ENABLE) (Continued)

Bits	Field	Type/ HW Rst	Description
0	enable	R/W 0x0	Controls whether the I^2C is enabled. 0x0 = disables I^2C (TX and RX FIFOs are held in an erased state) 0x1 = enables I^2C software can disable I^2C while it is active. however, it is important that care be taken to ensure that I^2C is disabled properly. when I^2C is disabled, the following occurs: • TX FIFO and RX FIFO get flushed. • Status bits in the IC_INTR_STAT register are still active until I^2C goes into IDLE state. If the module is transmitting, it stops as well as deletes the contents of the transmit buffer after the current transfer is complete. If the module is receiving, the I^2C stops the current transfer at the end of the current byte and does not acknowledge the transfer. In systems with asynchronous pclk and ic_clk when IC_CLK_TYPE parameter set to asynchronous (1), there is a 2 ic_clk delay when enabling or disabling the I^2C.

A.7.2.29 IC_STATUS Register

This is a read-only register used to indicate the current transfer status and FIFO status. The status register may be read at any time. None of the bits in this register request an interrupt. When the I2C is disabled by writing 0 in bit 0 of the IC_ENABLE register:

- Bits 1 and 2 are set to 1
- Bits 3 and 4 are set to 0

When the master or slave state machines goes to idle and ic_en=0:

- Bits 5 and 6 are set to 0

Instance Name	Offset
IC_STATUS	0x70

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																									slv_activity	mst_activity	rff	rfne	tfe	tfnf	activity
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	1	1	0

Table 398: I2C Status Register (IC_STATUS)

Bits	Field	Type/ HW Rst	Description
31:7	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

Table 398: I2C Status Register (IC_STATUS) (Continued)

Bits	Field	Type/ HW Rst	Description
6	slv_activity	R 0x0	Slave FSM Activity Status When the Slave Finite State Machine (FSM) is not in the IDLE state, this bit is set. 0x0 = slave FSM is in IDLE state so the slave part of I^2C is not active 0x1 = slave FSM is not in IDLE state so the slave part of I^2C is active
5	mst_activity	R 0x0	Master FSM Activity Status When the Master Finite State Machine (FSM) is not in the IDLE state, this bit is set. **Note:** IC_STATUS[0]-that is, ACTIVITY bit-is the OR of SLV_ ACTIVITY and MST_ACTIVITY bits. 0x0 = master FSM is in IDLE state so the master part of I^2C is not active 0x1 = master FSM is not in IDLE state so the master part of I^2C is active note IC_STATUS[0]-that is, ACTIVITY bit-is the OR of SLV_ACTIVITY and MST_ACTIVITY bits
4	rff	R 0x0	Receive FIFO Completely Full When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared. 0x0 = receive FIFO is not full 0x1 = receive FIFO is full
3	rfne	R 0x0	Receive FIFO Not Empty This bit is set when the receive FIFO contains one or more entries; it is cleared when the receive FIFO is empty. 0x0 = receive FIFO is empty 0x1 = receive FIFO is not empty
2	tfe	R 0x1	Transmit FIFO Completely Empty When the transmit FIFO is completely empty, this bit is set. When it contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. 0x0 = transmit FIFO is not empty 0x1 = transmit FIFO is empty
1	tfnf	R 0x1	Transmit FIFO Not Full Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. 0x0 = transmit FIFO is full 0x1 = transmit FIFO is not full
0	activity	R 0x0	I2C Activity Status

A.7.2.30 IC_TXFLR Register

- Size: TX_ABW + 1

This register contains the number of valid data entries in the transmit FIFO buffer. It is cleared whenever:

- The I2C is disabled.
- There is a transmit abort that is, TX_ABRT bit is set in the IC_RAW_INTR_STAT register.
- The slave bulk transmit mode is aborted.

The register increments whenever data is placed into the transmit FIFO and decrements when data is taken from the transmit FIFO.

Instance Name	Offset
IC_TXFLR	0x74

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

Table 399: I2C Transmit FIFO Level Register (IC_TXFLR)

Bits	Field	Type/ HW Rst	Description
31:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4:0	txflr	R 0x0	Transmit FIFO Level. Contains the number of valid data entries in the transmit FIFO.

A.7.2.31 IC_RXFLR Register

- Size: RX_ABW + 1

This register contains the number of valid data entries in the receive FIFO buffer. It is cleared whenever:

- The I2C is disabled.
- Whenever there is a transmit abort caused by any of the events tracked in IC_TX_ABRT_SOURCE.

The register increments whenever data is placed into the receive FIFO and decrements when data is taken from the receive FIFO.

Instance Name	Offset
IC_RXFLR	0x78

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

Table 400: I2C Receive FIFO Level Register (IC_RXFLR)

Bits	Field	Type/ HW Rst	Description
31:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4:0	rxflr	R 0x0	Receive FIFO Level. Contains the number of valid data entries in the receive FIFO.

A.7.2.32 IC_SDA_HOLD Register

This register controls the amount of time delay (in terms of number of ic_clk clock periods) introduced in the falling edge of SCL, relative to SDA changing, when I^2C services a read request in a slave-transmitter operation. The relevant I2C requirement is thd:DAT as detailed in the I2C Bus Specification.

Instance Name	Offset
IC_SDA_HOLD	0x7C

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

Table 401: I2C SDA Hold Register (IC_SDA_HOLD)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	ic_sda_hold	R/W 0x1	SDA Hold • Default Reset value: 0x1, but can be hard-coded by setting the IC_DEFAULT_SDA_HOLD configuration parameter.

A.7.2.33 IC_TX_ABRT_SOURCE Register

This register has 16 bits that indicate the source of the TX_ABRT bit. Except for Bit 9, this register is cleared whenever the IC_CLR_TX_ABRT register or the IC_CLR_INTR register is read. To clear Bit 9, the source of the ABRT_SBYTE_NORSTRT must be fixed first; RESTART must be enabled (IC_CON[5]=1), the bit must be cleared (IC_TAR[11]), or the GC_OR_START bit must be cleared (IC_TAR[10]). Once the source of the ABRT_SBYTE_NORSTRT is fixed, then this bit can be cleared in the same manner as other bits in this register. If the source of the ABRT_SBYTE_NORSTRT is not fixed before attempting to clear this bit, Bit 9 clears for one cycle and is then re-asserted.

Instance Name	Offset	
IC_TX_ABRT_SOURCE	0x80	SPECIAL

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																abrt_slvrd_intx	abrt_slv_arblost	abrt_slvflush_txfifo	arb_lost	abrt_master_dis	abrt_10b_rd_norstrt	abrt_sbyte_norstrt	abrt_hs_norstrt	abrt_sbyte_ackdet	abrt_hs_ackdet	abrt_gcall_read	abrt_gcall_noack	abrt_txdata_noack	abrt_10addr2_noack	abrt_10addr1_noack	abrt_7b_addr_noack
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 402: I2C Transmit Abort Source Register (IC_TX_ABRT_SOURCE)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15	abrt_slvrd_intx	R 0x0	Slave-Transmitter 0x1 = when the processor side responds to a slave mode request for data to be transmitted to a remote master and user writes a 1 in CMD (bit 8) of IC_DATA_CMD register
14	abrt_slv_arblost	R 0x0	Slave-Transmitter • Even though the slave never 'owns' the bus, something could go wrong on the bus. This is a fail safe check. For instance, during a data transmission at the low-to-high transition of SCL, if what is on the data bus is not what is supposed to be transmitted, then I^2C no longer own the bus. 0x1 = slave lost the bus while transmitting data to a remote master. IC_TX_ABRT_SOURCE[12] is set at the same time.
13	abrt_slvflush_txfifo	R 0x0	Slave-Transmitter 0x1 = slave has received a read command and some data exists in the TX FIFO so the slave issues a TX_ABRT interrupt to flush old data in TX FIFO
12	arb_lost	R 0x0	Master-Transmitter or Slave-Transmitter • I2C can be both master and slave at the same time. 0x1 = master has lost arbitration, or if IC_TX_ABRT_SOURCE[14] is also set, then the slave transmitter has lost arbitration
11	abrt_master_dis	R 0x0	Master-Transmitter or Master-Receiver 0x1 = user tries to initiate a master operation with the master mode disabled

Table 402: I2C Transmit Abort Source Register (IC_TX_ABRT_SOURCE) (Continued)

Bits	Field	Type/ HW Rst	Description
10	abrt_10b_rd_norstrt	R 0x0	Master-Receiver 0x1 = the restart is disabled (IC_RESTART_EN bit (IC_CON[5]) =0) and the master sends a read command in 10-bit addressing mode
9	abrt_sbyte_norstrt	R 0x0	Master To clear Bit 9, the source of the ABRT_SBYTE_NORSTRT must be fixed first; restart must be enabled (IC_CON[5]=1), the SPECIAL bit must be cleared (IC_TAR[11]), or the GC_OR_START bit must be cleared (IC_TAR[10]). Once the source of the ABRT_SBYTE_NORSTRT is fixed, then this bit can be cleared in the same manner as other bits in this register. If the source of the ABRT_SBYTE_NORSTRT is not fixed before attempting to clear this bit, bit 9 clears for one cycle and then gets reasserted. 0x1 = the restart is disabled (IC_RESTART_EN bit (IC_CON[5]) =0) and the user is trying to send a START byte
8	abrt_hs_norstrt	R 0x0	Master-Transmitter or Master-Receiver 0x1 = the restart is disabled (IC_RESTART_EN bit (IC_CON[5]) =0) and the user is trying to use the master to transfer data in high speed mode
7	abrt_sbyte_ackdet	R 0x0	Master 0x1 = master has sent a START byte and the START byte was acknowledged (wrong behavior)
6	abrt_hs_ackdet	R 0x0	Master 0x1 = master is in high speed mode and the high speed master code was acknowledged (wrong behavior)
5	abrt_gcall_read	R 0x0	Master-Transmitter 0x1 = I^2C in master mode sent a general call but the user programmed the byte following the general call to be a read from the bus (IC_DATA_CMD[9] is set to 1)
4	abrt_gcall_noack	R 0x0	Master-Transmitter 0x1 = I^2C in master mode sent a general call and no slave on the bus acknowledged the general call
3	abrt_txdata_noack	R 0x0	Master-Transmitter 0x1 = this is a master-mode only bit. master has received an acknowledgement for the address, but when it sent data byte(s) following the address, it did not receive an acknowledge from the remote slave(s).
2	abrt_10addr2_noack	R 0x0	Master-Transmitter or Master-Receiver 0x1 = master is in 10-bit address mode and the second address byte of the 10-bit address was not acknowledged by any slave
1	abrt_10addr1_noack	R 0x0	Master-Transmitter or Master-Receiver 0x1 = master is in 10-bit address mode and the first 10-bit address byte was not acknowledged by any slave

Table 402: I2C Transmit Abort Source Register (IC_TX_ABRT_SOURCE) (Continued)

Bits	Field	Type/ HW Rst	Description
0	abrt_7b_addr_noack	R 0x0	Master-Transmitter or Master-Receiver 0x1 = master is in 7-bit addressing mode and the address sent was not acknowledged by any slave

A.7.2.34 IC_SLV_DATA_NACK_ONLY Register

The register is used to generate a NACK for the data part of a transfer when I^2C is acting as a slave-receiver. This register only exists when the IC_SLV_DATA_NACK_ONLY parameter is set to 1. When this parameter disabled, this register does not exist and writing to the register's address has no effect.

Instance Name	Offset
IC_SLV_DATA_NACK_ONLY	0x84

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

Table 403: Generate Slave Data NACK Register (IC_SLV_DATA_NACK_ONLY)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	nack	R/W 0x0	Generate NACK This NACK generation only occurs when I^2C is a slave-receiver. If this register is set to a value of 1, it can only generate a NACK after a data byte is received; hence, the data transfer is aborted and the data received is not pushed to the receive buffer. When the register is set to a value of 0, it generates NACK/ACK, depending on normal criteria. 0x0 = generate NACK/ACK normally 0x1 = generate NACK after data byte received

A.7.2.35 IC_DMA_CR Register

This register is only valid when I^2C is configured with a set of DMA Controller interface signals (IC_HAS_DMA = 1). When I^2C is not configured for DMA operation, this register does not exist and writing to the register address has no effect and reading from this register address will return zero. The register is used to enable the DMA Controller interface operation. There is a separate bit for transmit and receive. This can be programmed regardless of the state of IC_ENABLE.

Instance Name	Offset
IC_DMA_CR	0x88

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

Table 404: DMA Control Register (IC_DMA_CR)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	tdmae	R/W 0x0	Transmit DMA Enable This bit enables/disables the transmit FIFO DMA channel. 0x0 = transmit DMA disabled 0x1 = transmit DMA enabled
0	rdmae	R/W 0x0	Receive DMA Enable This bit enables/disables the receive FIFO DMA channel. 0x0 = receive DMA disabled 0x1 = receive DMA enabled

A.7.2.36 IC_DMA_TDLR Register

This register is only valid when the I2C is configured with a set of DMA interface signals (IC_HAS_DMA = 1). When I2C is not configured for DMA operation, this register does not exist; writing to its address has no effect; reading from its address returns zero.

Instance Name	Offset
IC_DMA_TDLR	0x8C

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

Table 405: DMA Transmit Data Level Register (IC_DMA_TDLR)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3:0	dmatdl	R/W 0x0	Transmit Data Level This bit field controls the level at which a DMA request is made by the transmit logic. It is equal to the watermark level; that is, the dma_tx_req signal is generated when the number of valid data entries in the transmit FIFO is equal to or below this field value, and TDMAE = 1.

A.7.2.37 IC_DMA_RDLR Register

This register is only valid when I2C is configured with a set of DMA interface signals (IC_HAS_DMA = 1). When I2C is not configured for DMA operation, this register does not exist; writing to its address has no effect; reading from its address returns zero.

Instance Name	Offset
IC_DMA_RDLR	0x90

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

Table 406: I2C Receive Data Level Register (IC_DMA_RDLR)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

Table 406: I2C Receive Data Level Register (IC_DMA_RDLR) (Continued)

Bits	Field	Type/ HW Rst	Description
3:0	dmardl	R/W 0x0	Receive Data Level This bit field controls the level at which a DMA request is made by the receive logic. The watermark level = DMARDL+1; that is, dma_rx_req is generated when the number of valid data entries in the receive FIFO is equal to or more than this field value + 1, and RDMAE =1. For instance, when DMARDL is 0, then dma_rx_req is asserted when 1 or more data entries are present in the receive FIFO.

A.7.2.38 IC_SDA_SETUP Register

This register controls the amount of time delay (in terms of number of ic_clk clock periods) introduced in the rising edge of SCL, relative to SDA changing, when I^2C services a read request in a slave-transmitter operation. The relevant I2C requirement is tSU:DAT (note 4) as detailed in the I2C Bus Specification.

Instance Name	Offset
IC_SDA_SETUP	0x94

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

Table 407: I2C SDA Setup Register (IC_SDA_SETUP)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7:0	sda_setup	R/W 0x64	SDA Setup. It is recommended that if the required delay is 1000ns, then for an ic_clk frequency of 10 MHz, IC_SDA_SETUP should be programmed to a value of 11. • Default Reset value: 0x64, but can be hard-coded by setting the IC_DEFAULT_SDA_SETUP configuration parameter.

A.7.2.39 IC_ACK_GENERAL_CALL Register

The register controls whether I^{2}C responds with a ACK or NACK when it receives an I2C General Call address.

Instance Name	Offset
IC_ACK_GENERAL_CALL	0x98

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

Table 408: I2C ACK General Call Register (IC_ACK_GENERAL_CALL)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	ack_gen_call	R/W 0x1	ACK General Call When set to 1, the I^{2}C responds with a ACK (by asserting ic_data_oe) when it receives a General Call. Otherwise, the I^{2}C responds with a NACK (by negating ic_data_oe). • Default Reset value: 0x1, but can be hard-coded by setting the IC_DEFAULT_ACK_GENERAL_CALL configuration parameter.

A.7.2.40 IC_ENABLE_STATUS Register

The register is used to report the I^2C hardware status when the IC_ENABLE register is set from 1 to 0; that is, when the I^2C is disabled. If IC_ENABLE has been set to 1, bits [2:1] are forced to 0, and bit 0 is forced to 1. If IC_ENABLE has been set to 0, bits [2:1] is only be valid as soon as bit 0 is read as 0.

When IC_ENABLE has been written with 0 a delay occurs for bit 0 to be read as '0' because disabling the I^2C depends on I2C bus activities.

Instance Name	Offset
IC_ENABLE_STATUS	0x9C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																													slv_rx_data_lost	slv_disabled_while_busy	ic_en
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0

Table 409: I2C Enable Status Register (IC_ENABLE_STATUS)

Bits	Field	Type/ HW Rst	Description
31:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2	slv_rx_data_lost	R 0x0	Slave Received Data Lost This bit indicates if a Slave-Receiver operation has been aborted with at least one data byte received from an I2C transfer due to the setting of IC_ENABLE from 1 to 0. When read as 1, the I^2C is deemed to have been actively engaged in an aborted I2C transfer (with matching address) and the data phase of the I2C transfer has been entered, even though a data byte has been responded with a NACK. • If the remote I2C master terminates the transfer with a STOP condition before the I^2C has a chance to NACK a transfer, and IC_ENABLE has been set to 0, then this bit is also set to 1. When read as 0, the I^2C is deemed to have been disabled without being actively involved in the data phase of a Slave-Receiver transfer. • The CPU can safely read this bit when IC_EN (bit 0) is read as 0.

Table 409: I2C Enable Status Register (IC_ENABLE_STATUS) (Continued)

Bits	Field	Type/ HW Rst	Description
1	slv_disabled_while_busy	R 0x0	Slave Disabled While Busy (Transmit, Receive) This bit indicates if a potential or active Slave operation has been aborted due to the setting of the IC_ENABLE register from 1 to 0. This bit is set when the CPU writes a 0 to the IC_ENABLE register while: (a) I^2C is receiving the address byte of the Slave-Transmitter operation from a remote master; OR, (b) address and data bytes of the Slave-Receiver operation from a remote master. When read as 1, I^2C is deemed to have forced a NACK during any part of an I2C transfer, irrespective of whether the I2C address matches the slave address set in I^2C (IC_SAR register) OR if the transfer is completed before IC_ENABLE is set to 0 but has not taken effect. • If the remote I2C master terminates the transfer with a STOP condition before the I^2C has a chance to NACK a transfer, and IC_ENABLE has been set to 0, then this bit will also be set to 1. When read as 0, I^2C is deemed to have been disabled when there is master activity, or when the I2C bus is idle. • The CPU can safely read this bit when IC_EN (bit 0) is read as 0.
0	ic_en	R 0x0	ic_en Status This bit always reflects the value driven on the output port ic_en. When read as 1, I^2C is deemed to be in an enabled state. When read as 0, I^2C is deemed completely inactive. • The CPU can safely read this bit anytime. When this bit is read as 0, the CPU can safely read SLV_RX_DATA_LOST (bit 2) and SLV_DISABLED_WHILE_BUSY (bit 1).

A.7.2.41 IC_FS_SPKLEN Register

This register is used to store the duration, measured in ic_clk cycles, of the longest spike that is filtered out by the spike suppression logic w hen the component is operating in SS or FS modes. The relevant I2C requirement is tSP (table 4) as detailed in the I2C Bus Specification. This register must be programmed with a minimum value of 2.

Instance Name	Offset
IC_FS_SPKLEN	0xA0

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

Table 410: I2C SS and FS Spike Suppression Limit Register (IC_FS_SPKLEN)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7:0	ic_fs_spklen	R/W 0x6	This register must be set before any I2C bus transaction can take place to ensure stable operation. This register sets the duration, measured in ic_clk cycles, of the longest spike in the SCL or SDA lines that will be filtered out by the spike suppression logic. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. The minimum valid value is 2; hardware prevents values less than this being written, and if attempted results in 2 being set. • Default Reset value: IC_DEFAULT_FS_SPKLEN configuration parameter.

A.7.2.42 IC_HS_SPKLEN Register

This register is used to store the duration, measured in ic_clk cycles, of the longest spike that is filtered out by the spike suppression logic w hen the component is operating in HS modes. The relevant I2C requirement is tSP (table 6) as detailed in the I2C Bus Specification. This register must be programmed with a minimum value of 2.

Instance Name	Offset
IC_HS_SPKLEN	0xA4

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

Table 411: I2C HS Spike Suppression Limit Register (IC_HS_SPKLEN)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7:0	ic_hs_spklen	R/W 0x2	This register must be set before any I2C bus transaction can take place to ensure stable operation. This register sets the duration, measured in ic_clk cycles, of the longest spike in the SCL or SDA lines that will be filtered out by the spike suppression logic. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. The minimum valid value is 2; hardware prevents values less than this being written, and if attempted results in 2 being set. • Default Reset value: IC_DEFAULT_HS_SPKLEN configuration parameter.

A.7.2.43 IC_COMP_PARAM_1 Register

This is a constant read-only register that contains encoded information about the component's parameter settings. The reset value depends on coreConsultant parameter(s).

Instance Name	Offset
IC_COMP_PARAM_1	0xF4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved								tx_buffer_depth								rx_buffer_depth								add_encoded_params	has_dma	intr_io	hc_count_values	max_speed_mode		apb_data_width	
HW Rst	?	?	?	?	?	?	?	?	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1	0	1	1	1	0

Table 412: Component Parameter Register 1 (IC_COMP_PARAM_1)

Bits	Field	Type/ HW Rst	Description
31:24	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
23:16	tx_buffer_depth	R 0xF	The value of this register is derived from the IC_TX_BUFFER_ DEPTH coreConsultant parameter. 0x0 = reserved 0x1 = 2 0x2 = 3 to 0xFF = 256
15:8	rx_buffer_depth	R 0xF	The value of this register is derived from the IC_RX_BUFFER_ DEPTH coreConsultant parameter. 0x0 = reserved 0x1 = 2 0x2 = 3 to 0xFF = 256
7	add_encoded_params	R 0x1	The value of this register is derived from the IC_ADD_ENCODED_ PARAMS coreConsultant parameter. Reading 1 in this bit means that the capability of reading these encoded parameters with software has been included. Otherwise, the entire register is 0 regardless of the setting of any other parameters that are encoded in the bits. 0x0 = false 0x1 = true
6	has_dma	R 0x1	The value of this register is derived from the IC_HAS_DMA coreConsultant parameter 0x0 = false 0x1 = true

Table 412: Component Parameter Register 1 (IC_COMP_PARAM_1) (Continued)

Bits	Field	Type/ HW Rst	Description
5	intr_io	R 0x1	The value of this register is derived from the IC_INTR_IO coreConsultant parameter 0x0 = individual 0x1 = combined
4	hc_count_values	R 0x0	The value of this register is derived from the IC_HC_COUNT VALUES coreConsultant parameter 0x0 = false 0x1 = true
3:2	max_speed_mode	R 0x3	The value of this register is derived from the IC_MAX_SPEED_ MODE coreConsultant parameter. 0x0 = reserved 0x1 = standard 0x2 = fast 0x3 = high
1:0	apb_data_width	R 0x2	The value of this register is derived from the APB_DATA_WIDTH coreConsultant parameter. 0x0 = 8 bits 0x1 = 16 bits 0x2 = 32 bits 0x3 = reserved

A.7.2.44 IC_COMP_VERSION Register

Instance Name	Offset
IC_COMP_VERSION	0xF8

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

Table 413: I2C Component Version Register (IC_COMP_VERSION)

Bits	Field	Type/ HW Rst	Description
31:0	ic_comp_version	R 0x3131_ 352A	Specific values for this register are described in the Releases Table in the I^2C Release Notes.

A.7.2.45 IC_COMP_TYPE Register

Instance Name	Offset
IC_COMP_TYPE	0xFC

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

Table 414: I2C Component Type Register (IC_COMP_TYPE)

Bits	Field	Type/ HW Rst	Description
31:0	ic_comp_type	R 0x4457_ 0140	Designware Component Type number = 0x44_57_01_40. This assigned unique hex value is constant and is derived from the 2 ASCII letters 'DW' followed by a 16-bit unsigned number.

THIS PAGE INTENTIONALLY LEFT BLANK

A.8 QSPI Address Block

A.8.1 QSPI Register Map

Table 415: QSPI Register Map

Offset	Name	HW Rst	Description	Details
0x00	Cntl	0x0000_0052	Serial Interface Control Register	Page: 594
0x04	Conf	0x0000_0002	Serial Interface Configuration Register	Page: 595
0x08	Dout	0x0000_0000	Serial Interface Data Out Register	Page: 598
0x0C	Din	0x0000_0000	Serial Interface Data Input Register	Page: 599
0x10	Instr	0x0000_0000	Serial Interface Instruction Register	Page: 600
0x14	Addr	0x0000_0000	Serial Interface Address Register	Page: 601
0x18	RdMode	0x0000_0000	Serial Interface Read Mode Register	Page: 602
0x1C	HdrCnt	0x0000_0000	Serial Interface Header Count Register	Page: 603
0x20	DInCnt	0x0000_0000	Serial Interface Data Input Count Register	Page: 604
0x24	Timing	0x0000_0111	Serial Interface Timing Register	Page: 605
0x28	Conf2	0x0000_1100	Serial Interface Configuration 2 Register	Page: 606
0x2C	ISR	0x0000_0000	Serial Interface Interrupt Status Register	Page: 607
0x30	IMR	0x0000_0FFF	Serial Interface Interrupt Mask Register	Page: 609
0x34	IRSR	0x0000_005B	Serial Interface Interrupt Raw Status Register	Page: 610
0x38	ISC	0x0000_0000	Serial Interface Interrupt Clear Register	Page: 612

A.8.2 QSPI Registers

A.8.2.1 Serial Interface Control Register (Cntl)

Instance Name	Offset
Cntl	0x00

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																				wfifo_overflw	wfifo_undrflw	rfifo_ovrflw	rfifo_undrflw	wfifo_full	wfifo_empty	rfifo_full	rfifo_empty	Reserved		xfer_rdy	ss_en
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	1	0	1	?	?	1	0

Table 416: Serial Interface Control Register (Cntl)

Bits	Field	Type/ HW Rst	Description
31:12	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
11	wfifo_overflw	R 0x0	Write FIFO Overflow 0x0 = write FIFO is not overflowed 0x1 = write FIFO is overflowed
10	wfifo_undrflw	R 0x0	Write FIFO Underflow 0x0 = write FIFO is not underflowed 0x1 = write FIFO is underflowed
9	rfifo_ovrflw	R 0x0	Read FIFO Overflow 0x0 = read FIFO is not overflowed 0x1 = read FIFO is overflowed
8	rfifo_undrflw	R 0x0	Read FIFO Underflow 0x0 = read FIFO is not underflowed 0x1 = read FIFO is underflowed
7	wfifo_full	R 0x0	Write FIFO Full 0x0 = write FIFO is not fulled 0x1 = write FIFO is fulled
6	wfifo_empty	R 0x1	Write FIFO Empty 0x0 = write FIFO is not emptied 0x1 = write FIFO is emptied
5	rfifo_full	R 0x0	Read FIFO Full 0x0 = read FIFO is not fulled 0x1 = read FIFO is fulled
4	rfifo_empty	R 0x1	Read FIFO Empty 0x0 = read FIFO is not emptied 0x1 = read FIFO is emptied

Table 416: Serial Interface Control Register (Cntl) (Continued)

Bits	Field	Type/ HW Rst	Description
3:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	xfer_rdy	R 0x1	Serial Interface Transfer Ready 0x0 = serial interface is currently transferring data 0x1 = serial interface is ready for a new transfer
0	ss_en	R/W 0x0	Serial Select Enable 0x0 = serial select is de-activated, ss_n (serial interface select) output is driven high 0x1 = serial select is activated, ss_n (serial interface select) output is driven low

A.8.2.2 Serial Interface Configuration Register (Conf)

Instance Name	Offset
Conf	0x04

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																xfer_start	xfer_stop	rw_en	addr_pin	data_pin		fifo_flush	clk_pol	clk_pha	Reserved	byte_len	clk_prescale					
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	?	0	0	0	0	1	0

Table 417: Serial Interface Configuration Register (Conf)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15	xfer_start	R/W 0x0	Transfer Start This bit starts the serial interface I/O transfer. For read transfers, RW_EN (R04[13]) = 0, the hardware resets this bit to 0 when the number of bytes indicated in DInCnt (R20h) register have been read in from the interface. For write transfers, RW_EN (R04[13]) = 1, firmware sets XFER_ STOP (R04h [14]) = 1 when all data have been written to the WFIFO and WFIFO_EMPTY (R00h [6]) = 1. Hardware resets this bit to 0 when all data have been written out the the interface. 0x0 = transfer has completed 0x1 = transfer has started

Table 417: Serial Interface Configuration Register (Conf) (Continued)

Bits	Field	Type/ HW Rst	Description
14	xfer_stop	R/W 0x0	Transfer Stop This bit stops the serial interface I/O transfer. The transfer stops at either a 1-byte or 4-byte boundary, depending on the setting of BYTE_LEN (R04h [5]). Once the byte boundary is reached, the hardware resets XFER_START (R04h [15]) to 0. Hardware resets this bit to 0 after XFER_START has been reset. 0x0 = continue current transfer 0x1 = stop current transfer
13	rw_en	R/W 0x0	Read Write Enable 0x0 = read data from the serial interface 0x1 = write data to the serial interface
12	addr_pin	R/W 0x0	Address Transfer Pin Number of pins used for transferring the content of the Addr (R14h) register. 0x0 = use one serial interface pin 0x1 = use the number of pins as indicated in DATA_PIN (R04h [11:10])
11:10	data_pin	R/W 0x0	Data Transfer Pin Number of pins used for transferring the non-command and nonaddress portions of each serial interface I/O transfer. 0x0 = use 1 serial interface pin (use in single mode) 0x1 = use 2 serial interface pins (use in dual mode) 0x2 = use 4 serial interface pins (use in quad mode) 0x3 = reserved
9	fifo_flush	R/W 0x0	Flush Read and Write FIFOs This bit flushes the Read and Write FIFOs. The FIFOs are emptied after being flushed. Hardware resets this bit to 0 after flushing. 0x0 = read and write FIFOs are not flushed 0x1 = read and write FIFOs are flushed
8	clk_pol	R/W 0x0	Serial Interface Clock Polarity Selects the serial interface clock as high or low when inactive. 0x0 = serial interface clock is low when inactive 0x1 = serial interface clock is high when inactive
7	clk_pha	R/W 0x0	Serial Interface Clock Phase Selects the serial interface clock phase. 0x0 = data is latched at the rising edge of the serial interface clock when CLK_POL (R04h [8]) = 0, and at the falling edge of the serial interface clock when CLK_POL = 1 0x1 = data is latched at the falling edge of the serial interface clock when CLK_POL = 0, and at the rising edge of the serial interface clock when CLK_POL = 1
6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

Table 417: Serial Interface Configuration Register (Conf) (Continued)

Bits	Field	Type/ HW Rst	Description
5	byte_len	R/W 0x0	Byte Length The number of bytes in each serial interface I/O transfer. 0x0 = 1 byte 0x1 = 4 bytes
4:0	clk_prescale	R/W 0x2	Serial Interface Clock Prescaler (from SPI clock) 0x00 = SPI clock/1 0x01 = SPI clock/1 0x02 = SPI clock/2 0x03 = SPI clock/3 0x04 = SPI clock/4 0x05 = SPI clock/5 ... 0x0D = SPI clock/13 0x0E = SPI clock/14 0x0F = SPI clock/15 0x10 = SPI clock/2 0x11 = SPI clock/2 0x12 = SPI clock/4 0x13 = SPI clock/6 0x14 = SPI clock/8 0x15 = SPI clock/10 ... 0x1D = SPI clock/26 0x1E = SPI clock/28 0x1F = SPI clock/30

A.8.2.3 Serial Interface Data Out Register (Dout)

Instance Name	Offset
Dout	0x08

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

Table 418: Serial Interface Data Out Register (Dout)

Bits	Field	Type/ HW Rst	Description
31:0	data_out	R/W 0x0	Serial Interface Data Out Writing to this register stores the data in a 8X32 bit Write FIFO. After the contents of the Instruction register (R10h), the Address register (R14h), the Read Mode register (R18h) and Dummy value are transferred out to the the serial interface, the data in the Write FIFO is shifted out. The serial interface clock stops when a Write FIFO empty condition occurs, WFIFO_EMPTY (R00h [6]) = 1. The clock restarts when Write FIFO is not empty, WFIFO_EMPTY (R00h [6]) = 0. When BYTE_LEN (R04h [5]) = 0, bits [7:0] are shifted out with bit 7 shifted out first and bit 0 shifted out last. When BYTE_LEN (R04h [5]) = 1, bits [7:0] are shifted out (bit 7 shifted out first and bit 0 shifted out last), followed by bits [15:8] (bit 15 shifted out first and bit 8 shifted out last), then bits [23:16] (bit 23 shifted out first and bit 16 shifted out last) and finally bits [31:24] (bit 31 shifted out first and bit 24 shifted out last). To avoid a Write FIFO overflow condition (WFIFO_OVRFLW (R00h [11]) = 1), check WFIFO_FULL (R00h [7]) is equal to 0 before writing to this register.

A.8.2.4 Serial Interface Data Input Register (Din)

Instance Name	Offset
Din	0x0C

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

Table 419: Serial Interface Data Input Register (Din)

Bits	Field	Type/ HW Rst	Description
31:0	data_in	R 0x0	Serial Interface Data In For read transfers, RW_EN (R04h [13]) = 0, data from the serial interface input pins are shifted in and stores in a 8X32 bit Read FIFO. The contents of the Read FIFO is read from this register. The serial interface clock stops when a Read FIFO full condition occurs, RFIFO_FULL (R00h [5]) = 1. The clock restarts when Read FIFO is not full, RFIFO_FULL (R00h [5]) = 0. When BYTE_LEN (R04h [5]) = 0, data is shifted into bits [7:0]. When BYTE_LEN (R04h [5]) = 1, data is shifted into bits [7:0] first, followed by bits [15:8], then bits [23:16] and finally bits [31:24]. To avoid a Read FIFO underflow condition, RFIFO_UNDRFLW (R00h [8]) = 1, check RFIFO_EMPTY (R00h [4]) is equal to 0 before reading this register.

A.8.2.5 Serial Interface Instruction Register (Instr)

Instance Name	Offset
Instr	0x10

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

Table 420: Serial Interface Instruction Register (Instr)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	instr	R/W 0x0	Instruction After XFER_START (R04h [15]) is set to 1, the content of this register is shifted out to the serial interface. When INSTR_CNT (R1Ch [1:0]) = 0, the content of this register is not shifted out to the serial interface. When INSTR_CNT (R1Ch [1:0]) = 1, bits [7:0] are shifted out. When INSTR_CNT (R1Ch [1:0]) = 2, bits [15:8] are shifted out first, followed by bits [7:0].

A.8.2.6 Serial Interface Address Register (Addr)

Instance Name	Offset
Addr	0x14

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

Table 421: Serial Interface Address Register (Addr)

Bits	Field	Type/ HW Rst	Description
31:0	addr	R/W 0x0	Serial Interface Address After Instr (R10h) is shifted out, the content of this register is shifted out to the serial interface. When ADDR_CNT (R1Ch [6:4]) = 0, the content of this register is not shifted out to the serial interface. When ADDR_CNT (R1Ch [6:4]) = 1, bits [7:0] are shifted out. When ADDR_CNT (R1Ch [6:4]) = 2, bits [15:8] are shifted out first, followed by bits [7:0]. When ADDR_CNT (R1Ch [6:4]) = 3, bits [23:16] are shifted out first, followed by bits [15:8], then bits [7:0]. When ADDR_CNT (R1Ch [6:4]) = 4, bits [31:24] are shifted out first, followed by bits [23:16], then bits [15:8] and finally bits [7:0].

A.8.2.7 Serial Interface Read Mode Register (RdMode)

Instance Name	Offset
RdMode	0x18

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

Table 422: Serial Interface Read Mode Register (RdMode)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	rmode	R/W 0x0	Serial Interface Read Mode After Addr (R14h) is shifted out, the content of this register is shifted out to the serial interface. When RM_CNT (R1Ch [9:8]) = 0, the content of this register is not shifted out to the serial interface. When RM_CNT (R1Ch [9:8]) = 1, bits [7:0] are shifted out. When RM_CNT (R1Ch [9:8]) = 2, bits [15:8] are shifted out first, followed by bits [7:0].

A.8.2.8 Serial Interface Header Count Register (HdrCnt)

Instance Name	Offset
HdrCnt	0x1C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																		dummy_cnt		Reserved		rm_cnt		Reserved		addr_cnt			Reserved		instr_cnt	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	?	?	0	0	?	0	0	0	?	?	0	0

Table 423: Serial Interface Header Count Register (HdrCnt)

Bits	Field	Type/ HW Rst	Description
31:14	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
13:12	dummy_cnt	R/W 0x0	Dummy Count Number of bytes to shift out to the serial interface after the content of RdMode (R18h) register is shifted out. The value being shifted out is 0. 0x0 = 0 byte 0x1 = 1 byte 0x2 = 2 bytes 0x3 = 3 bytes
11:10	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
9:8	rm_cnt	R/W 0x0	Read Mode Count Number of bytes in RdMode (R18h) register to shift out to the serial interface. 0x0 = 0 byte 0x1 = 1 byte 0x2 = 2 bytes 0x3 = reserved
7	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
6:4	addr_cnt	R/W 0x0	Address Count Number of bytes in Addr (R14h) register to shift out to the serial interface. 0x0 = 0 byte 0x1 = 1 byte 0x2 = 2 bytes 0x3 = 3 bytes 0x4 = 4 bytes others = reserved
3:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

Table 423: Serial Interface Header Count Register (HdrCnt) (Continued)

Bits	Field	Type/ HW Rst	Description
1:0	instr_cnt	R/W 0x0	Instruction Count Number of bytes in Instr (R10h) register to shift out to the serial interface. 0x0 = 0 byte 0x1 = 1 byte 0x2 = 2 bytes 0x3 = reserved

A.8.2.9 Serial Interface Data Input Count Register (DInCnt)

Instance Name	Offset
DInCnt	0x20

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

Table 424: Serial Interface Data Input Count Register (DInCnt)

Bits	Field	Type/ HW Rst	Description
31:20	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
19:0	data_in_cnt	R/W 0x0	Serial Interface Data In Count For read transfers, RW_EN (R04h [13]) = 0, this register indicates the number of bytes of data to shift in from the serial interface and stores in the 8X32 bit Read FIFO. When this register is set to 0, data is shifted in continuously until XFER_STOP (R04h [14]) bit is set to 1 by firmware.

A.8.2.10 Serial Interface Timing Register (Timing)

Instance Name	Offset
Timing	0x24

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

Table 425: Serial Interface Timing Register (Timing)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
6	clk_capt_edge	R/W 0x0	Serial Interface Capture Clock Edge Capture serial interface input data on either the rising or falling edge of the serial interface clock. This bit is used to allow more time to capture the input data. 0x0 = • When CLK_POL (R04 [8]) = 0 and CLK_PHA (R04 [7]) = 0, capture input data on rising edge of the serial interface clock • When CLK_POL (R04 [8]) = 0 and CLK_PHA (R04 [7]) = 1, capture input data on falling edge of the serial interface clock • When CLK_POL (R04 [8]) = 1 and CLK_PHA (R04 [7]) = 0, capture input data on falling edge of the serial interface clock • When CLK_POL (R04 [8]) = 1 and CLK_PHA (R04 [7]) = 1, capture input data on rising edge of the serial interface clock 0x1 = • When CLK_POL (R04 [8]) = 0 and CLK_PHA (R04 [7]) = 0, capture input data on falling edge of the serial interface clock • When CLK_POL (R04 [8]) = 0 and CLK_PHA (R04 [7]) = 1, capture input data on rising edge of the serial interface clock When CLK_PHA (R04 [7]) = 1, this bit must be set to 0.
5:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.8.2.11 Serial Interface Configuration 2 Register (Conf2)

Instance Name	Offset
Conf2	0x28

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																		dma_wr_burst		Reserved		dma_rd_burst		Reserved					dma_wr_en	dma_rd_en	srst
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	1	?	?	0	1	?	?	?	?	?	0	0	0

Table 426: Serial Interface Configuration 2 Register (Conf2)

Bits	Field	Type/ HW Rst	Description
31:14	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
13:12	dma_wr_burst	R/W 0x1	DMA Write Burst Number of data, each of width BYTE_LEN (R04h [5]), which can be written to the Write FIFO, without overflowing the FIFO, before a transmit burst request is made to the DMA controller. 0x0 = 1 data 0x1 = 4 data 0x2 = 8 data 0x3 = reserved
11:10	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
9:8	dma_rd_burst	R/W 0x1	DMA Read Burst Number of data, each of width BYTE_LEN (R04[5]), which is available in the Read FIFO before a receive burst request is made to the DMA controller. 0x0 = 1 data 0x1 = 4 data 0x2 = 8 data 0x3 = reserved
7:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2	dma_wr_en	R/W 0x0	DMA Write Enable 0x0 = DMA write is disabled 0x1 = DMA write is enabled
1	dma_rd_en	R/W 0x0	DMA Read Enable 0x0 = DMA read is disabled 0x1 = DMA read is enabled

Table 426: Serial Interface Configuration 2 Register (Conf2) (Continued)

Bits	Field	Type/ HW Rst	Description
0	srst	R/W 0x0	Soft Reset Allows firmware to reset the hardware. After setting this bit to 1, firmware has to reset this bit to 0 before starting any transfer. 0x0 = hardware is not in reset 0x1 = hardware is in reset

A.8.2.12 Serial Interface Interrupt Status Register (ISR)

Instance Name	Offset
ISR	0x2C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																				wfifo_ovrflw_is	wfifo_undrflw_is	rfifo_ovrflw_is	rfifo_undrflw_is	wfifo_full_is	wfifo_empty_is	rfifo_full_is	rfifo_empty_is	wfifo_dma_burst_is	rfifo_dma_burst_is	xfer_rdy_is	xfer_done_is
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0

Table 427: Serial Interface Interrupt Status Register (ISR)

Bits	Field	Type/ HW Rst	Description
31:12	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
11	wfifo_ovrflw_is	R 0x0	Write FIFO Overflow Interrupt Status 0x0 = write FIFO is not overflowed after masking 0x1 = write FIFO is overflowed after masking
10	wfifo_undrflw_is	R 0x0	Write FIFO Underflow Interrupt Status 0x0 = write FIFO is not underflowed after masking 0x1 = write FIFO is underflowed after masking
9	rfifo_ovrflw_is	R 0x0	Read FIFO Overflow Interrupt Status 0x0 = read FIFO is not overflowed after masking 0x1 = read FIFO is overflowed after masking
8	rfifo_undrflw_is	R 0x0	Read FIFO Underflow Interrupt Status 0x0 = read FIFO is not underflowed after masking 0x1 = read FIFO is underflowed after masking

Table 427: Serial Interface Interrupt Status Register (ISR) (Continued)

Bits	Field	Type/ HW Rst	Description
7	wfifo_full_is	R 0x0	Write FIFO Full Interrupt Status 0x0 = write FIFO is not fulled after masking 0x1 = write FIFO is fulled after masking
6	wfifo_empty_is	R 0x0	Write FIFO Empty Interrupt Status 0x0 = write FIFO is not emptied after masking 0x1 = write FIFO is emptied after masking
5	rfifo_full_is	R 0x0	Read FIFO Full Interrupt Status 0x0 = read FIFO is not fulled after masking 0x1 = read FIFO is fulled after masking
4	rfifo_empty_is	R 0x0	Read FIFO Empty Interrupt Status 0x0 = read FIFO is not emptied after masking 0x1 = read FIFO is empty after masking
3	wfifo_dma_burst_is	R 0x0	Write FIFO DMA Burst Interrupt Status 0x0 = number of unused entries in the write FIFO is less than DMA_WR_BURST (R28h [13:12]) after masking 0x1 = number of unused entries in the write FIFO is greater than or equal to DMA_WR_BURST (R28h [13:12]) after masking
2	rfifo_dma_burst_is	R 0x0	Read FIFO DMA Burst Interrupt Status 0x0 = number of available data in the read FIFO is less than DMA_RD_BURST (R28h [9:8]) after masking 0x1 = number of available data in the read FIFO is greater than or equal to DMA_RD_BURST (R28h [9:8]) after masking
1	xfer_rdy_is	R 0x0	Serial Interface Transfer Ready Interrupt Status 0x0 = serial interface is currently transferring data after masking 0x1 = serial interface is ready for a new transfer after masking
0	xfer_done_is	R 0x0	Transfer Done Interrupt Status 0x0 = transfer has not completed after masking 0x1 = transfer has completed after masking

A.8.2.13 Serial Interface Interrupt Mask Register (IMR)

Instance Name	Offset
IMR	0x30

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																				wfifo_ovrflw_im	wfifo_undrflw_im	rfifo_ovrflw_im	rfifo_undrflw_im	wfifo_full_im	wfifo_empty_im	rfifo_full_im	rfifo_empty_im	wfifo_dma_burst_im	rfifo_dma_burst_im	xfer_rdy_im	xfer_done_im
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	1	1	1	1	1	1	1	1	1	1	1

Table 428: Serial Interface Interrupt Mask Register (IMR)

Bits	Field	Type/ HW Rst	Description
31:12	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
11	wfifo_ovrflw_im	R 0x1	Write FIFO Overflow Interrupt Mask 0x0 = write FIFO overflow interrupt is not masked 0x1 = write FIFO overflow interrupt is masked
10	wfifo_undrflw_im	R 0x1	Write FIFO Underflow Interrupt Mask 0x0 = write FIFO underflow interrupt is not masked 0x1 = write FIFO underflow interrupt is masked
9	rfifo_ovrflw_im	R 0x1	Read FIFO Overflow Interrupt Mask 0x0 = read FIFO overflow interrupt is not masked 0x1 = read FIFO overflow interrupt is masked
8	rfifo_undrflw_im	R 0x1	Read FIFO Underflow Interrupt Mask 0x0 = read FIFO underflow interrupt is not masked 0x1 = read FIFO underflow interrupt is masked
7	wfifo_full_im	R 0x1	Write FIFO Full Interrupt Mask 0x0 = write FIFO full interrupt is not masked 0x1 = write FIFO full interrupt is masked
6	wfifo_empty_im	R 0x1	Write FIFO Empty Interrupt Mask 0x0 = write FIFO empty interrupt is not masked 0x1 = write FIFO empty interrupt is masked
5	rfifo_full_im	R 0x1	Read FIFO Full Interrupt Mask 0x0 = read FIFO full interrupt is not masked 0x1 = read FIFO full interrupt is masked
4	rfifo_empty_im	R 0x1	Read FIFO Empty Interrupt Mask 0x0 = read FIFO empty interrupt is not masked 0x1 = read FIFO empty interrupt is masked

Table 428: Serial Interface Interrupt Mask Register (IMR) (Continued)

Bits	Field	Type/ HW Rst	Description
3	wfifo_dma_burst_im	R 0x1	Write FIFO DMA Burst Interrupt Mask 0x0 = write FIFO DMA burst interrupt is not masked 0x1 = write FIFO DMA burst interrupt is masked
2	rfifo_dma_burst_im	R 0x1	Read FIFO DMA Burst Interrupt Mask 0x0 = read FIFO DMA burst interrupt is not masked 0x1 = read FIFO DMA burst interrupt is masked
1	xfer_rdy_im	R 0x1	Serial Interface Transfer Ready Mask 0x0 = transfer ready interrupt is not masked 0x1 = transfer ready interrupt is masked
0	xfer_done_im	R 0x1	Transfer Done Interrupt Mask 0x0 = transfer done interrupt is not masked 0x1 = transfer done interrupt is masked

A.8.2.14 Serial Interface Interrupt Raw Status Register (IRSR)

Instance Name	Offset
IRSR	0x34

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																				wfifo_ovrflw_ir	wfifo_undrflw_ir	rfifo_ovrflw_ir	rfifo_undrflw_ir	wfifo_full_ir	wfifo_empty_ir	rfifo_full_ir	rfifo_empty_ir	wfifo_dma_burst_ir	rfifo_dma_burst_ir	xfer_rdy_ir	xfer_done_ir
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	1	0	1	1	0	1	1

Table 429: Serial Interface Interrupt Raw Status Register (IRSR)

Bits	Field	Type/ HW Rst	Description
31:12	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
11	wfifo_ovrflw_ir	R 0x0	Write FIFO Overflow Interrupt Raw 0x0 = write FIFO is not overflowed before masking 0x1 = write FIFO is overflowed before masking
10	wfifo_undrflw_ir	R 0x0	Write FIFO Underflow Interrupt Raw 0x0 = write FIFO is not underflowed before masking 0x1 = write FIFO is underflowed before masking

Table 429: Serial Interface Interrupt Raw Status Register (IRSR) (Continued)

Bits	Field	Type/ HW Rst	Description
9	rfifo_ovrflw_ir	R 0x0	Read FIFO Overflow Interrupt Raw 0x0 = read FIFO is not overflowed before masking 0x1 = read FIFO is overflowed before masking
8	rfifo_undrflw_ir	R 0x0	Read FIFO Underflow Interrupt Raw 0x0 = read FIFO is not underflowed before masking 0x1 = read FIFO is underflowed before masking
7	wfifo_full_ir	R 0x0	Write FIFO Full Interrupt Raw 0x0 = write FIFO is not fulled before masking 0x1 = write FIFO is fulled before masking
6	wfifo_empty_ir	R 0x1	Write FIFO Empty Interrupt Raw 0x0 = write FIFO is not emptied before masking 0x1 = write FIFO is emptied before masking
5	rfifo_full_ir	R 0x0	Read FIFO Full Interrupt Raw 0x0 = read FIFO is not fulled before masking 0x1 = read FIFO is fulled before masking
4	rfifo_empty_ir	R 0x1	Read FIFO Empty Interrupt Raw 0x0 = read FIFO is not emptied before masking 0x1 = read FIFO is empty before masking
3	wfifo_dma_burst_ir	R 0x1	Write FIFO DMA Burst Interrupt Raw 0x0 = number of unused entries in the write FIFO is less than DMA_WR_BURST (R28h [13:12]) before masking 0x1 = number of unused entries in the write FIFO is greater than or equal to DMA_WR_BURST (R28h [13:12]) before masking
2	rfifo_dma_burst_ir	R 0x0	Read FIFO DMA Burst Interrupt Raw 0x0 = number of available data in the read FIFO is less than DMA_RD_BURST (R28h [9:8]) before masking 0x1 = number of available data in the read FIFO is greater than or equal to DMA_RD_BURST (R28h [9:8]) before masking
1	xfer_rdy_ir	R 0x1	Serial Interface Transfer Ready Raw 0x0 = serial interface is currently transferring data before masking 0x1 = serial interface is ready for a new transfer before masking
0	xfer_done_ir	R 0x1	Transfer Done Interrupt Raw 0x0 = transfer has not completed before masking 0x1 = transfer has completed before masking

A.8.2.15 Serial Interface Interrupt Clear Register (ISC)

Instance Name	Offset
ISC	0x38

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

Table 430: Serial Interface Interrupt Clear Register (ISC)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	xfer_done_ic	W 0x0	Transfer Done Interrupt Clear 0x0 = transfer done interrupt is not cleared 0x1 = transfer done interrupt is cleared

A.9 SSP Address Block

A.9.1 SSP Register Map

Table 431: SSP Register Map

Offset	Name	HW Rst	Description	Details
0x00	SSCR0	0x0000_0000	SSP Control Register 0	Page: 614
0x04	SSCR1	0x0000_0000	SSP Control Register 1	Page: 616
0x08	SSSR	0x0000_F004	SSP Status Register	Page: 619
0x0C	SSITR	0x0000_0000	SSP Interrupt Test Register	Page: 622
0x10	SSDR	0x0000_0000	SSP Data Register	Page: 623
0x28	Reserved	0x0000_0000	Reserved	--
0x2C	SSPSP	0x0000_0000	SSP Programmable Serial Protocol Register	Page: 624
0x30	SSTSA	0x0000_0000	SSP TX Time Slot Active Register	Page: 626
0x34	SSRSA	0x0000_0000	SSP RX Time Slot Active Register	Page: 627
0x38	SSTSS	0x0000_0000	SSP Time Slot Status Register	Page: 628

A.9.2 SSP Registers

A.9.2.1 SSP Control Register 0 (SSCR0)

The SSP Control 0 registers contain 13 different bit fields that control various functions within the SSP port. The following table shows the bit locations corresponding to the different control bit fields within the SSP Control 0 Register. The reset state of all bits are as shown, but they must be programmed to their preferred values before enabling the SSP port.

Writes to reserved bits must be 0b0, and read values of reserved bits are undefined.

Instance Name	Offset
SSCR0	0x00

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	mod	Reserved	fpcke	rhcd	mcrt	frdc			tim	rim	Reserved	edss	Reserved												sse	Reserved	frf		dss				
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 432: SSP Control Register 0 (SSCR0)

Bits	Field	Type/ HW Rst	Description
31	mod	R/W 0x0	Mode 0x0 = normal SSP mode 0x1 = network mode
30	Reserved	R/W 0x0	Reserved. Do not change the reset value.
29	fpcke	R/W 0x0	FIFO Packing Enable 0x0 = FIFO packing mode disabled 0x1 = FIFO packing mode enabled
28	rhcd	R/W 0x0	RX Half Cycle Delay 0x0 = not seclect RX half cycle delay 0x1 = receive data delay half cycle
27	mcrt	R/W 0x0	Master Clock Return 0x0 = not seclect master return_clock 0x1 = master clock delay
26:24	frdc	R/W 0x0	Frame Rate Divider Control Value of 0x0-0x7 specifies the number of time slots per frame when in network mode (the actual number of time slots is this field +1, so 1 to 8 time slots can be specified).
23	tim	R/W 0x0	Transmit FIFO Underrun Interrupt Mask 0x0 = TUR events generate an SSP interrupt 0x1 = TUR events do NOT generate an SSP interrupt
22	rim	R/W 0x0	Receive FIFO Overrun Interrupt Mask 0x0 = ROR events generate an SSP interrupt 0x1 = ROR events do NOT generate an SSP interrupt

Table 432: SSP Control Register 0 (SSCR0) (Continued)

Bits	Field	Type/ HW Rst	Description
21	Reserved	R/W 0x0	Reserved. Do not change the reset value.
20	edss	R/W 0x0	Extended Data Size Select 0x0 = a 0 is pre-appended to the DSS value to set the DSS range from 8 to 16 bits 0x1 = a 1 is pre-appended to the DSS value to set the DSS range from 18 to 32 bits
19:8	Reserved	R/W 0x0	Reserved. Do not change the reset value.
7	sse	R/W 0x0	Synchronous Serial Port Enable 0x0 = SSPx port is disabled 0x1 = SSPx port is enabled
6	Reserved	R/W 0x0	Reserved. Do not change the reset value.
5:4	frf	R/W 0x0	Frame Format This field must be written with 0x3 = to select the PSP format. 0x0 = Motorola* Serial Peripheral Interface (SPI) 0x1 = Texas Instruments* Synchronous Serial Protocol (SSP) 0x2 = National Semiconductor* microwire 0x3 = Programmable Serial Protocol (PSP)
3:0	dss	R/W 0x0	Data Size Select EDSS DSS Data Size 1 0001 18-bit data 1 1111 32-bit data EDSS DSS Data Size 0 0011 4-bit data 0 0111 8-bit data 0 1111 16-bit data

A.9.2.2 SSP Control Register 1 (SSCR1)

The SSP Port Control 1 registers contain bit fields that control various SSP port functions. The following table shows bit locations corresponding to control bit fields in SSCR1. The reset state of all bits is shown, but must be set to the preferred value before enabling the SSP port by setting the <Synchronous Serial Port Enable> field in the SSP Control Register 0.

Write 0b0 to reserved bits, the read values of reserved bits are undetermined.

Instance Name	Offset
SSCR1	0x04

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	ttelp	tte	ebcei	scfr	ecra	ecrb	sclkdir	sfrmdir	rwot	Reserved	tsre	rsre	Reserved		Reserved	ifs	strf	efwr	rft				tft				mwds	sph	spo	lbm	tie	rie
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 433: SSP Control Register 1 (SSCR1)

Bits	Field	Type/ HW Rst	Description
31	ttelp	R/W 0x0	TXD 3-state Enable On Last Phase 0x0 = TXDx is 3-stated 1/2 clock cycle after the beginning of the LSb 0x1 = TXDx output signal is 3-stated on the clock edge that ends the LSb
30	tte	R/W 0x0	TXD 3-State Enable 0x0 = TXDx output signal is not 3-stated 0x1 = TXD is 3-stated when not transmitting data
29	ebcei	R/W 0x0	Enable Bit Count Error Interrupt 0x0 = interrupt due to a bit count error is disabled 0x1 = interrupt due to a bit count error is enabled
28	scfr	R/W 0x0	Slave Clock Free Running 0x0 = clock input to SSPSCLKx is continuously running 0x1 = clock input to SSPSCLKx is only active during data transfers(required for slave mode.)
27	ecra	R/W 0x0	Enable Clock Request A 0x0 = clock request from other SSPx is disabled 0x1 = clock request from other SSPx is enabled
26	ecrb	R/W 0x0	Enable Clock Request B 0x0 = clock request from other SSPx is disabled 0x1 = clock request from other SSPx is enabled
25	sclkdir	R/W 0x0	SSP Serial Bit Rate Clock (SSPSCLKx) Direction 0x0 = master mode, SSPx port drives SSPSCLKx 0x1 = slave mode, SSPx port receives SSPSCLKx

Table 433: SSP Control Register 1 (SSCR1) (Continued)

Bits	Field	Type/ HW Rst	Description
24	sfrmdir	R/W 0x0	SSP Frame (SSPSFRMx) Direction 0x0 = master mode, SSPx port drives SSPSFRMx 0x1 = slave mode, SSPx port receives SSPSFRMx
23	rwot	R/W 0x0	Receive Without Transmit 0x0 = transmit/Receive mode 0x1 = receive without transmit mode
22	Reserved	RSVD --	Reserved Always write 0. Ignore read value.
21	tsre	R/W 0x0	Transmit Service Request Enable 0x0 = DMA service request is disabled 0x1 = DMA service request is enabled
20	rsre	R/W 0x0	Receive Service Request Enable 0x0 = DMA service request is disabled 0x1 = DMA service request is enabled
19:17	Reserved	RSVD --	Reserved Always write 0. Ignore read value.
16	ifs	R/W 0x0	Invert Frame Signal 0x0 = SSPSFRMx polarity is determined by the PSP polarity bits 0x1 = SSPSFRMx is inverted from normal-SSPSFRMx (as defined by the PSP polarity bits). (Works in all frame formats: SPI, SSP, and PSP)
15	strf	R/W 0x0	Select FIFO For Efwr Select FIFO For Efwr (Test Mode Bit). Only when EFWR = 1 0x0 = TXFIFO is selected for both writes and reads through the SSP data register 0x1 = RXFIFO is selected for both writes and reads through the SSP data register
14	efwr	R/W 0x0	Enable FIFO Write/read Enable FIFO Write/read (Test Mode Bit) 0x0 = FIFO write/read special function is disabled (normal SSPx operational mode) 0x1 = FIFO write/read special function is enabled
13:10	rft	R/W 0x0	RXFIFO Trigger Threshold Sets threshold level at which RXFIFO asserts interrupt. Level should be set to the preferred threshold value minus 1.
9:6	tft	R/W 0x0	TXFIFO Trigger Threshold Sets threshold level at which TXFIFO asserts interrupt. Level should be set to the preferred threshold value minus 1.
5	mwds	R/W 0x0	Microwire Date Size 0x0 = 8 bits 0x1 = 16 bits

Table 433: SSP Control Register 1 (SSCR1) (Continued)

Bits	Field	Type/ HW Rst	Description
4	sph	R/W 0x0	Motorola SPI SSPSCLK Phase Setting 0x0 = SSPSCLKx is inactive until one cycle after the start of a frame and active until 1/2 cycle before the end of a frame 0x1 = SSPSCLKx is inactive until 1/2 cycle after the start of a frame and active until one cycle
3	spo	R/W 0x0	Motorola SPI SSPSCLK Polarity Setting 0x0 = the inactive or idle state of SSPSCLKx is low 0x1 = the inactive or idle state of SSPSCLKx is high
2	lbm	R/W 0x0	Loopback Mode Loopback Mode (Test Mode Bit) 0x0 = normal serial port operation is enabled 0x1 = output of TX serial shifter is internally connected to input of RX serial shifter
1	tie	R/W 0x0	Transmit FIFO Interrupt Enable 0x0 = TXFIFO threshold-level-reached interrupt is disabled 0x1 = TXFIFO threshold-level-reached interrupt is enabled
0	rie	R/W 0x0	Receive FIFO Interrupt Enable 0x0 = RXFIFO threshold-level-reached interrupt is disabled 0x1 = RXFIFO threshold-level-reached interrupt is enabled

A.9.2.3 SSP Status Register (SSSR)

The SSP Port Status registers contain bits that signal overrun errors as well as the TXFIFO and RXFIFO service requests. Each of these hardware-detected events signals an interrupt request to the interrupt controller, or a DMA request. The Status register also contains flags that indicate if the SSPx port is actively transmitting data, if the TXFIFO is not full, and if the RXFIFO is not empty. A signal- interrupt signal is sent to the interrupt controller for each SSPx port. These events can cause an interrupt request or a DMA request: RXFIFO overrun, RXFIFO service request, and TXFIFO service request.

Bits that cause an interrupt request remain set until they are cleared by writing a 0b1 to each bit. Once a status bit is cleared, the interrupt is cleared. Read-write bits are called status bits (status bits are referred to as sticky and once set by hardware, they can only be cleared by writing a 0b1 to each bit), read-only bits are called flags. Writing a 0b1 to a sticky status bit clears it, writing a 0b0 has no effect. Read-only flags are set to 0b1 and are cleared automatically to 0b0 by hardware, and writes have no effect. Some bits that cause interrupt requests have corresponding mask bits in the Control registers and are indicated in the section headings that follow.

The following table shows the bit locations corresponding to the status and flag bits within the SSP Port Status Register. All bits are read-only except the <Receive FIFO Overrun>, <Transmit FIFO Underrun>, and <Bit Count Error>, which are all read-write. The reset state of read-write bits is 0b0 and all bits return to their reset state when <Synchronous Serial Port Enable> field in the SSP Control Register 0 is cleared.

Write 0b0 to reserved bits, reads from reserved bits are undetermined.

Instance Name	Offset
SSSR	0x08

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	oss	tx_oss	Reserved						bce	css	tur	Reserved					rfl				tfl				ror	rfs	tfs	bsy	rne	tnf	Reserved	
HW Rst	0	0	?	?	?	?	?	?	0	0	0	0	0	0	?	?	1	1	1	1	0	0	0	0	0	0	0	0	0	1	?	?

Table 434: SSP Status Register (SSSR)

Bits	Field	Type/ HW Rst	Description
31	oss	R 0x0	Odd Sample Status **Note:** that this bit needs to be looked at only when FIFO Packing is enabled (<FIFO Packing Enable> field in SSP Control Register 0 is set). Otherwise this bit is zero. When SSPx port is in Packed mode, and the CPU is used instead of DMA to read the RxFIFO, CPU should make sure that <Receive FIFO Not Empty>=1 AND this field=0 before it attempts to read the RxFIFO. 0x0 = RxFIFO entry has 2 samples 0x1 = RxFIFO entry has 1 sample

Table 434: SSP Status Register (SSSR) (Continued)

Bits	Field	Type/ HW Rst	Description
30	tx_oss	R 0x0	TX FIFO Odd Sample Status When SSPx port is in packed mode, the number of samples in the TX FIFO is: (<Transmit FIFO Level>*2 + this field), when <Transmit FIFO Not Full>=1 32, when <Transmit FIFO Not Full>=0.The TX FIFO cannot accept new data when <Transmit FIFO Not Full>=1 and <Transmit FIFO Level>=15 and this field=1. (The TX FIFO has 31 samples). **Note:** that this bit needs to be read only when FIFO Packing is enabled (<FIFO Packing Enable> in the SSP Control Register 0 set). Otherwise this bit is zero. 0x0 = TxFIFO entry has an even number of samples 0x1 = TxFIFO entry has an odd number of sample
29:24	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
23	bce	R/W 0x0	Bit Count Error 0x0 = the SSPx port has not experienced a bit count error 0x1 = the SSPSFRMx signal was asserted when the bit counter was not zero
22	css	R 0x0	Clock Synchronization Status 0x0 = the SSPx port is ready for slave clock operations 0x1 = the SSPx port is currently busy synchronizing slave mode signals
21	tur	R/W 0x0	Transmit FIFO Underrun 0x0 = the TXFIFO has not experienced an underrun 0x1 = a read from the TXFIFO was attempted when the TXFIFO was empty, causes an interrupt if it is enabled (<Transmit FIFO underrun interrupt Mask> in the SSP control register 0 is 0)
20:16	Reserved	RSVD --	Reserved Always write 0. Ignore read value.
15:12	rfl	R 0xF	Receive FIFO Level Number of entries minus one in RXFIFO. **Note:** When the value 0xF is read, the RXFIFO is either empty or full, and software should read the <Receive FIFO Not Empty> field.
11:8	tfl	R 0x0	Transmit FIFO Level Number of entries in TXFIFO. **Note:** When the value 0x0 is read, the TXFIFO is either empty or full, and software should read the <Transmit FIFO Not Full> field.
7	ror	R/W 0x0	Receive FIFO Overrun 0x0 = RXFIFO has not experienced an overrun 0x1 = attempted data write to full RXFIFO, causes an interrupt request

Table 434: SSP Status Register (SSSR) (Continued)

Bits	Field	Type/ HW Rst	Description
6	rfs	R 0x0	Receive FIFO Service Request 0x0 = RXFIFO level is at or below RFT threshold (RFT), or SSPx port is disabled 0x1 = RXFIFO level exceeds RFT threshold (RFT), causes an interrupt request
5	tfs	R 0x0	Transmit FIFO Service Request 0x0 = TX FIFO level exceeds the TFT threshold (TFT + 1), or SSPx port disabled 0x1 = TXFIFO level is at or below TFT threshold (TFT + 1), causes an interrupt request
4	bsy	R 0x0	SSP Busy 0x0 = SSPx port is idle or disabled 0x1 = SSPx port is currently transmitting or receiving framed data
3	rne	R 0x0	Receive FIFO Not Empty 0x0 = RXFIFO is empty 0x1 = RXFIFO is not empty
2	tnf	R 0x1	Transmit FIFO Not Full 0x0 = TXFIFO is full 0x1 = TXFIFO is not full
1:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.9.2.4 SSP Interrupt Test Register (SSITR)

Setting the <Test TXFIFO Service Request> field generates a non-maskable interrupt request and a DMA service request for the TXFIFO.

Write 0b0 to reserved bits, reads from reserved bits are undetermined.

Instance Name	Offset
SSITR	0x0C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																								tror	trfs	ttfs	Reserved				
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	?	?	?	?	?

Table 435: SSP Interrupt Test Register (SSITR)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7	tror	R/W 0x0	Test RXFIFO Overrun 0x0 = no RXFIFO-overrun service request 0x1 = generates a non-maskable RXFIFO-overrun interrupt request. no DMA request is generated
6	trfs	R/W 0x0	Test RXFIFO Service Request 0x0 = no RXFIFO-service request 0x1 = generates a non-maskable RXFIFO-service interrupt request and DMA request
5	ttfs	R/W 0x0	Test TXFIFO Service Request 0x0 = no TXFIFO-service request 0x1 = generates a non-maskable TXFIFO-service interrupt request and DMA request
4:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.9.2.5 SSP Data Register (SSDR)

The SSP Data Registers are each two physical registers that have a common address. One SSDR_x is temporary storage for data that is transferred automatically into the TXFIFO, the other SSDR_x is temporary storage for data that is transferred automatically from the RXFIFO.

As programmed I/O or DMA access the SSDR_x, the TXFIFO or RXFIFO control logic transfers data automatically between the SSDR_x and the FIFO as fast as the system moves it. Data in the TXFIFO shifts up to accommodate new data that is written to the SSDR_X, unless it is an attempted write to a full TXFIFO. Data in the RXFIFO shifts down to accommodate data that is read from the SSP Data Register. The <Transmit FIFO Level>, <Receive FIFO Level>, <Receive FIFO Not Empty>, and <Transmit FIFO Not Full> fields in the SSP Status Register show whether the FIFO is full, above/below a programmable FIFO trigger threshold level, or empty.

When using programmed I/O, data can be written to the SSP Data Register anytime the TXFIFO falls below its trigger threshold level.

When a data sample size of less than 32-bits is selected, or 16 bits for packed mode, software should right-justify the data that is written to the SSP Data Register for automatic insertion into the TXFIFO. The transmit logic left-justifies the data and ignores any unused bits. Received data of less than 32 bits is right-justified automatically in the RXFIFO (cannot perform a write in packed mode of less than 32 bits wide). The TXFIFO and RXFIFO are cleared to 0b0 when the SSPx port is reset or disabled (by writing a 0b0 to the <Synchronous Serial Port Enable> field in the SSP Control Register 0).

The reset state of SSDR_x is undetermined. The following table shows the location of the SSPx port SSDR_x.

Instance Name	Offset
SSDR	0x10

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

Table 436: SSP Data Register (SSDR)

Bits	Field	Type/ HW Rst	Description
31:0	data	R/W 0x0	DATA Data to be written to the TXFIFO read from the RXFIFO

A.9.2.6 SSP Programmable Serial Protocol Register (SSPSP)

The SSP Programmable Serial Protocol registers contain 8 fields that program the various programmable serial-protocol (PSP) parameters. When using Programmable Serial Protocol (PSP) format in network mode, the parameters <Serial Frame Delay>, <Serial Frame Delay>, <Start Delay>, <Dummy Stop>, <Extended Dummy Stop>, <Dummy Start>, and <Extended Dummy Start> must be set to 0b0. The other parameters <Serial Frame Polarity>, <Serial Bit-rate Clock Mode>, <Frame Sync Relative Timing Bit>, and <Serial Frame Width> are programmable.

Writes 0b0 to reserved bits, reads from reserved bits are undetermined.

Instance Name	Offset
SSPSP	0x2C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved	edmystop			edmystrt		fsrt	dmystop		Reserved	sfrmwdth						sfrmdly							dmystrt		strtdly			etds	sfrmp	scmode	
HW Rst	?	0	0	0	0	0	0	0	0	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 437: SSP Programmable Serial Protocol Register (SSPSP)

Bits	Field	Type/ HW Rst	Description
31	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
30:28	edmystop	R/W 0x0	Extended Dummy Stop The most-significant bits of the dummy stop delay **Note:** Do not use in PSP Network mode.
27:26	edmystrt	R/W 0x0	Extended Dummy Start The most-significant bits of the dummy start delay **Note:** Do not use in PSP Network mode.
25	fsrt	R/W 0x0	Frame Sync Relative Timing Bit 0x0 = next frame is asserted after the end of the DMTSTOP timing 0x1 = next frame is asserted with the LSb of the previous frame
24:23	dmystop	R/W 0x0	Dummy Stop The least-significant bits of the dummy stop delay. Programmed value of <Extended Dummy Stop> + this field specifies the number (0-31) of active clocks (SSPSCLKx) that follow the end of the transmitted data. **Note:** Do not use in PSP Network mode.
22	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
21:16	sfrmwdth	R/W 0x0	Serial Frame Width Least-significant bits of the serial frame width Programmed value of this field specifies the frame width from 0x00 (one SSPSCLKx cycle) to 0x3F (63 SSPSCLKx cycles).

Table 437: SSP Programmable Serial Protocol Register (SSPSP) (Continued)

Bits	Field	Type/ HW Rst	Description
15:9	sfrmdly	R/W 0x0	Serial Frame Delay Programmed value specifies the number (0 -127) of active one-half clocks (SSPSCLKx) asserted from the most-significant bit of TXDx (output) or RXD (input) being driven to SSPSFRMx. **Note:** Do not use in PSP Network mode.
8:7	dmystrt	R/W 0x0	Dummy Start Least-significant bits of the dummy start delay Programmed value of this field specifies the number (0-15) of active clocks (SSPSCLKs) between the end of start delay and when the most-significant bit of transmit/receive data is driven **Note:** Do not use in PSP Network mode.
6:4	strtdly	R/W 0x0	Start Delay Programmed value specifies the number (0-7) of non-active clocks (SSPSCLKx) that define the duration of idle time **Note:** Do not use in PSP Network mode.
3	etds	R/W 0x0	End Of Transfer Data State
2	sfrmp	R/W 0x0	Serial Frame Polarity 0x0 = SSPSFRMx is active low (0b0) 0x1 = SSPSFRMx is active high (0b1)
1:0	scmode	R/W 0x0	Serial Bit-rate Clock Mode 0x0 = data driven (Falling), data sampled (Rising), idle state (Low) 0x1 = data driven (Rising), data sampled (Falling), idle state (Low) 0x2 = data driven (Rising), data sampled (Falling), idle state (High) 0x3 = data driven (Falling), data sampled (Rising), idle state (High)

A.9.2.7 SSP TX Time Slot Active Register (SSTSA)

Only used in Network mode (<Mode> in SSP Control Register 0 set), the read-write SSP TX Time Slot Active registers specify in which time slot the SSPx port transmits data.

The 8-bit <TX Time Slot Active> field specifies in which time slots the SSPx port transmits data and in which time slots the SSPx port does not transmit data. Bits beyond the <Frame Rate Divider Control> field in the SSP Control Register 0 value are ignored (for example, if <Frame Rate Divider Control>= 0x3, specifying that 4 time slots are used, then <TX Time Slot Active> bits 7:4 are ignored). If the <TXD 3-State Enable> field in the SSP Control Register 1 is set, the SSPx port 3-states the SSPTXDx interface output signal line during time slots that have associated TTSA bits programmed to 0b0.

Write 0b0 to reserved bits, reads from reserved bits are undetermined.

Instance Name	Offset
SSTSA	0x30

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

Table 438: SSP TX Time Slot Active Register (SSTSA)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7:0	ttsa	R/W 0x0	TX Time Slot Active 0x0 = SSPx port does NOT transmit data in this time slot 0x1 = SSPx port does transmit data in this time slot

A.9.2.8 SSP RX Time Slot Active Register (SSRSA)

Only used in Network mode (<Mode> in SSP Control Register 0 set), the read-write SSP RX Time Slot Active registers specify in which time slot the SSPx port receives data.

The 8-bit <RX Time Slot Active> field specifies in which time slots the SSPx port receives data and in which time slots the SSPx port does not receive data. Bits beyond the <Frame Rate Divider Control> field in the SSP Control Register 0 value are ignored. For example, if <Frame Rate Divider Control>=0x3, specifying that 4 time slots are used, then ?RX Time Slot Active> bits 7:4 are ignored.

Write 0b0 to reserved bits, reads from reserved bits are undetermined.

Instance Name	Offset
SSRSA	0x34

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

Table 439: SSP RX Time Slot Active Register (SSRSA)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7:0	rtsa	R/W 0x0	RX Time Slot Active 0x0 = SSPx port does not receive data in this time slot 0x1 = SSPx port receives data in this time slot

A.9.2.9 SSP Time Slot Status Register (SSTSS)

The <Network Mode Busy> field shows when the SSPx port is within a frame only when in Network mode (<Mode> in SSP Control Register 0 set). It can be used by software to determine when a clean shutdown of the SSPx port can be initiated. Software should:

1. Determine that the TXFIFO is either empty or is emptied at the end of the next frame.
2. Deactivate the TXFIFO DMA service requests.
3. Clear <Mode> in SSP Control Register 0 (to exit network mode).
4. Then poll <Network Mode Busy> until it is cleared before disabling the SSPx port by clearing the <Synchronous Serial Port Enable> field in the SSP Control Register 0.

When the SSPx port is a master of the frame signal (<SSP Frame (SSPSFRMx) Direction> field in SSP Control Register 1 set), <Network Mode Busy> is set as long as the port remains in Network mode. When the SSPx port is a slave of the frame signal, the <Network Mode Busy> field is cleared if the current frame (number of bits per sample * number of time slots per frame) has not expired since the last SSPSFRMx interface signal (in/out) was asserted.

Time Slot Status (TSS)

The 3-bit <Time Slot Status> field value identifies the time slot in which the SSPx port is operating. Due to synchronization between the SSPSCLKx domain and an internal bus clock domain, the TSS value becomes stable approximately two internal bus clock cycles after the beginning of the associated time slot. The <Time Slot Status> value is not valid if the <Network Mode Busy> field is cleared.

Write 0b0 to reserved bits, reads from reserved bits are undetermined.

Instance Name	Offset
SSTSS	0x38

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

Table 440: SSP Time Slot Status Register (SSTSS)

Bits	Field	Type/ HW Rst	Description
31	nmbsy	R 0x0	Network Mode Busy 0x0 = SSPx port is in network mode and no frame is currently active 0x1 = SSPx port is in network mode and a frame is currently active
30:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2:0	tss	R 0x0	Time Slot Status Value indicates which time slot is currently active. Because of synchronization between the SSPx port's SSPSCLKx domain and an internal bus clock domain, the value in this field becomes stable between the beginning and end of the currently active time slot.

A.10 UART Address Block

A.10.1 UART Register Map

Table 441: UART Register Map

Offset	Name	HW Rst	Description	Details
0x00	RBR	0x0000_0000	Receive Buffer Register	Page: 630
0x00	THR	0x0000_0000	Transmit Holding Register	Page: 631
0x00	DLL	0x0000_0002	Divisor Latch Low Byte Registers	Page: 632
0x04	DLH	0x0000_0000	Divisor Latch High Byte Registers	Page: 632
0x04	IER	0x0000_0000	Interrupt Enable Register	Page: 633
0x08	IIR	0x0000_0001	Interrupt Identification Register	Page: 634
0x08	FCR	0x0000_0000	FIFO Control Register	Page: 635
0x0C	LCR	0x0000_0000	Line Control Register	Page: 636
0x10	MCR	0x0000_0000	Modem Control Register	Page: 638
0x14	LSR	0x0000_0060	Line Status Register	Page: 639
0x18	MSR	0x0000_0000	Modem Status Register	Page: 640
0x1C	SCR	0x0000_0000	Scratchpad Register	Page: 641
0x20	ISR	0x0000_0000	Infrared Selection Register	Page: 641
0x24	RFOR	0x0000_0000	Receive FIFO Occupancy Register	Page: 642
0x28	ABR	0x0000_0000	Auto-Baud Control Register	Page: 643
0x2C	ACR	0x0000_0000	Auto-Baud Count Register	Page: 644

A.10.2 UART Registers

A.10.2.1 Receive Buffer Register (RBR)

Instance Name	Offset
RBR	0x00

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	byte_3								byte_2								byte_1								byte_0							
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 442: Receive Buffer Register (RBR)

Bits	Field	Type/ HW Rst	Description
31:24	byte_3	R 0x0	Byte 3 (valid Only in 32-bit Peripheral Bus mode)
23:16	byte_2	R 0x0	Byte 2 (valid Only in 32-bit Peripheral Bus mode)
15:8	byte_1	R 0x0	Byte 1 (valid Only in 32-bit Peripheral Bus mode)
7:0	byte_0	R 0x0	Byte 0

A.10.2.2 Transmit Holding Register (THR)

Instance Name	Offset
THR	0x00

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	byte_3								byte_2								byte_1								byte_0							
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 443: Transmit Holding Register (THR)

Bits	Field	Type/ HW Rst	Description
31:24	byte_3	W 0x0	Byte 3 (valid Only in 32-bit Peripheral Bus mode)
23:16	byte_2	W 0x0	Byte 2 (valid Only in 32-bit Peripheral Bus mode)
15:8	byte_1	W 0x0	Byte 1 (valid Only in 32-bit Peripheral Bus mode)
7:0	byte_0	W 0x0	Byte 0

A.10.2.3 Divisor Latch Low Byte Registers (DLL)

Instance Name	Offset
DLL	0x00

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

Table 444: Divisor Latch Low Byte Registers (DLL)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7:0	dll	R/W 0x2	DLL, Low-byte Compare Value to Generate Baud Rate

A.10.2.4 Divisor Latch High Byte Registers (DLH)

Instance Name	Offset
DLH	0x04

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

Table 445: Divisor Latch High Byte Registers (DLH)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7:0	dlh	R/W 0x0	DLH, High-byte Compare Value to Generate Baud Rate

A.10.2.5 Interrupt Enable Register (IER)

Instance Name	Offset
IER	0x04

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																							hse	dmae	uue	nrze	rtoie	mie	rlse	tie	ravie
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0

Table 446: Interrupt Enable Register (IER)

Bits	Field	Type/ HW Rst	Description
31:9	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
8	hse	R/W 0x0	High Speed UART Enable (HSE)
7	dmae	R/W 0x0	DMA Requests Enable 0x0 = DMA requests are disabled 0x1 = DMA requests are enabled
6	uue	R/W 0x0	UART Unit Enable 0x0 = the unit is disabled 0x1 = the unit is enabled
5	nrze	R/W 0x0	NRZ Coding Enable NRZ encoding/decoding is only used in UART mode, not in infrared mode. If the serial infrared receiver or transmitter is enabled, NRZ coding is disabled. 0x0 = NRZ coding disabled 0x1 = NRZ coding enabled
4	rtoie	R/W 0x0	Receiver Timeout Interrupt Enable (Source IIR[TOD]) 0x0 = receiver data timeout interrupt disabled 0x1 = receiver data timeout interrupt enabled
3	mie	R/W 0x0	Modem Interrupt Enable (Source IIR[IID]) 0x0 = modem status interrupt disabled 0x1 = modem status interrupt enabled
2	rlse	R/W 0x0	Receiver Line Status Interrupt Enable (Source IIR[IID]) 0x0 = receiver line status interrupt disabled 0x1 = receiver line status interrupt enabled
1	tie	R/W 0x0	Transmit Data Request Interrupt Enable(Source IIR[IID]) 0x0 = transmit FIFO data request interrupt disabled 0x1 = transmit FIFO data request interrupt enabled

Table 446: Interrupt Enable Register (IER) (Continued)

Bits	Field	Type/ HW Rst	Description
0	ravie	R/W 0x0	Receiver Data Available Interrupt Enable (Source IIR[IID]) 0x0 = receiver data availble(trigger threshold reached) interrupt disabled 0x1 = receiver data availble(trigger threshold reached) interrupt enabled

A.10.2.6 Interrupt Identification Register (IIR)

Instance Name	Offset
IIR	0x08

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																								fifoes10		eoc	abl	tod	iid10		nip
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	1

Table 447: Interrupt Identification Register (IIR)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7:6	fifoes10	R 0x0	FIFO Mode Enable Status 0x0 = Non-FIFO mode is selected 0x1 = reserved 0x2 = reserved 0x3 = FIFO mode is selected
5	eoc	R 0x0	DMA End of Descriptor Chain 0x0 = DMA has not signaled the end of its programmed descriptor chain 0x1 = DMA has signaled the end of its programmed descriptor chain
4	abl	R 0x0	Auto-baud Lock 0x0 = Auto-baud circuitry has not programmed divisor latch registers (DLR) 0x1 = divisor latch registers (DLR) programmed by auto-baud circuitry
3	tod	R 0x0	Timeout Detected 0x0 = no timeout interrupt is pending 0x1 = timeout interrupt is pending (FIFO mode only)

Table 447: Interrupt Identification Register (IIR) (Continued)

Bits	Field	Type/ HW Rst	Description
2:1	iid10	R 0x0	Interrupt Source Encoded 0x0 = modern Status (CTS) 0x1 = transmit FIFO request data 0x2 = receive data available 0x3 = receive error (overrun)
0	nip	R 0x1	Interrupt Is Pending 0x0 = interrupt is pending (active low) 0x1 = no interrupt is pending

A.10.2.7 FIFO Control Register (FCR)

Instance Name	Offset
FCR	0x08

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																								itl		bus	Reserved	til	resettf	resetrf	trfifoe
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	?	0	0	0	0

Table 448: FIFO Control Register (FCR)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7:6	itl	W 0x0	Interrupt Trigger Level When the number of the bytes in the receive FIFO equals the interrupt trigger threshold programmed into this field and the received-data-available interrupt is enabled with the IER, an interrupt is generated and appropriate bits are set in the IIR. The receive DMA request is also generated when the trigger threshold is reached. 0x0 = 1 byte or more in FIFO causes interrupt (Not valid in DMA mode) 0x1 = 8 bytes or more in FIFO cause interrupt and DMA request 0x2 = 16 bytes or more in FIFO causes interrupt and DMA request 0x3 = 32 bytes or more in FIFO causes interrupt and DMA request
5	bus	W 0x0	32-Bit Peripheral Bus 0x0 = 8-bit peripheral bus 0x1 = 32-bit peripheral bus, transmit and receive FIFO need to be enabled in this mode

Table 448: FIFO Control Register (FCR) (Continued)

Bits	Field	Type/ HW Rst	Description
4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3	til	W 0x0	Transmitter Interrupt Level 0x0 = interrupt/DMA request when FIFO is half empty 0x1 = interrupt/DMA request when FIFO is empty
2	resettf	W 0x0	Reset Transmit FIFO When this field is set, all the bytes in the transmit FIFO are cleared. The TDRQ bit in the LSR is set and the IIR shows a transmitter requests data interrupt, if the TIE bit in the IER register is set. The Transmit shift register is not cleared, and it complete the current transmission. 0x0 = writing 0 has no effect 0x1 = the transmit FIFO is cleared
1	resetrf	W 0x0	Reset Receive FIFO 0x0 = writing 0 has no effect 0x1 = the receive FIFO is cleared
0	trfifoe	W 0x0	Transmit and Receive FIFO Enable 0x0 = FIFOs are disabled 0x1 = FIFOs are enabled

A.10.2.8 Line Control Register (LCR)

Instance Name	Offset
LCR	0x0C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																								dlab	sb	stkyp	eps	pen	stb	wls10	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0

Table 449: Line Control Register (LCR)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7	dlab	R/W 0x0	Divisor Latch Access 0x0 = access transmit holding register, receive buffer register, and interrupt enable register 0x1 = access divisor latch registers

Table 449: Line Control Register (LCR) (Continued)

Bits	Field	Type/ HW Rst	Description
6	sb	R/W 0x0	Set Break 0x0 = no effect on TXD output 0x1 = forces TXD output to 0
5	stkyp	R/W 0x0	Sticky Parity 0x0 = no effect on parity bit 0x1 = forces parity bit to be opposite of EPS bit value
4	eps	R/W 0x0	Even Parity Select 0x0 = sends or checks for odd parity 0x1 = sends or checks for even parity
3	pen	R/W 0x0	Parity Enable 0x0 = no parity 0x1 = Parity
2	stb	R/W 0x0	Stop Bits 0x0 = 1 stop bit 0x1 = 2 stop bit
1:0	wls10	R/W 0x0	World Length Select 0x0 = 5-bit character 0x1 = 6-bit character 0x2 = 7-bit character 0x3 = 8-bit character

A.10.2.9 Modem Control Register (MCR)

Instance Name	Offset
MCR	0x10

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																										afe	loop	Reserfed		rts	Reserved
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0

Table 450: Modem Control Register (MCR)

Bits	Field	Type/ HW Rst	Description
31:6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
5	afe	R/W 0x0	Auto-flow Control Enable 0x0 = auto-RTS and auto-CTS are disabled 0x1 = auto-RTS and auto-CTS are enabled
4	loop	R/W 0x0	Loopback Mode 0x0 = normal UART operation 0x1 = loopback mode UART operation
3:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	rts	R/W 0x0	Request to Send 0x0 = non-auto-flow mode 0x1 = auto-flow mode
0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.10.2.10 Line Status Register (LSR)

Instance Name	Offset
LSR	0x14

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																								fifoe	temt	tdrq	bi	fe	pe	oe	dr
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	1	1	0	0	0	0	0

Table 451: Line Status Register (LSR)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7	fifoe	R 0x0	FIFO Error Status 0x0 = no FIFO or no errors in receive FIFO 0x1 = at least one character in receive FIFO has errors
6	temt	R 0x1	Transmitter Empty 0x0 = there is data in the transmit shift register, the transmit holding register, or the FIFO 0x1 = all the data in the transmitter has been shifted out
5	tdrq	R 0x1	Transmit Data Request 0x0 = there is data in the holding register or FIFO waiting to be shifted out 0x1 = transmit FIFO has half or less then half data
4	bi	R 0x0	Break Interrupt 0x0 = no break signal has been received 0x1 = break signal received
3	fe	R 0x0	Framing Error 0x0 = no framing error 0x1 = invalid stop bit has been detected
2	pe	R 0x0	Parity Error 0x0 = no parity error 0x1 = parity error has been detected
1	oe	R 0x0	Overrun Error 0x0 = no data has been lost 0x1 = receive data has been lost
0	dr	R 0x0	Data Ready 0x0 = no data has been received 0x1 = data is available in RBR or the FIFO

A.10.2.11 Modem Status Register (MSR)

Instance Name	Offset
MSR	0x18

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

Table 452: Modem Status Register (MSR)

Bits	Field	Type/ HW Rst	Description
31:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4	cts	R 0x0	Clear to Send 0x0 = nCTS pin is 1 0x1 = nCTS pin is 0
3:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	dcts	R 0x0	Delta Clear to Send 0x0 = no change in nCTS pin since last read of MSR 0x1 = nCTS pin has changed state

A.10.2.12 Scratchpad Register (SCR)

Instance Name	Offset
SCR	0x1C

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

Table 453: Scratchpad Register (SCR)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7:0	scratchpad	R/W 0x0	No Effect on UART Functions

A.10.2.13 Infrared Selection Register (ISR)

Instance Name	Offset
ISR	0x20

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																											rxpl	txpl	xmode	rcveir	xmitir
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0

Table 454: Infrared Selection Register (ISR)

Bits	Field	Type/ HW Rst	Description
31:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4	rxpl	R/W 0x0	Receive Data Polarity 0x0 = SIR decoder takes positive pulses as 0s 0x1 = SIR decoder takes negative pulses as 0s
3	txpl	R/W 0x0	Transmit Data Polarity 0x0 = SIR encoder generates a positive pulse for a data bit of 0 0x1 = SIR encoder generates a negetive pulse for a data bit of 0
2	xmode	R/W 0x0	Transmit Pulse Width Select 0x0 = transmit pulse width is 3/16 of a bit time wide 0x1 = transmit pulse width is 1.6 ms

Table 454: Infrared Selection Register (ISR) (Continued)

Bits	Field	Type/ HW Rst	Description
1	rcveir	R/W 0x0	Receiver SIR Enable 0x0 = receiver is in UART mode 0x1 = receiver is in infrared mode
0	xmitir	R/W 0x0	Transmitter SIR Enable 0x0 = transmitter is in UART mode 0x1 = transmitter is in infrared mode

A.10.2.14 Receive FIFO Occupancy Register (RFOR)

Instance Name	Offset
RFOR	0x24

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

Table 455: Receive FIFO Occupancy Register (RFOR)

Bits	Field	Type/ HW Rst	Description
31:6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
5:0	byte_count	R 0x0	Number of Bytes (0-63) Remaining in Receive FIFO In non-FIFO mode, 0 is returned.

A.10.2.15 Auto-Baud Control Register (ABR)

Instance Name	Offset
ABR	0x28

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																												abt	abup	ablie	abe
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0

Table 456: Auto-Baud Control Register (ABR)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3	abt	R/W 0x0	Auto-Baud Table 0x0 = formula used to calculate baud rates 0x1 = table used to calculate baud rates, which limits UART to choosing common baud rates
2	abup	R/W 0x0	Auto-baud Programmer Select 0x0 = CPU programs divisor latch register 0x1 = UART programs divisor latch register
1	ablie	R/W 0x0	Auto-baud-lock Interrupt Enable 0x0 = auto-baud-lock interrupt is disabled 0x1 = auto-baud-lock interrupt is enabled
0	abe	R/W 0x0	Auto-baud Enable 0x0 = auto-baud disabled 0x1 = auto-baud enabled

A.10.2.16 Auto-Baud Count Register (ACR)

Instance Name	Offset
ACR	0x2C

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

Table 457: Auto-Baud Count Register (ACR)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	count_value	R 0x0	Number of 14.857 MHz Clock Cycles Within a Start-Bit Pulse

A.11 GPIO Address Block

A.11.1 GPIO Register Map

Table 458: GPIO Register Map

Offset	Name	HW Rst	Description	Details
0x00	GPIO_GPLR_REG0	0x0000_0000	GPIO Pin Level Register0	Page: 647
0x04	GPIO_GPLR_REG1	0x0000_0000	GPIO Pin Level Register1	Page: 647
0x0C	GPIO_GPDR_REG0	0x0000_0000	GPIO Pin Direction Register0	Page: 648
0x10	GPIO_GPDR_REG1	0x0000_0000	GPIO Pin Direction Register1	Page: 648
0x18	GPIO_GPSR_REG0	0x0000_0000	GPIO Pin Output Set Register 0	Page: 649
0x1C	GPIO_GPSR_REG1	0x0000_0000	GPIO Pin Output Set Register 1	Page: 649
0x24	GPIO_GPCR_REG0	0x0000_0000	GPIO Pin Output Clear Register 0	Page: 650
0x28	GPIO_GPCR_REG1	0x0000_0000	GPIO Pin Output Clear Register 1	Page: 650
0x30	GPIO_GRER_REG0	0x0000_0000	GPIO Rising Edge Detect Enable Register 0	Page: 651
0x34	GPIO_GRER_REG1	0x0000_0000	GPIO Rising Edge Detect Enable Register 1	Page: 651
0x3C	GPIO_GFER_REG0	0x0000_0000	GPIO Falling Edge Detect Enable Register 0	Page: 652
0x40	GPIO_GFER_REG1	0x0000_0000	GPIO Falling Edge Detect Enable Register 1	Page: 652
0x48	GPIO_GEDR_REG0	0x0000_0000	GPIO Edge Detect Status Register 0	Page: 653
0x4C	GPIO_GEDR_REG1	0x0000_0000	GPIO Edge Detect Status Register 1	Page: 653
0x54	GPIO_GSDR_REG0	0x0000_0000	GPIO Pin Bitwise Set Direction Register 0	Page: 654
0x58	GPIO_GSDR_REG1	0x0000_0000	GPIO Pin Bitwise Set Direction Register 1	Page: 654
0x60	GPIO_GCDR_REG0	0x0000_0000	GPIO Pin Bitwise Clear Direction Register 0	Page: 655
0x64	GPIO_GCDR_REG1	0x0000_0000	GPIO Pin Bitwise Clear Direction Register 1	Page: 655
0x6C	GPIO_GSRER_REG0	0x0000_0000	GPIO Bitwise Set Rising Edge Detect Enable Register 0	Page: 656
0x70	GPIO_GSRER_REG1	0x0000_0000	GPIO Bitwise Set Rising Edge Detect Enable Register 1	Page: 656
0x78	GPIO_GCRER_REG0	0x0000_0000	GPIO Bitwise Clear Rising Edge Detect Enable Register 0	Page: 657
0x7C	GPIO_GCRER_REG1	0x0000_0000	GPIO Bitwise Clear Rising Edge Detect Enable Register 1	Page: 657
0x84	GPIO_GSFER_REG0	0x0000_0000	GPIO Bitwise Set Falling Edge Detect Enable Register 0	Page: 658

Table 458: GPIO Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x88	GPIO_GSFER_REG1	0x0000_0000	GPIO Bitwise Set Falling Edge Detect Enable Register 1	Page: 658
0x90	GPIO_GCFER_REG0	0x0000_0000	GPIO Bitwise Clear Falling Edge Detect Enable Register 0	Page: 659
0x94	GPIO_GCFER_REG1	0x0000_0000	GPIO Bitwise Clear Falling Edge Detect Enable Register 1	Page: 659
0x9C	APMASK_REG0	0x0000_0000	GPIO Bitwise Mask of Edge Detect Status Register 0	Page: 660
0xA0	APMASK_REG1	0x0000_0000	GPIO Bitwise Mask of Edge Detect Status Register 1	Page: 660

A.11.2 GPIO Registers

A.11.2.1 GPIO Pin Level Register0 (GPIO_GPLR_REG0)

Instance Name	Offset
GPIO_GPLR_REG0	0x00

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	gplr_reg0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 459: GPIO Pin Level Register0 (GPIO_GPLR_REG0)

Bits	Field	Type/ HW Rst	Description
31:0	gplr_reg0	R 0x0	GPLR Reg0 0x0 = port state low 0x1 = port state high

A.11.2.2 GPIO Pin Level Register1 (GPIO_GPLR_REG1)

Instance Name	Offset
GPIO_GPLR_REG1	0x04

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														gplr_reg1																	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 460: GPIO Pin Level Register1 (GPIO_GPLR_REG1)

Bits	Field	Type/ HW Rst	Description
31:18	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
17:0	gplr_reg1	R 0x0	GPLR Reg1 0x0 = port state low 0x1 = port state high

A.11.2.3 GPIO Pin Direction Register0 (GPIO_GPDR_REG0)

Instance Name	Offset
GPIO_GPDR_REG0	0x0C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	gpdr_reg0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 461: GPIO Pin Direction Register0 (GPIO_GPDR_REG0)

Bits	Field	Type/ HW Rst	Description
31:0	gpdr_reg0	R/W 0x0	GPDR Reg0 0x0 = input port 0x1 = output port

A.11.2.4 GPIO Pin Direction Register1 (GPIO_GPDR_REG1)

Instance Name	Offset
GPIO_GPDR_REG1	0x10

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														gpdr_reg1																	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 462: GPIO Pin Direction Register1 (GPIO_GPDR_REG1)

Bits	Field	Type/ HW Rst	Description
31:18	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
17:0	gpdr_reg1	R/W 0x0	GPDR Reg1 0x0 = input port 0x1 = output port

A.11.2.5 GPIO Pin Output Set Register 0 (GPIO_GPSR_REG0)

Instance Name	Offset
GPIO_GPSR_REG0	0x18

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	gpsr_reg0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 463: GPIO Pin Output Set Register 0 (GPIO_GPSR_REG0)

Bits	Field	Type/ HW Rst	Description
31:0	gpsr_reg0	W 0x0	GPSR Reg0 0x0 = unaffected 0x1 = port set if GPIO is configured as output

A.11.2.6 GPIO Pin Output Set Register 1 (GPIO_GPSR_REG1)

Instance Name	Offset
GPIO_GPSR_REG1	0x1C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														gpsr_reg1																	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 464: GPIO Pin Output Set Register 1 (GPIO_GPSR_REG1)

Bits	Field	Type/ HW Rst	Description
31:18	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
17:0	gpsr_reg1	W 0x0	GPSR Reg1 0x0 = unaffected 0x1 = port set if GPIO is configured as output

A.11.2.7 GPIO Pin Output Clear Register 0 (GPIO_GPCR_REG0)

Instance Name	Offset
GPIO_GPCR_REG0	0x24

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	gpcr_reg0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 465: GPIO Pin Output Clear Register 0 (GPIO_GPCR_REG0)

Bits	Field	Type/ HW Rst	Description
31:0	gpcr_reg0	W 0x0	GPCR Reg0 0x0 = unaffected 0x1 = port clear if GPIO is configured as output

A.11.2.8 GPIO Pin Output Clear Register 1 (GPIO_GPCR_REG1)

Instance Name	Offset
GPIO_GPCR_REG1	0x28

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														gpcr_reg1																	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 466: GPIO Pin Output Clear Register 1 (GPIO_GPCR_REG1)

Bits	Field	Type/ HW Rst	Description
31:18	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
17:0	gpcr_reg1	W 0x0	GPCR Reg1 0x0 = unaffected 0x1 = port clear if GPIO is configured as output

A.11.2.9 GPIO Rising Edge Detect Enable Register 0 (GPIO_GRER_REG0)

Instance Name	Offset
GPIO_GRER_REG0	0x30

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	grer_reg0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 467: GPIO Rising Edge Detect Enable Register 0 (GPIO_GRER_REG0)

Bits	Field	Type/ HW Rst	Description
31:0	grer_reg0	R/W 0x0	GRER Reg0 0x0 = disable rising edge detection 0x1 = set corresponding GEDR status bit when rising edge is detected on GPIO input

A.11.2.10 GPIO Rising Edge Detect Enable Register 1 (GPIO_GRER_REG1)

Instance Name	Offset
GPIO_GRER_REG1	0x34

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														grer_reg1																	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 468: GPIO Rising Edge Detect Enable Register 1 (GPIO_GRER_REG1)

Bits	Field	Type/ HW Rst	Description
31:18	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
17:0	grer_reg1	R/W 0x0	GRER Reg1 0x0 = disable rising edge detection 0x1 = set corresponding GEDR status bit when rising edge is detected on GPIO input

A.11.2.11 GPIO Falling Edge Detect Enable Register 0 (GPIO_GFER_REG0)

Instance Name	Offset
GPIO_GFER_REG0	0x3C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	gfer_reg0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 469: GPIO Falling Edge Detect Enable Register 0 (GPIO_GFER_REG0)

Bits	Field	Type/ HW Rst	Description
31:0	gfer_reg0	R/W 0x0	GFER Reg0 0x0 = disable falling edge detection 0x1 = set corresponding GEDR status bit when falling edge is detected on GPIO input

A.11.2.12 GPIO Falling Edge Detect Enable Register 1 (GPIO_GFER_REG1)

Instance Name	Offset
GPIO_GFER_REG1	0x40

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field															gfer_reg1																	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 470: GPIO Falling Edge Detect Enable Register 1 (GPIO_GFER_REG1)

Bits	Field	Type/ HW Rst	Description
31:18	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
17:0	gfer_reg1	R/W 0x0	GFER Reg1 0x0 = disable falling edge detection 0x1 = set corresponding GEDR status bit when falling edge is detected on GPIO input

A.11.2.13 GPIO Edge Detect Status Register 0 (GPIO_GEDR_REG0)

Instance Name	Offset
GPIO_GEDR_REG0	0x48

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	gedr_reg0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 471: GPIO Edge Detect Status Register 0 (GPIO_GEDR_REG0)

Bits	Field	Type/ HW Rst	Description
31:0	gedr_reg0	R/W1CLR 0x0	GEDR Reg0 0x0 = no edge detected on a port as specified by GRERx or GFERx 0x1 = edge detected on a port as specified by GRERx or GFERx

A.11.2.14 GPIO Edge Detect Status Register 1 (GPIO_GEDR_REG1)

Instance Name	Offset
GPIO_GEDR_REG1	0x4C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														gedr_reg1																	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 472: GPIO Edge Detect Status Register 1 (GPIO_GEDR_REG1)

Bits	Field	Type/ HW Rst	Description
31:18	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
17:0	gedr_reg1	R/W1CLR 0x0	GEDR Reg1 0x0 = no edge detected on a port as specified by GRERx or GFERx 0x1 = edge detected on a port as specified by GRERx or GFERx

A.11.2.15 GPIO Pin Bitwise Set Direction Register 0 (GPIO_GSDR_REG0)

Instance Name	Offset
GPIO_GSDR_REG0	0x54

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	gsdr_reg0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 473: GPIO Pin Bitwise Set Direction Register 0 (GPIO_GSDR_REG0)

Bits	Field	Type/ HW Rst	Description
31:0	gsdr_reg0	W 0x0	GSDR Reg0 0x0 = GPDR bit unaffected 0x1 = GPDR bit set and GPIO pin is set as output

A.11.2.16 GPIO Pin Bitwise Set Direction Register 1 (GPIO_GSDR_REG1)

Instance Name	Offset
GPIO_GSDR_REG1	0x58

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														gsdr_reg1																	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 474: GPIO Pin Bitwise Set Direction Register 1 (GPIO_GSDR_REG1)

Bits	Field	Type/ HW Rst	Description
31:18	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
17:0	gsdr_reg1	W 0x0	GSDR Reg1 0x0 = GPDR bit unaffected 0x1 = GPDR bit set and GPIO pin is set as output

A.11.2.17 GPIO Pin Bitwise Clear Direction Register 0 (GPIO_GCDR_REG0)

Instance Name	Offset
GPIO_GCDR_REG0	0x60

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	gcdr_reg0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 475: GPIO Pin Bitwise Clear Direction Register 0 (GPIO_GCDR_REG0)

Bits	Field	Type/ HW Rst	Description
31:0	gcdr_reg0	W 0x0	GCDR Reg0 0x0 = GPDR bit unaffected 0x1 = GPDR bit clear and GPIO pin is set as input

A.11.2.18 GPIO Pin Bitwise Clear Direction Register 1 (GPIO_GCDR_REG1)

Instance Name	Offset
GPIO_GCDR_REG1	0x64

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														gcdr_reg1																	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 476: GPIO Pin Bitwise Clear Direction Register 1 (GPIO_GCDR_REG1)

Bits	Field	Type/ HW Rst	Description
31:18	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
17:0	gcdr_reg1	W 0x0	GCDR Reg1 0x0 = GPDR bit unaffected 0x1 = GPDR bit clear and GPIO pin is set as input

A.11.2.19 GPIO Bitwise Set Rising Edge Detect Enable Register 0 (GPIO_GSRER_REG0)

Instance Name	Offset
GPIO_GSRER_REG0	0x6C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	gsrer_reg0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 477: GPIO Bitwise Set Rising Edge Detect Enable Register 0 (GPIO_GSRER_REG0)

Bits	Field	Type/ HW Rst	Description
31:0	gsrer_reg0	W 0x0	GSRER Reg0 0x0 = GRER bit unaffected 0x1 = GRER bit set

A.11.2.20 GPIO Bitwise Set Rising Edge Detect Enable Register 1 (GPIO_GSRER_REG1)

Instance Name	Offset
GPIO_GSRER_REG1	0x70

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														gsrer_reg1																	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 478: GPIO Bitwise Set Rising Edge Detect Enable Register 1 (GPIO_GSRER_REG1)

Bits	Field	Type/ HW Rst	Description
31:18	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
17:0	gsrer_reg1	W 0x0	GSRER Reg1 0x0 = GRER bit unaffected 0x1 = GRER bit set

A.11.2.21 GPIO Bitwise Clear Rising Edge Detect Enable Register 0 (GPIO_GCRER_REG0)

Instance Name	Offset
GPIO_GCRER_REG0	0x78

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	gcrer_reg0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 479: GPIO Bitwise Clear Rising Edge Detect Enable Register 0 (GPIO_GCRER_REG0)

Bits	Field	Type/ HW Rst	Description
31:0	gcrer_reg0	W 0x0	GCRER Reg0 0x0 = GRER bit unaffected 0x1 = GRER bit clear

A.11.2.22 GPIO Bitwise Clear Rising Edge Detect Enable Register 1 (GPIO_GCRER_REG1)

Instance Name	Offset
GPIO_GCRER_REG1	0x7C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														gcrer_reg1																	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 480: GPIO Bitwise Clear Rising Edge Detect Enable Register 1 (GPIO_GCRER_REG1)

Bits	Field	Type/ HW Rst	Description
31:18	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
17:0	gcrer_reg1	W 0x0	GCRER Reg1 0x0 = GRER bit unaffected 0x1 = GRER bit clear

A.11.2.23 GPIO Bitwise Set Falling Edge Detect Enable Register 0 (GPIO_GSFER_REG0)

Instance Name	Offset
GPIO_GSFER_REG0	0x84

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	gsfer_reg0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 481: GPIO Bitwise Set Falling Edge Detect Enable Register 0 (GPIO_GSFER_REG0)

Bits	Field	Type/ HW Rst	Description
31:0	gsfer_reg0	W 0x0	GSFER Reg0 0x0 = GFER bit unaffected 0x1 = GFER bit set

A.11.2.24 GPIO Bitwise Set Falling Edge Detect Enable Register 1 (GPIO_GSFER_REG1)

Instance Name	Offset
GPIO_GSFER_REG1	0x88

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														gsfer_reg1																	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 482: GPIO Bitwise Set Falling Edge Detect Enable Register 1 (GPIO_GSFER_REG1)

Bits	Field	Type/ HW Rst	Description
31:18	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
17:0	gsfer_reg1	W 0x0	GSFER Reg1 0x0 = GFER bit unaffected 0x1 = GFER bit set

A.11.2.25 GPIO Bitwise Clear Falling Edge Detect Enable Register 0 (GPIO_GCFER_REG0)

Instance Name	Offset
GPIO_GCFER_REG0	0x90

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	gcfer_reg0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 483: GPIO Bitwise Clear Falling Edge Detect Enable Register 0 (GPIO_GCFER_REG0)

Bits	Field	Type/ HW Rst	Description
31:0	gcfer_reg0	W 0x0	GCFER Reg0 0x0 = GFER bit unaffected 0x1 = GFER bit clear

A.11.2.26 GPIO Bitwise Clear Falling Edge Detect Enable Register 1 (GPIO_GCFER_REG1)

Instance Name	Offset
GPIO_GCFER_REG1	0x94

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														gcfer_reg1																	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 484: GPIO Bitwise Clear Falling Edge Detect Enable Register 1 (GPIO_GCFER_REG1)

Bits	Field	Type/ HW Rst	Description
31:18	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
17:0	gcfer_reg1	W 0x0	GCFER Reg1 0x0 = GFER bit unaffected 0x1 = GFER bit clear

A.11.2.27 GPIO Bitwise Mask of Edge Detect Status Register 0 (APMASK_REG0)

Instance Name	Offset
APMASK_REG0	0x9C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	apmask_reg0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 485: GPIO Bitwise Mask of Edge Detect Status Register 0 (APMASK_REG0)

Bits	Field	Type/ HW Rst	Description
31:0	apmask_reg0	R/W 0x0	APMASK Reg0 0x0 = GPIO edge detects are masked 0x1 = GPIO edge detects are not masked

A.11.2.28 GPIO Bitwise Mask of Edge Detect Status Register 1 (APMASK_REG1)

Instance Name	Offset
APMASK_REG1	0xA0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														apmask_reg1																	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 486: GPIO Bitwise Mask of Edge Detect Status Register 1 (APMASK_REG1)

Bits	Field	Type/ HW Rst	Description
31:18	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
17:0	apmask_reg1	R/W 0x0	APMASK Reg1 0x0 = GPIO edge detects are masked 0x1 = GPIO edge detects are not masked

A.12 GPT Address Block

A.12.1 GPT Register Map

Table 487: GPT Register Map

Offset	Name	HW Rst	Description	Details
0x000	CNT_EN_REG	0x0000_0000	Counter Enable Register	Page: 662
0x020	STS_REG	0x0000_0000	Status Register	Page: 664
0x024	INT_REG	0x0000_0000	Interrupt Register	Page: 666
0x028	INT_MSK_REG	0x0301_3F3F	Interrupt Mask Register	Page: 668
0x040	CNT_CNTL_REG	0x0000_0000	Counter Control Register	Page: 670
0x050	CNT_VAL_REG	0x0000_0000	Counter Value Register	Page: 671
0x060	CNT_UPP_VAL_REG	0xFFFF_FFFF	Counter Upper Value Register	Page: 671
0x080	CLK_CNTL_REG	0x0000_0000	Clock Control Register	Page: 672
0x088	IC_CNTL_REG	0x0000_0000	Input Capture Control Register	Page: 673
0x0A0	DMA_CNTL_EN_REG	0x0000_0000	DMA Control Enable Register	Page: 674
0x0A4	DMA_CNTL_CH_REG	0x0000_0000	DMA Control Channel Register	Page: 675
0x0D0	ADCT_REG	0x0000_0000	ADC Trigger Control Register	Page: 676
0x0D8	ADCT_DLY_REG	0x0000_0000	ADC Trigger Delay Register	Page: 677
0x0F0	USER_REQ_REG	0x0000_0000	User Request Register	Page: 678
0x200	CH0_CNTL_REG	0x0000_0000	Channel 0 Control Register	Page: 680
0x210	CH0_CMR0_REG	0x0000_0000	Channel 0 Counter Match Register 0	Page: 681
0x214	CH0_STS_REG	0x0000_0000	Channel 0 Status Register 0	Page: 682
0x220	CH0_CMR1_REG	0x0000_0000	Channel 0 Counter Match Register 1	Page: 683
0x240	CH1_CNTL_REG	0x0000_0000	Channel 1 Control Register	Page: 680
0x250	CH1_CMR0_REG	0x0000_0000	Channel 1 Counter Match Register 0	Page: 681
0x254	CH1_STS_REG	0x0000_0000	Channel 1 Status Register 0	Page: 682
0x260	CH1_CMR1_REG	0x0000_0000	Channel 1 Counter Match Register 1	Page: 683
0x280	CH2_CNTL_REG	0x0000_0000	Channel 2 Control Register	Page: 680
0x290	CH2_CMR0_REG	0x0000_0000	Channel 2 Counter Match Register 0	Page: 681
0x294	CH2_STS_REG	0x0000_0000	Channel 2 Status Register 0	Page: 682
0x2A0	CH2_CMR1_REG	0x0000_0000	Channel 2 Counter Match Register 1	Page: 683
0x2C0	CH3_CNTL_REG	0x0000_0000	Channel 3 Control Register	Page: 680

Table 487: GPT Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x2D0	CH3_CMR0_REG	0x0000_0000	Channel 3 Counter Match Register 0	Page: 681
0x2D4	CH3_STS_REG	0x0000_0000	Channel 3 Status Register 0	Page: 682
0x2E0	CH3_CMR1_REG	0x0000_0000	Channel 3 Counter Match Register 1	Page: 683
0x300	CH4_CNTL_REG	0x0000_0000	Channel 4 Control Register	Page: 680
0x310	CH4_CMR0_REG	0x0000_0000	Channel 4 Counter Match Register 0	Page: 681
0x314	CH4_STS_REG	0x0000_0000	Channel 4 Status Register 0	Page: 682
0x320	CH4_CMR1_REG	0x0000_0000	Channel 4 Counter Match Register 1	Page: 683
0x340	CH5_CNTL_REG	0x0000_0000	Channel 5 Control Register	Page: 680
0x350	CH5_CMR0_REG	0x0000_0000	Channel 5 Counter Match Register 0	Page: 681
0x354	CH5_STS_REG	0x0000_0000	Channel 5 Status Register 0	Page: 682
0x360	CH5_CMR1_REG	0x0000_0000	Channel 5 Counter Match Register 1	Page: 683

A.12.2 GPT Registers

A.12.2.1 Counter Enable Register (CNT_EN_REG)

Instance Name	Offset
CNT_EN_REG	0x000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved													sts_resetn	cnt_rst_done	cnt_run	Reserved													cnt_reset	cnt_stop	cnt_start
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0

Table 488: Counter Enable Register (CNT_EN_REG)

Bits	Field	Type/ HW Rst	Description
31:19	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
18	sts_resetn	R 0x0	System Reset Status CPU must poll this bit for a 1 before accessing any other registers. 0x0 = indicates that the system reset is still asserted 0x1 = indicates that the system reset is deasserted

Table 488: Counter Enable Register (CNT_EN_REG) (Continued)

Bits	Field	Type/ HW Rst	Description
17	cnt_rst_done	R 0x0	Counter Reset Done Status Writing 1 to CNT_RESET will set this bit to 0 until the counter finishes resetting. 0x0 = indicates that the counter is still resetting 0x1 = indicates that the counter has been reset
16	cnt_run	R 0x0	Counter Enabled Status This bit can be polled to see when the counter is really enabled. 0x0 = counter is disabled 0x1 = counter is enabled
15:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2	cnt_reset	W 0x0	Counter Reset 0x0 = no action 0x1 = reset the counter (counter is reset to 0; channel output states are reset to 0; poll CNT_RST_DONE for 1 before writing to any other registers)
1	cnt_stop	W 0x0	Counter Stop 0x0 = no action 0x1 = disable the counter (poll CNT_RUN for 0 to confirm that the counter is disabled internally)
0	cnt_start	W 0x0	Counter Start 0x0 = no action 0x1 = enable the counter (poll CNT_RUN for 1 to confirm that the counter is enabled internally)

A.12.2.2 Status Register (STS_REG)

Instance Name	Offset
STS_REG	0x020

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved						dma1_of_sts.	dma0_of_sts.	Reserved							cnt_upp_sts	Reserved		ch5_err_sts	ch4_err_sts	ch3_err_sts	ch2_err_sts	ch1_err_sts	ch0_err_sts	Reserved		ch5_sts	ch4_sts	ch3_sts	ch2_sts	ch1_sts	ch0_sts
HW Rst	?	?	?	?	?	?	0	0	?	?	?	?	?	?	?	0	?	?	0	0	0	0	0	0	?	?	0	0	0	0	0	0

Table 489: Status Register (STS_REG)

Bits	Field	Type/ HW Rst	Description
31:26	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
25	dma1_of_sts.	R/W1CLR 0x0	See DMA0_OF_STS
24	dma0_of_sts.	R/W1CLR 0x0	DMA Overflow Status 0x0 = status cleared 0x1 = indicates that there has been a new input capture before this DMA channel could transfer the captured data away
23:17	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
16	cnt_upp_sts	R/W1CLR 0x0	Counter-Reach-Upper Status Indicates that the counter has reached UPP_VAL when incrementing.
15:14	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
13	ch5_err_sts	R/W1CLR 0x0	Channel 5 Error Status 0x0 = status cleared 0x1 = an error has occurred in this channel
12	ch4_err_sts	R/W1CLR 0x0	Channel 4 Error Status 0x0 = status cleared 0x1 = an error has occurred in this channel
11	ch3_err_sts	R/W1CLR 0x0	Channel 3 Error Status 0x0 = status cleared 0x1 = an error has occurred in this channel
10	ch2_err_sts	R/W1CLR 0x0	Channel 2 Error Status 0x0 = status cleared 0x1 = an error has occurred in this channel

Table 489: Status Register (STS_REG) (Continued)

Bits	Field	Type/ HW Rst	Description
9	ch1_err_sts	R/W1CLR 0x0	Channel 1 Error Status 0x0 = status cleared 0x1 = an error has occurred in this channel
8	ch0_err_sts	R/W1CLR 0x0	Channel 0 Error Status 0x0 = status cleared 0x1 = an error has occurred in this channel
7:6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
5	ch5_sts	R/W1CLR 0x0	Channel 5 Status 0x0 = status cleared 0x1 = status bit for this channel has been set
4	ch4_sts	R/W1CLR 0x0	Channel 4 Status 0x0 = status cleared 0x1 = status bit for this channel has been set
3	ch3_sts	R/W1CLR 0x0	Channel 3 Status 0x0 = status cleared 0x1 = status bit for this channel has been set
2	ch2_sts	R/W1CLR 0x0	Channel 2 Status 0x0 = status cleared 0x1 = status bit for this channel has been set
1	ch1_sts	R/W1CLR 0x0	Channel 1 Status 0x0 = status cleared 0x1 = status bit for this channel has been set
0	ch0_sts	R/W1CLR 0x0	Channel 0 Status 0x0 = status cleared 0x1 = status bit for this channel has been set

A.12.2.3 Interrupt Register (INT_REG)

INT_MSK_REG is combined with STS_REG to form this register. Masked bits will be 0 while unmasked bits will be the same value as the ones in STS_REG.

If any bits in this register is 1, an interrupt will be generated.

Instance Name	Offset
INT_REG	0x024

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved						dma1_of_intr	dma0_of_intr	Reserved							cnt_upp_intr	Reserved		ch5_err_intr	ch4_err_intr	ch3_err_intr	ch2_err_intr	ch1_err_intr	ch0_err_intr	Reserved		ch5_intr	ch4_intr	ch3_intr	ch2_intr	ch1_intr	ch0_intr
HW Rst	?	?	?	?	?	?	0	0	?	?	?	?	?	?	?	0	?	?	0	0	0	0	0	0	?	?	0	0	0	0	0	0

Table 490: Interrupt Register (INT_REG)

Bits	Field	Type/ HW Rst	Description
31:26	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
25	dma1_of_intr	R 0x0	Masked Signal of DMA1_OF_STS
24	dma0_of_intr	R 0x0	Masked Signal of DMA0_OF_STS
23:17	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
16	cnt_upp_intr	R 0x0	Masked Signal of CNT_UPP_STS
15:14	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
13	ch5_err_intr	R 0x0	Masked Signal of CH5_ERR_STS
12	ch4_err_intr	R 0x0	Masked Signal of CH4_ERR_STS
11	ch3_err_intr	R 0x0	Masked Signal of CH3_ERR_STS
10	ch2_err_intr	R 0x0	Masked Signal of CH2_ERR_STS
9	ch1_err_intr	R 0x0	Masked Signal of CH1_ERR_STS
8	ch0_err_intr	R 0x0	Masked Signal of CH0_ERR_STS
7:6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

Table 490: Interrupt Register (INT_REG) (Continued)

Bits	Field	Type/ HW Rst	Description
5	ch5_intr	R 0x0	Masked Signal of CH5_STS
4	ch4_intr	R 0x0	Masked Signal of CH4_STS
3	ch3_intr	R 0x0	Masked Signal of CH3_STS
2	ch2_intr	R 0x0	Masked Signal of CH2_STS
1	ch1_intr	R 0x0	Masked Signal of CH1_STS
0	ch0_intr	R 0x0	Masked Signal of CH0_STS

A.12.2.4 Interrupt Mask Register (INT_MSK_REG)

INT_MSK_REG is combined with STS_REG to form INT_REG. Masked bits will be 0 while unmasked bits will be the same value as the ones in STS_REG.

Instance Name	Offset
INT_MSK_REG	0x028

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved						dma1_of_msk	dma0_of_msk	Reserved							cnt_upp_msk	Reserved		ch5_err_msk	ch4_err_msk	ch3_err_msk	ch2_err_msk	ch1_err_msk	ch0_err_msk	Reserved		ch5_msk	ch4_msk	ch3_msk	ch2_msk	ch1_msk	ch0_msk
HW Rst	?	?	?	?	?	?	1	1	?	?	?	?	?	?	?	1	?	?	1	1	1	1	1	1	?	?	1	1	1	1	1	1

Table 491: Interrupt Mask Register (INT_MSK_REG)

Bits	Field	Type/ HW Rst	Description
31:26	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
25	dma1_of_msk	R/W 0x1	See DMA0_OF_MSK
24	dma0_of_msk	R/W 0x1	DMA Channel Overflow Mask 0x0 = do not mask DMA0_OF_STS 0x1 = mask DMA0_OF_STS
23:17	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
16	cnt_upp_msk	R/W 0x1	Upper Value Interrupt Mask 0x0 = do not mask CNT_UPP_STS 0x1 = mask CNT_UPP_STS
15:14	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
13	ch5_err_msk	R/W 0x1	Channel Error Interrupt Mask 0x0 = do not mask CH5_ERR_STS 0x1 = mask CH5_ERR_STS
12	ch4_err_msk	R/W 0x1	Channel Error Interrupt Mask 0x0 = do not mask CH4_ERR_STS 0x1 = mask CH4_ERR_STS
11	ch3_err_msk	R/W 0x1	Channel Error Interrupt Mask 0x0 = do not mask CH3_ERR_STS 0x1 = mask CH3_ERR_STS
10	ch2_err_msk	R/W 0x1	Channel Error Interrupt Mask 0x0 = do not mask CH2_ERR_STS 0x1 = mask CH2_ERR_STS

Table 491: Interrupt Mask Register (INT_MSK_REG) (Continued)

Bits	Field	Type/ HW Rst	Description
9	ch1_err_msk	R/W 0x1	Channel Error Interrupt Mask 0x0 = do not mask CH1_ERR_STS 0x1 = mask CH1_ERR_STS
8	ch0_err_msk	R/W 0x1	Channel Error Interrupt Mask 0x0 = do not mask CH0_ERR_STS 0x1 = mask CH0_ERR_STS
7:6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
5	ch5_msk	R/W 0x1	Channel Interrupt Mask 0x0 = do not mask CH5_STS 0x1 = mask CH5_STS
4	ch4_msk	R/W 0x1	Channel Interrupt Mask 0x0 = do not mask CH4_STS 0x1 = mask CH4_STS
3	ch3_msk	R/W 0x1	Channel Interrupt Mask 0x0 = do not mask CH3_STS 0x1 = mask CH3_STS
2	ch2_msk	R/W 0x1	Channel Interrupt Mask 0x0 = do not mask CH2_STS 0x1 = mask CH2_STS
1	ch1_msk	R/W 0x1	Channel Interrupt Mask 0x0 = do not mask CH1_STS 0x1 = mask CH1_STS
0	ch0_msk	R/W 0x1	Channel Interrupt Mask 0x0 = do not mask CH0_STS 0x1 = mask CH0_STS

A.12.2.5 Counter Control Register (CNT_CNTL_REG)

Instance Name	Offset
CNT_CNTL_REG	0x040

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

Table 492: Counter Control Register (CNT_CNTL_REG)

Bits	Field	Type/ HW Rst	Description
31:10	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
9:8	cnt_updt_mod	R/W 0x0	Counter Value Update Mode 0x0 = Auto-update normal (can be used for any clock relationship between the counter clock and the APB clock; only every 3-4 counter ticks are updated to CNT_VAL) 0x1 = Auto-update fast (use when counter clock is at least 5 times slower than the APB clock; every counter tick is updated to CNT_VAL) 0x2 = reserved 0x3 = update off (of CNT_VAL does not need to be read, CNT_UPDT_MOD can be set to off to save power)
7:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4	cnt_dbg_act	R/W 0x0	Counter Debug Mode Action Mask 0x0 = in debug mode, stop the counter 0x1 = in debug mode, the counter is not affected
3:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.12.2.6 Counter Value Register (CNT_VAL_REG)

Instance Name	Offset
CNT_VAL_REG	0x050

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

Table 493: Counter Value Register (CNT_VAL_REG)

Bits	Field	Type/ HW Rst	Description
31:0	cnt_val	R 0x0	Counter Value This register is used to view the current value of the main counter. The update of this register is based on CNT_UPDT_MOD.

A.12.2.7 Counter Upper Value Register (CNT_UPP_VAL_REG)

Instance Name	Offset
CNT_UPP_VAL_REG	0x060

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

Table 494: Counter Upper Value Register (CNT_UPP_VAL_REG)

Bits	Field	Type/ HW Rst	Description
31:0	upp_val	R/W 0xFFFF_FFFF	Counter Upper Value Do not set to 0. The reset value is the maximum value of the counter, where all bits are 1. In the event that the counter reaches this value (counter-reach-upper), the counter will overflow to 0. Setting this value to all 1s is equivalent to a free running up-counter. Writing to this register will shadow it and start the internal shadow register update. The update finishes during the next counter-reach-upper event and will continue to update if it detects a new UPP_VAL. To guarantee an immediate update with the current UPP_VAL, write 1 to CNT_RESET.

A.12.2.8 Clock Control Register (CLK_CNTL_REG)

Instance Name	Offset
CLK_CNTL_REG	0x080

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved								clk_pre								Reserved				clk_div				Reserved							clk_src
HW Rst	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	?	?	?	?	0	0	0	0	?	?	?	?	?	?	?	0

Table 495: Clock Control Register (CLK_CNTL_REG)

Bits	Field	Type/ HW Rst	Description
31:24	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
23:16	clk_pre	R/W 0x0	Clock Pre-Scalar This can be used together with CLK_DIV. The frequency of the prescaled clock (f_pre) is calculated from the frequency of the counter clock (f_clk) using this formula: f_pre = f_clk / (CLK_PRE + 1)
15:12	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
11:8	clk_div	R/W 0x0	Clock Divider This can be used together with CLK_PRE. The frequency of the divided clock (f_div) is calculated from the frequency of the counter clock (f_clk) using this formula: f_div = f_clk / (2 ^ CLK_DIV)
7:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	clk_src	R/W 0x0	Counter Clock Select 0x0 = select clock 0 0x1 = select clock 1

A.12.2.9 Input Capture Control Register (IC_CNTL_REG)

Instance Name	Offset
IC_CNTL_REG	0x088

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

Table 496: Input Capture Control Register (IC_CNTL_REG)

Bits	Field	Type/ HW Rst	Description
31:7	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
6:4	chx_ic_div	R/W 0x0	Input Capture Sampling Clock Divider This field divides the sampling clock used to sample the input trigger. The frequency of the divided sampling clock(f_sdiv) is calculated from the frequency of the sampling clock (f_sclk) using the following formula: f_sdiv = f_sclk / (2 ^ CHx_IC_DIV)
3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2:0	chx_ic_width	R/W 0x0	Input Capture Filter Width The input trigger must be sampled for this many consecutive cycles before it is considered a valid edge. Any glitch pulse shorter than this many cycles will be filtered. 0x0 = no filtering (1 cycle) 0x1 = 2 cycles 0x2 = 3 cycles 0x3 = 4 cycles 0x4 = 5 cycles 0x5 = 6 cycles 0x6 = 7 cycles 0x7 = reserved

A.12.2.10 DMA Control Enable Register (DMA_CNTL_EN_REG)

Instance Name	Offset
DMA_CNTL_EN_REG	0x0A0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																														dma1_en	dma0_en
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0

Table 497: DMA Control Enable Register (DMA_CNTL_EN_REG)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	dma1_en	R/W 0x0	See DMA0_EN
0	dma0_en	R/W 0x0	DMA Channel Enable 0x0 = disable this DMA channel 0x1 = enable this DMA channel. in input capture mode, DMA controller will be notified when a value is captured.

A.12.2.11 DMA Control Channel Register (DMA_CNTL_CH_REG)

Instance Name	Offset
DMA_CNTL_CH_REG	0x0A4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																									dma1_ch			Reserved	dma0_ch		
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	?	0	0	0

Table 498: DMA Control Channel Register (DMA_CNTL_CH_REG)

Bits	Field	Type/ HW Rst	Description
31:7	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
6:4	dma1_ch	R/W 0x0	See DMA0_CH
3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2:0	dma0_ch	R/W 0x0	DMA Channel Select Select the counter channel to which this DMA channel is connected. 0x0 = connect to channel 0 0x1 = connect to channel 1 0x2 = connect to channel 2 0x3 = connect to channel 3 0x4 = connect to channel 4 0x5 = connect to channel 5 others = reserved

A.12.2.12 ADC Trigger Control Register (ADCT_REG)

Instance Name	Offset
ADCT_REG	0x0D0

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

Table 499: ADC Trigger Control Register (ADCT_REG)

Bits	Field	Type/ HW Rst	Description
31:9	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
8	adct_en	R/W 0x0	ADC Trigger Enable 0x0 = disable the ADC trigger 0x1 = enable the ADC trigger
7:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2:0	adct_chsel	R/W 0x0	ADC Trigger Channel Select Select which counter channel to multiplex to the ADC trigger. 0x0 = connect to channel 0 0x1 = connect to channel 1 0x2 = connect to channel 2 0x3 = connect to channel 3 0x4 = connect to channel 4 0x5 = connect to channel 5 others = reserved

A.12.2.13 ADC Trigger Delay Register (ADCT_DLY_REG)

Instance Name	Offset
ADCT_DLY_REG	0x0D8

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

Table 500: ADC Trigger Delay Register (ADCT_DLY_REG)

Bits	Field	Type/ HW Rst	Description
31:0	adct_dly	R/W 0x0	ADC Trigger Delay At the end of each PWM period, after the effective ADC trigger delay (adly_eff), a trigger will be generated to signal the ADC to begin a conversion. adly_eff is 4 times ADCT_DLY, as shown below: adly_eff = ADCT_DLY x 4 For the ADC trigger to work properly, adly_eff must be shorter than the PWM period. The PWM period must be longer than the ADC conversion period, with appropriate margins.

A.12.2.14 User Request Register (USER_REQ_REG)

Instance Name	Offset
USER_REQ_REG	0x0F0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved										ch5_cmr_updt	ch4_cmr_updt	ch3_cmr_updt	ch2_cmr_updt	ch1_cmr_updt	ch0_cmr_updt	Reserved		ch5_rst	ch4_rst	ch3_rst	ch2_rst	ch1_rst	ch0_rst	Reserved		ch5_user_itrig	ch4_user_itrig	ch3_user_itrig	ch2_user_itrig	ch1_user_itrig	ch0_user_itrig
HW Rst	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	?	?	0	0	0	0	0	0	?	?	0	0	0	0	0	0

Table 501: User Request Register (USER_REQ_REG)

Bits	Field	Type/ HW Rst	Description
31:22	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
21	ch5_cmr_updt	W 0x0	See CH0_CMR_UPDT
20	ch4_cmr_updt	W 0x0	See CH0_CMR_UPDT
19	ch3_cmr_updt	W 0x0	See CH0_CMR_UPDT
18	ch2_cmr_updt	W 0x0	See CH0_CMR_UPDT
17	ch1_cmr_updt	W 0x0	See CH0_CMR_UPDT
16	ch0_cmr_updt	W 0x0	Channel CMR Update Write to this field to udpate CMR0 and CMR1 to the internal shadow registers. Writing to this field before the internal update is done will cause the second update to be discarded, and CH0_ERR_STS will be set. In PWM or One-shot mode, the internal update occurs at the end of period or when idle. In any other channel mode, the CMR update completes immediately. In any channel mode, the CMR update completes immediately if there is a counter or channel reset. 0x0 = no action 0x1 = update CMR0 and CMR1 to the internal shadow registers
15:14	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
13	ch5_rst	W 0x0	See CH0_RST
12	ch4_rst	W 0x0	See CH0_RST

Table 501: User Request Register (USER_REQ_REG) (Continued)

Bits	Field	Type/ HW Rst	Description
11	ch3_rst	W 0x0	See CH0_RST
10	ch2_rst	W 0x0	See CH0_RST
9	ch1_rst	W 0x0	See CH0_RST
8	ch0_rst	W 0x0	Channel Reset Writing 1 to this register will reset the channel and kick start the corresponding mode determined by CHx_IO. Only write to this field when CNT_RUN is set. In One-shot pulse mode, the output state will be reset, and a pulse will occur. In One-shot edge mode, an edge transition will occur, but the output state will NOT be reset. In all other modes, the output state will be reset. Resetting multiple channels in the same APB write will synchronize the start periods of the PWM and One-shot outputs. 0x0 = no action 0x1 = reset this channel
7:6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
5	ch5_user_itrig	W 0x0	See CH0_USER_ITRIG
4	ch4_user_itrig	W 0x0	See CH0_USER_ITRIG
3	ch3_user_itrig	W 0x0	See CH0_USER_ITRIG
2	ch2_user_itrig	W 0x0	See CH0_USER_ITRIG
1	ch1_user_itrig	W 0x0	See CH0_USER_ITRIG
0	ch0_user_itrig	W 0x0	User Input Trigger 0x0 = no action 0x1 = if this channel (channel 0) is configured to input-capture mode, generate a manual input trigger to capture the current counter value (this trigger bypasses the input capture control settings)

A.12.2.15 Channel X Control Register (CHx_CNTL_REG)

Instance Name	Offset
CH0_CNTL_REG	0x200
CH1_CNTL_REG	0x240
CH2_CNTL_REG	0x280
CH3_CNTL_REG	0x2C0
CH4_CNTL_REG	0x300
CH5_CNTL_REG	0x340

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved															pol	Reserved	ic_edge			Reserved									chx_io		
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	?	0	0	0	?	?	?	?	?	?	?	?	?	0	0	0

Table 502: Channel X Control Register (CHx_CNTL_REG)

Bits	Field	Type/ HW Rst	Description
31:17	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
16	pol	R/W 0x0	Channel Polarity Default output state of this channel after reset. This field determines the waveform polarity in PWM mode and one-shot mode. 0x0 = reset to 0 0x1 = reset to 1
15	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
14:12	ic_edge	R/W 0x0	Channel Input Capture Edge Determines which edge of the input trigger to capture. 0x0 = capture rising edge in CMR0 0x1 = capture falling edge in CMR0
11:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2:0	chx_io	R/W 0x0	Channel Mode 0x0 = no function 0x1 = input capture mode 0x4 = One-shot mode (pulse) 0x5 = One-shot mode (edge) 0x6 = PWM mode (edge-aligned) 0x7 = PWM mode (center-aligned) others = reserved

A.12.2.16 Channel Counter Match Register 0 (CHx_CMR0_REG)

Instance Name	Offset
CH0_CMR0_REG	0x210
CH1_CMR0_REG	0x250
CH2_CMR0_REG	0x290
CH3_CMR0_REG	0x2D0
CH4_CMR0_REG	0x310
CH5_CMR0_REG	0x350

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	cmr0																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 503: Channel Counter Match Register 0 (CHx_CMR0_REG)

Bits	Field	Type/ HW Rst	Description
31:0	cmr0	R/W 0x0	Channel Counter Match Register 0 In order for the counter to use CMR0 and CMR1, write to CHx_ CMR_UPDT to update CMR0 and CMR1 to the internal shadow registers. In any PWM mode, setting both CMR0 and CMR1 to 0 will result in a degenerate case that shuts off the PWM generator. To reset the PWM, set at least one of CMR0 or CMR1 to a non-zero value and write to CHx_RST.

A.12.2.17 Channel Status Register 0 (CHx_STS_REG)

Instance Name	Offset
CH0_STS_REG	0x214
CH1_STS_REG	0x254
CH2_STS_REG	0x294
CH3_STS_REG	0x2D4
CH4_STS_REG	0x314
CH5_STS_REG	0x354

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

Table 504: Channel Status Register 0 (CHx_STS_REG)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	out_st	R 0x0	Channel Output State Displays the current output state of this channel.

A.12.2.18 Channel Counter Match Register 1 (CHx_CMR1_REG)

Instance Name	Offset
CH0_CMR1_REG	0x220
CH1_CMR1_REG	0x260
CH2_CMR1_REG	0x2A0
CH3_CMR1_REG	0x2E0
CH4_CMR1_REG	0x320
CH5_CMR1_REG	0x360

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	cmr1																															
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 505: Channel Counter Match Register 1 (CHx_CMR1_REG)

Bits	Field	Type/ HW Rst	Description
31:0	cmr1	R/W 0x0	Channel Counter Match Register 1 In order for the counter to use CMR0 and CMR1, write to CHx_ CMR_UPDT to update CMR0 and CMR1 to the internal shadow registers. In any PWM mode, setting both CMR0 and CMR1 to 0 will result in a degenerate case that shuts off the PWM generator. To reset the PWM, set at least one of CMR0 or CMR1 to a non-zero value and write to CHx_RST.

THIS PAGE INTENTIONALLY LEFT BLANK

A.13 RC32 Address Block

A.13.1 RC32 Register Map

Table 506: RC32 Register Map

Offset	Name	HW Rst	Description	Details
0x00	ctrl	0x0000_07F1	Control Register	Page: 685
0x04	status	0x0000_0000	Status Register	Page: 687
0x08	isr	0x0000_0000	Interrupt Status Register	Page: 688
0x0C	imr	0x0000_0003	Interrupt Mask Register	Page: 689
0x10	irsr	0x0000_0000	Interrupt Raw Status Register	Page: 689
0x14	icr	0x0000_0000	Interrupt Clear Register	Page: 690
0x18	clk	0x0000_0004	Clock Register	Page: 690
0x1C	rst	0x0000_0000	Soft Reset Register	Page: 691

A.13.2 RC32 Registers

A.13.2.1 Control Register (ctrl)

Instance Name	Offset
ctrl	0x00

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																		Reserved		code_fr_ext								pd	ext_code_en	cal_en	en
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	1	1	1	1	1	1	1	0	0	0	1

Table 507: Control Register (ctrl)

Bits	Field	Type/ HW Rst	Description
31:14	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
13:12	Reserved	R/W 0x0	Reserved. Do not change the reset value.
11:4	code_fr_ext	R/W 0x7F	External Code Input for Calibration

Table 507: Control Register (ctrl) (Continued)

Bits	Field	Type/ HW Rst	Description
3	pd	R/W 0x0	Clock Power Down 0x0 = power up 0x1 = power down
2	ext_code_en	R/W 0x0	Calibration Code from External Enable 0x0 = calibration code from internal 0x1 = calibration code from external
1	cal_en	R/W 0x0	Calibration Enable 0x0 = disable 0x1 = enable
0	en	R/W 0x1	Calibration Reference Clock Enable 0x0 = disable 0x1 = enable

A.13.2.2 Status Register (status)

Instance Name	Offset
status	0x04

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																						code_fr_cal								cal_done	clk_rdy
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0

Table 508: Status Register (status)

Bits	Field	Type/ HW Rst	Description
31:10	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
9:2	code_fr_cal	R 0x0	Calibration Code
1	cal_done	R 0x0	Calibration Finish Flag 0x0 = calibration not done 0x1 = calibration done
0	clk_rdy	R 0x0	Clock Ready 1 indicates whether HFRC clock is ready.

A.13.2.3 Interrupt Status Register (isr)

Instance Name	Offset
isr	0x08

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

Table 509: Interrupt Status Register (isr)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	ckrdy_int	R 0x0	Clock Ready Interrupt 1 indicates clock out ready interrupt flag if ckrdy_int_raw is not masked.
0	caldon_int	R 0x0	Calibration Done Interrupt 1 indicates calibration done interrupt flag if caldon_int_raw isnot masked.

A.13.2.4 Interrupt Mask Register (imr)

Instance Name	Offset
imr	0x0C

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

Table 510: Interrupt Mask Register (imr)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	ckrdy_int_msk	R/W 0x1	Clock Ready Interrupt Mask Set to 1 to mask off clock ready interrupt.
0	caldon_int_msk	R/W 0x1	Calibration Done Interrupt Mask Set to 1 to mask off calibration done interrupt.

A.13.2.5 Interrupt Raw Status Register (irsr)

Instance Name	Offset
irsr	0x10

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

Table 511: Interrupt Raw Status Register (irsr)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	ckrdy_int_raw	R 0x0	Clock Ready Interrupt Raw 1 indicates clock out ready interrupt flag regardless the mask.
0	caldon_int_raw	R 0x0	Calibration Done Interrupt Raw 1 indicates calibration done interrupt flag regardless the mask.

A.13.2.6 Interrupt Clear Register (icr)

Instance Name	Offset
icr	0x14

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

Table 512: Interrupt Clear Register (icr)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	ckrdy_int_clr	R/W 0x0	Clock Ready Interrupt Clear
0	caldon_int_clr	R/W 0x0	Calibration Done Interrupt Raw

A.13.2.7 Clock Register (clk)

Instance Name	Offset
clk	0x18

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

Table 513: Clock Register (clk)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3	soft_clk_rst	R/W 0x0	Soft Reset for Clock Divider 0x0 = no action 0x1 = Reset

Table 513: Clock Register (clk) (Continued)

Bits	Field	Type/ HW Rst	Description
2	ref_sel	R/W 0x1	Reference Clock Frequency Select 0x0 = half-divided reference clock 0x1 = original reference clock
1:0	Reserved	R/W 0x0	Reserved. Do not change the reset value.

A.13.2.8 Soft Reset Register (rst)

Instance Name	Offset
rst	0x1C

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

Table 514: Soft Reset Register (rst)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	soft_rst	R/W 0x0	Soft Reset for Module, Active High 0x0 = no action 0x1 = reset

THIS PAGE INTENTIONALLY LEFT BLANK

A.14 ADC Address Block

A.14.1 ADC Register Map

Table 515: ADC Register Map

Offset	Name	HW Rst	Description	Details
0x00	adc_reg_cmd	0x0000_0000	ADC Command Register	Page: 694
0x04	adc_reg_general	0x0000_0001	ADC General Register	Page: 695
0x08	adc_reg_config	0x0000_0100	ADC Configuration Register	Page: 696
0x0C	adc_reg_interval	0x0000_000F	ADC Interval Register	Page: 698
0x10	adc_reg_ana	0x0000_A810	ADC ANA Register	Page: 699
0x18	adc_reg_scn1	0x0000_0000	ADC Conversion Sequence 1 Register	Page: 700
0x1C	adc_reg_scn2	0x0000_0000	ADC Conversion Sequence 2 Register	Page: 702
0x20	adc_reg_result_buf	0x0000_0000	ADC Result Buffer Register	Page: 703
0x28	adc_reg_dmar	0x0000_0000	ADC DMAR Register	Page: 704
0x2C	adc_reg_status	0x0000_0000	ADC Status Register	Page: 705
0x30	adc_reg_isr	0x0000_0000	ADC ISR Register	Page: 706
0x34	adc_reg_imr	0x0000_007F	ADC IMR Register	Page: 707
0x38	adc_reg_irsr	0x0000_0000	ADC IRSR Register	Page: 708
0x3C	adc_reg_icr	0x0000_0000	ADC ICR Register	Page: 709
0x44	adc_reg_result	0x0000_0000	ADC Result Register	Page: 710
0x48	adc_reg_raw_result	0x0000_0000	ADC Raw Result Register	Page: 710
0x4C	adc_reg_offset_cal	0x0000_0000	ADC Offset Calibration Register	Page: 711
0x50	adc_reg_gain_cal	0x0000_0000	ADC Gain Calibration Register	Page: 711
0x54	adc_reg_test	0x0000_0000	ADC Test Register	Page: 712
0x58	adc_reg_audio	0x0000_0300	ADC Audio Register	Page: 713
0x5C	adc_reg_voice_det	0x0000_0000	ADC Voice Detect Register	Page: 714
0x60	adc_reg_rsvd	0x0000_FF00	ADC Reserved Register	Page: 715

A.14.2 ADC Registers

A.14.2.1 ADC Command Register (adc_reg_cmd)

Instance Name	Offset
adc_reg_cmd	0x00

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																													soft_clk_rst	soft_rst	conv_start
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0

Table 516: ADC Command Register (adc_reg_cmd)

Bits	Field	Type/ HW Rst	Description
31:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2	soft_clk_rst	R/W 0x0	User Reset Clock
1	soft_rst	R/W 0x0	User Reset the Whole Block
0	conv_start	R/W 0x0	Conversion Control 0x0 = stop conversion 0x1 = start conversion (this will clear the FIFO)

A.14.2.2 ADC General Register (adc_reg_general)

Instance Name	Offset
adc_reg_general	0x04

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																		clk_div_ratio						Reserved		adc_cal_en	clk_ana2m_inv	clk_ana64m_inv	Reserved	global_en	Reserved
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	?	?	0	0	0	1	0	0

Table 517: ADC General Register (adc_reg_general)

Bits	Field	Type/ HW Rst	Description
31:14	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
13:8	clk_div_ratio	R/W 0x0	Analog 64M Clock Division Ratio 0x00 = divide by 1 0x01 = divide by 1 0x02 = divide by 2 ... 0x0F = divide by 15 ... 0x20 = divide by the 32
7:6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
5	adc_cal_en	R/W 0x0	Calibration Enable, Auto Cleared After Calibration Done
4	clk_ana2m_inv	R/W 0x0	Analog Clock 2M Inverted
3	clk_ana64m_inv	R/W 0x0	Analog Clock 64M Inverted
2	Reserved	R/W 0x0	Reserved. Always write 1. Ignore read value.
1	global_en	R/W 0x0	ADC Enable/disable
0	Reserved	R/W 0x1	Reserved. Always write 1. Ignore read value.

A.14.2.3 ADC Configuration Register (adc_reg_config)

Instance Name	Offset
adc_reg_config	0x08

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved											pwr_mode	scan_length				avg_sel			cal_data_sel	cal_data_rst	cal_vref_sel	data_format_sel	cont_conv_en	Reserved	Reserved	Reserved	trigger_en	trigger_sel			
HW Rst	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	1	?	0	0	0	0	0	0	0

Table 518: ADC Configuration Register (adc_reg_config)

Bits	Field	Type/ HW Rst	Description
31:21	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
20	pwr_mode	R/W 0x0	ADC Power Mode Select 0x0 = power mode 0 Analog biasing and reference block are powered up when both global_en and gpadc_conv_start is 1. 0x1 = power mode 1 Analog biasing and reference block are powered up once gpadc_ global_en is 1.
19:16	scan_length	R/W 0x0	Scan Conversion Length (actual Length Is scan_length+1)
15:13	avg_sel	R/W 0x0	Moving Average Length 0x0 = no average 0x1 = average by 2 0x2 = average by 4 0x3 = average by 8 0x4 = average by 16 others = reserved
12	cal_data_sel	R/W 0x0	Select Calibration Data Source 0x0 = use self calibration data 0x1 = user defined calibration data
11	cal_data_rst	R/W 0x0	Reset the Self Calibration Data 0x0 = no reset 0x1 = reset
10	cal_vref_sel	R/W 0x0	Select Input Reference Channel for Gain Calibration 0x0 = select internal vref as input for calibration 0x1 = select external vref as input for calibration

Table 518: ADC Configuration Register (adc_reg_config) (Continued)

Bits	Field	Type/ HW Rst	Description
9	data_format_sel	R/W 0x0	Set Data Format for the Final Data 0x0 = signed differential code in 2's complement 0x1 = unsigned single-end code
8	cont_conv_en	R/W 0x1	To Enable Continuous Conversion 0x0 = 1-shot conversion 0x1 = continuous conversion
7	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
6:5	Reserved	R/W 0x0	Reserved. Always write 0. Ignore read value.
4	trigger_en	R/W 0x0	External Level Trigger Enable Supports gpadc_trigger/gpadc_data_valid handshake. 0x0 = disabled 0x1 = conversion start further controlled by external level signal
3:0	trigger_sel	R/W 0x0	External Trigger Source Select 0x0 = gpadc_trigger[0] . gpt 0 0x1 = gpadc_trigger[1] . acomparator out 0x2 = gpadc_trigger[2] . gpio 40 0x3 = gpadc_trigger[3] . gpio 41 others = reserved

A.14.2.4 ADC Interval Register (adc_reg_interval)

Instance Name	Offset
adc_reg_interval	0x0C

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

Table 519: ADC Interval Register (adc_reg_interval)

Bits	Field	Type/ HW Rst	Description
31:6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
5	bypass_warmup	R/W 0x0	Bypass Warm-Up State Inside ADC 0x0 = ADC warm-up state enabled (warm-up period is controlled by warm-up time) 0x1 = ADC warm-up state bypassed
4:0	warmup_time	R/W 0xF	Warm-Up Time Should be set equal to or higher than 1 us. ADC warm-up time is set to Warmup_time+1 us, for example: 0x00 = ADC warm-up is 1 us ... 0x1F = ADC warm-up is 32 us

A.14.2.5 ADC ANA Register (adc_reg_ana)

Instance Name	Offset
adc_reg_ana	0x10

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved													res_sel		bias_sel	chop_en	inbuf_en	inbuf_chop_en	inbuf_gain		singlediff	Reserved				vref_sel		vref_chop_en	vref_scf_bypass	ts_en	tsext_sel
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	1	0	1	0	1	0	?	?	?	?	0	1	0	0	0	0

Table 520: ADC ANA Register (adc_reg_ana)

Bits	Field	Type/ HW Rst	Description
31:19	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
18:17	res_sel	R/W 0x0	ADC Resolution/data Rate Select 0x0 = 12-bit 2 ms/s pipelined-sar mode 0x1 = 14-bit 181.8 ks/s extended-counting mode 0x2 = 16-bit 57.1 ks/s extended-counting mode 0x3 = 16-bit 16 ks/s extended-counting mode
16	bias_sel	R/W 0x0	ADC Analog Portion Low-Power Mode Select Half the biasing current for modulator when enabled. 0x0 = full biasing current 0x1 = half biasing current
15	chop_en	R/W 0x1	ADC Chopper/Auto-Zero (only in 12-bit mode) Enable 0x0 = disable chopper 0x1 = enable chopper
14	inbuf_en	R/W 0x0	GPADC Input Gain Buffer Enable 0x0 = input gain buffer disabled 0x1 = input gain buffer enabled
13	inbuf_chop_en	R/W 0x1	Input Buffer Chopper Enable 0x0 = disable chopper 0x1 = enable chopper
12:11	inbuf_gain	R/W 0x1	ADC Gain Control Also selects input voltage range. 0x0 = PGA gain is 0.5 (input voltage range is 2*vref) 0x1 = PGA gain is 1 (input voltage range is vref) 0x2 = PGA gain is 2 (input voltage range is 0.5*vref) 0x3 = reserved
10	singlediff	R/W 0x0	Select Single Ended or Differential Input 0x0 = single-ended input 0x1 = differential input

Table 520: ADC ANA Register (adc_reg_ana) (Continued)

Bits	Field	Type/ HW Rst	Description
9:6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
5:4	vref_sel	R/W 0x1	ADC Reference Voltage Select 0x0 = internal 1.8V AVDD18 0x1 = internal 1.2V bandgap 0x2 = external single-ended reference (gpadc_ch[3]) 0x3 = internal 1.2V bandgap with external bypass pin (gpadc_ch[3])
3	vref_chop_en	R/W 0x0	ADC Voltage Reference Buffer Chopper Enable 0x0 = disable chopper 0x1 = enable chopper
2	vref_scf_bypass	R/W 0x0	ADC Voltage Reference Buffer sc-filter Bypass 0x0 = not bypass sc-filter 0x1 = bypass sc-filter
1	ts_en	R/W 0x0	Temperature Sensor Enable Only enable when channel source is temperature sensor. 0x0 = disable 0x1 = enable
0	tsext_sel	R/W 0x0	Temperature Sensor Diode Select 0x0 = internal diode mode 0x1 = external diode mode

A.14.2.6 ADC Conversion Sequence 1 Register (adc_reg_scn1)

Instance Name	Offset
adc_reg_scn1	0x18

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	scan_ch_7				scan_ch_6				scan_ch_5				scan_ch_4				scan_ch_3				scan_ch_2				scan_ch_1				scan_ch_0			
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 521: ADC Conversion Sequence 1 Register (adc_reg_scn1)

Bits	Field	Type/ HW Rst	Description
31:28	scan_ch_7	R/W 0x0	Amux Source 7
27:24	scan_ch_6	R/W 0x0	Amux Source 6

Table 521: ADC Conversion Sequence 1 Register (adc_reg_scn1) (Continued)

Bits	Field	Type/ HW Rst	Description
23:20	scan_ch_5	R/W 0x0	Amux Source 5
19:16	scan_ch_4	R/W 0x0	Amux Source 4
15:12	scan_ch_3	R/W 0x0	Amux Source 3
11:8	scan_ch_2	R/W 0x0	Amux Source 2
7:4	scan_ch_1	R/W 0x0	Amux Source 1
3:0	scan_ch_0	R/W 0x0	Amux Source 0 When singlediff=0 (positive and negative): 0x0 = gpadc_ch[0] and vssa 0x1 = gpadc_ch[1] and vssa 0x2 = gpadc_ch[2] and vssa 0x3 = gpadc_ch[3] and vssa 0x4 = gpadc_ch[4] and vssa 0x5 = gpadc_ch[5] and vssa 0x6 = gpadc_ch[6] and vssa 0x7 = gpadc_ch[7] and vssa 0x8 = vbat_s and vssa 0x9 = vref(1.2V) and vssa 0xA = daca and vssa 0xB = dacb and vssa 0xC = vssa and vssa 0xF = temp_p and vssa others = reserved When singlediff=1 (positive and negative): 0x0 = gpadc_ch[0] and gpadc_ch[1] 0x1 = gpadc_ch[2] and gpadc_ch[3] 0x2 = gpadc_ch[4] and gpadc_ch[5] 0x3 = gpadc_ch[6] and gpadc_ch[7] 0x4 = daca and dacb 0x5 = PGA positive (gpadc_ch[0]) and PGA negative(gpadc_ch[1]) 0xF = temp_p and temp_n others = reserved

A.14.2.7 ADC Conversion Sequence 2 Register (adc_reg_scn2)

Instance Name	Offset
adc_reg_scn2	0x1C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	scan_ch_15				scan_ch_14				scan_ch_13				scan_ch_12				scan_ch_11				scan_ch_10				scan_ch_9				scan_ch_8			
HW Rst	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 522: ADC Conversion Sequence 2 Register (adc_reg_scn2)

Bits	Field	Type/ HW Rst	Description
31:28	scan_ch_15	R/W 0x0	Amux Source 15
27:24	scan_ch_14	R/W 0x0	Amux Source 14
23:20	scan_ch_13	R/W 0x0	Amux Source 13
19:16	scan_ch_12	R/W 0x0	Amux Source 12
15:12	scan_ch_11	R/W 0x0	Amux Source 11
11:8	scan_ch_10	R/W 0x0	Amux Source 10
7:4	scan_ch_9	R/W 0x0	Amux Source 9
3:0	scan_ch_8	R/W 0x0	Amux Source 8

A.14.2.8 ADC Result Buffer Register (adc_reg_result_buf)

Instance Name	Offset
adc_reg_result_buf	0x20

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

Table 523: ADC Result Buffer Register (adc_reg_result_buf)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	width_sel	R/W 0x0	ADC Final Result FIFO Data Packed Format Select Must set scan_length as even when choosing 32 bits. 0x0 = 16-bits and adc_reg_result FIFO is lower 16-bits effective 0x1 = 32-bits and adc_reg_result FIFO is 32-bits effective

A.14.2.9 ADC DMAR Register (adc_reg_dmar)

Instance Name	Offset
adc_reg_dmar	0x28

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

Table 524: ADC DMAR Register (adc_reg_dmar)

Bits	Field	Type/ HW Rst	Description
31:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2:1	fifo_thl	R/W 0x0	FIFO Threshold 0x0 = 1 data 0x1 = 4 data 0x2 = 8 data 0x3 = 16 data
0	dma_en	R/W 0x0	DMA Enable 0x0 = disable DMA handshake (this will also clear remaining DMA request to system DMAC) 0x1 = enable DMA handshake (must enable after conv_start is asserted to ensure FIFO is cleared)

A.14.2.10 ADC Status Register (adc_reg_status)

Instance Name	Offset
adc_reg_status	0x2C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																							fifo_data_count						fifo_full	fifo_ne	act
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0

Table 525: ADC Status Register (adc_reg_status)

Bits	Field	Type/ HW Rst	Description
31:9	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
8:3	fifo_data_count	R 0x0	FIFO Data Number
2	fifo_full	R 0x0	FIFO Full Status
1	fifo_ne	R 0x0	FIFO Not Empty Status
0	act	R 0x0	ADC Status 0x0 = ADC conversion inactive status 0x1 = ADC conversion active status

A.14.2.11 ADC ISR Register (adc_reg_isr)

Instance Name	Offset
adc_reg_isr	0x30

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																									fifo_underrun	fifo_overrun	datasat_pos	datasat_neg	offsat	gainsat	rdy
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0

Table 526: ADC ISR Register (adc_reg_isr)

Bits	Field	Type/ HW Rst	Description
31:7	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
6	fifo_underrun	R 0x0	FIFO Underrun Interrupt Flag
5	fifo_overrun	R 0x0	FIFO Overrun Interrupt Flag
4	datasat_pos	R 0x0	ADC Data Positive Side Saturation Interrupt Flag
3	datasat_neg	R 0x0	ADC Data Negative Side Saturation Interrupt Flag
2	offsat	R 0x0	Offset Correction Saturation Interrupt Flag
1	gainsat	R 0x0	Gain Correction Saturation Interrupt Flag
0	rdy	R 0x0	Conversion Data Ready Interrupt Flag

A.14.2.12 ADC IMR Register (adc_reg_imr)

Instance Name	Offset
adc_reg_imr	0x34

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																									fifo_underrun_mask	fifo_overrun_mask	datasat_pos_mask	datasat_neg_mask	offsat_mask	gainsat_mask	rdy_mask
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	1	1	1	1	1	1

Table 527: ADC IMR Register (adc_reg_imr)

Bits	Field	Type/ HW Rst	Description
31:7	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
6	fifo_underrun_mask	R/W 0x1	Write 1 Mask
5	fifo_overrun_mask	R/W 0x1	Write 1 Mask
4	datasat_pos_mask	R/W 0x1	Write 1 Mask
3	datasat_neg_mask	R/W 0x1	Write 1 Mask
2	offsat_mask	R/W 0x1	Write 1 Mask
1	gainsat_mask	R/W 0x1	Write 1 Mask
0	rdy_mask	R/W 0x1	Write 1 Mask

A.14.2.13 ADC IRSR Register (adc_reg_irsr)

Instance Name	Offset
adc_reg_irsr	0x38

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																									fifo_underrun_raw	fifo_overrun_raw	datasat_pos_raw	datasat_neg_raw	offsat_raw	gainsat_raw	rdy_raw
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0

Table 528: ADC IRSR Register (adc_reg_irsr)

Bits	Field	Type/ HW Rst	Description
31:7	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
6	fifo_underrun_raw	R 0x0	FIFO Underrun Raw The corresponding flag will be captured into this register regardless the interrupt mask. Will be cleared only when int_clr is asserted.
5	fifo_overrun_raw	R 0x0	FIFO Underrun Raw The corresponding flag will be captured into this register regardless the interrupt mask. Will be cleared only when int_clr is asserted.
4	datasat_pos_raw	R 0x0	Datasat Positive Raw The corresponding flag will be captured into this register regardless the interrupt mask. Will be cleared only when int_clr is asserted.
3	datasat_neg_raw	R 0x0	Datasat Negative Raw The corresponding flag will be captured into this register regardless the interrupt mask. Will be cleared only when int_clr is asserted.
2	offsat_raw	R 0x0	Offsat Raw The corresponding flag will be captured into this register regardless the interrupt mask. Will be cleared only when int_clr is asserted.
1	gainsat_raw	R 0x0	Gainsat Raw The corresponding flag will be captured into this register regardless the interrupt mask. Will be cleared only when int_clr is asserted.
0	rdy_raw	R 0x0	Ready Raw The corresponding flag will be captured into this register regardless the interrupt mask. Will be cleared only when int_clr is asserted.

A.14.2.14 ADC ICR Register (adc_reg_icr)

Instance Name	Offset
adc_reg_icr	0x3C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																									fifo_underrun_clr	fifo_overrun_clr	datasat_pos_clr	datasat_neg_clr	offsat_clr	gainsat_clr	rdy_clr
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0

Table 529: ADC ICR Register (adc_reg_icr)

Bits	Field	Type/ HW Rst	Description
31:7	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
6	fifo_underrun_clr	R/W 0x0	Write 1 to Clear Both adc_reg_irsr and adc_reg_isr
5	fifo_overrun_clr	R/W 0x0	Write 1 to Clear Both adc_reg_irsr and adc_reg_isr
4	datasat_pos_clr	R/W 0x0	Write 1 to Clear Both adc_reg_irsr and adc_reg_isr
3	datasat_neg_clr	R/W 0x0	Write 1 to Clear Both adc_reg_irsr and adc_reg_isr
2	offsat_clr	R/W 0x0	Write 1 to Clear Both adc_reg_irsr and adc_reg_isr
1	gainsat_clr	R/W 0x0	Write 1 to Clear Both adc_reg_irsr and adc_reg_isr
0	rdy_clr	R/W 0x0	Write 1 to Clear Both adc_reg_irsr and adc_reg_isr

A.14.2.15 ADC Result Register (adc_reg_result)

Instance Name	Offset
adc_reg_result	0x44

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

Table 530: ADC Result Register (adc_reg_result)

Bits	Field	Type/ HW Rst	Description
31:0	data	R 0x0	ADC Finial Conversion Result Data After calibration and signed/unsigned process.

A.14.2.16 ADC Raw Result Register (adc_reg_raw_result)

Instance Name	Offset
adc_reg_raw_result	0x48

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

Table 531: ADC Raw Result Register (adc_reg_raw_result)

Bits	Field	Type/ HW Rst	Description
31:22	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
21:0	raw_data	R 0x0	ADC Raw Data in Signed 22-Bit Format

A.14.2.17 ADC Offset Calibration Register (adc_reg_offset_cal)

Instance Name	Offset
adc_reg_offset_cal	0x4C

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

Table 532: ADC Offset Calibration Register (adc_reg_offset_cal)

Bits	Field	Type/ HW Rst	Description
31:16	offset_cal_usr	R/W 0x0	User Offset Calibration Data (16-bit signed)
15:0	offset_cal	R 0x0	ADC Self Offset Calibration Value (16-bit signed)

A.14.2.18 ADC Gain Calibration Register (adc_reg_gain_cal)

Instance Name	Offset
adc_reg_gain_cal	0x50

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

Table 533: ADC Gain Calibration Register (adc_reg_gain_cal)

Bits	Field	Type/ HW Rst	Description
31:16	gain_cal_usr	R/W 0x0	ADC User Gain Calibration Value (16-bit signed)
15:0	gain_cal	R 0x0	ADC Self Gain Calibration Value (16-bit signed)

A.14.2.19 ADC Test Register (adc_reg_test)

Instance Name	Offset
adc_reg_test	0x54

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

Table 534: ADC Test Register (adc_reg_test)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3:1	test_sel	R/W 0x0	Test Select
0	test_en	R/W 0x0	Analog Test Enable 0x0 = disable analog test (adc_atest=Hi-Z) 0x1 = enable analog test

A.14.2.20 ADC Audio Register (adc_reg_audio)

Instance Name	Offset
adc_reg_audio	0x58

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																						pga_chop_en	pga_cm			pga_gain			Reserved		en
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	1	0	0	0	0	0	?	?	0

Table 535: ADC Audio Register (adc_reg_audio)

Bits	Field	Type/ HW Rst	Description
31:10	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
9	pga_chop_en	R/W 0x1	Audio PGA Chopper Enable 0x0 = disable audio PGA chopper 0x1 = enable audio PGA chopper
8:6	pga_cm	R/W 0x4	Audio PGA Output Common Mode Control 0x0 = common mode is 0.82V 0x1 = common mode is 0.84V 0x2 = common mode is 0.86V 0x3 = common mode is 0.88V 0x4 = common mode is 0.90V 0x5 = common mode is 0.92V 0x6 = common mode is 0.94V 0x7 = common mode is 0.96V
5:3	pga_gain	R/W 0x0	Audio PGA Voltage Gain Select 0x0 = PGA gain is 4 0x1 = PGA gain is 8 0x2 = PGA gain is 16 0x3 = PGA gain is 32
2:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	en	R/W 0x0	Audio Enable 0x0 = disable audio PGA and decimation rate select 0x1 = enable audio PGA and decimation rate select

A.14.2.21 ADC Voice Detect Register (adc_reg_voice_det)

Instance Name	Offset
adc_reg_voice_det	0x5C

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

Table 536: ADC Voice Detect Register (adc_reg_voice_det)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3:1	level_sel	R/W 0x0	Voice Level Selection 0x0 = input voice level is greater than +255LSbB or smaller than -256LSb 0x1 = input voice level is greater than +511LSb or smaller than -512LSb 0x2 = input voice level is greater than +1023LSb or smaller than -1024LSb 0x3 = input voice level is greater than +2047LSb or smaller than -2048LSb others = reserved
0	det_en	R/W 0x0	Voice Level Detection Enable Select 0x0 = disable level detection 0x1 = enable level detection

A.14.2.22 ADC Reserved Register (adc_reg_rsvd)

Instance Name	Offset
adc_reg_rsvd	0x60

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

Table 537: ADC Reserved Register (adc_reg_rsvd)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:0	unused_adc	R/W 0xFF00	Unused ADC Control Bits Do not change the reset value.

THIS PAGE INTENTIONALLY LEFT BLANK

A.15 DAC Address Block

A.15.1 DAC Register Map

Table 538: DAC Register Map

Offset	Name	HW Rst	Description	Details
0x00	ctrl	0x0000_0000	DAC Control Register	Page: 717
0x04	status	0x0000_0000	DAC Status Register	Page: 718
0x08	actrl	0x000C_0038	Channel A Control Register	Page: 719
0x0C	bctrl	0x0000_1838	Channel B Control Register	Page: 721
0x10	adata	0x0000_0000	Channel A Data Register	Page: 722
0x14	bdata	0x0000_0000	Channel B Data Register	Page: 723
0x18	isr	0x0000_0000	Interrupt Status Register	Page: 723
0x1C	imr	0x0000_001F	Interrupt Mask Register	Page: 724
0x20	irsr	0x0000_0000	Interrupt Raw Status Register	Page: 725
0x24	icr	0x0000_0000	Interrupt Clear Register	Page: 726
0x28	clk	0x0000_0000	Clock Register	Page: 727
0x2C	rst	0x0000_0000	Soft Reset Register	Page: 728

A.15.2 DAC Registers

A.15.2.1 DAC Control Register (ctrl)

Instance Name	Offset
ctrl	0x00

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

Table 539: DAC Control Register (ctrl)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

Table 539: DAC Control Register (ctrl) (Continued)

Bits	Field	Type/ HW Rst	Description
0	ref_sel	R/W 0x0	Reference Selector 0x0 = internal reference 0x1 = external reference

A.15.2.2 DAC Status Register (status)

Instance Name	Offset
status	0x04

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

Table 540: DAC Status Register (status)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	b_dv	R 0x0	DACB Conversion Status 0x0 = channel b conversion is not done 0x1 = channel b conversion complete
0	a_dv	R 0x0	DACA Conversion Status 0x0 = channel a conversion is not done 0x1 = channel a conversion complete

A.15.2.3 Channel A Control Register (actrl)

Instance Name	Offset
actrl	0x08

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved												a_range		a_wave		a_tria_step_sel		a_tria_mamp_sel				a_tria_half	a_time_mode	a_den	a_trig_typ		a_trig_sel		a_trig_en	a_io_en	a_en
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0

Table 541: Channel A Control Register (actrl)

Bits	Field	Type/ HW Rst	Description
31:20	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
19:18	a_range	R/W 0x3	Output Voltage Range Control, with Internal/External Reference 0x0 = 0.16+(0.64*input code/1023) with ref_sel=0(internal)/0.08* Vref_ext+(0.32* Vref_ext*input_code/1023) with ref_ sel=1(external) 0x1 = 0.19+(1.01*input code /1023) with ref_sel=0(internal)/0.095* Vref_ext+(0.505* Vref_ext*input_code/1023) with ref_ sel=1(external) 0x2 = 0.19+(1.01*input code /1023) with ref_sel=0(internal)/0.095* Vref_ext+(0.505* Vref_ext*input_code/1023) with ref_ sel=1(external) 0x3 = 0.18+(1.42*input code /1023) with ref_ sel=0(internal)/0.09*Vref_ext+(0.71* Vref_ext*input_ code/1023) with ref_sel=1(external)
17:16	a_wave	R/W 0x0	Channel A Wave Type Select 0x0 = normal 0x1 = triangle wave 0x3 = noise 0x2 = sine wave
15:14	a_tria_step_sel	R/W 0x0	Channel A Triangle Wave Step Selector 0x0 = 1 0x1 = 3 0x2 = 15 0x3 = 511

Table 541: Channel A Control Register (actrl) (Continued)

Bits	Field	Type/ HW Rst	Description
13:10	a_tria_mamp_sel	R/W 0x0	Channel A Triangle Wave Max Amplitude Selector 0x0 = 63 0x1 = 127 0x2 = 191 0x3 = 255 0x4 = 319 0x5 = 383 0x6 = 447 0x7 = 511 0x8 = 575 0x9 = 639 0xA = 703 0xB = 767 0xC = 831 0xD = 895 0xE = 959 0xF = 1023
9	a_tria_half	R/W 0x0	Channel A Triangle Wave Type Selector 0x0 = full triangle 0x1 = half triangle
8	a_time_mode	R/W 0x0	Channel A Mode 0x0 = non-timing related 0x1 = timing related
7	a_den	R/W 0x0	Channel A DMA Enable 0x0 = DMA data transfer disabled 0x1 = DMA data transfer enabled
6:5	a_trig_typ	R/W 0x1	Channel A Trigger Type 0x0 = reserved 0x1 = rising edge trigger 0x2 = falling edge trigger 0x3 = both rising and falling edge trigger
4:3	a_trig_sel	R/W 0x3	Channel A Trigger Selector
2	a_trig_en	R/W 0x0	Channel A Trigger Enable 0x0 = channel a conversion triggered by external event disabled 0x1 = channel a conversion triggered by external event enabled
1	a_io_en	R/W 0x0	Channel A Conversion Output to Pad Enable 0x0 = disable channel a conversion result to GPIO 0x1 = enable channel a conversion result to GPIO

Table 541: Channel A Control Register (actrl) (Continued)

Bits	Field	Type/ HW Rst	Description
0	a_en	R/W 0x0	Channel A Enable/Disable Signal 0x0 = disable channel a conversion 0x1 = enable channel a conversion

A.15.2.4 Channel B Control Register (bctrl)

Instance Name	Offset
bctrl	0x0C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																			Reserved		b_wave		b_time_mode	b_den	b_trig_typ		b_trig_sel		b_trig_en	b_io_en	b_en
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	1	0	0	0	0	0	1	1	1	0	0	0

Table 542: Channel B Control Register (bctrl)

Bits	Field	Type/ HW Rst	Description
31:13	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
12:11	Reserved	R/W 0x3	Reserved. Do not change the reset value.
10:9	b_wave	R/W 0x0	Channel B Wave Type Select 0x0 = normal 0x1 = reserved 0x2 = reserved 0x3 = differential mode with channel A
8	b_time_mode	R/W 0x0	Channel B Mode 0x0 = non-timing related 0x1 = timing related
7	b_den	R/W 0x0	Channel B DMA Enable 0x0 = DMA data transfer disabled 0x1 = DMA data transfer enabled
6:5	b_trig_typ	R/W 0x1	Channel B Trigger Type 0x0 = reserved 0x1 = rising edge trigger 0x2 = falling edge trigger 0x3 = both rising and falling edge trigger

Table 542: Channel B Control Register (bctrl) (Continued)

Bits	Field	Type/ HW Rst	Description
4:3	b_trig_sel	R/W 0x3	Channel B Trigger Selector
2	b_trig_en	R/W 0x0	Channel B Trigger Enable 0x0 = channel B conversion triggered by external event disabled 0x1 = channel B conversion triggered by external event enabled
1	b_io_en	R/W 0x0	Channel B Conversion Output to Pad Enable 0x0 = disable channel B conversion result to GPIO 0x1 = enable channel B conversion result to GPIO
0	b_en	R/W 0x0	Channel B Enable/Disable Signal 0x0 = disable channel B conversion 0x1 = enable channel B conversion

A.15.2.5 Channel A Data Register (adata)

Instance Name	Offset
adata	0x10

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

Table 543: Channel A Data Register (adata)

Bits	Field	Type/ HW Rst	Description
31:10	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
9:0	a_data	R/W 0x0	Channel A Data Input This field is also used as base_value when choose triangle wave.

A.15.2.6 Channel B Data Register (bdata)

Instance Name	Offset
bdata	0x14

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

Table 544: Channel B Data Register (bdata)

Bits	Field	Type/ HW Rst	Description
31:10	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
9:0	b_data	R/W 0x0	Channel B Data Input

A.15.2.7 Interrupt Status Register (isr)

Instance Name	Offset
isr	0x18

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																											tria_ovfl_int	b_to_int	a_to_int	b_rdy_int	a_rdy_int
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0

Table 545: Interrupt Status Register (isr)

Bits	Field	Type/ HW Rst	Description
31:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4	tria_ovfl_int	R 0x0	Triangle Overflow 1 indicates that triangle wave configuration overflow happened.
3	b_to_int	R 0x0	Channel B Timeout 1 indicates Channel B timeout interrupt flag if reg_imr[1] not masked.
2	a_to_int	R 0x0	Channel A Timeout 1 indicates Channel A timeout interrupt flag if reg_imr[0] not masked.
1	b_rdy_int	R 0x0	Channel B Data Ready 1 indicates Channel B data ready interrupt flag if reg_imr[1] not masked.

Table 545: Interrupt Status Register (isr) (Continued)

Bits	Field	Type/ HW Rst	Description
0	a_rdy_int	R 0x0	Channel A Data Ready 1 indicates Channel A data ready interrupt flag if reg_imr[0] not masked.

A.15.2.8 Interrupt Mask Register (imr)

Instance Name	Offset
imr	0x1C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																											tria_ovfl_int_msk	b_to_int_msk	a_to_int_msk	b_rdy_int_msk	a_rdy_int_msk
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	1	1	1	1

Table 546: Interrupt Mask Register (imr)

Bits	Field	Type/ HW Rst	Description
31:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4	tria_ovfl_int_msk	R/W 0x1	Triangle Overflow Mask Set to 1 to mask off triangle wave configuration overflow interrupt.
3	b_to_int_msk	R/W 0x1	Channel B Timeout Mask Set to 1 to mask off Channel B timeout interrupt.
2	a_to_int_msk	R/W 0x1	Channel A Timeout Mask Set to 1 to mask off Channel A timeout interrupt.
1	b_rdy_int_msk	R/W 0x1	Channel B Data Ready Mask Set to 1 to mask off Channel B data ready interrupt.
0	a_rdy_int_msk	R/W 0x1	Channel A Data Ready Mask Set to 1 to mask off Channel A data ready interrupt.

A.15.2.9 Interrupt Raw Status Register (irsr)

Instance Name	Offset
irsr	0x20

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																											tria_ovfl_int_raw	b_to_int_raw	a_to_int_raw	b_rdy_int_raw	a_rdy_int_raw
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0

Table 547: Interrupt Raw Status Register (irsr)

Bits	Field	Type/ HW Rst	Description
31:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4	tria_ovfl_int_raw	R 0x0	Triangle Overflow Raw 1 indicates triangle wave configuration overflow interrupt flag, regardless of the mask.
3	b_to_int_raw	R 0x0	Channel B Timeout Raw 1 indicates Channel B timeout interrupt flag, regardless the mask.
2	a_to_int_raw	R 0x0	Channel A Timeout Raw 1 indicates Channel A timeout interrupt flag, regardless the mask.
1	b_rdy_int_raw	R 0x0	Channel B Data Ready Raw 1 indicates Channel B data ready interrupt flag, regardless the mask.
0	a_rdy_int_raw	R 0x0	Channel A Data Ready Raw 1 indicates Channel A data ready interrupt flag, regardless the mask.

A.15.2.10 Interrupt Clear Register (icr)

Instance Name	Offset
icr	0x24

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																											tria_ovfl_int_clr	b_to_int_clr	a_to_int_clr	b_rdy_int_clr	a_rdy_int_clr
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0

Table 548: Interrupt Clear Register (icr)

Bits	Field	Type/ HW Rst	Description
31:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4	tria_ovfl_int_clr	R/W 0x0	Triangle Overflow Clear Write 1 to clear triangle wave configuration overflow interrupt flag.
3	b_to_int_clr	R/W 0x0	Channel B Timeout Clear Write 1 to clear Channel B timeout interrupt flag.
2	a_to_int_clr	R/W 0x0	Channel A Timeout Clear Write 1 to clear Channel A timeout interrupt flag.
1	b_rdy_int_clr	R/W 0x0	Channel B Data Ready Clear Write 1 to clear Channel B data ready interrupt flag.
0	a_rdy_int_clr	R/W 0x0	Channel A Data Ready Clear Write 1 to clear Channel A data ready interrupt flag.

A.15.2.11 Clock Register (clk)

Instance Name	Offset
clk	0x28

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

Table 549: Clock Register (clk)

Bits	Field	Type/ HW Rst	Description
31:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4	soft_clk_rst	R/W 0x0	Soft Reset for Clock Divider 0x0 = normal 0x1 = reset
3	Reserved	R/W 0x0	Reserved. Do not change the reset value.
2:1	clk_ctrl	R/W 0x0	DAC Conversion Rate Selector 0x0 = 62.5K 0x1 = 125K 0x2 = 250K 0x3 = 500K
0	Reserved	R/W 0x0	Reserved. Do not change the reset value.

A.15.2.12 Soft Reset Register (rst)

Instance Name	Offset
rst	0x2C

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

Table 550: Soft Reset Register (rst)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	b_soft_rst	R/W 0x0	Soft Reset for DAC Channel B, Active High 0x0 = no action 0x1 = reset
0	a_soft_rst	R/W 0x0	Soft Reset for DAC Channel A, Active High 0x0 = no action 0x1 = reset

A.16 ACOMP Address Block

A.16.1 ACOMP Register Map

Table 551: ACMOP Register Map

Offset	Name	HW Rst	Description	Details
0x00	ctrl0	0x4000_0000	ACOMP0 Control Register	Page: 730
0x04	ctrl1	0x4000_0000	ACOMP1 Control Register	Page: 733
0x08	status0	0x0000_0000	ACOMP0 Status Register	Page: 736
0x0C	status1	0x0000_0000	ACOMP1 Status Register	Page: 736
0x10	route0	0x0000_0000	ACOMP0 Route Register	Page: 737
0x14	route1	0x0000_0000	ACOMP1 Route Register	Page: 737
0x18	isr0	0x0000_0000	ACOMP0 Interrupt Status Register	Page: 738
0x1C	isr1	0x0000_0000	ACOMP1 Interrupt Status Register	Page: 739
0x20	imr0	0x0000_0003	ACOMP0 Interrupt Mask Register	Page: 740
0x24	imr1	0x0000_0003	ACOMP1 Interrupt Mask Register	Page: 740
0x28	irsr0	0x0000_0000	ACOMP0 Interrupt Raw Status Register	Page: 741
0x2C	irsr1	0x0000_0000	ACOMP1 Interrupt Raw Status Register	Page: 742
0x30	icr0	0x0000_0000	ACOMP0 Interrupt Clear Register	Page: 743
0x34	icr1	0x0000_0000	ACOMP1 Interrupt Clear Register	Page: 743
0x38	rst0	0x0000_0000	ACOMP0 Soft Reset Register	Page: 744
0x3C	rst1	0x0000_0000	ACOMP1 Soft Reset Register	Page: 744
0x48	clk	0x0000_0000	Clock Register	Page: 745

A.16.2 ACOMP Registers

A.16.2.1 ACOMP0 Control Register (ctrl0)

Instance Name	Offset
ctrl0	0x00

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	edge_levl_sel	int_act_hi	fie	rie	inact_val	muxen	pos_sel				neg_sel				level_sel						bias_prog		hyst_selp			hyst_seln			warmtime		gpioinv	en
HW Rst	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 552: ACOMP0 Control Register (ctrl0)

Bits	Field	Type/ HW Rst	Description
31	edge_levl_sel	R/W 0x0	ACOMP0 Interrupt Type Select 0x0 = level triggered interrupt 0x1 = edge triggered interrupt
30	int_act_hi	R/W 0x1	ACOMP0 Interrupt Active Mode Select 0x0 = low level or falling edge triggered interrupt 0x1 = high level or rising edge triggered interrupt
29	fie	R/W 0x0	ACOMP0 Enable/disable Falling Edge Triggered Edge Pulse 0x0 = disable 0x1 = enable
28	rie	R/W 0x0	ACOMP0 Enable/disable Rising Edge Triggered Edge Pulse 0x0 = disable 0x1 = enable
27	inact_val	R/W 0x0	Set Output Value When ACOMP0 Is Inactive 0x0 = output 0 when ACOMP0 is inactive 0x1 = output 1 when ACOMP0 is inactive
26	muxen	R/W 0x0	ACOMP0 Input MUX Enable Bit This bit should be asserted earlier than 'en' bit. 0x0 = disable input mux 0x1 = enable input mux

Table 552: ACOMP0 Control Register (ctrl0) (Continued)

Bits	Field	Type/ HW Rst	Description
25:22	pos_sel	R/W 0x0	ACOMP0 Positive Input Select Bits 0x0 = acomp_ch<0> 0x1 = acomp_ch<1> 0x2 = acomp_ch<2> 0x3 = acomp_ch<3> 0x4 = acomp_ch<4> 0x5 = acomp_ch<5> 0x6 = acomp_ch<6> 0x7 = acomp_ch<7> 0x8 = daca 0x9 = dacb others = reserved
21:18	neg_sel	R/W 0x0	ACOMP0 Negative Input Select Bits 0x0 = acomp_ch<0> 0x1 = acomp_ch<1> 0x2 = acomp_ch<2> 0x3 = acomp_ch<3> 0x4 = acomp_ch<4> 0x5 = acomp_ch<5> 0x6 = acomp_ch<6> 0x7 = acomp_ch<7> 0x8 = daca 0x9 = dacb 0xA = vref_1p25 0xB = vssa 0xC = VDDIO_3*scaling factor 0xD = VDDIO_3*scaling factor 0xE = VDDIO_3*scaling factor 0xF = VDDIO_3*scaling factor
17:12	level_sel	R/W 0x0	Scaling Factor Select Bits for VDDIO_3 Reference Level 0x0X = scaling factor=0.25 0x1X = scaling factor= 0.5 0x2X = scaling factor= 0.75 0x3X = scaling factor= 1 others = reserved
11:10	bias_prog	R/W 0x0	ACOMP0 Bias Current Control Bits or Response Time Control Bits 0x0 = power mode1 (slow response mode) 0x1 = power mode2 (medium response mode) 0x2 = power mode3 (fast response mode) 0x3 = reserved

Table 552: ACOMP0 Control Register (ctrl0) (Continued)

Bits	Field	Type/ HW Rst	Description
9:7	hyst_selp	R/W 0x0	Select ACOMP0 Positive Hysteresis Voltage Level 0x0 = no hysteresis 0x1 = +10 mV hysteresis 0x2 = +20 mV hysteresis 0x3 = +30 mV hysteresis 0x4 = +40 mV hysteresis 0x5 = +50 mV hysteresis 0x6 = +60 mV hysteresis 0x7 = +70 mV hysteresis
6:4	hyst_seln	R/W 0x0	Select ACOMP0 Negative Hysteresis Voltage Level 0x0 = no hysteresis 0x1 = -10 mV hysteresis 0x2 = -20 mV hysteresis 0x3 = -30 mV hysteresis 0x4 = -40 mV hysteresis 0x5 = -50 mV hysteresis 0x6 = -60 mV hysteresis 0x7 = -70 mV hysteresis
3:2	warmtime	R/W 0x0	Set ACOMP0 Warm-Up Time 0x0 = 1 us 0x1 = 2 us 0x2 = 4 us 0x3 = 8 us
1	gpioinv	R/W 0x0	Enable/Disable Inversion of ACOMP0 Output to GPIO 0x0 = do not invert ACOMP0 output 0x1 = invert ACOMP0 output
0	en	R/W 0x0	ACOMP0 Enable 0x0 = disable 0x1 = enable

A.16.2.2 ACOMP1 Control Register (ctrl1)

Instance Name	Offset
ctrl1	0x04

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	edge_levl_sel	int_act_hi	fie	rie	inact_val	muxen	pos_sel				neg_sel				level_sel						bias_prog		hyst_selp			hyst_seln			warmtime		gpioinv	en
HW Rst	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 553: ACOMP1 Control Register (ctrl1)

Bits	Field	Type/ HW Rst	Description
31	edge_levl_sel	R/W 0x0	ACOMP1 Interrupt Type Select 0x0 = level triggered interrupt 0x1 = edge triggered interrupt
30	int_act_hi	R/W 0x1	ACOMP1 Interrupt Active Mode Select 0x0 = low level or falling edge triggered interrupt 0x1 = high level or rising edge triggered interrupt
29	fie	R/W 0x0	ACOMP1 Enable/disable Falling Edge Triggered Edge Pulse 0x0 = disable 0x1 = enable
28	rie	R/W 0x0	ACOMP1 Enable/disable Rising Edge Triggered Edge Pulse 0x0 = disable 0x1 = enable
27	inact_val	R/W 0x0	Set Output Value When ACOMP1 Is Inactive 0x0 = output 0 when ACOMP1 is inactive 0x1 = output 1 when ACOMP1 is inactive
26	muxen	R/W 0x0	ACOMP1 Input MUX Enable This bit should be asserted earlier than 'en' bit. 0x0 = disable input mux 0x1 = enable input mux

Table 553: ACOMP1 Control Register (ctrl1) (Continued)

Bits	Field	Type/ HW Rst	Description
25:22	pos_sel	R/W 0x0	ACOMP1 Positive Input Select 0x0 = acomp_ch<0> 0x1 = acomp_ch<1> 0x2 = acomp_ch<2> 0x3 = acomp_ch<3> 0x4 = acomp_ch<4> 0x5 = acomp_ch<5> 0x6 = acomp_ch<6> 0x7 = acomp_ch<7> 0x8 = daca 0x9 = dacb others = reserved
21:18	neg_sel	R/W 0x0	ACOMP1 Negative Input Select 0x0 = acomp_ch<0> 0x1 = acomp_ch<1> 0x2 = acomp_ch<2> 0x3 = acomp_ch<3> 0x4 = acomp_ch<4> 0x5 = acomp_ch<5> 0x6 = acomp_ch<6> 0x7 = acomp_ch<7> 0x8 = daca 0x9 = dacb 0xA = vref_1p25 0xB = vssa 0xC = VDDIO_3*scaling factor 0xD = VDDIO_3*scaling factor 0xE = VDDIO_3*scaling factor 0xF = VDDIO_3*scaling factor
17:12	level_sel	R/W 0x0	Scaling Factor Select Bits for VDDIO_3 Reference Level 0x0X = scaling factor=0.25 0x1X = scaling factor= 0.5 0x2X = scaling factor= 0.75 0x3X = scaling factor= 1 others = reserved
11:10	bias_prog	R/W 0x0	ACOMP1 Bias Current Control Bits Or Response Time Control Bits 0x0 = power mode1 (Slow response mode) 0x1 = power mode2 (Medium response mode) 0x2 = power mode3 (Fast response mode) 0x3 = reserved

Table 553: ACOMP1 Control Register (ctrl1) (Continued)

Bits	Field	Type/ HW Rst	Description
9:7	hyst_selp	R/W 0x0	Select ACOMP1 Positive Hysteresis Voltage Level 0x0 = no hysteresis 0x1 = +10 mV hysteresis 0x2 = +20 mV hysteresis 0x3 = +30 mV hysteresis 0x4 = +40 mV hysteresis 0x5 = +50 mV hysteresis 0x6 = +60 mV hysteresis 0x7 = +70 mV hysteresis
6:4	hyst_seln	R/W 0x0	Select ACOMP1 Negative Hysteresis Voltage Level 0x0 = no hysteresis 0x1 = -10 mV hysteresis 0x2 = -20 mV hysteresis 0x3 = -30 mV hysteresis 0x4 = -40 mV hysteresis 0x5 = -50 mV hysteresis 0x6 = -60 mV hysteresis 0x7 = -70 mV hysteresis
3:2	warmtime	R/W 0x0	Set ACOMP1 warm-up Time 0x0 = 1 us 0x1 = 2 us 0x2 = 4 us 0x3 = 8 us
1	gpioinv	R/W 0x0	Enable/disable Inversion of ACOMP1 Output to GPIO 0x0 = do not invert ACOMP1 output 0x1 = invert ACOMP1 output
0	en	R/W 0x0	ACOMP1 Enable Bit 0x0 = disable 0x1 = enable

A.16.2.3 ACOMP0 Status Register (status0)

Instance Name	Offset
status0	0x08

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

Table 554: ACOMP0 Status Register (status0)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	out	R 0x0	ACOMP0 Comparison Output Value
0	act	R 0x0	ACOMP0 Active Status 0x0 = ACOMP0 is inactive 0x1 = ACOMP0 is active

A.16.2.4 ACOMP1 Status Register (status1)

Instance Name	Offset
status1	0x0C

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

Table 555: ACOMP1 Status Register (status1)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	out	R 0x0	ACOMP1 Comparison Output Value
0	act	R 0x0	ACOMP1 Active Status 0x0 = ACOMP1 is inactive 0x1 = ACOMP1 is active

A.16.2.5 ACOMP0 Route Register (route0)

Instance Name	Offset
route0	0x10

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

Table 556: ACOMP0 Route Register (route0)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	pe	R/W 0x0	Enable/disable ACOMP0 Output to Pin 0x0 = disable 0x1 = enable
0	outsel	R/W 0x0	Select ACOMP0 Synchronous or Asynchronous Output to Pin 0x0 = synchronous output 0x1 = asynchronous output

A.16.2.6 ACOMP1 Route Register (route1)

Instance Name	Offset
route1	0x14

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

Table 557: ACOMP1 Route Register (route1)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	pe	R/W 0x0	Enable/disable ACOMP1 Output to Pin 0x0 = disable 0x1 = enable
0	outsel	R/W 0x0	Select ACOMP1 Synchronous or Asynchronous Output to Pin 0x0 = synchronous output 0x1 = asynchronous output

A.16.2.7 ACOMP0 Interrupt Status Register (isr0)

Instance Name	Offset
isr0	0x18

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

Table 558: ACOMP0 Interrupt Status Register (isr0)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	outa_int	R 0x0	ACOMP0 Asynchronized Output Interrupt This bit is set to 1 when ACOMP0 asynchronized output changes from 0 to 1. If corresponding mask enable bit is set, outa_int signal will not be captured in this register. Will be cleared only when outa_ int_clr is asserted.
0	out_int	R 0x0	ACOMP0 Synchronized Output Interrupt This bit is set to 1 when ACOMP0 synchronized output changes from 0 to 1. If corresponding mask enable bit is set, out_int signal will not be captured in this register. Will be cleared only when out_int_clr is asserted.

A.16.2.8 ACOMP1 Interrupt Status Register (isr1)

Instance Name	Offset
isr1	0x1C

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

Table 559: ACOMP1 Interrupt Status Register (isr1)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	outa_int	R 0x0	ACOMP1 Asynchronized Output Interrupt This bit is set to 1 when ACOMP1 asynchronized output changes from 0 to 1. If corresponding mask enable bit is set, outa_int signal will not be captured in this register. Will be cleared only when outa_ int_clr is asserted.
0	out_int	R 0x0	ACOMP1 Synchronized Output Interrupt This bit is set to 1 when ACOMP1 synchronized output changes from 0 to 1. If corresponding mask enable bit is set, out_int signal will not be captured in this register. Will be cleared only when out_int_clr is asserted.

A.16.2.9 ACOMP0 Interrupt Mask Register (imr0)

Instance Name	Offset
imr0	0x20

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

Table 560: ACOMP0 Interrupt Mask Register (imr0)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	outa_int_mask	R/W 0x1	Mask Asynchronized Interrupt Set to 1 to mask off the ACOMP0 asynchronized output interrupt.
0	out_int_mask	R/W 0x1	Mask Synchronized Interrupt Set to 1 to mask off the ACOMP0 synchronized output interrupt.

A.16.2.10 ACOMP1 Interrupt Mask Register (imr1)

Instance Name	Offset
imr1	0x24

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

Table 561: ACOMP1 Interrupt Mask Register (imr1)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	outa_int_mask	R/W 0x1	Mask Asynchronized Interrupt Set to 1 to mask off the ACOMP1 asynchronized output interrupt.
0	out_int_mask	R/W 0x1	Mask Synchronized Interrupt Set to 1 to mask off the ACOMP1 synchronized output interrupt.

A.16.2.11 ACOMP0 Interrupt Raw Status Register (irsr0)

Instance Name	Offset
irsr0	0x28

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

Table 562: ACOMP0 Interrupt Raw Status Register (irsr0)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	outa_int_raw	R 0x0	Raw Mask Asynchronized Interrupt ACOMP0 asynchronized output will be captured into this register regardless the interrupt mask. Will be cleared only when int_clr is asserted.
0	out_int_raw	R 0x0	Raw Mask Synchronized Interrupt ACOMP0 synchronized output will be captured into this register regardless the interrupt mask. Will be cleared only when int_clr is asserted.

A.16.2.12 ACOMP1 Interrupt Raw Status Register (irsr1)

Instance Name	Offset
irsr1	0x2C

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

Table 563: ACOMP1 Interrupt Raw Status Register (irsr1)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	outa_int_raw	R 0x0	Raw Mask Asynchronized Interrupt ACOMP1 asynchronized output will be captured into this register regardless the interrupt mask. Will be cleared only when int_clr is asserted.
0	out_int_raw	R 0x0	Raw Mask Synchronized Interrupt ACOMP1 synchronized output will be captured into this register regardless the interrupt mask. Will be cleared only when int_clr is asserted.

A.16.2.13 ACOMP0 Interrupt Clear Register (icr0)

Instance Name	Offset
icr0	0x30

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

Table 564: ACOMP0 Interrupt Clear Register (icr0)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	outa_int_clr	R/W 0x0	ACOMP0 Asyncrhonized Output Interrupt Flag Clear Signal Write 1 to clear ACOMP0 outa_int and outa_int_raw.
0	out_int_clr	R/W 0x0	ACOMP0 Syncrhonized Output Interrupt Flag Clear Signal Write 1 to clear ACOMP0 outa_int and outa_int_raw.

A.16.2.14 ACOMP1 Interrupt Clear Register (icr1)

Instance Name	Offset
icr1	0x34

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

Table 565: ACOMP1 Interrupt Clear Register (icr1)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	outa_int_clr	R/W 0x0	ACOMP1 Asyncrhonized Output Interrupt Flag Clear Signal Write 1 to clear ACOMP1 outa_int and outa_int_raw.
0	out_int_clr	R/W 0x0	ACOMP1 Syncrhonized Output Interrupt Flag Clear Signal Write 1 to clear ACOMP1 outa_int and outa_int_raw.

A.16.2.15 ACOMP0 Soft Reset Register (rst0)

Instance Name	Offset
rst0	0x38

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

Table 566: ACOMP0 Soft Reset Register (rst0)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	soft_rst	R/W 0x0	Soft Reset for ACOMP0 (active high) 0x0 = no action 0x1 = reset

A.16.2.16 ACOMP1 Soft Reset Register (rst1)

Instance Name	Offset
rst1	0x3C

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

Table 567: ACOMP1 Soft Reset Register (rst1)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	soft_rst	R/W 0x0	Soft Reset for ACOMP1 (active high) 0x0 = no action 0x1 = reset

A.16.2.17 Clock Register (clk)

Instance Name	Offset
clk	0x48

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

Table 568: Clock Register (clk)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	soft_clk_rst	R/W 0x0	Soft Reset for Clock Divider 0x0 = no action 0x1 = reset
0	Reserved	R/W 0x0	Reserved. Do not change the reset value.

THIS PAGE INTENTIONALLY LEFT BLANK

A.17 PINMUX Address Block

A.17.1 PINMUX Register Map

Table 569: PINMUX Register Map

Offset	Name	HW Rst	Description	Details
0x00	_GPIO0	0x0000_0000	Padring Pin Register	Page: 749
0x04	_GPIO1	0x0000_0000	Padring Pin Register	Page: 749
0x08	_GPIO2	0x0000_0000	Padring Pin Register	Page: 749
0x0C	_GPIO3	0x0000_0000	Padring Pin Register	Page: 749
0x10	_GPIO4	0x0000_0000	Padring Pin Register	Page: 749
0x14	_GPIO5	0x0000_0000	Padring Pin Register	Page: 749
0x18	_GPIO6	0x0000_0000	Padring Pin Register	Page: 749
0x1C	_GPIO7	0x0000_0000	Padring Pin Register	Page: 749
0x20	_GPIO8	0x0000_0000	Padring Pin Register	Page: 749
0x24	_GPIO9	0x0000_0000	Padring Pin Register	Page: 749
0x28	_GPIO10	0x0000_0000	Padring Pin Register	Page: 749
0x2C	_GPIO11	0x0000_0000	Padring Pin Register	Page: 749
0x30	_GPIO12	0x0000_0000	Padring Pin Register	Page: 749
0x34	_GPIO13	0x0000_0000	Padring Pin Register	Page: 749
0x38	_GPIO14	0x0000_0000	Padring Pin Register	Page: 749
0x3C	_GPIO15	0x0000_0000	Padring Pin Register	Page: 749
0x40	_GPIO16	0x0000_0000	Padring Pin Register	Page: 749
0x44	_GPIO17	0x0000_0000	Padring Pin Register	Page: 749
0x48	_GPIO18	0x0000_0000	Padring Pin Register	Page: 749
0x4C	_GPIO19	0x0000_0000	Padring Pin Register	Page: 749
0x50	_GPIO20	0x0000_0000	Padring Pin Register	Page: 749
0x54	_GPIO21	0x0000_0000	Padring Pin Register	Page: 749
0x58	_GPIO22	0x0000_0000	Padring Pin Register	Page: 749
0x5C	_GPIO23	0x0000_0000	Padring Pin Register	Page: 749
0x60	_GPIO24	0x0000_0000	Padring Pin Register	Page: 749
0x64	_GPIO25	0x0000_0000	Padring Pin Register	Page: 749
0x68	_GPIO26	0x0000_0000	Padring Pin Register	Page: 749

Table 569: PINMUX Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x6C	_GPIO27	0x0000_0000	Padring Pin Register	Page: 749
0x70	_GPIO28	0x0000_0000	Padring Pin Register	Page: 749
0x74	_GPIO29	0x0000_0000	Padring Pin Register	Page: 749
0x78	_GPIO30	0x0000_0000	Padring Pin Register	Page: 749
0x7C	_GPIO31	0x0000_0000	Padring Pin Register	Page: 749
0x80	_GPIO32	0x0000_0000	Padring Pin Register	Page: 749
0x84	_GPIO33	0x0000_0000	Padring Pin Register	Page: 749
0x88	_GPIO34	0x0000_0000	Padring Pin Register	Page: 749
0x8C	_GPIO35	0x0000_0000	Padring Pin Register	Page: 749
0x90	_GPIO36	0x0000_0000	Padring Pin Register	Page: 749
0x94	_GPIO37	0x0000_0000	Padring Pin Register	Page: 749
0x98	_GPIO38	0x0000_0000	Padring Pin Register	Page: 749
0x9C	_GPIO39	0x0000_0000	Padring Pin Register	Page: 749
0xA0	_GPIO40	0x0000_0000	Padring Pin Register	Page: 749
0xA4	_GPIO41	0x0000_0000	Padring Pin Register	Page: 749
0xA8	_GPIO42	0x0000_0000	Padring Pin Register	Page: 749
0xAC	_GPIO43	0x0000_0000	Padring Pin Register	Page: 749
0xB0	_GPIO44	0x0000_0000	Padring Pin Register	Page: 749
0xB4	_GPIO45	0x0000_0000	Padring Pin Register	Page: 749
0xB8	_GPIO46	0x0000_0000	Padring Pin Register	Page: 749
0xBC	_GPIO47	0x0000_0000	Padring Pin Register	Page: 749
0xC0	_GPIO48	0x0000_0000	Padring Pin Register	Page: 749
0xC4	_GPIO49	0x0000_0000	Padring Pin Register	Page: 749

A.17.2 PINMUX Registers (_GPIO)

Instance Name	Offset	Count
_GPIO[0:49]	0x00 to 0xC4	50

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																pio_pull_sel	pio_pull_up	pio_pull_dn	Reserved							slp_oe	slp_val	di_en	fsel		
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	?	?	?	?	?	?	?	0	0	0	0	0	0

Table 570: Padring Pin Registers (_GPIO)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15	pio_pull_sel	R/W 0x0	Custom Pull-up and Pull-down Configuration Control 0x0 = pulled up by default 50 kΩ internal pull-up 0x1 = pull-up and pull-down by software from bits[14:13]
14	pio_pull_up	R/W 0x0	Pull-up Enable 0x0 = pull-up disabled 0x1 = pull-up enabled
13	pio_pull_dn	R/W 0x0	Pull-down Enable 0x0 = pull-down disabled 0x1 = pull-down enabled
12:6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
5	slp_oe	R/W 0x0	Reserved for Test Purpose Output enable.
4	slp_val	R/W 0x0	Reserved for Test Purpose Output data.
3	di_en	R/W 0x0	Input Enable Control 0x0 = receiver will be tri-stated 0x1 = receive data from PAD
2:0	fsel	R/W 0x0	Padring Function Select Bits[2:0] selects function[7]...function[0].

THIS PAGE INTENTIONALLY LEFT BLANK

A.18 WDT Address Block

A.18.1 WDT Register Map

Table 571: WDT Register Map

Offset	Name	HW Rst	Description	Details
0x00	WDT_CR	0x0000_000A	WDT Control Register	Page: 751
0x04	WDT_TORR	0x0000_0000	WDT Timeout Range Register	Page: 753
0x08	WDT_CCVR	0x0000_FFFF	WDT Current Counter Value Register	Page: 755
0x0C	WDT_CRR	0x0000_0000	WDT Counter Restart Register	Page: 755
0x10	WDT_STAT	0x0000_0000	WDT Interrupt Status Register	Page: 756
0x14	WDT_EOI	0x0000_0000	WDT Interrupt Clear Register	Page: 756
0xE4	WDT_COMP_PARAM_5	0x0000_0000	WDT Component Parameters Register 5	Page: 757
0xE8	WDT_COMP_PARAM_4	0x0000_0000	WDT Component Parameters Register 4	Page: 757
0xEC	WDT_COMP_PARAM_3	0x0000_0000	WDT Component Parameters Register 3	Page: 758
0xF0	WDT_COMP_PARAM_2	0x0000_FFFF	WDT Component Parameters Register 2	Page: 758
0xF4	WDT_COMP_PARAM_1	0x0000_0000	WDT Component Parameters Register 1	Page: 759
0xF8	WDT_COMP_VERSION	0x3130_372A	WDT Component Version Register	Page: 760
0xFC	WDT_COMP_TYPE	0x4457_0120	WDT Component Type Register	Page: 760

A.18.2 WDT Registers

A.18.2.1 WDT_CR Register

Instance Name	Offset
WDT_CR	0x00

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																											rpl			rmod	wdt_en
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	1	0	1	0

Table 572: WDT Control Register (WDT_CR)

Bits	Field	Type/ HW Rst	Description
31:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

Table 572: WDT Control Register (WDT_CR) (Continued)

Bits	Field	Type/ HW Rst	Description
4:2	rpl	R/W 0x2	Reset Pulse Length Writes have no effect when the configuration parameter WDT_HC_RPL is 1, making the register bits read-only. This is used to select the number of pclk cycles for which the system reset stays asserted. The range of values available is 2 to 256 pclk cycles. 0x0 = 2 pclk cycles 0x1 = 4 pclk cycles 0x2 = 8 pclk cycles 0x3 = 16 pclk cycles 0x4 = 32 pclk cycles 0x5 = 64 pclk cycles 0x6 = 128 pclk cycles 0x7 = 256 pclk cycles
1	rmod	R/W 0x1	Response Mode Writes have no effect when the parameter WDT_HC_RMOD = 1, thus this register becomes read-only. Selects the output response generated to a timeout. 0x0 = generate a system reset 0x1 = first generate an interrupt and if it is not cleared by the time a second timeout occurs then generate a system reset
0	wdt_en	R/W 0x0	WDT Enable Writable when the configuration parameter WDT_ALWAYS_EN = 1, otherwise, it is readable. This bit is used to enable and disable the DW_apb_wdt. When disabled, the counter does not decrement. Thus, no interrupts or system resets are generated. Once this bit has been enabled, it can be cleared only by a system reset. 0x0 = WDT disabled 0x1 = WDT enabled

A.18.2.2 WDT_TORR Register

Instance Name	Offset
WDT_TORR	0x04

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

Table 573: WDT Timeout Range Register (WDT_TORR)

Bits	Field	Type/ HW Rst	Description
31:8	reserved	R 0x0	Reserved and read as 0.
7:4	top_init	R/W 0x0	Timeout Period For Initialization Writes to these register bits have no effect when the configuration parameter WDT_HC_TOP = 1 or WDT_ALWAYS_EN = 1. Used to select the timeout period that the watchdog counter restarts from for the first counter restart (kick). This register should be written after reset and before the WDT is enabled. A change of the TOP_INIT is seen only once the WDT has been enabled, and any change after the first kick is not seen as subsequent kicks use the period specified by the TOP bits. The range of values is limited by the WDT_CNT_ WIDTH. If TOP_INIT is programmed to select a range that is greater than the counter width, the timeout period is truncated to fit to the counter width. This affects only the non-user specified values as users are limited to these boundaries during configuration. The range of values available for a 32-bit watchdog counter are: Where i = TOP_INIT and t = timeout period For i = 0 to 15 if WDT_USE_FIX_TOP==1 t = 2(16 + i) else t = WDT_USER_TOP_INIT_(i) • These bits exist only when the configuration parameter WDT_ DUAL_TOP = 1, otherwise, they are fixed at zero.

Table 573: WDT Timeout Range Register (WDT_TORR) (Continued)

Bits	Field	Type/ HW Rst	Description
3:0	top	R/W 0x0	Timeout Period Writes have no effect when the configuration parameter WDT_HC_TOP = 1, thus making this register read-only. This field is used to select the timeout period from which the watchdog counter restarts. A change of the timeout period takes effect only after the next counter restart (kick). The range of values is limited by the WDT_CNT_WIDTH. If TOP is programmed to select a range that is greater than the counter width, the timeout period is truncated to fit to the counter width. This affects only the non-user specified values as users are limited to these boundaries during configuration. The range of values available for a 32-bit watchdog counter are: Where i = TOP and t = timeout period For i = 0 to 15 if WDT_USE_FIX_TOP==1 t = 2(16 + i) else t = WDT_USER_TOP_(i)

A.18.2.3 WDT_CCVR Register

Instance Name	Offset
WDT_CCVR	0x08

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

Table 574: WDT Current Counter Value Register (WDT_CCVR)

Bits	Field	Type/ HW Rst	Description
31:0	wdt_ccvr	R 0xFFFF	This register, when read, is the current value of the internal counter. This value is read coherently when ever it is read, which is relevant when the APB_DATA_WIDTH is less than the counter width.

A.18.2.4 WDT_CRR Register

Instance Name	Offset
WDT_CRR	0x0C

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

Table 575: WDT Counter Restart Register (WDT_CRR)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7:0	wdt_crr	W 0x0	This register is used to restart the WDT counter. As a safety feature to prevent accidental restarts, the value 0x76 must be written. A restart also clears the WDT interrupt. Reading this register returns zero.

A.18.2.5 WDT_STAT Register

Instance Name	Offset
WDT_STAT	0x10

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

Table 576: WDT Interrupt Status Register (WDT_STAT)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	wdt_stat	R 0x0	This register shows the interrupt status of the WDT. 0x0 = interrupt is inactive 0x1 = interrupt is active regardless of polarity

A.18.2.6 WDT_EOI Register

Instance Name	Offset
WDT_EOI	0x14

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

Table 577: WDT Interrupt Clear Register (WDT_EOI)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	wdt_eoi	R 0x0	Clears the watchdog interrupt. This can be used to clear the interrupt without restarting the watchdog counter.

A.18.2.7 WDT_COMP_PARAM_5 Register

Instance Name	Offset
WDT_COMP_PARAM_5	0xE4

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

Table 578: WDT Component Parameters Register 5 (WDT_COMP_PARAM_5)

Bits	Field	Type/ HW Rst	Description
31:0	data	R 0x0	Data

A.18.2.8 WDT_COMP_PARAM_4 Register

Instance Name	Offset
WDT_COMP_PARAM_4	0xE8

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

Table 579: WDT Component Parameters Register 4 (WDT_COMP_PARAM_4)

Bits	Field	Type/ HW Rst	Description
31:0	data	R 0x0	Data

A.18.2.9 WDT_COMP_PARAM_3 Register

Instance Name	Offset
WDT_COMP_PARAM_3	0xEC

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

Table 580: WDT Component Parameters Register 3 (WDT_COMP_PARAM_3)

Bits	Field	Type/ HW Rst	Description
31:0	data	R 0x0	Data

A.18.2.10 WDT_COMP_PARAM_2 Register

Instance Name	Offset
WDT_COMP_PARAM_2	0xF0

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

Table 581: WDT Component Parameters Register 2 (WDT_COMP_PARAM_2)

Bits	Field	Type/ HW Rst	Description
31:0	data	R 0xFFFF	Data

A.18.2.11 WDT_COMP_PARAM_1 Register

Instance Name	Offset
WDT_COMP_PARAM_1	0xF4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved			wdt_cnt_width					wdt_dflt_top_init				wdt_dflt_top				Reserved			wdt_dflt_rpl			apb_data_width		wdt_pause	wdt_use_fix_top	wdt_hc_top	wdt_hc_rpl	wdt_hc_rmod	wdt_dual_top	wdt_dflt_rmod	wdt_always_en
HW Rst	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 582: WDT Component Parameters Register 1 (WDT_COMP_PARAM_1)

Bits	Field	Type/ HW Rst	Description
31:29	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
28:24	wdt_cnt_width	R 0x0	Count Width
23:20	wdt_dflt_top_init	R 0x0	Dflt Top Init
19:16	wdt_dflt_top	R 0x0	Dflt Top
15:13	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
12:10	wdt_dflt_rpl	R 0x0	Dflt Rpl
9:8	apb_data_width	R 0x0	APB Data Width
7	wdt_pause	R 0x0	Pause
6	wdt_use_fix_top	R 0x0	Use Fix Top
5	wdt_hc_top	R 0x0	HC Top
4	wdt_hc_rpl	R 0x0	HC Rpl
3	wdt_hc_rmod	R 0x0	HC Rmod
2	wdt_dual_top	R 0x0	Dual Top
1	wdt_dflt_rmod	R 0x0	Dflt Rmod

Table 582: WDT Component Parameters Register 1 (WDT_COMP_PARAM_1) (Continued)

Bits	Field	Type/ HW Rst	Description
0	wdt_always_en	R 0x0	Always Enable

A.18.2.12 WDT_COMP_VERSION Register

Instance Name	Offset
WDT_COMP_VERSION	0xF8

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

Table 583: WDT Component Version Register (WDT_COMP_VERSION)

Bits	Field	Type/ HW Rst	Description
31:0	wdt_comp_version	R 0x3130_ 372A	ASCII value for each number in the version, followed by *. For example, 32_30_31_2A represents the version 2.01*. Reset Value: See the Releases table in the DW_apb_rtc Release Notes.

A.18.2.13 WDT_COMP_TYPE Register

Instance Name	Offset
WDT_COMP_TYPE	0xFC

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

Table 584: WDT Component Type Register (WDT_COMP_TYPE)

Bits	Field	Type/ HW Rst	Description
31:0	wdt_comp_type	R 0x4457_ 0120	Component Type

A.19 RTC Address Block

A.19.1 RTC Register Map

Table 585: RTC Register Map

Offset	Name	HW Rst	Description	Details
0x00	CNT_EN_REG	0x0000_0000	Counter Enable Register	Page: 761
0x20	INT_RAW_REG	0x0000_0000	Interrupt Raw Register	Page: 763
0x24	INT_REG	0x0000_0000	Interrupt Register	Page: 764
0x28	INT_MSK_REG	0x0001_8000	Interrupt Mask Register	Page: 765
0x40	CNT_CNTL_REG	0x0000_0000	Counter Control Register	Page: 766
0x50	CNT_VAL_REG	0x0000_0000	Counter Value Register	Page: 767
0x60	CNT_UPP_VAL_REG	0xFFFF_FFFF	Counter Upper Value Register	Page: 767
0x70	CNT_ALARM_VAL_REG	0xFFFF_FFFF	Counter Alarm Value Register	Page: 768
0x80	CLK_CNTL_REG	0x0000_0000	Clock Control Register	Page: 768

A.19.2 RTC Registers

A.19.2.1 Counter Enable Register (CNT_EN_REG)

Instance Name	Offset
CNT_EN_REG	0x00

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved													sts_resetn	cnt_rst_done	cnt_run	Reserved													cnt_reset	cnt_stop	cnt_start
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0

Table 586: Counter Enable Register (CNT_EN_REG)

Bits	Field	Type/ HW Rst	Description
31:19	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
18	sts_resetn	R 0x0	System Reset Status CPU must poll this bit for a 1 before accessing any other registers. 0x0 = indicates that the system reset is still asserted 0x1 = indicates that the system reset is deasserted

Table 586: Counter Enable Register (CNT_EN_REG) (Continued)

Bits	Field	Type/ HW Rst	Description
17	cnt_rst_done	R 0x0	Counter Reset Done Status Writing 1 to CNT_RESET will set this bit to 0 until the counter finishes resetting. 0x0 = indicates that the counter is still resetting 0x1 = indicates that the counter has been reset
16	cnt_run	R 0x0	Counter Enabled Status This bit can be polled to see when the counter is really enabled. 0x0 = counter is disabled 0x1 = counter is enabled
15:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2	cnt_reset	W 0x0	Counter Reset 0x0 = no action 0x1 = reset the counter (counter is reset to 0; channel output states are reset to 0; poll CNT_RST_DONE for 1 before writing to any other registers)
1	cnt_stop	W 0x0	Counter Disable 0x0 = no action 0x1 = disable the counter (poll CNT_RUN for 0 to confirm that the counter is disabled internally)
0	cnt_start	W 0x0	Counter Enable 0x0 = no action 0x1 = enable the counter (poll CNT_RUN for 1 to confirm that the counter is enabled internally)

A.19.2.2 Interrupt Raw Register (INT_RAW_REG)

Instance Name	Offset
INT_RAW_REG	0x20

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

Table 587: Interrupt Raw Register (INT_RAW_REG)

Bits	Field	Type/ HW Rst	Description
31:17	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
16	cnt_upp_int	R/W1CLR 0x0	Counter-Reach-Upper Interrupt Status If UPP_VAL is set to the maximum value, then this bit is equivalent to an overflow status. 0x0 = status cleared 0x1 = counter has reached UPP_VAL
15	cnt_alarm_int	R/W1CLR 0x0	Counter-Reach-Alarm Interrupt Status If ALARM_VAL is set to the maximum value, then this bit is equivalent to an overflow status. 0x0 = status cleared 0x1 = counter has reached ALARM_VAL
14:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.19.2.3 Interrupt Register (INT_REG)

INT_MSK_REG is combined with STS_REG to form this register. Masked bits will be 0 while unmasked bits will be the same value as the ones in STS_REG.

If any bits in this register is 1, an interrupt will be generated.

Instance Name	Offset
INT_REG	0x24

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

Table 588: Interrupt Register (INT_REG)

Bits	Field	Type/ HW Rst	Description
31:17	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
16	cnt_upp_intr	R 0x0	Masked Signal of CNT_UPP_INT
15	cnt_alarm_intr	R 0x0	Masked Signal of CNT_ALARM_INT
14:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.19.2.4 Interrupt Mask Register (INT_MSK_REG)

INT_MSK_REG is combined with STS_REG to form INT_REG. Masked bits will be 0 while unmasked bits will be the same value as the ones in STS_REG.

Instance Name	Offset
INT_MSK_REG	0x28

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

Table 589: Interrupt Mask Register (INT_MSK_REG)

Bits	Field	Type/ HW Rst	Description
31:17	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
16	cnt_upp_msk	R/W 0x1	CNT_UPP_INT Interrupt Mask 0x0 = do not mask CNT_UPP_INT 0x1 = mask interrupt CNT_UPP_INT
15	cnt_alarm_msk	R/W 0x1	CNT_ALARM_INT Interrupt Mask 0x0 = do not mask CNT_ALARM_INT 0x1 = mask interrupt CNT_ALARM_INT
14:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.19.2.5 Counter Control Register (CNT_CNTL_REG)

Instance Name	Offset
CNT_CNTL_REG	0x40

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

Table 590: Counter Control Register (CNT_CNTL_REG)

Bits	Field	Type/ HW Rst	Description
31:10	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
9:8	cnt_updt_mod	R/W 0x0	Counter Update Mode 0x0 = update off 0x1 = reserved 0x2 = Auto-update (use when counter clock is at least 5 times slower than the register bus clock) 0x3 = reserved
7:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4	cnt_dbg_act	R/W 0x0	Counter Debug Mode Action Mask 0x0 = in debug mode, stop the counters 0x1 = in debug mode, counters are not affected
3:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.19.2.6 Counter Value Register (CNT_VAL_REG)

Instance Name	Offset
CNT_VAL_REG	0x50

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

Table 591: Counter Value Register (CNT_VAL_REG)

Bits	Field	Type/ HW Rst	Description
31:0	cnt_val	R 0x0	Counter Value This register displays the current counter value. The update method for CNT_VAL is chosen in CNT_UPDT_MOD.

A.19.2.7 Counter Upper Value Register (CNT_UPP_VAL_REG)

Instance Name	Offset
CNT_UPP_VAL_REG	0x60

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

Table 592: Counter Upper Value Register (CNT_UPP_VAL_REG)

Bits	Field	Type/ HW Rst	Description
31:0	upp_val	R/W 0xFFFF_FFFF	Counter Upper Value Do not set to 0. The reset value is the maximum value of the counter, where all bits are 1. In the event that the counter reaches this value (counter-reach-upper), the counter will overflow to 0. Setting this value to all 1s is equivalent to a free running up-counter. Writing to this register will shadow it and start the internal shadow register update. The update finishes during the next counter-reach-upper event and will continue to update if it detects a new UPP_VAL. To guarantee an immediate update with the current UPP_VAL, write 1 to CNT_RESET.

A.19.2.8 Counter Alarm Value Register (CNT_ALARM_VAL_REG)

Instance Name	Offset
CNT_ALARM_VAL_REG	0x70

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

Table 593: Counter Alarm Value Register (CNT_ALARM_VAL_REG)

Bits	Field	Type/ HW Rst	Description
31:0	alarm_val	R/W 0xFFFF_FFFF	Counter Alarm Value Do not set to 0. The reset value is the maximum value of the counter, where all bits are 1. In the event that the counter reaches this value (counter-reach-upper), the counter will overflow to 0. Setting this value to all 1s is equivalent to a free running up-counter. Writing to this register will shadow it and start the internal shadow register update. The update finishes during the next counter-reach-upper event and will continue to update if it detects a new UPP_VAL. To guarantee an immediate update with the current UPP_VAL, write 1 to CNT_RESET.

A.19.2.9 Clock Control Register (CLK_CNTL_REG)

Instance Name	Offset
CLK_CNTL_REG	0x80

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

Table 594: Clock Control Register (CLK_CNTL_REG)

Bits	Field	Type/ HW Rst	Description
31:12	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
11:8	clk_div	R/W 0x0	Clock Divider The frequency of the divided clock (f_div) is calculated from the frequency of the counter clock (f_clk) using this formula: f_div = f_clk / (2 ^ CLK_DIV)
7:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.20 PMU Address Block

A.20.1 PMU Register Map

Table 595: PMU Register Map

Offset	Name	HW Rst	Description	Details
0x000	PWR_MODE	0x0000_0000	Power Mode Control Register	Page: 771
0x004	BOOT_JTAG	0x0000_000E	BOOT_JTAG Register	Page: 772
0x008	LAST_RST_CAUSE	0x0000_0000	Last Reset Cause Register	Page: 773
0x00C	LAST_RST_CLR	0x0000_0000	Last Reset Cause Clear Register	Page: 774
0x010	WAKE_SRC_CLR	0x0000_0000	Wake-up Source Clear Register	Page: 775
0x014	PWR_MODE_STATUS	0x0000_0000	Power Mode Status Register	Page: 776
0x018	CLK_SRC	0x0000_0001	Clock Source Selection Register	Page: 776
0x01C	WAKEUP_STATUS	0x0000_0000	Wake-up Status Register	Page: 777
0x020	PMIP_BRN_INT_SEL	0x0000_0000	PMIP Brown Interrupt Select	Page: 778
0x028	CLK_RDY	0x0000_0000	Clock Ready Register	Page: 778
0x02C	RC32M_CTRL	0x0000_0000	RC 32M Control Register	Page: 779
0x034	SFLL_CTRL1	0x0000_0060	SFLL Control Register 1	Page: 780
0x038	ANA_GRP_CTRL0	0x0000_0004	Analog Group Control Register	Page: 781
0x03C	SFLL_CTRL0	0x0210_7800	SFLL Control Register 2	Page: 781
0x040	PWR_CFG	0x0000_07F0	Power Configuration Register	Page: 782
0x044	PWR_STAT	0x0000_0000	Power Status Register	Page: 783
0x048	WF_OPT0	0x0000_0000	WF OPT Power-Saving Register 0	Page: 784
0x04C	WF_OPT1	0x0000_0000	WF OPT Power-Saving Register 1	Page: 784
0x054	PMIP_BRN_CFG	0x0000_0000	Brown-out Configuration Register	Page: 785
0x058	AUPLL_LOCK	0x0000_0000	AUPLL Lock Status Register	Page: 786
0x05C	ANA_GRP_CTRL1	0x0000_0024	BG Control Register	Page: 787
0x060	PMIP_PWR_CONFIG	0x0000_0003	Power Configuration Register	Page: 788
0x06C	PMIP_TEST	0x0000_0000	PMIP Test Register	Page: 789
0x078	AUPLL_CTRL0	0x0000_AAAB	Audio PLL Control Register	Page: 790
0x07C	PERI_CLK_EN	0x04FA_0F90	Peripheral Clock Enable Register	Page: 790
0x080	UART_FAST_CLK_DIV	0x0083_94E3	UART Fast Clock Div Register	Page: 792

Table 595: PMU Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x084	UART_SLOW_CLK_DIV	0x003D_0890	UART Slow Clock Div Register	Page: 793
0x088	UART_CLK_SEL	0x0000_0000	UART Clock Select Register	Page: 793
0x08C	MCU_CORE_CLK_DIV	0x0000_0001	MCU CORE Clock Divider Ratio Register	Page: 794
0x090	PERI0_CLK_DIV	0x0821_0842	Peripheral0 Clock Divider Ratio Register	Page: 795
0x094	PERI1_CLK_DIV	0x0020_1110	Peripheral1 Clock Divider Ratio Register	Page: 796
0x098	PERI2_CLK_DIV	0x0010_0001	Peripheral2 Clock Divider Ratio Register	Page: 796
0x09C	GAU_CLK_SEL	0x0000_0000	Select Signal for GAU MCLK Register	Page: 798
0x0A0	LOW_PWR_CTRL	0x0000_0002	Low-Power Control in PM3/PM4 Mode Register	Page: 799
0x0A4	IO_PAD_PWR_CFG	0x0009_F00F	I/O Pad Power Configuration Register	Page: 800
0x0A8	EXT_SEL_REG0	0x0000_0000	Extra Interrupt Select Register 0	Page: 802
0x0B0	AUPLL_CTRL1	0x2000_CC12	USB and Audio PLL Control Register	Page: 804
0x0B4	GAU_CTRL	0x0000_001F	GAU Control Register	Page: 805
0x0B8	RC32K_CTRL0	0x0000_0000	RC32k Control 0 Register	Page: 806
0x0BC	RC32K_CTRL1	0x0018_0000	RC32k Control 1 Register	Page: 807
0x0C0	XTAL32K_CTRL	0x0000_0204	XTAL32k Control Register	Page: 808
0x0C4	PMIP_CMP_CTRL	0x0000_0100	PMIP Comparator Control Register	Page: 809
0x0C8	PMIP_BRNDET_AV18	0x0001_28C3	PMIP Brown-out AV18 Register	Page: 810
0x0D0	PMIP_BRNDET_VBAT	0x0004_A328	PMIP Brown-out VBAT Register	Page: 811
0x0D4	PMIP_BRNDET_V12	0x0000_2510	PMIP Brown-out V12 Register	Page: 812
0x0D8	PMIP_LDO_CTRL	0x0000_0CB3	PMIP LDO Control Register	Page: 813
0x0DC	PERI_CLK_SRC	0x0000_0000	PERI Clock Source Register	Page: 814
0x0E0	PMIP_RSVD	0x0000_0000	Unused Register	Page: 815
0x0E4	GPT0_CTRL	0x0000_0001	GPT0 Control Register	Page: 815
0x0E8	GPT1_CTRL	0x0000_0001	GPT1 Control Register	Page: 816
0x0EC	GPT2_CTRL	0x0000_0001	GPT2 Control Register	Page: 817
0x0F0	GPT3_CTRL	0x0000_0001	GPT3 Control Register	Page: 818
0x0F4	WAKEUP_EDGE_ DETECT	0x0000_0003	Wake-up Edge Detect Register	Page: 818
0x0F8	AON_CLK_CTRL	0x0000_0012	AON Clock Control Register	Page: 819
0x0FC	PERI3_CTRL	0x0004_C304	PERI3 Control Register	Page: 820

Table 595: PMU Register Map (Continued)

Offset	Name	HW Rst	Description	Details
0x100 to 0x110	Reserved	0x0000_0000	Reserved	--
0x114	wakeup_mask	0x0000_0000	Wake-up Mask Interrupt Register	Page: 821
0x118	wlan_ctrl	0x0001_2000	WLAN Control Register	Page: 822
0x11C	wlan_ctrl1	0x0000_0000	WLAN Control 1 Register	Page: 823

A.20.2 PMU Registers

A.20.2.1 Power Mode Control Register (PWR_MODE)

Instance Name	Offset
PWR_MODE	0x000

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

Table 596: Power Mode Control Register (PWR_MODE)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1:0	pwr_mode	W 0x0	Power Mode Switch 0x0 = PM0 or PM1 0x1 = PM2 0x2 = PM3 0x3 = PM4

A.20.2.2 BOOT_JTAG Register (BOOT_JTAG)

Instance Name	Offset
BOOT_JTAG	0x004

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

Table 597: BOOT_JTAG Register (BOOT_JTAG)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	jtag_en	R/W 0x0	JTAG Enable 0x0 = disable JTAG 0x1 = enable JTAG

A.20.2.3 Last Reset Cause Register (LAST_RST_CAUSE)

Instance Name	Offset
LAST_RST_CAUSE	0x008

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																										wdt_rst	cm3_lockup	cm3_sysresetreq	brownout_av18	brownout_v12	brownout_vbat
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0

Table 598: Last Reset Cause Register (LAST_RST_CAUSE)

Bits	Field	Type/ HW Rst	Description
31:6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
5	wdt_rst	R 0x0	WDT Reset 0x0 = reset cause is not watchdog timer 0x1 = reset cause is watchdog timer
4	cm4_lockup	R 0x0	CM4 Lockup 0x0 = reset cause is not lockup 0x1 = reset cause is lockup
3	cm4_sysresetreq	R 0x0	CM4 System Software Reset Request 0x0 = reset cause is not system software reset request 0x1 = reset cause is system software reset request
2	brownout_av18	R 0x0	AV18 Power Brown-out 0x0 = AV18 power brown-out not detected 0x1 = AV18 power brown-out detected
1	brownout_v12	R 0x0	AV12 Power Brown-out 0x0 = AV12 power brown-out not detected 0x1 = AV12 power brown-out detected
0	brownout_vbat	R 0x0	VBAT Power Brown-out 0x0 = VBAT power brown-out not detected 0x1 = VBAT power brown-out detected

A.20.2.4 Last Reset Cause Clear Register (LAST_RST_CLR)

Instance Name	Offset
LAST_RST_CLR	0x00C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																										wdt_rst_clr	cm3_lockup_clr	cm3_sysresetreq_clr	brownout_av18_clr	brownout_v12_clr	brownout_vbat_clr
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0

Table 599: Last Reset Cause Clear Register (LAST_RST_CLR)

Bits	Field	Type/ HW Rst	Description
31:6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
5	wdt_rst_clr	R/W 0x0	Clear Watchdog Timer Reset Request
4	cm4_lockup_clr	R/W 0x0	Clear Lockup Request
3	cm4_sysresetreq_clr	R/W 0x0	Clear System Reset Request
2	brownout_av18_clr	R/W 0x0	Brown-out V18 Clear
1	brownout_v12_clr	R/W 0x0	Brown-out V12 Clear
0	brownout_vbat_clr	R/W 0x0	Brown-out VBAT Clear

A.20.2.5 Wake-up Source Clear Register (WAKE_SRC_CLR)

Instance Name	Offset
WAKE_SRC_CLR	0x010

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																											clr_comp_int	clr_rtc_int	clr_wl_int	clr_pin1_int	clr_pin0_int
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0

Table 600: Wake-up Source Clear Register (WAKE_SRC_CLR)

Bits	Field	Type/ HW Rst	Description
31:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4	clr_comp_int	R/W 0x0	Clear PMIP Comp Interrupt Request
3	clr_rtc_int	R/W 0x0	Clear RTC Interrupt Request
2	clr_wl_int	R/W 0x0	Clear WL Interrupt Request
1	clr_pin1_int	R/W 0x0	Clear Pin1 Interrupt Request
0	clr_pin0_int	R/W 0x0	Clear Pin0 Interrupt Request

A.20.2.6 Power Mode Status Register (PWR_MODE_STATUS)

Instance Name	Offset
PWR_MODE_STATUS	0x014

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

Table 601: Power Mode Status Register (PWR_MODE_STATUS)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1:0	pwr_mode_status	R 0x0	Power Mode Status Indicates which power mode the device will wake up from. 0x0 = reserved 0x1 = wake up from PM2 0x2 = wake up from PM3 0x3 = wake up from PM4

A.20.2.7 Clock Source Selection Register (CLK_SRC)

Instance Name	Offset
CLK_SRC	0x018

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

Table 602: Clock Source Selection Register (CLK_SRC)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1:0	sys_clk_sel	R/W 0x1	System Clock Select 0x0 = SFLL 200 MHz clock 0x1 = RC 32 MHz clock 0x2 = XTAL 32 MHz clock 0x3 = RC 32 MHz clock

A.20.2.8 Wake-up Status Register (WAKEUP_STATUS)

Instance Name	Offset
WAKEUP_STATUS	0x01C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																											pmip_comp_wakeup_status	rtc_wakeup_status	wlint_wakeup_status	pin1_wakeup_status	pin0_wakeup_status
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0

Table 603: Wake-up Status Register (WAKEUP_STATUS)

Bits	Field	Type/ HW Rst	Description
31:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4	pmip_comp_wakeup_ status	R 0x0	pmip_comp Wake-up Status
3	rtc_wakeup_status	R 0x0	RTC Wake-up Status
2	wlint_wakeup_status	R 0x0	WLAN Interrupt Wake-up Status
1	pin1_wakeup_status	R 0x0	External Pin1 Wake-up Status
0	pin0_wakeup_status	R 0x0	External Pin0 Wake-up Status

A.20.2.9 PMIP Brown Interrupt Select (PMIP_BRN_INT_SEL)

Instance Name	Offset
PMIP_BRN_INT_SEL	0x020

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

Table 604: PMIP Brown Interrupt Select (PMIP_BRN_INT_SEL)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	pmip_brn_int_sel	R/W 0x1	PMIP Brown-out Interrupt Select 0x0 = reset chip when VBAT brown-out 0x1 = generate interrupt when VBAT brown-out

A.20.2.10 Clock Ready Register (CLK_RDY)

Instance Name	Offset
CLK_RDY	0x028

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																									xtal32m_clk_rdy	Reserved	pll_audio_rdy	x32k_rdy	rc32m_rdy	Reserved	pll_clk_rdy
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	?	0	0	0	?	0

Table 605: Clock Ready Register (CLK_RDY)

Bits	Field	Type/ HW Rst	Description
31:7	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
6	xtal32m_clk_rdy	R 0x0	XTAL32M Clock Ready
5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

Table 605: Clock Ready Register (CLK_RDY) (Continued)

Bits	Field	Type/ HW Rst	Description
4	pll_audio_rdy	R 0x0	PLL Audio Ready 0x0 = PLL audio clock not ready for use 0x1 = PLL audio clock ready for use
3	x32k_rdy	R 0x0	XTAL 32k Ready 0x0 = XTAL 32k clock not ready for use 0x1 = XTAL 32k clock ready for use
2	rc32m_rdy	R 0x0	RC 32M Ready 0x0 = RC 32M clock not ready for use 0x1 = RC 32M clock ready for use
1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	pll_clk_rdy	R 0x0	PLL Clock Ready 0x0 = PLL clock not ready for use 0x1 = PLL clock ready for use

A.20.2.11 RC 32M Control Register (RC32M_CTRL)

Instance Name	Offset
RC32M_CTRL	0x02C

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

Table 606: RC 32M Control Register (RC32M_CTRL)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	cal_allow	R/W 0x0	Allow Calibration Command from PMU
0	cal_in_progress	R 0x0	Asserts High When Calibration in Progress

A.20.2.12 SFLL Control Register 1 (SFLL_CTRL1)

Instance Name	Offset
SFLL_CTRL1	0x034

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reg_pll_pu_int	sfll_reserve_in								Reserved		sfll_div_sel		Reserved							sfll_test_ana			sfll_refdiv								
HW Rst	0	0	0	0	0	0	0	0	0	?	?	0	0	?	?	?	?	?	?	?	0	0	0	0	0	1	1	0	0	0	0	0

Table 607: SFLL Control Register 1 (SFLL_CTRL1)

Bits	Field	Type/ HW Rst	Description
31	reg_pll_pu_int	R/W 0x0	PLL PU Int
30:23	sfll_reserve_in	R/W 0x0	SFLL Reserved Input
22:21	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
20:19	sfll_div_sel	R/W 0x0	Post Divider 0x0 = divide by 1 (bypass) 0x1 = divide by 2 0x2 = divide by 4 0x3 = divide by 8
18:12	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
11:9	sfll_test_ana	R/W 0x0	DC Points Testing Control
8:0	sfll_refdiv	R/W 0x60	Reference Clock Divider Select

A.20.2.13 Analog Group Control Register (ANA_GRP_CTRL0)

Instance Name	Offset
ANA_GRP_CTRL0	0x038

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																													pu	pu_xtal	pu_osc
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	0	0

Table 608: Analog Group Control Register (ANA_GRP_CTRL0)

Bits	Field	Type/ HW Rst	Description
31:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2	pu	R/W 0x1	Power-up Signal for Whole Block
1	pu_xtal	R/W 0x0	Power-up Signal for XTAL Circuit
0	pu_osc	R/W 0x0	Power-up Signal for OSC Circuit

A.20.2.14 SFLL Control Register 2 (SFLL_CTRL0)

Instance Name	Offset
SFLL_CTRL0	0x03C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved					sfll_lock	sfll_refclk_sel	Reserved			sfll_kvco		Reserved				sfll_fbdiv									Reserved							sfll_pu
HW Rst	?	?	?	?	?	0	1	?	?	?	0	1	?	?	?	?	0	1	1	1	1	0	0	0	0	?	?	?	?	?	?	0

Table 609: SFLL Control Register 2 (SFLL_CTRL0)

Bits	Field	Type/ HW Rst	Description
31:27	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
26	sfll_lock	R 0x0	SFLL Lock 0x0 = PLL module unlocked 0x1 = PLL module locked
25	sfll_refclk_sel	R/W 0x1	Reference Clock Source Select 0x0 = RC 32M 0x1 = XTAL 32M from WLAN

Table 609: SFLL Control Register 2 (SFLL_CTRL0) (Continued)

Bits	Field	Type/ HW Rst	Description
24:22	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
21:20	sfll_kvco	R/W 0x1	Select VCO Running Range Default Value for Output clock=200M 0x0 = 150 MHz to 177 MHz 0x1 = 177 MHz to 212 MHz 0x2 = 212 MHz to 240 MHz 0x3 = 240 MHz to 300 MHz
19:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:7	sfll_fbdiv	R/W 0xF0	Feedback Clock Divider Select
6:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	sfll_pu	R/W 0x0	Power-up Signal for the Flock 0x0 = power down 0x1 = power up

A.20.2.15 Power Configuration Register (PWR_CFG)

Instance Name	Offset
PWR_CFG	0x040

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																					pm3_ret_mem_cfg							Reserved			
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	1	1	1	1	1	1	?	?	?	?

Table 610: Power Configuration Register (PWR_CFG)

Bits	Field	Type/ HW Rst	Description
31:11	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
10:4	pm3_ret_mem_cfg	R/W 0x7F	Retention Memory Enable Register in PM3 Mode
3:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.20.2.16 Power Status Register (PWR_STAT)

Instance Name	Offset
PWR_STAT	0x044

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																								av18_rdy	Reserved					v12_ldo_rdy	Reserved
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	?	?	?	?	?	0	?

Table 611: Power Status Register (PWR_STAT)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7	av18_rdy	R 0x0	av18_rdy
6:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	v12_ldo_rdy	R 0x0	v12_ldo_rdy
0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.20.2.17 WF OPT Power-Saving Register 0 (WF_OPT0)

Instance Name	Offset
WF_OPT0	0x048

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

Table 612: WF OPT Power-Saving Register 0 (WF_OPT0)

Bits	Field	Type/ HW Rst	Description
31:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2	mem_ctrl	R/W 0x0	sram_memory Control 0x0 = normal mode 0x1 = limit SRAM to 192K
1	max_freq_ctrl	R/W 0x0	Maximum Frequency Control 0x0 = normal mode 0x1 = maximum frequency is 150 MHz
0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.20.2.18 WF OPT Power-Saving Register 1 (WF_OPT1)

Instance Name	Offset
WF_OPT1	0x04C

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

Table 613: WF OPT Power-Saving Register 1 (WF_OPT1)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3:2	spare	R/W 0x0	Spare

Table 613: WF OPT Power-Saving Register 1 (WF_OPT1) (Continued)

Bits	Field	Type/ HW Rst	Description
1:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.20.2.19 Brown-out Configuration Register (PMIP_BRN_CFG)

Instance Name	Offset
PMIP_BRN_CFG	0x054

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																												brndet_av18_rst_en	Reserved	brndet_vbat_rst_en	brndet_v12_rst_en
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	?	0	0

Table 614: Brown-out Configuration Register (PMIP_BRN_CFG)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3	brndet_av18_rst_en	R/W 0x0	Brown-out AV18 Reset Enable
2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	brndet_vbat_rst_en	R/W 0x0	Brown-out VBAT Reset Enable
0	brndet_v12_rst_en	R/W 0x0	Brown-out AV12 Reset Enable

A.20.2.20 AUPLL Lock Status Register (AUPLL_LOCK)

Instance Name	Offset
AUPLL_LOCK	0x058

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

Table 615: AUPLL Lock Status Register (AUPLL_LOCK)

Bits	Field	Type/ HW Rst	Description
31:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2	aupll_lock	R 0x0	AUPLL Lock
1:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.20.2.21 BG Control Register (ANA_GRP_CTRL1)

Instance Name	Offset
ANA_GRP_CTRL1	0x05C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																					bypass	test			bg_sel		r_orien_sel	gainx2	bg_ctrl		
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	1	0	0	1	0	0

Table 616: BG Control Register (ANA_GRP_CTRL1)

Bits	Field	Type/ HW Rst	Description
31:11	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
10	bypass	R/W 0x0	XTAL OSC Bypass Control Signal
9:7	test	R/W 0x0	Analog Test Control Bits
6:5	bg_sel	R/W 0x1	Selects the Bandgap Voltage
4	r_orien_sel	R/W 0x0	RPP Resister Orientation Selection
3	gainx2	R/W 0x0	OSC Gain Control
2:0	bg_ctrl	R/W 0x4	Bandgap Control

A.20.2.22 Power Configuration Register (PMIP_PWR_CONFIG)

Instance Name	Offset
PMIP_PWR_CONFIG	0x060

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																													av18_ext	status_del_sel	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	1	1

Table 617: Power Configuration Register (PMIP_PWR_CONFIG)

Bits	Field	Type/ HW Rst	Description
31:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2	av18_ext	R/W 0x0	AV18 External Assert high if external DC/DC chip will provide AV18=1.8V during PM0/1 modes.
1:0	status_del_sel	R/W 0x3	Control Counter in Delay for Rdy/rdy<0> Handshaking

A.20.2.23 PMIP Test Register (PMIP_TEST)

Instance Name	Offset
PMIP_TEST	0x06C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																						pmu_pmip_test_en	pmu_pmip_test				pmip_test_en	pmip_test			
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0

Table 618: PMIP Test Register (PMIP_TEST)

Bits	Field	Type/ HW Rst	Description
31:10	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
9	pmu_pmip_test_en	R/W 0x0	Enable Test Mux
8:5	pmu_pmip_test	R/W 0x0	Test Mux Input
4	pmip_test_en	R/W 0x0	PMIP Test Enable 0x0 = disable PMIP analog test mux 0x1 = enable PMIP analog test mux
3:0	pmip_test	R/W 0x0	Test Mux Output

A.20.2.24 Audio PLL Control Register (AUPLL_CTRL0)

Instance Name	Offset
AUPLL_CTRL0	0x078

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

Table 619: Audio PLL Control Register (AUPLL_CTRL0)

Bits	Field	Type/ HW Rst	Description
31:21	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
20	pu	R/W 0x0	Power-up Signal for the PLL
19:0	fract	R/W 0xAAAB	Fractional Part of PLL Feedback Divider

A.20.2.25 Peripheral Clock Enable Register (PERI_CLK_EN)

Instance Name	Offset
PERI_CLK_EN	0x07C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved	usbc_ahb_clk_en	Reserved		usbc_clk_en	Reserved	Reserved	Reserved	wdt_clk_en	gpt3_clk_en	gpt2_clk_en	Reserved	i2c1_clk_en	Reserved	ssp2_clk_en	uart3_clk_en	uart2_clk_en	Reserved			gpt1_clk_en	gpt0_clk_en	ssp1_clk_en	ssp0_clk_en	i2c0_clk_en	uart1_clk_en	uart0_clk_en	gpio_clk_en	Reserved	Reserved	Reserved	Reserved
HW Rst	0	0	?	?	0	?	0	0	1	1	1	1	1	?	1	0	0	?	?	?	1	1	1	1	1	0	0	1	0	0	0	0

Table 620: Peripheral Clock Enable Register (PERI_CLK_EN)

Bits	Field	Type/ HW Rst	Description
31	Reserved	RSVD 0x0	Reserved. Do not change the reset value.
30	usbc_ahb_clk_en	R/W 0x0	AHB USBC Clock Enable
29:28	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
27	usbc_clk_en	R/W 0x0	USBC Clock Enable
26	Reserved	RSVD 0x1	Reserved. Do not change the reset value.

Table 620: Peripheral Clock Enable Register (PERI_CLK_EN) (Continued)

Bits	Field	Type/ HW Rst	Description
25:24	Reserved	RSVD 0x0	Reserved. Do not change the reset value.
23	wdt_clk_en	R/W 0x1	WDT Clock Enable
22	gpt3_clk_en	R/W 0x1	GPT3 Clock Enable
21	gpt2_clk_en	R/W 0x1	GPT2 Clock Enable
20	Reserved	RSVD 0x1	Reserved. Do not change the reset value.
19	i2c1_clk_en	R/W 0x1	I2C1 Clock Enable
18	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
17	ssp2_clk_en	R/W 0x1	SSP2 Clock Enable
16	uart3_clk_en	R/W 0x0	UART3 Clock Enable
15	uart2_clk_en	R/W 0x0	UART2 Clock Enable
14:12	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
11	gpt1_clk_en	R/W 0x1	GPT1 Clock Enable
10	gpt0_clk_en	R/W 0x1	GPT0 Clock Enable
9	ssp1_clk_en	R/W 0x1	SSP1 Clock Enable
8	ssp0_clk_en	R/W 0x1	SSP0 Clock Enable
7	i2c0_clk_en	R/W 0x1	I2C0 Clock Enable
6	uart1_clk_en	R/W 0x0	UART1 Clock Enable
5	uart0_clk_en	R/W 0x0	UART0 Clock Enable
4	gpio_clk_en	R/W 0x1	GPIO Clock Enable

Table 620: Peripheral Clock Enable Register (PERI_CLK_EN) (Continued)

Bits	Field	Type/ HW Rst	Description
3:0	Reserved	RSVD 0x0	Reserved. Do not change the reset value.

A.20.2.26 UART Fast Clock Div Register (UART_FAST_CLK_DIV)

Instance Name	Offset
UART_FAST_CLK_DIV	0x080

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

Table 621: UART Fast Clock Div Register (UART_FAST_CLK_DIV)

Bits	Field	Type/ HW Rst	Description
31:24	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
23:11	nominator	R/W 0x1072	13-Bit Nominator for Fraction Divider
10:0	denominator	R/W 0x4E3	11-Bit Denominator for Fractional Divider

A.20.2.27 UART Slow Clock Div Register (UART_SLOW_CLK_DIV)

Instance Name	Offset
UART_SLOW_CLK_DIV	0x084

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

Table 622: UART Slow Clock Div Register (UART_SLOW_CLK_DIV)

Bits	Field	Type/ HW Rst	Description
31:24	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
23:11	nominator	R/W 0x7A1	13-Bit Nominator for Fraction Divider
10:0	denominator	R/W 0x90	11-Bit Denominator for Fractional Divider

A.20.2.28 UART Clock Select Register (UART_CLK_SEL)

Instance Name	Offset
UART_CLK_SEL	0x088

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																												uart3_clk_sel	uart2_clk_sel	uart1_clk_sel	uart0_clk_sel
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0

Table 623: UART Clock Select Register (UART_CLK_SEL)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3	uart3_clk_sel	R/W 0x0	UART3 APB1 UART Clock Select 0x0 = slow 0x1 = fast
2	uart2_clk_sel	R/W 0x0	UART2 APB1 UART Clock Select 0x0 = slow 0x1 = fast
1	uart1_clk_sel	R/W 0x0	UART1 APB0 UART Clock Select 0x0 = slow 0x1 = fast

Table 623: UART Clock Select Register (UART_CLK_SEL) (Continued)

Bits	Field	Type/ HW Rst	Description
0	uart0_clk_sel	R/W 0x0	UART0 APB0 UART Clock Select 0x0 = slow 0x1 = fast

A.20.2.29 MCU CORE Clock Divider Ratio Register (MCU_CORE_CLK_DIV)

Instance Name	Offset
MCU_CORE_CLK_DIV	0x08C

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

Table 624: MCU CORE Clock Divider Ratio Register (MCU_CORE_CLK_DIV)

Bits	Field	Type/ HW Rst	Description
31:6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
5:0	fclk_div	R/W 0x1	FCLK Divisor

A.20.2.30 Peripheral0 Clock Divider Ratio Register (PERI0_CLK_DIV)

Instance Name	Offset
PERI0_CLK_DIV	0x090

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved	Reserved					Reserved						Reserved				Reserved	ssp2_clk_div					ssp1_clk_div					ssp0_clk_div				
HW Rst	?	0	0	0	1	0	0	0	0	0	1	0	0	0	0	1	?	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0

Table 625: Peripheral0 Clock Divider Ratio Register (PERI0_CLK_DIV)

Bits	Field	Type/ HW Rst	Description
31	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
30:16	Reserved	RSVD 0x021	Reserved. Do not change the reset value.
15	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
14:10	ssp2_clk_div	R/W 0x2	SSP2 APB1 Clock Divisor
9:5	ssp1_clk_div	R/W 0x2	SSP1 APB0 Clock Divisor
4:0	ssp0_clk_div	R/W 0x2	SSP0 APB0 Clock Divisor

A.20.2.31 Peripheral1 Clock Divider Ratio Register (PERI1_CLK_DIV)

Instance Name	Offset
PERI1_CLK_DIV	0x094

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

Table 626: Peripheral1 Clock Divider Ratio Register (PERI1_CLK_DIV)

Bits	Field	Type/ HW Rst	Description
31:7	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
6:4	flash_clk_div	R/W 0x1	Flash QSPI Clock Divisor
3:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.20.2.32 Peripheral2 Clock Divider Ratio Register (PERI2_CLK_DIV)

Instance Name	Offset
PERI2_CLK_DIV	0x098

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved			wdt_clk_div_2_2	Reserved		wdt_clk_div_1_0		Reserved		i2c_clk_div		Reserved					gpt3_clk_div_5_3			Reserved	gpt3_clk_div_2_0			Reserved	wdt_clk_div_5_3			Reserved		gpt_sample_clk_div	
HW Rst	?	?	?	0	?	?	0	0	?	?	0	1	?	?	?	?	?	0	0	0	?	0	0	0	?	0	0	0	?	0	0	1

Table 627: Peripheral2 Clock Divider Ratio Register (PERI2_CLK_DIV)

Bits	Field	Type/ HW Rst	Description
31:29	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
28	wdt_clk_div_2_2	R/W 0x0	See bit[25:24]
27:26	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

Table 627: Peripheral2 Clock Divider Ratio Register (PERI2_CLK_DIV) (Continued)

Bits	Field	Type/ HW Rst	Description
25:24	wdt_clk_div_1_0	R/W 0x0	WDT Clock Divisor bit[6:4], bit[28], bit[25:24] combine to become a 6-bit WDT clock divisor. 0x0 = (divisor = 1) others = (divisor = 2^value)
23:22	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
21:20	i2c_clk_div	R/W 0x1	i2c Function Clock Divisor, Divisor = i2c_clk_div 0x0 = (divisor = 1) others = (divisor = i2c_clk_div[21:20])
19:15	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
14:12	gpt3_clk_div_5_3	R/W 0x0	See bit[10:8]
11	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
10:8	gpt3_clk_div_2_0	R/W 0x0	GPT3 Clock divisor[2:0] bit[14:12], bit[10:8] combine to become a 6-bit GPT3 clock divisor. 0x0 = (divisor = 1) others = (divisor = gpt3 clock divisor[5:0])
7	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
6:4	wdt_clk_div_5_3	R/W 0x0	See bit[25:24]
3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2:0	gpt_sample_clk_div	R/W 0x1	GPT Sample Clock Divisor

A.20.2.33 Select Signal for GAU MCLK Register (GAU_CLK_SEL)

Instance Name	Offset
GAU_CLK_SEL	0x09C

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

Table 628: Select Signal for GAU MCLK Register (GAU_CLK_SEL)

Bits	Field	Type/ HW Rst	Description
31:3	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
2	gau_sw_gate	R/W 0x0	Gate Signal for GAU MCLK 0x0 = enable 0x1 = disable
1:0	gau_clk_sel	R/W 0x0	Select Signal for GAU MCLK 0x0 = PLL clock 0x1 = RC32M clock 0x2 = XTAL 38.4 MHz clock 0x3 = AUPLL clock

A.20.2.34 Low-Power Control in PM3/PM4 Mode Register (LOW_PWR_CTRL)

Instance Name	Offset
LOW_PWR_CTRL	0x0A0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																											rc_osc_sel	slp_ctrl	Reserved	cache_line_flush	Reserved
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	?	1	?

Table 629: Low-Power Control in PM3/PM4 Mode Register (LOW_PWR_CTRL)

Bits	Field	Type/ HW Rst	Description
31:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4	rc_osc_sel	R/W 0x0	RC32k and XTAL32k Output Clock Selection
3	slp_ctrl	R/W 0x0	32k Output Clock Enable Signal 0x0 = PU enabled while PD disabled 0x1 = PU disabled while PD enabled
2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	cache_line_flush	R/W 0x1	Flushes the Cache
0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.20.2.35 I/O Pad Power Configuration Register (IO_PAD_PWR_CFG)

Instance Name	Offset
IO_PAD_PWR_CFG	0x0A4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved												gpio_aon_pdb	gpio_aon_v18	Reserved	Reserved	gpio3_low_vddb	gpio2_low_vddb	gpio1_low_vddb	gpio0_low_vddb	gpio3_v18	gpio2_v18	gpio1_v18	gpio0_v18	Reserved				gpio3_pdb	gpio2_pdb	gpio1_pdb	gpio0_pdb
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	1	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1

Table 630: I/O Pad Power Configuration Register (IO_PAD_PWR_CFG)

Bits	Field	Type/ HW Rst	Description
31:20	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
19	gpio_aon_pdb	R/W 0x1	GPIO AON Pad Group 0 = regulator operates in power-down mode 1 = regulator operates in normal mode
18	gpio_aon_v18	R/W 0x0	GPIO AON Pad Groups I/O Voltage Should be consistent with the actual I/O supply. [gpio_aon_v18,gpio_aon_v25] = [0, 0] IO 3.3V [gpio_aon_v18,gpio_aon_v25] = [0, 1] IO 2.5V [gpio_aon_v18,gpio_aon_v25] = [1, x] IO 1.8V
17	gpio_aon_v25	RW 0x0	See gpio_aon_v18
16	Reserved	RSVD 0x1	Reserved. Always write 0. Ignore read value.
15	gpio3_low_vddb	R/W 0x1	GPIO3 Pad Groups Power Configuration 0 = power off 1 = power on
14	gpio2_low_vddb	R/W 0x1	GPIO2 Pad Groups Power Configuration 0 = power off 1 = power on
13	gpio1_low_vddb	R/W 0x1	GPIO1 Pad Groups Power Configuration 0 = power off 1 = power on
12	gpio0_low_vddb	R/W 0x1	GPIO0 Pad Groups Power Configuration 0 = power off 1 = power on
11	gpio3_v18	R/W 0x0	GPIO3 Pad Groups I/O Voltage Should be consistent with real I/O supply. [gpio3_v18,gpio3_v25] = [0, 0] IO 3.3V [gpio3_v18,gpio3_v25] = [0, 1] IO 2.5V [gpio3_v18,gpio3_v25] = [1, x] IO 1.8V

Table 630: I/O Pad Power Configuration Register (IO_PAD_PWR_CFG) (Continued)

Bits	Field	Type/ HW Rst	Description
10	gpio2_v18	R/W 0x0	GPIO2 Pad Groups I/O Voltage Should be consistent with real I/O supply. [gpio2_v18,gpio2_v25] = [0, 0] IO 3.3V [gpio2_v18,gpio2_v25] = [0, 1] IO 2.5V [gpio2_v18,gpio2_v25] = [1, x] IO 1.8V
9	gpio1_v18	R/W 0x0	GPIO1 Pad Groups I/O Voltage Should be consistent with real I/O supply. [gpio1_v18,gpio1_v25] = [0, 0] IO 3.3V [gpio1_v18,gpio1_v25] = [0, 1] IO 2.5V [gpio1_v18,gpio1_v25] = [1, x] IO 1.8V
8	gpio0_v18	R/W 0x0	GPIO0 Pad Groups I/O Voltage Should be consistent with real I/O supply. [gpio0_v18,gpio0_v25] = [0, 0] IO 3.3V [gpio0_v18,gpio0_v25] = [0, 1] IO 2.5V [gpio0_v18,gpio0_v25] = [1, x] IO 1.8V
7	gpio3_v25	R/W 0x0	See gpio3_v18
6	gpio2_v25	R/W 0x0	See gpio2_v18
5	gpio1_v25	R/W 0x0	See gpio1_v18
4	gpio0_v25	R/W 0x0	See gpio0_v18
3	gpio3_pdb	R/W 0x1	GPIO3 Pad Group 0 = regulator operates in power-down mode 1 = regulator operates in normal mode
2	gpio2_pdb	R/W 0x1	GPIO2 Pad Group 0 = regulator operates in power-down mode 1 = regulator operates in normal mode
1	gpio1_pdb	R/W 0x1	GPIO1 Pad Group 0 = regulator operates in power-down mode 1 = regulator operates in normal mode
0	gpio0_pdb	R/W 0x1	GPIO0 Pad Group 0 = regulator operates in power-down mode 1 = regulator operates in normal mode

A.20.2.36 Extra Interrupt Select Register 0 (EXT_SEL_REG0)

Instance Name	Offset
EXT_SEL_REG0	0x0A8

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved							sel_58	sel_57	sel_56	sel_55	sel_54	sel_53	sel_52	sel_51	sel_50	sel_49	sel_48	sel_47	sel_46	sel_45	sel_44	sel_43	sel_42	sel_41	sel_40	sel_39	sel_38	sel_37	sel_36	sel_35	sel_34
HW Rst	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 631: Extra Interrupt Select Register 0 (EXT_SEL_REG0)

Bits	Field	Type/ HW Rst	Description
31:25	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
24	sel_58	R/W 0x0	Select Signal for Extra Interrupt 58 0x0 = from GPIO_49 0x1 = from GPIO_48
23	sel_57	R/W 0x0	Select Signal for Extra Interrupt 57 0x0 = from GPIO_47 0x1 = from GPIO_46
22	sel_56	R/W 0x0	Select Signal for Extra Interrupt 56 0x0 = from GPIO_45 0x1 = from GPIO_44
21	sel_55	R/W 0x0	Select Signal for Extra Interrupt 55 0x0 = from GPIO_43 0x1 = from GPIO_42
20	sel_54	R/W 0x0	Select Signal for Extra Interrupt 54 0x0 = from GPIO_41 0x1 = from GPIO_40
19	sel_53	R/W 0x0	Select Signal for Extra Interrupt 53 0x0 = from GPIO_39 0x1 = from GPIO_38
18	sel_52	R/W 0x0	Select Signal for Extra Interrupt 52 0x0 = from GPIO_37 0x1 = from GPIO_36
17	sel_51	R/W 0x0	Select Signal for Extra Interrupt 51 0x0 = from GPIO_35 0x1 = from GPIO_34
16	sel_50	R/W 0x0	Select Signal for Extra Interrupt 50 0x0 = from GPIO_34 0x1 = from GPIO_32

Table 631: Extra Interrupt Select Register 0 (EXT_SEL_REG0) (Continued)

Bits	Field	Type/ HW Rst	Description
15	sel_49	R/W 0x0	Select Signal for Extra Interrupt 49 0x0 = from GPIO_31 0x1 = from GPIO_30
14	sel_48	R/W 0x0	Select Signal for Extra Interrupt 48 0x0 = from GPIO_29 0x1 = from GPIO_28
13	sel_47	R/W 0x0	Select Signal for Extra Interrupt 47 0x0 = from GPIO_27 0x1 = from GPIO_26
12	sel_46	R/W 0x0	Select Signal for Extra Interrupt 46 0x0 = from GPIO_25 0x1 = from GPIO_24
11	sel_45	R/W 0x0	Select Signal for Extra Interrupt 45 0x0 = from GPIO_23 0x1 = from GPIO_22
10	sel_44	R/W 0x0	Select Signal for Extra Interrupt 44 0x0 = from GPIO_21 0x1 = from GPIO_20
9	sel_43	R/W 0x0	Select Signal for Extra Interrupt 43 0x0 = from GPIO_19 0x1 = from GPIO_18
8	sel_42	R/W 0x0	Select Signal for Extra Interrupt 42 0x0 = from GPIO_17 0x1 = from GPIO_16
7	sel_41	R/W 0x0	Select Signal for Extra Interrupt 41 0x0 = from GPIO_15 0x1 = from GPIO_14
6	sel_40	R/W 0x0	Select Signal for Extra Interrupt 40 0x0 = from GPIO_13 0x1 = from GPIO_12
5	sel_39	R/W 0x0	Select Signal for Extra Interrupt 39 0x0 = from GPIO_11 0x1 = from GPIO_10
4	sel_38	R/W 0x0	Select Signal for Extra Interrupt 38 0x0 = from GPIO_9 0x1 = from GPIO_8
3	sel_37	R/W 0x0	Select Signal for Extra Interrupt 37 0x0 = from GPIO_7 0x1 = from GPIO_6

Table 631: Extra Interrupt Select Register 0 (EXT_SEL_REG0) (Continued)

Bits	Field	Type/ HW Rst	Description
2	sel_36	R/W 0x0	Select Signal for Extra Interrupt 36 0x0 = from GPIO_5 0x1 = from GPIO_4
1	sel_35	R/W 0x0	Select Signal for Extra Interrupt 35 0x0 = from GPIO_3 0x1 = from GPIO_2
0	sel_34	R/W 0x0	Select Signal for Extra Interrupt 34 0x0 = from GPIO_1 0x1 = from GPIO_0

A.20.2.37 USB and Audio PLL Control Register (AUPLL_CTRL1)

Instance Name	Offset
AUPLL_CTRL1	0x0B0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved		en_vcox2	Reserved					dig_tstpnt		ana_tstpnt		div_fbcclk						div_mclk				div_oclk_modulo			div_oclk_pattern		ena_dither	icp		refclk_sel	pd_ovprot
HW Rst	?	?	1	?	?	?	?	?	0	0	0	0	0	0	0	0	1	1	0	0	1	1	0	0	0	0	0	1	0	0	1	0

Table 632: USB and Audio PLL Control Register (AUPLL_CTRL1)

Bits	Field	Type/ HW Rst	Description
31:30	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
29	en_vcox2	R/W 0x1	Enable or Disable VCOCLK_X2
28:24	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
23:22	dig_tstpnt	R/W 0x0	Digital Test Select
21:20	ana_tstpnt	R/W 0x0	Analog Test Select
19:14	div_fbcclk	R/W 0x3	FBC Divider
13:10	div_mclk	R/W 0x3	MCLK Divider

Table 632: USB and Audio PLL Control Register (AUPLL_CTRL1) (Continued)

Bits	Field	Type/ HW Rst	Description
9:7	div_oclk_modulo	R/W 0x0	Output Clock Divider Control
6:5	div_oclk_pattern	R/W 0x0	Output Clock Divider Control
4	ena_dither	R/W 0x1	ENA Dither
3:2	icp	R/W 0x0	Charge-Pump Current Control Bits
1	refclk_sel	R/W 0x1	Reference Clock Selection
0	pd_ovprot	R/W 0x0	Enable Over-Voltage Protection on VCO

A.20.2.38 GAU Control Register (GAU_CTRL)

Instance Name	Offset
GAU_CTRL	0x0B4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																											gau_bg_mclk_en	gau_gpadc0_mclk_en	Reserved	gau_gpdac_mclk_en	gau_acomp_mclk_en
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	1	1	1	1

Table 633: GAU Control Register (GAU_CTRL)

Bits	Field	Type/ HW Rst	Description
31:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4	gau_bg_mclk_en	R/W 0x1	gau_bg Module Main Clock Enable Signal 0x0 = disable 0x1 = enable
3	gau_gpadc0_mclk_en	R/W 0x1	gau_gpadc0 Module Main Clock Enable Signal 0x0 = disable 0x1 = enable
2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

Table 633: GAU Control Register (GAU_CTRL) (Continued)

Bits	Field	Type/ HW Rst	Description
1	gau_gpdac_mclk_en	R/W 0x1	gau_gpdac Module Main Clock Enable Signal 0x0 = disable 0x1 = enable
0	gau_acomp_mclk_en	R/W 0x1	gau_acomp Module Main Clock Enable Signal 0x0 = disable 0x1 = enable

A.20.2.39 RC32k Control 0 Register (RC32K_CTRL0)

Instance Name	Offset
RC32K_CTRL0	0x0B8

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																rc32k_pd	rc32k_cal_en	rc32k_code_fr_ext												
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 634: RC32k Control 0 Register (RC32K_CTRL0)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15	rc32k_pd	R/W 0x0	Power Down 32k Oscillator
14	rc32k_cal_en	R/W 0x0	Enable Calibration of 32k Oscillator
13:0	rc32k_code_fr_ext	R/W 0x0	External Code In for Frequency Setting

A.20.2.40 RC32k Control 1 Register (RC32K_CTRL1)

Instance Name	Offset
RC32K_CTRL1	0x0BC

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved								rc32k_ext_code_en	rc32k_refclk32k	rc32k_cal_div			rc32k_allow_cal	rc32k_code_fr_cal														rc32k_cal_inprogress	rc32k	rc32k_rdy	rc32k_cal_done
HW Rst	?	?	?	?	?	?	?	?	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 635: RC32k Control 1 Register (RC32K_CTRL1)

Bits	Field	Type/ HW Rst	Description
31:24	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
23	rc32k_ext_code_en	R/W 0x0	Allow External Code in to Go into the Ckt
22	rc32k_refclk32k	R/W 0x0	RC32k Reference Clock
21:19	rc32k_cal_div	R/W 0x3	Divider for the Clock Step During Calibration
18	rc32k_allow_cal	R/W 0x0	Allow Calibration to Be Performed (monitor System clock)
17:4	rc32k_code_fr_cal	R 0x0	After Calibration Hold Calibrated Code
3	rc32k_cal_inprogress	R 0x0	Asserts High When Calibration Is in Progress
2	rc32k	R 0x0	RC32k
1	rc32k_rdy	R 0x0	Asserts High When 32k Clock Is Ready upon Pwrup
0	rc32k_cal_done	R 0x0	Asserts High When Calibration Is Done

A.20.2.41 XTAL32k Control Register (XTAL32K_CTRL)

Instance Name	Offset
XTAL32K_CTRL	0x0C0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																	x32k_dly_sel		x32k_en	x32k_ext_osc_en	x32k_vddxo_cntl		Reserved		x32k_tmode		x32k_test_en	x32k_stup_assist		xclk32k	x32k_rdy
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	1	?	?	0	0	0	0	1	0	0

Table 636: XTAL32k Control Register (XTAL32K_CTRL)

Bits	Field	Type/ HW Rst	Description
31:15	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
14:13	x32k_dly_sel	R/W 0x0	32k Delay Select
12	x32k_en	R/W 0x0	Enable 32k Oscillator
11	x32k_ext_osc_en	R/W 0x0	Enable External Oscillator Mode for Outside Clock
10:9	x32k_vddxo_cntl	R/W 0x1	Control VDDXO Level
8:7	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
6:5	x32k_tmode	R/W 0x0	Test Mode Enabling for 32k Xtal Ckt
4	x32k_test_en	R/W 0x0	Test Enabling for 32k Xtal Ckt
3:2	x32k_stup_assist	R/W 0x1	Use Startup Assist Ckt for 32 kHz Xosc
1	xclk32k	R 0x0	xclk32k
0	x32k_rdy	R 0x0	Assert High When Ready

A.20.2.42 PMIP Comparator Control Register (PMIP_CMP_CTRL)

Instance Name	Offset
PMIP_CMP_CTRL	0x0C4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																						gau_ref_en	comp_hyst		comp_en	comp_diff_en		comp_ref_sel		comp_rdy	comp_out
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	1	0	0	0	0	0	0	0	0

Table 637: PMIP Comparator Control Register (PMIP_CMP_CTRL)

Bits	Field	Type/ HW Rst	Description
31:10	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
9	gau_ref_en	R/W 0x0	GAU Reference Enable
8:7	comp_hyst	R/W 0x2	Control of Comparator Hysteresis
6	comp_en	R/W 0x0	Enable AON Domain Comparator
5	comp_diff_en	R/W 0x0	Enable Differential Mode for AON Comparator
4:2	comp_ref_sel	R/W 0x0	Select Comparator Reference for Single-Ended Mode 0x0 = 0.2V 0x1 = 0.4V 0x2 = 0.6V 0x3 = 0.8V 0x4 = 1.0V 0x5 = 1.2V 0x6 = 1.4V 0x7 = 1.6V
1	comp_rdy	R 0x0	Ready to Use AON Domain Comparator
0	comp_out	R 0x0	Output of AON Domain Comparator

A.20.2.43 PMIP Brown-out AV18 Register (PMIP_BRNDET_AV18)

Instance Name	Offset
PMIP_BRNDET_AV18	0x0C8

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved												del_av18_hyst		brndet_av18_en	brntrig_av18_cntl			brnhyst_av18_cntl		brndet_av18_filt		brndet_av18_rdy	brndet_av18_out	Reserved							
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	1	0	0	1	0	1	0	0	0	1	1	0	0	0	0	1	1

Table 638: PMIP Brown-out AV18 Register (PMIP_BRNDET_AV18)

Bits	Field	Type/ HW Rst	Description
31:20	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
19:18	del_av18_hyst	R/W 0x0	Del av18 Hysteresis
17	brndet_av18_en	R/W 0x0	Enable av18 Brown-out Detector
16:14	brntrig_av18_cntl	R/W 0x4	Control Trigger Voltage of av18 Brndet
13:12	brnhyst_av18_cntl	R/W 0x2	Control of av18 Brown-out Detector Hysteresis
11:10	brndet_av18_filt	R/W 0x2	Select Filtering Level for av18 Pulse to av18 Brndet
9	brndet_av18_rdy	R 0x0	Assert High If av18 Brown-out Is Rdy --> out Can Be Taken
8	brndet_av18_out	R 0x0	Assert High If av18 Brown-out Happened
7:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.20.2.44 PMIP Brown-out VBAT Register (PMIP_BRNDET_VBAT)

Instance Name	Offset
PMIP_BRNDET_VBAT	0x0D0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved												brndet_vbat_en	brntrig_vbat_cntl			brnhyst_vbat_cntl		brndet_vbat_filt		brndet_vbat_rdy	brndet_vbat_out	Reserved									
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	0	1	0	0	1	0	1	0	0	0	1	1	0	0	1	0	1	0	0	0

Table 639: PMIP Brown-out VBAT Register (PMIP_BRNDET_VBAT)

Bits	Field	Type/ HW Rst	Description
31:20	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
19	brndet_vbat_en	R/W 0x0	Enable Vbat Brown-out Detector
18:16	brntrig_vbat_cntl	R/W 0x4	Control Trigger Voltage of Vbat Brndet
15:14	brnhyst_vbat_cntl	R/W 0x2	Control of Vbat Brown-out Detector Hysteresis
13:12	brndet_vbat_filt	R/W 0x2	Select Filtering Level for Vbat Pulse to Vbat Brndet
11	brndet_vbat_rdy	R 0x0	Assert High If Vbat Brown-out Is rdy--> out Can Be Taken
10	brndet_vbat_out	R 0x0	Assert High If Vbat Brown-out Happened
31:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.20.2.45 PMIP Brown-out V12 Register (PMIP_BRNDET_V12)

Instance Name	Offset
PMIP_BRNDET_V12	0x0D4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																	brndet_v12_en	brntrig_v12_cntl			brnhyst_v12_cntl		brndet_v12_filt		brndet_v12_rdy	brndet_v12_out	ldo_aon_v12_sel			ldo_aon_v12_hyst	
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	1	0	0	1	0	1	0	0	0	1	0	0	0	0

Table 640: PMIP Brown-out V12 Register (PMIP_BRNDET_V12)

Bits	Field	Type/ HW Rst	Description
31:15	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
14	brndet_v12_en	R/W 0x0	Enable v12 Brown-out Detector
13:11	brntrig_v12_cntl	R/W 0x4	Control Trigger Voltage of v12 Brndet
10:9	brnhyst_v12_cntl	R/W 0x2	Control of v12 Brown-out Detector Hysteresis
8:7	brndet_v12_filt	R/W 0x2	Select Filtering Level for v12 Pulse to v12 Brndet
6	brndet_v12_rdy	R 0x0	Assert High If v12 Brown-out Is rdy--> out Can Be Taken
5	brndet_v12_out	R 0x0	Assert High If v12 Brown-out Happened
4:2	ldo_aon_v12_sel	R/W 0x4	Select Output Voltage of ldo_aon_v12
1:0	ldo_aon_v12_hyst	R/W 0x0	Control of ldo_aon_v12 Hysteresis

A.20.2.46 PMIP LDO Control Register (PMIP_LDO_CTRL)

Instance Name	Offset
PMIP_LDO_CTRL	0x0D8

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																				ldo_av18_en	Reserved					ldo_v12_en	ldo_v12_vout_sel			Reserved		
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	1	1	0	0	1	0	1	1	0	0	1	1

Table 641: PMIP LDO Control Register (PMIP_LDO_CTRL)

Bits	Field	Type/ HW Rst	Description
31:12	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
11	ldo_av18_en	R/W 0x1	Enable ldo_av18
10:6	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
5	ldo_v12_en	R/W 0x1	Enable ldo_v12
4:2	ldo_v12_vout_sel	R/W 0x4	Select Output Voltage for v12
1:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.20.2.47 PERI Clock Source Register (PERI_CLK_SRC)

Instance Name	Offset
PERI_CLK_SRC	0x0DC

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																		Reserved	Reserved	Reserved								Reserved	ssp2_audio_sel	ssp1_audio_sel	ssp0_audio_sel
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	?	0	0	0

Table 642: PERI Clock Source Register (PERI_CLK_SRC)

Bits	Field	Type/ HW Rst	Description
31:14	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
13:3	Reserved	RSVD 0x0	Reserved. Do not change the reset value.
2	ssp2_audio_sel	R/W 0x0	SSP2 Audio Select 0 = divided by system clock 1 = audio PLL clock
1	ssp1_audio_sel	R/W 0x0	SSP1 Audio Select 0 = divided by system clock 1 = audio PLL clock
0	ssp0_audio_sel	R/W 0x0	SSP0 Audio Select 0 = divided by system clock 1 = audio PLL clock

A.20.2.48 Unused Register (PMIP_RSVD)

Instance Name	Offset
PMIP_RSVD	0x0E0

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

Table 643: Unused Register (PMIP_RSVD)

Bits	Field	Type/ HW Rst	Description
31:16	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
15:10	reserve_out	R 0x0	Unused. Do Not Change the Reset Value. This field exists, but is not connected to logic.
9:0	reserve_in	R/W 0x0	Unused. Do Not Change the Reset Value. This field exists, but is not connected to logic.

A.20.2.49 GPT0 Control Register (GPT0_CTRL)

Instance Name	Offset
GPT0_CTRL	0x0E4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																					gpt0_clk_sel0		gpt0_clk_sel1		gpt0_freq_change	gpt0_clk_div					
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	1

Table 644: GPT0 Control Register (GPT0_CTRL)

Bits	Field	Type/ HW Rst	Description
31:11	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
10:9	gpt0_clk_sel0	R/W 0x0	Select Signal for Mux Before Frequency Divisor
8:7	gpt0_clk_sel1	R/W 0x0	Select Signal for Mux After Frequency Divisor
6	gpt0_freq_change	R/W 0x0	Frequency Change Enable
5:0	gpt0_clk_div	R/W 0x1	Clock Divisor

A.20.2.50 GPT1 Control Register (GPT1_CTRL)

Instance Name	Offset
GPT1_CTRL	0x0E8

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																					gpt1_clk_sel0		gpt1_clk_sel1		gpt1_freq_change	gpt1_clk_div					
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	1

Table 645: GPT1 Control Register (GPT1_CTRL)

Bits	Field	Type/ HW Rst	Description
31:11	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
10:9	gpt1_clk_sel0	R/W 0x0	Select Signal for Mux Before Frequency Divisor
8:7	gpt1_clk_sel1	R/W 0x0	Select Signal for Mux After Frequency Divisor
6	gpt1_freq_change	R/W 0x0	Frequency Change Enable
5:0	gpt1_clk_div	R/W 0x1	Clock Divisor

A.20.2.51 GPT2 Control Register (GPT2_CTRL)

Instance Name	Offset
GPT2_CTRL	0x0EC

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																					gpt2_clk_sel0		gpt2_clk_sel1		gpt2_freq_change	gpt2_clk_div					
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	1

Table 646: GPT2 Control Register (GPT2_CTRL)

Bits	Field	Type/ HW Rst	Description
31:11	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
10:9	gpt2_clk_sel0	R/W 0x0	Select Signal for Mux Before Frequency Divisor
8:7	gpt2_clk_sel1	R/W 0x0	Select Signal for Mux After Frequency Divisor
6	gpt2_freq_change	R/W 0x0	Frequency Change Enable
5:0	gpt2_clk_div	R/W 0x1	Clock Divisor

A.20.2.52 GPT3 Control Register (GPT3_CTRL)

Instance Name	Offset
GPT3_CTRL	0x0F0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																					gpt3_clk_sel0		gpt3_clk_sel1		gpt3_freq_change	gpt3_clk_div					
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	0	0	0	0	1

Table 647: GPT3 Control Register (GPT3_CTRL)

Bits	Field	Type/ HW Rst	Description
31:11	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
10:9	gpt3_clk_sel0	R/W 0x0	Select Signal for Mux Before Frequency Divisor
8:7	gpt3_clk_sel1	R/W 0x0	Select Signal for Mux After Frequency Divisor
6	gpt3_freq_change	R/W 0x0	Frequency Change Enable
5:0	gpt3_clk_div	R/W 0x1	Clock Divisor

A.20.2.53 Wake-up Edge Detect Register (WAKEUP_EDGE_DETECT)

Instance Name	Offset
WAKEUP_EDGE_DETECT	0x0F4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																														wakeup1	wakeup0
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	1

Table 648: Wake-up Edge Detect Register (WAKEUP_EDGE_DETECT)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	wakeup1	R/W 0x1	Pin 1 Wakeup 0 = positive level wakeup 1 = negative level wakeup

Table 648: Wake-up Edge Detect Register (WAKEUP_EDGE_DETECT) (Continued)

Bits	Field	Type/ HW Rst	Description
0	wakeup0	R/W 0x1	Pin 0 Wakeup 0 = positive level wakeup 1 = negative level wakeup

A.20.2.54 AON Clock Control Register (AON_CLK_CTRL)

Instance Name	Offset
AON_CLK_CTRL	0x0F8

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																					apb1_clk_div		apb0_clk_div		dma_clk_gate_en	rtc_int_sel	rtc_clk_en	pmu_clk_div			
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	0	1	0	0	1	0

Table 649: AON Clock Control Register (AON_CLK_CTRL)

Bits	Field	Type/ HW Rst	Description
31:11	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
10:9	apb1_clk_div	R/W 0x0	APB1 Clock Divisor
8:7	apb0_clk_div	R/W 0x0	APB0 Clock Divisor
6	dma_clk_gate_en	R/W 0x0	DMA Clock Gate Enable
5	rtc_int_sel	R/W 0x0	RTC Interrupt Select
4	rtc_clk_en	R/W 0x1	RTC Clock Enable
3:0	pmu_clk_div	R/W 0x2	PMU Clock Divisor

A.20.2.55 PERI3 Control Register (PERI3_CTRL)

Instance Name	Offset
PERI3_CTRL	0x0FC

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved													rc32m_gate	rc32m_div					gau_div					Reserved	Reserved	Reserved	Reserved	Reserved	Reserved		
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	1	0	0	1	1	0	0	0	0	1	1	0	0	0	0	0	1	0	0

Table 650: PERI3 Control Register (PERI3_CTRL)

Bits	Field	Type/ HW Rst	Description
31:19	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
18	rc32m_gate	R/W 0x1	RC32M Reference Clock Gate
17:13	rc32m_div	R/W 0x6	RC32M Clock Div Ratio
12:8	gau_div	R/W 0x3	GAU Clock Div Ratio
7:0	Reserved	R/W 0x04	Reserved. Do not change the reset value.

A.20.2.56 Wake-up Mask Interrupt Register (wakeup_mask)

Instance Name	Offset
wakeup_mask	0x114

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																								wl_wakeup_mask	pmip_comp_wakeup_mask	rtc_wakeup_mask	pin1_wakeup_mask	pin0_wakeup_mask	Reserved		
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0	?	?	?

Table 651: Wake-up Mask Interrupt Register (wakeup_mask)

Bits	Field	Type/ HW Rst	Description
31:8	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
7	wl_wakeup_mask	R/W 0x0	WLAN Wake-up Mask 0x0 = mask WLAN wake-up interrupt 0x1 = unmask WLAN wake-up interrupt
6	pmip_comp_wakeup_ mask	R/W 0x0	PMIP Comparator Wake-up Mask 0x0 = mask PMIP comparator wake-up interrupt 0x1 = unmask PMIP comparator wake-up interrupt
5	rtc_wakeup_mask	R/W 0x0	RTC Wake-up Mask 0x0 = mask RTC wake-up interrupt 0x1 = unmask RTC wake-up interrupt
4	pin1_wakeup_mask	R/W 0x0	Pin1 Wake-up Mask 0x0 = mask pin1 wake-up interrupt 0x1 = unmask pin1 wake-up interrupt
3	pin0_wakeup_mask	R/W 0x0	Pin0 Wake-up Mask 0x0 = mask pin0 wake-up interrupt 0x1 = unmask pin0 wake-up interrupt
2:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

A.20.2.57 WLAN Control Register (wlan_ctrl)

Instance Name	Offset
wlan_ctrl	0x118

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														wl_pd_del_cfg											refclk_usb_rdy	refclk_aud_rdy	refclk_sys_rdy	refclk_usb_req	refclk_aud_req	refclk_sys_req	pd
HW Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 652: WLAN Control Register (wlan_ctrl)

Bits	Field	Type/ HW Rst	Description
31:18	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
17:7	wl_pd_del_cfg	R/W 0x3C0	Count for 30 ms
6	refclk_usb_rdy	R 0x0	WLAN Reference Clock Ready 0x0 = not ready 0x1 = ready
5	refclk_aud_rdy	R 0x0	WLAN Reference Clock Ready 0x0 = not ready 0x1 = ready
4	refclk_sys_rdy	R 0x0	WLAN Reference Clock Ready 0x0 = not ready 0x1 = ready
3	refclk_usb_req	R/W 0x0	WLAN USB Reference Clock Request 0x0 = not request 0x1 = request
2	refclk_aud_req	R/W 0x0	WLAN AUD Reference Clock Request 0x0 = not request 0x1 = request
1	refclk_sys_req	R/W 0x0	WLAN SYS Reference Clock Request 0x0 = not request 0x1 = request
0	pd	R/W 0x0	WLAN Power down Function 0x0 = power down 0x1 = power on

A.20.2.58 WLAN Control 1 Register (wlan_ctrl1)

Instance Name	Offset
wlan_ctrl1	0x11C

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

Table 653: WLAN Control 1 Register (wlan_ctrl1)

Bits	Field	Type/ HW Rst	Description
31:12	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
11	mci_wl_wakeup	R/W 0x0	MCI_WL_WAKEUP Send wakeup signal to WLAN from MCI.
10:0	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.

THIS PAGE INTENTIONALLY LEFT BLANK

A.21 System Control Address Block

A.21.1 System Control Register Map

Table 654: System Control Register Map

Offset	Name	HW Rst	Description	Details
0x00	REV_ID	0x8813_0A41	Chip Revision Register	Page: 825
0x04	Reserved	0x0000_0000	Reserved	--
0x08	RAM0	0x0000_0006	RAM0 Control Register	Page: 826
0x0C	RAM1	0x0000_0006	RAM1 Control Register	Page: 826
0x10	RAM2	0x0000_0006	RAM2 Control Register	Page: 827
0x14	RAM3	0x0000_0006	RAM3 Control Register	Page: 827
0x28	ROM	0x0000_0016	ROM Control Register	Page: 828
0x2C	AON_MEM	0x0000_0006	AON_MEM Control Register	Page: 828
0x34	GPT_in	0x0000_0000	GPT Pin-in Selection Register	Page: 829
0x38	CAL	0x0000_0001	Calibration Channel Selection Register	Page: 829
0x3C	PERI_SW_RST	0x05FF_FFFF	Peripheral Software Reset Register	Page: 830
0x40	USB_CTRL	0x0002_B300	USB Control Register	Page: 832
0x4C	Reserved	0x0000_0006	Reserved	--

A.21.2 System Control Registers

A.21.2.1 Chip Revision Register (REV_ID)

Instance Name	Offset
REV_ID	0x00

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

Table 655: Chip Revision Register (REV_ID)

Bits	Field	Type/ HW Rst	Description
31:0	rev_id	R 0x8813_ 0A41	Chip Revision ID

A.21.2.2 RAM0 Control Register (RAM0)

Instance Name	Offset
RAM0	0x08

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

Table 656: RAM0 Control Register (RAM0)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3:2	wtc	R/W 0x1	WTC
1:0	rtc	R/W 0x2	RTC

A.21.2.3 RAM1 Control Register (RAM1)

Instance Name	Offset
RAM1	0x0C

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

Table 657: RAM1 Control Register (RAM1)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3:2	wtc	R/W 0x1	WTC
1:0	rtc	R/W 0x2	RTC

A.21.2.4 RAM2 Control Register (RAM2)

Instance Name	Offset
RAM2	0x10

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

Table 658: RAM2 Control Register (RAM2)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3:2	wtc	R/W 0x1	WTC
1:0	rtc	R/W 0x2	RTC

A.21.2.5 RAM3 Control Register (RAM3)

Instance Name	Offset
RAM3	0x14

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

Table 659: RAM3 Control Register (RAM3)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3:2	wtc	R/W 0x1	WTC
1:0	rtc	R/W 0x2	RTC

A.21.2.6 ROM Control Register (ROM)

Instance Name	Offset
ROM	0x28

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

Table 660: ROM Control Register (ROM)

Bits	Field	Type/ HW Rst	Description
31:5	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
4:3	rtc_ref	R/W 0x2	RTC Reference
2:0	rtc	R/W 0x6	RTC Reference

A.21.2.7 AON_MEM Control Register (AON_MEM)

Instance Name	Offset
AON_MEM	0x2C

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

Table 661: AON_MEM Control Register (AON_MEM)

Bits	Field	Type/ HW Rst	Description
31:4	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
3:2	wtc	R/W 0x1	WTC
1:0	rtc	R/W 0x2	RTC

A.21.2.8 GPT Pin-in Selection Register (GPT_in)

Instance Name	Offset
GPT_in	0x34

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

Table 662: GPT Pin-in Selection Register (GPT_in)

Bits	Field	Type/ HW Rst	Description
31:1	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
0	sel	R/W 0x0	Select GPT Pin

A.21.2.9 Calibration Channel Selection Register (CAL)

Instance Name	Offset
CAL	0x38

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

Table 663: Calibration Channel Selection Register (CAL)

Bits	Field	Type/ HW Rst	Description
31:2	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
1	rtc_trig	R/W 0x0	RTC Trigger
0	rtc_duty	R/W 0x1	RTC Duty

A.21.2.10 Peripheral Software Reset Register (PERI_SW_RST)

Instance Name	Offset
PERI_SW_RST	0x3C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved					gau_rstn_en	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	flash_qspi_rstn_en	uart0_rstn_en	Reserved	uart2_rstn_en	uart3_rstn_en	i2c0_rstn_en	i2c1_rstn_en	Reserved	ssp0_rstn_en	ssp1_rstn_en	ssp2_rstn_en	gpt0_rstn_en	gpt1_rstn_en	gpt2_rstn_en	gpt3_rstn_en	Reserved	Reserved	usb_rstn_en	wdt_rstn_en
HW Rst	?	?	?	?	?	1	?	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 664: Peripheral Software Reset Register (PERI_SW_RST)

Bits	Field	Type/ HW Rst	Description
31:27	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
26	gau_rstn_en	R/W 0x1	GAU Reset_n Enable
25	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
24:19	Reserved	RSVD 0x7	Reserved. Do not change the reset value.
18	flash_qspi_rstn_en	R/W 0x1	Flash QSPI Reset_n Enable
17	uart0_rstn_en	R/W 0x1	UART0 Reset_n Enable
16	Reserved	RSVD 0x1	Reserved. Do not change the reset value.
15	uart2_rstn_en	R/W 0x1	UART2 Reset_n Enable
14	uart3_rstn_en	R/W 0x1	UART3 Reset_n Enable
13	i2c0_rstn_en	R/W 0x1	I2C0 Reset_n Enable
12	i2c1_rstn_en	R/W 0x1	I2C1 Reset_n Enable
11	Reserved	RSVD 0x1	Reserved. Do not change the reset value.
10	ssp0_rstn_en	R/W 0x1	SSP0 Reset_n Enable

Table 664: Peripheral Software Reset Register (PERI_SW_RST) (Continued)

Bits	Field	Type/ HW Rst	Description
9	ssp1_rstn_en	R/W 0x1	SSP01 Reset_n Enable
8	ssp2_rstn_en	R/W 0x1	SSP2 Reset_n Enable
7	gpt0_rstn_en	R/W 0x1	GPT0 Reset_n Enable
6	gpt1_rstn_en	R/W 0x1	GPT1 Reset_n Enable
5	gpt2_rstn_en	R/W 0x1	GPT2 Reset_n Enable
4	gpt3_rstn_en	R/W 0x1	GPT3 Reset_n Enable
3	Reserved	RSVD 0x1	Reserved. Do not change the reset value.
2	Reserved	RSVD 0x1	Reserved. Do not change the reset value.
1	usb_rstn_en	R/W 0x1	USB Reset_n Enable
0	wdt_rstn_en	R/W 0x1	WDT Reset_n Enable

A.21.2.11 USB Control Register (USB_CTRL)

Instance Name	Offset
USB_CTRL	0x40

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved									mac_ctrl_sel	soft_utmi_iddig	soft_utmi_xvalid	soft_utmi_sessend	phy_reset_sel	soft_phy_reset	iddq_test	usb_resume	reg_tx_buf_wtc		reg_tx_buf_rtc		reg_rx_buf_wtc		reg_rx_buf_rtc		reg_tx_pdlvmc	reg_tx_pdfvssm	reg_rx_pdlvmc	reg_rx_pdfvssm	usb_pu	usb_pu_otg	usb_pu_pll
HW Rst	?	?	?	?	?	?	?	?	?	0	0	0	0	0	1	0	1	0	1	1	0	0	1	1	0	0	0	0	0	0	0	0

Table 665: USB Control Register (USB_CTRL)

Bits	Field	Type/ HW Rst	Description
31:23	Reserved	RSVD --	Reserved. Always write 0. Ignore read value.
22	mac_ctrl_sel	R/W 0x0	MAC Control Select
21	soft_utmi_iddig	R/W 0x0	Soft UTMI Iddig
20	soft_utmi_xvalid	R/W 0x0	Soft UTMI Xvalid
19	soft_utmi_sessend	R/W 0x0	Soft UTMI Sessend
18	phy_reset_sel	R/W 0x0	PHY Reset Select
17	soft_phy_reset	R/W 0x1	Soft PHY Reset
16	iddq_test	R/W 0x0	Iddq Test
15	usb_resume	R 0x1	USB Resume
14:13	reg_tx_buf_wtc	R/W 0x1	reg_tx_buf_wtc
12:11	reg_tx_buf_rtc	R/W 0x2	reg_tx_buf_rtc
10:9	reg_rx_buf_wtc	R/W 0x1	reg_rx_buf_wtc
8:7	reg_rx_buf_rtc	R/W 0x2	reg_rx_buf_rtc

Table 665: USB Control Register (USB_CTRL) (Continued)

Bits	Field	Type/ HW Rst	Description
6	reg_tx_pdlvmc	R/W 0x0	reg_tx_pdlvmc
5	reg_tx_pdfvssm	R/W 0x0	reg_tx_pdfvssm
4	reg_rx_pdlvmc	R/W 0x0	reg_rx_pdlvmc
3	reg_rx_pdfvssm	R/W 0x0	reg_rx_pdfvssm
2	usb_pu	R/W 0x0	USB PU
1	usb_pu_otg	R/W 0x0	USB PU OTG
0	usb_pu_pll	R/W 0x0	USB PU PLL

THIS PAGE INTENTIONALLY LEFT BLANK

B Acronyms and Abbreviation

Table 666: Acronyms and Abbreviations

Acronym	Definition
ACK	Acknowledgement
ACOMP	Analog Comparator
ADC	Analog-to-Digital Converter
AES	Advanced Encryption Standard
AHB	Advanced High-performance Bus
AMUX	Analog Multiplexer
APB	Advanced Peripheral Bus
AON	Always ON
ARM	Advanced RISC Machine
CBC	Cipher Block Chaining
CCM	Carrier Controlled Modulation
CCM	Continuity Check Message
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CTR	Counter (encryption mode)
DAC	Digital-to-Analog Converter
DMA	Direct Memory Access
DMAC	Direct Memory Access Controller
ECB	Electronic Code Book
EHCI	Enhanced Host Controller Interface
FIFO	First In First Out
GAU	General Analog Unit
GPIO	General Purpose Input/Output
GPT	General Purpose Timer
I/O	Input/Output
I/F	Interface
IoT	Internet of Things
JTAG	Joint Test Action Group
LQFN	Low Quad Flat Non-leaded
LSb	Least Significant bit
LSB	Least Significant Byte
MIC	Message Integrity Code

Table 666: Acronyms and Abbreviations (Continued)

Acronym	Definition
MII	Media Independent Interface
MIMO	Multiple Input Multiple Output
MMO	Matyas-Meyer-Oseas (encryption algorithm)
MPU	Memory Protection Unit
MSb	Most Significant bit
MSB	Most Significant Byte
M2M	Machine to Machine
NACK	Negative Acknowledgement
NVIC	Nested Vectored Interrupt Controller
OFDM	Orthogonal Frequency Division Multiplexing
OTG	On-The-Go
OTP	One Time Programmable
PGA	Programmable Gain Amplifier
PHY	Physical Layer
PLL	Phase-Locked Loop
PMU	Power Management Unit
POR	Power-On Reset
POS	Point of Sale
PSP	Programmable Serial Protocol
PWM	Pulse Width Modulation
QFN	Quad Flat Non-leaded Package
QSPI	Quad Serial Peripheral Interface
RAM	Random Access Memory
ROM	Read Only Memory
RTC	Real Time Clock
SCL	Serial Interface Clock
SDA	Serial Interface Data
SoC	System-on-Chip
SPH	Serial Clock Phase or Clock Out Phase Control
SPI	Serial Peripheral Interface
SRAM	Static Random Access Memory
SSP	Synchronous Serial Protocol
SWD	Serial Wire Debug
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus
UTMI	USB 2.0 Transceiver Macrocell Interface
VBAT	Voltage of Battery (Battery Voltage)

Table 666: Acronyms and Abbreviations (Continued)

Acronym	Definition
VCO	Voltage Controlled Oscillator
WDT	Watchdog Timer
WLAN	Wireless Local Area Network
XIP	eXecute In Place
XOSC	Crystal Oscillator
XTAL	Crystal

THIS PAGE INTENTIONALLY LEFT BLANK

C Revision History

Table 667: Revision History

Document Type	Document Revision
Release	Rev. C

Product Overview
- Figure 1, Block Diagram, on page 4: changed microcontroller LDO12 to LDO11

Package
- Table 5, USB 2.0 OTG Interface, on page 45: changed USB_VBUS to input/output
- Table 12, Clock/Control Interface, on page 61: added description for RESETn
- Table 16, Power and Ground, on page 64: changed BUCK18_VBAT_IN from 2.7V to 2.4V and VBAT from 2.0V to 1.84V

Core and System Control
- Table 19, RAM Blocks, on page 72: added

Power, Reset, and Clock Control
- Section 3.2.1.2, Power-up Requirements, on page 80: updated
- Figure 9, Power Supply Blocks, on page 79: added
- Figure 10, Power Option, on page 80: updated
- Figure 11, Power-Up Sequence, on page 81: updated
- Figure 13, High-Level Clocking Diagram, on page 94: moved APB0/1 to its own block

Boot ROM
- Figure 16, Boot ROM Flow, QSPI Loading, on page 108: updated Init QSPI for Flash reading
- Figure 17, Boot ROM Flow, Boot Mode Confirmed to be QSPI Loading, on page 109: added notes
- Figure 20, Boot ROM Flow, UART Loading, on page 111: updated; added notes
- Section 4.7.1, Code Image, on page 114: added “When Boot ROM starts upon reset...”
- Figure 25, Security/Erase and Acknowledgment Packets, on page 126: updated password to 32 bytes
- Section 4.12.3, Sample of Code Loading Through UART, on page 135: updated #3

GPIO
- Section 6.2, I/O Configuration, on page 143: added 50 kohm to PU/PD
- Section 6.3.2, GPIO Ports, on page 161: changed GPIO_PORT1 to 18 pins

UART
- Table 129, Pad Interface Signals, on page 214: removed UART_DCDn, UART_DSRn, UART_RIn, and UART_DTRn
- Figure 55, UART Block Diagram, on page 216: removed uart_dcd_n, uart_dsr_n, uart_ri_n, and uart_dtr_n
- Table 136, Modem I/O Signals in DTE Mode, on page 226: removed UART_DCDn, UART_DSRn, UART_RIn, and UART_DTRn

DAC
- Section 20.4.1, Configuration, on page 302: added 5th bullet with Table 156, Clock Divisor, on page 302

Electricals
- Table 158, Absolute Maximum Ratings, on page 315: updated description for VDDIO_0 to AON; added GPIO V_{IL} and GPIO V_{IH}
- Table 159, Recommended Operating Conditions, on page 316: changed VBAT_IN minimum value from 2.0V to 1.84V
- Table 159, Recommended Operating Conditions, on page 316: changed BUCK18_VBAT_IN minimum value from 2.7V to 2.4V
- Table 160, I/O Static Ratings, 1.8V VDDIO, on page 317: updated IOL, IOH

Table 667: Revision History (Continued)

Document Type	Document Revision
	• Table 161, I/O Static Ratings, 2.5V VDDIO, on page 318: updated IOL, IOH • Table 162, I/O Static Ratings, 3.3V VDDIO, on page 318: updated IOL, IOH • Table 163, WLAN Tx/Rx Current Consumption, on page 319: added • Table 164, WLAN Estimated Tx PA Power vs. Current Consumption, on page 320: added • Table 165, MCI VBAT Consumption, on page 321: added • Table 166, LDO11 Specifications, on page 322: added • Table 167, BUCK18 Specifications, on page 322: added • Table 168, Efficiency vs. BUCK18_VBAT_IN @ 147 mA Output, on page 323: added • Figure 112, BUCK18_VBAT_IN = 2.7V Efficiency vs. Current, on page 323: added • Table NOTE:, LDO11 Specifications, on page 323: added • Table NOTE:, LDO18 Specifications, on page 324: added • Table 177, VBAT BOD Timing Data, on page 329: added VBAT brown-out detector turn-on time • Table 188, LNA and Rx RF Mixer Specifications—802.11n/g/b, on page 344: added adjacent channel rejection, maximum input level, and maximum receive sensitivity for 802.11n/g/b • Table 189, Tx Mode Specifications—802.11n/g/b, on page 345: added Tx power with mask and EVM compliance to IEEE limits (normal and low power modes) ***Registers*** • Table A.9.2.2, SSP Control Register 1 (SSCR1), on page 616: reserved Bit[17] • Table A.10.2.9, Modem Control Register (MCR), on page 638: reserved Bits[3:2] and [0] • Table A.10.2.11, Modem Status Register (MSR), on page 640: reserved Bits[7:5] and [3:1] • Table A.11.2, GPIO Registers, on page 647: expanded GPIO register 0 group to 32 bits • Table A.17.2, PINMUX Registers (_GPIO), on page 749: changed Bit[15] to “pulled up by default 50 kohm internal pull-up” • Table A.20.2.7, Clock Source Selection Register (CLK_SRC), on page 776[1:0]: updated description • Table A.20.2.25, Peripheral Clock Enable Register (PERI_CLK_EN), on page 790: reserved Bit[3] • Table A.20.2.30, Peripheral0 Clock Divider Ratio Register (PERI0_CLK_DIV), on page 795: reserved Bits[30:26] • Table A.20.2.36, Extra Interrupt Select Register 0 (EXT_SEL_REG0), on page 802: corrected bit order for all bits (were reversed) • Table A.20.2.47, PERI Clock Source Register (PERI_CLK_SRC), on page 814: reserved Bit[12] • Table A.20.2.53, Wake-up Edge Detect Register (WAKEUP_EDGE_DETECT), on page 818: updated description • Table A.21.2.10, Peripheral Software Reset Register (PERI_SW_RST), on page 830: reserved Bit[22]

THIS PAGE INTENTIONALLY LEFT BLANK

Marvell Technology Group
http://www.marvell.com

Marvell. Moving Forward Faster