UR0431A Chip Specification

Project Name	
Functions	
Project Code	
Author	
Date	2013-04-12
Note	

Contents

1 Introduction

1.1 Overview

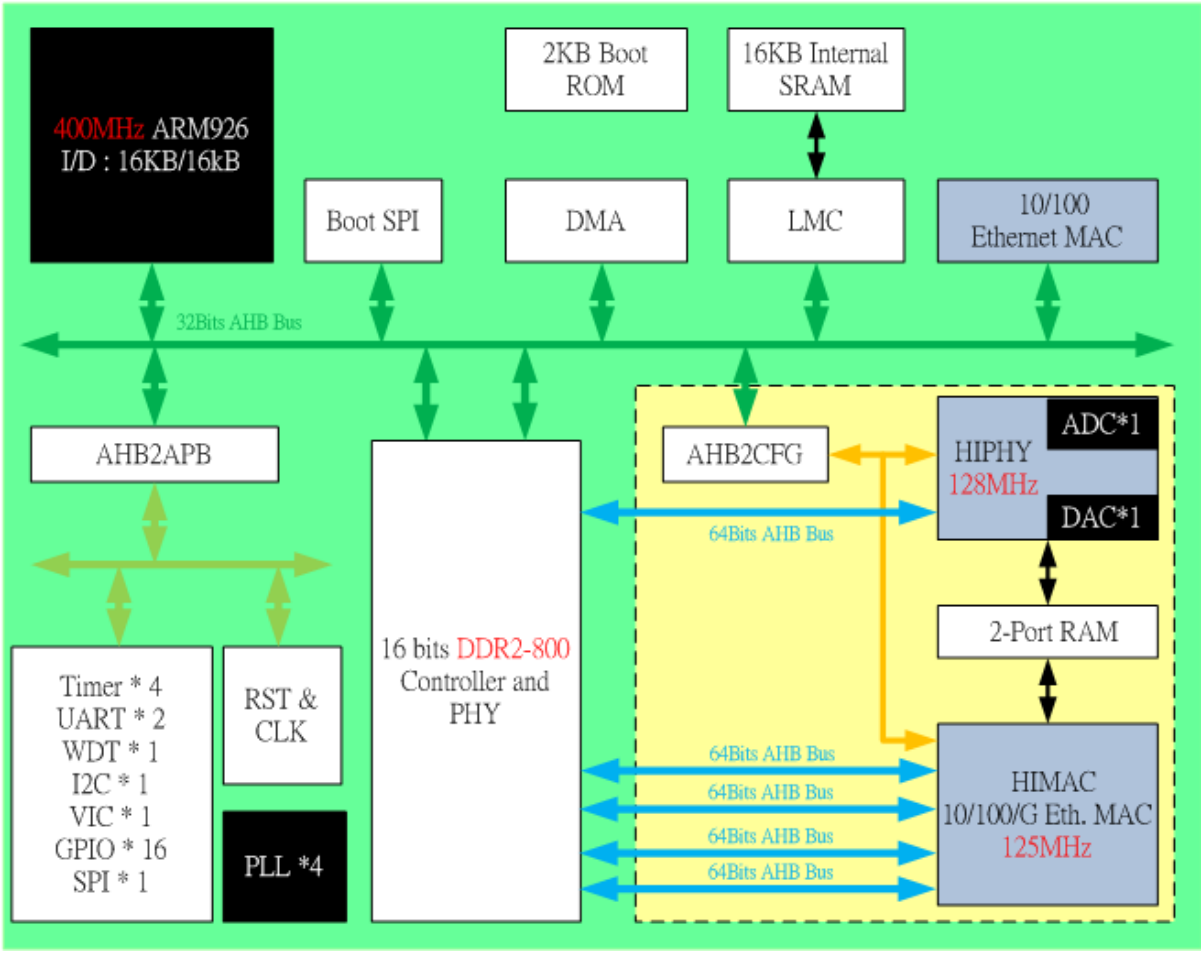
1.1.1 Introduction

The chip is a highly integrated ARM-based SOC solution for cable base band application with the high resolution ADC/DAC IPs. It provides various interfaces such as DDR2, MII, GMII, PCI, and IDE. Based on TSMC 65um LP technology process, the operating frequency of the embedded ARM926E core and the AHB bus is targeted at 400MHz and 200MHz respectively, and supports DDR2-800 Mbps interface.

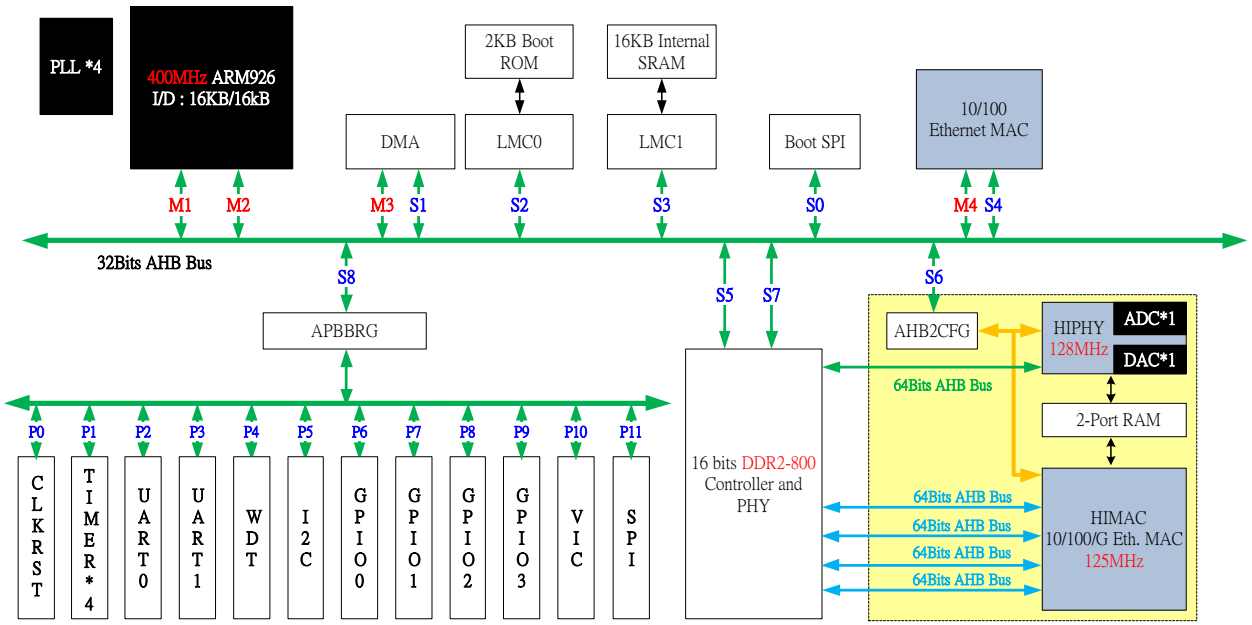
1.1.2 Features

- Embedded ARM926EJS Core (with 16KB/16KB I/D caches and 16k/16k I/D TCM)
- 10/100/1000M GMII 接口，数据上下行带宽总和到 160M（平滑带宽上下行综合为 320M，峰值带宽为 2G）
- 10/100M MII 接口（管理通道用）(改为 MII 接口)
- DDR2-800Mbps Interface, update to 4Gbyte capacity
- IGMP Snooping protocol
- SNMP protocol (HB 充当 SNMP Agent 和 SNMP 被管理设备，HM 充当 SNMP 被管理设备）
- 2 个 UART 接口、1 个 I2C 接口、1 个 SPI 接口、1 个 SPI BOOT 接口、16 个 GPIO 接口
- SPI flash memory update to 32Mbytes
- Hinoc PHY：
  - 采用正交幅度调制(QAM)，并根据同轴电缆不同的噪声、衰减、反射等情况自适应使用 QPSK、8QAM、16QAM、32QAM、64QAM、128QAM、256QAM、512QAM、1024QAM 的调制格式。
  - 为了避免由于多径效应引起的码间串扰问题，提高信道的利用率，选用了正交频分复用(OFDM)技术传输数据。
  - 在纠错码方面，采用高编码效率、低实现复杂度的 BCH 码。可以在 16MHz 的信道带宽上实现 100Mb/s 以上的物理层传输速率，其有效带宽的频率利用率可达到 7bit/s/Hz。
- Hinoc MAC：
  - 支持 QOS
  - 支持 IGMP IGMPv2 和 MLDv1。
  - 支持 HINOC 协议的接纳及链路维护功能；
  - 支持 HINOC 协议的带宽分配功能；
  - 上下行带宽支持到 160Mhz。
  - 根据 vlan 进行优先级调度，支持 8 个优先级，每个端口 4 个队列
  - HB 可以支持 32 个 HM；
  - 可以配置每个 HM 的带宽。
  - 认证 HM 合法性；
  - 可以使能/不使能节点的流量；
  - MAC 地址学习 1k 个地址，需要满足每秒学习 64 个 mac 地址。
  - 组播支持 128 个组播组。
  - 支持地址老化。老化时间可配。

1.1.3 Block Diagram Overview



1.1.4 Bus Architecture Overview



1.2 Description

1.2.1 UR0431A IO description

Signal	I/O	Bit	Voltage	Description
JTAG				
TCK	I	1	3.3	JTAG Test Clock In. This clock controls the updates to the TAP Controller and the shifts through the Instruction register or selected data registers. Connect to GND for normal operation.
TDI	I	1	3.3	JTAG Test Data In. Serial test data is input on this pin and is shifted into the instruction or data register. This input is sampled on the rising edge of TCK.Connect to GND for normal operation.
RTCK	O	1	3.3	JTAG Test Clock Out. This clock controls the output data. Leave unconnected for normal operation.
TDO	IO	1	3.3	JTAG Test Data Out. The AR7400 outputs serial test data on this pin from the instruction or data register. This signal changes on the rising edge of RTCK. This pin is also sampled during TRST# – if TDO is sampled high during TRST, then the top-level DFT TAP controller is selected to be connected to the JTAG I/O pads. If TDO is sampled low during TRST, then the ARMs TAP controller is connected to the JTAG I/O pads. Leave unconnected for normal operation.
TMS	I	1	3.3	JTAG Test Mode Select. This input is the control signal for the TAP Controller.It is sampled on the rising edge of TCK.Connect to GND for normal operation.
TRST_N	I	1	3.3	JTAG Test Reset. This signal is asserted asynchronously to reset the TAP Controller, instruction register. Connect to GND for normal operation.
Reset				
RESET_N	I	1	3.3	Chip reset pin. Active low. (From power management IC)
PHY_RESET	O	1	3.3	External PHY reset. Active low.(SW register output)
Clock				
OSCIN_32M	I	1	3.3	Input clock. 32MHz.
OSCIN_25M	I	1	3.3	Input clock. 25Mhz.
DDR				
DDR_A14~0	O	15	1.8	DDR2 address
DDR_BA1~0	O	2	1.8	DDR2 bank Select.
DDR_CS_N	O	1	1.8	DDR2 chip Select.
DDR_RAS_N	O	1	1.8	DDR2 row Address Strobe.
DDR_CAS_N	O	1	1.8	DDR2 column Address Strobe.
DDR_WE_N	O	1	1.8	DDR2 write Enable.
DDR_CKE	O	1	1.8	DDR2 clock Enable. If low, this signal indicates to the DDR2 to enter the power-down state.
DDR_DQM1~0	O	2	1.8	DDR2 data Mask. DDR_DQM0 corresponds to DDR_DQ[7:0], others [15:8]
DDR_DQ15~0	IO	16	1.8	DDR2 data bus
DDR_DQS0P	IO	1	1.8	DDR2 data Strobe. For lower byte.
DDR_DQS0N	IO	1	1.8	DDR2 data Strobe. For lower byte.
DDR_DQS1P	IO	1	1.8	DDR2 data Strobe. For upper byte.
DDR_DQS1N	IO	1	1.8	DDR2 data Strobe. For upper byte.
DDR_CLKP	O	1	1.8	DDR2 differential Clock.
DDR_CLKN	O	1	1.8	DDR2 differential Clock.
DDR_ODT	O	1	1.8	DDR2 on die termination
10/100 MAC MII				
CPU_TX_CLK	I	1	3.3	发送和接收同步时钟信号，是由外部的时钟源提供。
CPU_TXD3~0	O	4	3.3	发送数据线
CPU_TX_EN	O	1	3.3	发送使能信号
CPU_TX_ER	O	1	3.3	发送错误信号

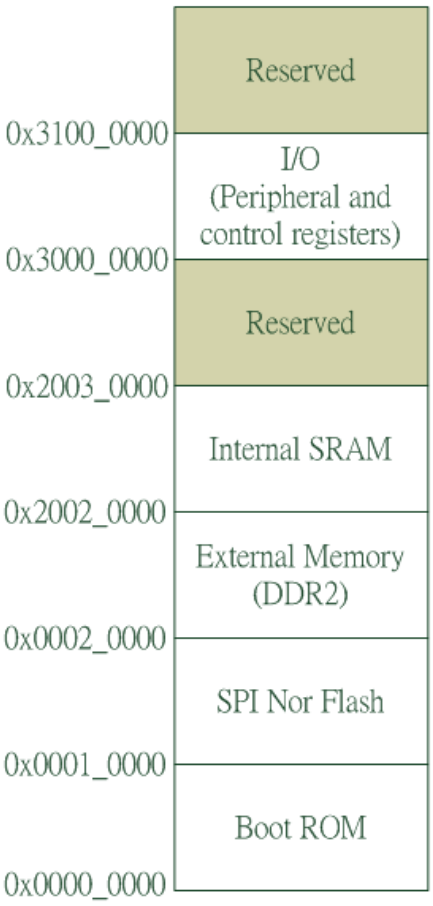
Signal	I/O	Bit	Voltage	Description
CPU_RXD3~0	I	4	3.3	接收数据线
CPU_RX_CLK	I	1	3.3	接收时钟
CPU_RX_ER	I	1	3.3	接收错误信号
CPU_RX_DV	I	1	3.3	接收错误信号
CPU_CRS	I	1	3.3	载波侦听有效信号
CPU_COL	I	1	3.3	冲突检测信号
CPU_MDC	O	1	3.3	MII Management Data Clock. The MAC core sources MDC as the timing reference for transfer of information on the MDI/MDO signals. MDC signal has no maximum high or low times. MDC minimum high and low times are 160 ns each, and the minimum period for MDC is 400 ns.
CPU_MDIO	IO	1	3.3	MII Management Data In/Out. This is the data input signal from the PHY controller. The PHY drives the Read Data synchronously with respect to the MDC clock during the read cycles. This is also the data output signal from the MAC core that drives the control information during the Read/Write cycles to the PHY controller. The MAC core drives the MDO signal synchronously with respect to the MDC.
HIMAC GMII				
GMTX_GCLK	O	1	3.3	当 EMAC 工作在1000M 模式下的发送时钟，为125MHz
GMTX_CLK	I	1	3.3	当 EMAC 工作在 10/100M 模式下的发送时钟，为 25Mhz
GMTX_D7~0	O	8	3.3	RGMII transmitter data bus
GMTX_ER	O	1	3.3	RGMII transmitter error signal
GMTX_EN	O	1	3.3	RGMII transmitter enable signal
GMRX_CLK	I	1	3.3	RGMII receiver clock
GMRX_D7~0	I	8	3.3	RGMII receiver data bus
GMRX_DV	I	1	3.3	RGMII receiver enable signal
GMRX_ER	I	1	3.3	RGMII receiver error signal
GMRX_COL	I	1	3.3	RGMII Collision Detec
GMRX_CRS	I	1	3.3	RGMII Carrier Sense
Nor Flash SPI				
SPI_MEM_CS	O	1	3.3	SPI Chip Select. SPI_CS# is the chip select used to enable additional SPI devices.
SPI_ MEM_CLK	O	1	3.3	SPI Clock.
SPI_ MEM_DI	I	1	3.3	SPI Data In. The SPI_DI is used to transfer serial data into the ASIC.
SPI_ MEM_DO	O	1	3.3	SPI Data Out. SPI_DO is used to transfer serial data out of the ASIC.
SPI_ MEM_WP	O	1	3.3	Write protection
SPI				
SPI_CLK	O	1	3.3	SPI Clock.
SPI_CS	O	1	3.3	SPI Chip Select. SPI_CS# is the chip select used to enable additional SPI devices.
SPI_DI	I	1	3.3	SPI Data In. The SPI_DI is used to transfer serial data into the ASIC.
SPI_DO	O	1	3.3	SPI Data Out. SPI_DO is used to transfer serial data out of the ASIC.
GPIO				
GPIO15~0	IO	16	3.3	General Purpose I/O. GPIO1 could be output of Sys_Test_CLK divided by 16. GPIO2 could be output of Mem_Test_CLK divided by 16. GPIO3 could be output of HiMAC_Test_CLK divided by 4. GPIO5 could be output of HiPY_Test_CLK divided by 4. GPIO0/4/8/12 could be input for external interrupt individually.
UART				

Signal	I/O	Bit	Voltage	Description
UART_TX1	O	1	3.3	UART1 transmit data output
UART_RX1	I	1	3.3	UART1 receive data input
UART_TX2	O	1	3.3	UART2 transmit data output
UART_RX2	I	1	3.3	UART2 receive data input
I2C				
I2C_SCLK	O	1	3.3	Serial clock output of I2C bus (control RF)
I2C_SDA	IO	1	3.3	Serial data input/output of I2C bus (control RF)
System Application				
TESTM3	I	1	3.3	Function modes selection 4'b0000: From SPI - HM ARMJTAG 4'b0001: From SPI - HM Non ARMJTAG 4'b0010: From SPI - HB ARMJTAG 4'b0011: From SPI - HB Non ARMJTAG 4'b0100: From InternalROM - HM ARMJTAG 4'b0101: From InternalROM - HM Non ARMJTAG 4'b0110: From InternalROM - HB ARMJTAG 4'b0111: From InternalROM - HB Non ARMJTAG Others: Debugging modes
TESTM2	I	1	3.3	
TESTM1	I	1	3.3	
TESTM0	I	1	3.3	
HIPHY AFE – ADC				
RXQN	I	1	2.5	Q channel analog base-band data negative input for receive mode
RXQP	I	1	2.5	Q channel analog base-band data positive input for receive mode
RXIN	I	1	2.5	I channel analog base-band data negative input for receive mode
RXIP	I	1	2.5	I channel analog base-band data positive input for receive mode
AVDDREFI	I	1	1.2	I channel Positive reference
AGNDREFI	I	1	-	I channel Negative reference
AVDDREFQ	I	1	1.2	Q channel Positive reference
AGNDREFQ	I	1	-	Q channel Negative reference
VCMINREF	O	1	2.5	Output reference voltage to be taken as reference for the ADC input signal common mode level
HIPHY AFE – DAC				
TXQN	O	1	2.5	Q channel analog base-band data negative output for transmit mode
TXQP	O	1	2.5	Q channel analog base-band data positive output for transmit mode
TXIN	O	1	2.5	I channel analog base-band data negative output for transmit mode
TXIP	O	1	2.5	I channel analog base-band data positive output for transmit mode
Iref	IO	1	2.5	Reference Current. To be used with an External Reference Resistor.
Vdref	IO	1	2.5	Reference voltage decoupling
AFE_POWERDN	O	1	3.3	Reference or Bias for HINOC power amplifier
AFE_TXEN	O	1	3.3	TX output Port
AFE_RXEN	O	1	3.3	RX output Port
AFE_GAIN4~0	IO	5	3.3	The control port of AFE module
PD_IN	I	1	3.3	
PD_OUT	O	1	3.3	

2 Memory Mapping

2.1 Introduction

Memory map is composed of memory space (SPI Nor Flash and DDR2), register space (Internal SRAM, control register, etc.), and I/O space (Peripherals) which are shared among the chip modules.
The DDR2 interface and SPI Nor Flash interface are dedicated to memory connection.
Internal SRAM allows sharing of critical resource such as memories between CPU subsystem and system DMA.
I/O space represents a generic address space used to access several sub modules used to control peripherals and manage their accesses.



Memory Space	CPU Access	System DMA Access
Boot ROM	Yes	
SPI Nor Flash	Yes	
DDR2	Yes	Yes
Internal SRAM	Yes	Yes
I/O	Yes	

2.2 Detailed Mapping

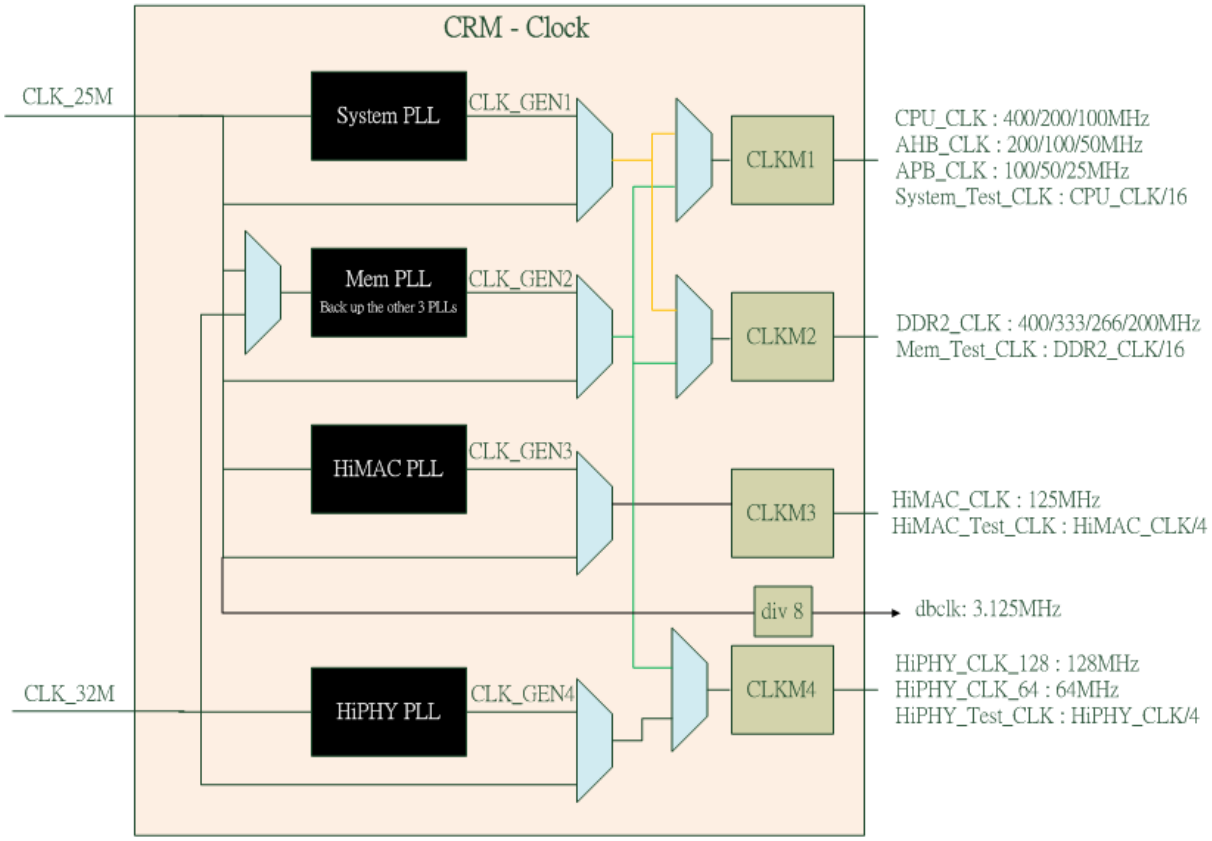
Name	Start Address	End Address	Description
AHB IP			
SPI boot	0x0001_0000	0x0001_FFFF	SPI NOR Flash memory, 64Mb
DMA	0x3000_0000	0x3000_FFFF	
DDR2 Controller	0x3001_0000	0x3001_FFFF	
LMC	0x3002_0000	0x3002_FFFF	
10/100 ethernet	0x3003_0000	0x3003_FFFF	
HiNOC	0x3004_0000	0x3005_FFFF	
Reserved	0x3006_000	0x300F_FFFF	
APB Bridge Base	0x3010_0000	0x301F_FFFF	
Reserved	0x3020_0000	0x30FF_FFFF	

APB IP	Start Address	End Address	Description
CLKRST	0x3010_0000	0x3010_FFFF	PLL control and clock mux selection
TIMER	0x3011_0000	0x3011_FFFF	
UART1	0x3012_0000	0x3012_FFFF	
UART2	0x3013_0000	0x3013_FFFF	
WDT	0x3014_0000	0x3014_FFFF	
I2C	0x3015_0000	0x3015_FFFF	
GPIO_INS0	0x3016_0000	0x3016_FFFF	
GPIO_INS1	0x3017_0000	0x3017_FFFF	
GPIO_INS2	0x3018_0000	0x3018_FFFF	
GPIO_INS3	0x3019_0000	0x3019_FFFF	
VIC	0x301A_0000	0x301A_FFFF	
SPI	0x301B_0000	0x301B_FFFF	
Reserved	0x301C_0000	0x301F_FFFF	

3 Clocks, Reset

3.1 Introduction

The device clocks/reset/power architecture includes the functional clocks, interface clocks, reset distribution and wake-up requests for the platform, and peripherals. The functional and interface clocks are all generated from two input clocks: 25-MHz and 32-MHz. The clocks are generated



The CRM performs the following functions:

* Controls four on-chip PLLs that generate CPU_CLK, DDR2, HiMAC_CLK and HiPHY_CLK.
* Manages the clocks and resets distributed to the platform and to certain peripherals.
* Performs the transition between the power modes: awake, half speed and off.

The System PLL is located in the platform; it converts the input clock (25 MHz) to a high-frequency clock (maximum 800 MHz/1000MHz), which is then distributed within the platform and to the various on-chip peripherals.

The Mem PLL is located in the platform; it converts the input clock (25 MHz) to a high-frequency clock (maximum 800 MHz/1000MHz) or converts the input clock (32 MHz) to a high-frequency clock (maximum 256MHz), which is then distributed within the various on-chip peripherals or HiPHY PLL.

The HiMAC PLL is located in the platform; it converts the input clock (25 MHz) to a high-frequency clock (maximum 250 MHz), which is then distributed within the platform and to HiNoC peripherals.

The HiPHY PLL is located in the platform; it converts the input clock (32 MHz) to a high-frequency clock (maximum 256 MHz), which is then distributed within the platform and to HiNoC peripherals.

The platform clocking and reset architecture handles management and distribution of peripheral clocks. It consists of four clock managers (CLKM1, CLKM2, CLKM3 and CLKM4).

Reset manager module (RSTM) receives reset inputs from external port (Hard_Reset) and generates reset signals for the platform modules and for the peripherals. From a functional point of view, the RSTM module is tightly coupled with CLKMx modules as it shares the same set of registers.

3.2 Clocks

The db_clk is dedicated for one of dw_gpio's clock which is used for debounce circuit to filer noise during interrupt.

Normal Mode
In normal mode, the CLKM1, CLKM2, CLKM3 and CLKM4 domains are run at different clock speeds. (System PLL, Mem PLL and HiMAC PLL are within the same clock source [CLK_25M]). It's default mode. And, operated speed can be configured when system is ready.

1	System PLL	CPU_CLK: 400MHz/200MHz/100MHz (Dynamic switch) AHB_CLK: 200MHz APB_CLK: 100MHz DDR2_CLK: 400MHz
	HiMAC PLL	HiMAC_CLK: 125MHz
	HiPHY PLL	HiPHY_CLK: 128MHz
2 (Typical corner)	System PLL	CPU_CLK: 480MHz/240MHz (Dynamic switch) AHB_CLK: 240MHz APB_CLK: 120MHz
	Mem PLL	DDR2_CLK: 400MHz
	HiMAC PLL	HiMAC_CLK: 125MHz
	HiPHY PLL	HiPHY_CLK: 128MHz

Fully-synchronous Mode
In fully synchronous mode, the CLKM1 and CLKM2 run at the same clock frequency derived from System PLL (Clock source : CLK_25M and disable HiPHY PLL). And,CLKM4 clock source comes from Mem PLL (Clock source : CLK_32M). The fully-synchronous mode is a special case of normal mode, where the clock divider bits for CPU_CLK and DDR2_CLK domains are equal.

3	System PLL	CPU_CLK: 400MHz/200MHz (Dynamic switch) AHB_CLK: 200MHz APB_CLK: 100MHz DDR2_CLK: 400MHz
	HiMAC PLL	HiMAC_CLK: 125MHz
	Mem PLL	HiPHY_CLK: 128MHz

Test Mode
In test mode, operated speed and clock MUX selection are default. And, monitor PLL test clocks (System_Test_CLK, Mem_Test_CLK, HiMAC_Test_CLK and HiPHY_Test_CLK) by separate function ports.

4	System PLL	Sys_Test_CLK: CPU_CLK/16
	Mem PLL	Mem_Test_CLK: DDR2_CLK/16
	HiMAC PLL	HiMAC_Test_CLK: HiMAC_CLK/4
	HiPHY PLL	HiPY_Test_CLK: HiPHY_CLK/4

Bypass Mode

In bypass mode, all PLLs are bypassed. And, monitor PLL test clocks (System_Test_CLK, Mem_Test_CLK, HiMAC_Test_CLK and HiPHY_Test_CLK) by separate function ports.

5	System PLL	Clock out of System PLL
	Mem PLL	Clock out of Mem PLL
	HiMAC PLL	Clock out of HiMAC PLL
	HiPHY PLL	Clock out of HiPHY PLL

3.3 Resets diagram

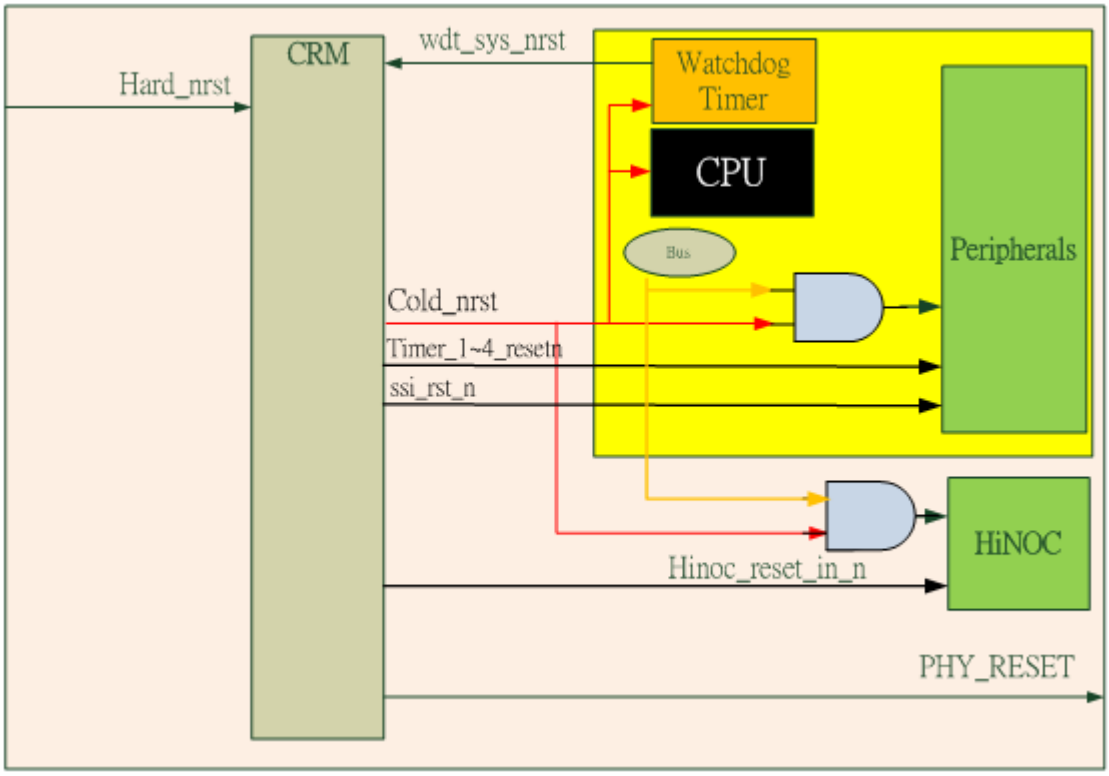
Reset overview

Hardware Resets

External hardware reset (Hard_nrst) is hardware reset. And, CRM generates internal resets (Cold_nrst) for each module and generate one reset pulse (PHY_RESET) for external devices.

Software Resets - Individually Resetting Modules (Presetn [n:0])

To software-reset an individual module, use the reset bit field of module register. Refer to the corresponding module chapter for further information.

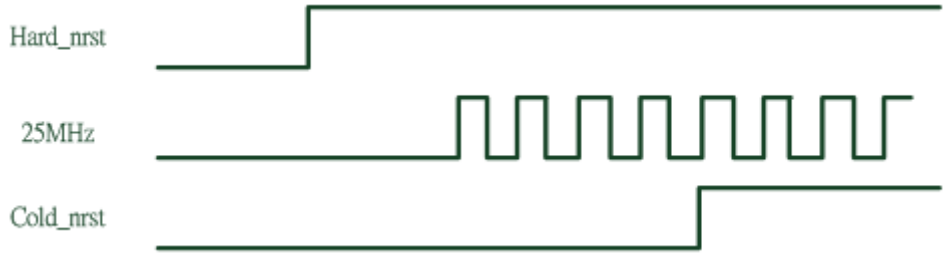


Reset Timing

Cold Reset Sequence Description

When Hard_nrst is asserted high, the CRM asynchronously asserts Cold_nrst low. The CRM maintains signal low for a predefined period before releasing them:

Cold_nrst are asserted high after 20 cycles of 25MHz.



Peripherals Reset
Hardware Resets : Cold_nrst from Hard_nrst or PoR_nrst
Software Resets : Presetn (from AMBA) or internal reset register

Reset Mapping:

Reset Name	**Description**	**From**	**To**
Hard_nrst	Device power-up reset	Hard_Reset port	CRM
Cold_nrst	Hard_nrst & por_nrst	CRM	Whole chip
PHY_RESET	Cold_nrst & register (system control)	CRM	External Device
hinoc_reset_in_n	hard_nrst & PoR_nrst & wdt_sys_nrst & register (system control).	CRM	For HiNOC
Presetn[n:0]	CPU	BUS	Peripherals

Module and Reset

Module	Cold_nrst	Presetn	wdt_sys_nrst
SyePLL	*		
MemPLL	*		
HiMAC PLL	*		
HiPHY PLL	*		
LMC	*	*	*
DMA	*	*	*
CRM	*	*	*
VIC	*	*	*
Timer1	*	*	*
Timer2	*	*	*
Timer3	*	*	*
Timer4	*	*	*
WDT	*	*	*

Peripherals Resets

Peripheral	Cold_nrst	Presetn	wdt_sys_nrst	Soft_nrst
DDR2	*	*	*	*
HiNoC	*	*	*	
10/100 Mac	*	*	*	
Boot SPI	*	*	*	
UART1	*	*	*	*
UART2	*	*	*	*
I2C	*	*	*	
SPI	*	*	*	
GPIO	*	*	*	

Programming Guide

Reset Type	Causal Event	Resulting reset	Reset Action	Reset Flag
Cold reset	Low-level on/ Power-on reset	Hard_nrst	Whole chip	
Cold reset	Power-on reset	PoR_nrst	Whole chip	
Warm reset	Watchdog timeout	wdt_sys_nrst		
Soft reset		Presetn[n:0] / Internal reset register		

3.4 Register Manual

3.5 PLL control setting

3.5.1 System PLL / Mem PLL / HiMAC PLL control setting

For **System PLL**, **Mem PLL** and **HiMAC PLL,** the output frequency is calculated according to following table and formulas:

Parameter	SYM	Condition		min(MHz)	max(MHz)
Comparison Freq	Fref	Fref = Fin / NR		10	50
Input clock Freq	Fin	Fin = NR * Fref		10	400
Output clock Freq	Fout	Fout = Fvco / OD	High-band	62.5	1000
			Low-band	37.5	600
VCO operating Range	Fvco	Fvco = Fref * NF	High-band	500	1000
			Low-band	300	600

PD	BP	OD	Description	Fout
0	0	0	Normal mode	Fref * NF
		1		Fref * NF / 2
		2		Fref * NF / 4
		3		Fref * NF / 8
0	1	x	Bypass	Fin
1	x	x	Power down	0

BS	Description
0	low band
1	High band

For example, to obtain a 800MHz output with an input frequency of 25MHz:

Fin = 25MHz
Select normal mode in high band: BS = 1
Fref = Fin / NR = 25MHz / 1 = 25MHz
Fout = Fvco / OD => 800MHz = Fvco/1 => Fvco = 800MHz.
Fvco = 800MHz = Fref * NF => 800MHz = 25MHz * NF => NF = 64

PLL Configuration									
Fin (MHz)	Fout (MHz)	BS	BP	PD	OD[1:0]	F[6:0]	R[4:0]	Fref	Fvco
25	800	1	0	0	0	63	1	12.5	800

25	400	1	0	0	1	63	1	12.5	800
25	200	1	0	0	2	63	1	12.5	800
25	125	1	0	0	2	39	1	12.5	500
32	128	1	0	0	2	31	1	16	512

3.5.2 HiPHY PLL control setting

The VCO oscillating frequency is a function of the input reference frequency and of the multiplication /division ratios. It can be determined in the following way:

Fvco = (M/N) * Fclkin

where M is the Feedback Multiplication Ratio and N is the Input Frequency Division Ratio.
However, some limits apply:

60MHz >= (Fclkin/N) >= 5MHz

For VCORANGE = 0
200MHz <= Fvco (=Fclkin*M/N) <= 500MHz

For VCORANGE = 1
500MHz <= Fvco (=Fclkin*M/N) <= 1000MHz

Finally, the output clock frequency is derived from a programmable division of the VCO frequency:
Fclkoit1 = (1/P) * Fvco *
Fclkoit2 = (1/R) * Fvco *
Fclkoit3 = (1/S) * Fvco *

4 Interconnect

4.1 Overview

All IPs on AMBA bus are 32bits data bus. Please refer 1.1.3 Block Diagram overview.

4.2 Clock Connection:

Module	From	Module	To	Description
PLL CRM	Sys_Test_CLK	**TOP ports**	GPIO1	The clock after divided by 16, GPIO is controlled by register
	Mem_Test_CLK		GPIO2	
	HiMAC_Test_CLK		GPIO3	The clock after divided by 4, GPIO is controlled by register
	HiPY_Test_CLK		GPIO5	
PLL CRM	pclk	**TIMER**	timer_1_clk	Define 4 input clocks of TIMER
			timer_2_clk	
TOP ports	OSCIN_25M		timer_3_clk	
			timer_4_clk	
PLL CRM	pclk	**I2C**	ic_clk	
PLL CRM	pclk	**SPI(APB)**	ssi_clk	
PLL CRM	dbclk	**GPIO**	dbclk	Create gpio_dbclk to for GPIO interrupt feature

4.3 Control signals Connection:

HiNOC connection:
There are no hresp[1:0] signals of all 5 AHB masters from HiNOC to DDR, they only support “OK” response

Module	From	Module	To	Description
HiNOC	gmii_tx_gclk	**TOP ports**	GMTX_GCLK	
	gmii_tx_clk		GMTX_CLK	
	gmii_tx_data[7:0]		GMTX_D7~0	
	gmii_tx_en		GMTX_ER	
	gmii_tx_er		GMTX_EN	
	gmii_rx_clk		GMRX_CLK	
	gmii_rx_data[7:0]		GMRX_D7~0	
	gmii_rx_dv		GMRX_DV	
	gmii_rx_er		GMRX_ER	
	gmii_col		GMRX_COL	
	gmii_crs		GMRX_CRS	
	afe_powerdn		AFE_POWERDN	Direct connect from HiNOC’s signals to TOP ports
	afe_txen		AFE_TXEN	
	afe_rxen		AFE_RXEN	
	pd_in		PD_IN	
	pd_out		PD_OUT	
	afe_gain[4:0]		AFE_GAIN4~0	
	test_mode_pad_input[3]		TESTM3	
	test_mode_pad_input[2]		TESTM2	
	test_mode_pad_input[1]		TESTM1	

Peripheral connection:

DDR connection:

	test_mode_pad_input[0]		TESTM0	
HiNOC	adc_vinpi_pad_input	**TOP ports**	RXIP	
	adc_vinni_pad_input		RXIN	
	adc_vinpq_pad_input		RXQP	
	adc_vinnq_pad_input		RXQN	
	adc_avddrefi_pad_input		AVDDREFI	
	adc_agndrefi_pad_input		AGNDREFI	
	adc_avddrefq_pad_input		AVDDREFQ	
	adc_agndrefq_pad_input		AGNDREFQ	
	adc_vcminref_pad_output		VCMINREF	
HiNOC	dac_iref_pad_inout	**TOP ports**	IREF	Connect iref and ireffb together. From datasheet description.
	dac_ireffb_pad_input			
	dac_vdref_pad_inout		VDREF	
	dac_ioutpi_pad_output		TXIP	
	dac_ioutni_pad_output		TXIN	
	dac_ioutpq_pad_output		TXQP	
	dac_ioutnq_pad_output		TXQN	

Module	From	Module	To	Description
Boot SPI	ss_pad_o	**TOP ports**	SPI_MEM_CS	Direct connect from Boot-SPI's signals to TOP ports
	sclk_pad_o		SPI_ MEM_CLK	
	mosi_pad_o		SPI_ MEM_DO	
	miso_pad_i		SPI_ MEM_DI	
	wp_pad_o		SPI_ MEM_WP	
gpio_ins0	Port A/B/C/D	**TOP ports**	GPIO0/1/2/3	There are 4 * dw_gpio libraries. Connect each gpio signals to TOP ports
gpio_ins1	Port A/B/C/D		GPIO4/5/6/7	
gpio_ins2	Port A/B/C/D		GPIO8/9/10/11	
gpio_ins3	Port A/B/C/D		GPIO12/13/14/15	

Module	From	Module	To	Description
DDR	AHB0 (64 btis)	**HiNOC**	hiphy	HiPHY_CLK_128 (128MHz) domain
	AHB1 (64 btis)		pmau_read	HiMAC_CLK (125MHz) domain
	AHB2 (64 btis)		data_read	
	AHB3 (64 btis)		pmau_write	
	AHB4 (64 btis)		data_write	
DDR	AHB5 (32 btis)	**TOP**	AMBA_BUS	
TOP	1'b0	**DDR**	Srefresh_enter ahbY_HAPCMD	Won't use those signals. Memory will be re-flash/ Auto-Precharge by DDR controller.

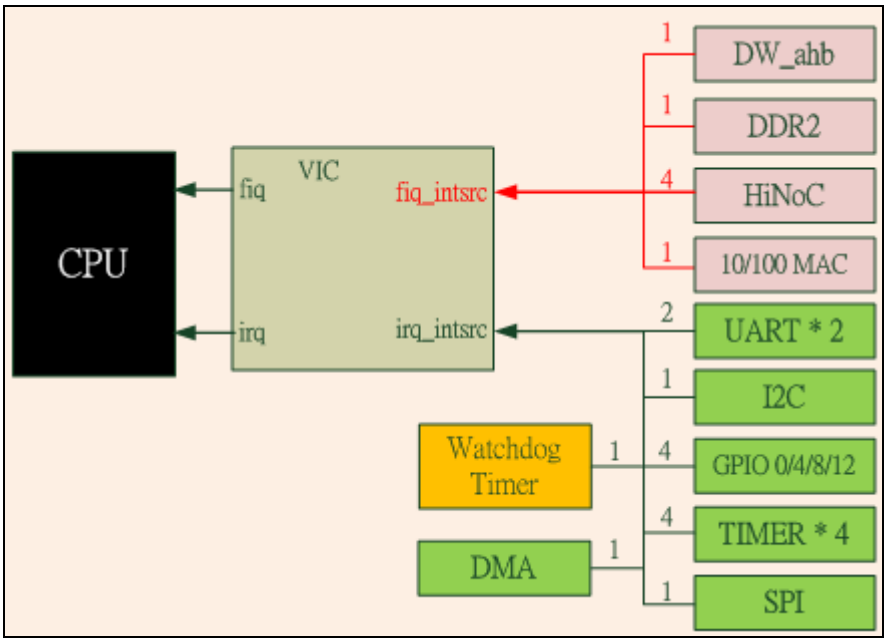
4.4 ADC Test Mode Connection:

On ADC test mode (TESTM3~0 is 4'b1000), some ADC input signals should be connected to high/low in order to reduce test pins number. Others signals connection, please prefer to attach file on chapter 7: Test_Plan.

Signals	Value
envcminref	1'b1
bctrl [4:0]	5'b01000
bctrlbuf[2:0]	3'b010
en_static_bias_buf	1'b0
endoutz	1'b0

4.5 Interrupt signals Connection:

The interrupt polarity of HiNOC: **hinoc_int[3:0]** are all active high and leveling trigger.



5 Direct Memory Access (DMA)

5.1 Overview

5.2 System DMA

6 Interrupts

7 Memory Interface

7.1 DDR2

7.1.1 Overview

7.1.2 Environment

7.1.3 DDR2 Function Interface

7.1.4 Architecture and Function Description

7.1.5 DDR2 Programming

7.1.6 DDR2 Register Manual

7.2 SPI Nor Flash Controller

7.2.1 Overview

7.2.2 Environment

7.2.3 SPI Nor Flash Function Interface

7.2.4 Architecture and Function Description

7.2.5 SPI Nor Flash Register Manual

7.3 Internal SRAM

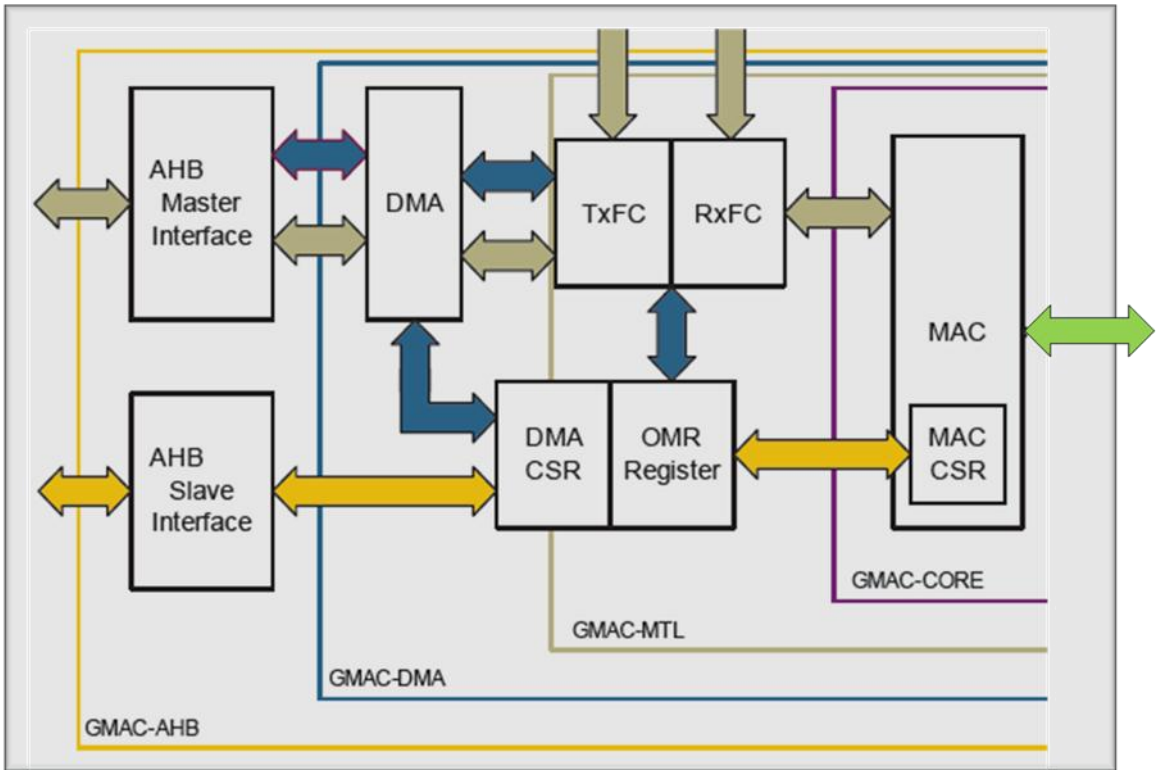
8 HiNoC

9 10/100 Ethernet Mac

9.1 10/100 Ethernet Mac Introduction

The DWC_gmac enables a host to transmit and receive data over Ethernet in compliance with the IEEE 802.3 standard and it supports the Media Independent Interface (MII). User can use the DWC_gmac in number of applications such as switches and network interface cards.

Table 10-1 DWC_gmac block diagram



Transmit and Receive FIFOs

The Transmit FIFO (Tx FIFO) buffers the data read, from the system memory, by the DMA before transmission by the MAC. Similarly, the Receive FIFO (Rx FIFO) stores the Ethernet frames received from the line until the DMA transfers them to system memory.

These are asynchronous FIFOs, as they also transfer the data between the application clock and the MAC line clocks. Both FIFOs are required to be two ported RAM of configurable width (35, 68, or 133 bits wide for 32, 64, or 128 data bus widths, respectively).

MAC

The MAC supports the following features:

- 10 and 100Mbps data transfer rates with the PHY interfaces: IEEE 802.3-compliant GMII/MII interface to communicate with an external Gigabit/Fast Ethernet PHY
- Full-duplex operation:
  - IEEE 802.3x flow control automatic transmission of zero-quanta pause frame on flow control input de-assertion
  - Optional forwarding of received pause control frames to the user application
- Half-duplex operation:
  - CSMA/CD Protocol support
  - Flow control using back-pressure support

  - Frame bursting and frame extension in 1000 Mbps half-duplex operation
- Preamble and start-of-frame data (SFD) insertion in Transmit path
- Preamble and SFD deletion in the Receive path
- Automatic CRC and pad generation controllable on a per-frame basis
- Automatic Pad and CRC Stripping options for receive frames
- Flexible address filtering modes, such as:
  - Up to 31 additional 48-bit perfect (DA) address filters with masks for each byte
  - Up to 96 additional 48-bit perfect (DA) address filters that can be selected in blocks of 32 and 64 registers
  - Up to 31 48-bit SA address comparison check with masks for each byte
  - Pass all incoming packets (as per filter) with a status report
- Programmable frame length to support Standard or Jumbo Ethernet frames with up to 16 KB of size
- Programmable Interframe Gap (IFG) (40-96 bit times in steps of 8)
- Option to transmit frames with reduced preamble size
- Separate 32-bit status for transmit and receive packets
- IEEE 802.1Q VLAN tag detection for reception frames
- Additional frame filtering:
  - VLAN tag-based: Perfect match and Hash-based (optional) filtering
  - Layer 3 and Layer 4-based: TCP or UDP over IPv4 or IPv6
- Separate transmission, reception, and control interfaces to the application
- Configurable big-endian and little-endian for Transmit and Receive paths
- 32-bit, 64-bit, or 128-bit data transfer interface on system-side
- Network statistics (optional) with RMON/MIB Counters (RFC2819/RFC2665)
- Optional module to detect LAN wake-up frames and AMD Magic Packet frames
- Optional Receive module for checksum off-load for received IPv4 and TCP packets encapsulated by the Ethernet frame (Type 1)
- Optional Enhanced Receive module for checking IPv4 header checksum and TCP, UDP, or ICMP checksum encapsulated in IPv4 or IPv6 datagrams (Type 2)
- Optional module to support Ethernet frame timestamping as described in IEEE 1588-2002 and IEEE 1588-2008 The 64-bit timestamps are given in the transmit or receive status of each frame
- MDIO Master interface (optional) for PHY device configuration and management
- Standard IEEE P802.3az, version D3.2 for Energy Efficient Ethernet
- Flexibility to control the Pulse-Per-Second (PPS) output signal (ptp_pps_o)
- CRC replacement, Source Address field insertion or replacement, and VLAN insertion, replacement, and deletion in transmitted frames with per-frame control

Transaction Layer (MTL)

The MTL block consists of two sets of FIFOs: a Transmit FIFO with programmable threshold capability and a Receive FIFO with a programmable threshold (64 bytes).

The MTL block supports the following features:

- 32-bit, 64-bit, or 128-bit Transaction Layer block (bridges the application and the GMAC-CORE)
- Single-channel Transmit and Receive engines
- Data transfers executed using simple FIFO-protocol
- Synchronization for all clocks in the design (Transmit, Receive, and system clocks)
- Optimization for packet-oriented transfers with frame delimiters
- Four separate ports for system-side and GMAC-CORE-side transmission and reception
- Two 2-port RAM-based asynchronous FIFOs with synchronous/asynchronous Read and Write operation with respect to the Read and Write clocks (one for transmission and one for reception)
- FIFO instantiation outside the top-level module to facilitate memory testing/instantiation
- 128-byte, 256-byte, or 512-byte, or 1-KB, 2-KB, 4-KB, 8-KB, 16-KB, or 32-KB receive FIFO sizes on reception
- Optional interface to indicate the length of a received frame at the top of the MTL Rx FIFO in the GMAC-MTL configuration
- Programmable burst-length for starting a burst up to half the size of the MTL Rx and Tx FIFO in the GMAC-MTL configuration
- Insertion of Receive Status vectors into the Receive FIFO after the EOF transfer This enables multiple-frame storage in the Receive FIFO without requiring another FIFO to store those frames' Receive Status.

- Configurable Receive FIFO threshold (default fixed at 64 bytes) in Cut-Through or Threshold mode
- Option to filter all error frames on reception and not forward them to the application in Store-and-Forward mode
- Option to forward under-sized good frames
- Supports statistics by generating pulses for frames dropped or corrupted (due to overflow) in the Receive FIFO
- 256-byte or 512-byte, or 1-KB, 2-KB, 4-KB, 8-KB, or 16-KB FIFO sizes on transmission
- Store and Forward mechanism for transmission to the MAC
- Threshold control for transmit buffer management
- Configurable number of frames to be stored in FIFO at any time The default is two frames (fixed) with internal DMA, and up to eight frames in GMAC-MTL configuration
- Automatic generation of PAUSE frame control or backpressure signal to the MAC based on Receive FIFO-fill (threshold configurable) level
- Automatic retransmission of Collision frames for transmission
- Discard frames on late collision, excessive collisions, excessive deferral, and under-run conditions
- Software control to flush Tx FIFO
- Disabling of Data FIFO RAM chip-select when inactive to reduce power consumption
- Optional module to calculate and insert IPv4 header checksum and TCP, UDP, or ICMP checksum in frames transmitted in the store-and-forward mode

DMA Block

The DMA block exchanges data between the MTL block and host memory. The host can use a set of registers (DMA CSR) to control the DMA operations.

The DMA block supports the following features:

- 32-bit, 64-bit, and 128-bit data transfers
- Single-channel Transmit and Receive engines
- Fully synchronous design operating on a single system clock (except for CSR module, when a separate CSR clock is configured)
- Optimization for packet-oriented DMA transfers with frame delimiters
- Byte-aligned addressing for data buffer support
- Dual-buffer (ring) or linked-list (chained) descriptor chaining
- Descriptor architecture to allow large blocks of data transfer with minimum CPU intervention (each descriptor can transfer up to 8 KB of data)
- Comprehensive status reporting for normal operation and transfers with errors
- Individual programmable burst size for Transmit and Receive DMA Engines for optimal host bus utilization
- Programmable interrupt options for different operational conditions
- Per-frame Transmit or Receive complete interrupt control
- Round-robin or fixed-priority arbitration between Receive and Transmit engines
- Start and Stop modes
- Separate ports for host CSR access and host data interface

9.2 DMA controller:

The DMA has independent Transmit and Receive engines, and a CSR space. The Transmit engine transfers data from system memory to the device port (MTL), while the Receive engine transfers data from the device port to the system memory. The controller use descriptors to efficiently move data from source to destination with minimal Host CPU intervention. The DMA is designed for packet-oriented data transfers such as frames in Ethernet. The controller can be programmed to interrupt the Host CPU for situations such as Frame Transmit and Receive transfer completion, and other normal/error conditions.

The DMA and the Host driver communicate through two data structures:

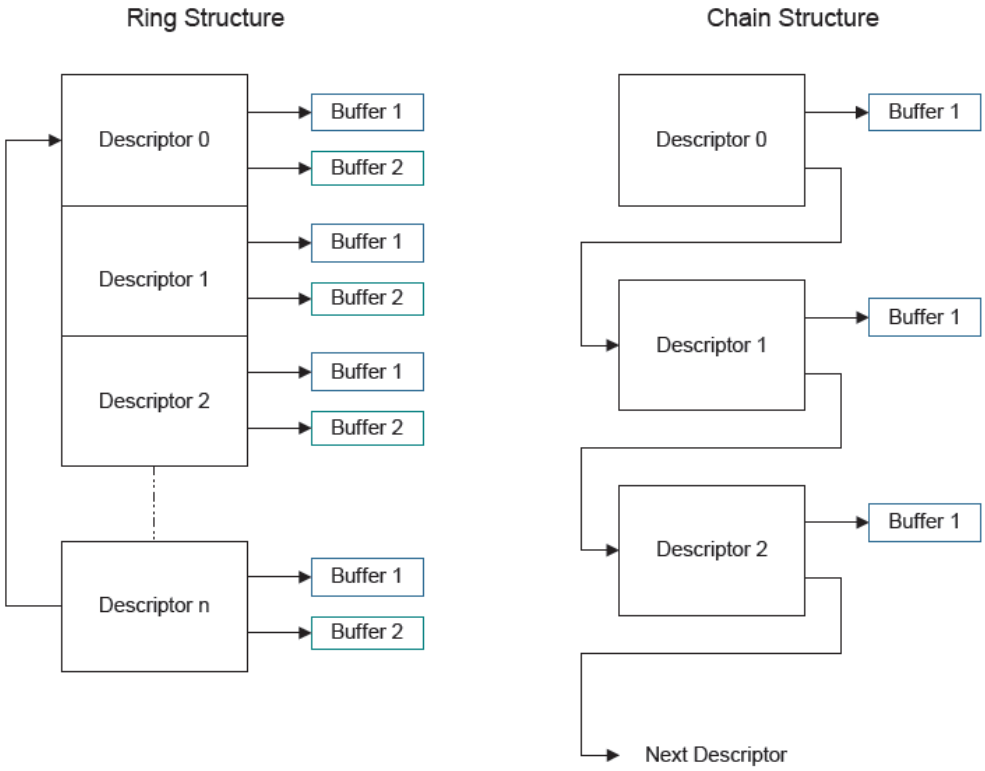
- Control and Status registers (CSR)
- Descriptor lists and data buffers

The DMA transfers data frames received by the MAC to the Receive Buffer in the Host memory, and Transmit data frames from the Transmit Buffer in the Host memory. Descriptors that reside in the Host memory act as pointers to these buffers.

There are two descriptor lists; one for reception, and one for transmission. The base address of each list is written into Register 3 (Receive Descriptor List Address Register) and Register 4 (Transmit Descriptor List Address Register), respectively. A descriptor list is forward linked (either implicitly or explicitly). The last descriptor may point back to the first entry to create a ring structure. Explicit chaining of descriptors is accomplished by setting the second address chained in both Receive and Transmit descriptors (RDES1[24] and TDES1[24]). The descriptor lists resides in the Host physical memory address space. Each descriptor can point to a maximum of two buffers. This enables two buffers to be used, physically addressed, rather than contiguous buffers in memory.

A data buffer resides in the Host physical memory space, and consists of an entire frame or part of a frame, but cannot exceed a single frame. Buffers contain only data, buffer status is maintained in the descriptor. Data chaining refers to frames that span multiple data buffers. However, a single descriptor cannot span multiple frames. The DMA skips to the next frame buffer when end-of-frame is detected. Data chaining can be enabled or disabled.

Table 10-2 Descriptor ring and chain structure



9.2.1 Initialization

Initialization for the DWC_gmac is as follows.

1. Write to Register 0 (Bus Mode Register) to set Host bus access parameters.
2. Write to Register 7 (Interrupt Enable Register) to mask unnecessary interrupt causes.
3. Create the Transmit and Receive descriptor lists, and then write to DMA Register 3 (Receive Descriptor List Address Register) and Register 4 (Transmit Descriptor List Address Register), providing the DMA with the starting address of each list.
4. Write to Register 1 (MAC Frame Filter), Register 2 (Hash Table High Register), and Register 3 (Hash Table Low Register) for desired filtering options.

5. Write to Register 1 (MAC Frame Filter) to configure the operating mode and enable the transmit operation (Bit 3: Transmitter Enable). The PS and DM bits are set based on the auto-negotiation result (read from the PHY).
6. Write to Register 6 (Operation Mode Register) to set Bits 13 and 1 to start transmission and reception.
7. Write to Register 0 (MAC Configuration Register) to enable the Receive operation (Bit 2: Receiver Enable).
   The Transmit and Receive engines enter the Running state and attempt to acquire descriptors from the respective descriptor lists. The Receive and Transmit engines then begin processing Receive and Transmit operations. The Transmit and Receive processes are independent of each other and can be started or stopped separately.

9.2.2 Transmission

The Transmit DMA engine in default mode proceeds as follows:

1. The Host sets up the transmit descriptor (TDES0-TDES3) and sets the Own bit (TDES0[31]) after setting up the corresponding data buffer(s) with Ethernet Frame data.
2. When Bit 13 (ST) of Register 6 (Operation Mode Register) is set, the DMA enters the Run state.
3. While in the Run state, the DMA polls the Transmit Descriptor list for frames requiring transmission. After polling starts, it continues in either sequential descriptor ring order or chained order. If the DMA detects a descriptor flagged as owned by the Host (TDES0[31] = 1'b0), or if an error condition occurs, transmission is suspended and both the Bit 2 (Transmit Buffer Unavailable) and Bit 16 (Normal Interrupt Summary) of the Register 5 (Status Register) are set. The Transmit Engine proceeds to Step 9.
4. If the acquired descriptor is flagged as owned by DMA (TDES0[31] = 1'b1), the DMA decodes the Transmit Data Buffer address from the acquired descriptor.
5. The DMA fetches the Transmit data from the Host memory and transfers the data to the MTL for transmission.
6. If an Ethernet frame is stored over data buffers in multiple descriptors, the DMA closes the intermediate descriptor and fetches the next descriptor. Steps 3, 4, and 5 are repeated until the end-of-Ethernet-frame data is transferred to the MTL.
7. When frame transmission is complete, if IEEE 1588 timestamping was enabled for the frame (as indicated in the transmit status) the timestamp value obtained from MTL is written to the transmit descriptor (TDES2 and TDES3) that contains the end-of-frame buffer. The status information is then written to this transmit descriptor (TDES0). Because the Own bit is cleared during this step, the Host now owns this descriptor. If timestamping was not enabled for this frame, the DMA does not alter the contents of TDES2 and TDES3.
8. Bit 0 (Transmit Interrupt) of Register 5 (Status Register) is set after completing transmission of a frame that has Interrupt on Completion (TDES1[31]) set in its Last Descriptor. The DMA engine then returns to Step 3.
9. In the Suspend state, the DMA tries to re-acquire the descriptor (and thereby return to Step 3) when it receives a Transmit Poll demand and the Underflow Interrupt Status bit is cleared.

9.2.3 Reception

The Receive DMA engine's reception sequence is depicted in Figure 3-23 and proceeds as follows:

1. The host sets up Receive descriptors (RDES0-RDES3) and sets the Own bit (RDES0[31]).
2. When Bit 1 (SR) of Register 6 (Operation Mode Register) is set, the DMA enters the Run state. While in the Run state, the DMA polls the Receive Descriptor list, attempting to acquire free descriptors. If the fetched descriptor is not free (is owned by the host), the DMA enters the Suspend state and jumps to Step 9.
3. The DMA decodes the receive data buffer address from the acquired descriptors.
4. Incoming frames are processed and placed in the acquired descriptor's data buffers.
5. When the buffer is full or the frame transfer is complete, the Receive engine fetches the next descriptor.
6. If the current frame transfer is complete, the DMA proceeds to Step 7. If the DMA does not own the next fetched descriptor and the frame transfer is not complete (EOF is not yet transferred), the DMA sets the Descriptor Error bit in the RDES0 (unless flushing is disabled in Bit 24 of Register 6 (Operation Mode Register)). The DMA closes the current descriptor (clears the Own bit) and marks it as intermediate by clearing the Last Segment (LS) bit in the RDES0 value (marks it as Last Descriptor if flushing is not disabled), then proceeds to Step 8. If the DMA does own the next descriptor but the current frame transfer is not complete, the DMA closes the current descriptor as intermediate and reverts to Step 4.

7. If IEEE 1588 timestamping is enabled, the DMA writes the timestamp (if available) to the current descriptor's RDES2 and RDES3. It then takes the receive frame's status from the MTL and writes the status word to the current descriptor's RDES0, with the Own bit cleared and the Last Segment bit set.
8. The Receive engine checks the latest descriptor's Own bit. If the host owns the descriptor (Own bit is 1'b0), the Bit 7 (Receive Buffer Unavailable) of Register 5 (Status Register) is set and the DMA Receive engine enters the Suspended state (Step 9). If the DMA owns the descriptor, the engine returns to Step 4 and awaits the next frame.
9. Before the Receive engine enters the Suspend state, partial frames are flushed from the Receive FIFO. You can control flushing using Bit 24 of Register 6 (Operation Mode Register).
10. The Receive DMA exits the Suspend state when a Receive Poll demand is given or the start of next frame is available from the MTL's Receive FIFO. The engine proceeds to Step 2 and refetches the next descriptor.

9.2.4 Interrupts

Interrupts can be generated as a result of various events. The DMA Register 5 (Status Register) contains all the bits that might cause an interrupt. Register 7 (Interrupt Enable Register) contains an enable bit for each of the events that can cause an interrupt.

There are two groups of interrupts, Normal and Abnormal, as described in Register 5 (Status Register). Interrupts are cleared by writing a 1'b1 to the corresponding bit position. When all the enabled interrupts within a group are cleared, the corresponding summary bit is cleared. When both the summary bits are cleared, the interrupt signal sbd_intr_o is de-asserted. If the MAC is the cause for assertion of the interrupt, then any of the GLI, GMI, GPI, TTI, or GLPII bits of Register 5 (Status Register) are set high

Interrupts are not queued and if the interrupt event occurs before the driver has responded to it, no additional interrupts are generated. For example, Receive Interrupt [Bit 6 of Register 5 (Status Register)] indicates that one or more frames were transferred to the Host buffer. The driver must scan all descriptors, from the last recorded position to the first one owned by the DMA.

An interrupt is generated only once for simultaneous, multiple events. The driver must scan the Register 5 (Status Register) for the cause of the interrupt. The interrupt is not generated again unless a new interrupting event occurs, after the driver has cleared the appropriate bit in Register 5 (Status Register). For example, the controller generates a Receive interrupt [Bit 6 of Register 5 (Status Register)] and the driver begins reading Register 5 (Status Register). Next, Receive Buffer Unavailable [Bit 7 of Register 5 (Status Register)] occurs. The driver clears the Receive interrupt. Even then, the sbd_intr_o signal is not de-asserted, because of the active or pending Receive Buffer Unavailable interrupt.

An interrupt timer RIWT (Bits 7:0 in Register 9 (Receive Interrupt Watchdog Timer Register)) is given for flexible control of Receive Interrupt. When this Interrupt timer is programmed with a non-zero value, it gets activated as soon as the Rx DMA completes a transfer of a received frame to system memory without asserting the Receive Interrupt because it is not enabled in the corresponding Receive Descriptor (RDES1[31] in Table 8-8). When this timer runs out as per the programmed value, RI bit is set and the interrupt is asserted if the corresponding RI is enabled in Register 7 (Interrupt Enable Register). This timer gets disabled before it runs out, when a frame is transferred to memory and the RI is set because it is enabled for that descriptor.

9.2.5 Err Response

For any data transfer initiated by a DMA channel, if the slave replies with an error response, that DMA stops all operations and updates the error bits and the Fatal Bus Error bit in the Register 5 (Status Register). The DMA controller can resume operation only after soft resetting or hard resetting the DWC_gmac and re-initializing the DMA. This is applicable for all configurations with DMAs (GMAC-DMA, GMAC-AXI, and GMAC-AHB). In the GMAC-DMA configuration, the DMA receives the error response through mdc_error_i signal (shown in Figure 3-30). In the GMAC-AHB configuration, the error response is received through hresp_i (shown in Figure 3-13). In the GMAC-AXI configuration, the error response is received through rresp_m_i (shown in Figure 3-16) or bresp_m_i signal.

9.3 MAC Transaction Layer (MTL):

The MAC Transaction Layer provides FIFO memory to buffer and regulate the frames between the application system memory and the MAC. It also enables the data to be transferred between the application clock domain and the MAC clock domains. The MTL layer has 2 data paths, namely the Transmit path and the Receive Path. The data path for both directions is 32-bit, 64-bit, or 128-bit wide and operates with a simple FIFO protocol. The GMAC-MTL communicates with the application side with the Application Transmit Interface (ATI), Application Receive Interface (ARI), and the MAC Control Interface (MCI).

9.3.1 Transmit Path

the DMA controls all transactions for the transmit path through the ATI. The Ethernet frames read from the system memory are pushed into the FIFO by the DMA. The frame is then popped out and transferred to the MAC when triggered. When the end-of-frame is transferred, the status of the transmission is taken from the MAC and transferred back to the DMA.

The Transmit FIFO has a default depth of 2K bytes. FIFO-fill level is indicated to the DMA so that it can initiate a data fetch in required bursts from the system memory, using the AHB or AXI interface. The data from the AHB or AXI Master interface is pushed into the FIFO with the appropriate byte lanes qualified by the DMA. The DMA also indicates the start-of-frame (SOF) and end-of-frame (EOF) transfers along with a few sideband signals controlling the pad-insertion or CRC generation for that frame in the MAC.

The per-frame control bits, such as Automatic Pad or CRC Stripping disable, timestamp capture, and so forth are taken as sideband control inputs on the ATI, stored in a separate register FIFO, and passed on to the MAC transmitter when the corresponding frame data is read from the Transmit FIFO.

There are two modes of operation for popping data towards the MAC. In Threshold mode, as soon as the number of bytes in the FIFO crosses the configured threshold level (or when the end-of-frame is written before the threshold is crossed), the data is ready to be popped out and forwarded to the MAC. The threshold level is configured using the TTC bits of Register 0 (Bus Mode Register). In store-and-forward mode, the MTL pops the frame towards the MAC only when one or more of the following conditions are true:

- When a complete frame is stored in the FIFO
- When the TX FIFO becomes almost full
- When the ATI watermark becomes low. The watermark becomes low when the requested FIFO does not have space to accommodate the requested burst-length on the ATI.

Therefore, the MTL never stops in the store-and-forward mode even if the Ethernet frame length is bigger than the Tx FIFO size.

The application can flush the Transmit FIFO of all contents by setting the Bit 20 (FTF) of Register 6 (Operation Mode Register). This bit is self-clearing and initializes the FIFO pointers to the default state. If the FTF bit is set during a frame transfer from the MTL to the MAC, then the MTL stops further transfer as the FIFO is considered to be empty. Hence an underflow event occurs at the MAC transmitter and the corresponding Status word is forwarded to the DMA.

9.3.2 Receive Path

This module receives the frames given out by the MAC and pushes them into the Rx FIFO. The status (fill level) of this FIFO is indicated to the DMA once it crosses the configured Receive threshold (Bits [4:3] of Register 6 (Operation Mode Register)). The MTL also indicates the FIFO fill level so that the DMA can initiate pre-configured burst transfers towards the AHB interface.

During an Rx operation, the MTL is slaved to the MAC. The general sequence of Receive operation events is as follows:

1. When the MAC receives a frame, it pushes in data along with byte enables. The MAC also indicates the SOF and EOF. The MTL accepts the data and pushes it into the Rx FIFO. After the EOF is transferred, the MAC drives the status word, which is also pushed into the same Rx FIFO by the MTL.
2. When IEEE 1588 timestamping is enabled and the 64-bit timestamp is available along with the receive status, it is appended to the frame received from the MAC and is pushed into the Rx FIFO before the corresponding receive status word is written. Thus, in 32-bit data bus mode, two additional locations per

frame are taken for storing the timestamp in the Rx FIFO, while in 64-/128- bit mode, one additional location is taken.

3. The MTL_RX engine takes the data out of the FIFO and sends it to the DMA. In the default Cut- Through mode, when 64 bytes (configured with Bits [4:3] (RTC) of Register 6 (Operation Mode Register)) or a full packet of data are received into the FIFO, the MTL_RX engine pops out the data and indicates its availability to the DMA. Once the DMA initiates the transfer to the AHB or AXI interface, the MTL_RX engine continues to transfer data from the FIFO until a complete packet has been transferred. Upon completion of the EOF frame transfer, the MTL pops out the status word and sends it to the DMA controller. If the 64-bit timestamp is read out before the receive status, it is marked by the assertion of the ari_timestamp_val_o signal on the ARI. Otherwise, the status is given after the EOF data.
4. In the Rx FIFO store-and-forward mode (configured by the RSF bit of Register 6 (Operation Mode Register)), a frame is read out only after being written completely into the Receive FIFO. In this mode, all error frames are dropped (if the MAC is configured to do so) such that only valid frames are read out and forwarded to the application. In the Cut-Through mode, some error frames are not dropped, because the error status is received at the end-of-frame, by which time the start of that frame has already been read out of the FIFO.

9.4 MAC:

The MAC supports many interfaces towards the PHY chip. The PHY interface can be selected only once after reset. The MAC communicates with the application side with the MAC Transmit Interface (MTI), MAC Receive Interface (MRI) and the MAC Control Interface (MCI).

9.4.1 Transmission Path

Transmission is initiated when the MTL Application pushes in data with the SOF (mti_sof_i) signal asserted. When the SOF signal is detected, the MAC accepts the data and begins transmitting to the GMII or MII. The time required to transmit the frame data to the GMII or MII after the Application initiates transmission varies, depending on delay factors like IFG delay, time to transmit preamble or SFD, and any back-off delays for half-duplex mode. Until then, the MAC does not accept the data received from MTL by de-asserting the mti_rdy_o signal.

After the EOF (mti_eof_i) is transferred to the MAC, the MAC completes the normal transmission and gives the Status of Transmission to the MTL. If a normal collision (in half-duplex mode) occurs during transmission, the MAC makes valid the Transmit Status to the MTL. It then accepts and drops all further data until the next SOF is received. The MTL block should retransmit the same frame from SOF on observing a Retry request (in the Status) from the MAC.

The MAC issues an underflow status if the MTL is not able to provide the data continuously during the transmission. During the normal transfer of a frame from MTL, if the MAC receives a SOF without getting an EOF for the previous frame, then it ignores the SOF and considers the new frame as continuation of the previous frame.

9.4.2 Reception

A receive operation is initiated when the MAC detects an SFD on the GMII or MII. The MAC strips the preamble and SFD before proceeding to process the frame. The header fields are checked for the filtering and the FCS field used to verify the CRC for the frame. The received frame is stored in a shallow buffer until the address filtering is performed. The frame is dropped in the MAC if it fails the address filter.

9.5 Register

9.5.1 Register Map

The register maps in this section provide high-level summaries of each register or group of registers. The register names in the register maps (tables) are cross-referenced to the detailed register descriptions. To view the detailed description of a register, double-click the register name.

9.5.1.1 DMA Register Map

Table 10-3, provides the address map of the DMA registers. When you enable the AV feature, the information in below becomes specific to Channel 0.

Table 10-3 DMA Register Map

Register No.	Offset Address	Register Name and Description
0	0x1000	Register 0 (Bus Mode Register) Controls the Host Interface Mode.
1	0x1004	Register 1 (Transmit Poll Demand Register) Used by the host to instruct the DMA to poll the Transmit Descriptor list
2	0x1008	Register 2 (Receive Poll Demand Register) Used by the host to instruct the DMA to poll the Receive Descriptor list.
3	0x100C	Register 3 (Receive Descriptor List Address Register) Points the DMA to the start of the Receive Descriptor list.
4	0x1010	Register 4 (Transmit Descriptor List Address Register) Points the DMA to the start of the Transmit Descriptor list.
5	0x1014	Register 5 (Status Register) The Software driver (application) reads this register during interrupt service routine or polling to determine the status of the DMA.
6	0x1018	Register 6 (Operation Mode Register) Establishes the Receive and Transmit operating modes and command. Note: This register is valid and present in the GMAC-MTL configuration.
7	0x101C	Register 7 (Interrupt Enable Register) Enables the interrupts reported by the Status Register.
8	0x1020	Register 8 (Missed Frame and Buffer Overflow Counter Register) Contains the counters for discarded frames because no host Receive Descriptor was available or because of Receive FIFO Overflow.
9	0x1024	Register 9 (Receive Interrupt Watchdog Timer Register) Watchdog time-out for Receive Interrupt (RI) from DMA.
10	0x1028	Register 10 (AXI Bus Mode Register) Controls AXI Master behavior (mainly controls burst splitting and number of outstanding requests).
11	0x102C	Register 11 (AHB or AXI Status Register) Gives the idle status of the AHB Master interface in the GMAC-AHB configuration. Gives the idle status of the AXI master's read or write channel in the GMAC-AXI configuration.
12-17	0x1030-0x1044	Reserved
18	0x1048	Register 18 (Current Host Transmit Descriptor Register) Points to the start of current Transmit Descriptor read by the DMA.
19	0x104C	Register 19 (Current Host Receive Descriptor Register) Points to the start of current Receive Descriptor read by the DMA.
20	0x1050	Register 20 (Current Host Transmit Buffer Address Register) Points to the current Transmit Buffer address read by the DMA.
21	0x1054	Register 21 (Current Host Receive Buffer Address Register) Points to the current Receive Buffer address read by the DMA.

22	0x1058	Register 22 (HW Feature Register) Indicates the presence of the optional features of the core.

Table 10-4 provides the address map of the DMA registers for Channel 1. The address map in Table 6-2 is applicable only when you enable the AV Feature and select one or more additional Transmit channels. In the GMAC-MTL configuration, only registers from address 0x1160 to 0x1174 are present. In other configurations, all registers described in Table 10-4 are present.

Table 10-4 Channel 1 DMA Register Map

Register No.	Offset Address	Register Name and Description
64	0x1100	Register 64 (Channel 1 Bus Mode Register) Controls the Host Interface mode for Channel 1. For information about this register, see Register 0 (Bus Mode Register).
65	0x1104	Register 65 (Channel 1 Transmit Poll Demand Register) Used by the host to instruct the DMA to poll the Transmit Descriptor list. For information about this register, see Register 1 (Transmit Poll Demand Register).
66	0x1108	Register 66 (Channel 1 Receive Poll Demand Register) Used by the Host to instruct the DMA to poll the Receive Descriptor list. For information about this register, see Register 2 (Receive Poll Demand Register).
67	0x110C	Register 67 (Channel 1 Receive Descriptor List Address Register) Points the DMA to the start of the Receive Descriptor list. For information about this register, see Register 3 (Receive Descriptor List Address Register).
68	0x1110	Register 68 (Channel 1 Transmit Descriptor List Address Register) Points the DMA to the start of the Transmit Descriptor list. For information about this register, see Register 4 (Transmit Descriptor List Address Register).
69	0x1114	Register 69 (Channel 1 Status Register) The Software driver (application) reads this register during interrupt service routine or polling to determine the status of the DMA. Bits 29:26 are reserved for the Channel 1 Status Register. For information about this register, see Register 5 (Status Register).
70	0x1118	Register 70 (Channel 1 Operation Mode Register) Establishes the Receive and Transmit operating modes and command. For information about this register, see Register 6 (Operation Mode Register).
71	0x111C	Register 71 (Channel 1 Interrupt Enable Register) Enables the interrupts reported by the Status Register. For information about this register, see Register 7 (Interrupt Enable Register).
72	0x1120	Register 72 (Channel 1 Missed Frame and Buffer Overflow Counter Register) Contains the counters for discarded frames because no host Receive Descriptor was available, and discarded frames because of Receive FIFO Overflow. For information about this register, see Register 8 (Missed Frame and Buffer Overflow Counter Register).
73	0x1124	Register 73 (Channel 1 Receive Interrupt Watchdog Timer Register) Watchdog time-out for Receive Interrupt (RI) from DMA. For information about this register, see Register 9 (Receive Interrupt Watchdog Timer Register).
74-75	0x1128-0x112C	Reserved for this configuration.

76	0x1130	Register 76 (Channel 1 Slot Function Control and Status Register) Contains the control bits for slot function and its status for Channel 1 transmit path.
77-81	0x1134- 0x1144	Reserved
82	0x1148	Register 82 (Channel 1 Current Host Transmit Descriptor Register) Points to the start of current Transmit Descriptor read by the DMA. For information about this register, see Register 18 (Current Host Transmit Descriptor Register).
83	0x114C	Register 83 (Channel 1 Current Host Receive Descriptor Register) Points to the start of current Receive Descriptor read by the DMA. For information about this register, see Register 19 (Current Host Receive Descriptor Register).
84	0x1150	Register 84 (Channel 1 Current Host Transmit Buffer Address Register) Points to the current Transmit Buffer address read by the DMA. For information about this register, see Register 20 (Current Host Transmit Buffer Address Register).
95	0x1154	Register 85 (Channel 1 Current Host Receive Buffer Address Register) Points to the current Receive Buffer address read by the DMA. For information about this register, see Register 21 (Current Host Receive Buffer Address Register).
86-87	0x1158- 0x115C	Reserved The register at 0x1158 is a shadow register of Register 22 (HW Feature Register) at 0x1058 and it gives the same value on a read access.
88	0x1160	Register 88 (Channel 1 CBS Control Register) Controls the Channel 1 credit shaping operation on the transmit path.
89	0x1164	Register 89 (Channel 1 CBS Status Register) Provides the average traffic transmitted in Channel 1.
90	0x1168	Register 90 (Channel 1 idleSlopeCredit Register) Contains the idleSlope credit value required for the credit-based shaper algorithm for Channel 1.
91	0x116C	Register 91 (Channel 1 sendSlopeCredit Register) Contains the sendSlope credit value required for the credit-based shaper algorithm for Channel 1.
92	0x1170	Register 92 (Channel 1 hiCredit Register) Contains the hiCredit value required for the credit-based shaper algorithm for Channel 1.
93	0x1174	Register 93 (Channel 1 loCredit Register) Contains the loCredit value required for the credit-based shaper algorithm for Channel 1.

Table 10-5 provides the address map of the DMA registers for Channel 2. The address map in Table 6-3 is applicable only when you enable the AV feature and select two additional Transmit channels. In the GMAC-MTL configuration, only registers from address 0x1260 to 0x1274 are present. In other configurations, all registers mentioned in Table 10-5 are present.

Table 10- 5 Channel 2 DMA Register Map

Register No.	Offset Address	Register Name and Description
128	0x1200	Register 128 (Channel 2 Bus Mode Register) Controls the Host Interface mode for Channel 2. For information about this register, see Register 0 (Bus Mode Register).

129	0x1204	Register 129 (Channel 2 Transmit Poll Demand Register) Used by the host to instruct the DMA to poll the Transmit Descriptor list. For information about this register, see Register 1 (Transmit Poll Demand Register).
130	0x1208	Register 130 (Channel 2 Receive Poll Demand Register) Used by the Host to instruct the DMA to poll the Receive Descriptor list. For information about this register, see Register 2 (Receive Poll Demand Register).
131	0x120C	Register 131 (Channel 2 Receive Descriptor List Address Register) Points the DMA to the start of the Receive Descriptor list. For information about this register, see Register 3 (Receive Descriptor List Address Register).
132	0x1210	Register 132 (Channel 2 Transmit Descriptor List Address Register) Points the DMA to the start of the Transmit Descriptor List. For information about this register, see Register 4 (Transmit Descriptor List Address Register).
133	0x1214	Register 133 (Channel 2 Status Register) The software driver (application) reads this register during interrupt service routine or polling to determine the status of the DMA. Bits [29:26] are reserved for the Channel 2 Status Register. For information about this register, see Register 5 (Status Register).
134	0x1218	Register 134 (Channel 2 Operation Mode Register) Establishes the Receive and Transmit operating modes and command. For information about this register, see Register 6 (Operation Mode Register).
135	0x121C	Register 135 (Channel 2 Interrupt Enable Register) Enables the interrupts reported by the Status Register. For information about this register, see Register 7 (Interrupt Enable Register).
136	0x1220	Register 136 (Channel 2 Missed Frame and Buffer Overflow Counter Register) For information about this register, see Register 8 (Missed Frame and Buffer Overflow Counter Register).
137	0x1224	Register 137 (Channel 2 Receive Interrupt Watchdog Timer Register) Watchdog time-out for Receive Interrupt (RI) from DMA. For information about this register, see Register 9 (Receive Interrupt Watchdog Timer Register).
138-139	0x122C	Reserved for this configuration.
140	0x1230	Register 140 (Channel 2 Slot Function Control and Status Register) Contains the control bits for slot function and its status for Channel 2 transmit path. For information about this register, see Register 76 (Channel 1 Slot Function Control and Status Register).
141-145	0x1234- 0x1244	Reserved
146	0x1248	Register 146 (Channel 2 Current Host Transmit Descriptor Register) Points to the start of current Transmit Descriptor read by the DMA. For information about this register, see Register 18 (Current Host Transmit Descriptor Register).
147	0x124C	Register 147 (Channel 2 Current Host Receive Descriptor Register) Points to the start of current Receive Descriptor read by the DMA. For information about this register, see Register 19 (Current Host Receive Descriptor Register).

148	0x1230	Register 148 (Channel 2 Current Host Transmit Buffer Address Register) Points to the current Transmit Buffer address read by the DMA. For information about this register, see Register 20 (Current Host Transmit Buffer Address Register).
149	0x1234	Register 149 (Channel 2 Current Host Receive Buffer Address Register) Points to the current Receive Buffer address read by the DMA. For information about this register, see Register 21 (Current Host Receive Buffer Address Register).
150-151	0x1258-0x125C	Reserved The register at 0x1258 is a shadow register of Register 22 (HW Feature Register) at 0x1058 and it gives the same value on a read access.
152	0x1260	Register 152 (Channel 2 CBS Control Register) Controls the Channel 2 credit shaping operation on the transmit path. For information about this register, see Register 88 (Channel 1 CBS Control Register).
153	0x1264	Register 153 (Channel 2 CBS Status Register) Provides the average traffic transmitted in Channel 2. For information about this register, see Register 89 (Channel 1 CBS Status Register).
154	0x1268	Register 154 (Channel 2 idleSlopeCredit Register) Contains the idleSlope credit value required for the credit-based shaper algorithm for Channel 2. For information about this register, see Register 90 (Channel 1 idleSlopeCredit Register).
155	0x126C	Register 155 (Channel 2 sendSlopeCredit Register) Contains the sendSlope credit value required for the credit-based shaper algorithm for Channel 2. For information about this register, see Register 91 (Channel 1 sendSlopeCredit Register).
156	0x1270	Register 156 (Channel 2 hiCredit Register) Contains the hiCredit value required for the credit-based shaper algorithm for Channel 2. For information about this register, see Register 92 (Channel 1 hiCredit Register).
157	0x1274	Register 157 (Channel 2 loCredit Register) Contains the loCredit value required for the credit-based shaper algorithm for Channel 2. For information about this register, see Register 93 (Channel 1 loCredit Register).

9.5.1.2 GMA Register Map

Table 10-6 provides the address map of the GMAC registers. Most of the registers are optional and present in the source code only **if selected during configuration in coreConsultant**. If a register is not configured, then that address is reserved.

Table 10-6 GMAC Register Map

Register No.	Offset Address	Register Name and Description
0	0x0000	Register 0 (MAC Configuration Register) This is the operation mode register for the MAC.
1	0x0004	Register 1 (MAC Frame Filter) Contains the frame filtering controls.
2	0x0008	Register 2 (Hash Table High Register) Contains the higher 32 bits of the Multicast Hash table. This register is present only when you select the 64-bit Hash filter function in coreConsultant. (See Table 7-9.)

3	0x000C	Register 3 (Hash Table Low Register) Contains the lower 32 bits of the Multicast Hash table. This register is present only when you select the Hash filter function in coreConsultant. (See Table 7-9.)
4	0x0010	Register 4 (GMII Address Register) Controls the management cycles to an external PHY. This register is present only when you select the Station Management (MDIO) feature in coreConsultant. (See Table 7-26.)
5	0x0014	Register 5 (GMII Data Register) Contains the data to be written to or read from the PHY register. This register is present only when you select the Station Management (MDIO) feature in coreConsultant. (See Table 7-26.)
6	0x0018	Register 6 (Flow Control Register) Controls the generation of control frames.
7	0x001C	Register 7 (VLAN Tag Register) Identifies IEEE 802.1Q VLAN type frames.
8	0x0020	Register 8 (Version Register) Identifies the version of the Core.
9	0x0024	Register 9 (Debug Register) Gives the status of various internal blocks for debugging.
10	0x0028	Remote Wake-Up Frame Filter Register. This is the address through which the application writes or reads the remote wake-up frame filter registers (wkupfmfilter_reg). The wkupfmfilter_reg register is a pointer to eight wkupfmfilter_reg registers. The wkupfmfilter_reg register is loaded by sequentially loading the eight register values. Eight sequential writes to this address (0x0028) write all wkupfmfilter_reg registers. Similarly, eight sequential reads from this address (0x0028) read all wkupfmfilter_reg registers. This register contains the higher 16 bits of the seventh MAC address. This register is present only when you select the PMT module Remote Wake-up feature in coreConsultant.
11	0x002C	PMT Control and Status Register (on page 198) This register is present only when you select the PMT module in coreConsultant. (See Table 7-20.)
12	0x0030	Register 12 (LPI Control and Status Register) Controls the Low Power Idle (LPI) operations and provides the LPI status of the core. This register is present only when you select the Energy Efficient Ethernet feature in coreConsultant.
13	0x0034	Register 13 (LPI Timers Control Register) Controls the timeout values in LPI states. This register is present only when you select the Energy Efficient Ethernet feature in coreConsultant.
14	0x0038	Register 14 (Interrupt Status Register) Contains the interrupt status.
15	0x003C	Register 15 (Interrupt Mask Register) Contains the masks for generating the interrupts.
16	0x0040	Register 16 (MAC Address0 High Register) Contains the higher 16 bits of the first MAC address.
17	0x0044	Register 17 (MAC Address0 Low Register) Contains the lower 32 bits of the first MAC address.
18	0x0048	Register 18 (MAC Address1 High Register) Contains the higher 16 bits of the second MAC address. This register is present only when Enable MAC Address1 is selected in coreConsultant. (See Table 7-8).
19	0x004C	Register 19 (MAC Address1 Low Register) Contains the lower 32 bits of the second MAC address. This register is present only when Enable MAC Address1 is selected in coreConsultant. (See Table 7-8).

20	0x0050	MAC Address2 High Register Contains the lower 32 bits of the third MAC address. This register is present only when Enable MAC Address2 is selected in coreConsultant. (See Table 7-8).
21	0x0054	MAC Address2 Low Register Contains the lower 32 bits of the third MAC address. This register is present only when Enable MAC Address2 is selected in coreConsultant. (See Table 7-8).
22	0x0058	MAC Address3 High Register Contains the higher 16 bits of the fourth MAC address. This register is present only when Enable MAC Address3 is selected in coreConsultant. (See Table 7-8).
23	0x005C	MAC Address3 Low Register Contains the lower 32 bits of the fourth MAC address. This register is present only when Enable MAC Address3 is selected in coreConsultant. (See Table 7-8).
24	0x0060	MAC Address4 High Register Contains the higher 16 bits of the fifth MAC address. This register is present only when Enable MAC Address4 is selected in coreConsultant. (See Table 7-8).
25	0x0064	MAC Address4 Low Register Contains the lower 32 bits of the fifth MAC address. This register is present only when Enable MAC Address4 is selected in coreConsultant. (See Table 7-8).
26	0x0068	MAC Address5 High Register Contains the higher 16 bits of the sixth MAC address. This register is present only when Enable MAC Address5 is selected in coreConsultant. (See Table 7-8).
27	0x006C	MAC Address5 Low Register Contains the lower 32 bits of the sixth MAC address. This register is present only when Enable MAC Address5 is selected in coreConsultant. (See Table 7-8).
28	0x0070	MAC Address6 High Register Contains the higher 16 bits of the seventh MAC address.This register is present only when Enable MAC Address6 is selected in coreConsultant. (See Table 7-8).
29	0x0074	MAC Address6 Low Register Contains the lower 32 bits of the seventh MAC address. This register is present only when Enable MAC Address6 is selected in coreConsultant. (See Table 7-8).
30	0x0078	MAC Address7 High Register Contains the higher 16 bits of the eighth MAC address. This register is present only when Enable MAC Address7 is selected in coreConsultant. (See Table 7-8).
31	0x007C	MAC Address7 Low Register Contains the lower 32 bits of the eighth MAC address. This register is present only when Enable MAC Address7 is selected in coreConsultant. (See Table 7-8).
32	0x007C	MAC Address8 High Register Contains the higher 16 bits of the ninth MAC address. This register is present only when Enable MAC Address8 is selected in coreConsultant. (See Table 7-8).
33	0x0084	MAC Address8 Low Register Contains the lower 32 bits of the ninth MAC address. This register is present only when Enable MAC Address8 is selected in coreConsultant. (See Table 7-8).
34	0x0088	MAC Address9 High Register Contains the higher 16 bits of the 10th MAC address. This register is present only when Enable MAC Address9 is selected in coreConsultant. (See Table 7-8).
35	0x008C	MAC Address9 Low Register Contains the lower 32 bits of the 10th MAC address. This register is present only when Enable MAC Address9 is selected in coreConsultant. (See Table 7-8).
36	0x0090	MAC Address10 High Register Contains the higher 16 bits of the 11th MAC address. This register is present only when Enable MAC Address10 is selected in coreConsultant. (See Table 7-8).
37	0x0094	MAC Address10 Low Register Contains the lower 32 bits of the 11th MAC address. This register is present only when Enable MAC Address10 is selected in coreConsultant. (See Table 7-8).

38	0x0098	MAC Address11 High Register This register contains the higher 16 bits of the 12th MAC address. This register is present only when Enable MAC Address11 is selected in coreConsultant. (See Table 7-8).
39	0x009C	MAC Address11 Low Register Contains the lower 32 bits of the 12th MAC address. This register is present only when Enable MAC Address11 is selected in coreConsultant. (See Table 7-8).
40	0x00A0	MAC Address12 High Register Contains the higher 16 bits of the 13th MAC address. This register is present only when Enable MAC Address12 is selected in coreConsultant. (See Table 7-8).
41	0x00A4	MAC Address12 Low Register Contains the lower 32 bits of the 13th MAC address. This register is present only when Enable MAC Address12 is selected in coreConsultant. (See Table 7-8).
42	0x00A8	MAC Address13 High Register Contains the higher 16 bits of the 14th MAC address. This register is present only when Enable MAC Address13 is selected in coreConsultant. (See Table 7-8).
43	0x00AC	MAC Address13 Low Register Contains the lower 32 bits of the 14th MAC address. This register is present only when Enable MAC Address13 is selected in coreConsultant. (See Table 7-8).
44	0x00B0	MAC Address14 High Register Contains the higher 16 bits of the 15th MAC address. This register is present only when Enable MAC Address14 is selected in coreConsultant. (See Table 7-8).
45	0x00B4	MAC Address14 Low Register Contains the lower 32 bits of the 15th MAC address. This register is present only when Enable MAC Address14 is selected in coreConsultant. (See Table 7-8).
46	0x00B8	MAC Address15 High Register Contains the higher 16 bits of the 16th MAC address. This register is present only when Enable MAC Address15 is selected in coreConsultant. (See Table 7-8).
47	0x00BC	MAC Address15 Low Register Contains the lower 32 bits of the 16th MAC address. This register is present only when Enable MAC Address15 is selected in coreConsultant. (See Table 7-8).
48	0x00C0	Register 48 (AN Control Register) Enables and/or restarts auto-negotiation. This register also enables the Physical Coding Sublayer (PCS) loopback. This register is present only when you select the TBI, RTBI, or SGMII interface in coreConsultant.
49	0x00C4	Register 49 (AN Status Register) Indicates the link and auto-negotiation status. This register is present only when you select the TBI, RTBI, or SGMII interface in coreConsultant.
50	0x00C8	Register 50 (Auto-Negotiation Advertisement Register) This register is configured before auto-negotiation begins. It contains the advertised ability of the MAC. This register is present only when you select the TBI or RTBI interface in coreConsultant.
51	0x00CC	Register 51 (Auto-Negotiation Link Partner Ability Register) Contains the advertised ability of the link partner. Its value is valid after successful completion of auto-negotiation or when a new base page has been received (indicated in the Auto-Negotiation Expansion Register). This register is present only when you select the TBI or RTBI interface in coreConsultant.
52	0x00D0	Register 52 (Auto-Negotiation Expansion Register) Indicates whether a new base page has been received from the link partner. This register is present only when you select the TBI or RTBI interface in coreConsultant.
53	0x00D4	Register 53 (TBI Extended Status Register) Indicates all modes of operation of the MAC. This register is present only when you select the TBI or RTBI interface in coreConsultant.

54	0x00D8	Register 54 (SGMII/RGMII/SMII Status Register) Indicates the status signals received from the PHY through the SGMII, RGMII, or SMII interface. This register is present only when you select the SGMII, RGMII, or SMII interface in coreConsultant.
55-63	0x00DC – 0x00FC	Reserved
64-191	0x0100 – 0x02FC	MMC Register Map
192-255	0x0300 – 0x03FC	Reserved
256	0x0400	Register 256 (Layer 3 and Layer 4 Control Register 0) Controls the operations of the Layer 3 and Layer 4 frame filtering.
257	0x0404	Register 257 (Layer 4 Address Register 0) Layer 4 Port number field. It contains the 16-bit Source and Destination Port numbers of the TCP or UDP frame.
258-259	0x0408– 0x040C	Reserved
260	0x0410	Register 260 (Layer 3 Address 0 Register 0) Layer 3 Address field. For IPv4 frames, it contains the 32-bit IP Source Address field. For IPv6 frames, it contains Bits [31:0] of the 128-bit IP Source Address or Destination Address field.
261	0x0414	Register 261 (Layer 3 Address 1 Register 0) Layer 3 Address 1 field. For IPv4 frames, it contains the 32-bit IP Destination Address field. For IPv6 frames, it contains Bits [63:32] of the 128-bit IP Source Address or Destination Address field.
262	0x0418	Register 262 (Layer 3 Address 2 Register 0) Layer 3 Address 2 field. This register is reserved for IPv4 frames. For IPv6 frames, it contains Bits [95:64] of the 128-bit IP Source Address or Destination Address field.
263	0x041C	Register 263 (Layer 3 Address 3 Register 0) Layer 3 Address 3 field. This register is reserved for IPv4 frames. For IPv6 frames, it contains Bits [127:96] of the 128-bit IP Source Address or Destination Address field.
264-267	0x0420– 0x042C	Reserved
268	0x0430	Register 268 (Layer 3 and Layer 4 Control Register 1) Controls the operations of the Layer 3 and Layer 4 frame filtering. This register is similar to Register 256 (Layer 3 and Layer 4 Control Register 0).
269	0x0434	Register 269 (Layer 4 Address Register 1) Layer 4 Port number field. It contains the 16-bit Source and Destination Port numbers of TCP or UDP frame. This register is similar to Register 257 (Layer 4 Address Register 0).
270-271	0x0438– 0x043C	Reserved
272	0x0440	Register 272 (Layer 3 Address 0 Register 1) Layer 3 Address field. For IPv4 frames, it contains the 32-bit IP Source Address field. For IPv6 frames, it contains Bits [31:0] of the 128-bit IP Source Address or Destination Address field. This register is similar to Register 260 (Layer 3 Address 0 Register 0).
273	0x0444	Register 273 (Layer 3 Address 1 Register 1) Layer 3 Address 1 field. For IPv4 frames, it contains the 32-bit IP Destination Address field. For IPv6 frames, it contains Bits [63:32] of the 128-bit IP Source Address or Destination Address field. This register is similar to Register 261 (Layer 3 Address 1 Register 0).

274	0x0448	Register 274 (Layer 3 Address 2 Register 1) Layer 3 Address 2 field. This register is reserved for IPv4 frames. For IPv6 frames, it contains Bits [95:64] of the 128-bit IP Source Address or Destination Address field. This register is similar to Register 262 (Layer 3 Address 2 Register 0).
275	0x044C	Register 275 (Layer 3 Address 3 Register 1) Layer 3 Address 3 field. This register is reserved for IPv4 frames. For IPv6 frames, it contains Bits [127:96] of the 128-bit IP Source Address or Destination Address field. This register is similar to Register 263 (Layer 3 Address 3 Register 0).
2760279	0x450-0x45C	Reserved
280	0x0460	Register 280 (Layer 3 and Layer 4 Control Register 2) Controls the operations of the Layer 3 and Layer 4 frame filtering. This register is similar to Register 256 (Layer 3 and Layer 4 Control Register 0).
281	0x0460	Register 281 (Layer 4 Address Register 2) Layer 4 Port number field. It contains the 16-bit Source and Destination Port numbers of TCP or UDP frame. This register is similar to Register 257 (Layer 4 Address Register 0).
282-283	0x0468-0x046C	Reserved
284	0x0470	Register 284 (Layer 3 Address 0 Register 2) Layer 3 Address field. For IPv4 frames, it contains the 32-bit IP Source Address field. For IPv6 frames, it contains Bits [31:0] of the 128-bit IP Source Address or Destination Address field. This register is similar to Register 260 (Layer 3 Address 0 Register 0).
285	0x0474	Register 285 (Layer 3 Address 1 Register 2) Layer 3 Address 1 field. For IPv4 frames, it contains the 32-bit IP Destination Address field. For IPv6 frames, it contains Bits [63:32] of the 128-bit IP Source Address or Destination Address field. This register is similar to Register 261 (Layer 3 Address 1 Register 0).
286	0x0478	Register 286 (Layer 3 Address 2 Register 2) Layer 3 Address 2 field. This register is reserved for IPv4 frames. For IPv6 frames, it contains Bits [95:64] of the 128-bit IP Source Address or Destination Address field. This register is similar to Register 262 (Layer 3 Address 2 Register 0).
287	0x047C	Register 287 (Layer 3 Address 3 Register 2) Layer 3 Address 3 field. This register is reserved for IPv4 frames. For IPv6 frames, it contains Bits [127:96] of the 128-bit IP Source Address or Destination Address field. This register is similar to Register 263 (Layer 3 Address 3 Register 0).
288-291	0x0480-0x048C	Reserved
292	0x0490	Register 292 (Layer 3 and Layer 4 Control Register 3) Controls the operations of the Layer 3 and Layer 4 frame filtering. This register is similar to Register 256 (Layer 3 and Layer 4 Control Register 0).
293	0x0494	Register 293 (Layer 4 Address Register 3) Layer 4 Port number field. It contains the 16-bit Source and Destination Port numbers of TCP or UDP frame. This register is similar to Register 257 (Layer 4 Address Register 0).
294-295	0x0498-0x049C	Reserved
296	0x04A0	Register 296 (Layer 3 Address 0 Register 3) Layer 3 Address field. For IPv4 frames, it contains the 32-bit IP Source Address field. For IPv6 frames, it contains Bits [31:0] of the 128-bit IP Source Address or Destination Address field.

		This register is similar to Register 260 (Layer 3 Address 0 Register 0).
297	0x04A4	Register 285 (Layer 3 Address 1 Register 3) Layer 3 Address 1 field. For IPv4 frames, it contains the 32-bit IP Destination Address field. For IPv6 frames, it contains Bits [63:32] of the 128-bit IP Source Address or Destination Address field. This register is similar to Register 261 (Layer 3 Address 1 Register 0).
298	0x04A8	Register 286 (Layer 3 Address 2 Register 3) Layer 3 Address 2 field. This register is reserved for IPv4 frames. For IPv6 frames, it contains Bits [95:64] of the 128-bit IP Source Address or Destination Address field. This register is similar to Register 262 (Layer 3 Address 2 Register 0).
299	0x04AC	Register 287 (Layer 3 Address 3 Register 3) Layer 3 Address 3 field. This register is reserved for IPv4 frames. For IPv6 frames, it contains Bits [127:96] of the 128-bit IP Source Address or Destination Address field. This register is similar to Register 263 (Layer 3 Address 3 Register 0).
300-319	0x04B0-0x04FC	Reserved
320	0x0500	Register 320 (Hash Table Register 0) This register contains the first 32 bits of the hash table when the width of the Hash table is 128 bits or 256 bits.
321	0x0504	Register 321 (Hash Table Register 1) This register contains the second 32 bits of the hash table when the width of the Hash table is 128 bits or 256 bits. This register is similar to Register 320 (Hash Table Register 0).
322	0x0508	Register 322 (Hash Table Register 2) This register contains the third 32 bits of the hash table when the width of the Hash table is 128 bits or 256 bits. This register is similar to Register 320 (Hash Table Register 0).
323	0x0512	Register 323 (Hash Table Register 3) This register contains the fourth 32 bits of the hash table when the width of the Hash table is 128 bits or 256 bits. This register is similar to Register 320 (Hash Table Register 0).
324	0x0516	Register 324 (Hash Table Register 4) This register contains the fifth 32 bits of the hash table when the width of the Hash table is 256 bits. This register is similar to Register 320 (Hash Table Register 0).
325	0x0520	Register 325 (Hash Table Register 5) This register contains the sixth 32 bits of the hash table when the width of the Hash table is 256 bits. This register is similar to Register 320 (Hash Table Register 0).
326	0x0524	Register 326 (Hash Table Register 6) This register contains the seventh 32 bits of the hash table when the width of the Hash table is 256 bits. This register is similar to Register 320 (Hash Table Register 0).
327	0x0528	Register 327 (Hash Table Register 7) This register contains the eighth 32 bits of the hash table when the width of the Hash table is 256 bits. This register is similar to Register 320 (Hash Table Register 0).
328-352	0x0532-0x0580	Reserved
353	0x0584	Register 353 (VLAN Tag Inclusion or Replacement Register) This register contains the VLAN tag for insertion into or replacement in the transmit frames.

354	0x0588	Register 354 (VLAN Hash Table Register) This register contains the VLAN hash table.
355-447	0x058C-0x06FC	Reserved
448	0x0700	Register 448 (Timestamp Control Register) Controls the timestamp generation and update logic. This register is present only when IEEE1588 timestamping is enabled during coreConsultant configuration.
449	0x0704	Register 449 (Sub-Second Increment Register) Contains the 8-bit value by which the Sub-Second register is incremented. This register is present only when IEEE1588 timestamping is enabled without an external timestamp input.
450	0x0708	Register 450 (System Time – Seconds Register) Contains the lower 32 bits of the seconds field of the system time. This register is present only when IEEE1588 timestamping is enabled without an external timestamp input.
451	0x070C	Register 451 (System Time – Nanoseconds Register) Contains 32 bits of the nano-seconds field of the system time. This register is only present when IEEE1588 timestamping is enabled without an external timestamp input.
452	0x0710	Register 452 (System Time – Seconds Update Register) Contains the lower 32 bits of the seconds field to be written to, added to, or subtracted from the System Time value. This register is only present when IEEE1588 timestamping is enabled without an external timestamp input.
453	0x0714	Register 453 (System Time – Nanoseconds Update Register) Contains 32 bits of the nano-seconds field to be written to, added to, or subtracted from the System Time value. This register is only present when IEEE1588 timestamping is enabled without an external timestamp input.
454	0x0718	Register 454 (Timestamp Addend Register) This register is used by the software to readjust the clock frequency linearly to match the master clock frequency. This register is only present when IEEE1588 timestamping is enabled without an external timestamp input.
455	0x071C	Register 455 (Target Time Seconds Register) Contains the higher 32 bits of time to be compared with the system time for interrupt event generation or to start the PPS signal output generation. This register is present only when IEEE1588 timestamping is enabled without an external timestamp input.
456	0x0720	Register 456 (Target Time Nanoseconds Register) Contains the lower 32 bits of time to be compared with the system time for interrupt event generation or to start the PPS signal output generation. This register is present only when IEEE1588 timestamping is enabled without an external timestamp input.
457	0x0724	Register 457 (System Time - Higher Word Seconds Register) Contains the most significant 16-bits of the timestamp seconds value. This register is optional and can be selected using the parameter mentioned in “IEEE 1588 Timestamp Block” on page 448.
458	0x0728	Register 458 (Timestamp Status Register) Contains the PTP status. This register is available only when the advanced IEEE 1588 timestamp feature is selected.
459	0x072C	Register 459 (PPS Control Register) This register is used to control the interval of the PPS signal output. This register is available only when the advanced IEEE 1588 timestamp feature is selected.
460	0x0730	Register 460 (Auxiliary Timestamp – Nanoseconds Register) Contains the lower 32 bits (nano-seconds field) of the auxiliary timestamp register.
461	0x0734	Register 461 (Auxiliary Timestamp - Seconds Register) Contains the lower 32 bits of the Seconds field of the auxiliary timestamp register.

462	0x0738	Register 462 (AV MAC Control Register) Controls the AV traffic and queue management in the MAC Receiver. This register is present only when you select the AV feature in coreConsultant.
463-471	0x073C-0x075C	Reserved
472	0x0760	Register 472 (PPS0 Interval Register) Contains the number of units of sub-second increment value between the rising edges of PPS0 signal output. This register is available only when the flexible PPS feature is selected.
473	0x0764	Register 473 (PPS0 Width Register) Contains the number of units of sub-second increment value between the rising and corresponding falling edges of PPS0 signal output. This register is available only when the flexible PPS feature is selected.
474-479	0x0768-0x077C	Reserved
480	0x0780	Register 480 (PPS1 Target Time Seconds Register) Contains the higher 32 bits of time to be compared with the system time to generate the interrupt event or to start generating the PPS1 output signal. This register is present only when IEEE1588 timestamping is enabled without an external timestamp input and at least one additional PPS output is selected.
481	0x0781	Register 481 (PPS1 Target Time Nanoseconds Register) Contains the lower 32 bits of time to be compared with the system time to generate the interrupt event or to start generating the PPS1 output signal. This register is present only when IEEE1588 timestamping is enabled without an external timestamp input and at least one additional PPS output is selected.
482	0x0788	Register 482 (PPS1 Interval Register) Contains the number of units of sub-second increment value between the rising edges of the PPS1 output signal. This register is available only when the flexible PPS feature is selected and at least one additional PPS output is selected. This register is similar to Register 472 (PPS0 Interval Register).
483	0x078C	Register 483 (PPS1 Width Register) Contains the number of units of sub-second increment value between the rising and corresponding falling edges of the PPS1 output signal. This register is available only when the flexible PPS feature is selected and at least one additional PPS output is selected. This register is similar to Register 473 (PPS0 Width Register).
484-487	0x0790-0x079C	Reserved
488	0x07A0	Register 488 (PPS2 Target Time Seconds Register) Contains the higher 32 bits of time to be compared with the system time to generate the interrupt event or to start generating the PPS2 output signal. This register is present only when IEEE1588 timestamping is enabled without an external timestamp input and at least two additional PPS outputs are selected. This register is similar to Register 480 (PPS1 Target Time Seconds Register).
489	0x07A4	Register 489 (PPS2 Target Time Nanoseconds Register) Contains the lower 32 bits of time to be compared with the system time to generate the interrupt event or to start generating the PPS2 output signal. This register is present only when IEEE1588 timestamping is enabled without an external timestamp input and at least two additional PPS outputs are selected. This register is similar to Register 481 (PPS1 Target Time Nanoseconds Register).
490	0x07A8	Register 490 (PPS2 Interval Register) Contains the number of units of sub-second increment value between the rising edges of the PPS2 output signal. This register is available only when the flexible PPS feature is selected and at least two additional PPS outputs is selected. This register is similar to Register 472 (PPS0 Interval Register).

491	0x07AC	Register 491 (PPS2 Width Register) Contains the number of units of sub-second increment value between the rising and corresponding falling edges of the PPS2 output signal. This register is available only when the flexible PPS feature is selected and at least two additional PPS outputs are selected. This register is similar to Register 473 (PPS0 Width Register).
492-495	0x07B0- 0x07BC	Reserved
496	0x07C0	Register 496 (PPS3 Target Time Seconds Register) Contains the higher 32 bits of time to be compared with the system time to generate the interrupt event or to start generating the PPS3 output signal. This register is present only when IEEE1588 timestamping is enabled without an external timestamp input and three additional PPS outputs are selected. This register is similar to Register 480 (PPS1 Target Time Seconds Register).
497	0x07C4	Register 497 (PPS3 Target Time Nanoseconds Register) Contains the lower 32 bits of time to be compared with the system time to generate the interrupt event or to start generating the PPS3 output signal. This register is present only when IEEE1588 timestamping is enabled without an external timestamp input and three additional PPS outputs are selected. This register is similar to Register 481 (PPS1 Target Time Nanoseconds Register).
498	0x07C8	Register 498 (PPS3 Interval Register) Contains the number of units of sub-second increment value between the rising edges of the PPS3 output signal. This register is available only when the flexible PPS feature is selected and three additional PPS outputs is selected. This register is similar to Register 472 (PPS0 Interval Register).
499	0x07CC	Register 499 (PPS3 Width Register) Contains the number of units of sub-second increment value between the rising and corresponding falling edges of the PPS3 output signal. This register is available only when the flexible PPS feature is selected and three additional PPS outputs are selected. This register is similar to Register 473 (PPS0 Width Register).
500-511	0x07D0- 0x07FC	Reserved
512	0x0800	MAC Address 16 High Register Contains the higher 16 bits of the 17th MAC address. This register is present only when Enable MAC Address16 is selected in coreConsultant. (See Table 7-8).
513	0x0804	MAC Address 16 Low Register Contains the lower 32 bits of the 17th MAC address. This register is present only when Enable MAC Address16 is selected in coreConsultant. (See Table 7-8).
514	0x080C	MAC Address 17 High Register Contains the higher 16 bits of the 18th MAC address. This register is present only when Enable MAC Address17 is selected in coreConsultant. (See Table 7-8).
…		
543	0x87C	MAC Address 31 Low Register Contains the lower 32 bits of the 32nd MAC address. This register is present only when Enable MAC Address 31 is selected in coreConsultant. (See Table 7-8).
544	0x880	Register 544 (MAC Address32 High Register) Contains the higher 16 bits of the 33rd MAC address. This register is present only when Enable Additional 32 MAC Address Registers is selected in coreConsultant. (See Table 7-8).

545	0x884	MAC Address 32 Low Register Contains the lower 32 bits of the 33rd MAC address. This register is present only when Enable Additional 32 MAC Address Registers is selected in coreConsultant. (See Table 7-8).
…		
607	0x097C	MAC Address 63 Low Register Contains the lower 32 bits of the 64th MAC address. This register is present only when Enable Additional 32 MAC Address Registers is selected in coreConsultant. (See Table 7-8).
608	0x0980	MAC Address 64 High Register Contains the higher 32 bits of the 65th MAC address. This register is present only when Enable Additional 64 MAC Address Registers is selected in coreConsultant. (See Table 7-8).
609	0x0984	MAC Address 64 Low Register Contains the lower 32 bits of the 65th MAC address. This register is present only when Enable Additional 64 MAC Address Registers is selected in coreConsultant. (See Table 7-8).
…		
735	0x0B7C	MAC Address 127 Low Register Contains the lower 32 bits of the 128th MAC address. This register is present only when Enable Additional 64 MAC Address Registers is selected in coreConsultant. (See Table 7-8).
736-1023	0x0B80-0x0FFC	Reserved

9.5.2 Register Descriptions

The Access column of each register description that follows specifies how the application and the core can access the register fields of the CSRs.

The Access column uses the following conventions:

Read Only (RO)	Register field can only be read by the application. Writes to read-only fields have no effect.
Write Only (WO)	Register field can only be written by the application.
Read, Write, and Self Clear (R_W_SC)	Register field can be read and written by the application. The application can set this field by writing 1’b1 and can clear it by writing 1’b0.
Read, Write, and Self Clear (R_W_SC)	Register field can be read and written by the application. The bit can be cleared to 1’b0 either by the core itself (Self Clear) or by the application with a register write of 1’b0 (Write Clear).
Read, Self Set, and Write Clear (R_SS_WC)	Register field can be read by the application (Read), can be set to 1’b1 by the core on a certain internal event (Self Set), and can be cleared to 1’b0 by the application with a register write of 1’b1 (Write Clear). A register write of 1’b0 has no effect on this field. The conditions under which the core sets this field are explained in detail in the description of the field (for example, interrupt bits).
Read, Write Set, and Self Clear (R_WS_SC)	Register field can be read by the application (Read), can be set to 1’b1 by the application with a register write of 1’b1 (Write Set), and is cleared to 1’b0 by the core (Self Clear). The application cannot clear this type of field, and a register write of 1’b0 to this bit has no effect on this field. The conditions under which the core clears this field are explained in detail in the description of the field, for example, soft-reset signals.
Read, Self Set, and Self Clear or Write Clear (R_SS_SC_WC)	Register field can be read by the application (Read), can be set to 1’b1 by the core on a certain internal event (Self Set), and can be cleared to 1’b0 either by the core itself (Self Clear) or by the application with a register write of 1’b0 (Write Clear). A register write of 1’b1 to this bit has to no effect on this field. The conditions under which the core sets or clears this field are explained in detail in the description of the field.

Read Only and Write Trigger (RO_WT)	Register field can be read by the application, and when a write operation is performed with any data value, an event is triggered, as explained in the description of the field, for example, Tx Poll Demand register.
Read, Self Set, and Read Clear (R_SS_RC)	Register field can be read by the application (Read), can be set to 1'b1 by the core on a certain internal event (Self Set), and is automatically cleared to 1'b0 on a register read. A register write has no effect on this field. The conditions under which the core sets this field are explained in detail in the description of the field, for example, Overflow counter.
Read, Write, and Self Update (R_W_SU)	Register field can be read and written by the application. The register field updates itself based on the event. For example, system time in PTP configuration.
Read, Self Set, Self Clear, and Latch Low (R_SS_SC_LLO)	Register field can be read by the application (Read), can be set to 1'b1 by the core on a certain internal event (Self Set), can be cleared to 1'b0 by the core (Self Clear), and on reset, it is latched to low value (Latch Low). For example, link up and down status (LS: Link Status) in AN Status Register.

9.5.2.1 DMA Register Descriptions

This section defines the bits for each DMA register. The write data inputs to the DMA registers are qualified with the corresponding mci_be_i signal inputs (MCI interface). Thus, non-32 bit accesses are allowed as long as the address is Word-aligned. For the APB interface, only 32-bit accesses are possible, while for AHB slave interfaces, byte, half-word, or word accesses are possible.

9.5.2.1.1 Register 0 (Bus Mode Register)

Field	Field name	Description	Reset	Access
31:30	-	Reserved	00	RO
29:28	PRWG	Channel Priority Weights This field sets the priority weights for Channel 0 during the round-robin arbitration between the DMA channels for the system bus. • 00: The priority weight is 1. • 01: The priority weight is 2. • 10: The priority weight is 3. • 11: The priority weight is 4. This field is present in all DWC_gmac configurations except GMAC-AXI when you select the AV feature. Otherwise, this field is reserved and read-only (RO).	00	R_W
27	TXPR	Transmit Priority When set, this bit indicates that the transmit DMA has higher priority than the receive DMA during arbitration for the system-side bus. In the GMAC-AXI configuration, this bit is reserved and read-only (RO).	0	R_W
26	MB	Mixed Burst When this bit is set high and the FB bit is low, the AHB master interface starts all bursts of length more than 16 with INCR (undefined burst), whereas it reverts to fixed burst transfers (INCRx and SINGLE) for burst length of 16 and less. This bit is valid only in the GMAC-AHB configuration and reserved in all other configuration.	0	R_W
25	AAL	Address-Aligned Beats When this bit is set high and the FB bit is equal to 1, the AHB or AXI interface generates all bursts aligned to the start address LS bits. If the FB bit is equal to 0, the first burst (accessing the start address of data buffer) is not aligned, but subsequent bursts are aligned to the address.	0	R_W
24	8xPBL	8xPBL Mode When set high, this bit multiplies the programmed PBL value (Bits [22:17] and Bits [13:8]) eight times. Therefore, the DMA transfers the data in 8, 16, 32, 64, 128, and 256 beats depending on the PBL value. **Note:** This bit function is not backward compatible. Before release	0	R_W

		3.50a, this bit was 4xPBL.		
23	USP	Use Separate PBL When set high, this bit configures the Rx DMA to use the value configured in Bits [22:17] as PBL. The PBL value in Bits [13:8] is applicable only to the Tx DMA operations. When reset to low, the PBL value in Bits [13:8] is applicable for both DMA engines.	0	R_W
22:17	RPBL	Rx DMA PBL This field indicates the maximum number of beats to be transferred in one Rx DMA transaction. This is the maximum value that is used in a single block Read or Write. The Rx DMA always attempts to burst as specified in the RPBL bit each time it starts a Burst transfer on the host bus. You can program RPBL with values of 1, 2, 4, 8, 16, and 32. Any other value results in undefined behavior. This field is valid and applicable only when USP is set high	01H	R_W
16	FB	FB: Fixed Burst This bit controls whether the AHB Interface performs fixed burst transfers or not. When set, the AHB interface uses only SINGLE, INCR4, INCR8, or INCR16 during start of the normal burst transfers. When reset, the AHB interface uses SINGLE and INCR burst transfer operations.	0	R_W
15:14	PR	Priority Ratio These bits control the priority ratio in the weighted round-robin arbitration between the Rx DMA and Tx DMA. These bits are valid only when Bit 1 (DA) is reset. The priority ratio is Rx:Tx or Tx:Rx depending on whether Bit 27 (TXPR) is reset or set. • 00: The Priority Ratio is 1:1. • 01: The Priority Ratio is 2:1. • 10: The Priority Ratio is 3:1. • 11: The Priority Ratio is 4:1. In the GMAC-AXI configuration, these bits are reserved and read-only (RO).	00	R_W
13:8	PBL	Programmable Burst Length These bits indicate the maximum number of beats to be transferred in one DMA transaction. This is the maximum value that is used in a single block Read or Write. The DMA always attempts to burst as specified in PBL each time it starts a Burst transfer on the host bus. PBL can be programmed with permissible values of 1, 2, 4, 8, 16, and 32. Any other value results in undefined behavior. When USP is set high, this PBL value is applicable only for Tx DMA transactions. If the number of beats to be transferred is more than 32, then perform the following steps: 1. Set the 8xPBL mode. 2. Set the PBL For example, if the maximum number of beats to be transferred is 64, then first set 8xPBL to 1 and then set PBL to 8. The PBL values have the following limitation: The maximum number of possible beats (PBL) is limited by the size of the Tx FIFO and Rx FIFO in the MTL layer and the data bus width on the DMA. The FIFO has a constraint that the maximum beat supported is half the depth of the FIFO, except when specified. For different data bus widths and FIFO sizes, the valid PBL range (including x8 mode) is provided in the following list. If the PBL is common for both transmit and receive DMA, the minimum Rx FIFO and Tx FIFO depths must be considered. **Note**: In the half-duplex mode, the valid PBL range specified in the following list is applicable only for Tx FIFO. **• 32-Bit Data Bus Width**	01H	R_W

		- **128 Bytes FIFO Depth**: In the full-duplex mode, the valid PBL range is 16 or less. In the half-duplex mode, the valid PBL range is 8 or less for the 10 or 100 Mbps mode. - **256 Bytes FIFO Depth**: In the full-duplex and half-duplex (10 or 100 Mbps) modes, the valid PBL range is 32 or less. - **512 Bytes FIFO Depth**: In the full-duplex and half-duplex (10 or 100 Mbps) modes, the valid PBL range is 64 or less. - **1 KB FIFO Depth**: In the full-duplex mode, the valid PBL range is 128 or less. In the half-duplex mode, the valid PBL range is 128 or less in the 10 or 100 Mbps mode and 64 or less in the 1000 Mbps mode. **- 2 KB and Higher FIFO Depth**: All PBL values are supported in the full-duplex mode and half-duplex modes. **• 64-Bit Data Bus Width** - **128 Bytes FIFO Depth**: In the full-duplex mode, the valid PBL range is 8 or less. In the half-duplex mode, the valid PBL range is 4 or less for the 10 or 100 Mbps mode. - **256 Bytes FIFO Depth**: In the full-duplex and half-duplex (10 or 100 Mbps) modes, the valid PBL range is 16 or less. - **512 Bytes FIFO Depth**: In the full-duplex and half-duplex (10 or 100 Mbps) modes, the valid PBL range is 32 or less. - **1 KB FIFO Depth**: In the full-duplex mode, the valid PBL range is 64 or less. In the half-duplex mode, the valid PBL range is 64 or less in the 10 or 100 Mbps mode and 32 or less in the 1000 Mbps mode. - **2 KB FIFO Depth**: In the full-duplex and half-duplex (10 or 100 Mbps) modes, the valid PBL range is 128 or less. - **4 KB and Higher FIFO Depth**: All PBL values are supported in the full-duplex and half-duplex modes. **• 128-Bit Data Bus Width** - **128 Bytes FIFO Depth**: In the full-duplex mode, the valid PBL range is 4 or less. In the half-duplex mode, the valid PBL range is 2 or less for the 10 or 100 Mbps mode. - **256 Bytes FIFO Depth**: In the full-duplex and half-duplex (10 or 100 Mbps) modes, the valid PBL range is 8 or less. - **512 Bytes FIFO Depth**: In the full-duplex and half-duplex (10 or 100 Mbps) modes, the valid PBL range is 16 or less. - **1 KB FIFO Depth**: In the full-duplex mode, the valid PBL range is 32 or less. In the half-duplex mode, the valid PBL range is 32 or less in the 10 or 100 Mbps mode and 16 or less in the 1000-Mbps mode. **- 2 KB FIFO Depth**: In the full-duplex and half-duplex (10 or 100 Mbps) modes, the valid PBL range is 64 or less. **- 4 KB FIFO Depth**: In the full-duplex and half-duplex (10 or 100 Mbps) modes, the valid PBL range is 128 or less. - **8 KB and Higher FIFO Depth**: All PBL values are supported in the full-duplex and half-duplex modes.		
7	ATDS	Alternate Descriptor Size When set, the size of the alternate descriptor (described in “Alternate or Enhanced Descriptors” on page 501) increases to 32 bytes (8 DWORDS). This is required when the Advanced Timestamp feature or the IPC Full Checksum Offload Engine (Type 2) is enabled in the receiver. The enhanced descriptor is not required if the Advanced Timestamp and IPC Full Checksum Offload Engine (Type 2) features are not enabled. In such case, you can use the 16 bytes descriptor to save 4 bytes of memory. This bit is present only when you select the Alternate Descriptor feature and any one of the following features during core configuration: • Advanced Timestamp feature • IPC Full Checksum Offload Engine (Type 2) feature	0	R_W

		Otherwise, this bit is reserved and is read-only. When reset, the descriptor size reverts back to 4 DWORDs (16 bytes). This bit preserves the backward compatibility for the descriptor size. In versions prior to 3.50a, the descriptor size is 16 bytes for both normal and enhanced descriptors. In version 3.50a, descriptor size is increased to 32 bytes because of the Advanced Timestamp and IPC Full Checksum Offload Engine (Type 2) features.		
6;2	DSL	Descriptor Skip Length This bit specifies the number of Word, Dword, or Lword (depending on the 32-bit, 64-bit, or 128-bit bus) to skip between two unchained descriptors. The address skipping starts from the end of current descriptor to the start of next descriptor. When the DSL value is equal to zero, then the descriptor table is taken as contiguous by the DMA in Ring mode.	00H	R_W
1	DA	DMA Arbitration Scheme This bit specifies the arbitration scheme between the transmit and receive paths of Channel 0. • 0: Weighted round-robin with Rx:Tx or Tx:Rx. The priority between the paths is according to the priority specified in Bits [15:14] (PR) and priority weights specified in Bit 27 (TXPR). • 1: Fixed priority. The transmit path has priority over receive path when Bit 27 (TXPR) is set. Otherwise, receive path has priority over the transmit path. In the GMAC-AXI configuration, these bits are reserved and are read-only (RO).	0	R_W
0	SWR	Software Reset When this bit is set, the MAC DMA Controller resets the logic and all internal registers of the MAC. It is cleared automatically after the reset operation is complete in all of the DWC_gmac clock domains. Before reprogramming any register of the DWC_gmac, you should read a zero (0) value in this bit. **Note:** • The Software reset function is driven only by this bit. Bit 0 of Register 64 (Channel 1 Bus Mode Register) or Register 128 (Channel 2 Bus Mode Register) has no impact on the Software reset function. • The reset operation is completed only when all resets in all active clock domains are de-asserted. Therefore, it is essential that all PHY inputs clocks (applicable for the selected PHY interface) are present for the software reset completion. The time to complete the software reset operation depends on the frequency of the slowest active clock.	1	R_WS_SC

9.5.2.1.2 Register 1 (Transmit Poll Demand Register)

The Transmit Poll Demand register enables the Tx DMA to check whether or not the DMA owns the current descriptor. The Transmit Poll Demand command is given to wake up the Tx DMA if it is in the Suspend mode. The Tx DMA can go into the Suspend mode because of an Underflow error in a transmitted frame or the unavailability of descriptors owned by it. You can give this command anytime, and the Tx DMA resets this command when it again starts fetching the current descriptor from host memory.

Field	Field name	Description	Reset	Access
31:0	TPD	Transmit Poll Demand When these bits are written with any value, the DMA reads the current descriptor pointed to by Register 18 (Current Host Transmit	0000_0000H	RO_WT

		Descriptor Register). If that descriptor is not available (owned by the Host), the transmission returns to the Suspend state and the Bit 2 (TU) of Register 5 (Status Register) is asserted. If the descriptor is available, the transmission resumes.		

9.5.2.1.3 Register 2 (Receive Poll Demand Register)

The Receive Poll Demand register enables the Rx DMA to check for new descriptors. This command is given to wake up the Rx DMA from the SUSPEND state. The Rx DMA can go into the SUSPEND state only because of the unavailability of descriptors it owns.

Field	Field name	Description	Reset	Access
31:0	RPD	RPD: Receive Poll Demand When these bits are written with any value, the DMA reads the current descriptor pointed to by Register 19 (Current Host Receive Descriptor Register). If that descriptor is not available (owned by the Host), the reception returns to the Suspended state, and Bit 7 (RU) of Register 5 (Status Register) is not asserted. If the descriptor is available, the Rx DMA returns to the active state.	0000_0000H	RO_WT

9.5.2.1.4 Register 3 (Receive Descriptor List Address Register)

The Receive Descriptor List Address register points to the start of the Receive Descriptor List. The descriptor lists reside in the host's physical memory space and must be Word, Dword, or Lword-aligned (for 32-bit, 64- bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LS bits low. Writing to this register is permitted only when reception is stopped. When stopped, this register must be written to before the receive Start command is given.
You can write to this register only when Rx DMA has stopped, that is, Bit 1 (SR) is set to zero in Register 6 (Operation Mode Register). When stopped, this register can be written with a new descriptor list address. When you set the SR bit to 1, the DMA takes the newly programmed descriptor base address.
If this register is not changed when the SR bit is set to 0, then the DMA takes the descriptor address where it was stopped earlier.

Field	Field name	Description	Reset	Access
31:0	RDESLA	Start of Receive List This field contains the base address of the first descriptor in the Receive Descriptor list. The LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width are ignored and internally taken as all-zero by the DMA. Therefore, these LSB bits are read-only (RO).	0000_0000H	R_W

9.5.2.1.5 Register 4 (Transmit Descriptor List Address Register)

The Transmit Descriptor List Address register points to the start of the Transmit Descriptor List. The descriptor lists reside in the host's physical memory space and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LSB to low.
You can write to this register only when the Tx DMA has stopped, that is, Bit 13 (ST) is set to zero in Register 6 (Operation Mode Register). When stopped, this register can be written with a new descriptor list address. When you set the ST bit to 1, the DMA takes the newly programmed descriptor base address.

Field	Field name	Description	Reset	Access
31:0	TDESLA	Start of Transmit List This field contains the base address of the first descriptor in the Transmit Descriptor list. The LSB bits (1:0, 2:0, 3:0) for 32-bit, 64-bit, or 128-bit bus width are ignored and are internally taken as all-zero by the DMA. Therefore, these LSB bits are read-only (RO).	0000_0000H	R_W

9.5.2.1.6 Register 5 (Status Register)

The Status register contains all status bits that the DMA reports to the host. The Software driver reads this register during an interrupt service routine or polling. Most of the fields in this register cause the host to be interrupted. The bits of this register are not cleared when read. Writing 1'b1 to (unreserved) Bits [16:0] of this register clears these bits and writing 1'b0 has no effect. Each field (Bits[16:0]) can be masked by masking the appropriate bit in Register 7 (Interrupt Enable Register).

Field	Field name	Description	Reset	Access
31	-	Reserved	0	RO
30	GLPII -or- GTMSI	GLPII: GMAC LPI Interrupt (for Channel 0) This bit indicates an interrupt event in the LPI logic of the MAC. To reset this bit to 1'b0, the software must read the corresponding registers in the DWC_gmac to get the exact cause of the interrupt and clear its source. **Note**: GLPII status is given only in Channel 0 DMA register and is applicable only when the Energy Efficient Ethernet feature is enabled. Otherwise, this bit is reserved. When this bit is high, the interrupt signal from the MAC (sbd_intr_o) is high. -or- GTMSI: GMAC TMS Interrupt (for Channel 1 and Channel 2) This bit indicates an interrupt event in the traffic manager and scheduler logic of DWC_gmac. To reset this bit, the software must read the corresponding registers (Channel Status Register) to get the exact cause of the interrupt and clear its source. **Note**: GTMSI status is given only in Channel 1 and Channel 2 DMA register when the AV feature is enabled and corresponding additional transmit channels are present. Otherwise, this bit is reserved.	0	RO
29	TTI	Timestamp Trigger Interrupt This bit indicates an interrupt event in the Timestamp Generator block of the DWC_gmac. The software must read the corresponding registers in the DWC_gmac to get the exact cause of the interrupt and clear its source to reset this bit to 1'b0. When this bit is high, the interrupt signal from the DWC_gmac subsystem (sbd_intr_o) is high. This bit is applicable only when the IEEE 1588 Timestamp feature is enabled. Otherwise, this bit is reserved.	0	RO
28	GPI	GMAC PMT Interrupt This bit indicates an interrupt event in the PMT module of the DWC_gmac. The software must read the PMT Control and Status Register in the MAC to get the exact cause of interrupt and clear its source to reset this bit to 1'b0. The interrupt signal from the DWC_gmac subsystem (sbd_intr_o) is high when this bit is high. This bit is applicable only when the Power Management feature is enabled. Otherwise, this bit is reserved. **Note**: This interrupt is different from the pmt_intr_o interrupt.	0	RO
27	GMI	GMAC MMC Interrupt This bit reflects an interrupt event in the MMC module of the DWC_gmac. The software must read the corresponding registers in the DWC_gmac to get the exact cause of the interrupt and clear the source of interrupt to make this bit as 1'b0. The interrupt signal from the DWC_gmac subsystem (sbd_intr_o) is high when this bit is high. This bit is applicable only when the MAC Management Counters (MMC) are enabled. Otherwise, this bit is reserved.	0	RO
26	GLI	GMAC Line interface Interrupt	0	RO

		This bit reflects an interrupt event in the PCS (link change and AN complete), SMII (link change), or RGMII (link change) interface block of the DWC_gmac. The software must read the corresponding registers (Register 49 for PCS or Register 54 for SMII or RGMII) in the DWC_gmac to get the exact cause of the interrupt and clear the source of interrupt to make this bit as 1'b0. The interrupt signal from the DWC_gmac subsystem (sbd_intr_o) is high when this bit is high.		
25:23	EB	Error Bits This field indicates the type of error that caused a Bus Error, for example, error response on the AHB or AXI interface. This field is valid only when Bit 13 (FBI) is set. This field does not generate an interrupt. • Bit 23 - 1'b1: Error during data transfer by the Tx DMA - 1'b0: Error during data transfer by the Rx DMA • Bit 24 - 1'b1: Error during read transfer - 1'b0: Error during write transfer • Bit 25 - 1'b1: Error during descriptor access - 1'b0: Error during data buffer access	000	RO
22:20	TS	Transmit Process State This field indicates the Transmit DMA FSM state. This field does not generate an interrupt. • 3'b000: Stopped; Reset or Stop Transmit Command issued • 3'b001: Running; Fetching Transmit Transfer Descriptor • 3'b010: Running; Waiting for status • 3'b011: Running; Reading Data from host memory buffer and queuing it to transmit buffer (Tx FIFO) • 3'b100: TIME_STAMP write state • 3'b101: Reserved for future use • 3'b110: Suspended; Transmit Descriptor Unavailable or Transmit Buffer Underflow • 3'b111: Running; Closing Transmit Descriptor	000	RO
19:17	RS	Receive Process State This field indicates the Receive DMA FSM state. This field does not generate an interrupt. • 3'b000: Stopped: Reset or Stop Receive Command issued • 3'b001: Running: Fetching Receive Transfer Descriptor • 3'b010: Reserved for future use • 3'b011: Running: Waiting for receive packet • 3'b100: Suspended: Receive Descriptor Unavailable • 3'b101: Running: Closing Receive Descriptor • 3'b110: TIME_STAMP write state • 3'b111: Running: Transferring the receive packet data from receive buffer to host memory	000	RO
16	NIS	Normal Interrupt Summary Normal Interrupt Summary bit value is the logical OR of the following bits when the corresponding interrupt bits are enabled in Register 7 (Interrupt Enable Register): • Register 5[0]: Transmit Interrupt • Register 5[2]: Transmit Buffer Unavailable • Register 5[6]: Receive Interrupt • Register 5[14]: Early Receive Interrupt Only unmasked bits (interrupts for which interrupt enable is set in Register 7) affect the Normal Interrupt Summary bit. This is a sticky bit and must be cleared (by writing 1 to this bit) each time a corresponding bit, which causes NIS to be set, is cleared.	0	R_SS_WC

15	AIS	Abnormal Interrupt Summary Abnormal Interrupt Summary bit value is the logical OR of the following when the corresponding interrupt bits are enabled in Register 7 (Interrupt Enable Register): • Register 5[1]: Transmit Process Stopped • Register 5[3]: Transmit Jabber Timeout • Register 5[4]: Receive FIFO Overflow • Register 5[5]: Transmit Underflow • Register 5[7]: Receive Buffer Unavailable • Register 5[8]: Receive Process Stopped • Register 5[9]: Receive Watchdog Timeout • Register 5[10]: Early Transmit Interrupt • Register 5[13]: Fatal Bus Error Only unmasked bits affect the Abnormal Interrupt Summary bit. This is a sticky bit and must be cleared (by writing 1 to this bit) each time a corresponding bit, which causes AIS to be set, is cleared.	0	R_SS_WC
14	ERI	Early Receive Interrupt This bit indicates that the DMA had filled the first data buffer of the packet. The Bit 6 (RI) of this register automatically clears this bit.	0	R_SS_WC
13	FBI	Fatal Bus Error Interrupt This bit indicates that a bus error occurred, as described in Bits [25:23]. When this bit is set, the corresponding DMA engine disables all of its bus accesses.	0	R_SS_WC
12:11	-	Reserved	00	RO
10	ETI	Early Transmit Interrupt This bit indicates that the frame to be transmitted is fully transferred to the MTL Transmit FIFO.	0	R_SS_WC
9	RWT	Receive Watchdog Timeout This bit is asserted when a frame with length greater than 2,048 bytes is received (10, 240 when Jumbo Frame mode is enabled).	0	R_SS_WC
8	RPS	Receive Process Stopped This bit is asserted when the Receive Process enters the Stopped state.	0	R_SS_WC
7	RU	Receive Buffer Unavailable This bit indicates that the host owns the Next Descriptor in the Receive List and the DMA cannot acquire it. The Receive Process is suspended. To resume processing Receive descriptors, the host should change the ownership of the descriptor and issue a Receive Poll Demand command. If no Receive Poll Demand is issued, the Receive Process resumes when the next recognized incoming frame is received. This bit is set only when the previous Receive Descriptor is owned by the DMA.	0	R_SS_WC
6	RI	Receive Interrupt This bit indicates that the frame reception is complete. When reception is complete, the Bit 31 of RDES1 (Disable Interrupt on Completion) is reset in the last Descriptor, and the specific frame status information is updated in the descriptor. The reception remains in the Running state.	0	R_SS_WC
5	UNF	Transmit Underflow This bit indicates that the Transmit Buffer had an Underflow during frame transmission. Transmission is suspended and an Underflow Error TDES0[1] is set.	0	R_SS_WC
4	OVF	Receive Overflow This bit indicates that the Receive Buffer had an Overflow during frame reception. If the partial frame is transferred to the application, the overflow status is set in RDES0[11].	0	R_SS_WC

3	TJT	Transmit Jabber Timeout This bit indicates that the Transmit Jabber Timer expired, which happens when the frame size exceeds 2,048 (10,240 bytes when the Jumbo frame is enabled). When the Jabber Timeout occurs, the transmission process is aborted and placed in the Stopped state. This causes the Transmit Jabber Timeout TDES0[14] flag to assert.	0	R_SS_WC
2	TU	Transmit Buffer Unavailable This bit indicates that the host owns the Next Descriptor in the Transmit List and the DMA cannot acquire it. Transmission is suspended. Bits[22:20] explain the Transmit Process state transitions. To resume processing Transmit descriptors, the host should change the ownership of the descriptor by setting TDES0[31] and then issue a Transmit Poll Demand command.	0	R_SS_WC
1	TPS	Transmit Process Stopped This bit is set when the transmission is stopped.	0	R_SS_WC
0	TI	Transmit Interrupt This bit indicates that the frame transmission is complete. When transmission is complete, the Bit 31 (Interrupt on Completion) of TDES1 is reset in the first descriptor, and the specific frame status information is updated in the descriptor.	0	R_SS_WC

9.5.2.1.7 Register 6 (Operation Mode Register)

The Operation Mode register establishes the Transmit and Receive operating modes and commands. This register should be the last CSR to be written as part of the DMA initialization. This register is also present in the GMAC-MTL configuration with bits unused and reserved 24, 13, 2, and 1.

Field	Field name	Description	Reset	Access
31:27	-	Reserved	0H	RO
26	DT	Disable Dropping of TCP/IP Checksum Error Frames When this bit is set, the MAC does not drop the frames which only have errors detected by the Receive Checksum Offload engine. Such frames do not have any errors (including FCS error) in the Ethernet frame received by the MAC but have errors only in the encapsulated payload. When this bit is reset, all error frames are dropped if the FEF bit is reset. If the IPC Full Checksum Offload Engine (Type 2) is disabled, this bit is reserved (RO with value 1'b0).	0	R_W
25	RSF	Receive Store and Forward When this bit is set, the MTL reads a frame from the Rx FIFO only after the complete frame has been written to it, ignoring the RTC bits. When this bit is reset, the Rx FIFO operates in the cut-through mode, subject to the threshold specified by the RTC bits.	0	R_W
24	DFF	Disable Flushing of Received Frames When this bit is set, the Rx DMA does not flush any frames because of the unavailability of receive descriptors or buffers as it does normally when this bit is reset. (See “Receive Process Suspended” on page 77.) This bit is reserved (and RO) in the GMAC-MTL configuration.	0	R_W
23	RFA[2]	MSB of Threshold for Activating Flow Control If the DWC_gmac is configured for an Rx FIFO size of 8 KB or more, this bit (when set) provides additional threshold levels for activating the flow control in both half-duplex and full-duplex modes. This bit (as Most Significant Bit) along with the RFA (Bits [10:9]) gives the following thresholds for activating flow control:	0	R_W

		•100: Full minus 5 KB, that is, FULL—5KB. •101: Full minus 6 KB, that is, FULL—6KB. •110: Full minus 7 KB, that is, FULL—7KB. • 111: Reserved This bit is reserved (and RO) if the Rx FIFO is 4 KB or less deep.		
22	RFD[2]	MSB of Threshold for Deactivating Flow Control If the DWC_gmac is configured for Rx FIFO size of 8 KB or more, this bit (when set) provides additional threshold levels for deactivating the flow control in both half-duplex and full-duplex modes. This bit (as Most Significant Bit) along with the RFD (Bits [12:11]) gives the following thresholds for deactivating flow control: •100: Full minus 5 KB, that is, FULL—5KB •101: Full minus 6 KB, that is, FULL—6KB •110: Full minus 7 KB, that is, FULL—7KB • 111: Reserved This bit is reserved (and RO) if the Rx FIFO is 4 KB or less deep.	0	R_W
21	TSF	Transmit Store and Forward When this bit is set, transmission starts when a full frame resides in the MTL Transmit FIFO. When this bit is set, the TTC values specified in Bits [16:14] are ignored. This bit should be changed only when the transmission is stopped.	0	R_W
20	FTF	Flush Transmit FIFO When this bit is set, the transmit FIFO controller logic is reset to its default values and thus all data in the Tx FIFO is lost or flushed. This bit is cleared internally when the flushing operation is complete. The Operation Mode register should not be written to until this bit is cleared. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt frame transmission. **Note:** The flush operation is complete only when the Tx FIFO is emptied of its contents and all the pending Transmit Status of the transmitted frames are accepted by the host. In order to complete this flush operation, the PHY transmit clock (clk_tx_i) is required to be active.	0	R_WS_SC
19:17	-	Reserved	000	R_W
16:14	TTC	Transmit Threshold Control These bits control the threshold level of the MTL Transmit FIFO. Transmission starts when the frame size within the MTL Transmit FIFO is larger than the threshold. In addition, full frames with a length less than the threshold are also transmitted. These bits are used only when Bit 21 (TSF) is reset. • 000: 64 • 001: 128 • 010: 192 • 011: 256 • 100: 40 • 101: 32 • 110: 24 • 111: 16	000	R_W
13	ST	Start or Stop Transmission Command When this bit is set, transmission is placed in the Running state, and the DMA checks the Transmit List at the current position for a frame to be transmitted. Descriptor acquisition is attempted either from the current position in the list, which is the Transmit List Base Address set by Register 4 (Transmit Descriptor List Address Register), or from the position retained when transmission was stopped previously. If the DMA does not own the current descriptor, transmission enters the Suspended state		R_W

		and Bit 2 (Transmit Buffer Unavailable) of Register 5 (Status Register) is set. The Start Transmission command is effective only when transmission is stopped. If the command is issued before setting Register 4 (Transmit Descriptor List Address Register), then the DMA behavior is unpredictable. When this bit is reset, the transmission process is placed in the Stopped state after completing the transmission of the current frame. The Next Descriptor position in the Transmit List is saved, and it becomes the current position when transmission is restarted. To change the list address, you need to program Register 4 (Transmit Descriptor List Address Register) with a new value when this bit is reset. The new value is considered when this bit is set again. The stop transmission command is effective only when the transmission of the current frame is complete or the transmission is in the Suspended state.		
12:11	RFD	Threshold for Deactivating Flow Control (in half-duplex and full-duplex modes) These bits control the threshold (Fill-level of Rx FIFO) at which the flow control is de-asserted after activation. • 00: Full minus 1 KB, that is, FULL—1KB • 01: Full minus 2 KB, that is, FULL—2KB • 10: Full minus 3 KB, that is, FULL—3KB • 11: Full minus 4 KB, that is, FULL—4KB The de-assertion is effective only after flow control is asserted. If the Rx FIFO is 8 KB or more, an additional Bit (RFD[2]) is used for more threshold levels as described in Bit 22. These bits are reserved and read-only when the Rx FIFO depth is less than 4 KB.	00	R_W
10:9	RFA	Threshold for Activating Flow Control (in half-duplex and full-duplex modes) These bits control the threshold (Fill level of Rx FIFO) at which the flow control is activated. •00: Full minus 1 KB, that is, FULL—1KB. •01: Full minus 2 KB, that is, FULL—2KB. •10: Full minus 3 KB, that is, FULL—3KB. •11: Full minus 4 KB, that is, FULL—4KB. These values only apply to Rx FIFOs of 4 KB or more when the EFC bit is set high. If the Rx FIFO is 8 KB or more, an additional Bit (RFA[2]) is used for more threshold levels as described in Bit 23. These bits are reserved and read-only when the depth of Rx FIFO is less than 4 KB.	0	R_W
8	EFC	Enable HW Flow Control When this bit is set, the flow control signal operation based on the fill-level of Rx FIFO is enabled. When reset, the flow control operation is disabled. This bit is not used (reserved and always reset) when the Rx FIFO is less than 4 KB.	0	R_W
7	FEF	Forward Error Frames When this bit is reset, the Rx FIFO drops frames with error status (CRC error, collision error, GMII_ER, giant frame, watchdog timeout, or overflow). However, if the start byte (write) pointer of a frame is already transferred to the read controller side (in Threshold mode), then the frame is not dropped. In the GMAC-MTL configuration in which the Frame Length FIFO is also enabled during core configuration, the Rx FIFO drops the error frames if that frame's start byte is not transferred (output) on the ARI bus. When the FEF bit is set, all frames except runt error frames are forwarded to the DMA. If the Bit 25 (RSF) is set and the Rx FIFO overflows when a partial frame is written, then the frame is dropped irrespective of the FEF bit setting. However, if the Bit 25 (RSF) is reset and the Rx FIFO overflows when a partial frame is	0	R_W

		written, then a partial frame may be forwarded to the DMA.		
6	FUF	Forward Undersized Good Frames When set, the Rx FIFO forwards Undersized frames (frames with no Error and length less than 64 bytes) including pad-bytes and CRC. When reset, the Rx FIFO drops all frames of less than 64 bytes, unless a frame is already transferred because of the lower value of Receive Threshold, for example, RTC = 01.	0	R_W
5	-	Reserved	0	RO
4:3	RTC	Receive Threshold Control These two bits control the threshold level of the MTL Receive FIFO. Transfer (request) to DMA starts when the frame size within the MTL Receive FIFO is larger than the threshold. In addition, full frames with length less than the threshold are automatically transferred. The value of 11 is not applicable if the configured Receive FIFO size is 128 bytes. These bits are valid only when the RSF bit is zero, and are ignored when the RSF bit is set to 1. • 00: 64 • 01: 32 • 10: 96 • 11: 128	00	R_W
2	OSF	Operate on Second Frame When this bit is set, it instructs the DMA to process the second frame of the Transmit data even before the status for the first frame is obtained.	0	R_W
1	SR	Start or Stop Receive When this bit is set, the Receive process is placed in the Running state. The DMA attempts to acquire the descriptor from the Receive list and processes the incoming frames. The descriptor acquisition is attempted from the current position in the list, which is the address set by the Register 3 (Receive Descriptor List Address Register) or the position retained when the Receive process was previously stopped. If the DMA does not own the descriptor, reception is suspended and Bit 7 (Receive Buffer Unavailable) of Register 5 (Status Register) is set. The Start Receive command is effective only when the reception has stopped. If the command is issued before setting Register 3 (Receive Descriptor List Address Register), the DMA behavior is unpredictable. When this bit is cleared, the Rx DMA operation is stopped after the transfer of the current frame. The next descriptor position in the Receive list is saved and becomes the current position after the Receive process is restarted. The Stop Receive command is effective only when the Receive process is in either the Running (waiting for receive packet) or in the Suspended state.	0	R_W
0	-	Reserved	0	RO

9.5.2.1.8 Register 7 (Interrupt Enable Register)

The Interrupt Enable register enables the interrupts reported by Register 5 (Status Register). Setting a bit to 1’b1 enables a corresponding interrupt. After a hardware or software reset, all interrupts are disabled.

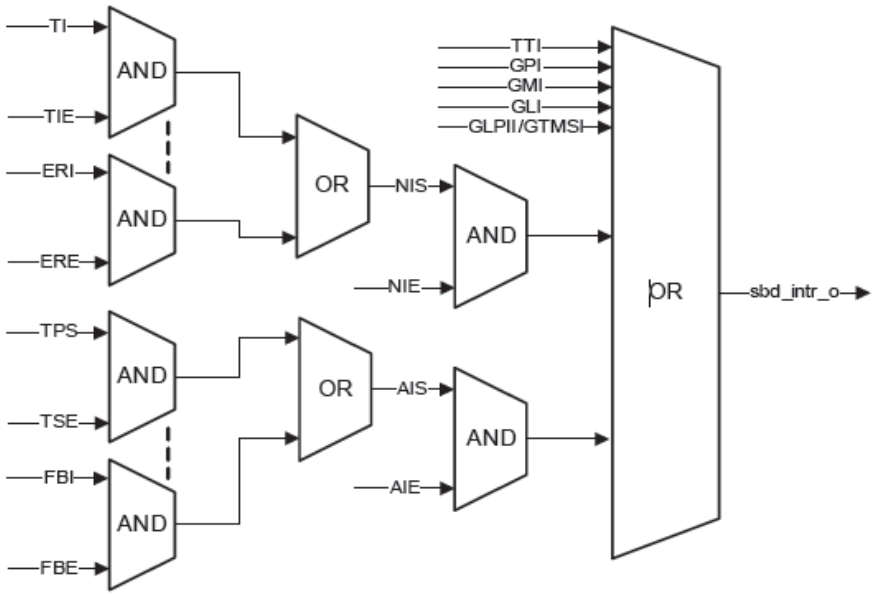
Field	Field name	Description	Reset	Access
31:17	-	Reserved	0000H	RO
16	NIE	Normal Interrupt Summary Enable When this bit is set, normal interrupt summary is enabled. When this bit is reset, normal interrupt summary is disabled. This bit enables the following	0	R_W

		interrupts in Register 5 (Status Register): • Register 5[0]: Transmit Interrupt • Register 5[2]: Transmit Buffer Unavailable • Register 5[6]: Receive Interrupt • Register 5[14]: Early Receive Interrupt		
15	AIE	Abnormal Interrupt Summary Enable When this bit is set, abnormal interrupt summary is enabled. When this bit is reset, the abnormal interrupt summary is disabled. This bit enables the following interrupts in Register 5 (Status Register): • Register 5[1]: Transmit Process Stopped • Register 5[3]: Transmit Jabber Timeout • Register 5[4]: Receive Overflow • Register 5[5]: Transmit Underflow • Register 5[7]: Receive Buffer Unavailable • Register 5[8]: Receive Process Stopped • Register 5[9]: Receive Watchdog Timeout • Register 5[10]: Early Transmit Interrupt • Register 5[13]: Fatal Bus Error	0	R_W
14	ERE	Early Receive Interrupt Enable When this bit is set with Normal Interrupt Summary Enable (Bit 16), the Early Receive Interrupt is enabled. When this bit is reset, the Early Receive Interrupt is disabled.	0	R_W
13	FBE	Fatal Bus Error Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Fatal Bus Error Interrupt is enabled. When this bit is reset, the Fatal Bus Error Enable Interrupt is disabled.	0	R_W
12:11	-	Reserved	00	RO
10	ETE	Early Transmit Interrupt Enable When this bit is set with an Abnormal Interrupt Summary Enable (Bit 15), the Early Transmit Interrupt is enabled. When this bit is reset, the Early Transmit Interrupt is disabled.	0	R_W
9	RWE	Receive Watchdog Timeout Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Receive Watchdog Timeout Interrupt is enabled. When this bit is reset, the Receive Watchdog Timeout Interrupt is disabled.	0	R_W
8	RSE	Receive Stopped Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Receive Stopped Interrupt is enabled. When this bit is reset, the Receive Stopped Interrupt is disabled.	0	R_W
7	RUE	Receive Buffer Unavailable Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Receive Buffer Unavailable Interrupt is enabled. When this bit is reset, the Receive Buffer Unavailable Interrupt is disabled.	0	R_W
6	RIE	Receive Interrupt Enable When this bit is set with Normal Interrupt Summary Enable (Bit 16), the Receive Interrupt is enabled. When this bit is reset, the Receive Interrupt is disabled.	0	R_W
5	UNE	Underflow Interrupt Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Transmit Underflow Interrupt is enabled. When this bit is reset, the Underflow Interrupt is disabled.	0	R_W
4	OVE	Overflow Interrupt Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Receive Overflow Interrupt is enabled. When this bit is reset, the Overflow Interrupt is disabled.	0	R_W
3	THE	Transmit Jabber Timeout Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Transmit Jabber Timeout Interrupt is enabled. When this	0	R_W

		bit is reset, the Transmit Jabber Timeout Interrupt is disabled.		
2	TUE	Transmit Buffer Unavailable Enable When this bit is set with Normal Interrupt Summary Enable (Bit 16), the Transmit Buffer Unavailable Interrupt is enabled. When this bit is reset, the Transmit Buffer Unavailable Interrupt is disabled.	0	R_W
1	TSE	Transmit Stopped Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Transmission Stopped Interrupt is enabled. When this bit is reset, the Transmission Stopped Interrupt is disabled	0	R_W
0	TIE	Transmit Interrupt Enable When this bit is set with Normal Interrupt Summary Enable (Bit 16), the Transmit Interrupt is enabled. When this bit is reset, the Transmit Interrupt is disabled.	0	R_W

The sbd_intr_o interrupt is generated as shown in Table 10-7 It is asserted only when the TTI, GPI, GMI, GLI, or GLPII bit of the DMA Status register is asserted, or when the NIS or AIS Status bit is asserted and the corresponding Interrupt Enable bits (NIE or AIE) are enabled.

Table 10-7 Sbd_intr_o Generation



Note: Signals NIS and AIS are registered.

9.5.2.1.9 Register 8 (Missed Frame and Buffer Overflow Counter Register)

The DMA maintains two counters to track the number of frames missed during reception. This register reports the current value of the counter. The counter is used for diagnostic purposes. Bits[15:0] indicate missed frames because of the host buffer being unavailable. Bits[27:17] indicate missed frames because of buffer overflow conditions (MTL and MAC) and runt frames (good frames of less than 64 bytes) dropped by the MTL.

Field	Field name	Description	Reset	Access
31:29	-	Reserved	000	RO
28	OVFCNTOVF	Overflow bit for FIFO Overflow Counter	0	R_SS_RC
27:17	OVFFRMCNT	This field indicates the number of frames missed by the application. This counter is incremented each time the	000H	R_SS_RC

		MTL asserts the sideband signal mtl_rxoverflow_o. The counter is cleared when this register is read with mci_be_i[2] at 1'b1.		
16	MISCNTOVF	Overflow bit for Missed Frame Counter	0	R_SS_RC
15:0	MISFRMCNT	This field indicates the number of frames missed by the controller because of the Host Receive Buffer being unavailable. This counter is incremented each time the DMA discards an incoming frame. The counter is cleared when this register is read with mci_be_i[0] at 1'b1.	0000H	R_SS_RC

9.5.2.1.10 Register 9 (Receive Interrupt Watchdog Timer Register)

This register, when written with a non-zero value, enables the watchdog timer for the Receive Interrupt (Bit 6) of Register 5 (Status Register).

Field	Field name	Description	Reset	Access
31:8	-	Reserved	000000H	RO
7:0	RIWT	RI Watchdog Timer Count This bit indicates the number of system clock cycles multiplied by 256 for which the watchdog timer is set. The watchdog timer gets triggered with the programmed value after the Rx DMA completes the transfer of a frame for which the RI status bit is not set because of the setting in the corresponding descriptor RDES1[31]. When the watchdog timer runs out, the RI bit is set and the timer is stopped. The watchdog timer is reset when the RI bit is set high because of automatic setting of RI as per RDES1[31] of any received frame.	00H	R_W

9.5.2.1.11 Register 11 (AHB Status Register)

This register provides the active status of the read and write channels of the AHB Master interface or AXI interface. This register is useful for debugging purposes. In addition, this register is valid only in the Channel 0 DMA when multiple channels are present in the AV mode.

Field	Field name	Description	Reset	Access
31:1	-	Reserved	0000_0000H	RO
0	AXWHSTS	AHB Master Status When high, it indicates that AXI Master's write channel is active and transferring data in the GMAC-AXI configuration. In the GMAC-AHB configuration, it indicates that the AHB master interface FSMs are in the non-idle state.	0	RO

9.5.2.1.12 Register 18 (Current Host Transmit Descriptor)

The Current Host Transmit Descriptor register points to the start address of the current Transmit Descriptor read by the DMA.

Field	Field name	Description	Reset	Access
31:0	CURTDESAPTR	Host Transmit Descriptor Address Pointer Cleared on Reset. Pointer updated by the DMA during operation.	0000_0000H	RO

9.5.2.1.13 Register 19 (Current Host Receive Descriptor)

The Current Host Receive Descriptor register points to the start address of the current Receive Descriptor read by the DMA.

Field	Field name	Description	Reset	Access
31:0	CURRDESAPTR	Host Receive Descriptor Address Pointer Cleared on Reset. Pointer updated by the DMA during operation.	0000_0000H	RO

9.5.2.1.14 Register 20 (Current Host Transmit Buffer Address Register)

The Current Host Transmit Buffer Address register points to the current Transmit Buffer Address being read by the DMA.

Field	Field name	Description	Reset	Access
31:0	CURTBUFAPT	Host Transmit Buffer Address Pointer Cleared on Reset. Pointer updated by the DMA during operation	0000_0000H	RO

9.5.2.1.15 Register 21 (Current Host Receive Buffer Address Register)

The Current Host Receive Buffer Address register points to the current Receive Buffer address being read by the DMA.

Field	Field name	Description	Reset	Access
31:0	CURRBUFAPTR	Host Receive Buffer Address Pointer Cleared on Reset. Pointer updated by the DMA during operation	0000_0000H	RO

9.5.2.1.16 Register 22 (HW Feature Register)

This register indicates the presence of the optional features or functions of the DWC_gmac. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.

Field	Field name	Description	Access
31	-	Reserved	RO
30:28	ACTPHYIF	Active or selected PHY interface When you have multiple PHY interfaces in your configuration, this field indicates the sampled value of phy_intf_sel_i during reset de-assertion. • 0000: GMII or MII • 0001: RGMII • 0010: SGMII • 0011: TBI • 0100: RMII • 0101: RTBI • 0110: SMII • 0111: RevMII • All Others: Reserved	RO
27	SAVLANINS	Source Address or VLAN Insertion	RO
26	FLEXIPPSEN	Flexible Pulse-Per-Second Output	RO
25	INTTSEN	Timestamping with Internal System Time	RO
24	ENHDESSEL	Alternate (Enhanced Descriptor)	RO
23:22	TXCHCNT	Number of additional Tx Channels	RO
21:20	RXCHCNT	Number of additional Rx Channels	RO
19	RXFIFOSIZE	Rx FIFO > 2,048 Bytes	RO
18	RXTYP2COE	IP Checksum Offload (Type 2) in Rx	RO
17	RXTYP1COE	IP Checksum Offload (Type 1) in Rx	RO
16	TXCOESEL	Checksum Offload in Tx	RO
15	AVSEL	AV Feature	RO
14	EEESEL	Energy Efficient Ethernet	RO
13	TSVER2SEL	IEEE 1588-2008 Advanced Timestamp	RO

12	TSVER1SEL	Only IEEE 1588-2002 Timestamp	RO
11	MMCSEL	RMON Module	RO
10	MGKSEL	PMT Magic Packet	RO
9	RWKSEL	PMT Remote Wakeup	RO
8	SMASEL	SMA (MDIO) Interface	RO
7	L3L4FLTREN	Layer 3 and Layer 4 Feature	RO
6	PCSSEL	PCS registers (TBI, SGMII, or RTBI PHY interface)	RO
5	ADDMACADRSEL	Multiple MAC Address Registers	RO
4	HASHSEL	HASH Filter	RO
3	EXTHASHEN	Expanded DA Hash Filter	RO
2	HDSEL	Half-Duplex Support	RO
1	GMIISEL	1000 Mbps Support	RO
0	MIISEL	10 or 100 Mbps Support	RO

9.5.2.1.17 Register 76 (Channel 1 Slot Function Control and Register)

This register controls the slot comparison feature that the Channel 1 transmit DMA uses to fetch the buffer data from system memory.

Field	Field name	Description	Reset	Access
31:20	-	Reserved	00H	RO
19:16	RSN	Reference Slot Number This field gives the current value of the reference slot number in DMA used for comparison checking	0H	RO
15:2	-	Reserved	000H	RO
1	ASC	Advance Slot Check When set, this bit enables the DMA to fetch the data from the buffer when the slot number (SLOTNUM) programmed in the transmit descriptor is: • equal to the reference slot number given in Bits [19:16] -or- • ahead of the reference slot number by up to two slots This bit is applicable only when Bit 0 (ESC) is set.	0	R_W
0	ESC	Enable Slot Comparison When set, this bit enables the checking of the slot numbers, programmed in the transmit descriptor, with the current reference given in Bits [19:16]. The DMA fetches the data from the corresponding buffer only when the slot number is equal to the reference slot number or is ahead of the reference slot number by one slot. When reset, this bit disables the checking of the slot numbers. The DMA fetches the data immediately after the descriptor is processed.	0	R_W

9.5.2.1.18 Register 88 (Channel 1 CBS Control Register)

This register controls the credit-based shaper algorithm in the Traffic Manager for scheduling the frames for transmission. This register is present only when you select the Transmit Channel 1 in the AV mode.

Field	Field name	Description	Reset	Access
31:18	-	Reserved	00H	RO
17	ABPSSIE	Average Bits Per Slot Interrupt Enable When this bit is set, the MAC asserts an interrupt (sbd_intr_o or mci_intr_o) when the average bits per slot status is updated (Bit	0	R_W

		17 (ABSU) in Register 89) for Channel 1. When this bit is cleared, interrupt is not asserted for such an event.		
16:7	-	Reserved	00H	RO
6:4	SLC	SLC: Slot Count The software can program the number of slots (of duration 125 micro-sec) over which the average transmitted bits per slot (provided in the CBS Status register) need to be computed for Channel 1 when the credit-based shaper algorithm is enabled. The encoding is as follows: • 3'b000: 1 Slot • 3'b001: 2 Slots • 3'b010: 4 Slots • 3'b011: 8 Slots • 3'b100: 16 Slots • 3'b101-3'b111: Reserved	00	R_W
3:2	-	Reserved	00	RO
1	CC	Credit Control When reset, the accumulated credit parameter in the credit-based shaper algorithm logic is set to zero when there is positive credit and no frame to transmit in Channel 1. When there is no frame waiting in Channel 1 and other channel is transmitting, no credit is accumulated. When set, the accumulated credit parameter in the credit-based shaper algorithm logic is not reset to zero when there is positive credit and no frame to transmit in Channel 1. The credit accumulates even when there is no frame waiting in Channel 1 and another channel is transmitting.	0	R_W
0	CBSD	Credit-Based Shaper Disable When set, the MAC disables the credit-based shaper algorithm for Channel 1 traffic and makes the traffic management algorithm to strict priority for Channel 1 over Channel 0. When reset, the credit-based shaper algorithm schedules the traffic in Channel 1 for transmission.	0	R_W

9.5.2.1.19 Register 89 (Channel 1 CBS Status Register)

This register provides the average traffic transmitted in Channel 1. This register is present only when you select the Transmit Channel 1 in the AV mode.

Field	Field name	Description	Reset	Access
31:18	-	Reserved	000H	RO
17	ABSU	ABS Updated When set, this bit indicates that the MAC has updated the ABS value. This bit is cleared when the application reads the ABS value.	0	R_SS_SC
16:0	ABS	Average Bits per Slot This field contains the average transmitted bits per slot. This field is computed over programmed number of slots (SLC bits in the CBS Control Register) for Channel 1 traffic. The maximum value is 0x30D4 for 100 Mbps and 0x1E848 for 1000 Mbps.	000H	RO

9.5.2.1.20 Register 90 (Channel 1 idleSlopeCredit Register)

This register provides the bandwidth allocated for the AV traffic on Channel 1. This register is present only when you select the Transmit Channel 1 in the AV mode.

Field	Field name	Description	Reset	Access
31:14	-	Reserved	000H	RO
13:0	ISC	idleSlopeCredit This field contains the idleSlopeCredit value required for the credit-based shaper algorithm for Channel 1. This is the rate of change of credit in bits per cycle (40ns and 8ns for 100 Mbps and 1000 Mbps respectively) when the credit is increasing. The software should program this field with computed credit in bits per cycle scaled by 1024. The maximum value is portTransmitRate, that is, 0x2000 in 1000 Mbps mode and 0x1000 in 100 Mbps mode.	000H	R_W

9.5.2.1.21 Register 91 (Channel 1 sendSlopeCredit Register)

This register provides the bandwidth that is available for the AV traffic on other channels. This register is present only when you select the Transmit Channel 1 in the AV mode.

Field	Field name	Description	Reset	Access
31:14	-	Reserved	000H	RO
13:0	SSC	sendSlopeCredit This field contains the sendSlopeCredit value required for credit-based shaper algorithm for Channel 1. This is the rate of change of credit in bits per cycle (40ns and 8ns for 100 Mbps and 1000 Mbps respectively) when the credit is decreasing. The software should program this field with computed credit in bits per cycle scaled by 1024. The maximum value is portTransmitRate, that is, 0x2000 in 1000 Mbps mode and 0x1000 in 100 Mbps mode. This field should be programmed with absolute sendSlopeCredit value. The credit-based shaper logic subtracts it from the accumulated credit when Channel 1 is selected for transmission.	000H	R_W

9.5.2.1.22 Register 92 (Channel 1 hiCredit Control Register)

This register provides the maximum value that can be accumulated for Channel 1 in the credit parameter of the credit-based shaper algorithm. This register is present only when you select the Transmit Channel 1 in the AV mode.

Field	Field name	Description	Reset	Access
31:29	-	Reserved	0H	RO
28:0	HC	hiCredit This field contains the hiCredit value required for the credit-based shaper algorithm for Channel 1. This is the maximum value that can be accumulated in the credit parameter. This is specified in bits scaled by 1,024. The maximum value is maxInterferenceSize, that is, best-effort maximum frame size which is 16,384 bytes or 131,072 bits. The value to be specified is 131,072 * 1,024 = 134,217,728 or 0x0800_0000.	0000000H	R_W

9.5.2.1.23 Register 93 (Channel 1 loCredit Control Register)

Field	Field name	Description	Reset	Access

31:29	-	Reserved	0x7	RO
28:0	LC	loCredit This field contains the loCredit value required for the credit-based shaper algorithm for Channel 1. This is the minimum value that can be accumulated in the credit parameter. This is specified in bits scaled by 1,024. The maximum value is maxInterferenceSize, that is, best-effort maximum frame size which is 16,384 bytes or 131,072 bits. The value to be specified is 131,072 * 1024 = 134,217,728 or 0x0800_0000. The programmed value is 2's complement (negative number), that is, 0xF800_0000.	0x1FFF_FFFF	R_W

9.5.2.2 GMAC Register Descriptions

9.5.2.2.1 Register 0 (MAC Configuration Register)

The MAC Configuration register establishes receive and transmit operating modes.

Field	Field name	Description	Reset	Access
31	-	Reserved	00	RO
30:28	SARC	Source Address Insertion or Replacement Control This field controls the source address insertion or replacement for all transmitted frames. Bit 30 specifies which MAC Address register (0 or 1) is used for source address insertion or replacement based on the values of Bits [29:28]: • **2'b0*x*:** The input signals mti_sa_ctrl_i and ati_sa_ctrl_i control the SA field generation. • **2'b10:** - If Bit 30 is set to 0, the MAC inserts the content of the MAC Address 0 registers (registers 16 and 17) in the SA field of all transmitted frames. - If Bit 30 is set to 1 and the Enable MAC Address Register 1 option is selected during core configuration, the MAC inserts the content of the MAC Address 1 registers (registers 18 and 19) in the SA field of all transmitted frames. • **2'b11**: - If Bit 30 is set to 0, the MAC replaces the content of the MAC Address 0 registers (registers 16 and 17) in the SA field of all transmitted frames. - If Bit 30 is set to 1 and the Enable MAC Address Register 1 option is selected during core configuration, the MAC replaces the content of the MAC Address 1 registers (registers 18 and 19) in the SA field of all transmitted frames. **Note**: • Changes to this field take effect only on the start of a frame. If you write this register field when a frame is being transmitted, only the subsequent frame can use the updated value, that is, the current frame does not use the updated value. • These bits are reserved and RO when the Enable SA, VLAN, and CRC Insertion on TX feature is not selected during core configuration.	00	R_W
27	2KPE	IEEE 802.3as Support for 2K Packets When set, the MAC considers all frames, with up to 2,000	0	R_W

		bytes length, as normal packets. When Bit 20 (Jumbo Enable) is not set, the MAC considers all received frames of size more than 2K bytes as Giant frames. When this bit is reset and Bit 20 (Jumbo Enable) is not set, the MAC considers all received frames of size more than 1,518 bytes (1,522 bytes for tagged) as Giant frames. When Bit 20 (Jumbo Enable) is set, setting this bit has no effect on Giant Frame status.		
26	SFTERR	SMII Force Transmit Error When set, this bit indicates to the PHY to force a transmit error in the SMII frame being transmitted. This bit is reserved if the SMII PHY port is not selected during core configuration.	0	R_W
25	CST	CRC stripping for Type frames When set, the last 4 bytes (FCS) of all frames of Ether type (type field greater than 0x0600) are stripped and dropped before forwarding the frame to the application. This function is not valid when the IP Checksum Engine (Type 1) is enabled in the MAC receiver.	0	R_W
24	TC	Transmit Configuration in RGMII, SGMII, or SMII When set, this bit enables the transmission of duplex mode, link speed, and link up or down information to the PHY in the RGMII, SMII, or SGMII port. When this bit is reset, no such information is driven to the PHY. This bit is reserved (and RO) if the RGMII, SMII, or SGMII PHY port is not selected during core configuration.	0	R_W
23	WD	Watchdog Disable When this bit is set, the MAC disables the watchdog timer on the receiver. The MAC can receive frames of up to 16,384 bytes. When this bit is reset, the MAC does not allow more than 2,048 bytes (10,240 if JE is set high) of the frame being received. The MAC cuts off any bytes received after 2,048 bytes.	0	R_W
22	JD	Jabber Disable When this bit is set, the MAC disables the jabber timer on the transmitter. The MAC can transfer frames of up to 16,384 bytes. When this bit is reset, the MAC cuts off the transmitter if the application sends out more than 2,048 bytes of data (10,240 if JE is set high) during transmission.	0	R_W
21	BE	Frame Burst Enable When this bit is set, the MAC allows frame bursting during transmission in the GMII half-duplex mode. This bit is reserved (and RO) in the 10/100 Mbps only or full-duplex-only configurations.	0	R_W
20	JE	Jumbo Frame Enable When this bit is set, the MAC allows Jumbo frames of 9,018 bytes (9,022 bytes for VLAN tagged frames) without reporting a giant frame error in the receive frame status.	0	R_W
19:17	IFG	Inter-Frame Gap These bits control the minimum IFG between frames during transmission. • 000: 96 bit times • 001: 88 bit times • 010: 80 bit times • ... • 111: 40 bit times In the half-duplex mode, the minimum IFG can be configured only for 64 bit times (IFG = 100). Lower values are not	000	R_W

		considered. In the 1000-Mbps mode, the minimum IFG supported is 64 bit times (and above) in the GMAC-CORE configuration and 80 bit times (and above) in other configurations.		
16	DCRS	Disable Carrier Sense During Transmission When set high, this bit makes the MAC transmitter ignore the (G)MII CRS signal during frame transmission in the half-duplex mode. This request results in no errors generated because of Loss of Carrier or No Carrier during such transmission. When this bit is low, the MAC transmitter generates such errors because of Carrier Sense and can even abort the transmissions. This bit is reserved (and RO) in the full-duplex-only configurations.	0	R_W
15	PS	Port Select This bit selects between GMII and MII: • 0: GMII (1000 Mbps) • 1: MII (10/100 Mbps) In the 10 or 100 Mbps-only (always 1) or 1000 Mbps-only (always 0) configurations, this bit is read-only with the appropriate value. In the default 10/100/1000 Mbps configurations, this bit is R_W. The mac_portselect_o signal reflects the value of this bit.	0	R_W
14	FES	Speed This bit selects the speed in the MII, RMII, SMII, RGMII, SGMII, or RevMII interface: • 0: 10 Mbps • 1: 100 Mbps This bit is reserved (RO) by default and is enabled only when the RMII, SMII, RGMII, SGMII, or RevMII interface is enabled during core configuration. This bit generates link speed encoding when TC (Bit 24) is set in the RGMII, SMII, or SGMII mode. This bit is always driven for RevMII. If the MAC is configured with an RMII, SMII, SGMII, or RGMII PHY interface, this bit can optionally be driven as an output signal (mac_speed_o[0]) to reflect the value of this bit in the mac_speed_o signal. In addition, this bit is reserved when RMII is enabled without enabling the Output Port for speed selection option during core configuration.	0	R_W
13	DO	Disable Receive Own When this bit is set, the MAC disables the reception of frames when the gmii_txen_o is asserted in the half-duplex mode. When this bit is reset, the MAC receives all packets that are given by the PHY while transmitting. This bit is not applicable if the MAC is operating in the full-duplex mode. This bit is reserved (RO with default value) if the MAC is configured for the full-duplex-only operation.	0	R_W
12	LM	Loopback Mode When this bit is set, the MAC operates in the loopback mode at GMII or MII. The (G)MII Receive clock input (clk_rx_i) is required for the loopback to work properly, because the Transmit clock is not looped-back internally.	0	R_W
11	DM	Duplex Mode When this bit is set, the MAC operates in the full-duplex mode where it can transmit and receive simultaneously. This bit is RO with default value of 1'b1 in the full-duplex-only configuration.	0	R_W
10	IPC	Checksum Offload	0	R_W

		When this bit is set, the MAC calculates the 16-bit one's complement of the one's complement sum of all received Ethernet frame payloads. It also checks whether the IPv4 Header checksum (assumed to be bytes 25–26 or 29–30 (VLAN-tagged) of the received Ethernet frame) is correct for the received frame and gives the status in the receive status word. The MAC also appends the 16-bit checksum calculated for the IP header datagram payload (bytes after the IPv4 header) and appends it to the Ethernet frame transferred to the application (when Type 2 COE is deselected). When this bit is reset, this function is disabled. When Type 2 COE is selected, this bit, when set, enables the IPv4 header checksum checking and IPv4 or IPv6 TCP, UDP, or ICMP payload checksum checking. When this bit is reset, the COE function in the receiver is disabled and the corresponding PCE and IP HCE status bits (see Table 3-8 on page 127) are always cleared. If the IP Checksum Offload feature is not enabled during core configuration, this bit is reserved (RO with default value).		
9	DR	Disable Retry When this bit is set, the MAC attempts only one transmission. When a collision occurs on the GMII or MII interface, the MAC ignores the current frame transmission and reports a Frame Abort with excessive collision error in the transmit frame status. When this bit is reset, the MAC attempts retries based on the settings of the BL field (Bits [6:5]). This bit is applicable only in the half-duplex mode and is reserved (RO with default value) in the full-duplex-only configuration.	0	R_W
8	LUD	Link Up or Down This bit indicates whether the link is up or down during the transmission of configuration in the RGMII, SGMII, or SMII interface: • 0: Link Down • 1: Link Up This bit is reserved (RO with default value) and is enabled when the RGMII, SGMII, or SMII interface is enabled during core configuration.	0	R_W
7	ACS	Automatic Pad or CRC Stripping When this bit is set, the MAC strips the Pad or FCS field on the incoming frames only if the value of the length field is less than 1,536 bytes. All received frames with length field greater than or equal to 1,536 bytes are passed to the application without stripping the Pad or FCS field. When this bit is reset, the MAC passes all incoming frames, without modifying them, to the Host.	0	R_W
6:5	BL	Back-Off Limit The Back-Off limit determines the random integer number (r) of slot time delays (4,096 bit times for 1000 Mbps and 512 bit times for 10/100 Mbps) for which the MAC waits before rescheduling a transmission attempt during retries after a collision. This bit is applicable only in the half-duplex mode and is reserved (RO) in the full-duplex-only configuration. • 00: k= min (*n*, 10) • 01: k = min (*n*, 8) • 10: k = min (*n*, 4) • 11: k = min (*n*, 1) where *n* = retransmission attempt. The random integer *r* takes the value in the range $0 \le r < 2k$	00	R_W

4	DC	Deferral Check When this bit is set, the deferral check function is enabled in the MAC. The MAC issues a Frame Abort status, along with the excessive deferral error bit set in the transmit frame status, when the transmit state machine is deferred for more than 24,288 bit times in the 10 or 100 Mbps mode. If the MAC is configured for 1000 Mbps operation, or if the Jumbo frame mode is enabled in the 10 or 100 Mbps mode, the threshold for deferral is 155,680 bits times. Deferral begins when the transmitter is ready to transmit, but is prevented because of an active carrier sense signal (CRS) on GMII or MII. Defer time is not cumulative. When the transmitter defers for 10,000 bit times, it transmits, collides, backs off, and then defers again after completion of back-off. The deferral timer resets to 0 and restarts. When this bit is reset, the deferral check function is disabled and the MAC defers until the CRS signal goes inactive. This bit is applicable only in half-duplex mode and is reserved (RO) in the full-duplex-only configuration.	0	R_W
3	TE	Transmitter Enable When this bit is set, the transmit state machine of the MAC is enabled for transmission on the GMII or MII. When this bit is reset, the MAC transmit state machine is disabled after the completion of the transmission of the current frame, and does not transmit any further frames.	0	R_W
2	RE	Receiver Enable When this bit is set, the receiver state machine of the MAC is enabled for receiving frames from the GMII or MII. When this bit is reset, the MAC receive state machine is disabled after the completion of the reception of the current frame, and does not receive any further frames from the GMII or MII.	0	R_W
1:0	PRELEN	Preamble Length for Transmit frames These bits control the number of preamble bytes that are added to the beginning of every Transmit frame. The preamble reduction occurs only when the MAC is operating in the full-duplex mode. • 2'b00: 7 bytes of preamble • 2'b01: 5 bytes of preamble • 2'b10: 3 bytes of preamble • 2'b11: Reserved	00	R_W

9.5.2.2.2 Register 1 (MAC Frame Filter)

The MAC Frame Filter register contains the filter controls for receiving frames. Some of the controls from this register go to the address check block of the MAC, which performs the first level of address filtering. The second level of filtering is performed on the incoming frame, based on other controls such as Pass Bad Frames and Pass Control Frames.

Field	Field name	Description	Reset	Access
31	RA-	Receive All When this bit is set, the MAC Receiver module passes all received frames, irrespective of whether they pass the address filter or not, to the Application. The result of the SA or DA filtering is updated (pass or fail) in the corresponding bits in the Receive Status Word. When this bit is reset, the Receiver module passes only those frames to the Application that pass the SA or DA address filter.	0	R_W
30:22	-	Reserved	00H	RO

21	DNTU	Drop non-TCP/UDP over IP Frames When set, this bit enables the MAC to drop the non-TCP or UDP over IP frames. The MAC forward only those frames that are processed by the Layer 4 filter. When reset, this bit enables the MAC to forward all non-TCP or UDP over IP frames. If the Layer 3 and Layer 4 Filtering feature is not selected during core configuration, this bit is reserved (RO with default value).	0	R_W
20	IPFE	Layer 3 and Layer 4 Filter Enable When set, this bit enables the MAC to drop frames that do not match the enabled Layer 3 and Layer 4 filters. If Layer 3 or Layer 4 filters are not enabled for matching, this bit does not have any effect. When reset, the MAC forwards all frames irrespective of the match status of the Layer 3 and Layer 4 fields. If the Layer 3 and Layer 4 Filtering feature is not selected during core configuration, this bit is reserved (RO with default value).	0	R_W
19:17	-	Reserved	000	RO
16	VTFE	VLAN Tag Filter Enable When set, this bit enables the MAC to drop VLAN tagged frames that do not match the VLAN Tag comparison. When reset, the MAC forwards all frames irrespective of the match status of the VLAN Tag.	0	R_W
15:11	-	Reserved	00	RO
10	HPF	Hash or Perfect Filter When this bit is set, it configures the address filter to pass a frame if it matches either the perfect filtering or the hash filtering as set by the HMC or HUC bits. When this bit is low and the HUC or HMC bit is set, the frame is passed only if it matches the Hash filter. This bit is reserved (and RO) if the Hash filter is not selected during core configuration.	0	R_W
9	SAF	Source Address Filter Enable When this bit is set, the MAC compares the SA field of the received frames with the values programmed in the enabled SA registers. If the comparison matches, then the SA Match bit of RxStatus Word is set high. When this bit is set high and the SA filter fails, the MAC drops the frame. When this bit is reset, the MAC forwards the received frame to the application and with the updated SA Match bit of the RxStatus depending on the SA address comparison.	0	R_W
8	SAIF	SA Inverse Filtering When this bit is set, the Address Check block operates in inverse filtering mode for the SA address comparison. The frames whose SA matches the SA registers are marked as failing the SA Address filter. When this bit is reset, frames whose SA does not match the SA registers are marked as failing the SA Address filter.	0	R_W
7:6	PCF	Pass Control Frames These bits control the forwarding of all control frames (including unicast and multicast PAUSE frames) •00: MAC filters all control frames from reaching the application. • 01: MAC forwards all control frames except PAUSE control frames to application even if they fail the Address filter. •10: MAC forwards all control frames to application even if they fail the Address Filter. •11: MAC forwards control frames that pass the Address Filter. The following conditions should be true for the PAUSE control frames processing: • **Condition 1:** The MAC is in the full-duplex mode and flow	00	R_W

		control is enabled by setting Bit 2 (RFE) of Register 6 (Flow Control Register) to 1. • **Condition 2:** The destination address (DA) of the received frame matches the special multicast address or the MAC Address 0 when Bit 3 (UP) of the Register 6 (Flow Control Register) is set. • **Condition 3**: The Type field of the received frame is 0x8808 and the OPCODE field is 0x0001. **Note:** This field should be set to 01 only when the Condition 1 is true, that is, the MAC is programmed to operate in the full-duplex mode and the RFE bit is enabled. Otherwise, the PAUSE frame filtering may be inconsistent. When Condition 1 is false, the PAUSE frames are considered as generic control frames. Therefore, to pass all control frames (including PAUSE control frames) when the full-duplex mode and flow control is not enabled, you should set the PCF field to 10 or 11 (as required by the application).		
5	DBF	Disable Broadcast Frames When this bit is set, the AFM module filters all incoming broadcast frames. In addition, it overrides all other filter settings. When this bit is reset, the AFM module passes all received broadcast frames.	0	R_W
4	PM	Pass All Multicast When set, this bit indicates that all received frames with a multicast destination address (first bit in the destination address field is '1') are passed. When reset, filtering of multicast frame depends on HMC bit.	0	R_W
3	DAIF	DA Inverse Filtering When this bit is set, the Address Check block operates in inverse filtering mode for the DA address comparison for both unicast and multicast frames. When reset, normal filtering of frames is performed.	0	R_W
2	HMC	Hash Multicast When set, MAC performs destination address filtering of received multicast frames according to the hash table. When reset, the MAC performs a perfect destination address filtering for multicast frames, that is, it compares the DA field with the values programmed in DA registers. If Hash Filter is not selected during core configuration, this bit is reserved (and RO).	0	R_W
1	HUC	Hash Unicast When set, MAC performs destination address filtering of unicast frames according to the hash table. When reset, the MAC performs a perfect destination address filtering for unicast frames, that is, it compares the DA field with the values programmed in DA registers. If Hash Filter is not selected during core configuration, this bit is reserved (and RO).	0	R_W
0	PR	Promiscuous Mode When this bit is set, the Address Filter module passes all incoming frames regardless of its destination or source address. The SA or DA Filter Fails status bits of the Receive Status Word are always cleared when PR is set.	0	R_W

9.5.2.2.3 Register 2 (Hash Table High Register)

The 64-bit Hash table is used for group address filtering. For hash filtering, the contents of the destination address in the incoming frame is passed through the CRC logic, and the upper 6 bits of the CRC register are used to index

the contents of the Hash table. The most significant bit determines the register to be used (Hash Table High or Hash Table Low), and the other 5 bits determine which bit within the register. A hash value of 5b'00000 selects Bit 0 of the selected register, and a value of 5b'11111 selects Bit 31 of the selected register.

The hash value of the destination address is calculated in the following way:

1. Calculate the 32-bit CRC for the DA (See IEEE 802.3, *Section 3.2.8* for the steps to calculate CRC32).
2. Perform bitwise reversal for the value obtained in Step 1.
3. Take the upper 6 bits from the value obtained in Step 2.

For example, if the DA of the incoming frame is received as 0x1F52419CB6AF (0x1F is the first byte received on GMII interface), then the internally calculated 6-bit Hash value is 0x2C and Bit 12 of Hash Table High register is checked for filtering. If the DA of the incoming frame is received as 0xA00A98000045, then the calculated 6-bit Hash value is 0x07 and Bit 7 of Hash Table Low register is checked for filtering.

If the corresponding bit value of the register is 1’b1, the frame is accepted. Otherwise, it is rejected. If the PM (Pass All Multicast) bit is set in Register 1, then all multicast frames are accepted regardless of the multicast hash values.

If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table High or Low registers are written. Consecutive writes to these register should be performed only after at least 4 clock cycles in the destination clock domain when double-synchronization is enabled.

The Hash Table High register contains the higher 32 bits of the Hash table.

Field	Field name	Description	Reset	Access
31:0	HTH	Hash Table High This field contains the upper 32 bits of the Hash table.	0000_0000H	R_W

9.5.2.2.4 Register 3 (Hash Table Low Register)

The Hash Table Low register contains the lower 32 bits of the Hash table. Both Register 2 and Register 3 are reserved if the Hash Filter Function is disabled or the 128-bit or 256-bit Hash Table is selected during core configuration.

Field	Field name	Description	Reset	Access
31:0	HTL	Hash Table Low This field contains the lower 32 bits of the Hash table.	0000_0000H	R_W

9.5.2.2.5 Register 4 (GMII Address Register)

The GMII Address register controls the management cycles to the external PHY through the management interface.

Field	Field name	Description	Reset	Access
31:16	-	Reserved	00H	RO
15:11	PA	Physical Layer Address This field indicates which of the 32 possible PHY devices are being accessed. For RevMII, this field gives the PHY Address of the RevMII module.	00H	R_W
10:6	GR	GMII Register These bits select the desired GMII register in the selected PHY device. For RevMII, these bits select the desired CSR register in the RevMII Registers set.	00H	R_W

5:2	CR	CSR Clock Range The CSR Clock Range selection determines the frequency of the MDC clock according to the clk_csr_i frequency used in your design. The suggested range of clk_csr_i frequency applicable for each value (when Bit[5] = 0) ensures that the MDC clock is approximately between the frequency range 1.0 MHz - 2.5 MHz. • 0000: The frequency of the clk_csr_i clock is 60-100 MHz and the MDC clock is clk_csr_i/42. • 0001: The frequency of the clk_csr_i clock is 100-150 MHz and the MDC clock is clk_csr_i/62. • 0010: The frequency of the clk_csr_i clock is 20-35 MHz and the MDC clock is clk_csr_i/16. • 0011: The frequency of the clk_csr_i clock is 35-60 MHz and the MDC clock is clk_csr_i/26. • 0100: The frequency of the clk_csr_i clock is 150-250 MHz and the MDC clock is clk_csr_i/102. • 0100: The frequency of the clk_csr_i clock is 250-300 MHz and the MDC clock is clk_csr_i/124. • 0110, 0111: Reserved When Bit 5 is set, you can achieve MDC clock of frequency higher than the IEEE 802.3 specified frequency limit of 2.5 MHz and program a clock divider of lower value. For example, when clk_csr_i is of 100 MHz frequency and you program these bits as 1010, then the resultant MDC clock is of 12.5 MHz which is outside the limit of IEEE 802.3 specified range. Program the following values only if the interfacing chips support faster MDC clocks. • 1000: clk_csr_i/4 • 1001: clk_csr_i/6 • 1010: clk_csr_i/8 • 1011: clk_csr_i/10 • 1100: clk_csr_i/12 • 1101: clk_csr_i/14 • 1110: clk_csr_i/16 • 1111: clk_csr_i/18 These bits are not used for accessing RevMII. These bits are read-only if the RevMII interface is selected as single PHY interface.	0000	R_W
1	GW	GMII Write When set, this bit indicates to the PHY or RevMII that this is a Write operation using the GMII Data register. If this bit is not set, it indicates that this is a Read operation, that is, placing the data in the GMII Data register.	0	R_W
0	GB	GMII Busy This bit should read logic 0 before writing to Register 4 and Register 5. During a PHY or RevMII register access, the software sets this bit to 1’b1 to indicate that a Read or Write access is in progress. The Register 5 is invalid until this bit is cleared by the MAC. Therefore, Register 5 (GMII Data) should be kept valid until the MAC clears this bit during a PHY Write operation. Similarly for a read operation, the contents of Register 5 are not valid until this bit is cleared. The subsequent read or write operation should happen only after the previous operation is complete. Because there is no acknowledgment from the PHY to MAC after a read or write operation is completed, there is no change in the functionality of this bit even when the PHY is not present.	0	R_WS_SC

9.5.2.2.6 Register 5 (GMII Data Register)

The GMII Data register stores Write data to be written to the PHY register located at the address specified in Register 4 (GMII Address Register). This register also stores the Read data from the PHY register located at the address specified by Register 4.

Field	Field name	Description	Reset	Access
31:16	-	Reserved	0000H	RO
15:0	GD	GD: GMII Data This field contains the 16-bit data value read from the PHY or RevMII after a Management Read operation or the 16-bit data value to be written to the PHY or RevMII before a Management Write operation.	0000H	R_W

9.5.2.2.7 Register 6 (Flow Control Register)

The Flow Control register controls the generation and reception of the Control (Pause Command) frames by the MAC's Flow control module. A Write to a register with the Busy bit set to '1' triggers the Flow Control block to generate a Pause Control frame. The fields of the control frame are selected as specified in the 802.3x specification, and the Pause Time value from this register is used in the Pause Time field of the control frame. The Busy bit remains set until the control frame is transferred onto the cable. The Host must make sure that the Busy bit is cleared before writing to the register.

Field	Field name	Description	Reset	Access
31:16	PT	Pause Time This field holds the value to be used in the Pause Time field in the transmit control frame. If the Pause Time bits is configured to be double-synchronized to the (G)MII clock domain, then consecutive writes to this register should be performed only after at least four clock cycles in the destination clock domain.	0000H	R_W
15:8	-	Reserved	00H	RO
7	DZPQ	Disable Zero-Quanta Pause When this bit is set, it disables the automatic generation of the Zero-Quanta Pause Control frames on the de-assertion of the flow-control signal from the FIFO layer (MTL or external sideband flow control signal sbd_flowctrl_i/mti_flowctrl_i). When this bit is reset, normal operation with automatic Zero-Quanta Pause Control frame generation is enabled.	0	R_W
6	-	Reserved	0	RO
5:4	PLT	Pause Low Threshold This field configures the threshold of the PAUSE timer at which the input flow control signal mti_flowctrl_i (or sbd_flowctrl_i) is checked for automatic retransmission of PAUSE Frame. The threshold values should be always less than the Pause Time configured in Bits[31:16]. For example, if PT = 100H (256 slot-times), and PLT = 01, then a second PAUSE frame is automatically transmitted if the mti_flowctrl_i signal is asserted at 228 (256 – 28) slot times after the first PAUSE frame is transmitted. The following list provides the threshold values for different values: • 00: The threshold is Pause time minus 4 slot times (PT – 4 slot times).	00	R_W

		• 01: The threshold is Pause time minus 28 slot times (PT – 28 slot times). • 10: The threshold is Pause time minus 144 slot times (PT – 144 slot times). • 11: The threshold is Pause time minus 256 slot times (PT – 256 slot times). The slot time is defined as the time taken to transmit 512 bits (64 bytes) on the GMII or MII interface.		
3	UP	Unicast Pause Frame Detect When this bit is set, then in addition to the detecting Pause frames with the unique multicast address, the MAC detects the Pause frames with the station's unicast address specified in the MAC Address0 High Register and MAC Address0 Low Register. When this bit is reset, the MAC detects only a Pause frame with the unique multicast address specified in the 802.3x standard.	0	R_W
2	RFE	Receive Flow Control Enable When this bit is set, the MAC decodes the received Pause frame and disables its transmitter for a specified (Pause) time. When this bit is reset, the decode function of the Pause frame is disabled.	0	R_W
1	TFE	Transmit Flow Control Enable In the full-duplex mode, when this bit is set, the MAC enables the flow control operation to transmit Pause frames. When this bit is reset, the flow control operation in the MAC is disabled, and the MAC does not transmit any Pause frames. In the half-duplex mode, when this bit is set, the MAC enables the backpressure operation. When this bit is reset, the back-pressure feature is disabled.	0	R_W
0	FCB_ BPA	Flow Control Busy or Backpressure Activate This bit initiates a Pause Control frame in the full-duplex mode and activates the backpressure function in the half-duplex mode if the TFE bit is set. In the full-duplex mode, this bit should be read as 1'b0 before writing to the Flow Control register. To initiate a Pause control frame, the Application must set this bit to 1'b1. During a transfer of the Control Frame, this bit continues to be set to signify that a frame transmission is in progress. After the completion of Pause control frame transmission, the MAC resets this bit to 1'b0. The Flow Control register should not be written to until this bit is cleared. In the half-duplex mode, when this bit is set (and TFE is set), then backpressure is asserted by the MAC. During backpressure, when the MAC receives a new frame, the transmitter starts sending a JAM pattern resulting in a collision. This control register bit is logically ORed with the mti_flowctrl_i input signal for the backpressure function. When the MAC is configured for the full-duplex mode, the BPA is automatically disabled.	0	R_WS_SC for FCB R_W for BPA

9.5.2.2.8 Register 7 (VLAN Tag Register)

The VLAN Tag register contains the IEEE 802.1Q VLAN Tag to identify the VLAN frames. The MAC compares the 13th and 14th bytes of the receiving frame (Length/Type) with 16'h8100, and the following two bytes are compared with the VLAN tag. If a match occurs, the MAC sets the received VLAN bit in the receive frame status. The legal length of the frame is increased from 1,518 bytes to 1,522 bytes. If the VLAN Tag register is configured

to be double-synchronized to the (G)MII clock domain, then consecutive writes to these register should be performed only after at least four clock cycles in the destination clock domain.

Field	Field name	Description	Reset	Access
31:20	-	Reserved	000H	RO
19	VTHM	VLAN Tag Hash Table Match Enable When set, the most significant four bits of the VLAN tag's CRC are used to index the content of Register 354 (VLAN Hash Table Register). A value of 1 in the VLAN Hash Table register, corresponding to the index, indicates that the frame matched the VLAN hash table. When Bit 16 (ETV) is set, the CRC of the 12-bit VLAN Identifier (VID) is used for comparison whereas when ETV is reset, the CRC of the 16-bit VLAN tag is used for comparison. When reset, the VLAN Hash Match operation is not performed. If the VLAN Hash feature is not enabled during core configuration, this bit is reserved (RO with default value).	0	R_W
18	ESVL	Enable S-VLAN When this bit is set, the MAC transmitter and receiver also consider the S-VLAN (Type = 0x88A8) frames as valid VLAN tagged frames.	0	R_W
17	VTIM	VLAN Tag Inverse Match Enable When set, this bit enables the VLAN Tag inverse matching. The frames that do not have matching VLAN Tag are marked as matched. When reset, this bit enables the VLAN Tag perfect matching. The frames with matched VLAN Tag are marked as matched.	0	R_W
16	EVT	Enable 12-Bit VLAN Tag Comparison When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged frame. Similarly, when enabled, only 12 bits of the VLAN tag in the received frame are used for hash-based VLAN filtering. When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN frame are used for comparison and VLAN hash filtering.	0	R_W
15:0	VL	VLAN Tag Identifier for Receive Frames This field contains the 802.1Q VLAN tag to identify the VLAN frames and is compared to the 15th and 16th bytes of the frames being received for VLAN frames. The following list describes the bits of this field: • Bits [15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN tag's VLAN Identifier (VID) field When the ETV bit is set, only the VID (Bits[11:0]) is used for comparison. If VL (VL[11:0] if ETV is set) is all zeros, the MAC does not check the fifteenth and 16th bytes for VLAN tag comparison, and declares all frames with a Type field value of 0x8100 or 0x88a8 as VLAN frames.	0000H	R_W

9.5.2.2.9 Register 8 (Vision Register)

The Version registers identifies the version of the DWC_gmac. This register contains two bytes: one that Synopsys uses to identify the core release number, and the other that you set during core configuration.

Field	Field name	Description	Reset	Access
31:16	-	Reserved	0000H	RO
15:8	USERVER	User-defined Version (configured with coreConsultant)	xxH	RO
7:0	SNPSVER	Synopsys-defined Version (3.7)	37H	RI

9.5.2.2.10 Register 9 (Debug Register)

The Debug register gives the status of all main modules of the transmit and receive data-paths and the FIFOs. An all-zero status indicates that the MAC is in idle state (and FIFOs are empty) and no activity is going on in the data-paths.

Field	Field name	Description	Reset	Access
31:26	-	Reserved	00H	RO
25	TXSTSFSTS	MTL TxStatus FIFO Full Status When high, this bit indicates that the MTL TxStatus FIFO is full. Therefore, the MTL cannot accept any more frames for transmission. This bit is reserved in the GMAC-AHB and GMAC-DMA configurations.	0	RO
24	TXFSTS	MTL Tx FIFO Not Empty Status When high, this bit indicates that the MTL Tx FIFO is not empty and some data is left for transmission.	0	RO
23	-	Reserved	0	RO
22	TWCSTS	MTL Tx FIFO Write Controller Status When high, this bit indicates that the MTL Tx FIFO Write Controller is active and is transferring data to the Tx FIFO.	0	RO
21:20	TRCSTS	MTL Tx FIFO Read Controller Status This field indicates the state of the Tx FIFO Read Controller: •00: IDLE state •01: READ state (transferring data to the MAC transmitter) •10: Waiting for TxStatus from the MAC transmitter •11: Writing the received TxStatus or flushing the Tx FIFO	00	RO
19	TXPAUSED	MAC Transmitter in PAUSE When high, this bit indicates that the MAC transmitter is in the PAUSE condition (in the full-duplex-only mode) and hence does not schedule any frame for transmission.	0	RO
18:17	TFCSTS	MAC Transmit Frame Controller Status This field indicates the state of the MAC Transmit Frame Controller module: • 00: IDLE state • 01: Waiting for status of previous frame or IFG or backoff period to be over • 10: Generating and transmitting a PAUSE control frame (in the full-duplex mode) • 11: Transferring input frame for transmission	00	RO
16	TPESTS	MAC GMII or MII Transmit Protocol Engine Status When high, this bit indicates that the MAC GMII or MII transmit protocol engine is actively transmitting data and is not in the IDLE state.	0	RO
15:10	-	Reserved	0H	RO
9:8	RXFSTS	MTL Rx FIFO Fill-level Status This field gives the status of the fill-level of the Rx FIFO: • 00: Rx FIFO Empty • 01: Rx FIFO fill-level below flow-control deactivate threshold	00	RO

		• 10: Rx FIFO fill-level above flow-control activate threshold • 11: Rx FIFO Full		
7	-	Reserved	0	RO
6:5	RRCSTS	MTL Rx FIFO Read Controller State This field gives the state of the Rx FIFO read Controller: • 00: IDLE state • 01: Reading frame data • 10: Reading frame status (or timestamp) • 11: Flushing the frame data and status	00	RO
4	RWCSTS	MTL Rx FIFO Write Controller Active Status When high, this bit indicates that the MTL Rx FIFO Write Controller is active and is transferring a received frame to the FIFO.	0	RO
3	-	Reserved	0	RO
2:1	RFCFCSTS	MAC Receive Frame Controller FIFO Status When high, this field indicates the active state of the small FIFO Read and Write controllers of the MAC Receive Frame Controller Module.	00	RO
0	RPESTS	MAC GMII or MII Receive Protocol Engine Status When high, this bit indicates that the MAC GMII or MII receive protocol engine is actively receiving data and not in IDLE state.	0	RO

9.5.2.2.11 Register 12 (LPI Control and Status Register)

The LPI Control and Status Register controls the LPI functions and provides the LPI interrupt status. The status bits are cleared when this register is read. This register is present only when you select the Energy Efficient Ethernet feature during core configuration.

Field	Field name	Description	Reset	Access
31:20	-	Reserved	00000H	RO
19	LPITXA	LPI TX Automate This bit controls the behavior of the MAC when it is entering or coming out of the LPI mode on the transmit side. This bit is not functional in the GMACCORE configuration in which the Tx clock gating is done during the LPI mode. If the LPITXA and LPIEN bits are set to 1, the MAC enters the LPI mode only after all outstanding frames (in the core) and pending frames (in the application interface) have been transmitted. The MAC comes out of the LPI mode when the application sends any frame for transmission or the application issues a TX FIFO Flush command. In addition, the MAC automatically clears the LPIEN bit when it exits the LPI state. If TX FIFO Flush is set, in Bit 20 of Register 6 (Operation Mode Register), when the MAC is in the LPI mode, the MAC exits the LPI mode. When this bit is 0, the LPIEN bit directly controls behavior of the MAC when it is entering or coming out of the LPI mode.	0	R_W
18	PLSEN	PHY Link Status Enable This bit enables the link status received on the RGMII, SGMII, or SMII receive paths to be used for activating the LPI LS TIMER. When set, the MAC uses the link-status bits of Register 54 (SGMII/RGMII/SMII Status Register) and Bit 17 (PLS) for the LPI LS Timer trigger. When cleared, the MAC ignores the link-status bits of Register 54 and takes only the PLS bit.	0	R_W

		This bit is RO and reserved if you have not selected the RGMII, SGMII, or SMII PHY interface.		
17	PLS	PHY Link Status This bit indicates the link status of the PHY. The MAC Transmitter asserts the LPI pattern only when the link status is up (okay) at least for the time indicated by the LPI LS TIMER. When set, the link is considered to be okay (up) and when reset, the link is considered to be down.	0	R_W
16	LPIEN	LPI Enable When set, this bit instructs the MAC Transmitter to enter the LPI state. When reset, this bit instructs the MAC to exit the LPI state and resume normal transmission. This bit is cleared when the LPITXA bit is set and the MAC exits the LPI state because of the arrival of a new packet for transmission.	0	R_W_SC
15:10	-	Reserved	00H	RO
9	RLPIST	Receive LPI State When set, this bit indicates that the MAC is receiving the LPI pattern on the GMII or MII interface.	0	RO
8	TLPIST	Transmit LPI State When set, this bit indicates that the MAC is transmitting the LPI pattern on the GMII or MII interface.	0	RO
7:4	-	Reserved	0H	RO
3	RLPIEX	Receive LPI Exit When set, this bit indicates that the MAC Receiver has stopped receiving the LPI pattern on the GMII or MII interface, exited the LPI state, and resumed the normal reception. This bit is cleared by a read into this register. **Note:** This bit may not get set if the MAC stops receiving the LPI pattern for a very short duration, such as, less than 3 clock cycles of clk_csr_i.	0	R_SS_RC
2	RLPIEN	Receive LPI Entry When set, this bit indicates that the MAC Receiver has received an LPI pattern and entered the LPI state. This bit is cleared by a read into this register. **Note:** This bit may not get set if the MAC stops receiving the LPI pattern for a very short duration, such as, less than 3 clock cycles of clk_csr_i.	0	R_SS_RC
1	TLPIEX	Transmit LPI Exit When set, this bit indicates that the MAC transmitter has exited the LPI state after the user has cleared the LPIEN bit and the LPI TW Timer has expired. This bit is cleared by a read into this register.	0	R_SS_RC
0	TLPIEN	Transmit LPI Entry When set, this bit indicates that the MAC Transmitter has entered the LPI state because of the setting of the LPIEN bit. This bit is cleared by a read into this register.	0	R_SS_RC

9.5.2.2.12 Register 13 (LPI Timer Control Register)

The LPI Timers Control register controls the timeout values in the LPI states. It specifies the time for which the MAC transmits the LPI pattern and also the time for which the MAC waits before resuming the normal transmission. This register is present only when you select the Energy Efficient Ethernet feature during core configuration.

Field	Field name	Description	Reset	Access
31:26	-	Reserved	00H	RO

25:16	LST	LPI LS TIMER This field specifies the minimum time (in milliseconds) for which the link status from the PHY should be up (OKAY) before the LPI pattern can be transmitted to the PHY. The MAC does not transmit the LPI pattern even when the LPIEN bit is set unless the LPI LS Timer reaches the programmed terminal count. The default value of the LPI LS Timer is 1000 (1 sec) as defined in the IEEE standard.	0x3E8	R_W
15:0	TWT	LPI TW TIMER This field specifies the minimum time (in microseconds) for which the MAC waits after it stops transmitting the LPI pattern to the PHY and before it resumes the normal transmission. The TLPIEX status bit is set after the expiry of this timer.	0	R_W

9.5.2.2.13 Register 14 (Interrupt Status Register)

The Interrupt Status register identifies the events in the MAC that can generate interrupt. All interrupt events are generated only when the corresponding optional feature is selected during core configuration and enabled during operation. Therefore, these bits are reserved when the corresponding features are not present in the core.

Field	Field name	Description	Reset	Access
31:11	-	Reserved	0000H	RO
10	LPIIS	LPI Interrupt Status When the Energy Efficient Ethernet feature is enabled, this bit is set for any LPI state entry or exit in the MAC Transmitter or Receiver. This bit is cleared on reading Bit 0 of Register 12 (LPI Control and Status Register). In all other modes, this bit is reserved.	0	RO
9	TSIS	Timestamp Interrupt Status When the Advanced Timestamp feature is enabled, this bit is set when any of the following conditions is true: • The system time value equals or exceeds the value specified in the Target Time High and Low registers. • There is an overflow in the seconds register. • The Auxiliary snapshot trigger is asserted. This bit is cleared on reading Bit 0 of Register 458 (Timestamp Status Register). If default Timestamping is enabled, when set, this bit indicates that the system time value is equal to or exceeds the value specified in the Target Time registers. In this mode, this bit is cleared after the completion of the read of this bit. In all other modes, this bit is reserved.	0	RO/R_SS_RC
8	-	Reserved	000H	RO
7	MMCRXIPIS	MMC Receive Checksum Offload Interrupt Status This bit is set high when an interrupt is generated in the MMC Receive Checksum Offload Interrupt Register. This bit is cleared when all the bits in this interrupt register are cleared. This bit is valid only when you select the optional MMC module and Checksum Offload Engine (Type 2) during core configuration.	0	RO
6	MMCTXIS	MMC Transmit Interrupt Status This bit is set high when an interrupt is generated in the MMC Transmit Interrupt Register. This bit is cleared when all the bits in this interrupt register are cleared. This bit is valid only when you select the optional MMC	0	RO

		module during core configuration.		
5	MMCRXIS	MMC Receive Interrupt Status This bit is set high when an interrupt is generated in the MMC Receive Interrupt Register. This bit is cleared when all the bits in this interrupt register are cleared. This bit is valid only when you select the optional MMC module during core configuration.	0	RO
4	MMCIS	MMC Interrupt Status This bit is set high when any of the Bits [7:5] is set high and cleared only when all of these bits are low. This bit is valid only when you select the optional MMC module during core configuration.	0	RO
3	PMTIS	PMT Interrupt Status This bit is set when a Magic packet or Wake-on-LAN frame is received in the power-down mode (see Bits 5 and 6 in the PMT Control and Status Register). This bit is cleared when both Bits[6:5] are cleared because of a read operation to the PMT Control and Status register. This bit is valid only when you select the optional PMT module during core configuration.	0	RO
2	PCSANCIS	PCS Auto-Negotiation Complete This bit is set when the Auto-negotiation is completed in the TBI, RTBI, or SGMII PHY interface (Bit 5 in Register 49 (AN Status Register)). This bit is cleared when you perform a read operation to the AN Status register. This bit is valid only when you select the optional TBI, RTBI, or SGMII PHY interface during core configuration and operation.	0	RO
1	PCSLCHGIS	PCS Link Status Changed This bit is set because of any change in Link Status in the TBI, RTBI, or SGMII PHY interface (Bit 2 in Register 49 (AN Status Register)). This bit is cleared when you perform a read operation on the AN Status register. This bit is valid only when you select the optional TBI, RTBI, or SGMII PHY interface during core configuration and operation.	0	RO
0	RGSMIIIS	RGMII or SMII Interrupt Status This bit is set because of any change in value of the Link Status of RGMII or SMII interface (Bit 3 in Register 54 (SGMII/RGMII/SMII Status Register)). This bit is cleared when you perform a read operation on the SGMII/RGMII/SMII Status Register. This bit is valid only when you select the optional RGMII or SMII PHY interface during core configuration and operation.	0	RO

9.5.2.2.14 Register 15 (Interrupt Mask Register)

The Interrupt Mask Register bits enable you to mask the interrupt signal because of the corresponding event in the Interrupt Status Register.

Field	Field name	Description	Reset	Access
31:11	-	Reserved	00H	RO
10	LPIIM	LPI Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of the LPI Interrupt Status bit in Register 14 (Interrupt Status Register).	0	R_W

		This bit is valid only when you select the Energy Efficient Ethernet feature during core configuration. In all other modes, this bit is reserved.		
9	TSIM	Timestamp Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of Timestamp Interrupt Status bit in Register 14 (Interrupt Status Register). This bit is valid only when IEEE1588 timestamping is enabled. In all other modes, this bit is reserved.	0	R_W
8:4	-	Reserved	00H	RO
3	PMTIM	PMT Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of PMT Interrupt Status bit in Register 14 (Interrupt Status Register).	0	R_W
2	PCSANCIM	PCS AN Completion Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of PCS Auto-negotiation complete bit in Register 14 (Interrupt Status Register).	0	R_W
1	PCSLCHGIM	PCS Link Status Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of the PCS Link-status changed bit in Register 14 (Interrupt Status Register).	0	R_W
0	RGSMIIIM	RGMII or SMII Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of the RGMII or SMII Interrupt Status bit in Register 14 (Interrupt Status Register).	0	R_W

9.5.2.2.15 Register 16 (MAC Address0 High Register)

The MAC Address0 High register holds the upper 16 bits of the first 6-byte MAC address of the station. The first DA byte that is received on the (G)MII interface corresponds to the LS byte (Bits [7:0]) of the MAC Address Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (G)MII as the destination address, then the MacAddress0 Register [47:0] is compared with 0x665544332211.
If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address0 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

Field	Field name	Description	Reset	Access
31	AE	Address Enable This bit is always set to 1.	1	RO
30:16	-	Reserved	0000H	RO
15:0	ADDRHI	MAC Address0 [47:32] This field contains the upper 16 bits (47:32) of the first 6-byte MAC address. The MAC uses this field for filtering the received frames and inserting the MAC address in the Transmit Flow Control (PAUSE) Frames.	FFFFH	R_W

9.5.2.2.16 Register 17 (MAC Address0 Low Register)

The MAC Address0 Low register holds the lower 32 bits of the 6-byte first MAC address of the station.

Field	Field name	Description	Reset	Access
31:0	ADDRLO	MAC Address0 [31:0] This field contains the lower 32 bits of the first 6-byte MAC	FFFF_FFFF	R_W

		address. This is used by the MAC for filtering the received frames and inserting the MAC address in the Transmit Flow Control (PAUSE) Frames.	H	

9.5.2.2.17 Register 18 (MAC Address1 High Register)

The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

Field	Field name	Description	Reset	Access
31	AE		0	RO
30	SA		0	R_W
29:24	MBC	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 registers. Each bit controls the masking of the bytes as follows: • Bit 29: Register 18[15:8] • Bit 28: Register 18[7:0] • Bit 27: Register 19[31:24] • ... • Bit 24: Register 19[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.	000000	R_W
23:16	-	Reserved	00H	RO
15:0	ADDRHI	MAC Address1 [47:32] This field contains the upper 16 bits (47:32) of the second 6-byte MAC address.	FFFFH	R_W

9.5.2.2.18 Register 19 (MAC Address1 Low Register)

The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.

Field	Field name	Description	Reset	Access
31:0	ADDRLO	MAC Address1 [31:0] This field contains the lower 32 bits of the second 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	FFFF_FFFFH	R_W

NOTE:

- The descriptions for registers 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, and 46 (MAC Address2 High Register through MAC Address15 High Register) are the same as for the Register 18 (MAC Address1 High Register).
- The descriptions for registers 21, 23, 25, 27, 29, 31, 33, 35, 37, 38, 41, 43, 45, and 47 (MAC Address2 Low Register through MAC Address15 Low Register) are the same as for the Register 19 (MAC Address1 Low Register).
- The descriptions for registers 512, 514, 516, 518, 520, 522, 524, 526, 528, 530, 532, 534, 536, 538, 540, and 542 (MAC Address16 High Register through MAC Address31 High Register) are the same as for the Register 18 (MAC Address1 High Register).
- The descriptions for registers 513, 515, 517, 519, 521, 523, 525, 527, 529, 531, 533, 535, 537, 539, 541, and 543 (MAC Address16 Low Register through MAC Address31 Low Register) are the same as for the Register 19 (MAC Address1 Low Register).

- The descriptions for registers 546, 548, 550, 552, 554, 556, 558, 560, 562, 564, 566, 568, 570, 572, 574, 576, 578, 580, 582, 584, 586, 588, 590, 592, 594, 596, 598, 600, 602, 604, 606, 608, 610, 612, 614, 616, 618, 620, 622, 624, 626, 628, 630, 632, 634, 636, 638, 640, 642, 644, 646, 648, 650, 652, 654, 656, 658, 660, 662, 664, 666, 668, 670, 672, 674, 676, 678, 680, 682, 684, 686, 688, 690, 692, 694, 696, 698, 700, 702, 704, 706, 708, 710, 712, 714, 716, 718, 720, 722, 724, 726, 728, 730, 732, and 734 (MAC Address33 High Register through MAC Address127 High Register) are the same as for the Register 544 (MAC Address32 High Register).
- The descriptions for registers 545, 547, 549, 551, 553, 555, 557, 559, 561, 563, 565, 567, 569, 571, 573, 575, 577, 579, 581, 583, 585, 587, 589, 591, 593, 595, 597, 599, 601, 603, 605, 607, 609, 611, 613, 615, 617, 619, 621, 623, 625, 627, 629, 631, 633, 635, 637, 639, 641, 643, 645, 647, 649, 651, 653, 655, 657, 659, 661, 663, 665, 667, 669, 671, 673, 675, 677, 679, 681, 683, 685, 687, 689, 691, 693, 695, 697, 699, 701, 703, 705, 707, 709, 711, 713, 715, 717, 719, 721, 723, 725, 727, 729, 731, 733, and 735 (MAC Address32 Low Register through MAC Address127 Low Register) are the same as for the Register 19 (MAC Address1 Low Register).

9.5.2.2.19 Register 544 (MAC Address32 High Register)

The MAC Address32 High register holds the upper 16 bits of the 33rd 6-byte MAC address of the station. You can configure the MAC address registers to be double-synchronized by selecting the Synchronize CSR MAC Address to Tx/Rx Clock Domain option in coreConsultant. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address Low Register (Register 545) are written. For proper synchronization updates, you should perform the consecutive writes to the MAC Address Low Register (Register 545) after at least four clock cycles of the destination clock.

Field	Field name	Description	Reset	Access
31	AR-	AE: Address Enable When this bit is set, the Address filter module uses the 33rd MAC address for perfect filtering. When reset, the address filter module ignores the address for filtering.	0	R_W
30:16	-	Reserved	0000H	RO
15:0	ADDRHI	ADDRHI: MAC Address32 [47:32] This field contains the upper 16 bits (47:32) of the 33rd 6-byte MAC address.	FFFFH	R_W

9.5.2.2.20 Register 48 (AN Control Register)

The AN Control register enables and/or restarts auto-negotiation. It also enables PCS loopback. This register is optional and is present only when the MAC is configured for the TBI, SGMII, or RTBI PHY interface.

Field	Field name	Description	Reset	Access
31:19	-	Reserved	0000H	RO
18	SGMRAL	SGMII RAL Control When set, this bit forces the SGMII RAL block to operate in the speed configured in the Speed and Port Select bits of the MAC Configuration register. This is useful when the SGMII interface is used in a direct MAC to MAC connection (without a PHY) and any MAC must reconfigure the speed. When reset, the SGMII RAL block operates according to the link speed status received on SGMII (from the PHY). This bit is reserved (and RO) if the SGMII PHY interface is not selected during core configuration.	0	R_W
17	LR	Lock to Reference When set, this bit enables the PHY to lock its PLL to the 125 MHz reference clock. This bit controls the pcs_lck_ref_o signal on the TBI, RTBI, or SGMII interface.	0	R_W
16	ECD	Enable Comma Detect When set, this bit enables the PHY for comma detection and word resynchronization. This bit controls the pcs_en_cdet_o signal on the TBI, RTBI, or SGMII interface.	0	R_W

15	-	Reserved	0	RO
14	ELE	External Loopback Enable When set, this bit causes the PHY to loopback the transmit data into the receive path. The pcs_ewrap_o signal is asserted high when this bit is set.	0	R_W
13	-	Reserved	0	RO
12	ANE	Auto-Negotiation Enable When set, this bit enables the MAC to perform auto-negotiation with the link partner. Clearing this bit disables the auto-negotiation.	0	R_W
11:10	-	Reserved	0h	RO
9	RAN	Restart Auto-Negotiation When set, this bit causes auto-negotiation to restart if Bit 12 (ANE) is set. This bit is self-clearing after auto-negotiation starts. This bit should be cleared for normal operation.	0	R_WS_SC
8:0	-	Reserved	00H	RO

9.5.2.2.21 Register 49 (AN Status Register)

The AN Status register indicates the link and the auto-negotiation status. This register is optional and is present only when the MAC is configured for the TBI, RTBI, or SGMII PHY interface.

Field	Field name	Description	Reset	Access
31:9	-	Reserved	00_0000H	RO
8	ES	Extended Status This bit is tied to high if the TBI or RTBI interface is selected during core configuration indicating that the MAC supports extended status information in Register 53 (TBI Extended Status Register). This bit is tied to low if the SGMII interface is selected and the TBI or RTBI interface is not selected during core configuration indicating that Register 53 is not present.	1: TBI or RTBI interface 0: SGMII interface without TBI or RTBI interface	RO
7:6	-	Reserved	00	RO
5	ANC	Auto-Negotiation Complete When set, this bit indicates that the auto-negotiation process is complete. This bit is cleared when auto-negotiation is reinitiated.	0	RO
4	-	Reserved	0	RO
3	ANA	Auto-Negotiation Ability This bit is always high because the MAC supports autonegotiation.	1	RO
2	LS	Link Status When set, this bit indicates that the link is up. When cleared, this bit indicates that the link is down.	0	R_SS_SC_LLO
1:0	-	Reserved	00	RO

9.5.2.2.22 Register 50 (Auto-Negotiation Register)

The Auto-Negotiation Advertisement register indicates the link and the auto-negotiation status. This register is optional and is present only when the MAC is configured for the TBI or RTBI PHY interface.

Field	Field name	Description	Reset	Access
31:16	-	Reserved	0000H	RO
15	NP	Next Page Support This bit is always low because the MAC does not support the next page.	0	RO
14	-	Reserved	0	RO

13:12	RFE	Remote Fault Encoding These bits provide a remote fault encoding, indicating to a link partner that a fault or error condition has occurred. The encoding of these bits is defined in IEEE 802.3z, *Section 37.2.1.5*.	00	R_W
11:9	-	Reserved	000	RO
8:7	PSE	Pause Encoding These bits provide an encoding for the PAUSE bits, indicating that the MAC is capable of configuring the PAUSE function as defined in IEEE 802.3x. The encoding of these bits is defined in IEEE 802.3z, *Section 37.2.1.4*.	11	R_W
6	HD	Half-Duplex When set high, this bit indicates that the MAC supports the half-duplex mode. This bit is always low (and RO) when the MAC is configured for the full-duplex-only mode.	1	R_w
5	FD	Full-Duplex When set high, this bit indicates that the MAC supports the full-duplex mode.	1	R_W
4:0	-	Reserved	00000	RO

9.5.2.2.23 Register 51 (Auto-Negotiation Link Partner Register)

The Auto-Negotiation Link Partner Ability register contains the advertised ability of the link partner. This register is optional and present only when the MAC is configured for the TBI or RTBI PHY interface.

Field	Field name	Description	Reset	Access
31:16	-	Reserved	00H	RO
15	NO	Next Page Support When set, this bit indicates that more next page information is available. When cleared, this bit indicates that next page exchange is not desired.	0	RO
14	ACK	Acknowledge When set, the auto-negotiation function uses this bit to indicate that the link partner has successfully received the base page of the MAC. When cleared, it indicates that the link partner did not successfully receive the base page of the MAC.	0	RO
13:12	RFE	RFE: Remote Fault Encoding These bits provide a remote fault encoding, indicating a fault or error condition of the link partner. The encoding of these bits is defined in IEEE 802.3z, *Section 37.2.1.5*.	00	RO
11:9	-	Reserved	000	RO
8:7	PSE	Pause Encoding These bits provide an encoding for the PAUSE bits, indicating that the link partner's capability of configuring the PAUSE function as defined in the IEEE 802.3x specification. The encoding of these bits is defined in IEEE 802.3z, *Section 37.2.1.4*.	00	RO
6	HD	Half-Duplex When set, this bit indicates that the link partner has the ability to operate in the half-duplex mode. When cleared, this bit indicates that the link partner does not have the ability to operate in the half-duplex mode.	0	RO
5	FD	Full-Duplex When set, this bit indicates that the link partner has the ability to operate in the fullduplex mode. When cleared, this bit indicates that the link partner does not have the ability to operate in the full-duplex mode.	0	RO
4:0	-	Reserved	00000	RO

9.5.2.2.24 Register 52 (Auto-Negotiation Expansion Register)

The Auto-Negotiation Expansion register indicates if the MAC received a new base page from the link partner. This register is optional and is present only when the MAC is configured for the TBI or RTBI PHY interface.

Field	Field name	Description	Reset	Access
31:3	-	Reserved	00H	RO
2	NPA	Next Page Ability This bit is always low because the MAC does not support the next page function.	0	RO
1	NPR	New Page Received When set, this bit indicates that the MAC has received a new page. This bit is cleared when read.	0	RO
0	-	Reserved	0	RO

9.5.2.2.25 Register 53 (TBI Extended Status Register)

The TBI Extended Status register indicates all modes of operation of the MAC. This register is optional and is present only when the MAC is configured for the TBI or RTBI PHY interface.

Field	Field name	Description	Reset	Access
31:16	-	Reserved	0000H	RO
15	GFD	1000BASE-X Full-Duplex Capable This bit indicates that the MAC is able to perform the full-duplex and 1000BASE-X operations.	1	RO
14	GHD	1000BASE-X Half-Duplex Capable This bit indicates that the MAC is able to perform the half-duplex and 1000BASE-X operations. This bit is always low when the MAC is configured for the full-duplex-only operation during core configuration.	1	RO
13:0	-	Reserved	0000H	RO

9.5.2.2.26 Register 256 (Layer 3 and Layer 4 Control Register 0)

This register controls the operations of the filter 0 of Layer 3 and Layer 4. This register is reserved if the Layer 3 and Layer 4 Filtering feature is not selected during core configuration.

Field	Field name	Description	Reset	Access
31:22	-	Reserved	000H	RO
21	L4DPIM0	Layer 4 Destination Port Inverse Match Enable When set, this bit indicates that the Layer 4 Destination Port number field is enabled for inverse matching. When reset, this bit indicates that the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when Bit 20 (L4DPM0) is set high.	0	R_W
20	L4DPM0	Layer 4 Destination Port Match Enable When set, this bit indicates that the Layer 4 Destination Port number field is enabled for matching. When reset, the MAC ignores the Layer 4 Destination Port number field for matching.	0	R_W
19	L4SPIM0	Layer 4 Source Port Inverse Match Enable When set, this bit indicates that the Layer 4 Source Port number field is enabled for inverse matching. When reset, this	0	R_W

		bit indicates that the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when Bit 18 (L4SPM0) is set high.		
18	L4SPM0	Layer 4 Source Port Match Enable When set, this bit indicates that the Layer 4 Source Port number field is enabled for matching. When reset, the MAC ignores the Layer 4 Source Port number field for matching.	0	R_W
17	-	Reserved	0	RO
16	L4PEN0	Layer 4 Protocol Enable When set, this bit indicates that the Source and Destination Port number fields for UDP frames are used for matching. When reset, this bit indicates that the Source and Destination Port number fields for TCP frames are used for matching. The Layer 4 matching is done only when either L4SPM0 or L4DPM0 bit is set high.	0	R_W
15:11	L3HDBM0	Layer 3 IP DA Higher Bits Match **IPv4 Frames:** This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 frames. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • ... • 31: All bits except MSb are masked. **IPv6 Frames:** Bits [12:11] of this field correspond to Bits [6:5] of L3HSBM0, which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 frames. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: •0: No bits are masked. •1: LSb[0] is masked. •2: Two LSbs [1:0] are masked. • … •127: All bits except MSb are masked. This field is valid and applicable only if L3DAM0 or L3SAM0 is set high.	00H	R_W
10:6	L3HSBM0	Layer 3 IP SA Higher Bits Match **IPv4 Frames:** This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 frames. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • ... • 31: All bits except MSb are masked. **IPv6 Frames:** This field contains Bits [4:0] of the field that indicates the number of higher bits of IP Source or Destination Address matched in the IPv6 frames. This field is valid and applicable only if L3DAM0 or L3SAM0 is set high.	00H	R_W
5	L3DAIM0	Layer 3 IP DA Inverse Match Enable When set, this bit indicates that the Layer 3 IP Destination Address field is enabled for inverse matching. When reset, this bit indicates that the Layer 3 IP Destination Address field is	0	R_W

		enabled for perfect matching. This bit is valid and applicable only when Bit 4 (L3DAM0) is set high.		
4	L3DAM0	Layer 3 IP DA Match Enable When set, this bit indicates that Layer 3 IP Destination Address field is enabled for matching. When reset, the MAC ignores the Layer 3 IP Destination Address field for matching. **Note**: When Bit 0 (L3PEN0) is set, you should set either this bit or Bit 2 (L3SAM0) because either IPv6 DA or SA can be checked for filtering.	0	R_W
3	L3SAIM0	Layer 3 IP SA Inverse Match Enable When set, this bit indicates that the Layer 3 IP Source Address field is enabled for inverse matching. When reset, this bit indicates that the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when Bit 2 (L3SAM0) is set high.	0	R_W
2	L3SAM0	Layer 3 IP SA Match Enable When set, this bit indicates that the Layer 3 IP Source Address field is enabled for matching. When reset, the MAC ignores the Layer 3 IP Source Address field for matching. **Note**: When Bit 0 (L3PEN0) is set, you should set either this bit or Bit 4 (L3DAM0) because either IPv6 SA or DA can be checked for filtering.	0	R_W
1	-	Reserved	0	RO
0	L3PEN0	Layer 3 Protocol Enable When set, this bit indicates that the Layer 3 IP Source or Destination Address matching is enabled for the IPv6 frames. When reset, this bit indicates that the Layer 3 IP Source or Destination Address matching is enabled for the IPv4 frames. The Layer 3 matching is done only when either L3SAM0 or L3DAM0 bit is set high.	0	R_W

9.5.2.2.27 Register 257 (Layer 4 Address Register 0)

You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option in coreConsultant. If the Layer 3 and Layer 4 Address Registers are configured to be double-synchronized to the Rx clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform the consecutive writes to the same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.
If the Layer 3 and Layer 4 Filtering feature is not selected during core configuration, this register and registers 260 through 299 are reserved (RO with default value).

Field	Field name	Description	Reset	Access
31:16	L4DP0	Layer 4 Destination Port Number Field When Bit 16 (L4PEN0) is reset and Bit 20 (L4DPM0) is set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 frames. When Bit 16 (L4PEN0) and Bit 20 (L4DPM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 frames.	0000H	R_W
15:0	L4SP0	Layer 4 Source Port Number Field When Bit 16 (L4PEN0) is reset and Bit 20 (L4DPM0) is set in	0000H	R_W

		Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 frames. When Bit 16 (L4PEN0) and Bit 20 (L4DPM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 frames.		

9.5.2.2.28 Register 260 (Layer 3 Address 0 Register 0)

For IPv4 frames, the Layer 3 Address 0 Register 0 contains the 32-bit IP Source Address field. For IPv6 frames, it contains Bits [31:0] of the 128-bit IP Source Address or Destination Address field.

Field	Field name	Description	Reset	Access
31:0	L3A00	Layer 3 Address 0 Field When Bit 0 (L3PEN0) and Bit 2 (L3SAM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with Bits [31:0] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN0) and Bit 4 (L3DAM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with Bits [31:0] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN0) is reset and Bit 2 (L3SAM0) is set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with the IP Source Address field in the IPv4 frames.	0H	R_W

9.5.2.2.29 Register 261 (Layer 3 Address 2 Register)

For IPv4 frames, the Layer 3 Address 1 Register 0 contains the 32-bit IP Destination Address field. For IPv6 frames, it contains Bits [63:32] of the 128-bit IP Source Address or Destination Address field. **Tabl**

Field	Field name	Description	Reset	Access
31:0	L3A10	Layer 3 Address 1 Field When Bit 0 (L3PEN0) and Bit 2 (L3SAM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with Bits [63:32] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN0) and Bit 4 (L3DAM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with Bits [63:32] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN0) is reset and Bit 4 (L3DAM0) is set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with the IP Destination Address field in the IPv4 frames.	0H	R_W

9.5.2.2.30 Register 262 (Layer 3 Address 3 Register)

For IPv4 frames, the Layer 3 Address 2 Register 0 is reserved. For IPv6 frames, it contains Bits [95:64] of the 128-bit IP Source Address or Destination Address field.

Field	Field name	Description	Reset	Access

31:0	L3A20	Layer 3 Address 2 Field When Bit 0 (L3PEN0) and Bit 2 (L3SAM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with Bits [95:64] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN0) and Bit 4 (L3DAM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains value to be matched with Bits [95:64] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN0) is reset in Register 256 (Layer 3 and Layer 4 Control Register 0), this register is not used.	0H	R_W

9.5.2.2.31 Register 263 (Layer 3 Address 4 Register)

Field	Field name	Description	Reset	Access
31:0	L3A30	Layer 3 Address 3 Field When Bit 0 (L3PEN0) and Bit 2 (L3SAM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with Bits [127:96] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN0) and Bit 4 (L3DAM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with Bits [127:96] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN0) is reset in Register 256 (Layer 3 and Layer 4 Control Register 0), this register is not used.	0H	R_W

- Registers 268, 280, and 292 are similar to Register 256 (Layer 3 and Layer 4 Control Register 0).
- Registers 269, 281, and 293 are similar to Register 257 (Layer 4 Address Register 0).
- Registers 272, 284, and 296 are similar to Register 260 (Layer 3 Address 0 Register 0).
- Registers 273, 285, and 297 are similar to Register 261 (Layer 3 Address 1 Register 0).
- Registers 274, 286, and 298 are similar to Register 262 (Layer 3 Address 2 Register 0).
- Registers 275, 287, and 299 are similar to Register 263 (Layer 3 Address 3 Register 0).
- Registers 268 through 275 are present when you select more than one Layer 3 and Layer 4 filters in coreConsultant.
- Registers 280 through 287 are present when you select more than two Layer 3 and Layer 4 filters in coreConsultant.
- Registers 292 through 299 are present when you select four Layer 3 and Layer 4 filters in coreConsultant.

9.5.2.2.32 Register 320 (Hash Table Register 0)

This register contains the first 32 bits of the hash table when the width of the Hash table is 128 bits or 256 bits. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.
The 128-bit or 256-bit Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming frame is passed through the CRC logic and the upper seven (eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register *X*), and the least significant five bits determine the bit within the register. For example, a hash value of 7b'1100000 (in 128-bit Hash) selects Bit 0 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.
The hash value of the destination address is calculated in the following way:

1. Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).
2. Perform bitwise reversal for the value obtained in Step 1.
3. Take the upper 7 (or 8) bits from the value obtained in Step 2.

If the corresponding bit value of the register is 1’b1, the frame is accepted. Otherwise, it is rejected. If the Bit 1 (Pass All Multicast) is set in Register 1 (MAC Frame Filter), then all multicast frames are accepted regardless of the multicast hash values.

If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register *X* registers are written.

Field	Field name	Description	Reset	Access
31:0	HT31T0	First 32 bits of Hash Table This field contains the first 32 Bits (31:0) of the Hash table.	0H	R_W

Registers 321 through 327 are similar to Register 320 (Hash Table Register 0). Registers 324 through 327 are present only when you select the 256-bit Hash table during core configuration.

9.5.2.2.33 Register 353 (VLAN Tag Inclusion or Replacement Register)

The VLAN Tag Inclusion or Replacement register contains the VLAN tag for insertion or replacement in the transmit frames. This register is present only when the Enable SA, VLAN, and CRC Insertion on TX option is selected during core configuration.

Field	Field name	Description	Reset	Access
31:20	-	Reserved	00H	RO
19	CSVL	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted frames. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the transmitted frames.	0	R_W
18	VLP	VLAN Priority Control When this bit is set, the control Bits [17:16] are used for VLAN deletion, insertion, or replacement. When this bit is reset, the mti_vlan_ctrl_i control input is used, and Bits [17:16] are ignored.	0	R_W
17:16	VLC	VLAN Tag Control in Transmit Frames • 2'b00: No VLAN tag deletion, insertion, or replacement • 2'b01: VLAN tag deletion The MAC removes the VLAN type (bytes 13 and 14) and VLAN tag (bytes 15 and 16) of all transmitted frames with VLAN tags. • 2'b10: VLAN tag insertion The MAC inserts VLT in bytes 15 and 16 of the frame after inserting the Type value (0x8100/0x88a8) in bytes 13 and 14. This operation is performed on all transmitted frames, irrespective of whether they already have a VLAN tag. • 2'b11: VLAN tag replacement The MAC replaces VLT in bytes 15 and 16 of all VLAN-type transmitted frames (Bytes 13 and 14 are 0x8100/0x88a8). **Note**: Changes to this field take effect only on the start of a frame. If you write this register field when a frame is being transmitted, only the subsequent frame can use the updated value, that is, the current frame does not use the updated value.	0	R_W
15:0	VLT	VLAN Tag for Transmit Frames This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority, Bit 12 is the CFI/DEI, and Bits[11:0] are the VLAN tag's VID field.	0	R_W

9.5.2.2.34 Register 354 (VLAN Hash Table Register)

The 16-bit Hash table is used for group address filtering based on VLAN tag when Bit 18 (VTHM) of Register 7 (VLAN Tag Register) is set. For hash filtering, the content of the 16-bit VLAN tag or 12-bit VLAN ID (based on Bit

16 (ETV) of VLAN Tag Register) in the incoming frame is passed through the CRC logic and the upper four bits of the calculated CRC are used to index the contents of the VLAN Hash table. For example, a hash value of 4b'1000 selects Bit 8 of the VLAN Hash table.
The hash value of the destination address is calculated in the following way:

1. Calculate the 32-bit CRC for the VLAN tag or ID (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).
2. Peform bitwise reversal for the value obtained in Step 1.
3. Take the upper four bits from the value obtained in Step 2.

If the corresponding bit value of the register is 1’b1, the frame is accepted. Otherwise, it is rejected. If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[15:8] (in little-endian mode) or Bits[7:0] (in big-endian mode) of this register are written.

Field	Field name	Description	Reset	Access
31:16	-	Reserved	0H	RO
15:0	VLHT	VLAN Hash Table This field contains the 16-bit VLAN Hash Table.	0H	R_W

9.5.2.3 IEEE 1588 Timestamp Registers

This section describes the updated registers required to support the IEEE 1588 functions. These registers, described at a high level in Table 10-6.

9.5.2.3.1 Register 448 (Timestamp Control Register)

This register controls the operation of the System Time generator and the processing of PTP packets for timestamping in the Receiver.

- Bits[5:1] are reserved when External Timestamp Input feature is enabled.
- Bits[19:8] are reserved and read-only when Advanced Timestamp feature is not enabled.
- Bits[28:24] are reserved and read-only when Auxiliary Snapshot feature is not enabled.
- Release 3.60a onwards, the functions of Bits 17 and 16 (SNAPTYPSEL) have changed. These functions are not backward compatible with the functions described in release 3.50a.

Field	Field name	Description	Reset	Access
31:29	-	Reserved	000	RO
28	ATSEN3	Auxiliary Snapshot 3 Enable This field controls capturing the Auxiliary Snapshot Trigger 3. When this bit is set, the Auxiliary snapshot of event on ptp_aux_trig_i[3] input is enabled. When this bit is reset, the events on this input are ignored. This bit is reserved when the Add IEEE 1588 Auxiliary Snapshot option is not selected during core configuration or the selected number in the Number of IEEE 1588 Auxiliary Snapshot Inputs option is less than four.	0	R_W
27	ATSEN2	Auxiliary Snapshot 2 Enable This field controls capturing the Auxiliary Snapshot Trigger 2. When this bit is set, the Auxiliary snapshot of event on ptp_aux_trig_i[2] input is enabled. When this bit is reset, the events on this input are ignored. This bit is reserved when the Add IEEE 1588 Auxiliary Snapshot option is not selected during core configuration or the selected number in the Number of IEEE 1588 Auxiliary Snapshot Inputs option is less than three.	0	R_W
26	ATSEN1	Auxiliary Snapshot 1 Enable This field controls capturing the Auxiliary Snapshot	0	R_W

		Trigger 1. When this bit is set, the Auxiliary snapshot of event on ptp_aux_trig_i[1] input is enabled. When this bit is reset, the events on this input are ignored. This bit is reserved when the Add IEEE 1588 Auxiliary Snapshot option is not selected during core configuration or the selected number in the Number of IEEE 1588 Auxiliary Snapshot Inputs option is less than two.		
25	ATSEN0	Auxiliary Snapshot 0 Enable This field controls capturing the Auxiliary Snapshot Trigger 0. When this bit is set, the Auxiliary snapshot of event on ptp_aux_trig_i[0] input is enabled. When this bit is reset, the events on this input are ignored. This bit is reserved when the Add IEEE 1588 Auxiliary Snapshot option is not selected during core configuration.	0	R_W
24	ATSFC	Auxiliary Snapshot FIFO Clear When set, it resets the pointers of the Auxiliary Snapshot FIFO. This bit is cleared when the pointers are reset and the FIFO is empty. When this bit is high, auxiliary snapshots get stored in the FIFO. This bit is reserved when the Add IEEE 1588 Auxiliary Snapshot option is not selected during core configuration.	0	R_WS_SC
23:19	-	Reserved	0	RO
18	TSENMACADDR	Enable MAC address for PTP Frame Filtering When set, the DA MAC address (that matches any MAC Address register) is used to filter the PTP frames when PTP is directly sent over Ethernet.	0	R_W
17:16	SNAPTYPSEL	Select PTP packets for Taking Snapshots These bits along with Bits 15 and 14 decide the set of PTP packet types for which snapshot needs to be taken.	0	R_W
15	TSMSTRENA	Enable Snapshot for Messages Relevant to Master When set, the snapshot is taken only for the messages relevant to the master node. Otherwise, the snapshot is taken for the messages relevant to the slave node.	0	R_W
14	TSEVNTENA	Enable Timestamp Snapshot for Event Messages When set, the timestamp snapshot is taken only for event messages (SYNC, Delay_Req, Pdelay_Req, or Pdelay_Resp). When reset, the snapshot is taken for all messages except Announce, Management, and Signaling.	0	R_W
13	TSIPV4ENA	Enable Processing of PTP Frames Sent over IPv4-UDP When set, the MAC receiver processes the PTP packets encapsulated in UDP over IPv4 packets. When this bit is clear, the MAC ignores the PTP transported over UDP-IPv4 packets. This bit is set by default.	0	R_W
12	TSIPV6ENA	Enable Processing of PTP Frames Sent over IPv6-UDP When set, the MAC receiver processes PTP packets encapsulated in UDP over IPv6 packets. When this bit is clear, the MAC ignores the PTP transported over UDP-IPv6 packets.	0	R_W
11	TSIPENA	Enable Processing of PTP over Ethernet Frames When set, the MAC receiver processes the PTP packets encapsulated directly in the Ethernet frames. When this bit is clear, the MAC ignores the PTP over Ethernet packets.	0	R_W
10	TSVER2ENA	Enable PTP packet Processing for Version 2 Format When set, the PTP packets are processed using the	0	R_W

		1588 version 2 format. Otherwise, the PTP packets are processed using the version 1 format. The IEEE 1588 Version 1 and Version 2 format are described		
9	TSCTRLSSR	Timestamp Digital or Binary Rollover Control When set, the Timestamp Low register rolls over after 0x3B9A_C9FF value (that is, 1 nanosecond accuracy) and increments the timestamp (High) seconds. When reset, the rollover value of sub-second register is 0x7FFF_FFFF. The sub-second increment has to be programmed correctly depending on the PTP reference clock frequency and the value of this bit.	0	R_W
8	TSENALL	Enable Timestamp for All Frames When set, the timestamp snapshot is enabled for all frames received by the MAC.	0	R_W
7:6	-	Reserved	0	RO
5	TSADDREG	Addend Reg Update When set, the content of the Timestamp Addend register is updated in the PTP block for fine correction. This is cleared when the update is completed. This register bit should be zero before setting it.	0	R_WS_SC
4	TSTRIG	Timestamp Interrupt Trigger Enable When set, the timestamp interrupt is generated when the System Time becomes greater than the value written in the Target Time register. This bit is reset after the generation of the Timestamp Trigger Interrupt.	0	R_WS_SC
3	TSUPDT	Timestamp Update When set, the system time is updated (added or subtracted) with the value specified in Register 452 (System Time – Seconds Update Register) and Register 453 (System Time – Nanoseconds Update Register). This bit should be read zero before updating it. This bit is reset when the update is completed in hardware. The “Timestamp Higher Word” register (if enabled during core configuration) is not updated.	0	R_WS_SC
2	TSINIT	Timestamp Initialize When set, the system time is initialized (overwritten) with the value specified in the Register 452 (System Time – Seconds Update Register) and Register 453 (System Time – Nanoseconds Update Register). This bit should be read zero before updating it. This bit is reset when the initialization is complete. The “Timestamp Higher Word” register (if enabled during core configuration) can only be initialized.	0	R_WS_SC
1	TSCFUPDT	Timestamp Fine or Coarse Update When set, this bit indicates that the system times update should be done using the fine update method. When reset, it indicates the system timestamp update should be done using the Coarse method.	0	R_W
0	TSENA	Timestamp Enable When set, the timestamp is added for the transmit and receive frames. When disabled, timestamp is not added for the transmit and receive frames and the Timestamp Generator is also suspended. You need to initialize the Timestamp (system time) after enabling this mode. On the receive side, the MAC processes the 1588 frames only if this bit is set.	0	R_W

Below table indicates the PTP messages, for which a snapshot is taken depending on Bits [17:14] (SNAPTYPSEL), in Register 448 (Timestamp Control Register).

SNAPTYPSEL(Bits 17:16)	TSMSTRENA (Bit 15)	TSEVNTENA (Bit 14)	PTP Messages
00	X	0	SYNC, Follow_Up, Delay_Req, Delay_Resp
00	0	1	SYNC
00	1	1	Delay_Req
01	X	0	SYNC, Follow_Up, Delay_Req, Delay_Resp, Pdelay_Req, Pdelay_Resp, Pdelay_Resp_Follow_Up
01	0	1	SYNC, Pdelay_Req, Pdelay_Resp
01	1	1	Delay_Req, Pdelay_Req, Pdelay_Resp
10	X	X	SYNC, Delay_Req
11	X	X	Pdelay_Req, Pdelay_Resp

9.5.2.3.2 Register 449 (Sub-Second Increment Register)

This register is present only when the IEEE 1588 timestamp feature is selected without an external timestamp input. In the Coarse Update mode (TSCFUPDT bit in Register 448), the value in this register is added to the system time every clock cycle of clk_ptp_ref_i. In the Fine Update mode, the value in this register is added to the system time whenever the Accumulator gets an overflow.

Field	Field name	Description	Reset	Access
31:8	-	Reserved	0H	RO
7:0	SSINC	Sub-second Increment Value The value programmed in this field is accumulated every clock cycle (of clk_ptp_i) with the contents of the sub-second register. For example, when PTP clock is 50 MHz (period is 20 ns), you should program 20 (0x14) when the System Time- Nanoseconds register has an accuracy of 1 ns [Bit 9 (TSCTRLSSR) is set in Register 448 (Timestamp Control Register)]. When TSCTRLSSR is clear, the Nanoseconds register has a resolution of ~0.465ns. In this case, you should program a value of 43 (0x2B) that is derived by 20ns/0.465.	00H	R_W

9.5.2.3.3 Register 450 (System Time - Seconds Register)

The System Time—Seconds register, along with System Time—Nanoseconds register, indicates the current value of the system time maintained by the MAC. Though it is updated on a continuous basis, there is some delay from the actual time because of clock domain transfer latencies (from clk_ptp_ref_i to clk_csr_i). These registers (450 and 451) are present only when the IEEE 1588 Timestamp feature is selected without external timestamp input.

Field	Field name	Description	Reset	Access
31:16	TSS	Timestamp Second The value in this field indicates the current value in seconds of the System Time maintained by the MAC.	0H	RO

9.5.2.3.4 Register 451 (System Time - Nanoseconds Register)

Field	Field name	Description	Reset	Access
31	-	Reserved	0H	RO
30:0	TSSS	Timestamp Sub Seconds The value in this field has the sub	0H	RO

		second representation of time, with an accuracy of 0.46 ns. When Bit 9 (TSCTRLSSR) is set in Register 448 (Timestamp Control Register), each bit represents 1 ns and the maximum value is 0x3B9A_C9FF, after which it rolls-over to zero.		

9.5.2.3.5 Register 452 (System Time – Seconds Update Register)

The System Time—Seconds Update register, along with the System Time—Nanoseconds Update register, initializes or updates the system time maintained by the MAC. You must write both of these registers before setting the TSINIT or TSUPDT bits in the Timestamp Control register. This register is present only when the IEEE 1588 Timestamp feature is selected without external timestamp input.

Field	Field name	Description	Reset	Access
31:0	TSS	Timestamp Second The value in this field indicates the time in seconds to be initialized or added to the system time.	0H	R_W

9.5.2.3.6 Register 453 (System Time – Nanoseconds Update Register)

This register is present only when IEEE 1588 timestamp feature is selected without external timestamp input.

Field	Field name	Description	Reset	Access
31	ADDSUB	Add or Subtract Time When this bit is set, the time value is subtracted with the contents of the update register. When this bit is reset, the time value is added with the contents of the update register.	0	R_W
30:0	TSSS	Timestamp Sub Seconds The value in this field has the sub second representation of time, with an accuracy of 0.46 ns. When Bit 9 (TSCTRLSSR) is set in Register 448 (Timestamp Control Register), each bit represents 1 ns and the programmed value should not exceed 0x3B9A_C9FF.	0	R_W

9.5.2.3.7 Register 454 (Timestamp Addend Register)

This register is present only when the IEEE 1588 Timestamp feature is selected without external timestamp input. This register value is used only when the system time is configured for Fine Update mode (TSCFUPDT bit in Register 448). This register content is added to a 32-bit accumulator in every clock cycle (of clk_ptp_ref_i) and the system time is updated whenever the accumulator overflows.

Field	Field name	Description	Reset	Access
31:0	TSAR	Timestamp Addend Register This field indicates the 32-bit time value to be added to the Accumulator register to achieve time synchronization.	0H	R_W

9.5.2.3.8 Register 455 (Target Time Seconds Register)

The Target Time Seconds register, along with Target Time Nanoseconds register, is used to schedule an interrupt event (Register 458[1] when Advanced Timestamping is enabled; otherwise, TS interrupt bit in Register14[9]) when the system time exceeds the value programmed in these registers. This register is present only when the IEEE 1588 Timestamp feature is selected without external timestamp input.

Field	Field name	Description	Reset	Access
31:0		Target Time Seconds Register This register stores the time in	0H	R_W

	TSTR	seconds. When the timestamp value matches or exceeds both Target Timestamp registers, then based on Bits [6:5] of Register 459 (PPS Control Register), the MAC starts or stops the PPS signal output and generates an interrupt (if enabled).		

9.5.2.3.9 Register 456 (Target Time Nanoseconds Register)

This register is present only when the IEEE 1588 Timestamp feature is selected without external timestamp input.

Field	Field name	Description	Reset	Access
31	TRGTBUSY	Target Time Register Busy The MAC sets this bit when the PPSCMD field (Bit [3:0]) in Register 459 (PPS Control Register) is programmed to 010 or 011. Programming the PPSCMD field to 010 or 011, instructs the MAC to synchronize the Target Time Registers to the PTP clock domain. The MAC clears this bit after synchronizing the Target Time Registers to the PTP clock domain The application must not update the Target Time Registers when this bit is read as 1. Otherwise, the synchronization of the previous programmed time gets corrupted. This bit is reserved when the Enable Flexible Pulse-Per-Second Output feature is not selected.	0H	R_WS_SC
30:0	TTSLO	Target Timestamp Low Register This register stores the time in (signed) nanoseconds. When the value of the timestamp matches the both Target Timestamp registers, then based on the TRGTMODSEL0 field (Bits [6:5]) in Register 459 (PPS Control Register), the MAC starts or stops the PPS signal output and generates an interrupt (if enabled). This value should not exceed 0x3B9A_C9FF when Bit 9 (TSCTRLSSR) is set in Register 448 (Timestamp Control Register). The actual start or stop time of the PPS signal output may have an error margin up to one unit of sub-second increment value.	0	R_W

9.5.2.3.10 Register 457 (System Time – Higher Word Seconds Register)

This register is present only when the IEEE 1588 Advanced Timestamp feature is selected without an external timestamp input.

Field	Field name	Description	Reset	Access
31:16	-	Reserved	0H	RO
15:0	TSHWR	Timestamp Higher Word Register This field contains the most significant 16-bits of the timestamp seconds value. This register is optional and can be selected using the Enable IEEE 1588 Higher Word Register option during core configuration. The register is directly written to initialize the value. This register is incremented when there is an overflow from the 32-bits of the System Time - Seconds register.	0H	R_W_SU

9.5.2.3.11 Register 458 (Timestamp Status Register)

This register is present only when the Advanced IEEE 1588 Timestamp feature is selected. All bits except Bits[27:25] gets cleared when the host reads this register.

Field	Field name	Description	Reset	Access
31:30	-	Reserved	0H	RO
29:25	ATSNS	Number of Auxiliary Timestamp Snapshots This field indicates the number of Snapshots available in the FIFO. A value equal to the selected depth of FIFO (4, 8, or 16) indicates that the Auxiliary Snapshot FIFO is full. These bits are cleared (to 00000) when the Auxiliary snapshot FIFO clear bit is set. This bit is valid only if the Add IEEE 1588 Auxiliary Snapshot option is selected during core configuration.	0	RO
24	ATSSTM	Auxiliary Timestamp Snapshot Trigger Missed This bit is set when the Auxiliary timestamp snapshot FIFO is full and external trigger was set. This indicates that the latest snapshot is not stored in the FIFO. This bit is valid only if the Add IEEE 1588 Auxiliary Snapshot option is selected during core configuration.		
23:20	-	Reserved	0	RO
19:16	ATSSTN	Auxiliary Timestamp Snapshot Trigger Identifier These bits identify the Auxiliary trigger inputs for which the timestamp available in the Auxiliary Snapshot Register is applicable. When more than one bit is set at the same time, it means that corresponding auxiliary triggers were sampled at the same clock. These bits are applicable only if the number of Auxiliary snapshots is more than one. One bit is assigned for each trigger as shown in the following list: • Bit 16: Auxiliary trigger 0 • Bit 17: Auxiliary trigger 1 • Bit 18: Auxiliary trigger 2 • Bit 19: Auxiliary trigger 3 The software can read this register to find the triggers that are set when the timestamp is taken.	0	R_SS_RC
15:10	-	Reserved	00H	RO
9	TSTRGTERR3	Timestamp Target Time Error This bit is set when the target time, being programmed in Register 496 and Register 497, is already elapsed. This bit is cleared when read by the application.	0	R_SS_RC
8	TSTARGT3	Timestamp Target Time Reached for Target Time PPS3 When set, this bit indicates that the value of system time is greater than or equal to the value specified in Register 496 (PPS3 Target Time High Register) and Register 497 (PPS3 Target Time Low Register).	0	R_SS_RC
7	TSTRGTERR2	Timestamp Target Time Error This bit is set when the target time, being programmed in Register 488 and Register 489, is already elapsed. This bit is cleared when read by the application.	0	R_SS_RC
6	TSTARGT2	Timestamp Target Time Reached for Target Time PPS2 When set, this bit indicates that the value of system time is greater than or equal to the value specified in Register 488 (PPS2 Target Time High Register) and Register 489 (PPS2 Target Time Low Register).	0	R_SS_RC
5	TSTRGTERR1	Timestamp Target Time Error This bit is set when the target time, being programmed in Register 480 and Register 481, is already elapsed. This bit is cleared when read by the application.	0	R_SS_RC

4	TSTARGT1	Timestamp Target Time Reached for Target Time PPS1 When set, this bit indicates that the value of system time is greater than or equal to the value specified in Register 480 (PPS1 Target Time High Register) and Register 481 (PPS1 Target Time Low Register).	0	R_SS_RC
3	TSTRGTERR	Timestamp Target Time Error This bit is set when the target time, being programmed in Register 455 and Register 456, is already elapsed. This bit is cleared when read by the application.	0	R_SS_RC
2	AUXTSTRIG	Auxiliary Timestamp Trigger Snapshot This bit is set high when the auxiliary snapshot is written to the FIFO. This bit is valid only if the Enable IEEE 1588 Auxiliary Snapshot feature is selected.	0	R_SS_RC
1	TSTARGT	Timestamp Target Time Reached When set, this bit indicates that the value of system time is greater than or equal to the value specified in the Register 455 (Target Time Seconds Register) and Register 456 (Target Time Nanoseconds Register).	0	R_SS_RC
0	TSSOVF	Timestamp Seconds Overflow When set, this bit indicates that the seconds value of the timestamp (when supporting version 2 format) has overflowed beyond 32'hFFFF_FFFF.	0	R_SS_RC

9.5.2.3.12 Register 459 (PPS Control Register)

This register is present only when the Advanced Timestamp feature is selected and External Timestamp is not enabled.

Note

- Bits[30:24] are valid only when four Flexible PPS outputs are selected.
- Bits[22:16] are valid only when three or more Flexible PPS outputs are selected.
- Bits[14:8] are valid only when two or more Flexible PPS outputs are selected.
- Bits[6:4] are valid only when Flexible PPS feature is selected.

Field	Field name	Description	Reset	Access
31	-	Reserved	0	RO
30:29	TRGTMODSEL3	Target Time Register Mode for PPS3 Output This field indicates the Target Time registers (register 496 and 497) mode for PPS3 output signal. This field is similar to the TRGTMODSEL0 field.	0	R_W
28:27	-	Reserved	0	RO
26:24	PPSCMD3	Flexible PPS3 Output Control This field controls the flexible PPS3 output (ptp_pps_o[3]) signal. This field is similar to PPSCMD0[2:0] in functionality.	0	R_WS_SC
23	-	Reserved	0	RO
22:21	TRGTMODSEL2	Target Time Register Mode for PPS2 Output This field indicates the Target Time registers (register 488 and 489) mode for PPS2 output signal. This field is similar to the TRGTMODSEL0 field.	0	R_W
20:19	-	Reserved	0	RO
18:16	PPSCMD2	Flexible PPS2 Output Control This field controls the flexible PPS2 output (ptp_pps_o[2]) signal. This field is similar to PPSCMD0[2:0] in functionality.	0	R_WS_SC
15	-	Reserved	0	RO
14:13	TRGTMODSEL1	Target Time Register Mode for PPS1 Output This field indicates the Target Time registers (register 480 and 481) mode for PPS1 output signal. This field is similar to	0	R_W

		the TRGTMODSEL0 field.		
12:11	-	Reserved	0	RO
10:8	PPSCMD1	Flexible PPS1 Output Control This field controls the flexible PPS1 output (ptp_pps_o[1]) signal. This field is similar to PPSCMD0[2:0] in functionality.	0	R_WS_SC
7	-	Reserved	0	RO
6:5	TRGTMODSEL0	Target Time Register Mode for PPS0 Output This field indicates the Target Time registers (register 455 and 456) mode for PPS0 output signal: • 00: Indicates that the Target Time registers are programmed only for generating the interrupt event. • 01: Reserved • 10: Indicates that the Target Time registers are programmed for generating the interrupt event and starting or stopping the generation of the PPS0 output signal. • 11: Indicates that the Target Time registers are programmed only for starting or stopping the generation of the PPS0 output signal. No interrupt is asserted.	0	R_W
4	PPSEN0	Flexible PPS Output Mode Enable When set low, Bits [3:0] function as PPSCTRL (backward compatible). When set high, Bits[3:0] function as PPSCMD.	0	R_W
3:0	PPSCTRL0	PPSCTRL0: PPS0 Output Frequency Control This field controls the frequency of the PPS0 output (ptp_pps_o[0]) signal. The default value of PPSCTRL is 0000, and the PPS output is 1 pulse (of width clk_ptp_i) every second. For other values of PPSCTRL, the PPS output becomes a generated clock of following frequencies: • 0001: The binary rollover is 2 Hz, and the digital rollover is 1 Hz. • 0010: The binary rollover is 4 Hz, and the digital rollover is 2 Hz. • 0011: The binary rollover is 8 Hz, and the digital rollover is 4 Hz. • 0100: The binary rollover is 16 Hz, and the digital rollover is 8 Hz. • ... • 1111: The binary rollover is 32.768 KHz, and the digital rollover is 16.384 KHz. **Note**: In the binary rollover mode, the PPS output (ptp_pps_o) has a duty cycle of 50 percent with these frequencies. In the digital rollover mode, the PPS output frequency is an average number. The actual clock is of different frequency that gets synchronized every second. For example: •When PPSCTRL = 0001, the PPS (1 Hz) has a low period of 537 ms and a high period of 463 ms •When PPSCTRL = 0010, the PPS (2 Hz) is a sequence of: - One clock of 50 percent duty cycle and 537 ms period - Second clock of 463 ms period (268 ms low and 195 ms high) •When PPSCTRL = 0011, the PPS (4 Hz) is a sequence of: - Three clocks of 50 percent duty cycle and 268 ms period - Fourth clock of 195 ms period (134 ms low and 61 ms high) This behavior is because of the non-linear toggling of bits in the digital rollover mode in Register 451 (System Time – Nanoseconds Register).	0	R_W

9.5.2.3.13 Register 460 (Auxiliary timestamp - Nanoseconds Register)

This register, along with Register 461 (Auxiliary Timestamp – Seconds Register), gives the 64-bit timestamp stored as auxiliary snapshot. The two registers together form the read port of a 64-bit wide FIFO with a depth of 4, 8, or 16 as selected during core configuration. Multiple snapshots can be stored in this FIFO. The ATSNS bits in the Timestamp Status register indicate the fill-level of this FIFO. The top of the FIFO is removed only when the last byte of Register 461 (Auxiliary Timestamp - Seconds Register) is read. In the little-endian mode, this means when Bits[31:24] are read. In big-endian mode, it corresponds to the reading of Bits[7:0] of Register 461 (Auxiliary Timestamp - Seconds Register).
This register and Register 461 (Auxiliary Timestamp - Seconds Register) are present only when you select the Auxiliary Snapshot Enable feature during core configuration.

Field	Field name	Description	Reset	Access
31	-	Reserved	0H	RO
30:0	AUXTSLO	Contains the lower 31 bits (nano-seconds field) of the auxiliary timestamp.	0H	RO

9.5.2.3.14 Register 461 (Auxiliary timestamp - Seconds Register)

Field	Field name	Description	Reset	Access
31:0	AUXTSHI	Contains the lower 32 bits of the Seconds field of the auxiliary timestamp.	0H	RO

9.5.2.3.15 Register 462 (AV MAC Control Register)

This register controls the AV traffic by identifying the AV traffic and queuing it to appropriate channel. This register is present only when you select the AV feature during core configuration.

Field	Field name	Description	Reset	Access
31:26	-	Reserved	0H	RO
25:24	PTPCH	Channel for Queuing the PTP Packets This field specifies the channel on which the untagged PTP packets, sent over the Ethernet payload and not over IPv4 or IPv6, are queued. • 00: Channel 0 • 01: Channel 1 • 10: Channel 2 • 11: Reserved These bits are reserved if the receive paths of Channel 1 or Channel 2 are not enabled.	0	R_W
23	-	Reserved	0	RO
22:21	AVCH	Channel for Queuing the AV Control Packets This field specifies the channel on which the received untagged AV control packets are queued. • 00: Channel 0 • 01: Channel 1 • 10: Channel 2 • 11: Reserved These bits are reserved if the receive paths of Channel 1 or Channel 2 are not enabled.	0	R_W
20	AVCD	AV Channel Disable When set, the MAC forwards all packets to the default Channel 0 and the values programmed in the AVP, AVCH, and PTPCH	0	R_W

		fields are ignored. This bit is reserved and read-only if Channel 1 or Channel 2 receive paths are not selected during core configuration.		
19	-	Reserved	0	RO
18:16	AVP	AV Priority for Queuing The value programmed in these bits control the receive channel (0, 1, or 2) to which an AV packet with a given priority must be queued. • If only Channel 1 receive path is enabled, then the AV packets with priority value greater than or equal to the programmed value are queued on Channel 1 and all other packets are queued on Channel 0. • If Channel 2 receive path is also enabled, then the AV packets with priority value greater than or equal to the programmed value are queued on Channel 2. The AV packets with value less than the programmed value on Channel 1 and all other packets are queued on Channel 0. These bits are applicable only if at least one additional receive channel is selected in the AV mode.	0	R_W
15:0	AVT	AV EtherType Value This field contains the value that is compared with the EtherType field of the incoming (tagged or untagged) Ethernet frame to detect an AV packet.	0	R_W

9.5.2.3.16 Register 472 (PPS0 Interval Register)

The PPS0 Interval register contains the number of units of sub-second increment value between the rising edges of PPS0 signal output (ptp_pps_o[0]).

Field	Field name	Description	Reset	Access
31:0	PPSINT	PPS0 Output Signal Interval These bits store the interval between the rising edges of PPS0 signal output in terms of units of sub-second increment value. You need to program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20ns), and desired interval between rising edges of PPS0 signal output is 100ns (that is, five units of sub-second increment value), then you should program value 4 (5 – 1) in this register.	0H	R_W

9.5.2.3.17 Register 473 (PPS0 Width Register)

The PPS0 Width register contains the number of units of sub-second increment value between the rising and corresponding falling edges of the PPS0 signal output (ptp_pps_o[0]).

Field	Field name	Description	Reset	Access
31:0	PPSWIDTH	PPS0 Output Signal Width These bits store the width between the rising edge and corresponding falling edge of the PPS0 signal output in terms of units of sub-second increment value. You need to program one value less than the required interval. For example, if PTP reference clock is 50 MHz (period of 20ns), and desired width between the rising and corresponding falling edges of PPS0 signal output is 80ns (that is, four units of sub-second increment value), then you should program value 3 (4 – 1) in this register. **Note:** The value programmed in this register must be lesser than the value programmed in Register 472 (PPS0 Interval Register).	0H	R_W

9.5.2.3.18 Register 480 (PPS1 Target Time Register)

The PPS1 Target Time Seconds register, along with PPS1 Target Time Nanoseconds register, is used to schedule an interrupt event (Bit 1 (TSTARGT) of Register 458 (Timestamp Status Register)) when the system time exceeds the value programmed in these registers. This register is present only when more than one Flexible PPS output is selected during core configuration.

Field	Field name	Description	Reset	Access
31:0	TSTRH1	PPS1 Target Time Seconds Register This register stores the time in seconds. When the timestamp value matches or exceeds both Target Timestamp registers, then based on Bits [14:13], TRGTMODSEL1, of Register 459 (PPS Control Register), the MAC starts or stops the PPS signal output and generates an interrupt (if enabled).	0H	R_W

9.5.2.3.19 Register 481 (PPS1 Target Time Nanoseconds Register)

This register is present only when more than one Flexible PPS output is selected during core configuration

Field	Field name	Description	Reset	Access
31	TRGTBUSY1	PPS1 Target Time Register Busy The MAC sets this bit when the PPSCMD1 field (Bits [10:8]) in Register 459 (PPS Control Register) is programmed to 010 or 011. Programming the PPSCMD1 field to 010 or 011 instructs the MAC to synchronize the Target Time Registers to the PTP clock domain. The MAC clears this bit after synchronizing the Target Time Registers to the PTP clock domain The application must not update the Target Time Registers when this bit is read as 1. Otherwise, the synchronization of the previous programmed time gets corrupted.	0H	R_WS_SC
30:0	TTSL1	Target Time Low for PPS1 Register This register stores the time in (signed) nanoseconds. When the value of the timestamp matches the both Target Timestamp registers, then based on the TRGTMODSEL1 field (Bits [14:13]) in Register 459 (PPS Control Register), the MAC starts or stops the PPS signal output and generates an interrupt (if enabled). This value should not exceed 0x3B9A_C9FF when Bit 9 (TSCTRLSSR) is set in Register 448 (Timestamp Control Register). The actual start or stop time of the PPS signal output may have an error margin up to one unit of sub-second increment value.	0H	R_W

10 Timers

10.1 Timers Overview

This component is an AMBA 2.0-compliant Advanced Peripheral Bus (APB) slave device.

10.2 OS Timer

10.3 CPU Watchdog Timer

10.4 General-Purpose Timer

~~10.4.1 Features~~

~~Timer has the following features:~~

- ~~Up to eight programmable timers~~
- ~~Configurable timer width: 8 to 32 bits~~
- ~~Support for two operation modes: free-running and user-defined count~~
- ~~Support for independent clocking of timers~~

- ~~Configurable polarity for each individual interrupt~~
- ~~Configurable option for a single or combined interrupt output flag~~
- ~~Configurable option to have read/write coherency registers for each timer~~
- ~~Configurable option to include timer toggle output, which toggles whenever timer counter reloads~~
- ~~Configurable option to enable programmable pulse-width modulation of timer toggle outputs~~

10.4.2 Function Description

10.4.2.1 Timer Operation

The Timers component implements up to four identical but separately-programmable timers.

Timers count down from a programmed value and generate an interrupt when the count reaches zero. ~~You can use the TIM_INTR_IO parameter (Single Combined Interrupt) to create a single combined interrupt, which is active whenever any of the individual timer interrupts is active.~~

The initial value for each timer – that is, the value from which it counts down – is loaded into the timer using the appropriate load count register (TimerNLoadCount). Two events can cause a timer to load the initial count from its Timer*N*LoadCount register:

- Timer is enabled after being reset or disabled
- Timer counts down to 0

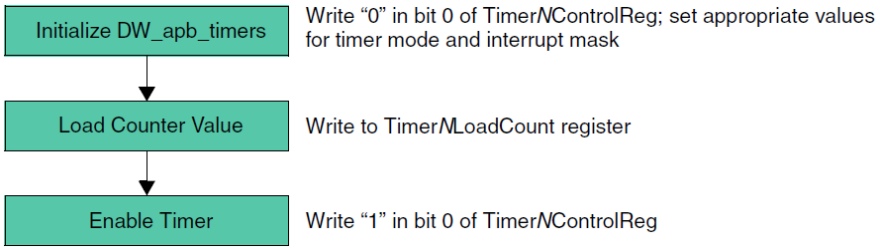
All interrupt status registers and end-of-interrupt registers can be accessed at any time.

10.4.2.2 Timers Usage Flow

The procedure is a basic flow to follow when programming the Timers. More advanced functions are discussed later in this chapter.

1. Initialize the timer through the TimerNControlReg register (where *N* is in the range 1 to 4):
   a. Disable the timer by writing a “0” to the timer enable bit (bit 0); accordingly, the timer_en output signal is de-asserted.
   b. Program the timer mode—user-defined or free-running—by writing a “0” or “1,” respectively, to the timer mode bit (bit 1).
   c. Set the interrupt mask as either masked or not masked by writing a “1” or “0,” respectively, to the timer interrupt mask bit (bit 2).
2. Load the timer counter value into the TimerNLoadCount register (where *N* is in the range 1 to 4).
3. Enable the timer by writing a “1” to bit 0 of TimerNControlReg.

Timer Usage flow:



10.4.2.3 Choosing the Number of Timers

There are several registers with names specific to the number of timers that you choose (where *N* is from 1 to 4):

- TimerNLoadCount – Timer*N* load count register
- ~~TimerNLoadCount2 (optional) – Timer*N* load count register for programming width of HIGH period of timer_*N*_toggle output~~
- TimerNCurrentValue – Timer*N* current value register
- TimerNControlReg – Timer*N* control register
- TimerNEOI – Timer*N* end-of-interrupt register
- TimerNIntStatus – Timer*N* interrupt status register

10.4.2.3.1 Enabling and Disabling a Timer

You use bit 0 of the TimerNControlReg, where *N* is in the range 1 to 4, to either enable or disable a timer.

~~10.4.2.3.2 Configuring the Width of a Timer~~

You configure the width of a timer through the TIMER_WIDTH_*N* parameter; each timer can be from 8 bits to 32 bits. You do this for each timer through the Timer *N* Configuration section of the Specify Configuration activity in coreConsultant.

10.4.2.3.3 Loading a Timer Countdown Value

When a timer counter is enabled after being reset or disabled, the count value is loaded from the TimerNLoadCount register; this occurs in both free-running and user-defined count modes. When a timer counts down to 0, it loads one of two values, depending on the timer operating mode:

- User-defined count mode – Timer loads the current value of the TimerNLoadCount register. Use this mode if you want a fixed, timed interrupt. Designate this mode by writing a "1" to bit 1 of TimerNControlReg.
- Free-running mode – Timer loads the maximum value, which is dependent on the timer width; that is, the Timer*N*LoadCount register is comprised of 2^{32}~~TIMER_WIDTH_*N*~~ – 1 bits, all of which are loaded with 1s. The timer counter wrapping to its maximum value allows time to reprogram or disable the timer before another interrupt occurs. Use this mode if you want a single timed interrupt. Designate this mode by writing a "0" to bit 1 of TimerNControlReg.

10.4.2.3.4 Working with Interrupts

The TimerNIntStatus and TimerNEOI registers handle interrupts in order to ensure safe operation of the interrupt clearing. Because of the hclk/pclk ratio, if pclk can perform a write to clear an interrupt, it could continue with another transfer on the bus without knowing whether the write has occurred. Therefore, it is much safer to clear the interrupt by a read operation.

Clearing Interrupts

Provided the timer is enabled, its interrupt remains asserted until it is cleared by reading one of two end-of-interrupt registers (TimerNEOI or TimersEOI, the individual and global end-of-interrupt registers, respectively). When the timer is disabled, the timer interrupt is cleared. You can clear an individual timer interrupt by reading its TimerNEOI register. You can clear all active timer interrupts at once by reading the global TimersEOI register or by disabling the timer.

Checking Interrupt Status

You can query the interrupt status of an individual timer without clearing its interrupt by reading the TimerNIntStatus register. You can query the interrupt status of all timers without clearing the interrupts by reading the global TimersIntStatus register.

Masking Interrupts

Each individual timer interrupt can be masked using its TimerNControlReg register. To mask an interrupt, you write a "1" to bit 2 of Timer*N*ControlReg.

Setting Interrupt Polarity

The polarity of the generated timer interrupts can be configured to be either active-high or active-low using the TIM_INTRPT_PLRITY parameter (Interrupt Polarity).

~~10.4.2.3.5 Generating Toggled Outputs~~

~~You can configure a timer through the TIMER_HAS_TOGGLE_*N* parameter in order to generate an output that toggles whenever the timer counter reaches 0.~~

~~10.4.2.3.6 Timer Pause Mode~~

~~The operation of a timer can be paused by asserting the respective timer_*N*_pause input signal, which is synchronized to the timer_*N*_clk domain.~~

10.4.3 Timers Registers

10.4.3.1 Register Memory Map

The following table lists the address ranges of the registers for each timer; they are aligned to 32-bit boundaries.

Table Memory Map of Timer 1 Registers

Address Range (Base +)	Function
0x00 to 0x10	Timer 1 Registers

Address Range (Base +)	Function
0x14 to 0x24	Timer 2 Registers
0x28 to 0x38	Timer 3 Registers
0x3c to 0x4c	Timer 4 Registers

Table lists registers associated with Timer 1; use this table as an example for timers 2-4.

Table Memory Map of Timer 1 Registers

Name	Address Offset	Width	R/W	Description
Timer1LoadCount	0x00, 0x01, 0x02, 0x03	See Description	R/W	Value to be loaded into Timer1 **Width:** ~~*TIMER_WIDTH_1 -* 1~~ 32 bits **Range:** 0 to [2^32~~*TIMER_WIDTH_1* –~~ 1] **Default value:** 0
~~Timer1LoadCount2~~	0xb0 0xb4 0xb8 0xbC 0xc0 0xc4 0xc8 0xcc	See Description	R/W	Value to be loaded into Timer1 when toggle output changes from 0 to 1 **Width:** ~~*TIMER_WIDTH_1 –* 1~~32 bits **Range:** 0 to [2^32~~*TIMER_WIDTH_1* -~~ 1] **Default value:** ~~*TIMER_WIDTH_132*~~'b0 ~~**Dependency:** Present only when TIM_NEWMODE = 0~~
Timer1CurrentValue	0x04, 0x05, 0x06, 0x07	See Description	R	Current Value of Timer1 **Width:** ~~*TIMER_WIDTH_1 –* 1~~32 bits **Range:** 0 to [2^32~~*TIMER_WIDTH_1* –~~ 1] **Default value:** 0
Timer1ControlReg	0x08	4 bits	R/W	Control Register for Timer1 **Default value:** 3'b0
Timer1EOI	0x0C	1 bit	R	Clears the interrupt from Timer1 **Default value:** 1'b0
Timer1IntStatus	0x10	1 bit	R	Contains the interrupt status for Timer1 **Default value:** 1'b0

Timers also contain the following three system registers, listed in Table.

Table Timers System Registers

Name	Address Offset	Width	R/W	Description
TimersIntStatus	0xa0	See Description	R	Contains the interrupt status of all timers in the component. **Width:** ~~*NUM_TIMERS* -~~13:0 **Default value:** ~~*NUM_TIMERS4*~~'b0
TimersEOI	0xa4	See Description	R	Returns all zeroes (0) and clears all active interrupts. **Width:** ~~*NUM_TIMERS* -~~13:0 **Default value:** ~~*NUM_TIMERS4*~~'b0
TimersRawIntStatus	0xa8	See Description	R	Contains the unmasked interrupt status of all timers in the component. **Width:** ~~*NUM_TIMERS* -~~13:0 **Default value:** ~~*NUM_TIMERS4*~~'b0
~~TIMERS_COMP_ VERSION~~	~~0xac~~	~~32 bits~~	~~R~~	~~Current revision number of the Timers component.~~

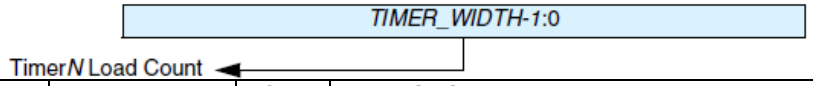
10.4.3.2 Register and Field Descriptions

The following sections contain the memory diagrams and field descriptions for the individual registers.

10.4.3.2.1 Timer*N*LoadCount

- **Name:** Timer*N* Load Count Register
- **Size:** ~~*TIMER_WIDTH_N-1, where N is 1...4*~~*32 bits*

- **Address Offset:**
  for *N* = 1, 0x00
  for *N* = 2, 0x14
  for *N* = 3, 0x28
  for *N* = 4, 0x3C
- **Read/write access:** read/write

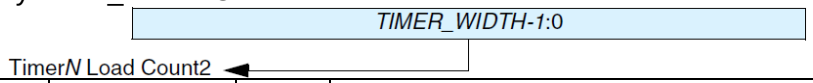


Bits	Name	R/W	Description
~~*TIMER_WIDTH_N-1*~~*31*:0, ~~where N is 1...4~~	Timer*N* Load Count Register	R/W	Value to be loaded into Timer*N*. This is the value from which counting commences. Any value written to this register is loaded into the associated timer.

~~10.4.3.2.2 Timer*N*LoadCount2~~

- **Name:** Timer*N* Load2 Count Register
- **Size:** ~~*TIMER_WIDTH_N-1, where N is 1...*~~*32 bits*
- **Address Offset:**
  for N = 1, 0xb0
  for N = 2, 0xb4
  for N = 3, 0xb8
  for N = 4, 0xbC
- **Read/write access:** read/writes

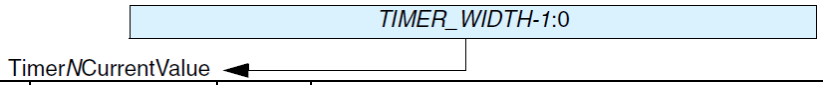
This register exists only if TIM_NEWMODE = 1



Bits	Name	R/W	Description
~~*TIMER_WIDTH_N-1*~~*31*:0, ~~where N is 1...4~~	Timer*N* Load Count2 Register	R/W	Value to be loaded into Timer*N* when timer_*N*_toggle output changes from 0 to 1. This value determines the width of the HIGH period of the timer_*N*_toggle output.

10.4.3.2.3 Timer*N*CurrentValue

- **Name:** Timer*N* Current Value Register
- **Size:** ~~*TIMER_WIDTH_N-1, where N is 1...*~~*32 bits*
- **Address Offset:**
  for *N* = 1, 0x04
  for *N* = 2, 0x18
  for *N* = 3, 0x2C
  for *N* = 4, 0x40
- **Read/write access:** read

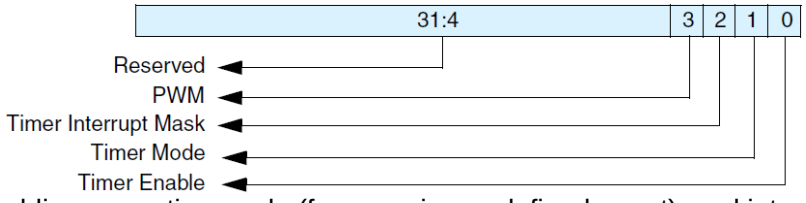


Bits	Name	R/W	Description
~~*TIMER_WIDTH_N-1*~~*31*:0~~, where N is 1...4~~	Timer *N* Current Value	R	Current Value of Timer*N*. ~~This register is supported only when timer_N_clk is synchronous to pclk.~~ Reading this register when using independent clocks results in an undefined value. This register is supported for N=1, 2.

10.4.3.2.4 Timer*N*ControlReg

- **Name:** Timer*N* Control Register
- **Size:** 4 bits
- **Address Offset:**
  for *N* = 1, 0x08
  for *N* = 2, 0x1C
  for *N* = 3, 0x30
  for *N* = 4, 0x44

- **Read/write access:** read/write

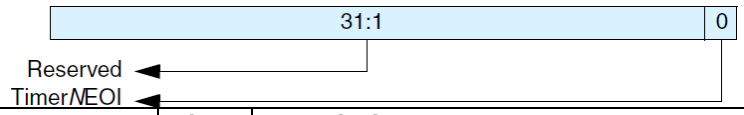


This register controls enabling, operating mode (free-running or defined-count), and interrupt mask of Timer*N*. You can program each Timer*N*ControlReg to enable or disable a specific timer and to control its mode of operation.

Bits	Name	R/W	Description
31:4	Reserved, read as zero		
~~3~~	~~TIMER_PWM~~	~~R/W~~	~~Pulse Width Modulation of timer_*N*_toggle output.~~ ~~0 – disabled~~ ~~1 – enabled~~ ~~**Dependency:** This field present only if TIM_NEWMODE=1~~
2	Timer Interrupt Mask	R/W	Timer interrupt mask for Timer*N* 0 – not masked 1 – masked
1	Timer Mode	R/W	Timer mode for Timer*N* 0 – free-running mode 1 – user-defined count mode
0	Timer Enable	R/W	Timer enable bit for Timer*N* 0 – disable 1 – enable

10.4.3.2.5 Timer*N*EOI

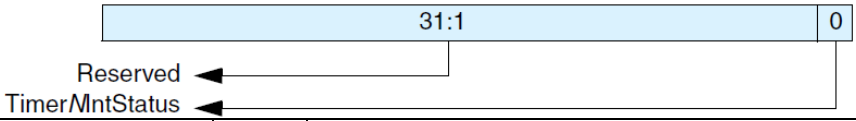
- **Name:** Timer*N* End-of-Interrupt Register
- **Size:** 1 bit
- **Address Offset:**
  for *N* = 1, 0x0C
  for *N* = 2, 0x20
  for *N* = 3, 0x34
  for *N* = 4, 0x48
- **Read/write access:** read



Bits	Name	R/W	Description
31:1	Reserved, read as zero		
0	Timer*N* End-of-Interrupt Register	R	Reading from this register returns all zeroes (0) and clears the interrupt from Timer*N*.

10.4.3.2.6 Timer*N*IntStatus

- **Name:** Timer*N* Interrupt Status Register
- **Size:** 1 bit
- **Address Offset:**
  for *N* = 1, 0x10
  for *N* = 2, 0x24
  for *N* = 3, 0x38
  for *N* = 4, 0x4C
- **Read/write access:** read

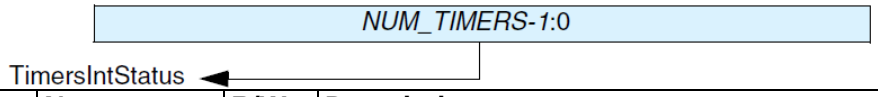


Bits	Name	R/W	Description
31:1	Reserved, read as zero		

Bits	Name	R/W	Description
0	Timer*N* Interrupt Status Register	R	Contains the interrupt status for Timer*N*.

10.4.3.2.7 TimersIntStatus

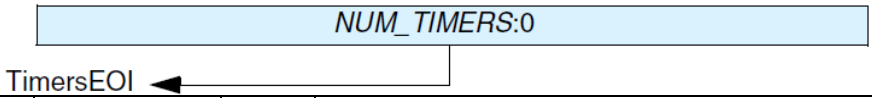
- **Name:** Timers Interrupt Status Register
- **Size:** ~~*NUM_TIMERS*~~ *4 bits*
- **Address Offset:** 0xa0
- **Read/write access:** read



Bits	Name	R/W	Description
~~*NUM_TIMERS-1*~~3:0	Timers Interrupt Status Register	R	Contains the interrupt status of all timers in the component. If a bit of this register is 0, then the corresponding timer interrupt is not active ~~– and the corresponding interrupt could be on either the timer_intr bus or the timer_intr_n bus, depending on the interrupt polarity you have chosen.~~ Similarly, if a bit of this register is 1, then the corresponding interrupt bit has been set in the relevant interrupt bus. In both cases, the status reported is the status after the interrupt mask has been applied. Reading from this register does not clear any active interrupts: 0 – ~~either~~ timer_intr ~~or timer_intr_n~~ is not active after masking 1 – ~~either~~ timer_intr ~~or timer_intr_n~~ is active after masking

10.4.3.2.8 TimersEOI

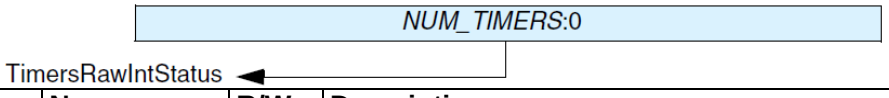
- **Name:** Timers End-of-Interrupt Register
- **Size:** ~~*NUM_TIMERS*~~ *4 bits*
- **Address Offset:** 0xa4
- **Read/write access:** read



Bits	Name	R/W	Description
~~*NUM_TIMERS-1*~~3:0	Timers End-of-Interrupt Register	R	Reading this register returns all zeroes (0) and clears all active interrupts.

10.4.3.2.9 TimersRawIntStatus

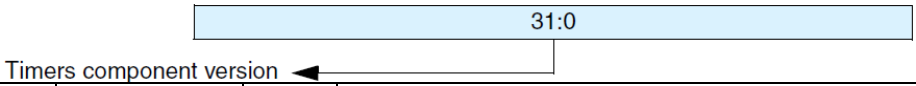
- **Name:** Timers Raw Interrupt Status Register
- **Size:** ~~*NUM_TIMERS*~~ *4 bits*
- **Address Offset:** 0xa8
- **Read/write access:** read



Bits	Name	R/W	Description
~~*NUM_TIMERS-1*~~3:0	Timers Raw Interrupt Status Register	R	The register contains the unmasked interrupt status of all timers in the component. 0 – ~~either~~ timer_intr ~~or timer_intr_n~~ is not active prior to masking 1 – ~~either~~ timer_intr ~~or timer_intr_n~~ is active prior to masking

~~10.4.3.2.10TIMERS_COMP_VERSION~~

- **Name:** Timers Component Version
- **Size:** 32 bits
- **Address Offset:** 0xac
- **Read/write access:** read



Bits	Name	R/W	Description
31:0	Timers Component Version	R	Current revision number of the Timers component. Reset Value: For the value, see the releases table in the AMBA 2 release notes

10.5 Watchdog Timer

10.5.1 Features

The WDT supports the following features:

- ~~Configurable APB data bus widths of 8, 16, and~~ 32 bits~~.~~
- ~~Configurable watchdog counter width of 16 to~~ 32 bits~~.~~
- Counter counts down from a preset value to 0 to indicate the occurrence of a timeout.
- ~~Optional external clock enable signal to control the rate at which the counter counts.~~
- If a timeout occurs the WDT can perform one of the following operations:
  - ❑ Generate a system reset
  - ❑ First generate an interrupt and if this is not cleared by the service routine by the time a second timeout occurs then generate a system reset
- Programmable timeout range (period). The option of hard coding this value during configuration is available to reduce the register requirements.
- ~~Optional dual programmable timeout period, used when the duration waited for the first kick is different than that required for subsequent kicks. The option of hard coding these values is available.~~
- Programmable and hard coded reset pulse length.
- Prevention of accidental restart of the WDT counter.
- Prevention of accidental disabling of the WDT.
- ~~Optional support for Pause mode with the use of external pause enable signal.~~
- ~~Optional support for asynchronous external timer clock. With this feature enabled, the timer interrupt and system reset can be generated, even when the APB bus clock is switched off.~~

10.5.2 Function Description

10.5.2.1 Counter

The WDT counts from a preset (timeout) value in descending order to zero. When the counter reaches zero, depending on the output response mode selected, either a system reset or an interrupt occurs. When the counter reaches zero, it wraps to the selected timeout value and continues decrementing. The user can restart the counter to its initial value. This is programmed by writing to the restart register 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).

10.5.2.2 Interrupts

The WDT can be programmed to generate an interrupt (and then a system reset) when a timeout occurs. When a 1 is written to the response mode field (RMOD, bit 1) of the Watchdog Timer Control Register (WDT_CR), 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 zero, an interrupt is not generated.

10.5.2.3 System Resets

When a 0 is written to the output response mode field (RMOD, bit 1) of the Watchdog Timer Control Register (WDT_CR), the WDT generates a system reset when a timeout occurs. The WDT can be configured so that it is always enabled upon reset of the WDT. If this is the case, it overrides whatever has been written in bit 0 of the WDT_CR register (the WDT enable field).

If a restart occurs at the same time the watchdog counter reaches zero, a system reset is not generated.

~~10.5.2.4 Pause Mode~~

~~The WDT can be configured to include a pause signal on the interface, which "freezes" the watchdog counter while the system is being paused. During pause mode, if the counter is frozen at the zero count, no interrupt or system reset is generated. When the pause is removed, the interrupt or system reset is asserted on the next rising edge of the clock. If wdt_clk_en is present, the interrupt or reset is generated only when the wdt_clk_en is asserted. If the counter is not zero when the pause is removed, the interrupt or system reset is not generated.~~

~~10.5.2.5 External Clock Enable~~

~~The WDT can be configured to include an external clock enable (wdt_clk_en) that controls the rate at which the counter decrements. When the counter reaches zero, the wdt_clk_en must be asserted for the interrupt or system reset to be generated.~~

~~If a restart occurs when the wdt_clk_en is low, the restart is internally extended until the next rising edge of the wdt_clk_en so that it may be seen and the counter can be restarted. Clearing of interrupts is independent to the clock enable, but both the interrupt and reset are generated only when the clock enable is high.~~

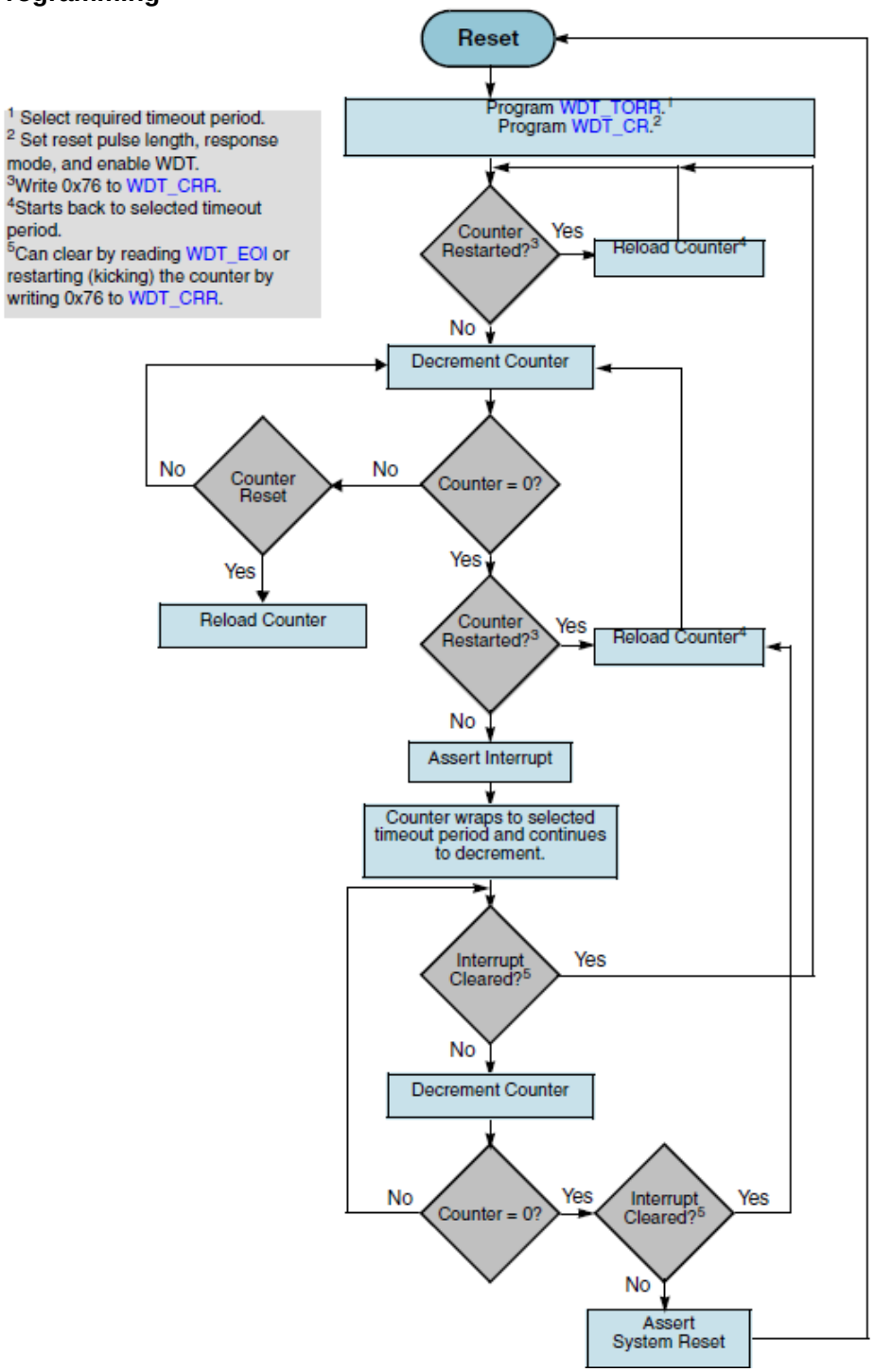
10.5.2.6 Reset Pulse Length

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 plus two cycles of synchronization delay, or until the system is reset. A counter restart has no effect on the system reset once it has been asserted.

10.5.2.7 Timeout Period Values

The WDT can be configured to have a fixed or user-defined timeout period range. In both cases, the values that may be selected are limited by the WDT counter width. If the timeout period range that is selected is greater than the counter width, the timeout period is truncated to fit to the counter width. In the case of user-defined timeout period ranges, the value is limited at the time of configuration, and values greater than the count are not accepted.

10.5.3 Function Programming



10.5.4 WDT Registers

10.5.4.1 Register Memory Map

Table shows the memory map for the WDT peripheral.

Table Memory Map of WDT

Name	Address Offset	Width	Description

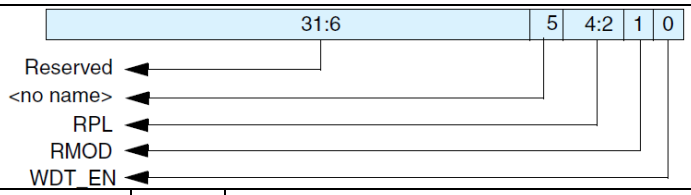
Name	Address Offset	Width	Description
WDT_CR	0x00	5 bits	Control register **Reset Value:** Register fields WDT_DFLT_RPL (3 bits), WDT_DFLT_RMOD (1 bit), and WDT_ALWAYS_EN (1 bit)
WDT_TORR	0x04	0x04 See Description	Timeout range register **Width:** WDT_TORR_WIDTH (an internal parameter that is either 4- or 8-bits wide, depending on the configuration parameter WDT_DUAL_TOP) **Reset Value:** WDT_TOP_RST
WDT_CCVR	0x08	0x08 See Description	Current counter value register **Width:** WDT_CNT_WIDTH **Reset Value:** WDT_CNT_RST
WDT_CRR	0x0c	8 bits	Counter restart register **Reset Value:** 0x0
WDT_STAT	0x10	1 bit	Interrupt status register **Reset Value:** 0x0
WDT_EOI	0x14	1 bit	Interrupt clear register **Reset Value:** 0x0
Reserved	0x18-0xe0		
WDT_COMP_PARAMS_5	0xe4	32 bits	Component Parameters Register 5 **Reset Value:** Reset value depends on user configuration, refer to relevant tables for more information.
WDT_COMP_PARAMS_4	0xe8	32 bits	Component Parameters Register 4 **Reset Value:** Reset value depends on user configuration, refer to relevant tables for more information.
WDT_COMP_PARAMS_3	0xec	32 bits	Component Parameters Register 3 **Reset Value:** Reset value depends on user configuration, refer to relevant tables for more information.
WDT_COMP_PARAMS_2	0xf0	32 bits	Component Parameters Register 2 **Reset Value:** Reset value depends on user configuration, refer to relevant tables for more information.
WDT_COMP_PARAMS_1	0xf4	32 bits	Component Parameters Register 1 **Reset Value:** Reset value depends on user configuration, refer to relevant tables for more information.
WDT_COMP_VERSION	0xf8	32 bits	DesignWare Component Version register **Reset Value:** ‘PRIORITY_1
WDT_COMP_TYPE	0xfc	32 bits	DesignWare Component Type register **Reset Value:** ‘PRIORITY_1

10.5.4.2 Register and Field Descriptions

The following sections contain the memory diagrams and field descriptions for the individual registers.

10.5.4.2.1 WDT_CR

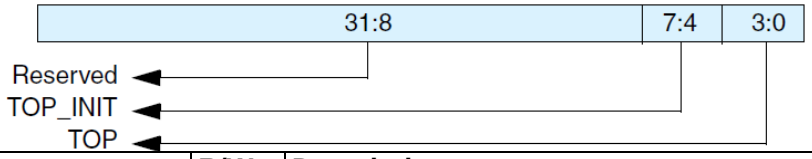
- **Name:** Control Register
- **Size:** 5 bits
- **Address Offset:** 0x00
- **Read/write access:** read/write



Bits	Name	R/W	Description
31:6	Reserved, read as zero		
5	<no name>	R/W	Redundant R/W bit. Included for ping test purposes, as it is the only R/W register bit that is in every configuration of the WDT.
4:2	RPL	R/W	R/W 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. 000 – 2 pclk cycles 001 – 4 pclk cycles 010 – 8 pclk cycles 011 – 16 pclk cycles 100 – 32 pclk cycles 101 – 64 pclk cycles 110 – 128 pclk cycles 111 – 256 pclk cycles **Reset Value:** *~~WDT_DFLT_RPL~~0x0*
1	RMOD	R/W	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. 0 = Generate a system reset. 1 = First generate an interrupt and if it is not cleared by the time a second timeout occurs then generate a system reset. **Reset Value:** ~~WDT_DFLT_RMOD~~0x0
0	WDT_EN	R/W	WDT enable. Writable ~~when the configuration parameter WDT_ALWAYS_EN = 0, otherwise~~, it is readable. This bit is used to enable and disable the 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. 0 = WDT disabled. 1 = WDT enabled. **Reset Value:** *~~WDT_ALWAYS_EN~~0x0*

10.5.4.2.2 WDT_TORR

- **Name:** Timeout Range Register
- **Size:** *~~WDT_TORR_WIDTH~~ 4 bits*
- **Address Offset:** 0x04
- **Read/write access:** read/write



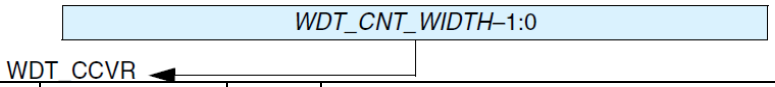
Bits	Name	R/W	Description
31:8	Reserved, read as zero		

Bits	Name	R/W	Description
7:4	~~TOP_INIT~~	~~R/W~~	~~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)~~ ~~**Reset Value:** Configuration parameter WDT_DFLT_TOP_INIT~~ ~~**NOTE:** These bits exist only when the configuration parameter WDT_DUAL_TOP =1, otherwise, they are fixed at zero.~~
3:0	TOP	R/W	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 32 bits ~~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)~~ **Reset Value:** ~~*WDT_DFLT_TOP*~~*0x0*

10.5.4.2.3 WDT_CCVR

- **Name:** Current Counter Value Register
- **Size:** ~~*WDT_CNT_WIDTH*~~ *32 bits*

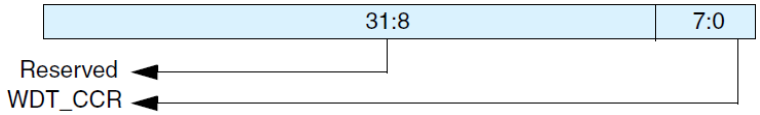
- **Address Offset:** 0x08
- **Read/write access:** read



Bits	Name	R/W	Description
~~*WDT_CNT_WIDTH*~~ 31:0	Current Counter Value Register	R	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. **Reset Value:** *WDT_CNT_RST*

10.5.4.2.4 WDT_CRR

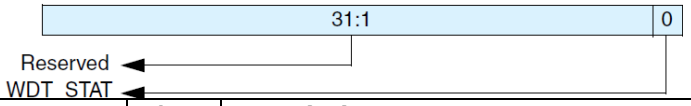
- **Name:** Counter Restart Register
- **Size:** 8 bits
- **Address Offset:** 0x0c
- **Read/write access:** write



Bits	Name	R/W	Description
31:8	Reserved, read as zero		
7:0	Counter Restart Register	W	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. **Reset Value:** 0

10.5.4.2.5 WDT_STAT

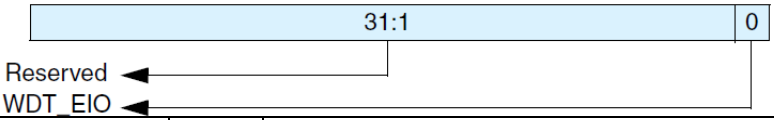
- **Name:** Interrupt Status Register
- **Size:** 1 bit
- **Address Offset:** 0x10
- **Read/write access:** read



Bits	Name	R/W	Description
31:1	Reserved, read as zero		
0	Interrupt Status Register	R	This register shows the interrupt status of the WDT. 1 = Interrupt is active regardless of polarity. 0 = Interrupt is inactive. **Reset Value:** 0

10.5.4.2.6 WDT_EOI

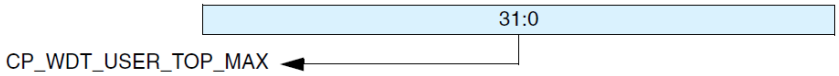
- **Name:** Interrupt Clear Register
- **Size:** 1 bit
- **Address Offset:** 0x14
- **Read/write access:** read



Bits	Name	R/W	Description
31:1	Reserved, read as zero		
0	Interrupt Clear Register	R	Clears the watchdog interrupt. This can be used to clear the interrupt without restarting the watchdog counter. **Reset Value:** 0

10.5.4.2.7 WDT_COMP_PARAMS_5

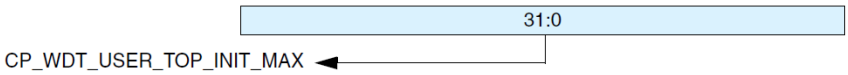
- **Name:** Component Parameters Register 5
- **Size:** 32 bits
- **Address Offset:** 0xe4
- **Read/write access:** read



Bits	Name	R/W	Description
31:0	CP_WDT_USER_TOP_MAX	R	Upper limit of Timeout Period parameters. The value of this register is derived from the WDT_USER_TOP_* coreConsultant parameters.

10.5.4.2.8 WDT_COMP_PARAMS_4

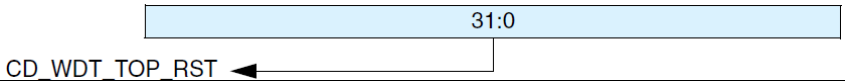
- **Name:** Component Parameters Register 4
- **Size:** 32 bits
- **Address Offset:** 0xe8
- **Read/write access:** read



Bits	Name	R/W	Description
31:0	CP_WDT_USER_TOP_INIT_MAX	R	Upper limit of Initial Timeout Period parameters. The value of this register is derived from the WDT_USER_TOP_INIT_* coreConsultant parameters.

10.5.4.2.9 WDT_COMP_PARAMS_3

- **Name:** Component Parameters Register 3
- **Size:** 32 bits
- **Address Offset:** 0xec
- **Read/write access:** read



Bits	Name	R/W	Description
31:0	CD_WDT_TOP_RST	R	The value of this register is derived from the WDT_TOP_RST coreConsultant parameter.

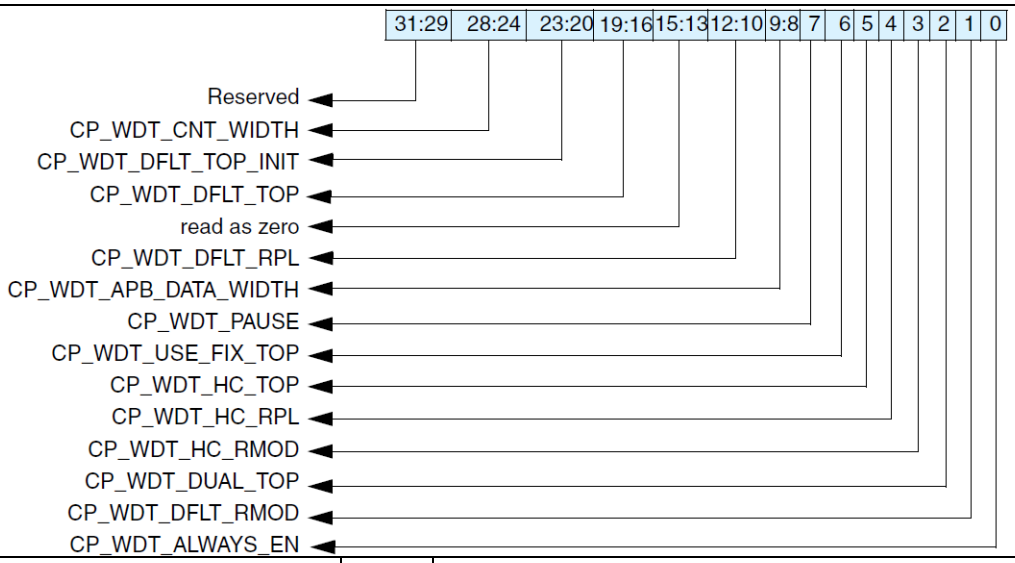
10.5.4.2.10 WDT_COMP_PARAMS_2

- **Name:** Component Parameters Register 2
- **Size:** 32 bits
- **Address Offset:** 0xf0
- **Read/write access:** read

Bits	Name	R/W	Description
31:0	CP_WDT_CNT_RST	R	The value of this register is derived from the WDT_RST_CNT coreConsultant parameter.

10.5.4.2.11 WDT_COMP_PARAMS_1

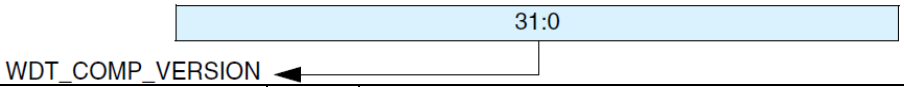
- **Name:** Component Parameters Register 1
- **Size:** 32 bits
- **Address Offset:** 0xf4
- **Read/write access:** read



Bits	Name	R/W	Description
31:29	Reserved and read as zero		
28:24	CP_WDT_CNT_WIDTH	R	~~WDT_CNT_WIDTH = 16~~ 16
23:20	CP_WDT_DFLT_TOP_INIT	R	~~WDT_DFLT_TOP_INIT~~ 0
19:16	CP_WDT_DFLT_TOP	R	~~WDT_DFLT_TOP~~ 0
15:13	Reserved and read as zero		
12:10	CP_WDT_DFLT_RPL	R	~~WDT_DFLT_RPL~~ 0
9:8	CP_WDT_APB_DATA_WIDTH	R	~~(APB_DATA_WIDTH == 8) = 0~~ ~~(APB_DATA_WIDTH == 16) = 1~~ (APB_DATA_WIDTH == 32) = 2 ~~reserved = 3~~
7	CP_WDT_PAUSE	R	**Reset Value:** 0
6	CP_WDT_USE_FIX_TOP	R	~~(WDT_USE_FIX_TOP == FALSE) =0~~ ~~(WDT_USE_FIX_TOP == TRUE) =1~~
5	CP_WDT_HC_TOP	R	~~(WDT_HC_TOP == FALSE) =0~~ ~~(WDT_HC_TOP == TRUE) =1~~
4	CP_WDT_HC_RPL	R	~~(WDT_HC_RPL == FALSE) =0~~ ~~(WDT_HC_RPL == TRUE) =1~~
3	CP_WDT_HC_RMOD	R	~~(WDT_HC_RMOD == FALSE) =0~~ ~~(WDT_HC_RMOD == TRUE) =1~~
2	CP_WDT_DUAL_TOP	R	~~(WDT_DUAL_TOP == FALSE) =0~~ ~~(WDT_DUAL_TOP == TRUE) =1~~
1	CP_WDT_DFLT_RMOD	R	~~(WDT_DFLT_RMOD == FALSE) =0~~ ~~(WDT_DFLT_RMOD == TRUE) =1~~
0	CP_WDT_ALWAYS_EN	R	~~(WDT_ALWAYS_EN == FALSE) =0~~ ~~(WDT_ALWAYS_EN == TRUE) =1~~

~~10.5.4.2.12WDT_COMP_VERSION~~

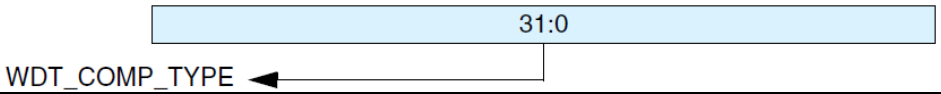
- **Name:** Component Version Register
- **Size:** 32 bits
- **Address Offset:** 0xf8
- **Read/write access:** read



Bits	Name	R/W	Description
31:0	WDT Component Version	R	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 AMBA 2 release notes

~~10.5.4.2.13WDT_COMP_TYPE~~

- **Name:** Component Type Register
- **Size:** 32 bits
- **Address Offset:** 0xfc
- **Read/write access:** read



Bits	Name	R/W	Description
31:0	Component Type Register	R	Designware Component Type number = 0x44_57_01_20. This assigned unique hex value is constant, and is derived from the two ASCII letters “DW” followed by a 16-bit unsigned number.

10.6 Frame Counter (FC)

11 UART

The UART is a programmable Universal Asynchronous Receiver/Transmitter (UART). The UART is modeled after the industry-standard 16550. However, the register address space is relocated to 32-bit data boundaries for APB bus implementation.

11.1.1 UART Features

- Configurable parameters for the following:
  - APB data bus widths of 8, 16 and 32
  - Additional DMA interface signals for compatibility with DesignWare DMA interface
  - DMA interface signal polarity
  - Transmit and receive FIFO depths of 0, 16, 32, 64, 128, 256, 512, 1024, 2048
  - Internal or external FIFO (RAM) selection
  - Use of two clocks—pclk and sclk—instead of just pclk
  - IrDA 1.0 SIR mode support with up to 115.2 Kbaud data rate and a pulse duration (width) as specified in the IrDA physical layer specification:
    width = 3/16 *bit period*
  - IrDA 1.0 SIR low-power reception capabilities
  - Baud clock reference output signal
  - Clock gate enable output(s) used to indicate that the TX and RX pipeline is clear (no data) and no activity has occurred for more than one character time, so that clocks can be gated
  - FIFO access mode—for FIFO testing—enabling the master to write to the receive FIFO and read from the transmit FIFO
  - Additional FIFO status registers
  - Shadow registers to reduce software overhead and also include a software programmable reset
  - Auto Flow Control mode, as specified in the 16750 standard
  - Loopback mode that enables greater testing of Modem Control and Auto Flow Control features (Loopback support in IrDA SIR mode is available)
  - Transmitter Holding Register Empty (THRE) interrupt mode
  - Busy functionality
- Ability to set some configuration parameters during instantiation
- Configuration identification registers present
- Functionality based on the 16550 industry standard
  - Programmable character properties, such as:
    - Number of data bits per character (5-8)
    - Optional parity bit (with odd, even select or Stick Parity)
    - Number of stop bits (1, 1.5 or 2)
  - Line break generation and detection
  - DMA signaling with two programmable modes
  - Prioritized interrupt identification
- Programmable FIFO enable/disable
- Programmable serial data baud rate as calculated by the following:
  baud rate = (*serial clock frequency*)/(16 *divisor*)
- External read enable signal for RAM wake-up when using external RAMs
- Modem and status lines are independently controlled

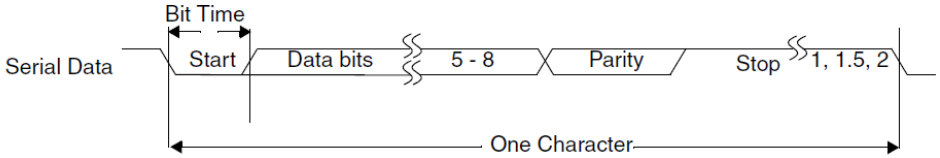
11.2 UART Function Description

This chapter describes the functional operation of UART.

11.2.1 UART (RS232) Serial Protocol

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

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 received data. The UART Line Control Register is used to control the serial character characteristics. The individual bits of the data word are sent after the start bit, starting with the leastsignificant bit (LSB). These are followed by the optional parity bit, followed by the stop bit(s), which can be 1, 1.5, or 2.

All the bits in the transmission are transmitted for exactly the same time duration; the exception to this is the half-stop bit when 1.5 stop bits are used. This duration is referred to as a Bit Period or Bit Time; one Bit Time equals sixteen baud clocks.

To ensure stability on the line, the receiver samples the serial input data at approximately the midpoint of the Bit Time once the start bit has been detected. Because the exact number of baud clocks is known for which each bit was transmitted, calculating the midpoint for sampling is not difficult; that is, every sixteen baud clocks after the midpoint sample of the start bit.

Together with serial input debouncing, this sampling helps to avoid the detection of false start bits. Short glitches are filtered out by debouncing, and no transition is detected on the line. If a glitch is wide enough to avoid filtering by debouncing, a falling edge is detected. However, a start bit is detected only if the line is again sampled low after half a bit time has elapsed.

~~11.2.2 IrDA 1.0 SIR Protocol~~

The Infrared Data Association (IrDA) 1.0 Serial Infrared (SIR) mode supports bi-directional data communications with remote devices using infrared radiation as the transmission medium. IrDA 1.0 SIR mode specifies a maximum baud rate of 115.2 Kbaud.

The data format is similar to the standard serial—sout and sin—data format. Each data character is sent serially in this order:

1. Begins with a start bit
2. Followed by 8 data bits
3. Ends with at least one stop bit

Thus, the number of data bits that can be sent is fixed. No parity information can be supplied, and only one stop bit is used in this mode. Trying to adjust the number of data bits sent or enable parity with the Line Control Register (LCR) has no effect.

Configuration of the UART for IrDA 1.0 SIR does the following:

- Bit 6 of the Mode Control Register (MCR) enables or disables the IrDA 1.0 SIR mode.
- Disabling IrDA SIR mode causes the logic to not be implemented; the mode cannot be activated, which reduces total gate counts.
- When IrDA SIR mode is enabled and active, serial data is transmitted and received on the sir_out_n and sir_in ports, respectively.

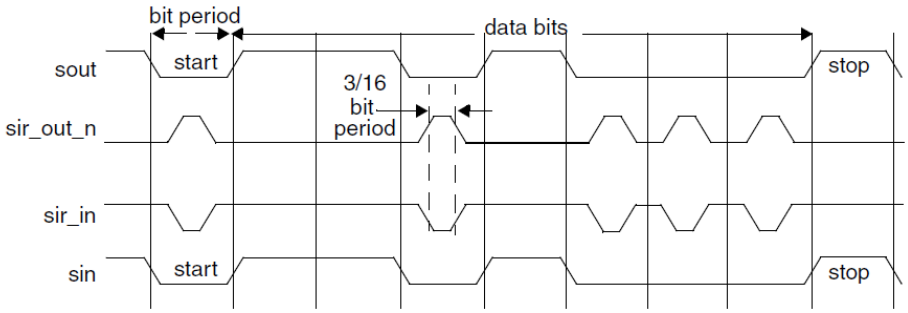
Transmission or non-transmission of a single infrared pulse indicates the following:

- Transmitting a single infrared pulse indicates logic 0
- Non-transmission of a pulse indicates logic 1

The width of each pulse is 3/16ths of a normal serial bit time. Thus, each new character begins with an infrared pulse for the start bit. However, received data is inverted from transmitted data due to infrared pulses energizing the photo transistor base of the IrDA receiver, which pulls its output low. This inverted transistor output is then fed to the UARTsir_in port, which gives it the correct UART polarity.

The following figure shows the timing diagram for the IrDA SIR data format in comparison to standard serial format.

IrDA SIR Data Format



As previously mentioned, the UARTcan be configured to support a low-power reception mode. When the UART is configured in this mode, it is possible to receive SIR pulses of 1.41 microseconds (minimum pulse duration), as well as nominal 3/16 of a normal serial bit time. In order to use this lowpower reception mode, you must program the Low Power Divisor Latch (LPDLL/LPDLH) registers.

When IrDA SIR mode is enabled, the UART operates in a manner similar to when the mode is disabled, with one exception: data transfers can only occur in half-duplex fashion when IrDA SIR mode is enabled. This is because the IrDA SIR physical layer specifies a minimum of 10ms delay between transmission and reception; this 10ms delay must be generated by software.

11.2.3 FIFO Support

UART can be configured to implement FIFOs that buffer transmit and receive data. If FIFO support is not selected, then no FIFOs are implemented and only a single receive data byte and transmit data byte can be stored at a time in the RBR and THR registers; this implies a 16450-compatible mode of operation. However, in this mode of operation, most of the enhanced features are unavailable.

In FIFO mode, the FIFOs can be selected to be either of the following:

- External customer-supplied FIFO RAMs
- Internal D-flip-flop-based RAMs

If the configured FIFO depth is greater than 256, the FIFO memory selection is restricted to be external. Additionally, selecting internal memory restricts the Memory Read Port Type to D-flip-flop-based Synchronous read port RAMs.

When external RAM support is chosen, either synchronous or asynchronous RAMs can be used. Asynchronous RAM provides read data during the clock cycle that has the memory address and read signals active, for sampling on the next rising clock edge. Synchronous single stage RAM registers the data at the current address out and is not available until the next clock cycle; that is, the second rising clock edge.

When FIFO support is selected, an optional programmable FIFO Access mode is available for test purposes, which allows:

- Receive FIFO to be written by master
- Transmit FIFO to be read by master

When FIFO Access mode is not selected, none of the corresponding logic is implemented and the mode cannot be enabled, reducing overall gate counts.

When FIFO Access mode has been selected it can be enabled with the FIFO Access Register (FAR[0]). Once enabled, the control portions of the transmit and receive FIFOs are reset and the FIFOs are treated as empty.

Data can be written to the transmit FIFO as normal; however no serial transmission occurs in this mode—normal operation halted—and thus no data leave the FIFO. The data that has been written to the transmit FIFO can be read back with the Transmit FIFO Read (TFR) register, which when read gives the current data at the top of the transmit FIFO.

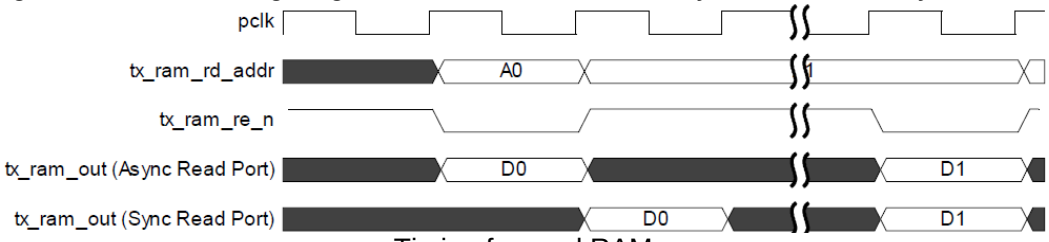
Similarly, data can be read from the receive FIFO as normal. Since the normal operation of the UART is halted in this mode, data must be written to the receive FIFO so the data can be read back.

Data is written to the receive FIFO using the Receive FIFO Write (RFW) register. The upper two bits of the 10-bit register are used to write framing error and parity error detection information to the receive FIFO, as follows:

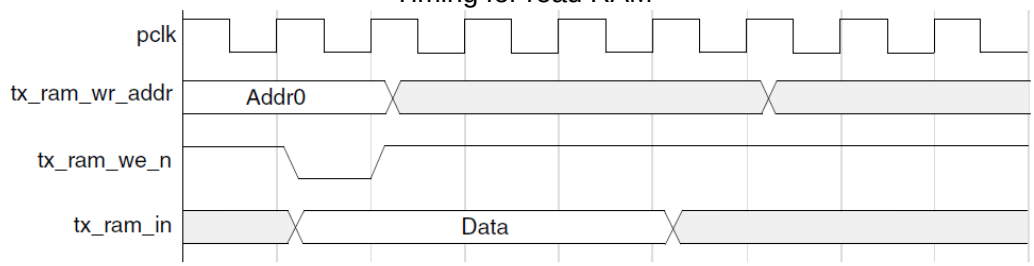
- RFW[9] indicates framing error
- RFW[8] indicates parity error

Although these bits cannot be read back through the Receive Buffer Register, they can be checked by reading the Line Status Register and checking the corresponding bits when the data in question is at the top of the receive FIFO.

The following figures show the timing diagram for read and write for asynchronous and synchronous RAM.



Timing for read RAM



Timing for write RAM

11.2.4 Clock support

The UART ~~can be configured to~~ have ~~either~~ one system clock (pclk) ~~or two system clocks (pclk and sclk). The second asynchronous serial clock (sclk) accommodates accurate serial baud rate settings, as well as APB bus interface requirements. When using a single system clock, available~~ system clock settings for accurate baud rates ~~are greatly restricted.~~

~~When a two-clock design is chosen, a synchronization module is implemented for synchronization of all control and data across the two-system clock boundaries.~~

11.2.5 Back-to-Back Character Stream Transmission

This section describes:

- Scenarios under which the UART is capable of transmitting back-to-back characters on the serial interface, with no idle time between them
- Worst-case idle time that exists between back-to-back characters

When the Transmit FIFO contains multiple data entries, the UART transmits the characters in the FIFO back-to-back on the serial bus. ~~However, if the CLOCK_MODE configuration parameter equals 2, synchronization delays in the UART can cause an IDLE period between the end of the current STOP bit and the beginning of the next START bit; this appears as an extended STOP bit duration on the serial bus.~~

~~11.2.5.1 Dual Clock Mode~~

When the CLOCK_MODE parameter equals 2—indicating an asynchronous relationship between pclk and sclk—the UART has a synchronization delay between the transmitter in the sclk domain and the TX FIFO in the pclk domain when querying if another character is ready for transmission. The transmitter begins the handshake one baud clock cycle before the end of the current STOP bit. The duration of the synchronization delay is given by the following equations:

sync_delay = (1sclk + 3pclk) + 1pclk + (1pclk + 3sclk)

sync_delay = 4sclk + 5pclk

If the *sync_delay* duration is longer than one baud clock period, an IDLE period will be inserted between the end of a STOP bit and the beginning of the next START bit.

To prevent insertion of the IDLE period, the following condition must be true:

sync_delay ≤ *bclk_period*

The baud clock period is given by the following equation:

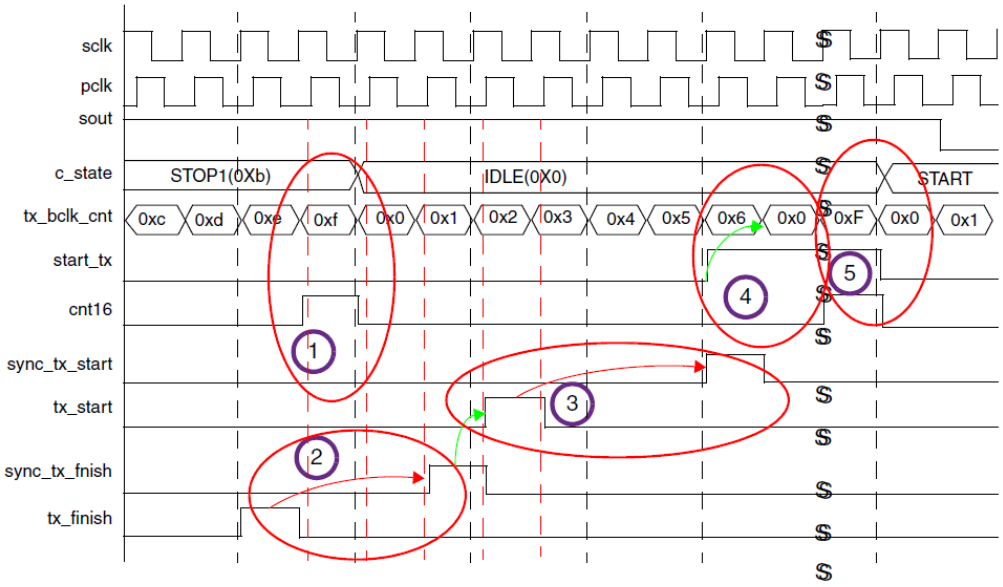
bclk_period = {DLH,DLL} * sclk

The worst case timing of the inserted IDLE period is given by:

worst_case_idle_duration = *sync_delay* + (15 * *bclk_period*)

The *worst_case_idle_duration* can be added to the programmed STOP bit duration to give the overall STOP bit period.

The following figure illustrates an example of character finish to character start delay.



1. The baud divisor is set to 1 ({DLH, DLL} = 1), so every sclk is a baud clock cycle. The transmit state machine changes state every sixteen baud clocks—eight in the case of a half STOP bit. At this point in above figure, after 16 baud clock cycles of the STOP1 state, the state machine enters the IDLE state on the next cycle because start_tx is not yet asserted.
2. One baud clock before the end of the STOP state, the transmit state machine decodes that the current character is complete and asserts tx_finish, which is synchronized to the pclk domain to become sync_tx_finish; this synchronization accounts for the “1sclk + 3pclk” term in *sync_delay*.
3. In the pclk domain, there is a one-pclk cycle delay—”1pclk” term in *sync_delay*—before the signal tx_start is asserted from the assertion of sync_tx_finish. Tx_start must then be synchronized to the sclk domain—”1pclk + 3sclk” term in *sync_delay*—to instruct the state machine to commence the START bit of the next character.
4. Start_tx asserts in the sclk domain, and causes the baud clock counter (tx_bclk_cnt) to go to 0.
5. Once sixteen baud clocks have been counted, the state machine can transition into the START state, and one cycle later sout is de-asserted.

11.2.5.2 Single Clock Mode

~~If CLOCK_MODE equals 1,~~ there is no idle time between back-to-back characters if data is ready in the transmit FIFO. In this case, because *sync_delay* equals one pclk ~~as described in “Dual Clock Mode”~~, the requirement to avoid idle time between consecutive characters is met for all {DLH,DLL} values.

sync_delay ≤ {DLH,DLL}*sclk

For example, when {DLH, DLL} equals 1 ~~(bearing in mind that when CLOCK_MODE = 1 : pclk = sclk),~~ then

1 pclk ≤ 1*pclk

11.2.6 Interrupts

Assertion of the UART interrupt output signal (intr)—a positive-level interrupt—occurs whenever one of the several prioritized interrupt types are enabled and active.

When an interrupt occurs, the master accesses the IIR register.

The following interrupt types can be enabled with the IER register:

- Receiver Error
- Receiver Data Available
- Character Timeout (in FIFO mode only)
- Transmitter Holding Register Empty at/below threshold (in Programmable THRE interrupt mode)

- Modem Status
- Busy Detect Indication

11.2.7 Auto Flow Control

The UART can be configured to have a 16750-compatible Auto RTS and Auto CTS serial data flow control mode available; if FIFOs are not implemented, this mode cannot be selected. When Auto Flow Control is not selected, none of the corresponding logic is implemented and the mode cannot be enabled, reducing overall gate counts. When Auto Flow Control mode is selected, it can be enabled with the Modem Control Register (MCR[5]).

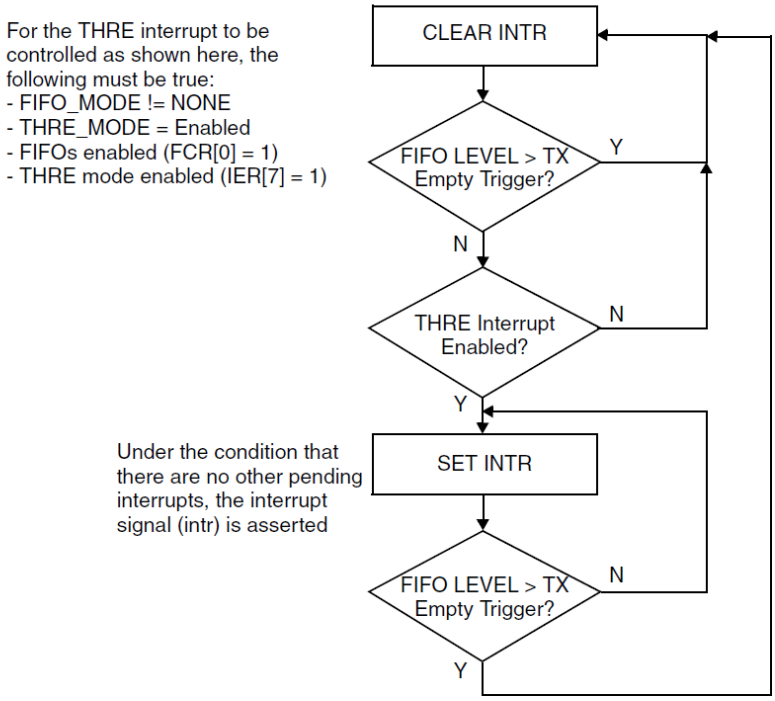
~~11.2.8 Programmable THRE Interrupt~~

The UART can be configured for a Programmable THRE Interrupt mode in order to increase system performance; if FIFOs are not implemented, then this mode cannot be selected.

- When Programmable THRE Interrupt mode is not selected, none of the logic is implemented and the mode cannot be enabled, reducing the overall gate counts.
- When Programmable THRE Interrupt mode is selected, it can be enabled using the Interrupt Enable Register (IER[7]).

When FIFOs and THRE mode are implemented and enabled, the THRE Interrupts and dma_tx_req_n are active at, and below, a programmed transmitter FIFO empty threshold level, as opposed to empty, as shown in the flowchart in following figure.

Flowchart of Interrupt Generation for Programmable THRE Interrupt Mode



The threshold level is programmed into FCR[5:4].

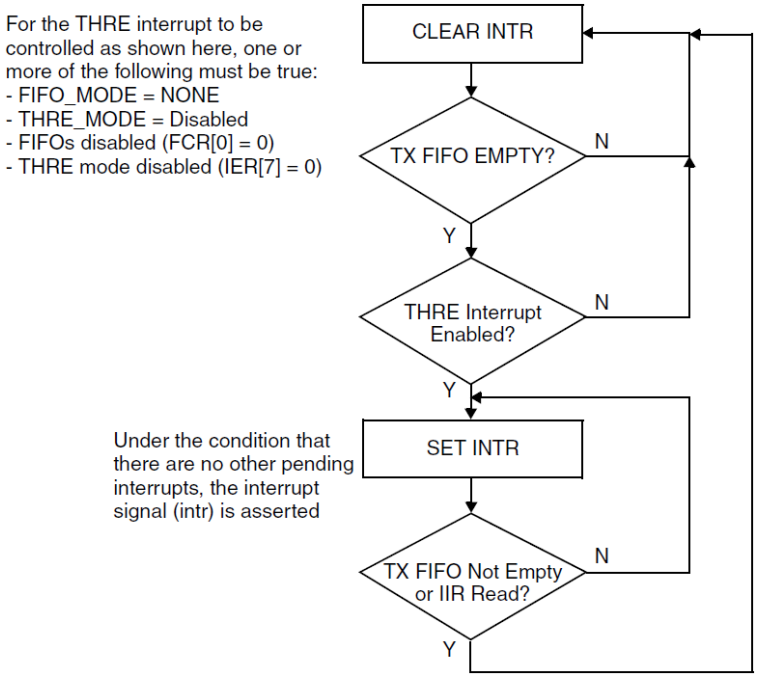
Selection of the best threshold value depends on the system's ability to begin a new transmission sequence in a timely manner. However, one of these thresholds should be optimal for increasing system performance by preventing the transmitter FIFO from running empty.

In addition to the interrupt change, the Line Status Register (LSR[5]) also switches from indicating that the transmitter FIFO is empty to the FIFO being full. This allows software to fill the FIFO for each transmit sequence by polling LSR[5] before writing another character. The flow then allows the transmitter FIFO to be filled whenever an interrupt occurs and there is data to transmit, rather than waiting until the FIFO is completely empty. Waiting until the FIFO is empty causes a reduction in performance whenever the system is too busy to respond immediately. Further system efficiency is achieved when this mode is enabled in combination with Auto Flow Control.

Even if everything else is selected and enabled, if the FIFOs are disabled using the FCR[0] bit, the Programmable THRE Interrupt mode is also disabled. When not selected or disabled, THRE interrupts and the LSR[5] bit

function normally, signifying an empty THR or FIFO. Folloginw figure illustrates the flowchart of THRE interrupt generation when not in programmable THRE interrupt mode.

Flowchart of Interrupt generation when not in Programmable THRE Interrupt Mode



~~11.2.9 DMA Support~~

The UART supports DMA signalling with the use of the dma_tx_req_n and dma_rx_req_n output signals to indicate:

- When data can be read
- When transmit FIFO is empty

~~11.2.9.1 DMA Modes~~

The UART uses two DMA channels—one for transmit data and one for receive data. There are two DMA modes:

- mode 0 – bit 3 of FIFO Control Register set to 0
- mode 1 – bit 3 of FIFO Control Register set to 1

DMA Mode 0

DMA mode 0 supports single DMA data transfers at a time.

In mode 0, the dma_tx_req_n signal:

- Goes active-low under the following conditions:
  - ❑ When Transmitter Holding Register is empty in non-FIFO mode
  - ❑ When transmitter FIFO is empty in FIFO mode with Programmable THRE interrupt mode disabled
  - ❑ When transmitter FIFO is at or below programmed threshold with Programmable THRE interrupt mode enabled
- Goes inactive when:
  - ❑ Single character has been written into Transmitter Holding Register or transmitter FIFO with Programmable THRE interrupt mode disabled
  - ❑ Transmitter FIFO is above threshold with Programmable THRE interrupt mode enabled

In mode 0, the dma_rx_req_n signal:

- ❑ Goes active-low when single character is available in Receiver FIFO or Receive Buffer Register
- ❑ Goes inactive when Receive Buffer Register or Receiver FIFO are empty, depending on FIFO Mode

DMA Mode 1

DMA mode 1 supports multi-DMA data transfers, where multiple transfers are made continuously until the receiver FIFO has been emptied or the transmit FIFO has been filled.

In mode 1, the dma_tx_req_n signal is asserted:

- When transmitter FIFO is empty with Programmable THRE interrupt mode disabled
- When transmitter FIFO is at or below programmed threshold with Programmable THRE interrupt mode enabled

In mode 1, the dma_tx_req_n signal is de-asserted when the transmitter FIFO is completely full.

In mode 1, the dma_rx_req_n signal is asserted:

- When Receiver FIFO is at or above programmed trigger level
- When character timeout has occurred; ERBFI does not need to be set

In mode 1, the dma_rx_req_n signal is de-asserted when the receiver FIFO becomes empty.

~~11.2.9.2 Transmit Watermark Level and Transmit FIFO Underflow~~

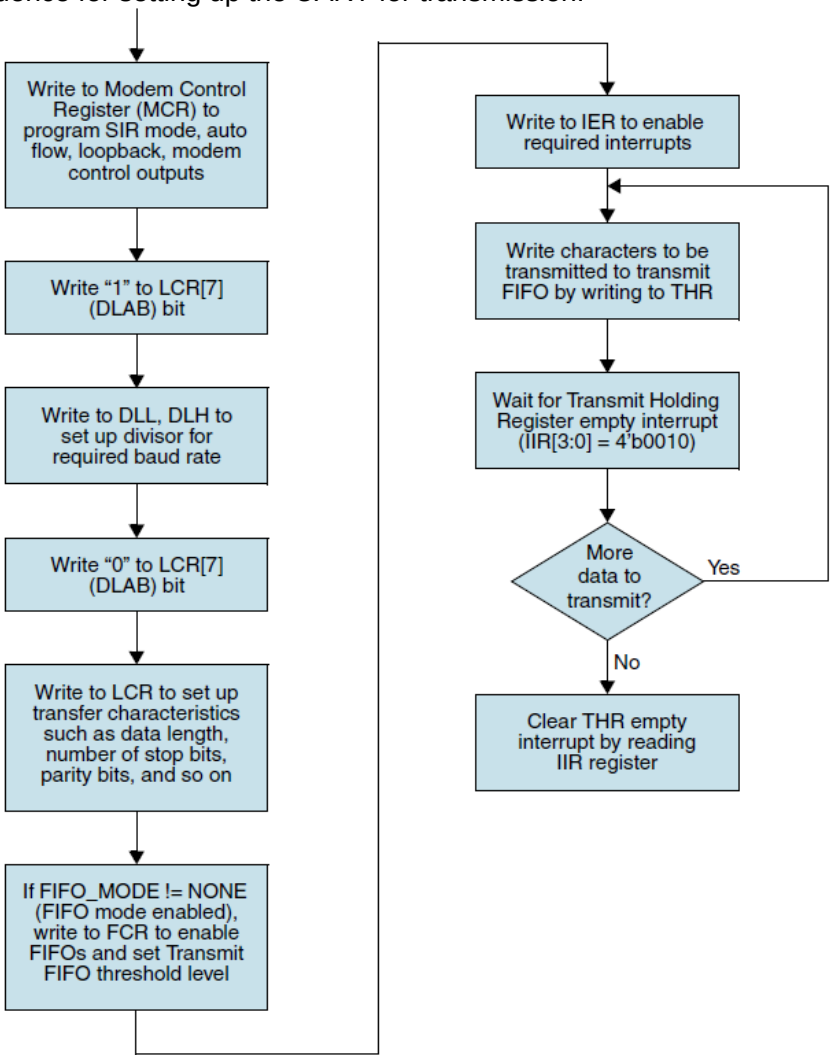
During UART serial transfers, transmit FIFO requests are made to the DW_ahb_dmac whenever the number of entries in the transmit FIFO is less than or equal to the decoded level of the Transmit Empty Trigger (TET) of the FCR register (bits 5:4); this is known as the watermark level. The DW_ahb_dmac responds by writing a burst of data to the transmit FIFO buffer, of length CTL*x*.DEST_MSIZE.

Data should be fetched from the DMA often enough for the transmit FIFO to perform serial transfers continuously; that is, when the FIFO begins to empty, another DMA request should be triggered. Otherwise the FIFO runs out of data (underflow). To prevent this condition, you must set the watermark level correctly.

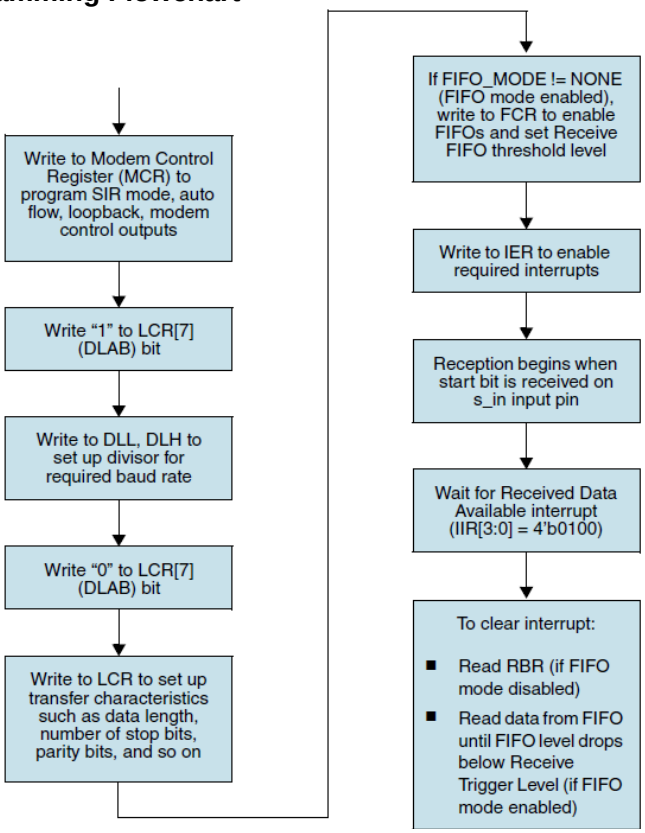
11.3 UART Programming

11.3.1 UART Transmit Programming Flowchart

The programming sequence for setting up the UART for transmission:



11.3.2 UART Receive Programming Flowchart



11.4 UART Registers

11.4.1 Register Memory Map

The UART has a number of internal registers that are accessed through the 5-bit address bus.

Name	Address Offset	Width	R/W	Description
RBR	0x00	32 bits	R	Receive Buffer Register **Reset Value:** 0x0 **Dependencies:** LCR[7] bit = 0
THR		32 bits	W	Transmit Holding Register **Reset Value:** 0x0 **Dependencies:** LCR[7] bit = 0
DLL		32 bits	R/W	Divisor Latch (Low) **Reset Value:** 0x0 **Dependencies:** LCR[7] bit = 1
DLH	0x04	32 bits	R/W	Divisor Latch (High) **Reset Value:** 0x0 **Dependencies:** LCR[7] bit = 1
IER		32 bits	R/W	Interrupt Enable Register **Reset Value:** 0x0 **Dependencies:** LCR[7] bit = 0
IIR	0x08	32 bits	R	Interrupt Identification Register **Reset Value:** 0x01
FCR		32 bits	W	FIFO Control Register **Reset Value:** 0x0

Name	Address Offset	Width	R/W	Description
LCR	0x0C	32 bits	R/W	Line Control Register **Reset Value:** 0x0
MCR	0x10	32 bits	R/W	Modem Control Register **Reset Value:** 0x0
LSR	0x14	32 bits	R	Line Status Register **Reset Value:** 0x60
MSR	0x18	32 bits	R	Modem Status Register **Reset Value:** 0x0
SCR	0x1C	32 bits	R/W	Scratchpad Register **Reset Value:** 0x0
~~LPDLL~~	~~0x20~~	~~32 bits~~	~~R/W~~	~~Low Power Divisor Latch (Low) Register~~ ~~**Reset Value:** 0x0~~ ~~**Dependencies:** LCR[7] bit = 1~~
~~LPDLH~~	~~0x24~~	~~32 bits~~	~~R/W~~	~~Low Power Divisor Latch (High) Register~~ ~~**Reset Value:** 0x0~~ ~~**Dependencies:** LCR[7] bit = 1~~
Reserved	0x28 - 0x2C	-	-	-
~~SRBR~~	~~0x30 - 0x6C~~	~~32 bits~~	~~R~~	~~Shadow Receive Buffer Register~~ ~~**Reset Value:** 0x0~~ ~~**Dependencies:** LCR[7] bit = 0~~
~~STHR~~		~~32 bits~~	~~W~~	~~Shadow Transmit Holding Register~~ ~~**Reset Value:** 0x0~~ ~~**Dependencies:** LCR[7] bit = 0~~
~~FAR~~	~~0x70~~	~~32 bits~~	~~R/W~~	~~FIFO Access Register~~ ~~**Reset Value:** 0x0~~
~~TFR~~	~~0x74~~	~~32 bits~~	~~R~~	~~Transmit FIFO Read~~ ~~**Reset Value:** 0x0~~
~~RFW~~	~~0x78~~	~~32 bits~~	~~W~~	~~Receive FIFO Write~~ ~~**Reset Value:** 0x0~~
USR	0x7C	32 bits	R	UART Status Register **Reset Value:** 0x6
~~TFL~~	~~0x80~~	~~See Description~~	~~R~~	~~Transmit FIFO Level~~ ~~**Width:** *FIFO_ADDR_WIDTH* + 1~~ ~~**Reset Value:** 0x0~~
~~RFL~~	~~0x84~~	~~See Description~~	~~R~~	~~Receive FIFO Level~~ ~~**Width:** *FIFO_ADDR_WIDTH* + 1~~ ~~**Reset Value:** 0x0~~
~~SRR~~	~~0x88~~	~~32 bits~~	~~W~~	~~Software Reset Register~~ ~~**Reset Value:** 0x0~~
~~SRTS~~	~~0x8C~~	~~32 bits~~	~~R/W~~	~~Shadow Request to Send~~ ~~**Reset Value:** 0x0~~
~~SBCR~~	~~0x90~~	~~32 bits~~	~~R/W~~	~~Shadow Break Control Register~~ ~~**Reset Value:** 0x0~~
~~SDMAM~~	~~0x94~~	~~32 bits~~	~~R/W~~	~~Shadow DMA Mode~~ ~~**Reset Value:** 0x0~~
~~SFE~~	~~0x98~~	~~32 bits~~	~~R/W~~	~~Shadow FIFO Enable~~ ~~**Reset Value:** 0x0~~
~~SRT~~	~~0x9C~~	~~32 bits~~	~~R/W~~	~~Shadow RCVR Trigger~~ ~~**Reset Value:** 0x0~~
~~STET~~	~~0xA0~~	~~32 bits~~	~~R/W~~	~~Shadow TX Empty Trigger~~ ~~**Reset Value:** 0x0~~
HTX	0xA4	32 bits	R/W	Halt TX **Reset Value:** 0x0
~~DMASA~~	~~0xA8~~	~~1 bit~~	~~W~~	~~DMA Software Acknowledge~~ ~~**Reset Value:** 0x0~~

Name	Address Offset	Width	R/W	Description
-	0xAC - 0xF0	-	-	-
~~CPR~~	~~0xF4~~	~~32 bits~~	~~R~~	~~Component Parameter Register~~ ~~**Reset Value:** Configuration-dependent~~
~~UCV~~	~~0xF8~~	~~32 bits~~	~~R~~	~~UART Component Version~~ ~~**Reset Value:** See the Releases table in the AMBA 2~~ ~~release notes.~~
~~CTR~~	~~0xFC~~	~~32 bits~~	~~R~~	~~Component Type Register~~ ~~**Reset Value:** 0x44570110~~

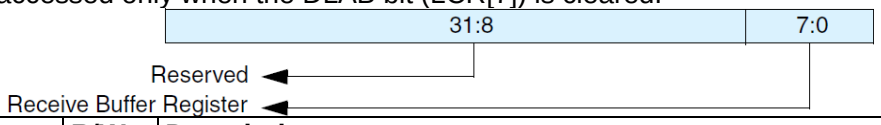
11.4.2 Register and Field Descriptions

The following subsections describe the data fields of the UART registers.

11.4.2.1 RBR

- **Name:** Receive Buffer Register
- **Size:** 32 bits
- **Address Offset:** 0x00
- **Read/write access:** read-only

This register can be accessed only when the DLAB bit (LCR[7]) is cleared.

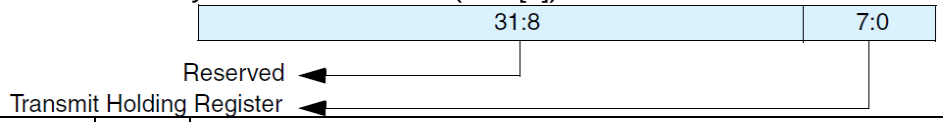


Bits	Name	R/W	Description
31:8	Reserved and read as 0		
7:0	Receive Buffer Register	R	Data byte received on the serial input port (sin) in UART mode, or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line Status Register (LSR) is set. If ~~in non-FIFO mode (FIFO_MODE = NONE) or~~ FIFOs are disabled (FCR[0] set to 0), the data in the RBR must be read before the next data arrives, otherwise it is overwritten, resulting in an over-run error. If in FIFO mode (~~FIFO_MODE != NONE~~) and FIFOs are enabled (FCR[0] set to 1), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO is preserved, but any incoming data are lost and an over-run error occurs. **Reset Value:** 0x0

11.4.2.2 THR

- **Name:** Transmit Holding Register
- **Size:** 32 bits
- **Address Offset:** 0x00
- **Read/write access:** write-only

This register can be accessed only when the DLAB bit (LCR[7]) is cleared.



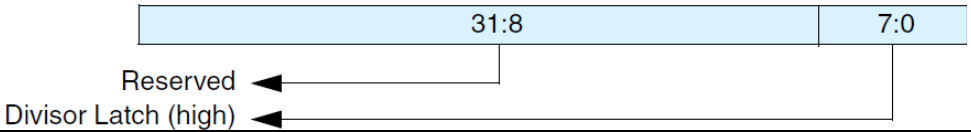
Bits	Name	R/W	Description
31:8	Reserved and read as 0		

Bits	Name	R/W	Description
7:0	Transmit Holding Register	W	Data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If in ~~non-FIFO mode or~~ FIFOs are disabled (FCR[0] = 0) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If in FIFO mode and FIFOs are enabled (FCR[0] = 1) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost. **Reset Value:** 0x0

11.4.2.3 DLH

- **Name:** Divisor Latch High
- **Size:** 32 bits
- **Address Offset:** 0x04
- **Read/write access:** read/write

~~If UART_16550_COMPATIBLE = No, then~~ this register can be accessed only when the DLAB bit (LCR[7]) is set and the UART is not busy—that is, USR[0] is 0; otherwise this register can be accessed only when the DLAB bit (LCR[7]) is set.

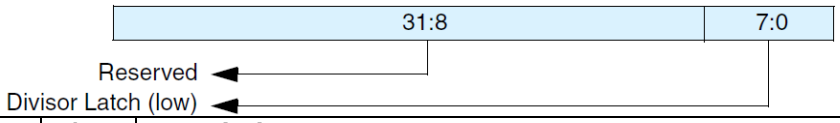


Bits	Name	R/W	Description
31:8	Reserved and read as 0		
7:0	Divisor Latch (High)	R/W	Upper 8-bits of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. The output baud rate is equal to the serial clock (pclk ~~if one clock design, sclk if two clock design (CLOCK_MODE = Enabled)~~ frequency divided by sixteen times the value of the baud rate divisor, as follows: baud rate = (serial clock freq) / (16 * divisor). Note that with the Divisor Latch Registers (DLL and DLH) set to 0, the baud clock is disabled and no serial communications occur. Also, once the DLH is set, at least 8 clock cycles of the slowest UART clock should be allowed to pass before transmitting or receiving data. **Reset Value:** 0x0

11.4.2.4 DLL

- **Name:** Divisor Latch Low
- **Size:** 32 bits
- **Address Offset:** 0x00
- **Read/write access:** read/write

~~If UART_16550_COMPATIBLE = No, then~~ this register can be accessed only when the DLAB bit (LCR[7]) is set and the UART is not busy—that is, USR[0] is 0; otherwise this register can be accessed only when the DLAB bit (LCR[7]) is set.



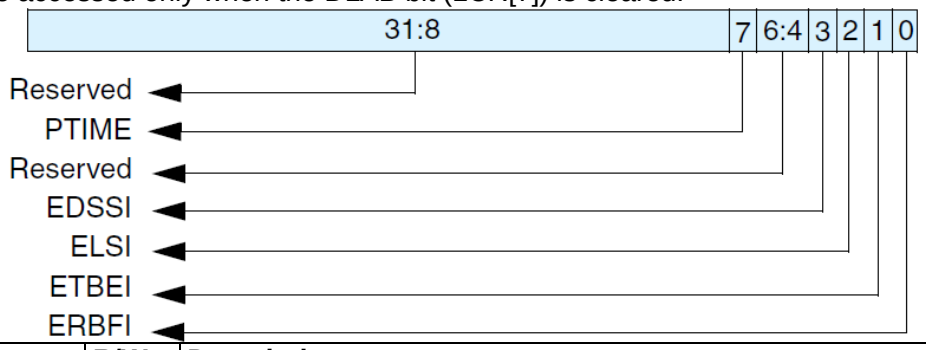
Bits	Name	R/W	Description
31:8	Reserved and read as 0		

Bits	Name	R/W	Description
7:0	Divisor Latch (Low)	R/W	Lower 8 bits of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. The output baud rate is equal to the serial clock (pclk ~~if one clock design, sclk if two clock design (CLOCK_MODE = Enabled)~~ frequency divided by sixteen times the value of the baud rate divisor, as follows: baud rate = (serial clock freq) / (16 * divisor). Note that with the Divisor Latch Registers (DLL and DLH) set to 0, the baud clock is disabled and no serial communications occur. Also, once the DLL is set, at least 8 clock cycles of the slowest UART clock should be allowed to pass before transmitting or receiving data. **Reset Value:** 0x0

11.4.2.5 IER

- **Name:** Interrupt Enable Register
- **Size:** 32 bits
- **Address Offset:** 0x04
- **Read/write access:** read/write

This register can be accessed only when the DLAB bit (LCR[7]) is cleared.

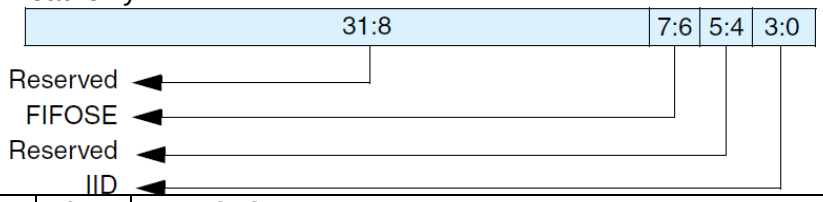


Bits	Name	R/W	Description
31:8	Reserved and read as 0		
7	~~PTIME~~	~~R/W~~	~~Programmable THRE Interrupt Mode Enable that can be written to only when THRE_MODE_USER = Enabled, always readable. This is used to enable/disable the generation of THRE Interrupt.~~ ~~■ 0 – disabled~~ ~~■ 1 – enabled~~ ~~**Reset Value:** 0x0~~
6:4	Reserved and read as 0		
3	EDSSI	R/W	Enable Modem Status Interrupt. This is used to enable/disable the generation of Modem Status Interrupt. This is the fourth highest priority interrupt. ■ 0 – disabled ■ 1 – enabled **Reset Value:** 0x0
2	ELSI	R/W	Enable Receiver Line Status Interrupt. This is used to enable/disable the generation of Receiver Line Status Interrupt. This is the highest priority interrupt. ■ 0 – disabled ■ 1 – enabled **Reset Value:** 0x0
1	ETBEI	R/W	Enable Transmit Holding Register Empty Interrupt. This is used to enable/disable the generation of Transmitter Holding Register Empty Interrupt. This is the third highest priority interrupt. ■ 0 – disabled ■ 1 – enabled **Reset Value:** 0x0

Bits	Name	R/W	Description
0	ERBFI	R/W	Enable Received Data Available Interrupt. This is used to enable/disable the generation of Received Data Available Interrupt and the Character Timeout Interrupt (if in FIFO mode and FIFOs enabled). These are the second highest priority interrupts. ■ 0 – disabled ■ 1 – enabled **Reset Value:** 0x0

11.4.2.6 IIR

- **Name:** Interrupt Identity Register
- **Size:** 32 bits
- **Address Offset:** 0x08
- **Read/write access:** read-only



Bits	Name	R/W	Description
31:8	Reserved and read as 0		
7:6	FIFOs Enabled (or FIFOSE)	R	FIFOs Enabled. This is used to indicate whether the FIFOs are enabled or disabled. ■ 00 – disabled ■ 11 – enabled **Reset Value:** 0x00
5:4	Reserved	N/A	Reserved and read as 0
3:0	Interrupt ID (or IID)	R	Interrupt ID. This indicates the highest priority pending interrupt which can be one of the following types: ■ 0000 – modem status ■ 0001 – no interrupt pending ■ 0010 – THR empty ■ 0100 – received data available ■ 0110 – receiver line status ■ 0111 – busy detect ■ 1100 – character timeout The interrupt priorities are split into several levels that are detailed in following Table. Bit 3 indicates an interrupt can only occur when the FIFOs are enabled and used to distinguish a Character Timeout condition interrupt. **Reset Value:** 0x01

Table Interrupt Control Functions

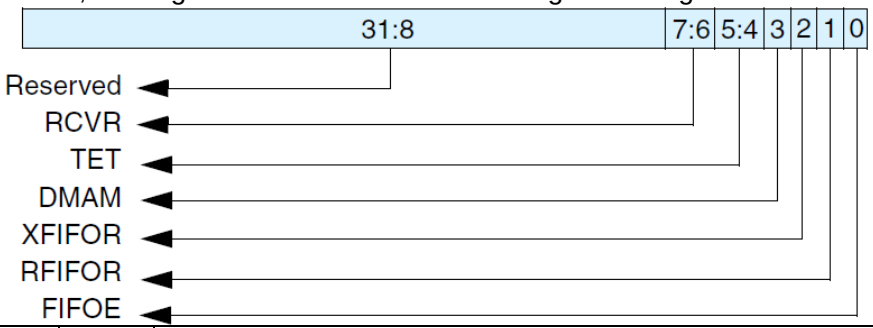
Interrupt ID				Interrupt Set and Reset Functions			
Bit3	**Bit2**	**Bit1**	**Bit0**	**Priority Level**	**Interrupt Type**	**Interrupt Source**	**Interrupt Reset Control**
0	0	0	1	-	None	None	–
0	1	1	0	Highest	Receiver line status	Overrun/parity/ framing errors or break interrupt	Reading the line status register
0	1	0	0	Second	Received data available	Receiver data available (~~non-FIFO mode or~~ FIFOs disabled) or RCVR FIFO trigger level reached (~~FIFO mode and~~ FIFOs enabled)	Reading the receiver buffer register (~~non-FIFO mode or~~ FIFOs disabled) or the FIFO drops below the trigger level (~~FIFO mode and~~ FIFOs enabled)

Interrupt ID				Interrupt Set and Reset Functions			
Bit3	**Bit2**	**Bit1**	**Bit0**	**Priority Level**	**Interrupt Type**	**Interrupt Source**	**Interrupt Reset Control**
1	1	0	0	Second	Character timeout indication	No characters in or out of the RCVR FIFO during the last 4 character times and there is at least 1 character in it during this time	Reading the receiver buffer register
0	0	1	0	Third	Transmit holding register empty	Transmitter holding register empty ~~(Prog. THRE Mode disabled) or XMIT FIFO at or below threshold (Prog. THRE Mode enabled)~~	Reading the IIR register (if source of interrupt); or, writing into THR ~~(FIFOs or THRE Mode not selected or disabled) or XMIT FIFO above threshold (FIFOs and THRE Mode selected and enabled).~~
0	0	0	0	Fourth	Modem status	Clear to send or data set ready or ring indicator or data carrier detect. Note that if auto flow control mode is enabled, a change in CTS (that is, DCTS set) does not cause an interrupt	Reading the Modem status Register
0	1	1	1	Fifth	Busy detect indication	~~UART_16550_COMPATIBLE = NO and~~ master has tried to write to the Line Control Register while the UART is busy (USR[0] is set to 1).	Reading the UART status register

11.4.2.7 FCR

- **Name:** FIFO Control Register
- **Size:** 32 bits
- **Address Offset:** 0x08
- **Read/write access:** write-only

~~This register is valid only when the UART is configured to have FIFOs implemented (FIFO_MODE != NONE). If FIFOs are not implemented, this register does not exist and writing to this register address has no effect.~~

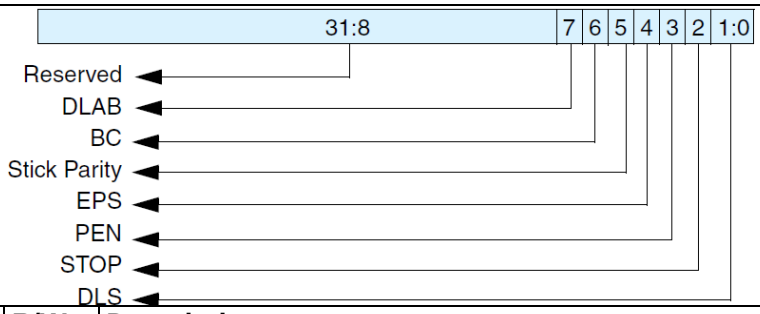


Bits	Name	R/W	Description
31:8	Reserved and read as 0		

Bits	Name	R/W	Description
7:6	RCVR Trigger (or RT)	W	RCVR Trigger. This is used to select the trigger level in the receiver FIFO at which the Received Data Available Interrupt is generated. In auto flow control mode, this trigger is used to determine when the rts_n signal is de-asserted only when RTC_FCT is disabled. It also determines when the dma_rx_req_n signal is asserted in certain modes of operation . The following trigger levels are supported: ■ 00 – 1 character in the FIFO ■ 01 – FIFO ¼ full ■ 10 – FIFO ½ full ■ 11 – FIFO 2 less than full **Reset Value:** 0x0
5:4	~~TX Empty Trigger (or TET)~~	W	TX Empty Trigger. Writes have no effect ~~when THRE_MODE_USER = Disabled~~. ~~This is used to select the empty threshold level at which the THRE Interrupts are generated when the mode is active.~~ It also determines when the dma_tx_req_n signal is asserted when in certain modes of operation. For details on DMA support. The following trigger levels are supported: ■ 00 – FIFO empty ■ 01 – 2 characters in the FIFO ■ 10 – FIFO ¼ full ■ 11 – FIFO ½ full **Reset Value:** 0x0
3	DMA Mode (or DMAM)	W	DMA Mode. This determines the DMA signalling mode used for the dma_tx_req_n and dma_rx_req_n output signals ~~when additional DMA handshaking signals are not selected (DMA_EXTRA = No)~~. ■ 0 – mode 0 ■ 1 – mode 1 **Reset Value:** 0x0
2	XMIT FIFO Reset (or XFIFOR)	W	XMIT FIFO Reset. This resets the control portion of the transmit FIFO and treats the FIFO as empty. ~~This also de-asserts the DMA TX request and single signals when additional DMA handshaking signals are selected (DMA_EXTRA = YES).~~ Note that this bit is 'self-clearing'. It is not necessary to clear this bit. **Reset Value:** 0x0
1	RCVR FIFO Reset (or RFIFOR)	W	RCVR FIFO Reset. This resets the control portion of the receive FIFO and treats the FIFO as empty. ~~This also de-asserts the DMA RX request and single signals when additional DMA handshaking signals are selected (DMA_EXTRA = YES).~~ Note that this bit is 'self-clearing'. It is not necessary to clear this bit. **Reset Value:** 0x0
0	FIFO Enable (or FIFOE)	W	FIFO Enable. This enables/disables the transmit (XMIT) and receive (RCVR) FIFOs. Whenever the value of this bit is changed both the XMIT and RCVR controller portion of FIFOs is reset. **Reset Value:** 0x0

11.4.2.8 LCR

- **Name:** Line Control Register
- **Size:** 32 bits
- **Address Offset:** 0x0C
- **Read/write access:** read/write

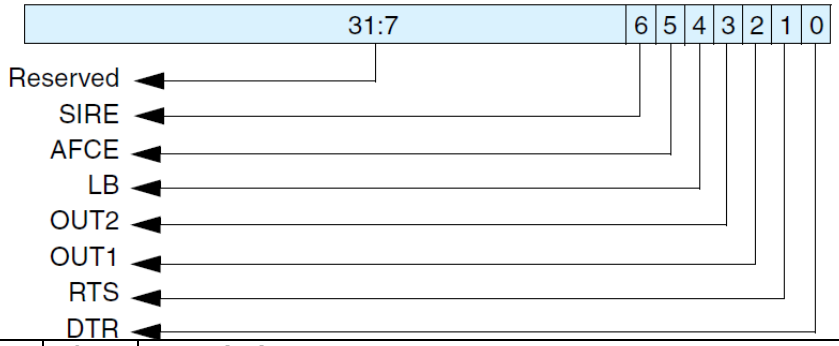


Bits	Name	R/W	Description
31:8	Reserved and read as 0		
7	DLAB	R/W	Divisor Latch Access Bit. ~~If UART_16550_COMPATIBLE = NO, then~~ writeable only when UART is not busy (USR[0] is 0); ~~otherwise always writable,~~ always readable. This bit is used to enable reading and writing of the Divisor Latch register (DLL and DLH/LPDLL and LPDLH) to set the baud rate of the UART. This bit must be cleared after initial baud rate setup in order to access other registers. **Reset Value:** 0x0
6	Break (or BC)	R/W	Break Control Bit.This is used to cause a break condition to be transmitted to the receiving device. If set to 1, the serial output is forced to the spacing (logic 0) state. When not in Loopback Mode, as determined by MCR[4], the sout line is forced low until the Break bit is cleared. ~~If SIR_MODE = Enabled and active (MCR[6] set to 1) the sir_out_n line is continuously pulsed.~~ When in Loopback Mode, the break condition is internally looped back to the receiver and the sir_out_n line is forced low. **Reset Value:** 0x0
5	Stick Parity	R/W	Stick Parity. ~~If UART_16550_COMPATIBLE = NO, then~~ writeable only when UART is not busy (USR[0] is 0); ~~otherwise always writable,~~ always readable. This bit is used to force parity value. When PEN, EPS, and Stick Parity are set to 1, the parity bit is transmitted and checked as logic 0. If PEN and Stick Parity are set to 1 and EPS is a logic 0, then parity bit is transmitted and checked as a logic 1. If this bit is set to 0, Stick Parity is disabled. **Reset Value:** 0x0
4	EPS	R/W	Even Parity Select. ~~If UART_16550_COMPATIBLE = NO, then~~ writeable only when UART is not busy (USR[0] is 0); ~~otherwise always writable,~~ always readable. This is used to select between even and odd parity, when parity is enabled (PEN set to 1). If set to 1, an even number of logic 1s is transmitted or checked. If set to 0, an odd number of logic 1s is transmitted or checked. **Reset Value:** 0x0
3	PEN	R/W	Parity Enable. ~~If UART_16550_COMPATIBLE = NO, then~~ writeable only when UART is not busy (USR[0] is 0); ~~otherwise always writable,~~ always readable. This bit is used to enable and disable parity generation and detection in transmitted and received serial character respectively. ■ 0 – parity disabled ■ 1 – parity enabled **Reset Value:** 0x0

Bits	Name	R/W	Description
2	STOP	R/W	Number of stop bits. ~~If UART_16550_COMPATIBLE = NO, then~~ writeable only when UART is not busy (USR[0] is 0); ~~otherwise always writable,~~ always readable. This is used to select the number of stop bits per character that the peripheral transmits and receives. If set to 0, one stop bit is transmitted in the serial data. If set to 1 and the data bits are set to 5 (LCR[1:0] set to 0) one and a half stop bits is transmitted. Otherwise, two stop bits are transmitted. Note that regardless of the number of stop bits selected, the receiver checks only the first stop bit. ■ 0 – 1 stop bit ■ 1 – 1.5 stop bits when DLS (LCR[1:0]) is 0, else 2 stop bit NOTE: The STOP bit duration implemented by UART may appear longer due to idle time inserted between characters for some configurations and baud clock divisor values in the transmit direction; for details on idle time between transmitted transfers. **Reset Value:** 0x0
1:0	DLS (or CLS, as used in legacy)	R/W	Data Length Select. ~~If UART_16550_COMPATIBLE = NO, then~~ writeable only when UART is not busy (USR[0] is 0); otherwise always writable, always readable. This is used to select the number of data bits per character that the peripheral transmits and receives. The number of bit that may be selected areas follows: ■ 00 – 5 bits ■ 01 – 6 bits ■ 10 – 7 bits ■ 11 – 8 bits **Reset Value:** 0x0

11.4.2.9 MCR

- **Name:** Modem Control Register
- **Size:** 32 bits
- **Address Offset:** 0x10
- **Read/write access:** read/write

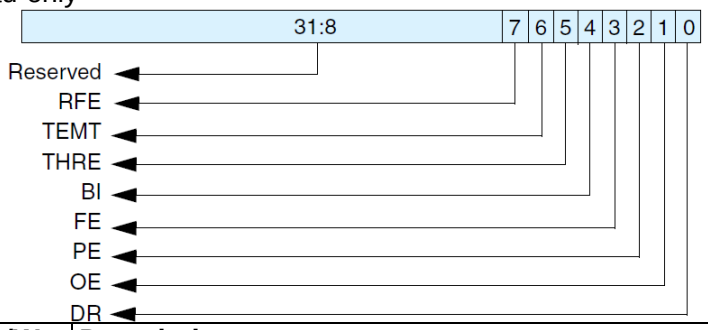


Bits	Name	R/W	Description
31:7	Reserved and read as 0		
6	~~SIRE~~	~~R/W~~	~~SIR Mode Enable. Writeable only when SIR_MODE = Enabled, always readable. This is used to enable/disable the IrDA SIR Mode features.~~ ~~■ 0 – IrDA SIR Mode disabled~~ ~~■ 1 – IrDA SIR Mode enabled~~ ~~**Reset Value:** 0x0~~
5	~~AFCE~~	~~R/W~~	~~Auto Flow Control Enable. Writeable only when AFCE_MODE = Enabled, always readable. When FIFOs are enabled and the Auto Flow Control Enable (AFCE) bit is set.~~ ~~■ 0 – Auto Flow Control Mode disabled~~ ~~■ 1 – Auto Flow Control Mode enabled~~ ~~**Reset Value:** 0x0~~

Bits	Name	R/W	Description
4	LoopBack (or LB)	R/W	LoopBack Bit. This is used to put the UART into a diagnostic mode for test purposes. If operating in UART mode ~~(SIR_MODE != Enabled or not active, MCR[6] set to 0)~~, data on the sout line is held high, while serial data output is looped back to the sin line, internally. In this mode all the interrupts are fully functional. Also, in loopback mode, the modem control inputs (dsr_n, cts_n, ri_n, dcd_n) are disconnected and the modem control outputs (dtr_n, rts_n, out1_n, out2_n) are looped back to the inputs, internally. ~~If operating in infrared mode (SIR_MODE = Enabled AND active, MCR[6] set to 1), data on the sir_out_n line is held low, while serial data output is inverted and looped back to the sir_in line.~~ **Reset Value:** 0x0
3	OUT2	R/W	OUT2. This is used to directly control the user-designated Output2 (out2_n) output. The value written to this location is inverted and driven out on out2_n, that is: ■ 0 – out2_n de-asserted (logic 1) ■ 1 – out2_n asserted (logic 0) Note that in Loopback mode (MCR[4] set to 1), the out2_n output is held inactive high while the value of this location is internally looped back to an input. **Reset Value:** 0x0
2	OUT1	R/W	OUT1. This is used to directly control the user-designated Output1 (out1_n) output. The value written to this location is inverted and driven out on out1_n, that is: ■ 0 – out1_n de-asserted (logic 1) ■ 1 – out1_n asserted (logic 0) Note that in Loopback mode (MCR[4] set to 1), the out1_n output is held inactive high while the value of this location is internally looped back to an input. **Reset Value:** 0x0
1	RTS	R/W	Request to Send. This is used to directly control the Request to Send (rts_n) output. The Request To Send (rts_n) output is used to inform the modem or data set that the UART is ready to exchange data. When Auto RTS Flow Control is not enabled (MCR[5] set to 0), the rts_n signal is set low by programming MCR[1] (RTS) to a high.~~In Auto Flow Control, AFCE_MODE = Enabled and active (MCR[5] set to 1) and FIFOs enable (FCR[0] set to 1), the rts_n output is controlled in the same way, but is also gated with the receiver FIFO threshold trigger (rts_n is inactive high when above the threshold) only when the RTC Flow Trigger is disabled; otherwise it is gated by the receiver FIFO almost-full trigger, where "almost full" refers to two available slots in the FIFO (rts_n is inactive high when above the threshold).~~ The rts_n signal is de-asserted when MCR[1] is set low. Note that in Loopback mode (MCR[4] set to 1), the rts_n output is held inactive high while the value of this location is internally looped back to an input. **Reset Value:** 0x0
0	DTR	R/W	Data Terminal Ready. This is used to directly control the Data Terminal Ready (dtr_n) output. The value written to this location is inverted and driven out on dtr_n, that is: ■ 0 – dtr_n de-asserted (logic 1) ■ 1 – dtr_n asserted (logic 0) The Data Terminal Ready output is used to inform the modem or data set that the UART is ready to establish communications. Note that in Loopback mode (MCR[4] set to 1), the dtr_n output is held inactive high while the value of this location is internally looped back to an input. **Reset Value:** 0x0

11.4.2.10 LSR

- **Name:** Line Status Register
- **Size:** 32 bits
- **Address Offset:** 0x14
- **Read/write access:** read-only



Bits	Name	R/W	Description
31:8	Reserved and read as 0		
7	RFE	R	Receiver FIFO Error bit. This bit is only relevant when ~~FIFO_MODE != NONE AND~~ FIFOs are enabled (FCR[0] set to 1). This is used to indicate if there is at least one parity error, framing error, or break indication in the FIFO. ■ 0 – no error in RX FIFO ■ 1 – error in RX FIFO This bit is cleared when the LSR is read and the character with the error is at the top of the receiver FIFO and there are no subsequent errors in the FIFO. **Reset Value:** 0x0
6	TEMT	R	Transmitter Empty bit. If ~~in FIFO mode (FIFO_MODE != NONE) and~~ FIFOs enabled (FCR[0] set to 1), this bit is set whenever the Transmitter Shift Register and the FIFO are both empty. If ~~in non-FIFO mode or~~ FIFOs are disabled, this bit is set whenever the Transmitter Holding Register and the Transmitter Shift Register are both empty. **Reset Value:** 0x1
5	THRE	R	Transmit Holding Register Empty bit. ~~If THRE_MODE_USER = Disabled or THRE mode is disabled (IER[7] set to 0) and regardless of FIFO's being implemented/enabled or not,~~ this bit indicates that ~~the THR or~~ TX FIFO is empty. This bit is set whenever data is transferred from ~~the THR or~~ TX FIFO to the transmitter shift register and no new data has been written to ~~the THR or~~ TX FIFO. ~~This also causes a THRE Interrupt to occur, if the THRE Interrupt is enabled. If THRE_MODE_USER = Enabled AND FIFO_MODE != NONE and both modes are active (IER[7] set to 1 and FCR[0] set to 1 respectively), the functionality is switched to indicate the transmitter FIFO is full, and no longer controls THRE interrupts, which are then controlled by the FCR[5:4] threshold setting.~~ **Reset Value:** 0x1

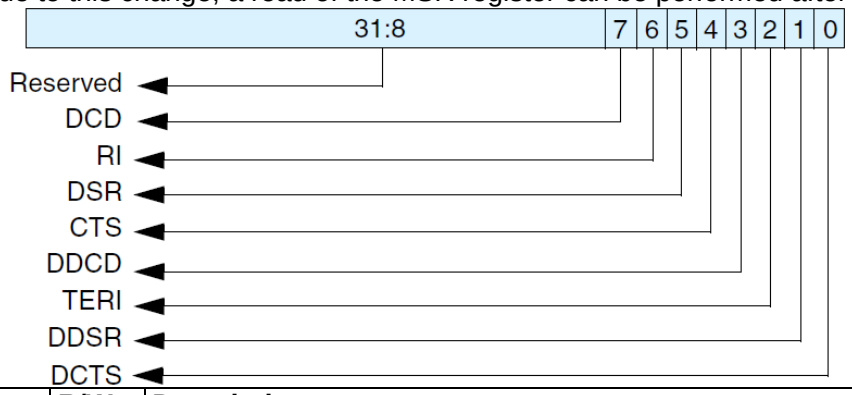
Bits	Name	R/W	Description
4	BI	R	Break Interrupt bit. This is used to indicate the detection of a break sequence on the serial input data. If in UART mode ~~(SIR_MODE = Disabled)~~, it is set whenever the serial input, sin, is held in a logic '0' state for longer than the sum of *start time* + *data bits* + *parity* + *stop bits*. ~~If in infrared mode (SIR_MODE = Enabled), it is set whenever the serial input, sir_in, is continuously pulsed to logic '0' for longer than the sum of *start time* + *data bits* + *parity* + *stop bits*. A break condition on serial input causes one and only one character, consisting of all 0s, to be received by the UART.~~ In FIFO mode, the character associated with the break condition is carried through the FIFO and is revealed when the character is at the top of the FIFO. Reading the LSR clears the BI bit. In non-FIFO mode, the BI indication occurs immediately and persists until the LSR is read. **NOTE:** If a FIFO is full when a break condition is received, a FIFO overrun occurs. The break condition and all the information associated with it—parity and framing errors—is discarded; any information that a break character was received is lost. **Reset Value:** 0x0
3	FE	R	Framing Error bit. This is used to indicate the occurrence of a framing error in the receiver. A framing error occurs when the receiver does not detect a valid STOP bit in the received data. In the FIFO mode, since the framing error is associated with a character received, it is revealed when the character with the framing error is at the top of the FIFO. When a framing error occurs, the UART tries to resynchronize. It does this by assuming that the error was due to the start bit of the next character and then continues receiving the other bit; that is, data, and/or parity and stop. It should be noted that the Framing Error (FE) bit (LSR[3]) is set if a break interrupt has occurred, as indicated by Break Interrupt (BI) bit (LSR[4]). This happens because the break character implicitly generates a framing error by holding the sin input to logic 0 for longer than the duration of a character. ■ 0 – no framing error ■ 1 – framing error Reading the LSR clears the FE bit. **Reset Value:** 0x0
2	PE	R	Parity Error bit. This is used to indicate the occurrence of a parity error in the receiver if the Parity Enable (PEN) bit (LCR[3]) is set. In the FIFO mode, since the parity error is associated with a character received, it is revealed when the character with the parity error arrives at the top of the FIFO. It should be noted that the Parity Error (PE) bit (LSR[2]) can be set if a break interrupt has occurred, as indicated by Break Interrupt (BI) bit (LSR[4]). In this situation, the Parity Error bit is set if parity generation and detection is enabled (LCR[3]=1) and the parity is set to odd (LCR[4]=0). ■ 0 – no parity error ■ 1 – parity error Reading the LSR clears the PE bit. **Reset Value:** 0x0

Bits	Name	R/W	Description
1	OE	R	Overrun error bit. This is used to indicate the occurrence of an overrun error. This occurs if a new data character was received before the previous data was read. In the non-FIFO mode, the OE bit is set when a new character arrives in the receiver before the previous character was read from the RBR. When this happens, the data in the RBR is overwritten. In the FIFO mode, an overrun error occurs when the FIFO is full and a new character arrives at the receiver. The data in the FIFO is retained and the data in the receive shift register is lost. ■ 0 – no overrun error ■ 1 – overrun error Reading the LSR clears the OE bit. **Reset Value:** 0x0
0	DR	R	Data Ready bit. This is used to indicate that the receiver contains at least one character in the RBR or the receiver FIFO. ■ 0 – no data ready ■ 1 – data ready This bit is cleared when the RBR is read in non-FIFO mode, or when the receiver FIFO is empty, in FIFO mode. **Reset Value:** 0x0

11.4.2.11 MSR

- **Name:** Modem Status Register
- **Size:** 32 bits
- **Address Offset:** 0x18
- **Read/write access:** read-only

Whenever bits 0, 1, 2 or 3 are set to logic 1, to indicate a change on the modem control inputs, a modem status interrupt is generated if enabled through the IER, regardless of when the change occurred. The bits of this register can be set after a reset—even though their respective modem signals are inactive—because the synchronized version of the modem signals have a reset value of 0 and change to value 1 after reset. To prevent unwanted interrupts due to this change, a read of the MSR register can be performed after reset.



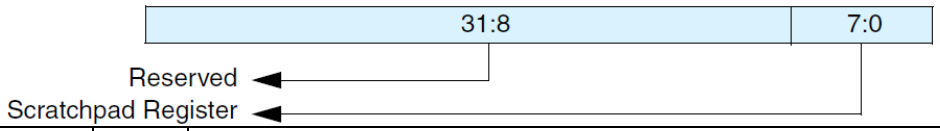
Bits	Name	R/W	Description
31:8	Reserved and read as 0		
7	DCD	R	Data Carrier Detect. This is used to indicate the current state of the modem control line dcd_n. This bit is the complement of dcd_n. When the Data Carrier Detect input (dcd_n) is asserted it is an indication that the carrier has been detected by the modem or data set. ■ 0 – dcd_n input is de-asserted (logic 1) ■ 1 – dcd_n input is asserted (logic 0) In Loopback Mode (MCR[4] set to 1), DCD is the same as MCR[3] (Out2). **Reset Value:** 0x0

Bits	Name	R/W	Description
6	RI	R	Ring Indicator. This is used to indicate the current state of the modem control line ri_n. This bit is the complement of ri_n. When the Ring Indicator input (ri_n) is asserted it is an indication that a telephone ringing signal has been received by the modem or data set. ■ 0 – ri_n input is de-asserted (logic 1) ■ 1 – ri_n input is asserted (logic 0) In Loopback Mode (MCR[4] set to 1), RI is the same as MCR[2] (Out1). **Reset Value:** 0x0
5	DSR	R	Data Set Ready. This is used to indicate the current state of the modem control line dsr_n. This bit is the complement of dsr_n. When the Data Set Ready input (dsr_n) is asserted it is an indication that the modem or data set is ready to establish communications with the UART. ■ 0 – dsr_n input is de-asserted (logic 1) ■ 1 – dsr_n input is asserted (logic 0) In Loopback Mode (MCR[4] set to 1), DSR is the same as MCR[0] (DTR). **Reset Value:** 0x0
4	CTS	R	Clear to Send. This is used to indicate the current state of the modem control line cts_n. This bit is the complement of cts_n. When the Clear to Send input (cts_n) is asserted it is an indication that the modem or data set is ready to exchange data with the UART. ■ 0 – cts_n input is de-asserted (logic 1) ■ 1 – cts_n input is asserted (logic 0) In Loopback Mode (MCR[4] = 1), CTS is the same as MCR[1] (RTS). **Reset Value:** 0x0
3	DDCD	R	Delta Data Carrier Detect. This is used to indicate that the modem control line dcd_n has changed since the last time the MSR was read. ■ 0 – no change on dcd_n since last read of MSR ■ 1 – change on dcd_n since last read of MSR Reading the MSR clears the DDCD bit. In Loopback Mode (MCR[4] = 1), DDCD reflects changes on MCR[3] (Out2). Note, if the DDCD bit is not set and the dcd_n signal is asserted (low) and a reset occurs (software or otherwise), then the DDCD bit is set when the reset is removed if the dcd_n signal remains asserted. **Reset Value:** 0x0
2	TERI	R	Trailing Edge of Ring Indicator. This is used to indicate that a change on the input ri_n (from an active-low to an inactive-high state) has occurred since the last time the MSR was read. ■ 0 – no change on ri_n since last read of MSR ■ 1 – change on ri_n since last read of MSR Reading the MSR clears the TERI bit. In Loopback Mode (MCR[4] = 1), TERI reflects when MCR[2] (Out1) has changed state from a high to a low. **Reset Value:** 0x0
1	DDSR	R	Delta Data Set Ready. This is used to indicate that the modem control line dsr_n has changed since the last time the MSR was read. ■ 0 – no change on dsr_n since last read of MSR ■ 1 – change on dsr_n since last read of MSR Reading the MSR clears the DDSR bit. In Loopback Mode (MCR[4] = 1), DDSR reflects changes on MCR[0] (DTR). Note, if the DDSR bit is not set and the dsr_n signal is asserted (low) and a reset occurs (software or otherwise), then the DDSR bit is set when the reset is removed if the dsr_n signal remains asserted. **Reset Value:** 0x0

Bits	Name	R/W	Description
0	DCTS	R	Delta Clear to Send. This is used to indicate that the modem control line cts_n has changed since the last time the MSR was read. ■ 0 – no change on cts_n since last read of MSR ■ 1 – change on cts_n since last read of MSR Reading the MSR clears the DCTS bit. In Loopback Mode (MCR[4] = 1), DCTS reflects changes on MCR[1] (RTS). Note, if the DCTS bit is not set and the cts_n signal is asserted (low) and a reset occurs (software or otherwise), then the DCTS bit is set when the reset is removed if the cts_n signal remains asserted. **Reset Value:** 0x0

11.4.2.12 SCR

- **Name:** Scratchpad Register
- **Size:** 32 bits
- **Address Offset:** 0x1C
- **Read/write access:** read/write

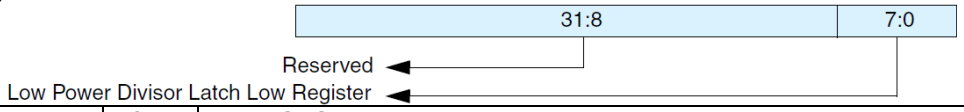


Bits	Name	R/W	Description
31:8	Reserved and read as 0		
7:0	Scratchpad Register	R/W	R/W This register is for programmers to use as a temporary storage space. It has no defined purpose in the UART. **Reset Value:** 0x0

~~11.4.2.13 LPDLL~~

- **Name:** Low Power Divisor Latch Low Register
- **Size:** 32 bits
- **Address Offset:** 0x20
- **Read/write access:** read/write

This register is only valid when the UART is configured to have SIR low-power reception capabilities implemented (SIR_LP_RX = Yes). If SIR low-power reception capabilities are not implemented, this register does not exist and reading from this register address returns 0.
If UART_16550_COMPATIBLE = No, then this register can be accessed only when the DLAB bit (LCR[7]) is set and the UART is not busy—that is, USR[0] is 0; otherwise this register can be accessed only when the DLAB bit (LCR[7]) is set.



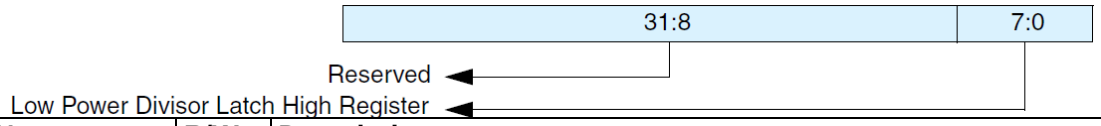
Bits	Name	R/W	Description
31:8	Reserved and read as 0		

Bits	Name	R/W	Description
7:0	LPDLL	R/W	This register makes up the lower 8-bits of a 16-bit, read/write, Low Power Divisor Latch register that contains the baud rate divisor for the UART, which must give a baud rate of 115.2K. This is required for SIR Low Power (minimum pulse width) detection at the receiver. The output low-power baud rate is equal to the serial clock (sclk) frequency divided by sixteen times the value of the baud rate divisor, as follows: Low power baud rate = (serial clock frequency)/(16* divisor) Therefore, a divisor must be selected to give a baud rate of 115.2K. **NOTE:** When the Low Power Divisor Latch registers (LPDLL and LPDLH) are set to 0, the low-power baud clock is disabled and no low-power pulse detection (or any pulse detection) occurs at the receiver. Also, once the LPDLL is set, at least eight clock cycles of the slowest UART clock should be allowed to pass before transmitting or receiving data. **Reset Value:** 0x0

~~11.4.2.14 LPDLH~~

- **Name:** Low Power Divisor Latch High Register
- **Size:** 32 bits
- **Address Offset:** 0x24
- **Read/write access:** read/write

This register is valid only when the UART is configured to have SIR low-power reception capabilities implemented (SIR_LP_RX = Yes). If SIR low-power reception capabilities are not implemented, this register does not exist and reading from this register address returns 0.
If UART_16550_COMPATIBLE = No, then this register can be accessed only when the DLAB bit (LCR[7]) is set and the UART is not busy—that is, USR[0] is 0; otherwise this register can be accessed only when the DLAB bit (LCR[7]) is set.

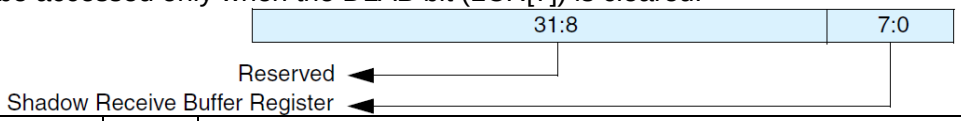


Bits	Name	R/W	Description
31:8	Reserved and read as 0		
7:0	LPDLH	R/W	This register makes up the upper 8-bits of a 16-bit, read/write, Low Power Divisor Latch register that contains the baud rate divisor for the UART, which must give a baud rate of 115.2K. This is required for SIR Low Power (minimum pulse width) detection at the receiver. The output low-power baud rate is equal to the serial clock (sclk) frequency divided by sixteen times the value of the baud rate divisor, as follows: Low power baud rate = (serial clock frequency)/(16* divisor) Therefore, a divisor must be selected to give a baud rate of 115.2K. **NOTE:** When the Low Power Divisor Latch registers (LPDLL and LPDLH) are set to 0, the low-power baud clock is disabled and no low-power pulse detection (or any pulse detection) occurs at the receiver. Also, once the LPDLH is set, at least eight clock cycles of the slowest UART clock should be allowed to pass before transmitting or receiving data. **Reset Value:** 0x0

~~11.4.2.15 SRBR~~

- **Name:** Shadow Receive Buffer Register
- **Size:** 32 bits
- **Address Offset:** 0x30 - 0x6C
- **Read/write access:** read-only

This register is valid only when the UART is configured to have additional shadow registers implemented (SHADOW = YES). If shadow registers are not implemented, this register does not exist and reading from this register address returns 0.
This register can be accessed only when the DLAB bit (LCR[7]) is cleared.

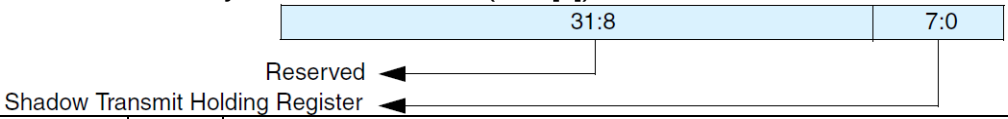


Bits	Name	R/W	Description
31:8	Reserved and read as 0		
7:0	Shadow Receive Buffer Register	R	This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If in non-FIFO mode (FIFO_MODE = NONE) or FIFOs are disabled (FCR[0] set to 0), the data in the RBR must be read before the next data arrives, otherwise it is overwritten, resulting in an overrun error. If in FIFO mode (FIFO_MODE != NONE) and FIFOs are enabled (FCR[0] set to 1), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO are preserved, but any incoming data is lost. An overrun error also occurs. **Reset Value:** 0x0

~~11.4.2.16 STHR~~

- **Name:** Shadow Transmit Holding Register
- **Size:** 32 bits
- **Address Offset:** 0x30 - 0x6C
- **Read/write access:** write

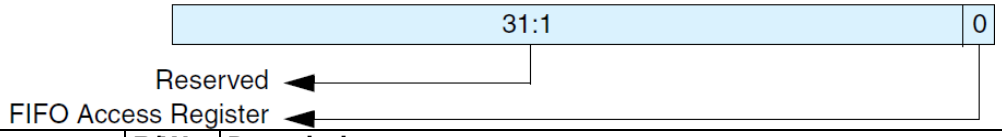
This register is valid only when the UART is configured to have additional shadow registers implemented (SHADOW = YES). If shadow registers are not implemented, this register does not exist, and reading from this register address returns 0.
This register can be accessed only when the DLAB bit (LCR[7]) is cleared.



Bits	Name	R/W	Description
31:8	Reserved and read as 0		
7:0	Shadow Transmit Holding Register	W	This is a shadow register for the THR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If in non-FIFO mode or FIFOs are disabled (FCR[0] set to 0) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If in FIFO mode and FIFOs are enabled (FCR[0] set to 1) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost. **Reset Value:** 0x0

~~11.4.2.17 FAR~~

- **Name:** FIFO Access Register
- **Size:** 32 bits
- **Address Offset:** 0x70
- **Read/write access:** read/write

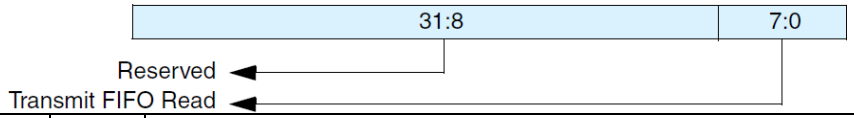


Bits	Name	R/W	Description
31:1	Reserved and read as 0		
0	FIFO Access Register	R/W	~~Writes have no effect when FIFO_ACCESS = No, a~~lways readable. This register is use to enable a FIFO access mode for testing, so that the receive FIFO can be written by the master and the transmit FIFO can be read by the master when FIFOs are implemented and enabled. When FIFOs are not implemented or not enabled it allows the RBR to be written by the master and the THR to be read by the master. ■ 0 – FIFO access mode disabled ■ 1 – FIFO access mode enabled Note, that when the FIFO access mode is enabled/disabled, the control portion of the receive FIFO and transmit FIFO is reset and the FIFOs are treated as empty. **Reset Value:** 0x0

~~11.4.2.18 TFR~~

- **Name:** Transmit FIFO Read
- **Size:** 32 bits
- **Address Offset:** 0x74
- **Read/write access:** read-only

This register is valid only when the UART is configured to have the FIFO access test mode available (FIFO_ACCESS = YES). If not configured, this register does not exist and reading from this register address returns 0.

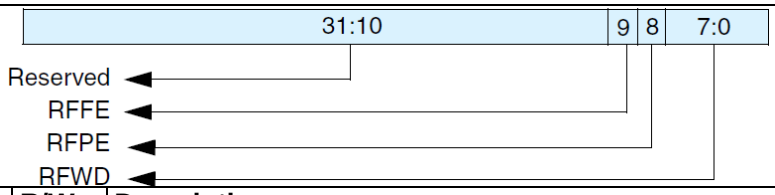


Bits	Name	R/W	Description
31:8	Reserved and read as 0		
7:0	Transmit FIFO Read	R	Transmit FIFO Read. These bits are only valid when FIFO access mode is enabled (FAR[0] is set to 1). When FIFOs are implemented and enabled, reading this register gives the data at the top of the transmit FIFO. Each consecutive read pops the transmit FIFO and gives the next data value that is currently at the top of the FIFO. When FIFOs are not implemented or not enabled, reading this register gives the data in the THR. **Reset Value:** 0x0

~~11.4.2.19 RFW~~

- **Name:** Receive FIFO Write
- **Size:** 32 bits
- **Address Offset:** 0x78
- **Read/write access:** write-only

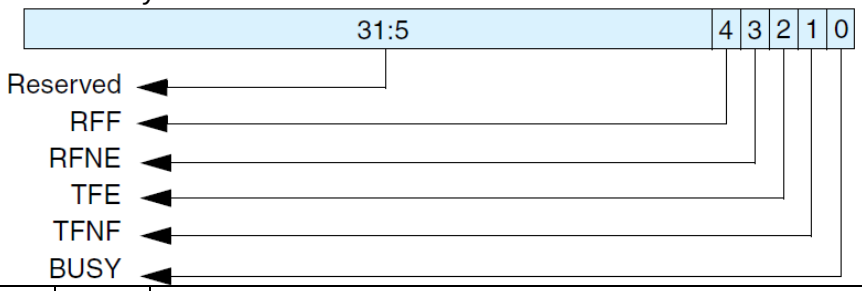
This register is valid only when the UART is configured to have the FIFO access test mode available (FIFO_ACCESS = YES). If not configured, this register does not exist and reading from this register address returns 0.



Bits	Name	R/W	Description
31:10	Reserved and read as 0		
9	RFFE	W	Receive FIFO Framing Error. These bits are only valid when FIFO access mode is enabled (FAR[0] is set to 1). When FIFOs are implemented and enabled, this bit is used to write framing error detection information to the receive FIFO. When FIFOs are not implemented or not enabled, this bit is used to write framing error detection information to the RBR. **Reset Value:** 0x0
8	RFPE	W	Receive FIFO Parity Error. These bits are only valid when FIFO access mode is enabled (FAR[0] is set to 1). When FIFOs are implemented and enabled, this bit is used to write parity error detection information to the receive FIFO. When FIFOs are not implemented or not enabled, this bit is used to write parity error detection information to the RBR. **Reset Value:** 0x0
7:0	RFWD	W	Receive FIFO Write Data. These bits are only valid when FIFO access mode is enabled (FAR[0] is set to 1). When FIFOs are implemented and enabled, the data that is written to the RFWD is pushed into the receive FIFO. Each consecutive write pushes the new data to the next write location in the receive FIFO. When FIFOs are not implemented or not enabled, the data that is written to the RFWD is pushed into the RBR. **Reset Value:** 0x0

11.4.2.20 USR

- **Name:** UART Status Register
- **Size:** 32 bits
- **Address Offset:** 0x7C
- **Read/write access:** read-only



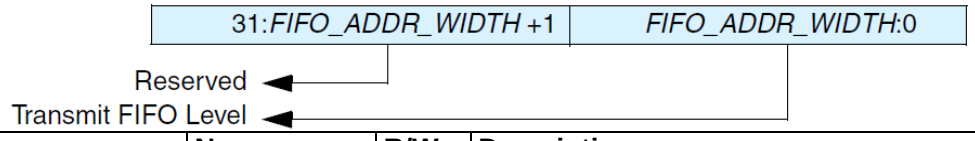
Bits	Name	R/W	Description
31:5	Reserved and read as 0		
4	~~RFF~~	~~R~~	~~Receive FIFO Full. This bit is only valid when FIFO_STAT = YES. This is used to indicate that the receive FIFO is completely full.~~ ~~■ 0 – Receive FIFO not full~~ ~~■ 1 – Receive FIFO Full~~ ~~This bit is cleared when the RX FIFO is no longer full.~~ ~~**Reset Value:** 0x0~~

Bits	Name	R/W	Description
3	~~RFNE~~	~~R~~	~~Receive FIFO Not Empty. This bit is only valid when FIFO_STAT = YES. This is used to indicate that the receive FIFO contains one or more entries.~~ ■ ~~0 – Receive FIFO is empty~~ ■ ~~1 – Receive FIFO is not empty~~ ~~This bit is cleared when the RX FIFO is empty.~~ **~~Reset Value:~~** ~~0x0~~
2	~~TFE~~	~~R~~	~~Transmit FIFO Empty. This bit is only valid when FIFO_STAT = YES. This is used to indicate that the transmit FIFO is completely empty.~~ ■ ~~0 – Transmit FIFO is not empty~~ ■ ~~1 – Transmit FIFO is empty~~ ~~This bit is cleared when the TX FIFO is no longer empty.~~ **~~Reset Value:~~** ~~0x1~~
1	~~TFNF~~	~~R~~	~~Transmit FIFO Not Full. This bit is only valid when FIFO_STAT = YES. This is used to indicate that the transmit FIFO in not full.~~ ■ ~~0 – Transmit FIFO is full~~ ■ ~~1 – Transmit FIFO is not full~~ ~~This bit is cleared when the TX FIFO is full.~~ **~~Reset Value:~~** ~~0x1~~
0	BUSY	R	UART Busy. This bit is valid only when ~~UART_16550_COMPATIBLE = NO and~~ indicates that a serial transfer is in progress; when cleared, indicates that the UART is idle or inactive. ■ 0 – UART is idle or inactive ■ 1 – UART is busy (actively transferring data) This bit will be set to 1 (busy) under any of the following conditions: 1. Transmission in progress on serial interface 2. Transmit data present in THR, when FIFO access mode is not being used ~~(FAR = 0)~~ and the baud divisor is non-zero ({DLH,DLL} does not equal 0) when the divisor latch access bit is 0 (LCR.DLAB = 0) 3. Reception in progress on the interface 4. Receive data present in RBR, when FIFO access mode is not being used ~~(FAR = 0)~~ **NOTE:** It is possible for the UART Busy bit to be cleared even though a new character may have been sent from another device. That is, if the UART has no data in THR and RBR and there is no transmission in progress and a start bit of a new character has just reached the UART. This is due to the fact that a valid start is not seen until the middle of the bit period and this duration is dependent on the baud divisor that has been programmed. ~~If a second system clock has been implemented (CLOCK_MODE = Enabled), the assertion of this bit is also delayed by several cycles of the slower clock.~~ **Reset Value:** 0x0

~~11.4.2.21 TFL~~

- **Name:** Transmit FIFO Level
- **Size:** *FIFO_ADDR_WIDTH* + 1
- **Address Offset:** 0x80
- **Read/write access:** read-only

This register is valid only when the UART is configured to have additional FIFO status registers implemented (FIFO_STAT = YES). If status registers are not implemented, this register does not exist and reading from this register address returns 0.



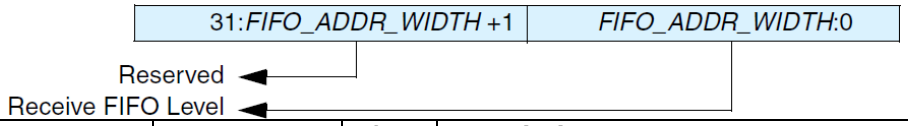
Bits	Name	R/W	Description

Bits	Name	R/W	Description
31:*FIFO_ADDR_WIDTH* + 1	Reserved and read as 0		
FIFO_ADDR_WIDTH:0	Transmit FIFO Level	R	Transmit FIFO Level. This is indicates the number of data entries in the transmit FIFO. **Reset Value:** 0x0

~~11.4.2.22 RFL~~

- **Name:** Receive FIFO Level
- **Size:** *FIFO_ADDR_WIDTH* + 1
- **Address Offset:** 0x84
- **Read/write access:** read-only

This register is valid only when the UART is configured to have additional FIFO status registers implemented (FIFO_STAT = YES). If status registers are not implemented, this register does not exist and reading from this register address returns 0.

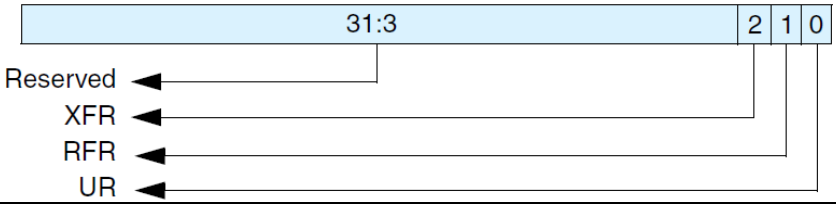


Bits	Name	R/W	Description
31:*FIFO_ADDR_WIDTH* + 1	Reserved and read as 0		
FIFO_ADDR_WIDTH:0	Receive FIFO Level	R	Receive FIFO Level. This is indicates the number of data entries in the receive FIFO. **Reset Value:** 0x0

~~11.4.2.23 SRR~~

- **Name:** Software Reset Register
- **Size:** 32 bits
- **Address Offset:** 0x88
- **Read/write access:** write-only

This register is valid only when the UART is configured to have additional shadow registers implemented (SHADOW = YES). If shadow registers are not implemented, this register does not exist and reading from this register address returns 0.



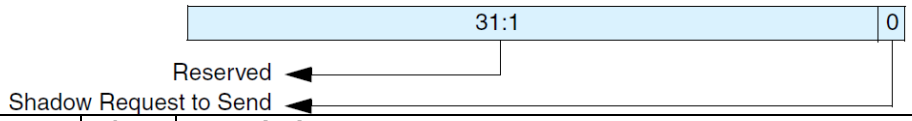
Bits	Name	R/W	Description
31:3	Reserved and read as 0		
2	XFR	W	XMIT FIFO Reset. This is a shadow register for the XMIT FIFO Reset bit (FCR[2]). This can be used to remove the burden on software having to store previously written FCR values (which are pretty static) just to reset the transmit FIFO. This resets the control portion of the transmit FIFO and treats the FIFO as empty. This also de-asserts the DMA TX request and single signals when additional DMA handshaking signals are selected (DMA_EXTRA = YES). Note that this bit is 'self-clearing'. It is not necessary to clear this bit. **Reset Value:** 0x0 **Dependencies:** Writes have no effect when FIFO_MODE = None.

Bits	Name	R/W	Description
1	RFR	W	RCVR FIFO Reset. This is a shadow register for the RCVR FIFO Reset bit (FCR[1]). This can be used to remove the burden on software having to store previously written FCR values (which are pretty static) just to reset the receive FIFO This resets the control portion of the receive FIFO and treats the FIFO as empty. This also de-asserts the DMA RX request and single signals when additional DMA handshaking signals are selected (DMA_EXTRA = YES). Note that this bit is 'self-clearing'. It is not necessary to clear this bit. **Reset Value:** 0x0 **Dependencies:** Writes have no effect when FIFO_MODE = None.
0	UR	W	UART Reset. This asynchronously resets the UART and synchronously removes the reset assertion. For a two clock implementation both pclk and sclk domains are reset. **Reset Value:** 0x0

~~11.4.2.24 SRTS~~

- **Name:** Shadow Request to Send
- **Size:** 32 bits
- **Address Offset:** 0x8C
- **Read/write access:** read/write

This register is valid only when the UART is configured to have additional shadow registers implemented (SHADOW = YES). If shadow registers are not implemented, this register does not exist and reading from this register address returns 0.

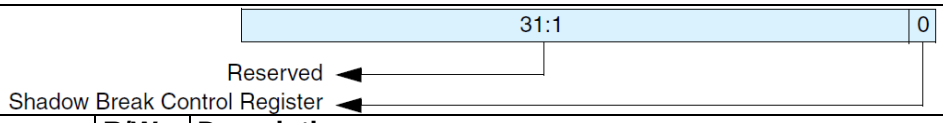


Bits	Name	R/W	Description
31:1	Reserved and read as 0		
0	Shadow Request to Send	R/W	Shadow Request to Send. This is a shadow register for the RTS bit (MCR[1]), this can be used to remove the burden of having to performing a read-modify-write on the MCR. This is used to directly control the Request to Send (rts_n) output. The Request To Send (rts_n) output is used to inform the modem or data set that the UART is ready to exchange data. When Auto RTS Flow Control is not enabled (MCR[5] = 0), the rts_n signal is set low by programming MCR[1] (RTS) to a high. In Auto Flow Control, AFCE_MODE = Enabled and active (MCR[5] = 1) and FIFOs enable (FCR[0] = 1), the rts_n output is controlled in the same way, but is also gated with the receiver FIFO threshold trigger (rts_n is inactive high when above the threshold) only when RTC Flow Trigger is disabled; otherwise it is gated by the receiver FIFO almost-full trigger, where “almost full” refers to two available slots in the FIFO (rts_n is inactive high when above the threshold). Note that in Loopback mode (MCR[4] = 1), the rts_n output is held inactive-high while the value of this location is internally looped back to an input. **Reset Value:** 0x0

~~11.4.2.25 SBCR~~

- **Name:** Shadow Break Control Register
- **Size:** 32 bits
- **Address Offset:** 0x90
- **Read/write access:** read/write

This register is valid only when the UART is configured to have additional shadow registers implemented (SHADOW = YES). If shadow registers are not implemented, this register does not exist and reading from this register address returns 0.

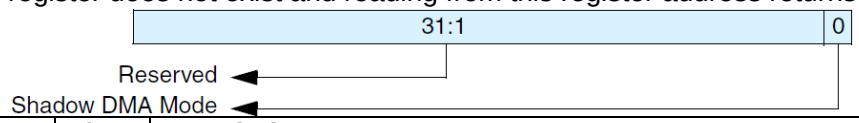


Bits	Name	R/W	Description
31:1	Reserved and read as 0		
0	Shadow Break Control Register	R/W	Shadow Break Control Bit. This is a shadow register for the Break bit (LCR[6]), this can be used to remove the burden of having to performing a read modify write on the LCR. This is used to cause a break condition to be transmitted to the receiving device. If set to 1, the serial output is forced to the spacing (logic 0) state. When not in Loopback Mode, as determined by MCR[4], the sout line is forced low until the Break bit is cleared. If SIR_MODE = Enabled and active (MCR[6] = 1) the sir_out_n line is continuously pulsed. When in Loopback Mode, the break condition is internally looped back to the receiver. **Reset Value:** 0x0

~~11.4.2.26 SDMAM~~

- **Name:** Shadow DMA Mode
- **Size:** 32 bits
- **Address Offset:** 0x94
- **Read/write access:** read/write

This register is valid only when the UART is configured to have additional FIFO registers implemented (FIFO_MODE != None) and additional shadow registers implemented (SHADOW = YES). If these registers are not implemented, this register does not exist and reading from this register address returns 0.

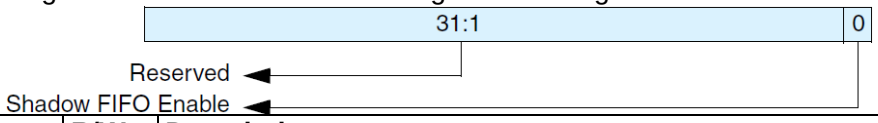


Bits	Name	R/W	Description
31:1	Reserved and read as 0		
0	Shadow DMA Mode	R/W	Shadow DMA Mode. This is a shadow register for the DMA mode bit (FCR[3]). This can be used to remove the burden of having to store the previously written value to the FCR in memory and having to mask this value so that only the DMA Mode bit gets updated. This determines the DMA signalling mode used for the dma_tx_req_n and dma_rx_req_n output signals when additional DMA handshaking signals are not selected (DMA_EXTRA = NO). ■ 0 – mode 0 ■ 1 – mode 1 **Reset Value:** 0x0

~~11.4.2.27 SFE~~

- **Name:** Shadow FIFO Enable
- **Size:** 32 bits
- **Address Offset:** 0x98
- **Read/write access:** read/write

This register is valid only when the UART is configured to have additional FIFO registers implemented (FIFO_MODE != None) and additional shadow registers implemented (SHADOW = YES). If these registers are not implemented, this register does not exist and reading from this register address returns 0.



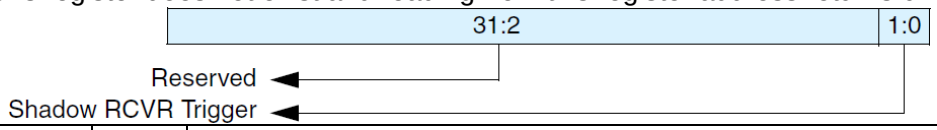
Bits	Name	R/W	Description
31:1	Reserved and read as 0		

Bits	Name	R/W	Description
0	Shadow FIFO Enable	R/W	Shadow FIFO Enable. This is a shadow register for the FIFO enable bit (FCR[0]). This can be used to remove the burden of having to store the previously written value to the FCR in memory and having to mask this value so that only the FIFO enable bit gets updated.This enables/disables the transmit (XMIT) and receive (RCVR) FIFOs. If this bit is set to 0 (disabled) after being enabled then both the XMIT and RCVR controller portion of FIFOs are reset. **Reset Value:** 0x0

~~11.4.2.28 SRT~~

- **Name:** Shadow RCVR Trigger
- **Size:** 32 bits
- **Address Offset:** 0x9C
- **Read/write access:** read/write

This register is valid only when the UART is configured to have additional FIFO registers implemented (FIFO_MODE != None) and additional shadow registers implemented (SHADOW = YES). If these registers are not implemented, this register does not exist and reading from this register address returns 0.

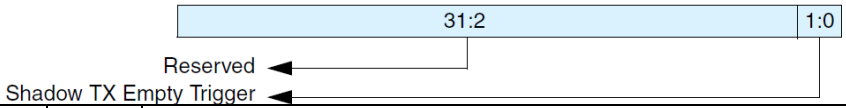


Bits	Name	R/W	Description
31:2	Reserved and read as 0		
1:0	Shadow RCVR Trigger	R/W	Shadow RCVR Trigger. This is a shadow register for the RCVR trigger bits (FCR[7:6]). This can be used to remove the burden of having to store the previously written value to the FCR in memory and having to mask this value so that only the RCVR trigger bit gets updated. This is used to select the trigger level in the receiver FIFO at which the Received Data Available Interrupt is generated. It also determines when the dma_rx_req_n signal is asserted when DMA Mode (FCR[3]) = 1. The following trigger levels are supported: ■ 00 – 1 character in the FIFO ■ 01 – FIFO ¼ full ■ 10 – FIFO ½ full ■ 11 – FIFO 2 less than full **Reset Value:** 0x0

~~11.4.2.29 STET~~

- **Name:** Shadow TX Empty Trigger
- **Size:** 32 bits
- **Address Offset:** 0xA0
- **Read/write access:** read/write

This register is valid only when the UART is configured to have FIFOs implemented (FIFO_MODE != NONE) and THRE interrupt support implemented (THRE_MODE_USER = Enabled) and additional shadow registers implemented (SHADOW = YES). If FIFOs are not implemented or THRE interrupt support is not implemented or shadow registers are not implemented, this register does not exist and reading from this register address returns 0.

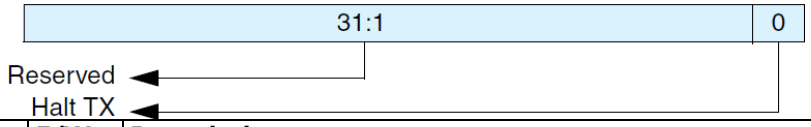


Bits	Name	R/W	Description
31:2	Reserved and read as 0		

Bits	Name	R/W	Description
1:0	Shadow TX Empty Trigger	R/W	Shadow TX Empty Trigger. This is a shadow register for the TX empty trigger bits (FCR[5:4]). This can be used to remove the burden of having to store the previously written value to the FCR in memory and having to mask this value so that only the TX empty trigger bit gets updated. This is used to select the empty threshold level at which the THRE Interrupts are generated when the mode is active. The following trigger levels are supported: ■ 00 – FIFO empty ■ 01 – 2 characters in the FIFO ■ 10 – FIFO ¼ full ■ 11 – FIFO ½ full **Reset Value:** 0x0 **Dependencies:** Writes have no effect when THRE_MODE_USER = Disabled.

11.4.2.30 HTX

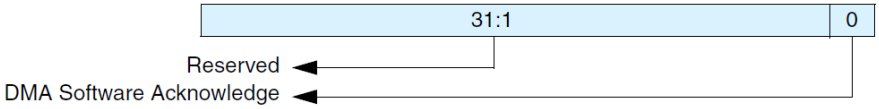
- **Name:** Halt TX
- **Size:** 32 bits
- **Address Offset:** 0xA4
- **Read/write access:** read/write



Bits	Name	R/W	Description
31:1	Reserved and read as 0		
0	Halt TX	R/W	This register is use to halt transmissions for testing, so that the transmit FIFO can be filled by the master when FIFOs are implemented and enabled. ■ 0 – Halt TX disabled ■ 1 – Halt TX enabled Note, if FIFOs are implemented and not enabled, the setting of the halt TX register has no effect on operation. **Reset Value:** 0x0 ~~**Dependencies:** Writes have no effect when FIFO_MODE = None.~~

~~11.4.2.31 DMASA~~

- **Name:** DMA Software Acknowledge
- **Size:** 32 bits
- **Address Offset:** 0xA8
- **Read/write access:** write

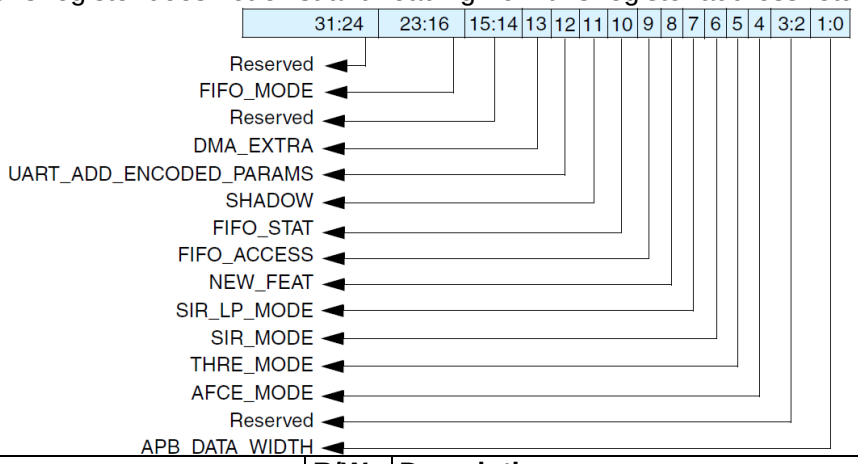


Bits	Name	R/W	Description
31:1	Reserved and read as 0		
0	DMA Software Acknowledge	W	This register is use to perform a DMA software acknowledge if a transfer needs to be terminated due to an error condition. For example, if the DMA disables the channel, then the UART should clear its request. This causes the TX request, TX single, RX request and RX single signals to de-assert. Note that this bit is 'self-clearing'. It is not necessary to clear this bit. **Reset Value:** 0x0 ~~**Dependencies:** Writes have no effect when DMA_EXTRA = No.~~

~~11.4.2.32 CPR~~

- **Name:** Component Parameter Register
- **Size:** 32 bits
- **Address Offset:** 0xF4
- **Read/write access:** read-only

This register is valid only when UART_ADD_ENCODED_PARAMS = 1. If the UART_ADD_ENCODED_PARAMS parameter is not set, this register does not exist and reading from this register address returns 0.



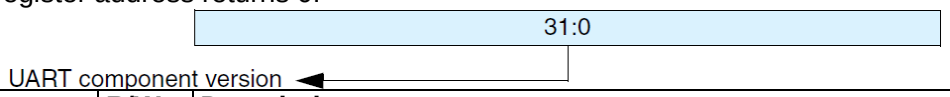
Bits	Name	R/W	Description
31:24	Reserved and read as 0		
23:16	FIFO_MODE	R	0x00 = 0 0x01 = 16 0x02 = 32 to 0x80 = 2048 0x81- 0xff = reserved
15:14	Reserved and read as 0	R	
13	DMA_EXTRA	R	0 – FALSE 1 – TRUE
12	UART_ADD_ENCODED_PARAMS	R	0 – FALSE 1 – TRUE
11	SHADOW	R	0 – FALSE 1 – TRUE
10	FIFO_STAT	R	0 – FALSE 1 – TRUE
9	FIFO_ACCESS	R	0 – FALSE 1 – TRUE
8	ADDITIONAL_FEAT	R	0 – FALSE 1 – TRUE
7	SIR_LP_MODE	R	0 – FALSE 1 – TRUE
6	SIR_MODE	R	0 – FALSE 1 – TRUE
5	THRE_MODE	R	0 – FALSE 1 – TRUE
4	AFCE_MODE	R	0 – FALSE 1 – TRUE
3:2	Reserved and read as 0		
1:0	APB_DATA_WIDTH	R	00 – 8 bits 01 – 16 bits 10 – 32 bits 11 – reserved

~~11.4.2.33 UCV~~

- **Name:** UART Component Version
- **Size:** 32 bits

- **Address Offset:** 0xF8
- **Read/write access:** read-only

This register is valid only when the UART is configured to have additional features implemented (ADDITIONAL_FEATURES = YES). If additional features are not implemented, this register does not exist and reading from this register address returns 0.

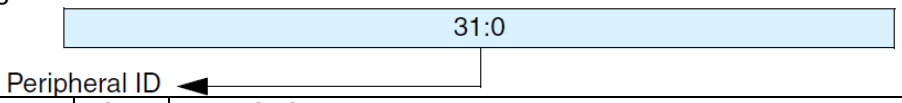


Bits	Name	R/W	Description
31:0	UART Component Version	R	ASCII value for each number in the version, followed by *. For example 32_30_31_2A represents the version 2.01* **Reset Value:** See the releases table in the AMBA 2 release notes.

~~11.4.2.34 CTR~~

- **Name:** Component Type Register
- **Size:** 32 bits
- **Address Offset:** 0xFC
- **Read/write access:** read-only

This register is valid only when the UART is configured to have additional features implemented (ADDITIONAL_FEATURES = YES). If additional features are not implemented, this register does not exist and reading from this register address returns 0.



Bits	Name	R/W	Description
31:0	Peripheral ID	R	This register contains the peripherals identification code. **Reset Value:** 0x44570110

12 I2C

12.1 I2C Overview

The I2C is a programmable control bus that provides support for the communications link between integrated circuits in a system. It is a simple two-wire bus with a software-defined protocol for system control, which is used in temperature sensors and voltage level translators to EEPROMs, general-purpose I/O, A/D and D/A converters, CODECs, and many types of microprocessors.

~~12.1.1 I2C Features~~

- ~~Two-wire I2C serial interface – consists of a serial data line (SDA) and a serial clock (SCL)~~
- ~~Three speeds:~~
  - ~~Standard mode (0 to 100 Kb/s)~~
  - ~~Fast mode (≤400 Kb/s)~~
  - ~~High-speed mode (≤3.4 Mb/s)~~
- ~~Clock synchronization~~
- ~~Master OR slave I2C operation~~
- ~~7- or 10-bit addressing~~
- ~~7- or 10-bit combined format transfers~~
- ~~Bulk transmit mode~~
- ~~Ignores CBUS addresses (an older ancestor of I2C that used to share the I2C bus)~~
- ~~Transmit and receive buffers~~
- ~~Interrupt or polled-mode operation~~
- ~~Handles Bit and Byte waiting at all bus speeds~~
- ~~Simple software interface consistent with DesignWare APB peripherals~~
- ~~Component parameters for configurable software driver support~~
- ~~DMA handshaking interface compatible with the DW_ahb_dmac handshaking interface~~
- ~~Programmable SDA hold time (tHD;DAT)~~

The following is not supported:

- Fast Mode Plus speed is untested and therefore Fast Mode Plus is unsupported at this time. However, if the dividers are set to the proper values, the Fast Mode Plus speed should work properly.

12.2 I2C Function Description

The I2C bus is a two-wire serial interface, consisting of a serial data line (SDA) and a serial clock (SCL). These wires carry information between the devices connected to the bus. Each device is recognized by a unique address and can operate as either a “transmitter” or “receiver,” depending on the function of the device. Devices can also be considered as masters or slaves when performing data transfers. A master is a device that initiates a

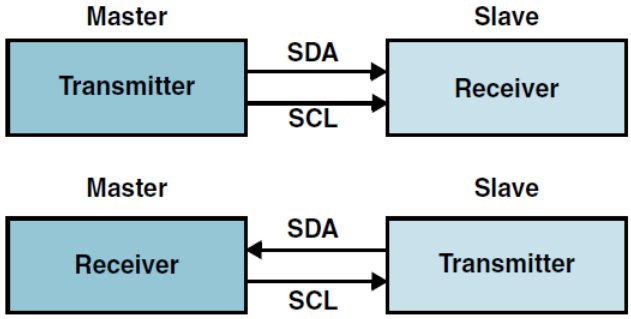
data transfer on the bus and generates the clock signals to permit that transfer. At that time, any device addressed is considered a slave.

12.2.1 I2C Bus Terms

The following terms relate to how the role of the I2C device and how it interacts with other I2C devices on the bus.

- **Transmitter** – the device that sends data to the bus. A 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** – the device that receives data from the 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** -– the component that initializes a transfer (START command), generates the clock (SCL) signal and terminates the transfer (STOP command). A master can be either a transmitter or a receiver.
- **Slave** – the device addressed by the master. A slave can be either receiver or transmitter.
- **Multi-master** – the ability for more than one master to co-exist on the bus at the same time without collision or data loss.
- **Arbitration** – the predefined procedure that authorizes only one master at a time to take control of the bus.
- **Synchronization** – the predefined procedure that synchronizes the clock signals provided by two or more masters.
- **SDA** – data signal line (Serial DAta)
- **SCL** – clock signal line (Serial CLock)

Master/Slave and Transmitter/Receiver Relationships



- **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.
- **STOP** – data transfer is terminated by a STOP condition. This 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

12.2.2 I2C Behavior

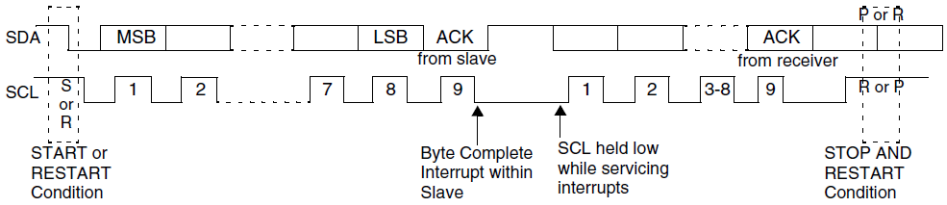
The I2C can be controlled via software to be ~~either~~:

- An I2C master only, communicating with other I2C slaves; OR
- ~~An I2C slave only, communicating with one more I2C masters.~~

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. As mentioned previously, 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 one 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 (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. This behavior is illustrated in following figure.



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.

12.2.2.1 START and STOP Generation

When operating as an I2C master, putting data into the transmit FIFO causes the I2C to generate a START condition on the I2C bus. If the IC_EMPTYFIFO_HOLD_MASTER_EN parameteris set to 0, allowing the transmit FIFO to empty causes the I2C to generate a STOP condition on the I2C bus. If IC_EMPTYFIFO_HOLD_MASTER_EN is set to 1, then writing a 1 to IC_DATA_CMD[9] causes the I2C to generate a STOP condition on the I2C bus; a STOP condition is not issued if this bit is not set, even if the transmit FIFO is empty.

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

12.2.2.2 Combined Formats

The I2C supports mixed read and write combined format transactions in both 7-bit and 10-bit addressing modes.

The I2C 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, IC_CON.IC_RESTART_EN should be set to 1. With this value set and operating as a master, when the I2C completes an I2C 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 I2C transfer completes—depending on the value of IC_EMPTYFIFO_HOLD_MASTER_EN:

- Either a STOP is issued or,
- IC_DATA_CMD[9] is checked *and*:
  - If set to 1, a STOP bit is issued.
  - If set to 0, the SCL is held low until the next command is written to the transmit FIFO.

12.2.3 Tx FIFO Management and START, STOP and RESTART Generation

When operating as a master, the I2C component supports two modes of Tx FIFO management.
You use the IC_EMPTYFIFO_HOLD_MASTER_EN parameter to select between these two modes:

- IC_EMPTYFIFO_HOLD_MASTER_EN equals 0
- IC_EMPTYFIFO_HOLD_MASTER_EN equals 1

12.2.3.1 Tx FIFO Management ~~When IC_EMPTYFIFO_HOLD_MASTER_EN = 0~~

When the value of IC_EMPTYFIFO_HOLD_MASTER_EN is 0, the component generates a STOP on the bus whenever the Tx FIFO becomes empty. If RESTART generation capability is enabled, the component generates a RESTART when the direction of the transfer in the Tx FIFO commands changes from Read to Write or vice-versa; if RESTART is not enabled, a STOP followed by a START is generated in this situation.

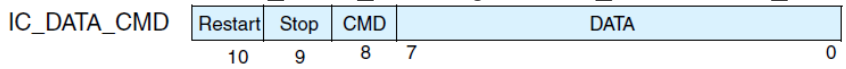
The following figure shows the bits in the IC_DATA_CMD register ~~if IC_EMPTYFIFO_HOLD_MASTER_EN = 0~~.

DATA –Read/Write field; data retrieved from slave is read from this field; data to be sent to slave is written to this field.
CMD –Write-only field; this bit determines whether transfer to be carried out is Read (CMD=1) or Write (CMD=0)

~~12.2.3.2 Tx FIFO Management When IC_EMPTYFIFO_HOLD_MASTER_EN = 1~~

~~When the value of IC_EMPTYFIFO_HOLD_MASTER_EN is 1, the component does not generate a STOP if the Tx FIFO becomes empty; in this situation the component holds the SCL line low, stalling the bus until a new entry is available in the Tx FIFO. A STOP condition is generated only when the user specifically requests it by setting bit 9 (Stop bit) of the command written to IC_DATA_CMD register.~~

~~The following figure shows the bits in the IC_DATA_CMD register if IC_EMPTYFIFO_HOLD_MASTER_EN = 1.~~



DATA –Read/Write field; data retrieved from slave is read from this field; data to be sent to slave is written to this field
CMD –Write-only field; this bit determines whether transfer to be carried out is Read (CMD=1) or Write (CMD=0)
Stop –Write-only field; this bit determines whether STOP is generated after data byte is sent or received
Restart – Write-only field; this bit determines whether RESTART (or STOP followed by START in case of restart capability is not enabled) is generated before data byte is sent of received

12.2.4 Operation Modes

This section provides information on operation modes.

~~12.2.4.1 Slave Mode Operation~~

This section discusses slave mode procedures.

~~12.2.4.1.1 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 IC_ENABLE register.
2. Write to the IC_SAR register (bits 9:0) to set the slave address. This is the address to which the I2C responds.
3. Write to the IC_CON register to specify which type of addressing is supported (7- or 10-bit by setting bit 3). Enable the I2C in slave-only mode by writing a ‘0’ into bit 6 (IC_SLAVE_DISABLE) and a ‘0’ to bit 0 (MASTER_MODE).
4. Enable the I2C by writing a ‘1’ in bit 0 of the IC_ENABLE register.

~~12.2.4.1.2 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 an I2C transfer with an address that matches the slave address in the IC_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 IC_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 IC_INTR_MASK[5] register (M_RD_REQ bit field) being set to 0, then it is recommended that a hardware and/or software timing routine be used to instruct the CPU to perform periodic reads of the IC_RAW_INTR_STAT register.

   a. Reads that indicate IC_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 kb/s, the timing interval is 25us.

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 IC_RAW_INTR_STAT register) to flush the old data from the TX FIFO.
If the TX_ABRT interrupt has been masked, due to of IC_INTR_MASK[6] register (M_TX_ABRT bit field) being set to 0, then it is recommended that re-using the timing routine (described in the previous step), or a similar one, be used to read the IC_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 IC_RAW_INTR_STAT[5] register.
5. Software writes to the IC_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 IC_RAW_INTR_STAT register before proceeding.
If the RD_REQ and/or TX_ABRT interrupts have been masked, then clearing of the IC_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 I2C releases the SCL and transmits the byte.
8. The master may hold the I2C bus by issuing a RESTART condition or release the bus by issuing a STOP condition.

~~12.2.4.1.3 Slave-Receiver Operation for a Single Byte~~

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

1. The other I2C master device initiates an I2C transfer with an address that matches the I2C's slave address in the IC_SAR register.
2. The I2C acknowledges the sent address and recognizes the direction of the transfer to indicate that the I2C is acting as a slave-receiver.
3. I2C receives the transmitted byte and places it in the receive buffer.
4. I2C asserts the RX_FULL interrupt (IC_RAW_INTR_STAT[2] register).
If the RX_FULL interrupt has been masked, due to setting IC_INTR_MASK[2] register to 0 or setting IC_TX_TL to a value larger than 0, then it is recommended that a timing routine (described in "Slave-Transmitter Operation for a Single Byte" on page 48) be implemented for periodic reads of the IC_STATUS register. Reads of the IC_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 IC_DATA_CMD register (bits 7:0).
6. The other master device may hold the I2C bus by issuing a RESTART condition or release the bus by issuing a STOP condition.

~~12.2.4.1.4 Slave-Transfer Operation For Bulk Transfers~~

In the standard I2C protocol, all transactions are single byte transactions and the programmer responds to a remote master read request by writing one byte into the slave's 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 one entry placed into the slave-transmitter's TX FIFO. I2C 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 one entry placed in the TX FIFO.

This mode only occurs when I2C 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's TX FIFO, the I2C holds the I2C 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 IC_INTR_STAT register being set to 0, then it is recommended that a timing routine be used to activate periodic reads of the IC_RAW_INTR_STAT register. Reads of IC_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.

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 you 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 I2C 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 I2C 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 I2C 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 the I2C 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's 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 is cleared at that time.

12.2.4.2 Master Mode Operation

This section discusses master mode procedures.

12.2.4.2.1 Initial Configuration

The initial configuration procedure for Master Mode Operation depends on the configuration parameter I2C_DYNAMIC_TAR_UPDATE. When set to "Yes" (1), the target address and address format can be changed dynamically without having to disable I2C. This parameter only applies to when I2C is acting as a master because the slave requires the component to be disabled before any changes can be made to the address.

The procedures are very similar and are only different with regard to where the IC_10BITADDR_MASTER bit is set (either bit 4 of IC_CON register or bit 12 of IC_TAR register).

~~1). I2C_DYNAMIC_TAR_UPDATE = 0~~

To use the I2C as a master ~~when the I2C_DYNAMIC_TAR_UPDATE configuration parameter is set to "No" (0)~~, perform the following steps:

1. Disable the I2C by writing 0 to bit 0 of the IC_ENABLE register.
2. Write to the IC_CON register to set the maximum speed mode supported (bits 2:1) and the desired speed of the I2C master-initiated transfers, either 7-bit or 10-bit addressing (bit 4). Ensure that bit 6 (IC_SLAVE_DISABLE) is written with a '1' and bit 0 (MASTER_MODE) is written with a '1'.
3. Write to the IC_TAR register the address of the I2C device to be addressed (bits 9:0). This register also indicates whether a General Call or a START BYTE command is going to be performed by I2C.
4. *Only applicable for high-speed mode transfers*. Write to the IC_HS_MADDR register the desired master code for the I2C. The master code is programmer-defined.
5. Enable the I2C by writing a 1 to bit 0 of the IC_ENABLE register.
6. Now write transfer direction and data to be sent to the IC_DATA_CMD register. If the IC_DATA_CMD register is written before the I2C is enabled, the data and commands are lost as the buffers are kept cleared when I2C is disabled.

This step generates the START condition and the address byte on the I2C. Once I2C is enabled and there is data in the TX FIFO, I2C starts reading the data.

~~2). I2C_DYNAMIC_TAR_UPDATE = 1~~

~~To use the I2C as a master when the I2C_DYNAMIC_TAR_UPDATE configuration parameter is set to "Yes" (1), perform the following steps:~~

~~1. Disable the I2C by writing 0 to bit 0 of the IC_ENABLE register.~~
~~2. Write to the IC_CON register to set the maximum speed mode supported for slave operation (bits 2:1) and to specify whether the I2C starts its transfers in 7/10 bit addressing mode when the device is a slave (bit 3).~~
~~3. Write to the IC_TAR register the address of the I2C device to be addressed. It also indicates whether a General Call or a START BYTE command is going to be performed by I2C. The desired speed of the I2C master-initiated transfers, either 7-bit or 10-bit addressing, is controlled by the IC_10BITADDR_MASTER bit field (bit 12).~~

4. *Only applicable for high-speed mode transfers*. Write to the IC_HS_MADDR register the desired master code for the I2C. The master code is programmer-defined.
5. Enable the I2C by writing a 1 to bit 0 of the IC_ENABLE register.
6. Now write the transfer direction and data to be sent to the IC_DATA_CMD register. If the IC_DATA_CMD register is written before the I2C is enabled, the data and commands are lost as the buffers are kept cleared when I2C is not enabled.

12.2.4.2.2 Dynamic IC_TAR or IC_10BITADDR_MASTER Update

The I2C supports dynamic updating of the IC_TAR (bits 9:0) and IC_10BITADDR_MASTER (bit 12) bit fields of the IC_TAR register. In order to perform a dynamic update of the IC_TAR register, the I2C_DYNAMIC_TAR_UPDATE configuration parameter must be set to "Yes" (1). You can dynamically write to the IC_TAR register provided the following conditions are met:

1. I2C is not enabled (IC_ENABLE[0]=0);
   OR
2. I2C is enabled (IC_ENABLE[0]=1); AND
   I2C is NOT engaged in any Master (tx, rx) operation (IC_STATUS[5]=0); AND
   I2C is enabled to operate in Master mode (IC_CON[0]=1); AND
   there are NO entries in the TX FIFO (IC_STATUS[2]=1)

12.2.4.2.3 Master Transmit and Master Receive

The I2C 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 I2C Rx/Tx Data Buffer and Command Register (IC_DATA_CMD). The *CMD* bit [8] should be written to 0 for I2C write operations. Subsequently, a read command may be issued by writing "don't cares" to the lower byte of the IC_DATA_CMD register, and a 1 should be written to the *CMD* bit. The I2C master continues to initiate transfers as long as there are commands present in the transmit FIFO. If the transmit FIFO becomes empty—depending on the value of IC_EMPTYFIFO_HOLD_MASTER_EN, the master either inserts a STOP condition after completing the current transfers, or it checks to see if IC_DATA_CMD[9] is set to 1.

- If set to 1, it issues a STOP condition after completing the current transfer.
- If set to 0, it holds SCL low until next command is written to the transmit FIFO.

12.2.4.3 Disabling I2C

The register IC_ENABLE_STATUS is added to allow software to unambiguously determine when the hardware has completely shutdown in response to bit 0 of the IC_ENABLE register being set from 1 to 0. Only one register is required to be monitored, as opposed to monitoring two registers (IC_STATUS and IC_RAW_INTR_STAT) which is a requirement for I2C versions 1.05a or earlier.

12.2.4.3.1 Procedure

1. Define a timer interval (ti2c_poll) equal to the 10 times the signaling period for the highest I2C transfer speed used in the system and supported by I2C. For example, if the highest I2C transfer mode is 400 kb/s, then this ti2c_poll is 25us.
2. Define a maximum time-out parameter, MAX_T_POLL_COUNT, such that if any repeated polling operation exceeds this maximum value, an error is reported.
3. Execute a blocking thread/process/function that prevents any further I2C master transactions to be started by software, but allows any pending transfers to be completed.
4. The variable POLL_COUNT is initialized to zero.
5. Set bit 0 of the IC_ENABLE register to 0.
6. Read the IC_ENABLE_STATUS register and test the IC_EN bit (bit 0). Increment POLL_COUNT by one. If POLL_COUNT >= MAX_T_POLL_COUNT, exit with the relevant error code.
7. If IC_ENABLE_STATUS[0] is 1, then sleep for ti2c_poll and proceed to the previous step. Otherwise, exit with a relevant success code.

12.2.4.4 Aborting I2C Transfers

The ABORT control bit of the IC_ENABLE register allows the software to relinquish the I2C bus before completing the issued transfer commands from the Tx FIFO. In response to an ABORT request, the controller issues the STOP condition over the I2C bus, followed by Tx FIFO flush. Aborting the transfer is allowed only in master mode of operation.

12.2.4.4.1 Procedure

1. Stop filling the Tx FIFO (IC_DATA_CMD) with new commands.
2. When operating in DMA mode, disable the transmit DMA by setting TDMAE to 0.
3. Set bit 1 of the IC_ENABLE register (ABORT) to 1.
4. Wait for the M_TX_ABRT interrupt.
5. Read the IC_TX_ABRT_SOURCE register to identify the source as ABRT_USER_ABRT.

12.2.5 Spike Suppression

The I2C contains programmable spike suppression logic that match requirements imposed by the *I2C Bus Specification* for SS/FS (tSP, Table 4) and HS (tSP, Table 6) modes.

This logic is based on counters that monitor the input signals (SCL and SDA), checking if they remain stable for a predetermined amount of ic_clk cycles before they are sampled internally. There is one separate counter for each signal (SCL and SDA). The number of ic_clk cycles can be programmed by the user and should be calculated taking into account the frequency of ic_clk and the relevant spike length specification.

The *I2C Bus Specification* calls for different maximum spike lengths according to the operating mode—50 ns for SS and FS; 10 ns for HS—so two registers are required to store the values needed for each case:

- Register IC_FS_SPKLEN holds the maximum spike length for SS and FS modes
- Register IC_HS_SPKLEN holds the maximum spike value for HS mode.

12.2.6 IC_CLK Frequency Configuration

When the I2C is configured as a master, the *CNT registers must be set before any I2C bus transaction can take place in order to ensure proper I/O timing. The *CNT registers are:

- IC_SS_SCL_HCNT
- IC_SS_SCL_LCNT
- IC_FS_SCL_HCNT
- IC_FS_SCL_LCNT
- IC_HS_SCL_HCNT
- IC_HS_SCL_LCNT

12.2.7 SDA Hold Time

The I2C protocol specification requires 300ns of hold time on the SDA signal (tHD;DAT) in standard and fast speed modes, and a hold time long enough to bridge the undefined part between logic 1 and logic 0 of the falling edge of SCL in high speed mode.

Board delays on the SCL and SDA signals can mean that the hold-time requirement is met at the I2C master, but not at the I2C slave (or vice-versa). As each application will encounter differing board delays, the I2C contains a software programmable register (IC_SDA_HOLD) to enable dynamic adjustment of the SDA hold-time.

The IC_SDA_HOLD register can be used to alter the timing of the generated SDA (ic_data_oe) signal by the I2C. Each value in the IC_SDA_HOLD register represents a unit of one ic_clk period.

When the I2C is operating in Master Mode, the minimum tHD:DAT timing is one ic_clk period. Therefore even when IC_SDA_HOLD has a value of zero, the I2C will drive SDA (ic_data_oe) one ic_clk cycle after driving SCL (ic_clk_oe) to logic 0. For all other values of IC_SDA_HOLD, the following is true:

- Drive on SDA (ic_data_oe) occurs *IC_SDA_HOLD* ic_clk cycles after driving SCL (ic_clk_oe) to logic 0

When the I2C is operating in Slave Mode, the minimum tHD:DAT timing is *SPKLEN* + 7 ic_clk periods, where SPKLEN is:

- IC_FS_SPKLEN if the component is operating in SS or FS
- IC_HS_SPKLEN if the component is operating in HS

This delay allows for synchronization and spike suppression on the SCL (ic_clk_in_a) sample. Therefore, even when IC_SDA_HOLD has a value less than *SPKLEN* + 7, the I2C drives SDA (ic_data_oe) *SPKLEN* + 7 ic_clk cycles after SCL (ic_clk_in) has transitioned to logic 0. For all other values of IC_SDA_HOLD, the following is true:

- Drive on SDA (ic_data_oe) occurs *IC_SDA_HOLD* ic_clk cycles after SCL (ic_clk_in_a) has transitioned to logic 0

If different SDA hold times are required for different speed modes, the IC_SDA_HOLD register must be reprogrammed when the speed mode is being changed. The IC_SDA_HOLD register can be programmed only when the I2C is disabled (IC_ENABLE[0] = 0).

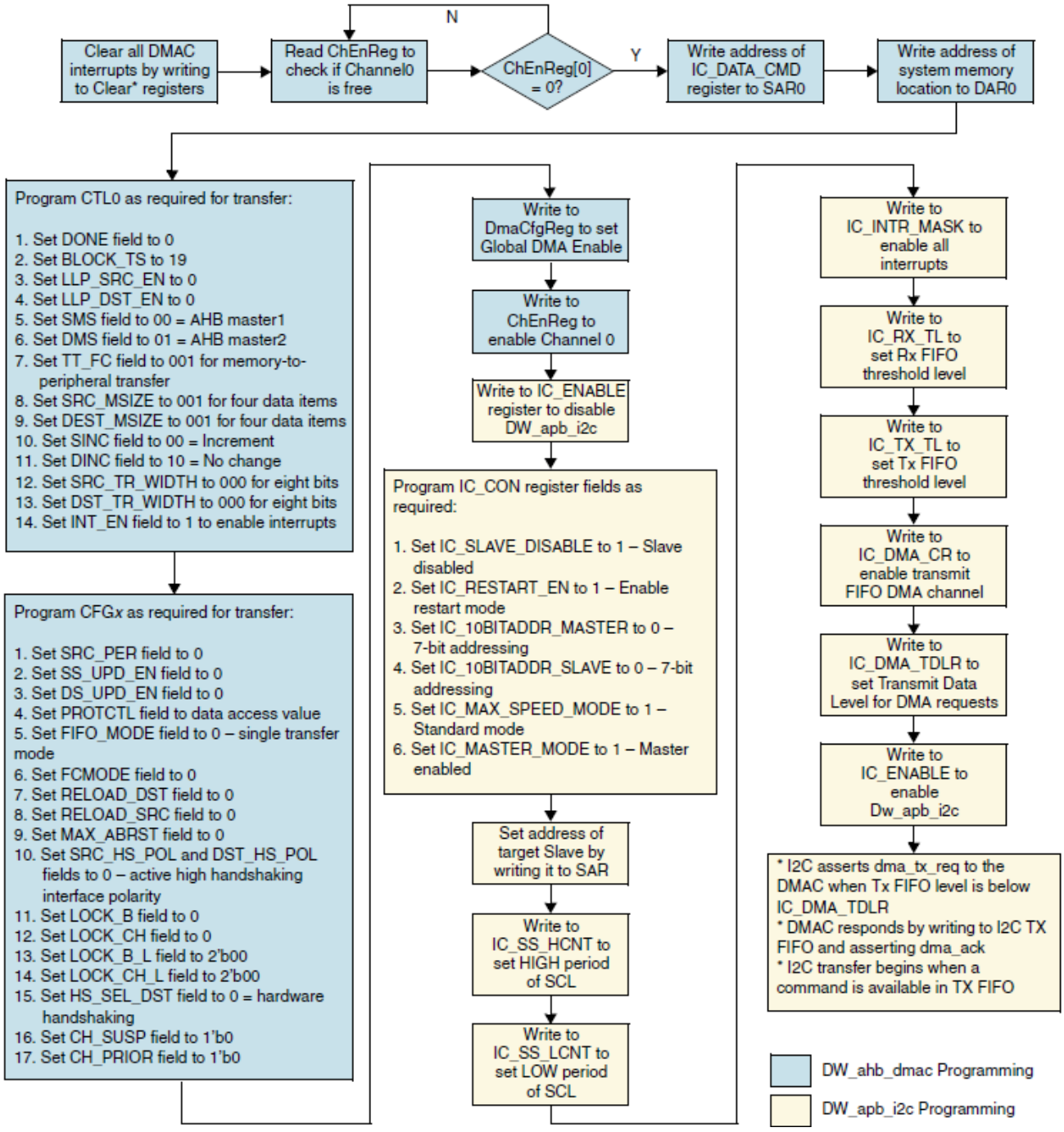
~~The reset value of the IC_SDA_HOLD register can be set via the coreConsultant parameter IC_DEFAULT_SDA_HOLD.~~

~~12.2.8 DMA Controller Interface~~

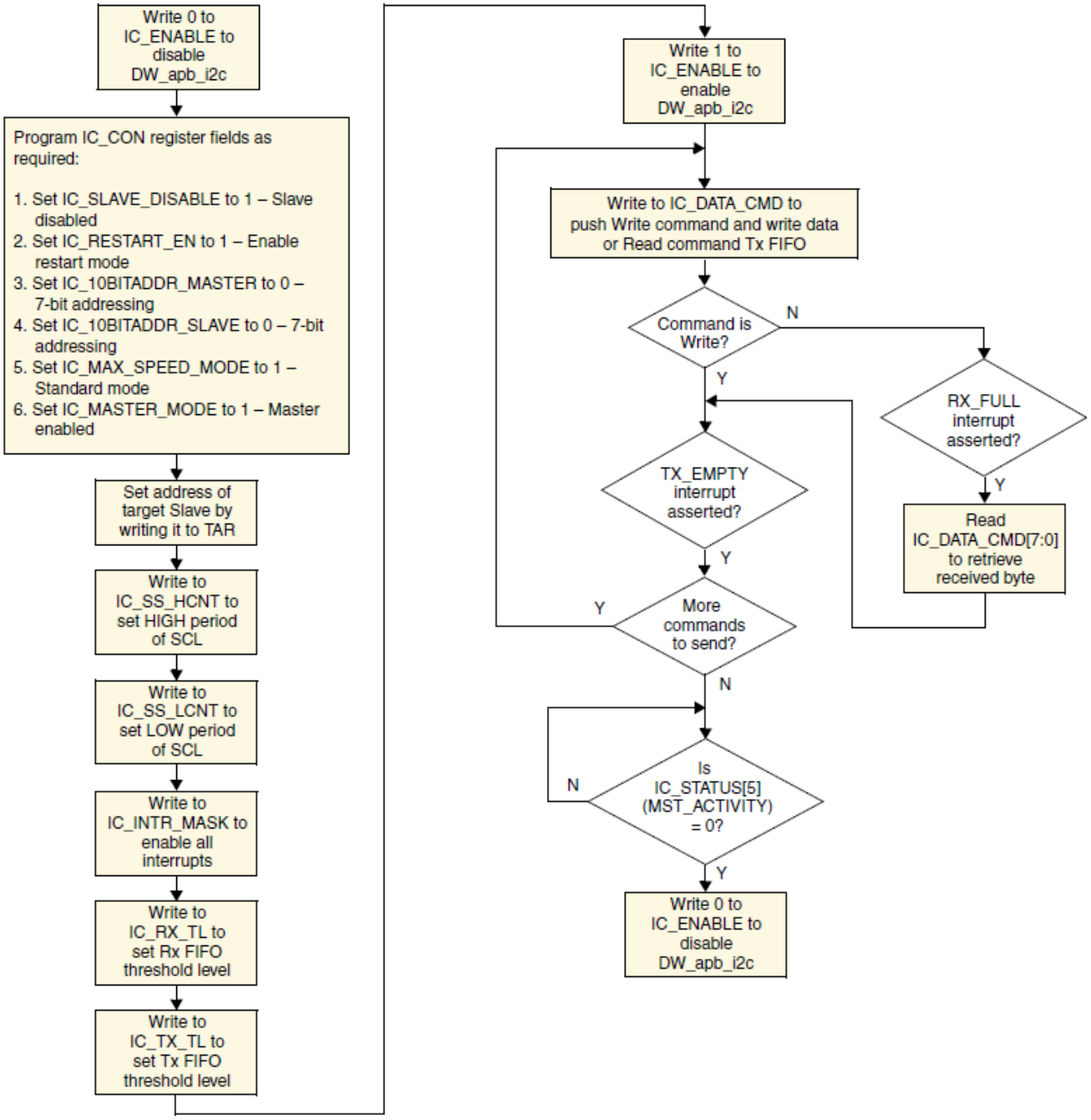
The I2C has an optional built-in DMA capability that can be selected at configuration time; it has a handshaking interface to a DMA Controller to request and control transfers. The APB bus is used to perform the data transfer to or from the DMA. While the I2C DMA operation is designed in a generic way to fit any DMA controller as easily as possible, it is designed to work seamlessly, and best used, with the DesignWare DMA Controller, the DW_ahb_dmac. The settings of the DW_ahb_dmac that are relevant to the operation of the I2C are discussed here, mainly bit fields in the DW_ahb_dmac channel control register, CTL*x*, where *x* is the channel number.

12.3 I2C Programming

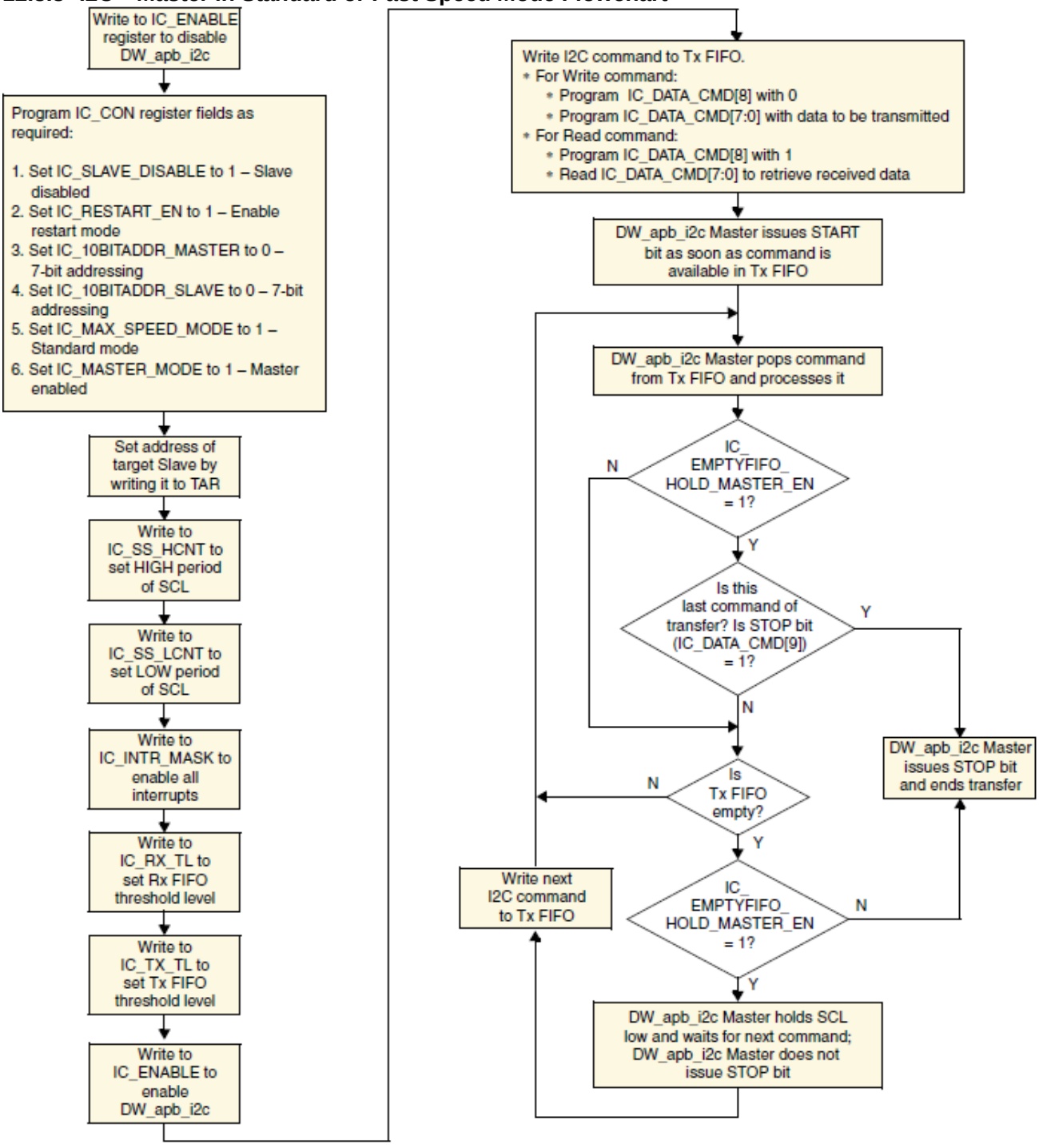
~~12.3.1 DW_DMAC and I2C Flowchart~~



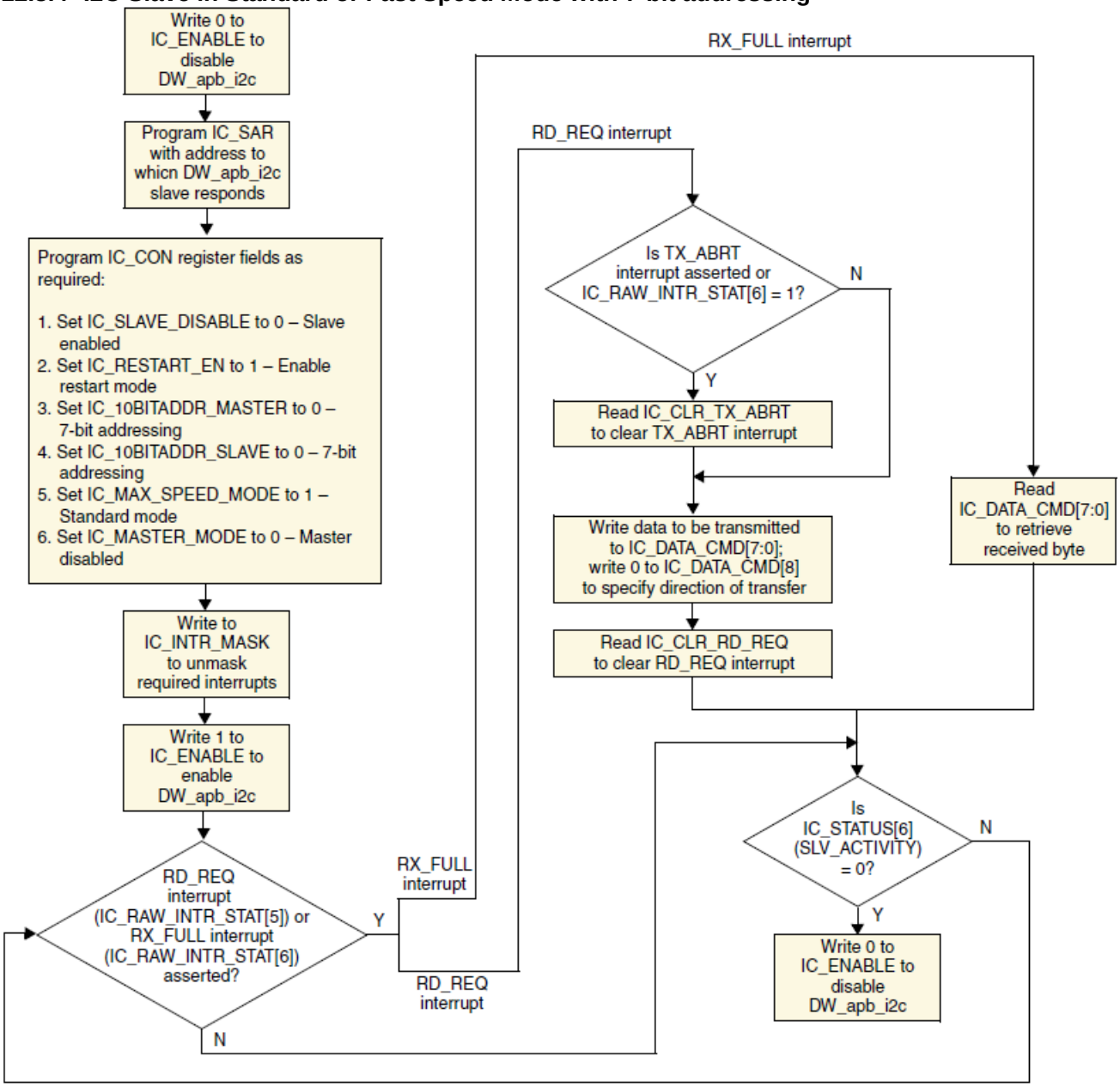
12.3.2 I2C Master Flowchart



12.3.3 I2C Master in Standard or Fast Speed Mode Flowchart



~~12.3.4 I2C Slave in Standard or Fast Speed Mode with 7-bit addressing~~



12.4 I2C Register

This section describes the programmable registers of the I2C.

12.4.1 Register Memory Map

Table Memory Map of I2C

Name	Address Offset	Width	R/W	Description

Name	Address Offset	Width	R/W	Description
IC_CON	0x00	7 bits	R/W ~~or R-only on bit 4~~	I2C Control R/W: If configuration parameter ~~I2C_DYNAMIC_TAR_UPDATE is 0,~~ all bits are Read/Write. ~~If I2C_DYNAMIC_TAR_UPDATE is 1, bit 4 is Read-only.~~ **Reset Value:** Reset values for the 6 bit fields correspond to the following configuration parameters: 6: ~~IC_SLAVE_DISABLE~~ 1'b1 5: ~~IC_RESTART_EN~~ 1'b1 4: ~~IC_10BITADDR_MASTER~~ 1'b1 3: ~~IC_10BITADDR_SLAVE~~ 1'b1 2:1:~~IC_MAX_SPEED_MODE~~ 2'b11 0: ~~IC_MASTER_MODE~~ 1'b1
IC_TAR	0x04	12 or 13 bits	R/W	I2C Target Address **Width:** ~~13, if I2C_DYNAMIC_TAR_UPDATE = 1~~ 12, if I2C_DYNAMIC_TAR_UPDATE = 0 **Reset Value:** Reset values for the four bit fields correspond to the following: 12: 1'b1 ~~IC_10BITADDR_MASTER configuration parameter~~ 11: 0x0 10: 0x0 9:0: 0x055 ~~IC_DEFAULT_TAR_SLAVE_ADDR~~
IC_SAR	0x08	10 bits	R/W	I2C Slave Address **Reset Value:** IC_DEFAULT_SLAVE_ADDR
IC_HS_MADDR	0x0C	3 bits	R/W	I2C HS Master Mode Code Address **Reset Value:** ~~IC_HS_MASTER_CODE~~ 0x055
IC_DATA_CMD	0x10	Refer to Description	R/W	I2C Rx/Tx Data Buffer and Command **Reset Value:** 0x0 Width: ■ ~~IC_EMPTYFIFO_HOLD_MASTER_EN=1 11 bits for write~~ ■ ~~IC_EMPTYFIFO_HOLD_MASTER_EN=0~~ 9 bits for write ■ 8 bits for read **NOTE**: With nine or eleven bits required for writes, the I2C requires 16-bit data on the APB bus transfers when writing into the transmit FIFO. Eight-bit transfers remain for reads from the receive FIFO. **NOTE:** In order for the I2C to continue acknowledging reads, a read command should be written for every byte that is to be received; otherwise the I2C will stop acknowledging.
IC_SS_SCL_HCNT	0x14	16 bits	R/W	Standard speed I2C Clock SCL High Count **Reset Value:** ~~IC_SS_SCL_HIGH_COUNT~~ 0x0190
IC_SS_SCL_LCNT	0x18	16 bits	R/W	Standard speed I2C Clock SCL Low Count **Reset Value:** ~~IC_SS_SCL_LOW_COUNT~~ 0x01d6
IC_FS_SCL_HCNT	0x1C	16 bits	R/W	Fast speed I2C Clock SCL High Count **Reset Value:** ~~IC_FS_SCL_HIGH_COUNT~~ 0x003c

Name	Address Offset	Width	R/W	Description
IC_FS_SCL_LCNT	0x20	16 bits	R/W	Fast speed I2C Clock SCL Low Count **Reset Value:** ~~IC_FS_SCL_LOW_COUNT~~ 0x0082
IC_HS_SCL_HCNT	0x24	16 bits	R/W	High speed I2C Clock SCL High Count **Reset Value:** ~~IC_HS_SCL_HIGH_COUNT~~ 0x0006
IC_HS_SCL_LCNT	0x28	16 bits	R/W	High speed I2C Clock SCL Low Count **Reset Value:** ~~IC_HS_SCL_LOW_COUNT~~ 0x0010
IC_INTR_STAT	0x2C	12 bits	R	I2C Interrupt Status **Reset Value:** 0x0
IC_INTR_MASK	0x30	12 bits	R/W	I2C Interrupt Mask **Reset Value:** 12'h8ff
IC_RAW_INTR_STAT	0x34	12 bits	R	I2C Raw Interrupt Status **Reset Value:** 0x0
IC_RX_TL	0x38	8 bits	R/W	I2C Receive FIFO Threshold **Reset Value:** ~~IC_RX_TL configuration parameter~~0x0
IC_TX_TL	0x3C	8 bits	R/W	I2C Transmit FIFO Threshold **Reset Value:** ~~IC_TX_TL configuration parameter~~0x0
IC_CLR_INTR	0x40	1 bits	R	Clear Combined and Individual Interrupts **Reset Value:** 0x0
IC_CLR_RX_UNDER	0x44	1 bits	R	Clear RX_UNDER Interrupt **Reset Value:** 0x0
IC_CLR_RX_OVER	0x48	1 bits	R	Clear RX_OVER Interrupt **Reset Value:** 0x0
IC_CLR_TX_OVER	0x4C	1 bits	R	Clear TX_OVER Interrupt **Reset Value:** 0x0
IC_CLR_RD_REQ	0x50	1 bits	R	Clear RD_REQ Interrupt **Reset Value:** 0x0
IC_CLR_TX_ABRT	0x54	1 bits	R	Clear TX_ABRT Interrupt **Reset Value:** 0x0
IC_CLR_RX_DONE	0x58	1 bits	R	Clear RX_DONE Interrupt **Reset Value:** 0x0
IC_CLR_ACTIVITY	0x5c	1 bits	R	Clear ACTIVITY Interrupt **Reset Value:** 0x0
IC_CLR_STOP_DET	0x60	1 bits	R	Clear STOP_DET Interrupt **Reset Value:** 0x0
IC_CLR_START_DET	0x64	1 bits	R	Clear START_DET Interrupt **Reset Value:** 0x0
IC_CLR_GEN_CALL	0x68	1 bits	R	Clear GEN_CALL Interrupt **Reset Value:** 0x0
IC_ENABLE	0x6C	2 bits	R/W	I2C Enable **Reset Value:** 0x0
IC_STATUS	0x70	7 bits	R	I2C Status register **Reset Value:** 0x6
IC_TXFLR	0x74	TX_ABW+1	R	Transmit FIFO Level Register **Reset Value:** 0x0
IC_RXFLR	0x78	RX_ABW+1	R	Receive FIFO Level Register **Reset Value:** 0x0
IC_SDA_HOLD	0x7C	16 bits	R/W	SDA hold time length register **Reset Value:** ~~IC_DEFAULT_SDA_HOLD~~ 0x1
IC_TX_ABRT_SOURCE	0x80	32 bits	R	I2C Transmit Abort Status Register **Reset Value:** 0x0

Name	Address Offset	Width	R/W	Description
~~IC_SLV_DATA_NACK_ONLY~~	~~0x84~~	~~1 bit~~	~~R/W~~	~~Generate SLV_DATA_NACK Register~~ **~~Reset Value:~~** ~~0x0~~
~~IC_DMA_CR~~	~~0x88~~	~~2 bits~~	~~R/W~~	~~DMA Control Register for transmit and receive handshaking interface~~ **~~Reset Value:~~** ~~0x0~~
~~IC_DMA_TDLR~~	~~0x8c~~	~~TX_ABW~~	~~R/W~~	~~DMA Transmit Data Level~~ **~~Reset Value:~~** ~~0x0~~
~~IC_DMA_RDLR~~	~~0x90~~	~~RX_ABW~~	~~R/W~~	~~DMA Receive Data Level~~ **~~Reset Value:~~** ~~0x0~~
IC_SDA_SETUP	0x94	8 bits	R/W	I2C SDA Setup Register **Reset Value:** ~~IC_DEFAULT_SDA_SETUP configuration parameter~~ 0x64
IC_ACK_GENERAL_CALL	0x98	1 bit	R/W	I2C ACK General Call Register **Reset Value:** ~~IC_DEFAULT_ACK_GENERAL_CALL configuration parameter~~0x1
IC_ENABLE_STATUS	0x9C	3 bits	R	I2C Enable Status Register **Reset Value:** 0x0
IC_FS_SPKLEN	0xA0	8 bits	R/W	ISS and FS spike suppression limit **Reset Value:** ~~IC_DEFAULT_FS_SPKLEN configuration parameter~~0x05
IC_HS_SPKLEN	0xA4	8 bits	R/W	HS spike suppression limit **Reset Value:** ~~IC_DEFAULT_HS_SPKLEN configuration parameter~~0x01
IC_COMP_PARAM_1	0xf4	32 bits	R	Component Parameter Register **Reset Value:** Reset value depends on configuration parameters.
IC_COMP_VERSION	0xf8	32 bits	R	Component Version ID **Reset Value:** See the releases table in the AMBA 2 release notes
IC_COMP_TYPE	0xfc	32 bits	R	DesignWare Component Type Register **Reset Value:** 0x44570140

12.4.2 Operation of Interrupt Registers

Table lists the operation of the I2C interrupt registers and how they are set and cleared. Some bits are set by hardware and cleared by software, whereas other bits are set and cleared by hardware.

Table Clearing and Setting of Interrupt Registers

Interrupt Bit Fields	Set by Hardware / Cleared by Software	Set and Cleared by Hardware
GEN_CALL	3	8
START_DET	3	8
STOP_DET	3	8
ACTIVITY	3	8
RX_DONE	3	8
TX_ABRT	3	8
RD_REQ	3	8
TX_EMPTY	8	3
TX_OVER	3	8
RX_FULL	8	3
RX_OVER	3	8
RX_UNDER	3	8

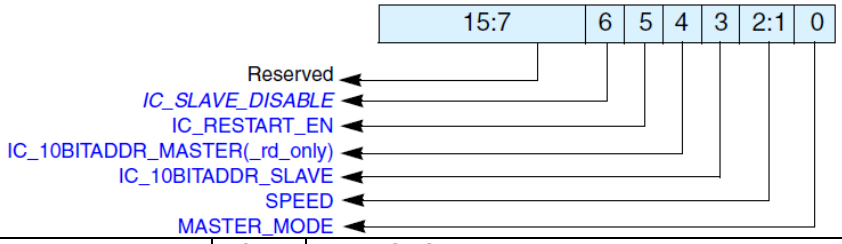
12.4.3 Registers and Field Descriptions

12.4.3.1 IC_CON

- **Name:** I2C Control Register
- **Size:** 7 bits
- **Address Offset:** 0x00
- **Read/Write Access:**

~~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 IC_ENABLE[0] being set to 0. Writes at other times have no effect.



Bits	Name	R/W	Description
15:7	Reserved	N/A	Reserved
6	*IC_SLAVE_DISABLE*	R/W	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. 0: slave is enabled 1: slave is disabled **Reset value:** ~~IC_SLAVE_DISABLE configuration parameter~~0x1 **NOTE**: Software should ensure that if this bit is written with '0,' then bit 0 should also be written with a '0'.
5	*IC_RESTART_EN*	R/W	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 I2C operations. 0: disable 1: enable When the RESTART is disabled, the I2C master is incapable of performing the following functions: ■ Sending a START BYTE ■ Performing any high-speed mode operation ■ Performing direction changes in combined format mode ■ Performing a read operation with a 10-bit address By replacing RESTART condition followed by a STOP and a subsequent START condition, split operations are broken down into multiple I2C 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~~0x1

Bits	Name	R/W	Description
4	*IC_10BITADDR_MASTER*-or *IC_10BITADDR_MASTER_rd_only*	R/W or R	If the I2C_DYNAMIC_TAR_UPDATE configuration parameter is set to "No" (0), this bit is named *IC_10BITADDR_MASTER* and controls whether the I2C 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*. 0: 7-bit addressing 1: 10-bit addressing **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 0x1
3	*IC_10BITADDR_SLAVE*	R/W	When acting as a slave, this bit controls whether the I2C responds to 7- or 10-bit addresses. 0: 7-bit addressing. The I2C ignores transactions that involve 10-bit addressing; for 7-bit addressing, only the lower 7 bits of the IC_SAR register are compared. 1: 10-bit addressing. The I2C responds to only 10-bit addressing transfers that match the full 10 bits of the IC_SAR register. **Reset value:** IC_10BITADDR_SLAVE configuration parameter 0x1
2:1	*SPEED*	R/W	These bits control at which speed the I2C operates; its setting is relevant only if one is operating the I2C 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 3IC_MAX_SPEED_MODE; otherwise, hardware updates this register with the value of 0x3 IC_MAX_SPEED_MODE. ■ 1: standard mode (0 to 100 kbit/s) ■ 2: fast mode (≤ 400 kbit/s) ■ 3: high speed mode (≤3.4 Mbit/s) **Reset value:** IC_MAX_SPEED_MODE configuration0x3
0	*MASTER_MODE*	R/W	This bit controls whether the I2C master is enabled. 0: master disabled 1: master enabled **Reset value:** IC_MASTER_MODE configuration parameter0x1 **NOTE**: Software should ensure that if this bit is written with '1,' then bit 6 should also be written with a '1'.

Certain combinations of the IC_SLAVE_DISABLE (bit 6) and MASTER_MODE (bit 0) result in a configuration error. Table lists the states that result from the combinations of these two bits.

IC_SLAVE_DISABLE (IC_CON[6])	MASTER_MODE IC_CON[0]	State
0	0	Slave Device
0	1	Config Error
1	0	Config Error
1	1	Master Device

12.4.3.2 IC_TAR

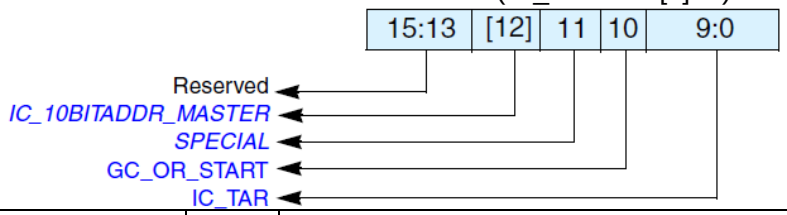
■ Name: I2C Target Address Register

- **Size:** 12 bits ~~or 13 bits; 13 bits only when I2C_DYNAMIC_TAR_UPDATE = 1~~
- **Address Offset:** 0x04
- **Read/Write Access:** Read/Write

~~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.~~ Writes to this register succeed only when IC_ENABLE[0] is set to 0.

~~However, if I2C_DYNAMIC_TAR_UPDATE = 1, then the register becomes 13 bits wide. In this case, writes to IC_TAR succeed when one of the following conditions are true:~~
- ~~I2C is NOT enabled (IC_ENABLE[0] is set to 0); or~~
- ~~I2C is enabled (IC_ENABLE[0]=1); AND~~

I2C is NOT engaged in any Master (tx, rx) operation (IC_STATUS[5]=0); AND I2C is enabled to operate in Master mode (IC_CON[0]=1); AND there are NO entries in the TX FIFO (IC_STATUS[2]=1)

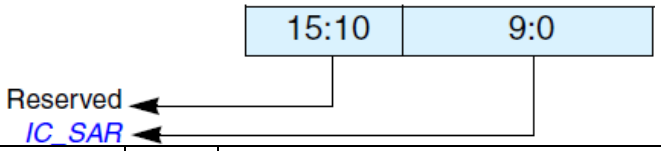


Bits	Name	R/W	Description
15:13	Reserved	N/A	Reserved
12	~~*IC_10BITADDR_MASTER*~~	~~R/W~~	~~This bit controls whether the I2C starts its transfers in 7- or 10-bit addressing mode when acting as a master.~~ ~~■ 0: 7-bit addressing~~ ~~■ 1: 10-bit addressing~~ ~~**Dependencies:** This bit exists in this register only if the I2C_DYNAMIC_TAR_UPDATE configuration parameter is set to "Yes" (1).~~ ~~Reset value: IC_10BITADDR_MASTER configuration parameter~~
11	*SPECIAL*	R/W	This bit indicates whether software performs a General Call or START BYTE command. ■ 0: ignore bit 10 *GC_OR_START* and use IC_TAR normally ■ 1: perform special I2C command as specified in *GC_OR_START* bit **Reset value:** 0x0
10	*GC_OR_START*	R/W	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 I2C. ■ 0: 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 I2C remains in General Call mode until the *SPECIAL* bit value (bit 11) is cleared. ■ 1: START BYTE **Reset value:** 0x0

Bits	Name	R/W	Description
9:0	*IC_TAR*	R/W	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~~ 0x055 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.

12.4.3.3 IC_SAR

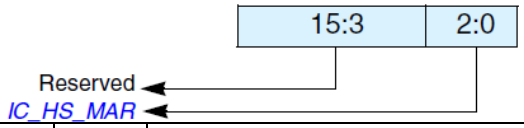
- Name: I2C Slave Address Register
- **Size:** 10 bits
- **Address Offset:** 0x08
- **Read/Write Access:** Read/Write



Bits	Name	R/W	Description
15:10	Reserved	N/A	Reserved
9:0	*IC_SAR*	R/W	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 IC_ENABLE[0] being set to 0. Writes at other times have no effect.

12.4.3.4 IC_HS_MADDR

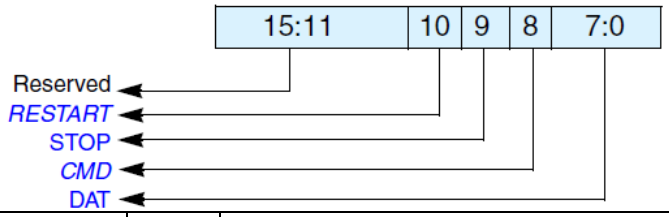
- Name: I2C High Speed Master Mode Code Address Register
- **Size:** 3 bits
- **Address Offset:** 0x0c
- **Read/Write Access:** Read/Write



Bits	Name	R/W	Description
15:3	Reserved	N/A	Reserved
2:0	*IC_HS_MAR*	R/W	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 highspeed 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 0's 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 IC_ENABLE[0] being set to 0. Writes at other times have no effect. **Reset value:** ~~IC_HS_MASTER_CODE configuration parameter~~0x3

12.4.3.5 IC_DATA_CMD

- **Name:** I2C Rx/Tx Data Buffer and Command 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:**
  - ~~IC_EMPTYFIFO_HOLD_MASTER_EN=1 — 11 bits (writes)~~
  - ~~IC_EMPTYFIFO_HOLD_MASTER_EN=0 —~~ 9 bits (writes), 8 bits (reads)
- **Address Offset:** 0x10
- **Read/Write Access:** Read/Write

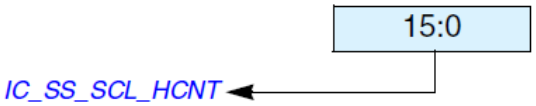


Bits	Name	R/W	Description
15:11	Reserved	N/A	Reserved
10	~~RESTART~~	W	This bit controls whether a RESTART is issued before the byte is sent or received. This bit is available only if IC_EMPTYFIFO_HOLD_MASTER_EN is configured to 1. ■ 1 – If IC_RESTART_EN is 1, a RESTART is issued before the data is sent/received (according to the value of CMD), regardless of whether or not the transfer direction is changing from the previous command; if IC_RESTART_EN is 0, a STOP followed by a START is issued instead. ■ 0 – If IC_RESTART_EN is 1, a RESTART is issued only if the transfer direction is changing from the previous command; if IC_RESTART_EN is 0, a STOP followed by a START is issued instead.
9	~~STOP~~	W	This bit controls whether a STOP is issued after the byte is sent or received. This bit is available only if IC_EMPTYFIFO_HOLD_MASTER_EN is configured to 1. ■ 1 – STOP is issued after this byte, regardless of whether or not the Tx FIFO is empty. If the Tx FIFO is not empty, the master immediately tries to start a new transfer by issuing a START and arbitrating for the bus. ■ 0 – STOP is not issued after this byte, regardless of whether or not the Tx FIFO is empty. If the Tx FIFO is not empty, the master continues the current transfer by sending/receiving data bytes according to the value of the CMD bit. If the Tx FIFO is empty, the master holds the SCL line low and stalls the bus until a new command is available in the Tx FIFO.

Bits	Name	R/W	Description
8	*CMD*	W	This bit controls whether a read or a write is performed. This bit does not control the direction when the I2C acts as a slave. It controls only the direction when it acts as a master. ■ 1 = Read ■ 0 = Write 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 the data in IC_DATA_CMD is to be transmitted. 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. **Reset value:** 0x0
7:0	*DAT*	R/W	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 I2C. However, when you read this register, these bits return the value of data received on the I2C interface. **Reset value:** 0x0

12.4.3.6 IC_SS_SCL_HCNT

- **Name:** Standard Speed I2C Clock SCL High Count Register
- **Size:** 16 bits
- **Address Offset:** 0x14
- **Read/Write Access:** Read/Write

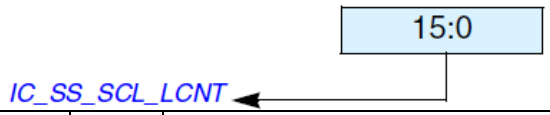


Bits	Name	R/W	Description

Bits	Name	R/W	Description
15:0	*IC_SS_SCL_HCNT*	R/W *	R/W1 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. This register can be written only when the I2C interface is disabled which corresponds to IC_ENABLE[0] 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 I2C.~~ 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.~~ **NOTE:** This register must not be programmed to a value higher than 65525, because I2C 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~~ 0x0190
~~*Read-only if IC_HC_COUNT_VALUES = 1.~~			

12.4.3.7 IC_SS_SCL_LCNT

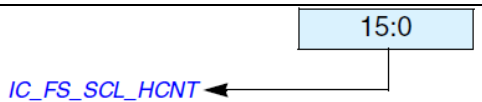
- **Name:** Standard Speed I2C Clock SCL Low Count Register
- **Size:** 16 bits
- **Address Offset:** 0x18
- **Read/Write Access:** Read/Write



Bits	Name	R/W	Description
15:0	*IC_SS_SCL_LCNT*	R/W *	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. This register can be written only when the I2C interface is disabled which corresponds to IC_ENABLE[0] 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 I2C.~~ 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~~ 0x01d6
~~*Read-only if IC_HC_COUNT_VALUES = 1.~~			

12.4.3.8 IC_FS_SCL_HCNT

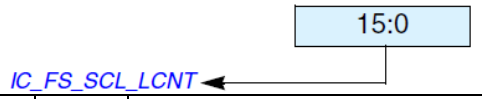
- **Name:** Fast Speed I2C Clock SCL High Count Register
- **Size:** 16 bits
- **Address Offset:** 0x1c
- **Read/Write Access:** Read/Write



Bits	Name	R/W	Description
15:0	*IC_FS_SCL_HCNT*	R/W	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. ~~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 IC_ENABLE[0] 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 I2C.~~ 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~~ 0x003c
~~*Read-only if IC_HC_COUNT_VALUES = 1~~			

12.4.3.9 IC_FS_SCL_LCNT

- **Name:** Fast Speed I2C Clock SCL Low Count Register
- **Size:** 16 bits
- **Address Offset:** 0x20
- **Read/Write Access:** Read/Write



Bits	Name	R/W	Description
15:0	*IC_FS_SCL_LCNT*	R/W	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. ~~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 IC_ENABLE[0] 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 I2C.~~ 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~~ 0x0082
~~*Read-only if IC_HC_COUNT_VALUES = 1~~			

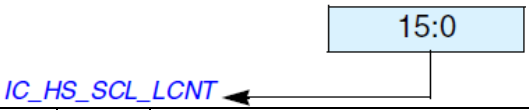
12.4.3.10 IC_HS_SCL_HCNT

- **Name:** High Speed I2C Clock SCL High Count Register
- **Size:** 16 bits
- **Address Offset:** 0x24
- **Read/Write Access:** Read/Write

Bits	Name	R/W	Description
15:0	*IC_HS_SCL_HCNT*	R/W	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 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 IC_ENABLE[0] 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 I2C.~~ 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~~ 0x0006
~~*Read-only if IC_HC_COUNT_VALUES = 1~~			

12.4.3.11 IC_HS_SCL_LCNT

- **Name:** High Speed I2C Clock SCL Low Count Register
- **Size:** 16 bits
- **Address Offset:** 0x28
- **Read/Write Access:** Read/Write



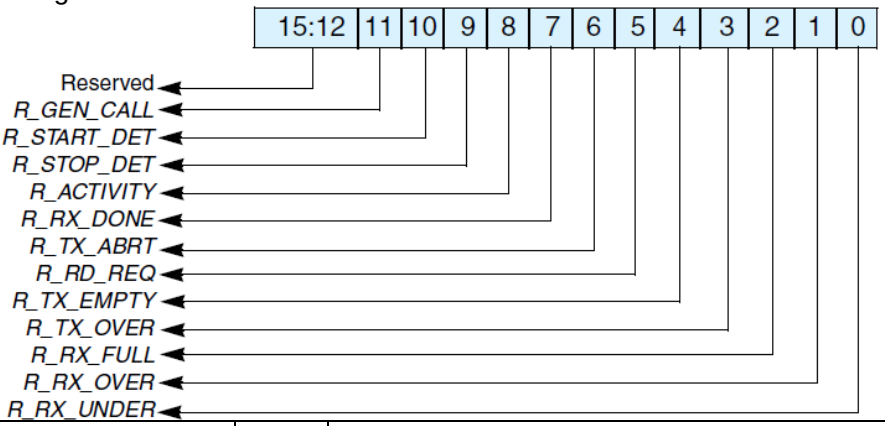
Bits	Name	R/W	Description

Bits	Name	R/W	Description
15:0	*IC_HS_SCL_LCNT*	R/W	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 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 IC_ENABLE[0] 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 I2C.~~ 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~~ 0x0010
~~*Read-only if IC_HC_COUNT_VALUES = 1~~			

12.4.3.12 IC_INTR_STAT

- **Name:** I2C Interrupt Status Register
- **Size:** 12 bits
- **Address Offset:** 0x2C
- **Read/Write Access:** Read

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.



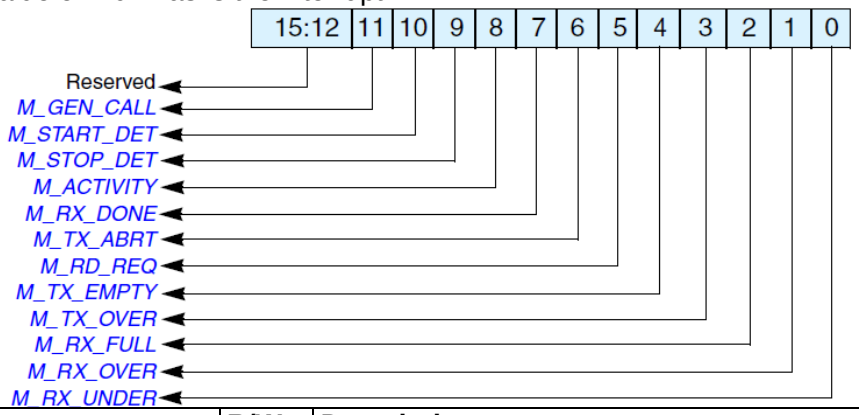
Bits	Name	R/W	Description
15:12	Reserved	N/A	Reserved
11	*R_GEN_CALL*	R	**See "IC_RAW_INTR_STAT" detailed description .** **Reset value:** 0x0
10	*R_START_DET*		
9	*R_STOP_DET*		
8	*R_ACTIVITY*		
7	*R_RX_DONE*		
6	*R_TX_ABRT*		
5	*R_RD_REQ*		
4	*R_TX_EMPTY*		
3	*R_TX_OVER*		

Bits	Name	R/W	Description
2	*R_RX_FULL*		
1	*R_RX_OVER*		
0	*R_RX_UNDER*		

12.4.3.13 IC_INTR_MASK

- **Name:** I2C Interrupt Mask Register
- **Size:** 12 bits
- **Address Offset:** 0x30
- **Read/Write Access:** Read/Write

These bits mask their corresponding interrupt status bits. This register is active low; a value of 0 masks the interrupt, whereas a value of 1 unmasks the interrupt.

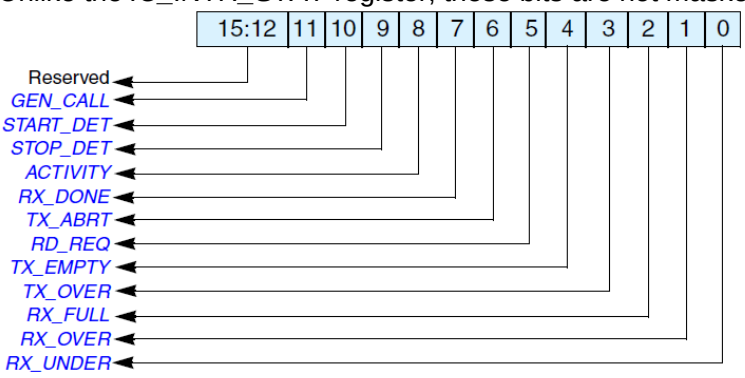


Bits	Name	R/W	Description
15:12	Reserved	N/A	Reserved
11	*M_GEN_CALL*	R/W	These bits mask their corresponding interrupt status bits in the IC_INTR_STAT register. **Reset value:** 12'h8ff
10	*M_START_DET*	R/W	These bits mask their corresponding interrupt status bits in the IC_INTR_STAT register. **Reset value:** 12'h8ff
9	*M_STOP_DET*		
8	*M_ACTIVITY*		
7	*M_RX_DONE*		
6	*M_TX_ABRT*		
5	*M_RD_REQ*		
4	*M_TX_EMPTY*		
3	*M_TX_OVER*		
2	*M_RX_FULL*		
1	*M_RX_OVER*		
0	*M_RX_UNDER*		

12.4.3.14 IC_RAW_INTR_STAT

- **Name:** I2C Raw Interrupt Status Register
- **Size:** 12 bits
- **Address Offset:** 0x34
- **Read/Write Access:** Read

Unlike the *IC_INTR_STAT* register, these bits are not masked so they always show the true status of the I2C.

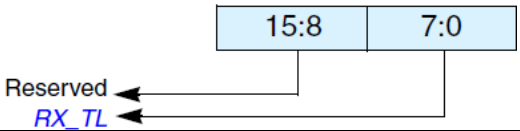


Bits	Name	R/W	Description
15:12	Reserved	N/A	Reserved
11	*GEN_CALL*	R	Set only when a General Call address is received and it is acknowledged. It stays set until it is cleared either by disabling I2C or when the CPU reads bit 0 of the *IC_CLR_GEN_CALL* register. I2C stores the received data in the Rx buffer. **Reset value:** 0x0
10	*START_DET*	R	Indicates whether a START or RESTART condition has occurred on the I2C interface regardless of whether I2C is operating in slave or master mode. **Reset value:** 0x0
9	*STOP_DET*	R	Indicates whether a STOP condition has occurred on the I2C interface regardless of whether I2C is operating in slave or master mode. **Reset value:** 0x0
8	*ACTIVITY*	R	This bit captures I2C activity and stays set until it is cleared. There are four ways to clear it: ■ Disabling the I2C ■ 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 four methods is used to clear it. Even if the I2C module is idle, this bit remains set until cleared, indicating that there was activity on the bus. **Reset value:** 0x0
7	*RX_DONE*	R	When the I2C 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. **Reset value:** 0x0
6	*TX_ABRT*	R	This bit indicates if I2C, 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. **NOTE:** The I2C 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. **Reset value:** 0x0
5	*RD_REQ*	R	This bit is set to 1 when I2C is acting as a slave and another I2C master is attempting to read data from I2C. The I2C 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. **Reset value:** 0x0

Bits	Name	R/W	Description
4	*TX_EMPTY*	R	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 IC_ENABLE[0] is set to 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. **Reset value:** 0x0
3	*TX_OVER*	R	Set during transmit if the transmit buffer is filled to 8 (buffer depth) ~~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. **Reset value:** 0x0
2	*RX_FULL*	R	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 IC_ENABLE[0] is set to 0, regardless of the activity that continues. **Reset value:** 0x0
1	*RX_OVER*	R	Set if the receive buffer is completely filled to 8 (buffer depth) ~~IC_RX_BUFFER_DEPTH~~ and an additional byte is received from an external I2C device. The I2C 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. **Reset value:** 0x0
0	*RX_UNDER*	R	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. **Reset value:** 0x0

12.4.3.15 IC_RX_TL

- **Name:** I2C Receive FIFO Threshold Register
- **Size:** 8bits
- **Address Offset:** 0x38
- **Read/Write Access:** Read/Write

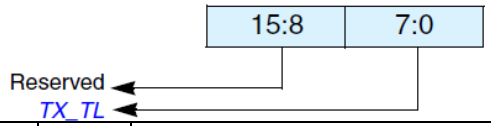


Bits	Name	R/W	Description
15:8	Reserved	N/A	Reserved

Bits	Name	R/W	Description
7:0	*RX_TL*	R/W	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~~ 0x0

12.4.3.16 IC_TX_TL

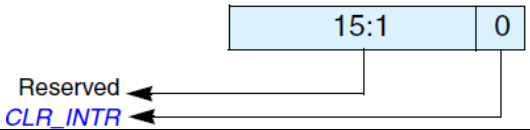
- **Name:** I2C Transmit FIFO Threshold Register
- **Size:** 8 bits
- **Address Offset:** 0x3c
- **Read/Write Access:** Read/Write



Bits	Name	R/W	Description
15:8	Reserved	N/A	Reserved
7:0	*TX_TL*	R/W	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~~ 0x0

12.4.3.17 IC_CLR_INTR

- **Name:** Clear Combined and Individual Interrupt Register
- **Size:** 1 bit
- **Address Offset:** 0x40
- **Read/Write Access:** Read



Bits	Name	R/W	Description
15:1	Reserved	N/A	Reserved
0	*CLR_INTR*	R	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*. **Reset value:** 0x0

12.4.3.18 IC_CLR_RX_UNDER

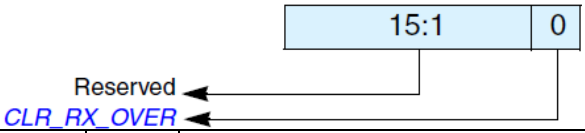
- **Name:** Clear RX_UNDER Interrupt Register
- **Size:** 1 bit
- **Address Offset:** 0x44

- **Read/Write Access:** Read

Bits	Name	R/W	Description
15:1	Reserved	N/A	Reserved
0	*CLR_RX_UNDER*	R	Read this register to clear the *RX_UNDER* interrupt (bit 0) of the *IC_RAW_INTR_STAT* register. **Reset value:** 0x0

12.4.3.19 IC_CLR_RX_OVER

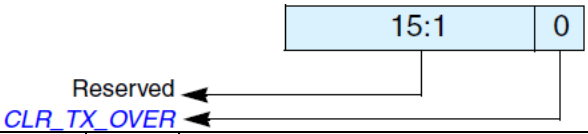
- **Name:** Clear RX_OVER Interrupt Register
- **Size:** 1 bit
- **Address Offset:** 0x48
- **Read/Write Access:** Read



Bits	Name	R/W	Description
15:1	Reserved	N/A	Reserved
0	*CLR_RX_OVER*	R	Read this register to clear the *RX_OVER* interrupt (bit 1) of the *IC_RAW_INTR_STAT* register. **Reset value:** 0x0

12.4.3.20 IC_CLR_TX_OVER

- **Name:** Clear TX_OVER Interrupt Register
- **Size:** 1 bit
- **Address Offset:** 0x4c
- **Read/Write Access:** Read



Bits	Name	R/W	Description
15:1	Reserved	N/A	Reserved
0	*CLR_TX_OVER*	R	Read this register to clear the *TX_OVER* interrupt (bit 3) of the *IC_RAW_INTR_STAT* register. **Reset value:** 0x0

12.4.3.21 IC_CLR_RD_REQ

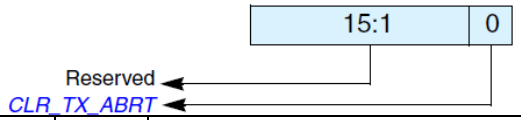
- **Name:** Clear RD_REQ Interrupt Register
- **Size:** 1 bit
- **Address Offset:** 0x50
- **Read/Write Access:** Read

Bits	Name	R/W	Description
15:1	Reserved	N/A	Reserved
0	*CLR_RD_REQ*	R	Read this register to clear the *RD_REQ* interrupt (bit 5) of the *IC_RAW_INTR_STAT* register. **Reset value:** 0x0

12.4.3.22 IC_CLR_TX_ABRT

- **Name:** Clear TX_ABRT Interrupt Register
- **Size:** 1 bit

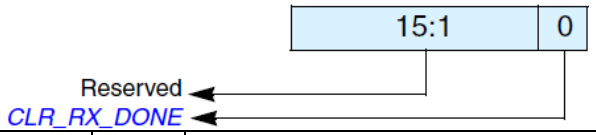
- **Address Offset:** 0x54
- **Read/Write Access:** Read



Bits	Name	R/W	Description
15:1	Reserved	N/A	Reserved
0	*CLR_TX_ABRT*	R	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*. **Reset value**: 0x0

12.4.3.23 IC_CLR_RX_DONE

- **Name:** Clear RX_DONE Interrupt Register
- **Size:** 1 bit
- **Address Offset:** 0x58
- **Read/Write Access:** Read



Bits	Name	R/W	Description
15:1	Reserved	N/A	Reserved
0	*CLR_RX_DONE*	R	Read this register to clear the *RX_DONE* interrupt (bit 7) of the *IC_RAW_INTR_STAT* register. **Reset value:** 0x0

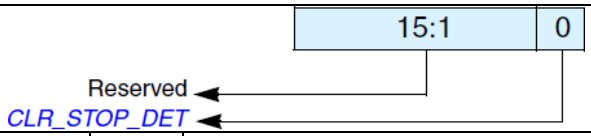
12.4.3.24 IC_CLR_ACTIVITY

- **Name:** Clear ACTIVITY Interrupt Register
- **Size:** 1 bit
- **Address Offset:** 0x5c
- **Read/Write Access:** Read

Bits	Name	R/W	Description
15:1	Reserved	N/A	Reserved
0	*CLR_ACTIVITY*	R	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. **Reset value:** 0x0

12.4.3.25 IC_CLR_STOP_DET

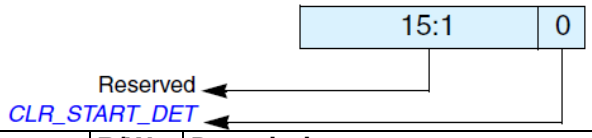
- **Name:** Clear STOP_DET Interrupt Register
- **Size:** 1 bit
- **Address Offset:** 0x60
- **Read/Write Access:** Read



Bits	Name	R/W	Description
15:1	Reserved	N/A	Reserved
0	*CLR_STOP_DET*	R	Read this register to clear the *STOP_DET* interrupt (bit 9) of the *IC_RAW_INTR_STAT* register. **Reset value:** 0x0

12.4.3.26 IC_CLR_START_DET

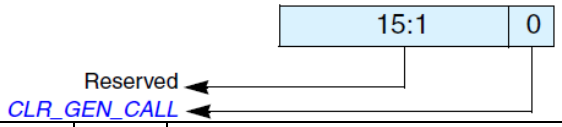
- **Name:** Clear START_DET Interrupt Register
- **Size:** 1 bit
- **Address Offset:** 0x64
- **Read/Write Access:** Read



Bits	Name	R/W	Description
15:1	Reserved	N/A	Reserved
0	*CLR_START_DET*	R	Read this register to clear the *START_DET* interrupt (bit 10) of the *IC_RAW_INTR_STAT* register. **Reset value:** 0x0

12.4.3.27 IC_CLR_GEN_CALL

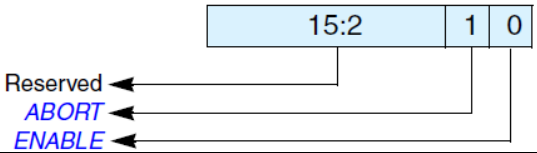
- **Name:** Clear GEN_CALL Interrupt Register
- **Size:** 1 bit
- **Address Offset:** 0x68
- **Read/Write Access:** Read



Bits	Name	R/W	Description
15:1	Reserved	N/A	Reserved
0	*CLR_GEN_CALL*	R	Read this register to clear the *GEN_CALL* interrupt (bit 11) of *IC_RAW_INTR_STAT* register. **Reset value:** 0x0

12.4.3.28 IC_ENABLE

- Name: I2C Enable Register
- **Size:** 2 bits
- **Address Offset:** 0x6c
- **Read/Write Access:** Read/Write



Bits	Name	R/W	Description
15:2	Reserved	N/A	Reserved

Bits	Name	R/W	Description
1	*ABORT*	R/W	When set, the controller initiates the transfer abort. ■ 0: ABORT not initiated or ABORT done ■ 1: ABORT operation in progress The software can abort the I2C transfer in master mode by setting this bit. The software can set this bit only when ENABLE is already set; otherwise, the controller ignores any write to ABORT bit. The software cannot clear the ABORT bit once set. In response to an ABORT, the controller issues a STOP and flushes the Tx FIFO after completing the current transfer, then sets the TX_ABORT interrupt after the abort operation. The ABORT bit is cleared automatically after the abort operation. **Reset value:** 0x0
0	*ENABLE*	R/W	Controls whether the I2C is enabled. ■ 0: Disables I2C (TX and RX FIFOs are held in an erased state) ■ 1: Enables I2C Software can disable I2C while it is active. However, it is important that care be taken to ensure that I2C is disabled properly. When I2C is disabled, the following occurs: ■ The TX FIFO and RX FIFO get flushed. ■ Status bits in the *IC_INTR_STAT* register are still active until I2C 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 I2C 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 two ic_clk delay when enabling or disabling the I2C. **Reset value:** 0x0

12.4.3.29 IC_STATUS

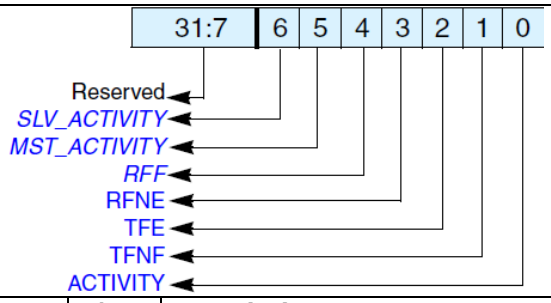
- **Name:** I2C Status Register
- **Size:** 7 bits
- **Address Offset:** 0x70
- **Read/Write Access:** Read

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



Bits	Name	R/W	Description
31:7	Reserved	N/A	Reserved
6	SLV_ACTIVITY	R	Slave FSM Activity Status. When the Slave Finite State Machine (FSM) is not in the IDLE state, this bit is set. ■ 0: Slave FSM is in IDLE state so the Slave part of I2C is not Active ■ 1: Slave FSM is not in IDLE state so the Slave part of I2C is Active **Reset value:** 0x0
5	MST_ACTIVITY	R	Master FSM Activity Status. When the Master Finite State Machine (FSM) is not in the IDLE state, this bit is set. ■ 0: Master FSM is in IDLE state so the Master part of I2C is not Active ■ 1: Master FSM is not in IDLE state so the Master part of I2C is Active **Reset value:** 0x0
4	*RFF*	R	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. ■ 0: Receive FIFO is not full ■ 1: Receive FIFO is full **Reset value:** 0x0
3	*RFNE*	R	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. ■ 0: Receive FIFO is empty ■ 1: Receive FIFO is not empty **Reset value:** 0x0
2	*TFE*	R	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. ■ 0: Transmit FIFO is not empty ■ 1: Transmit FIFO is empty **Reset value:** 0x1
1	*TFNF*	R	Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. ■ 0: Transmit FIFO is full ■ 1: Transmit FIFO is not full **Reset value:** 0x1
0	*ACTIVITY*	R	I2C Activity Status. **Reset value:** 0x0

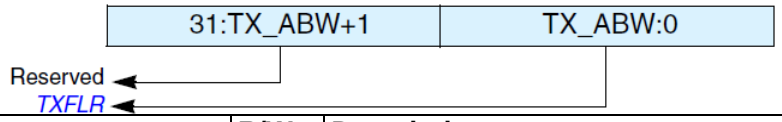
12.4.3.30 IC_TXFLR

- **Name:** I2C Transmit FIFO Level Register
- **Size:** ~~TX_ABW + 1~~ 4bits
- **Address Offset:** 0x74
- **Read/Write Access:** Read

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.



Bits	Name	R/W	Description
31: 4 ~~TX_ABW+1~~	Reserved	N/A	Reserved
~~TX_ABW~~3:0	*TXFLR*	R	**Transmit FIFO Level**. Contains the number of valid data entries in the transmit FIFO. **Reset value:** 0x0

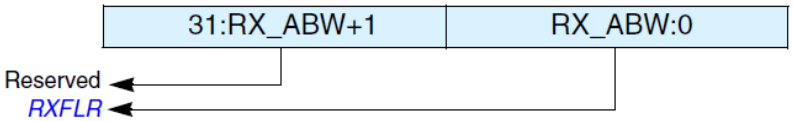
12.4.3.31 IC_RXFLR

- **Name:** I2C Receive FIFO Level Register
- **Size:** ~~RX_ABW + 1~~ 4 bits
- **Address Offset:** 0x78
- **Read/Write Access:** Read

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.



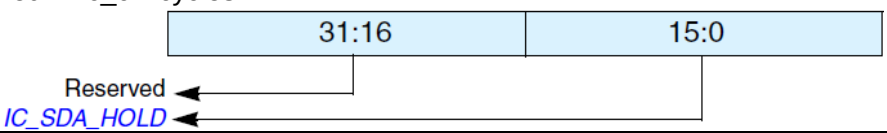
Bits	Name	R/W	Description
31: 4 ~~RX_ABW+1~~	Reserved	N/A	Reserved
~~RX_ABW~~3:0	*RXFLR*	R	**Receive FIFO Level**. Contains the number of valid data entries in the receive FIFO. **Reset value:** 0x0

12.4.3.32 IC_SDA_HOLD

- **Name:** I2C SDA Hold Time Length Register
- **Size:** 16 bits
- **Address Offset:** 0x7C
- **Read/Write Access:** Read/Write

This register controls the amount of hold time on the SDA signal after a negative edge of SCL in both master and slave mode, in units of ic_clk period. The value programmed must be greater than the minimum hold time in each mode for the value to be implemented—one cycle in master mode, seven cycles in slave mode. Writes to this register succeed only when IC_ENABLE[0]=0.

The programmed SDA hold time cannot exceed at any time the duration of the low part of scl. Therefore the programmed value cannot be larger than N_SCL_LOW-2, where N_SCL_LOW is the duration of the low part of the scl period measured in ic_clk cycles.



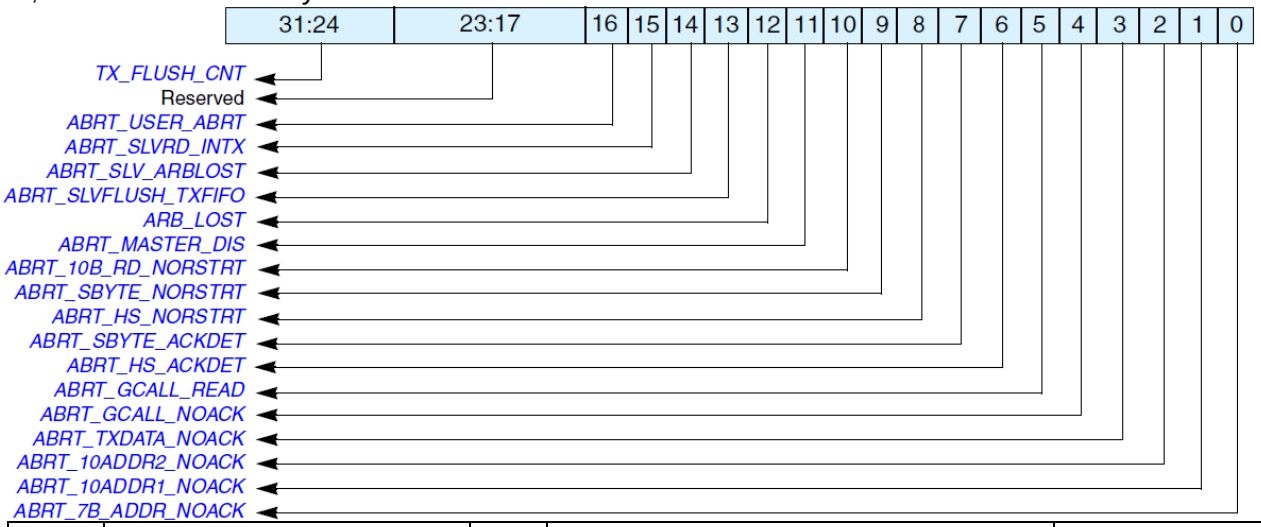
Bits	Name	R/W	Description
31:16	Reserved	N/A	Reserved
15:0	*IC_SDA_HOLD*	R/W	Sets the required SDA hold time in units of ic_clk period. **Reset value:** ~~IC_DEFAULT_SDA_HOLD~~ 0x01

12.4.3.33 IC_TX_ABRT_SOURCE

- **Name:** I2C Transmit Abort Source Register
- **Size:** 32 bits
- **Address Offset:** 0x80
- **Read/Write Access:** Read

This register has 32 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 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 is then re-asserted.



Bits	Name	R/W	Description	Role of I2C
31:24	*TX_FLUSH_CNT*	R	This field preserves the TXFLR value prior to the last TX_ABRT event. It is cleared whenever I2C is disabled. **Reset value:** 0x0	Master-Transmitter
23:17	Reserved			
16	*ABRT_USER_ABRT*	R	This is a master-mode-only bit. Master has detected the transfer abort (IC_ENABLE[1]). **Reset value:** 0x0	Master-Transmitter
15	*ABRT_SLVRD_INTX*	R	1: 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. **Reset value:** 0x0	Slave-Transmitter

Bits	Name	R/W	Description	Role of I2C
14	*ABRT_SLV_ARBLOST*	R	1: Slave lost the bus while transmitting data to a remote master. *IC_TX_ABRT_SOURCE*[12] is set at the same time. **Note:** 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 I2C no longer own the bus. **Reset value:** 0x0	Slave-Transmitter
13	*ABRT_SLVFLUSH_TXFIFO*	R	1: 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. **Reset value:** 0x0	Slave-Transmitter
12	*ARB_LOST*	R	1: Master has lost arbitration, or if *IC_TX_ABRT_SOURCE*[14] is also set, then the slave transmitter has lost arbitration. Note: I2C can be both master and slave at the same time. **Reset value:** 0x0	Master-Transmitter or Slave-Transmitter
11	*ABRT_MASTER_DIS*	R	1: User tries to initiate a Master operation with the Master mode disabled. **Reset value:** 0x0	Master-Transmitter or Master-Receiver
10	*ABRT_10B_RD_NORSTRT*	R	1: The restart is disabled (*IC_RESTART_EN* bit (IC_CON[5]) = 0) and the master sends a read command in 10-bit addressing mode. **Reset value:** 0x0	Master-Receiver
9	*ABRT_SBYTE_NORSTRT*	R	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 re-asserted. 1: The restart is disabled (*IC_RESTART_EN* bit (IC_CON[5]) = 0) and the user is trying to send a START Byte. **Reset value:** 0x0	Master

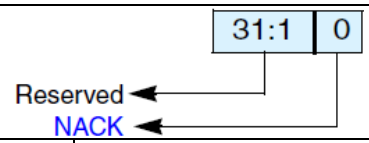
Bits	Name	R/W	Description	Role of I2C
8	*ABRT_HS_NORSTRT*	R	1: The restart is disabled (*IC_RESTART_EN* bit (IC_CON[5]) = 0) and the user is trying to use the master to transfer data in High Speed mode. **Reset value:** 0x0	Master-Transmitter or Master-Receiver
7	*ABRT_SBYTE_ACKDET*	R	1: Master has sent a START Byte and the START Byte was acknowledged (wrong behavior). **Reset value:** 0x0	Master
6	*ABRT_HS_ACKDET*	R	1: Master is in High Speed mode and the High Speed Master code was acknowledged (wrong behavior). **Reset value:** 0x0	Master
5	*ABRT_GCALL_READ*	R	1: I2C 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). **Reset value:** 0x0	Master-Transmitter
4	*ABRT_GCALL_NOACK*	R	1: I2C in master mode sent a General Call and no slave on the bus acknowledged the General Call. **Reset value:** 0x0	Master-Transmitter
3	*ABRT_TXDATA_NOACK*	R	1: 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). **Reset value:** 0x0	Master-Transmitter
2	*ABRT_10ADDR2_NOACK*	R	1: Master is in 10-bit address mode and the second address byte of the 10-bit address was not acknowledged by any slave. **Reset value:** 0x0	Master-Transmitter or Master-Receiver
1	*ABRT_10ADDR1_NOACK*	R	1: Master is in 10-bit address mode and the first 10-bit address byte was not acknowledged by any slave. **Reset value:** 0x0	Master-Transmitter or Master-Receiver
0	*ABRT_7B_ADDR_NOACK*	R	1: Master is in 7-bit addressing mode and the address sent was not acknowledged by any slave. **Reset value:** 0x0	Master-Transmitter or Master-Receiver

~~12.4.3.34 IC_SLV_DATA_NACK_ONLY~~

- **Name:** Generate Slave Data NACK Register
- **Size:** 1 bit
- **Address Offset:** 0x84
- **Read/Write Access:** Read/Write

The register is used to generate a NACK for the data part of a transfer when I2C 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. A write can occur on this register if either of the following conditions are met:

- I2C is disabled (IC_ENABLE[0] = 0)
- Slave part is inactive (IC_STATUS[6] = 0)

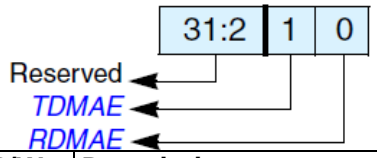


Bits	Name	R/W	Description
31:1	Reserved	N/A	Reserved
0	*NACK*	R/W	Generate NACK. This NACK generation only occurs when I2C is a slavereceiver. If this register is set to a value of 1, it can only generate a NACK after a data byte is received; hence, the data transfer is aborted and the data received 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. ■ 1 = generate NACK after data byte received ■ 0 = generate NACK/ACK normally **Reset value:** 0x0

~~12.4.3.35 IC_DMA_CR~~

- **Name:** DMA Control Register
- **Size:** 2 bits
- **Address Offset:** 0x88
- **Read/Write Access:** Read/Write

This register is only valid when I2C is configured with a set of DMA Controller interface signals (IC_HAS_DMA = 1). When I2C is not configured for DMA operation, this register does not exist and writing to the register's 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.

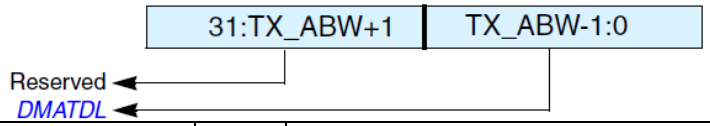


Bits	Name	R/W	Description
31:2	Reserved	N/A	Reserved
1	*TDMAE*	R/W	**Transmit DMA Enable**. This bit enables/disables the transmit FIFO DMA channel. ■ 0 = Transmit DMA disabled ■ 1 = Transmit DMA enabled **Reset value:** 0x0
0	*RDMAE*	R/W	**Receive DMA Enable**. This bit enables/disables the receive FIFO DMA channel. ■ 0 = Receive DMA disabled ■ 1 = Receive DMA enabled **Reset value:** 0x0

~~12.4.3.36 IC_DMA_TDLR~~

- **Name:** DMA Transmit Data Level Register
- **Size:** TX_ABW–1:0
- **Address Offset:** 0x8c
- **Read/Write Access:** Read/Write

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.



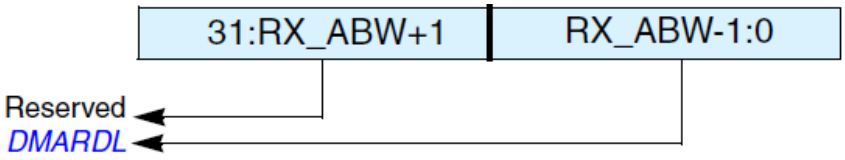
Bits	Name	R/W	Description
31:TX_ABW	Reserved	N/A	Reserved

Bits	Name	R/W	Description
TX_ABW–1:0	*DMATDL*	R/W	**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. **Reset value:** 0x0

~~12.4.3.37 IC_DMA_RDLR~~

- **Name:** I2C Receive Data Level Register
- **Size:** RX_ABW–1:0
- **Address Offset:** 0x90
- **Read/Write Access:** Read/Write

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.

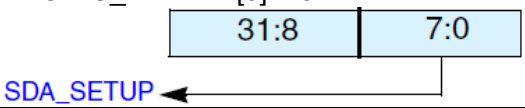


Bits	Name	R/W	Description
31:RX_ABW	Reserved	N/A	Reserved
RX_ABW–1:0	*DMARDL*	R/W	**Receive Data Level**. This bit field controls the level at which a DMA request is made by the receive logic. The watermark level = *DMARDL*+1; that is, dma_rx_req is generated when the number of valid data entries in 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. **Reset value:** 0x0

12.4.3.38 IC_SDA_SETUP

- **Name:** I2C SDA Setup Register
- **Size:** 8 bits
- **Address Offset:** 0x94
- **Read/Write Access:** Read/Write

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—by holding SCL low when I2C services a read request while operating as a slave-transmitter. The relevant I2C requirement is tSU:DAT (note 4) as detailed in the *I2C Bus Specification*. This register must be programmed with a value equal to or greater than 2.
Writes to this register succeed only when IC_ENABLE[0] = 0.

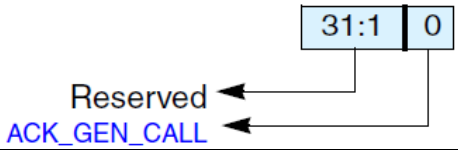


Bits	Name	R/W	Description
31:8	Reserved	N/A	Reserved
7:0	*SDA_SETUP*	R/W	**SDA Setup**. It is recommended that if the required delay is 1000ns, then for an ic_clk frequency of 10 MHz, IC_SDA_SETUP should be programmed to a value of 11. IC_SDA_SETUP must be programmed with a minimum value of 2. **Default Reset value:** 0x64~~, but can be hardcoded by setting the IC_DEFAULT_SDA_SETUP configuration parameter.~~

12.4.3.39 IC_ACK_GENERAL_CALL

- **Name:** I2C ACK General Call Register
- **Size:** 1 bit
- **Address Offset:** 0x98
- **Read/Write Access:** Read/Write

The register controls whether I2C responds with a ACK or NACK when it receives an I2C General Call address.



Bits	Name	R/W	Description
31:1	Reserved	N/A	Reserved
0	*ACK_GEN_CALL*	R/W	**ACK General Call**. When set to 1, I2C responds with a ACK (by asserting ic_data_oe) when it receives a General Call. When set to 0, the I2C does not generate General Call interrupts. **Default Reset value:** 0x1~~, but can be hardcoded by setting the IC_DEFAULT_ACK_GENERAL_CALL configuration parameter.~~

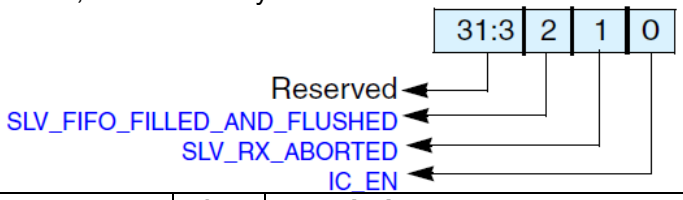
12.4.3.40 IC_ENABLE_STATUS

- **Name:** I2C Enable Status Register
- **Size:** 3 bits
- **Address Offset:** 0x9C
- **Read/Write Access:** Read

The register is used to report the I2C hardware status when IC_ENABLE[0] is set from 1 to 0; that is, when I2C is disabled.

If IC_ENABLE[0] has been set to 1, bits 2:1 are forced to 0, and bit 0 is forced to 1.

If IC_ENABLE[0] has been set to 0, bits 2:1 is only be valid as soon as bit 0 is read as '0'.



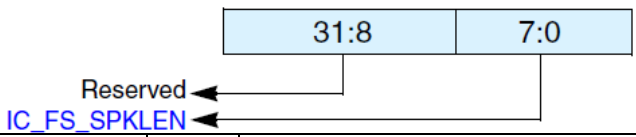
Bits	Name	R/W	Description
31:2	Reserved	N/A	Reserved
2	*SLV_RX_DATA_LOST*	R	**Slave Received Data Lost.** This bit indicates if a Slave-Receiver operation has been aborted with at least one data byte received from an I2C transfer due to setting IC_ENABLE[0] from 1 to 0. When read as 1, I2C is deemed 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. **NOTE**: If the remote I2C master terminates the transfer with a STOP condition before the I2C has a chance to NACK a transfer, and IC_ENABLE[0] has been set to 0, then this bit is also set to 1. When read as 0, I2C is deemed to have been disabled without being actively involved in the data phase of a Slave-Receiver transfer. **NOTE:** The CPU can safely read this bit when IC_EN (bit 0) is read as 0. **Reset value:** 0x0

Bits	Name	R/W	Description
1	*SLV_DISABLED_WHILE_BUSY*	R	**Slave Disabled While Busy (Transmit, Receive)**. This bit indicates if a potential or active Slave operation has been aborted due to setting bit 0 of the IC_ENABLE register from 1 to 0. This bit is set when the CPU writes a 0 to bit 0 of IC_ENABLE while: (a) I2C 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, I2C 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 I2C (IC_SAR register) OR if the transfer is completed before bit 0 of IC_ENABLE is set to 0, but has not taken effect. **NOTE**: If the remote I2C master terminates the transfer with a STOP condition before the I2C has a chance to NACK a transfer, and bit 0 of IC_ENABLE has been set to 0, then this bit will also be set to 1. When read as 0, I2C is deemed to have been disabled when there is master activity, or when the I2C bus is idle. **NOTE:** The CPU can safely read this bit when *IC_EN* (bit 0) is read as 0. **Reset value:** 0x0
0	*IC_EN*	R	**ic_en Status**. This bit always reflects the value driven on the output port ic_en. ■ When read as 1, I2C is deemed to be in an enabled state. ■ When read as 0, I2C is deemed completely inactive. **NOTE:** 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). **Reset value:** 0x0

12.4.3.41 IC_FS_SPKLEN

- **Name:** I2C SS and FS Spike Suppression Limit Register
- **Size:** 8 bits
- **Address Offset:** 0xA0
- **Read/Write Access:** Read/Write

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 when 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 1.



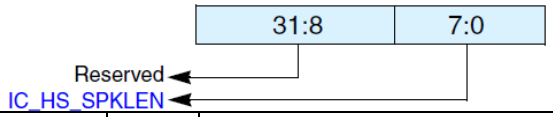
Bits	Name	R/W	Description
31:8	Reserved	N/A	Reserved

Bits	Name	R/W	Description
7:0	*IC_FS_SPKLEN*	R/W	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 are filtered out by the spike suppression logic; for more information. This register can be written only when the I2C interface is disabled, which corresponds to IC_ENABLE[0] being set to 0. Writes at other times have no effect. The minimum valid value is 1; hardware prevents values less than this being written, and if attempted, results in 1 being set. **Reset value:** ~~*IC_DEFAULT_FS_SPKLEN* configuration parameter~~ 0x05

12.4.3.42 IC_HS_SPKLEN

- **Name:** I2C HS Spike Suppression Limit Register
- **Size:** 8 bits
- **Address Offset:** 0xA4
- **Read/Write Access:** Read/Write

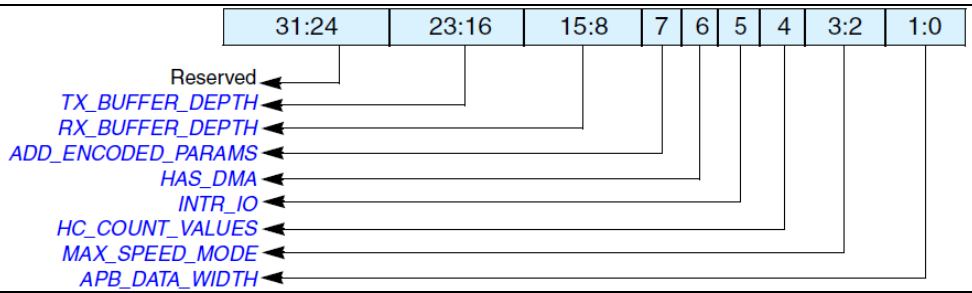
This register is used to store the duration, measured in ic_clk cycles, of the longest spike that is filtered out by the spike suppression logic when the component is operating in HS mode. The relevant I2C requirement is tSP (Table 6) as detailed in the *I2C Bus Specification*. This register must be programmed with a minimum value of 1~~ and is implemented only if the component is configured to support HS mode; that is, if the IC_MAX_SPEED_MODE parameter is set to 3.~~



Bits	Name	R/W	Description
31:8	Reserved	N/A	Reserved
7:0	*IC_HS_SPKLEN*	R/W	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 are filtered out by the spike suppression logic. This register can be written only when the I2C interface is disabled, which corresponds to IC_ENABLE[0] being set to 0. Writes at other times have no effect. The minimum valid value is 1; hardware prevents values less than this being written, and if attempted, results in 1 being set. ~~This register is implemented only if the component is configured to support HS mode; that is, if the IC_MAX_SPEED_MODE parameter is set to 3.~~ **Reset value:** ~~*IC_DEFAULT_HS_SPKLEN* configuration parameter~~ 0x01

12.4.3.43 IC_COMP_PARAM_1

- **Name:** Component Parameter Register 1
- **Size:** 32 bits
- **Address Offset:** 0xf4
- **Read/Write Access:** Read

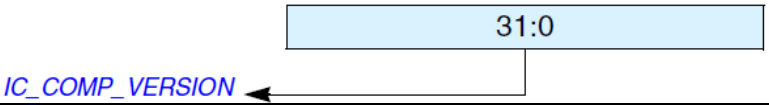


Bits	Name	R/W	Description
31:24	Reserved	N/A	Reserved
23:16	*TX_BUFFER_DEPTH*	R	The value of this register is 0x07 (depth is 8) ~~derived from the IC_TX_BUFFER_DEPTH coreConsultant parameter.~~ ■ ~~0x00 = Reserved~~ ■ ~~0x01 = 2~~ ■ ~~0x02 = 3~~ ~~…~~ ■ ~~0xFF = 256~~
15:8	*RX_BUFFER_DEPTH*	R	The value of this register is 0x07 (depth is 8) ~~derived from the IC_RX_BUFFER_DEPTH coreConsultant parameter.~~ ■ ~~0x00 = Reserved~~ ■ ~~0x01 = 2~~ ■ ~~0x02 = 3~~ ~~…~~ ■ ~~0xFF = 256~~
7	*ADD_ENCODED_PARAMS*	R	~~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 via software has been included. ~~Otherwise, the entire register is 0 regardless of the setting of any other parameters that are encoded in the bits.~~ ■ ~~0: False~~ ■ 1: True
6	*HAS_DMA*	R	~~The value of this register is derived from the IC_HAS_DMA coreConsultant parameter.~~ ■ 0: False ■ ~~1: True~~
5	*INTR_IO*	R	~~The value of this register is derived from the IC_INTR_IO coreConsultant parameter.~~ ■ ~~0: Individual~~ ■ 1: Combined
4	*HC_COUNT_VALUES*	R	~~The value of this register is derived from the IC_HC_COUNT_VALUES coreConsultant parameter.~~ ■ 0: False ■ ~~1: True~~
3:2	*MAX_SPEED_MODE*	R	~~The value of this register is derived from the IC_MAX_SPEED_MODE coreConsultant parameter~~ ■ ~~0x0 = Reserved~~ ■ ~~0x1 = Standard~~ ■ ~~0x2 = Fast~~ ■ 0x3 = High

Bits	Name	R/W	Description
1:0	*APB_DATA_WIDTH*	R	~~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~~

~~12.4.3.44 IC_COMP_VERSION~~

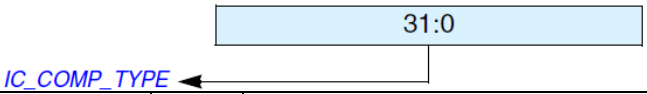
- **Name:** I2C Component Version Register
- **Size:** 32 bits
- **Address Offset:** 0xf8
- **Read/Write Access:** Read



Bits	Name	R/W	Description
31:0	IC_COMP_VERSION	R	Specific values for this register are described in the Releases Table in the AMBA 2 release notes

~~12.4.3.45 IC_COMP_TYPE~~

- **Name:** I2C Component Type Register
- **Size:** 32 bits
- **Address Offset:** 0xfc
- **Read/Write Access:** Read



Bits	Name	R/W	Description
31:0	IC_COMP_TYPE	R	Type number = 0x44_57_01_40. This assigned unique hex value is constant and is derived from the two ASCII letters “DW” followed by a 16-bit unsigned number.

13 SPI

SPI_User_manual

14 SSI

14.1 Function Description

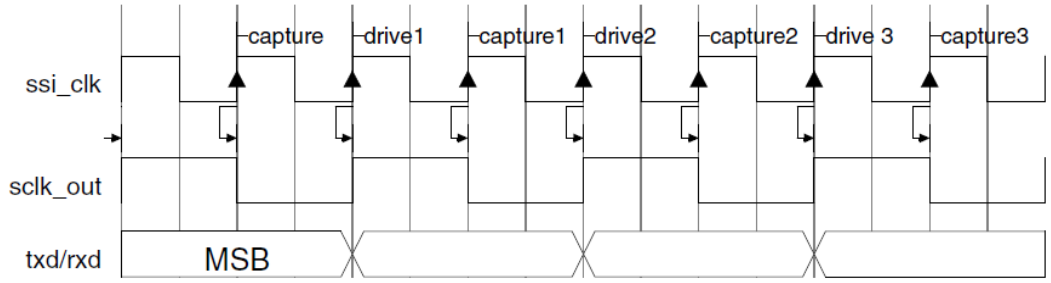
14.1.1 Clock Ratio

The frequency of the DW_apb_ssi serial input clock (ssi_clk) must be less than or equal to the frequency of pclk, which guarantees that control signals from the ssi_clk domain are synchronized to the pclk domain. When pclk and ssi_clk are asynchronous, synchronization logic transfers control signals from one clock domain to the other. If both clocks are synchronous, the synchronization logic is unneeded, thus reducing latency in the peripheral.

~~When~~ DW_apb_ssi is ~~configured as~~ a master ~~device~~, the maximum frequency of the bit-rate clock (sclk_out) is one-half the frequency of ssi_clk. This allows the shift control logic to capture data on one clock edge of sclk_out and propagate data on the opposite edge.

Figure 3-2 on page 32 illustrates the maximum ratio between sclk_out and ssi_clk.

Maximum sclk_out/ssi_clk Ratio



The sclk_out line toggles only when an active transfer is in progress. At all other times it is held in an inactive state, as defined by the serial protocol under which it operates.
The frequency of sclk_out can be derived from the following equation:

$$F_{sclk\_out} = \frac{F_{ssi\_clk}}{SCKDV}$$

SCKDV is a bit field in the programmable register BAUDR, holding any even value in the range 0 to 65,534. If SCKDV is 0, then sclk_out is disabled.

The DW_apb_gpio is a programmable General Purpose Programming I/O (GPIO) peripheral. The component is an AMBA 2.0-compliant Advanced Peripheral Bus (APB) slave device.
Following functional groupings of the main interfaces to the DW_apb_gpio block:

14.1.2 Transmit and Receive FIFO Buffers

The FIFO buffers used by the DW_apb_ssi are internal D-type flip-flops that can be configured in depth between 2 to 256. The width of both transmit and receive FIFO buffers is fixed at 16 bits due to the serial specifications, which state that a serial transfer (data frame) can be 4 to 16 bits in length. Data frames that are
less than 16 bits in size must be right-justified when written into the transmit FIFO buffer. The shift control logic automatically right-justifies receive data in the receive FIFO buffer.

Each data entry in the FIFO buffers contains a single data frame. It is impossible to store multiple data frames in a single FIFO location; for example, you may not store two 8-bit data frames in a single FIFO location. If an 8-bit data frame is required, the upper 8-bits of the FIFO entry are ignored or unused when the serial shifter transmits the data.

The transmit FIFO is loaded by APB write commands to the DW_apb_ssi data register (DR). Data are popped (removed) from the transmit FIFO by the shift control logic into the transmit shift register. The transmit FIFO generates a FIFO empty interrupt request (ssi_txe_intr) when the number of entries in the FIFO is less than or equal to the FIFO threshold value. The threshold value, set through the programmable register TXFTLR, determines the level of FIFO entries at which an interrupt is generated. The threshold value allows you to provide early indication to the processor that the transmit FIFO is nearly empty. A transmit FIFO overflow interrupt (ssi_txo_intr) is generated if you attempt to write data into an already full transmit FIFO.

Data are popped from the receive FIFO by APB read commands to the DW_apb_ssi data register (DR). The receive FIFO is loaded from the receive shift register by the shift control logic. The receive FIFO generates a FIFO-full interrupt request (ssi_rxf_intr) when the number of entries in the FIFO is greater than or equal to the FIFO threshold value plus 1. The threshold value, set through programmable register RXFTLR, determines the level of FIFO entries at which an interrupt is generated.

The threshold value allows you to provide early indication to the processor that the receive FIFO is nearly full. A receive FIFO overrun interrupt (ssi_rxo_intr) is generated when the receive shift logic attempts to load data into a completely full receive FIFO. However, this newly received data are lost. A receive FIFO underflow interrupt (ssi_rxu_intr) is generated if you attempt to read from an empty receive FIFO. This alerts the processor that the read data are invalid.

14.1.3 Interrupts

The DW_apb_ssi supports combined and individual interrupt requests, each of which can be masked. The combined interrupt request is the ORed result of all other DW_apb_ssi interrupts after masking. The system designer has the choice of routing individual interrupt requests or only the combined interrupt request to the Interrupt Controller. All DW_apb_ssi interrupts are level interrupts and have the same active polarity level; you can configure this polarity level as active-high or active-low.
The DW_apb_ssi interrupts are described as follows:

- Transmit FIFO Empty Interrupt (ssi_txe_intr) – Set when the transmit FIFO is equal to or below its threshold value and requires service to prevent an under-run. The threshold value, set through a software-programmable register, determines the level of transmit FIFO entries at which an interrupt is generated. This interrupt is cleared by hardware when data are written into the transmit FIFO buffer, bringing it over the threshold level.
- Transmit FIFO Overflow Interrupt (ssi_txo_intr) – Set when an APB access attempts to write into the transmit FIFO after it has been completely filled. When set, data written from the APB is discarded. This interrupt remains set until you read the transmit FIFO overflow interrupt clear register (TXOICR).

- Receive FIFO Full Interrupt (ssi_rxf_intr) – Set when the receive FIFO is equal to or above its threshold value plus 1 and requires service to prevent an overflow. The threshold value, set through a software-programmable register, determines the level of receive FIFO entries at which an interrupt is generated. This interrupt is cleared by hardware when data are read from the receive FIFO buffer, bringing it below the threshold level.
- Receive FIFO Overflow Interrupt (ssi_rxo_intr) – Set when the receive logic attempts to place data into the receive FIFO after it has been completely filled. When set, newly received data are discarded. This interrupt remains set until you read the receive FIFO overflow interrupt clear register (RXOICR).
- Receive FIFO Underflow Interrupt (ssi_rxu_intr) – Set when an APB access attempts to read from the receive FIFO when it is empty. When set, zeros are read back from the receive FIFO. This interrupt remains set until you read the receive FIFO underflow interrupt clear register (RXUICR).
- Multi-Master Contention Interrupt (ssi_mst_intr) – Present only when the DW_apb_ssi component is configured as a serial-master device. The interrupt is set when another serial master on the serial bus selects the DW_apb_ssi master as a serial-slave device and is actively transferring data. This informs the processor of possible contention on the serial bus. This interrupt remains set until you read the multi-master interrupt clear register (MSTICR).
- Combined Interrupt Request (ssi_intr) – OR'ed result of all the above interrupt requests after masking. To mask this interrupt signal, you must mask all other DW_apb_ssi interrupt requests.

14.1.4 Transfer Modes

When transferring data on the serial bus, the DW_apb_ssi operates in the modes discussed in this section.

14.1.4.1 Transmit and Receive

When TMOD = 2'b00, both transmit and receive logic are valid. The data transfer occurs as normal according to the selected frame format (serial protocol). Transmit data are popped from the transmit FIFO and sent through the txd line to the target device, which replies with data on the rxd line. The receive data from the target device is moved from the receive shift register into the receive FIFO at the end of each data frame.

14.1.4.2 Transmit Only

When TMOD = 2'b01, the receive data are invalid and should not be stored in the receive FIFO. The data transfer occurs as normal, according to the selected frame format (serial protocol). Transmit data are popped from the transmit FIFO and sent through the txd line to the target device, which replies with data on the rxd line. At the end of the data frame, the receive shift register does not load its newly received data into the receive FIFO. The data in the receive shift register is overwritten by the next transfer. You should mask interrupts originating from the receive logic when this mode is entered.

14.1.4.3 Receive Only

When TMOD = 2'b10, the transmit data are invalid. When configured as a slave, the transmit FIFO is never popped in Receive Only mode. The txd output remains at a constant logic level during the transmission. The data transfer occurs as normal according to the selected frame format (serial protocol). The receive data
from the target device is moved from the receive shift register into the receive FIFO at the end of each data frame. You should mask interrupts originating from the transmit logic when this mode is entered.

14.1.4.4 EEPROM Read

When TMOD = 2'b11, the transmit data is used to transmit an opcode and/or an address to the EEPROM device. Typically this takes three data frames (8-bit opcode followed by 8-bit upper address and 8-bit lower address). During the transmission of the opcode and address, no data is captured by the receive logic (as long as the DW_apb_ssi master is transmitting data on its txd line, data on the rxd line is ignored). The DW_apb_ssi master continues to transmit data until the transmit FIFO is empty. Therefore, you should ONLY have enough data frames in the transmit FIFO to supply the opcode and address to the EEPROM. If more data frames are in the transmit FIFO than are needed, then read data is lost.

When the transmit FIFO becomes empty (all control information has been sent), data on the receive line (rxd) is valid and is stored in the receive FIFO; the txd output is held at a constant logic level. The serial

transfer continues until the number of data frames received by the DW_apb_ssi master matches the value of the NDF field in the CTRLR1 register + 1.

14.1.5 Master SPI Serial Transfers

When the transfer mode is “transmit and receive” or “transmit only” (TMOD = 2'b00 or TMOD = 2'b01, respectively), transfers are terminated by the shift control logic when the transmit FIFO is empty. For continuous data transfers, you must ensure that the transmit FIFO buffer does not become empty before all the data have been transmitted. The transmit FIFO threshold level (TXFTLR) can be used to early interrupt (ssi_txe_intr) the processor indicating that the transmit FIFO buffer is nearly empty.

When the transfer mode is “receive only” (TMOD = 2'b10), a serial transfer is started by writing one “dummy” data word into the transmit FIFO when a serial slave is selected. The txd output from the DW_apb_ssi is held at a constant logic level for the duration of the serial transfer. The transmit FIFO is popped only once at the beginning and may remain empty for the duration of the serial transfer. The end of the serial transfer is controlled by the “number of data frames” (NDF) field in control register 1 (CTRLR1).

If, for example, you want to receive 24 data frames from a serial-slave peripheral, you should program the NDF field with the value 23; the receive logic terminates the serial transfer when the number of frames received is equal to the NDF value + 1. This transfer mode increases the bandwidth of the APB bus as the transmit FIFO never needs to be serviced during the transfer. The receive FIFO buffer should be read each time the receive FIFO generates a FIFO full interrupt request to prevent an overflow.

When the transfer mode is “eeprom_read” (TMOD = 2‘b11), a serial transfer is started by writing the opcode and/or address into the transmit FIFO when a serial slave (EEPROM) is selected. The opcode and address are transmitted to the EEPROM device, after which read data is received from the EEPROM device and stored in the receive FIFO. The end of the serial transfer is controlled by the NDF field in the control register 1 (CTRLR1).

The receive FIFO threshold level (RXFTLR) can be used to give early indication that the receive FIFO is nearly full.

A typical software flow for completing an SPI serial transfer from the DW_apb_ssi serial master is outlined as follows:

1. If the DW_apb_ssi is enabled, disable it by writing 0 to the SSI Enable register (SSIENR).
2. Set up the DW_apb_ssi control registers for the transfer; these registers can be set in any order.
   - Write Control Register 0 (CTRLR0). For SPI transfers, the serial clock polarity and serial clock phase parameters must be set identical to target slave device.
   - If the transfer mode is *receive only*, write CTRLR1 (Control Register 1) with the number of frames in the transfer minus 1; for example, if you want to receive four data frames, write this register with 3.
   - Write the Baud Rate Select Register (BAUDR) to set the baud rate for the transfer.
   - Write the Transmit and Receive FIFO Threshold Level registers (TXFTLR and RXFTLR, respectively) to set FIFO threshold levels.
   - Write the IMR register to set up interrupt masks.
   - The Slave Enable Register (SER) register can be written here to enable the target slave for selection. If a slave is enabled here, the transfer begins as soon as one valid data entry is present in the transmit FIFO. If no slaves are enabled prior to writing to the Data Register (DR), the transfer does not begin until a slave is enabled.
3. Enable the DW_apb_ssi by writing 1 to the SSIENR register.
4. Write data for transmission to the target slave into the transmit FIFO (write DR).

If no slaves were enabled in the SER register at this point, enable it now to begin the transfer.

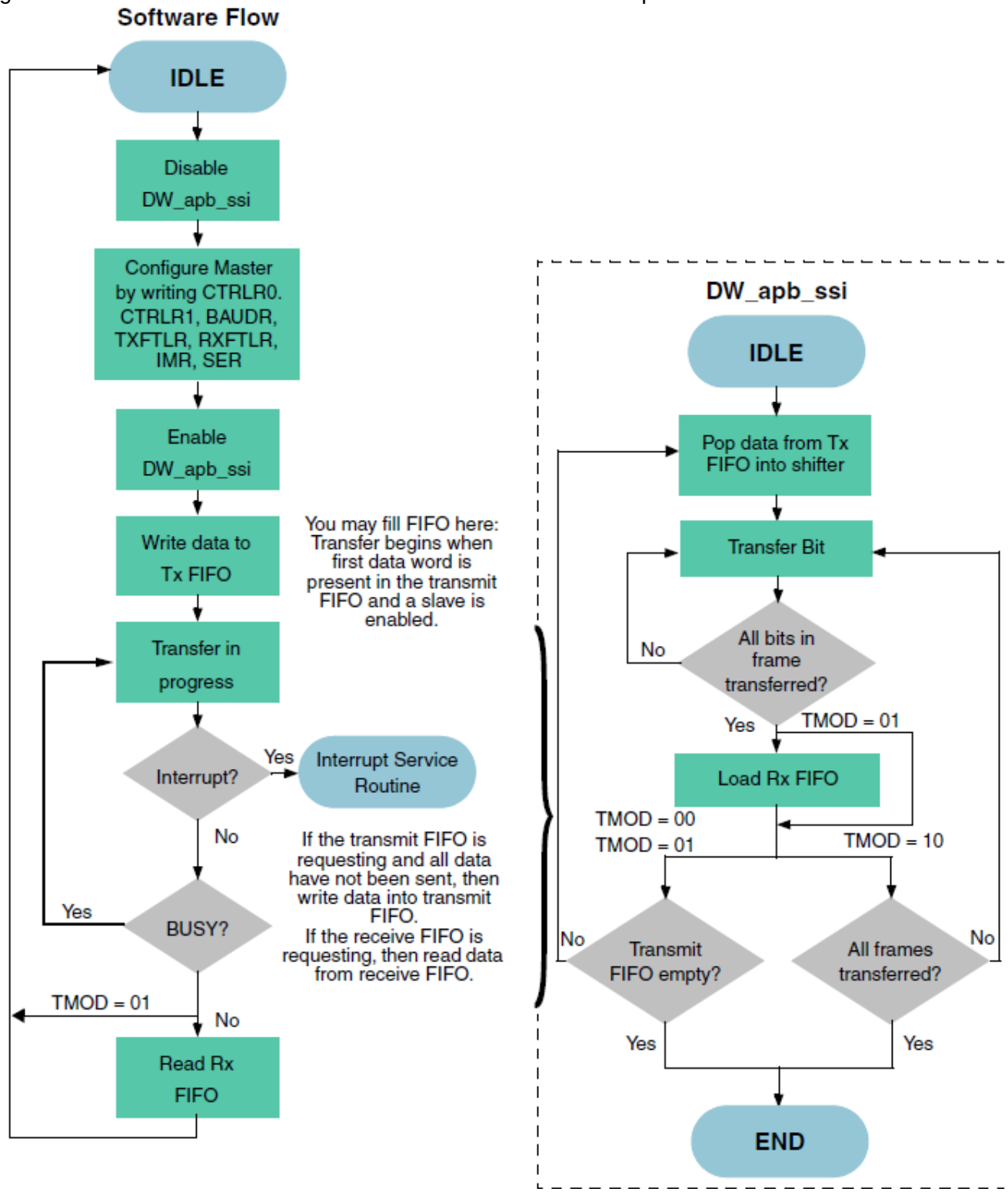
5. Poll the BUSY status to wait for completion of the transfer. The BUSY status cannot be polled immediately; for more information, see the note on page 40.

If a transmit FIFO empty interrupt request is made, write the transmit FIFO (write DR). If a receive FIFO full interrupt request is made, read the receive FIFO (read DR).

6. The transfer is stopped by the shift control logic when the transmit FIFO is empty. If the transfer mode is *receive only* (TMOD = 2'b10), the transfer is stopped by the shift control logic when the

specified number of frames have been received. When the transfer is done, the BUSY status is reset to 0.
7. If the transfer mode is not *transmit only* (TMOD != 01), read the receive FIFO until it is empty.
8. Disable the DW_apb_ssi by writing 0 to SSIENR.

Figure 3-8 shows a typical software flow for starting a DW_apb_ssi master SPI serial transfer. The diagram also shows the hardware flow inside the serial-master component.



14.2 SSI Registers

14.2.1 Register Memory Map

Name	Address Offset	Width	Description
CTRLR0	0x0	16 bits	Control Register 0 **Reset Value:** Configuration Dependent for some bit fields
CTRLR1	0x04	16 bits	Control Register 1 **Reset Value:** 0x0
SSIENR	0x08	1 bit	SSI Enable Register **Reset Value:** 0x0
MWCR	0x0C	3 bits	Microwire Control Register **Reset Value:** 0x0
SER	0x10	See Description	Slave Enable Register **Width:** *SSI_NUM_SLAVES* **Reset Value:** 0x0
BAUDR	0x14	16 bits	Baud Rate Select **Reset Value:** 0x0
TXFTLR	0x18	*TX_ABW*	Transmit FIFO Threshold Level **Reset Value:** 0x0
RXFTLR	0x1C	*RX_ABW*	Receive FIFO Threshold Level **Reset Value:** 0x0
TXFLR	0x20	See Description	Transmit FIFO Level Register **Width:** *TX_ABW*+1 **Reset Value:** 0x0
RXFLR	0x24	See Description	Receive FIFO Level Register **Width:** *RX_ABW*+1 **Reset Value:** 0x0
SR	0x28	7 bits	Status Register **Reset Value:** 0x6
IMR	0x2C	See Description	Interrupt Mask Register **Width:** 6 bits: when SSI_IS_MASTER =1) 5 bits: when SSI_IS_MASTER = 0) **Reset Value:** 0x3F/0x1F
ISR	0x30	6 bits	Interrupt Status Register **Reset Value:** 0x0
RISR	0x34	6 bits	Raw Interrupt Status Register **Reset Value:** 0x0
TXOICR	0x38	1 bit	Transmit FIFO Overflow Interrupt Clear Register **Reset Value:** 0x0
RXOICR	0x3C	1 bit	Receive FIFO Overflow Interrupt Clear Register **Reset Value:** 0x0
RXUICR	0x40	1 bit	Receive FIFO Underflow Interrupt Clear Register **Reset Value:** 0x0
MSTICR	0x44	1 bit	Multi-Master Interrupt Clear Register **Reset Value:** 0x0
ICR	0x48	1 bit	Interrupt Clear Register **Reset Value:** 0x0
DMACR	0x4C	2 bits	DMA Control Register **Reset Value:** 0x0
DMATDLR	0x50	*TX_ABW*	DMA Transmit Data Level **Reset Value:** 0x0
DMARDLR	0x54	*RX_ABW*	DMA Receive Data Level **Reset Value:** 0x0
IDR	0x58	32 bits	Identification Register **Reset Value:** Not affected by reset
SSI_COMP_VERSION	0x5C	32 bits	coreKit version ID register **Reset Value:** See the releases table in the AMBA 2 release notes
DR	0x60 - 0xec	16 bits	Data Register **Reset Value:** 0x0
RX_SAMPLE_DLY	0xf0	8 bits	RXD Sample Delay Register **Reset Value:** 0x0

Name	Address Offset	Width	Description
~~RSVD_0~~	~~0xf4~~	~~32 bits~~	~~Reserved location for future use~~ ~~**Reset Value:** Writes have no effect; reads return value of 0~~
~~RSVD_1~~	~~0xf8~~	~~32 bits~~	~~Reserved location for future use~~ ~~**Reset Value:** Writes have no effect; reads return value of 0~~
~~RSVD_2~~	~~0xfc~~	~~32 bits~~	~~Reserved location for future use~~ ~~**Reset Value:** Writes have no effect; reads return value of 0~~

14.2.2 Registers and Field Descriptions

14.2.2.1 CTRLR0

- **Name:** Control Register 0
- **Size:** 16 bits
- **Address Offset:** 0x0
- **Read/write access:** read/write

This register controls the serial data transfer. It is impossible to write to this register when the DW_apb_ssi is enabled. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.

Bits	Name	R/W	Description
15:12	~~CFS~~ reserved	~~R/W~~	~~Control Frame Size. Selects the length of the control word for the Microwire frame format.~~ ~~**Reset Value:** 0x0~~
11	SRL	R/W	Shift Register Loop. Used for testing purposes only. When internally active, connects the transmit shift register output to the receive shift register input. Can be used in both serialslave and serial-master modes. 0 – Normal Mode Operation 1 – Test Mode Operation ~~When the DW_apb_ssi is configured as a slave in loopback mode, the ss_in_n and ssi_clk signals must be provided by an external source. In this mode, the slave cannot generate these signals because there is nothing to which to loop back.~~ **Reset Value:** 0x0
10	~~SLV_OE~~ reserved	~~R/W~~	~~Slave Output Enable. Relevant only when the DW_apb_ssi is configured as a serial-slave device. When configured as a serial master, this bit field has no functionality. This bit enables or disables the setting of the ssi_oe_n output from the DW_apb_ssi serial slave. When SLV_OE = 1, the ssi_oe_n output can never be active. When the ssi_oe_n output controls the tri-state buffer on the txd output from the slave, a high impedance state is always present on the slave txd output when SLV_OE = 1.~~ ~~This is useful when the master transmits in broadcast mode (master transmits data to all slave devices). Only one slave may respond with data on the master rxd line. This bit is enabled after reset and must be disabled by software (when broadcast mode is used), if you do not want this device to respond with data.~~ ~~0 – Slave txd is enabled~~ ~~1 – Slave txd is disabled~~ ~~**Reset Value:** 0x0~~

Bits	Name	R/W	Description
9:8	TMOD	R/W	Transfer Mode. Selects the mode of transfer for serial communication. This field does not affect the transfer duplicity. Only indicates whether the receive or transmit data are valid. In transmit-only mode, data received from the external device is not valid and is not stored in the receive FIFO memory; it is overwritten on the next transfer. In receive-only mode, transmitted data are not valid. After the first write to the transmit FIFO, the same word is retransmitted for the duration of the transfer. In transmit-and-receive mode, both transmit and receive data are valid. The transfer continues until the transmit FIFO is empty. Data received from the external device are stored into the receive FIFO memory, where it can be accessed by the host processor. In eeprom-read mode, receive data is not valid while control data is being transmitted. When all control data is sent to the EEPROM, receive data becomes valid and transmit data becomes invalid. All data in the transmit FIFO is considered control data in this mode. This transfer mode is only valid when the DW_apb_ssi is configured as a master device. 00 –- Transmit & Receive 01 –- Transmit Only 10 –- Receive Only 11 –- EEPROM Read **Reset Value:** 0x0
7	SCPOL	R/W	Serial Clock Polarity. Valid when the frame format (FRF) is set to Motorola SPI. Used to select the polarity of the inactive serial clock, which is held inactive when the DW_apb_ssi master is not actively transferring data on the serial bus. 0 – Inactive state of serial clock is low 1 – Inactive state of serial clock is high ~~**Dependencies:** When SSI_HC_FRF=1, SCPOL bit is a read-only bit with its value set by SSI_DFLT_SCPOL.~~ **Reset Value:** ~~SSI_DFLT_SCPOL~~0x0
6	SCPH	R/W	Serial Clock Phase. Valid when the frame format (FRF) is set to Motorola SPI. The serial clock phase selects the relationship of the serial clock with the slave select signal. When SCPH = 0, data are captured on the first edge of the serial clock. When SCPH = 1, the serial clock starts toggling one cycle after the slave select line is activated, and data are captured on the second edge of the serial clock. 0: Serial clock toggles in middle of first data bit 1: Serial clock toggles at start of first data bit ~~**Dependencies:** When SSI_HC_FRF=1, SCPH bit is a read-only bit, with its value set by SSI_DFLT_SCPH.~~ **Reset Value:** ~~SSI_DFLT_SCPH~~ 0x0
5:4	FRF	R/W	Frame Format. Selects which serial protocol transfers the data. 00 –- Motorola SPI ~~01 –- Texas Instruments SSP~~ ~~10 –- National Semiconductors Microwire~~ ~~11 –- Reserved~~ ~~**Dependencies:** When SSI_HC_FRF=1, FRF is read-only and its value is set by SSI_DFLT_FRF.~~ **Reset Value:** ~~SSI_DFLT_FRF~~ 0x00
3:0	DFS	R/W	Data Frame Size. Selects the data frame length. When the data frame size is programmed to be less than 16 bits, the receive data are automatically right-justified by the receive logic, with the upper bits of the receive FIFO zero-padded. You must right-justify transmit data before writing into the transmit FIFO. The transmit logic ignores the upper unused bits when transmitting the data. **Reset Value:** 0x7

DFS Decode

DFS Value	Description
0000	Reserved
0001	Reserved
0010	Reserved
0011	4-bit serial data transfer
0100	5-bit serial data transfer
0101	6-bit serial data transfer
0110	7-bit serial data transfer
0111	8-bit serial data transfer
1000	9-bit serial data transfer
1001	10-bit serial data transfer
1010	11-bit serial data transfer
1011	12-bit serial data transfer
1100	13-bit serial data transfer
1101	14-bit serial data transfer
1110	15-bit serial data transfer
1111	16-bit serial data transfer

~~CFS Decode~~

DFS Value	Description
0000	1-bit serial data transfer
0001	2-bit serial data transfer
0010	3-bit serial data transfer
0011	4-bit serial data transfer
0100	5-bit serial data transfer
0101	6-bit serial data transfer
0110	7-bit serial data transfer
0111	8-bit serial data transfer
1000	9-bit serial data transfer
1001	10-bit serial data transfer
1010	11-bit serial data transfer
1011	12-bit serial data transfer
1100	13-bit serial data transfer
1101	14-bit serial data transfer
1110	15-bit serial data transfer
1111	16-bit serial data transfer

14.2.2.2 CTRLR1

- **Name:** Control Register 1
- **Size:** 16 bits
- **Address Offset:** 0x04
- **Read/write access:** read/write

Control register 1 controls the end of serial transfers when in receive-only mode. It is impossible to write to this register when the DW_apb_ssi is enabled. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.

Bits	Name	R/W	Description
15:0	NDF	R/W	Number of Data Frames. When TMOD = 10 or TMOD = 11, this register field sets the number of data frames to be continuously received by the DW_apb_ssi. The DW_apb_ssi continues to receive serial data until the number of data frames received is equal to this register value plus 1, which enables you to receive up to 64 KB of data in a continuous transfer. **Reset Value:** 0x0

14.2.2.3 SSIENR

- **Name:** SSI Enable Register
- **Size:** 1 bit
- **Address Offset:** 0x08
- **Read/write access:** read/write

This register enables and disables the DW_apb_ssi.

Bits	Name	R/W	Description
0	SSI_EN	R/W	SSI Enable. Enables and disables all DW_apb_ssi operations. When disabled, all serial transfers are halted immediately. Transmit and receive FIFO buffers are cleared when the device is disabled. It is impossible to program some of the DW_apb_ssi control registers when enabled. When disabled, the ssi_sleep output is set (after delay) to inform the system that it is safe to remove the ssi_clk, thus saving power consumption in the system. **Reset Value:** 0x0

~~14.2.2.4 MWCR~~

- **Name:** Microwire Control Register
- **Size:** 3 bits
- **Address Offset:** 0x0C
- **Read/write access:** read/write

This register controls the direction of the data word for the half-duplex Microwire serial protocol.

14.2.2.5 SER

- **Name:** Slave Enable Register
- **Size:** *SSI_NUM_SLAVES*
- **Address Offset:** 0x10
- **Read/write access:** read/write

Bits	Name	R/W	Description
31:1	Reserved		
0	SER	R/W	Slave Select Enable Flag. Each bit in this register corresponds to a slave select line from the DW_apb_ssi master. When the bit in this register is set (1), the ~~corresponding~~ slave select line from the master is activated when a serial transfer begins. It should be noted that setting or clearing bits in this register have no effect on the corresponding slave select outputs until a transfer is started. Before beginning a transfer, you should enable the bit in this register that corresponds to the slave device with which the master wants to communicate. When not operating in broadcast mode, only one bit in this field should be set. 1: Selected 0: Not Selected **Reset Value:** 0x0

14.2.2.6 BAUDR

- **Name:** Baud Rate Select
- **Size:** 16 bits
- **Address Offset:** 0x14
- **Read/write access:** read/write

The register derives the frequency of the serial clock that regulates the data transfer. The 16-bit field in this register defines the ssi_clk divider value. It is impossible to write to this register when the DW_apb_ssi is enabled. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.

Bits	Name	R/W	Description

Bits	Name	R/W	Description
15:0	SCKDV	R/W	SSI Clock Divider. The LSB for this field is always set to 0 and is unaffected by a write operation, which ensures an even value is held in this register. If the value is 0, the serial output clock (sclk_out) is disabled. The frequency of the sclk_out is derived from the following equation: $F_{sclk\_out} = \frac{F_{ssi\_clk}}{SCKDV}$ where SCKDV is any even value between 2 and 65534. For example: for Fssi_clk = 3.6864MHz and SCKDV =2 Fsclk out = 3.6864/2 = 1.8432MHz **Reset Value:** 0x0

14.2.2.7 TXFTLR

- **Name:** Transmit FIFO Threshold Level
- **Size:** ~~*TX_ABW*~~ *3 bits*
- **Address Offset:** 0x18
- **Read/write access:** read/write

This register controls the threshold value for the transmit FIFO memory. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.

Bits	Name	R/W	Description
31:3	Reserved		
2:0	TFT	R/W	Transmit FIFO Threshold. Controls the level of entries (or below) at which the transmit FIFO controller triggers an interrupt. ~~The FIFO depth is configurable in the range 2-256;~~ this register is sized to the number of address bits needed to access the FIFO. ~~If you attempt to set bits [7:0] of this register to a value greater than or equal to the depth of the FIFO, this field is not written and retains its current value. When the number of transmit FIFO entries is less than or equal to this value, the transmit FIFO empty interrupt is triggered. For field decode, refer to Table 6-4.~~ **Reset Value:** 0x0

TFT Decode

TFT Value	Description
0000	ssi_txe_intr is asserted when 0 data entries are present in transmit FIFO
0001	ssi_txe_intr is asserted when 1 data entries are present in transmit FIFO
0010	ssi_txe_intr is asserted when 2 data entries are present in transmit FIFO
0011	ssi_txe_intr is asserted when 3 data entries are present in transmit FIFO
0100	ssi_txe_intr is asserted when 4 data entries are present in transmit FIFO
0101	ssi_txe_intr is asserted when 5 data entries are present in transmit FIFO
0110	ssi_txe_intr is asserted when 6 data entries are present in transmit FIFO
0111	ssi_txe_intr is asserted when 7 data entries are present in transmit FIFO

14.2.2.8 RXFTLR

- **Name:** Receive FIFO Threshold Level
- **Size:** ~~*RX_ABW*~~ *3 bits*
- **Address Offset:** 0x1C
- **Read/write access:** read/write

This register controls the threshold value for the receive FIFO memory. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.

Bits	Name	R/W	Description
31:3	Reserved		

Bits	Name	R/W	Description
2:0	RFT	R/W	Receive FIFO Threshold. Controls the level of entries (or above) at which the receive FIFO controller triggers an interrupt. ~~The FIFO depth is configurable in the range 2-256.~~ This register is sized to the number of address bits needed to access the FIFO. If you attempt to set this value greater than the depth of the FIFO, this field is not written and retains its current value. When the number of receive FIFO entries is greater than or equal to this value + 1, the receive FIFO full interrupt is triggered. ~~For field decode, refer to Table 6-5.~~ **Reset Value:** 0x0

14.2.2.9 TXFLR

- **Name:** Transmit FIFO Level Register
- **Size:** ~~*TX_ABW* + 1~~ 4 bits
- **Address Offset:** 0x20
- **Read/write access:** read-only

This register contains the number of valid data entries in the transmit FIFO memory.

Bits	Name	R/W	Description
31:4	Reserved		
3:0	TXTFL	R	Transmit FIFO Level. Contains the number of valid data entries in the transmit FIFO. **Reset Value:** 0x0

14.2.2.10 RXFLR

- **Name:** Receive FIFO Level Register
- **Size:** ~~*RX_ABW* + 1~~ 4 bits
- **Address Offset:** 0x24
- **Read/write access:** read-only

This register contains the number of valid data entries in the receive FIFO memory. This register can be read
at any time.

Bits	Name	R/W	Description
31:4	Reserved		
3:0	RXTFL	R	Receive FIFO Level. Contains the number of valid data entries in the receive FIFO. **Reset Value:** 0x0

14.2.2.11 RXFLR

- **Name:** Status Register
- **Size:** 7 bits
- **Address Offset:** 0x28
- **Read/write access:** read-only

This is a read-only register used to indicate the current transfer status, FIFO status, and any transmission/reception errors that may have occurred. The status register may be read at any time. None of the bits in this register request an interrupt.

Bits	Name	R/W	Description
6	DCOL	R	Data Collision Error. ~~Relevant only when the DW_apb_ssi is configured as a master device.~~ This bit is set if the DW_apb_ssi master is actively transmitting when another master selects this device as a slave. This informs the processor that the last data transfer was halted before completion. This bit is cleared when read. 0 – No error 1 – Transmit data collision error **Reset Value:** 0x0

Bits	Name	R/W	Description
5	~~TXE~~	~~R~~	~~Transmission Error. Set if the transmit FIFO is empty when a transfer is started. This bit can be set only when the DW_apb_ssi is configured as a slave device. Data from the previous transmission is resent on the txd line. This bit is cleared when read. 0 – No error 1 – Transmission error~~ **Reset Value:** ~~0x0~~
4	RFF	R	Receive FIFO 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. 0 – Receive FIFO is not full 1 – Receive FIFO is full **Reset Value:** 0x0
3	RFNE	R	Receive FIFO Not Empty. Set when the receive FIFO contains one or more entries and is cleared when the receive FIFO is empty. This bit can be polled by software to completely empty the receive FIFO. 0 – Receive FIFO is empty 1 – Receive FIFO is not empty **Reset Value:** 0x0
2	TFE	R	Transmit FIFO Empty. When the transmit FIFO is completely empty, this bit is set. When the transmit FIFO contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. 0 – Transmit FIFO is not empty 1 – Transmit FIFO is empty **Reset Value:** 0x1
1	TFNF	R	Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. 0 – Transmit FIFO is full 1 – Transmit FIFO is not full **Reset Value:** 0x1
0	BUSY	R	SSI Busy Flag. When set, indicates that a serial transfer is in progress; when cleared indicates that the DW_apb_ssi is idle or disabled. 0 – DW_apb_ssi is idle or disabled 1 – DW_apb_ssi is actively transferring data **Reset Value:** 0x0

14.2.2.12 IMR

- **Name:** Interrupt Mask Register
- Size:
  6 bits: when SSI_IS_MASTER =1)
  5 bits: when SSI_IS_MASTER = 0)
- **Address Offset:** 0x2C
- **Read/write access:** read/write

Bits	Name	R/W	Description
31:6	Reserved		
5	MSTIM	R/W	Multi-Master Contention Interrupt Mask. ~~This bit field is not present if the DW_apb_ssi is configured as a serial-slave device.~~ 0 – ssi_mst_intr interrupt is masked 1 – ssi_mst_intr interrupt is not masked **Reset Value:** 0x1

Bits	Name	R/W	Description
4	RXFIM	R/W	Receive FIFO Full Interrupt Mask 0 – ssi_rxf_intr interrupt is masked 1 – ssi_rxf_intr interrupt is not masked **Reset Value:** 0x1
3	RXOIM	R/W	Receive FIFO Overflow Interrupt Mask 0 – ssi_rxo_intr interrupt is masked 1 – ssi_rxo_intr interrupt is not masked **Reset Value:** 0x1
2	RXUIM	R/W	Receive FIFO Underflow Interrupt Mask 0 – ssi_rxu_intr interrupt is masked 1 – ssi_rxu_intr interrupt is not masked **Reset Value:** 0x1
1	TXOIM	R/W	Transmit FIFO Overflow Interrupt Mask 0 – ssi_txo_intr interrupt is masked 1 – ssi_txo_intr interrupt is not masked **Reset Value:** 0x1
0	TXEIM	R/W	Transmit FIFO Empty Interrupt Mask 0 – ssi_txe_intr interrupt is masked 1 – ssi_txe_intr interrupt is not masked **Reset Value:** 0x1

14.2.2.13 ISR

- **Name:** Interrupt Status Register
- **Size:** 6 bits
- **Address Offset:** 0x30
- **Read/write access:** read-only

This register reports the status of the DW_apb_ssi interrupts after they have been masked.

Bits	Name	R/W	Description
31:6	Reserved		
5	MSTIS	R/W	Multi-Master Contention Interrupt Status. ~~This bit field is not present if the DW_apb_ssi is configured as a serial-slave device~~. 0 = ssi_mst_intr interrupt not active after masking 1 = ssi_mst_intr interrupt is active after masking **Reset Value:** 0x0
4	RXFIS	R/W	Receive FIFO Full Interrupt Status 0 = ssi_rxf_intr interrupt is not active after masking 1 = ssi_rxf_intr interrupt is full after masking **Reset Value:** 0x0
3	RXOIS	R/W	Receive FIFO Overflow Interrupt Status 0 = ssi_rxo_intr interrupt is not active after masking 1 = ssi_rxo_intr interrupt is active after masking **Reset Value:** 0x0
2	RXUIS	R/W	Receive FIFO Underflow Interrupt Status 0 = ssi_rxu_intr interrupt is not active after masking 1 = ssi_rxu_intr interrupt is active after masking **Reset Value:** 0x0
1	TXOIS	R/W	Transmit FIFO Overflow Interrupt Status 0 = ssi_txo_intr interrupt is not active after masking 1 = ssi_txo_intr interrupt is active after masking **Reset Value:** 0x0
0	TXEIS	R/W	Transmit FIFO Empty Interrupt Status 0 = ssi_txe_intr interrupt is not active after masking 1 = ssi_txe_intr interrupt is active after masking **Reset Value:** 0x0

14.2.2.14 RISR

- **Name:** Raw Interrupt Status Register
- **Size:** 32 bits
- **Address Offset:** 0x34

- **Read/write access:** read-only

This read-only register reports the status of the DW_apb_ssi interrupts prior to masking.

Bits	Name	R/W	Description
31:6	Reserved		
5	MSTIR	R/W	Multi-Master Contention Raw Interrupt Status. ~~This bit field is not present if the DW_apb_ssi is configured as a serial-slave device~~. 0 = ssi_mst_intr interrupt is not active prior to masking 1 = ssi_mst_intr interrupt is active prior masking **Reset Value:** 0x0
4	RXFIR	R/W	Receive FIFO Full Raw Interrupt Status 0 = ssi_rxf_intr interrupt is not active prior to masking 1 = ssi_rxf_intr interrupt is active prior to masking **Reset Value:** 0x0
3	RXOIR	R/W	Receive FIFO Overflow Raw Interrupt Status 0 = ssi_rxo_intr interrupt is not active prior to masking 1 = ssi_rxo_intr interrupt is active prior masking **Reset Value:** 0x0
2	RXUIR	R/W	Receive FIFO Underflow Raw Interrupt Status 0 = ssi_rxu_intr interrupt is not active prior to masking 1 = ssi_rxu_intr interrupt is active prior to masking **Reset Value:** 0x0
1	TXOIR	R/W	Transmit FIFO Overflow Raw Interrupt Status 0 = ssi_txo_intr interrupt is not active prior to masking 1 = ssi_txo_intr interrupt is active prior masking **Reset Value:** 0x0
0	TXEIR	R/W	Transmit FIFO Empty Raw Interrupt Status 0 = ssi_txe_intr interrupt is not active prior to masking 1 = ssi_txe_intr interrupt is active prior masking **Reset Value:** 0x0

14.2.2.15 TXOICR

- **Name:** Transmit FIFO Overflow Interrupt Clear Register
- **Size:** 1 bit
- **Address Offset:** 0x38
- **Read/write access:** read-only

Bits	Name	R/W	Description
0	TXOICR	R	Clear Transmit FIFO Overflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_txo_intr interrupt; writing has no effect. **Reset Value:** 0x0

14.2.2.16 RXOICR

- **Name:** Receive FIFO Overflow Interrupt Clear Register
- **Size:** 1 bit
- **Address Offset:** 0x3C
- **Read/write access:** read-only

Bits	Name	R/W	Description
0	RXOICR	R	Clear Receive FIFO Overflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_rxo_intr interrupt; writing has no effect. **Reset Value:** 0x0

14.2.2.17 RXUICR

- **Name:** Receive FIFO Underflow Interrupt Clear Register
- **Size:** 1 bit
- **Address Offset:** 0x40
- **Read/write access:** read-only

Bits	Name	R/W	Description

Bits	Name	R/W	Description
0	RXUICR	R	Clear Receive FIFO Underflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_rxu_intr interrupt; writing has no effect. **Reset Value:** 0x0

14.2.2.18 MSTICR

- **Name:** Multi-Master Interrupt Clear Register
- **Size:** 1 bit
- **Address Offset:** 0x44
- **Read/write access:** read-only

Bits	Name	R/W	Description
0	MSTICR	R	Clear Multi-Master Contention Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_mst_intr interrupt; writing has no effect. **Reset Value:** 0x0

14.2.2.19 ICR

- **Name:** Interrupt Clear Register
- **Size:** 1 bit
- **Address Offset:** 0x48
- **Read/write access:** read-only

Bits	Name	R/W	Description
0	ICR	R	Clear Interrupts. This register is set if any of the interrupts below are active. A read clears the ssi_txo_intr, ssi_rxu_intr, ssi_rxo_intr, and the ssi_mst_intr interrupts. Writing to this register has no effect. **Reset Value:** 0x0

~~14.2.2.20 DMACR~~

- **Name:** DMA Control Register
- **Size:** 2 bits
- **Address Offset:** 0x4C
- **Read/write access:** read/write

This register is only valid when DW_apb_ssi is configured with a set of DMA Controller interface signals (SSI_HAS_DMA = 1).

~~14.2.2.21 DMATDLR~~

- **Name:** DMA Transmit Data Level
- **Size:** *TX_ABW*
- **Address Offset:** 0x50
- **Read/write access:** read/write

This register is only valid when the DW_apb_ssi is configured with a set of DMA interface signals (SSI_HAS_DMA = 1).

~~14.2.2.22 DMARDLR~~

- **Name:** DMA Receive Data Level
- **Size:** *RX_ABW*
- **Address Offset:** 0x54
- **Read/write access:** read/write

This register is only valid when DW_apb_ssi is configured with a set of DMA interface signals (SSI_HAS_DMA = 1).

14.2.2.23 IDR

- **Name:** Identification Register
- **Size:** 32 bits
- **Address Offset:** 0x58
- **Read/write access:** read-only

Bits	Name	R/W	Description
31:0	IDCODE	R	Identification Code. This register contains the peripherals identification code, which is written into the register at configuration time using coreConsultant. **Reset Value:** 0xffffffff

~~14.2.2.24 SSI_COMP_VERSION~~

- **Name:** coreKit version ID register
- **Size:** 32 bits
- **Address Offset:** 0x5C
- **Read/write access:** read-only

14.2.2.25 DR

- **Name:** Data Register
- **Size:** 16 bits
- **Address Offset:** 0x60 to 0xec
- **Read/write access:** read/write

The DW_apb_ssi data register is a 16-bit read/write buffer for the transmit/receive FIFOs. When the register is read, data in the receive FIFO buffer is accessed. When it is written to, data are moved into the transmit FIFO buffer; a write can occur only when SSI_EN = 1. FIFOs are reset when SSI_EN = 0.

Bits	Name	R/W	Description
15:0	DR	R/W	Data Register. When writing to this register, you must right-justify the data. Read data are automatically right-justified. Read = Receive FIFO buffer Write = Transmit FIFO buffer **Reset Value:** 0x0

~~14.2.2.26 RX_SAMPLE_DLY~~

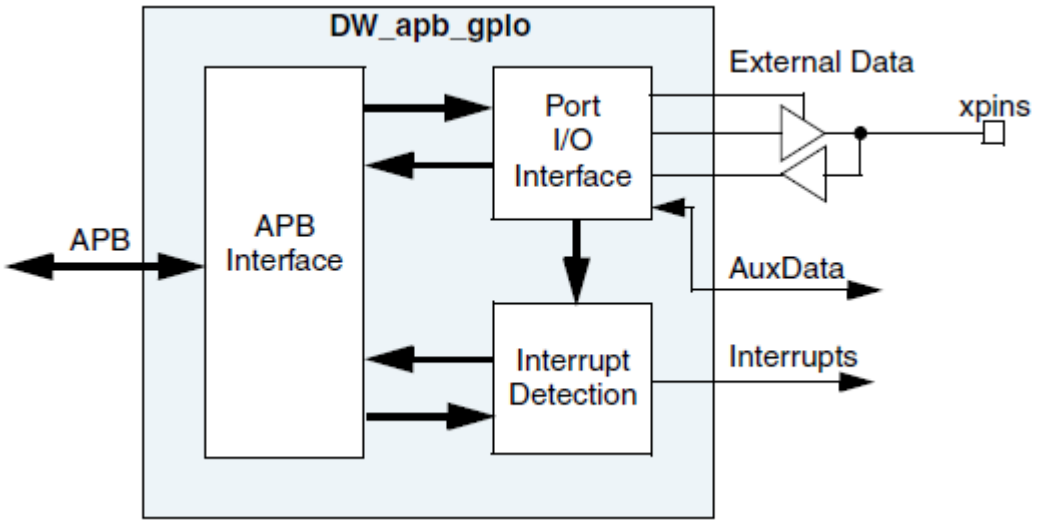
15 GPIO

15.1 GPIO Modules

The DW_apb_gpio is a programmable General Purpose Programming I/O (GPIO) peripheral. ~~The component is an AMBA 2.0-compliant Advanced Peripheral Bus (APB) slave device.~~

Following functional groupings of the main interfaces to the DW_apb_gpio block:

- ~~APB interface to or from APB bridge~~
- ~~External data Interface to or from I/O pads~~
- ~~Auxiliary hardware data interface to or from auxiliary data sink or source~~
- ~~The interrupt interface to or from interrupt controller~~



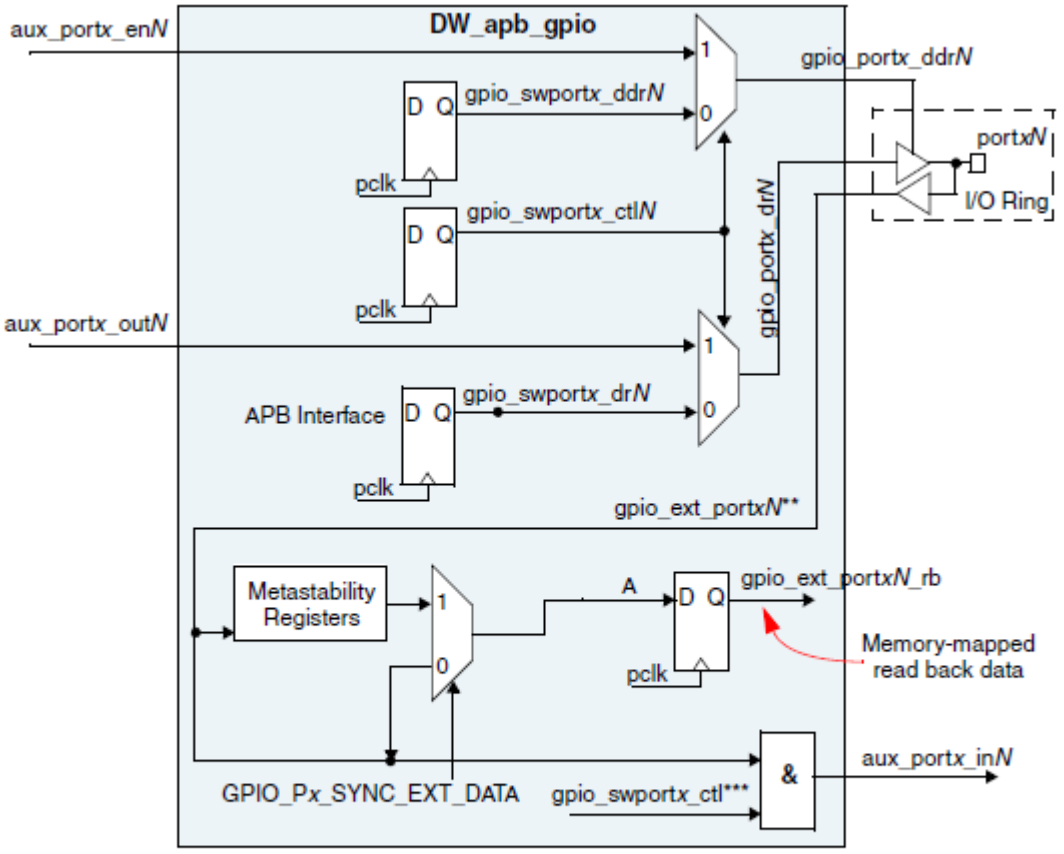
15.2 Function Description

15.2.1 Data and Control Flow

The DW_apb_gpio controls the output data and direction of external I/O pads. It also can read back the data on external pads using memory-mapped registers.

There are two methods for generating the default source of the input data, the output data, and the control of each signal—through software or hardware control. Software control occurs over the APB bus interface; hardware control is through the auxiliary hardware control interface.

The software option always exists and is described in more detail in “Software Control Mode” on page 22.The hardware option for each signal exists only if you choose it during configuration time. Provided the hardware option is built, you can switch between software and hardware modes by writing to a control register for the corresponding signal. Also, the device can be configured so that you can individually switch between hardware and software modes for each bit of each signal, provided that hardware mode exists; for more detail, refer to “Hardware Control Mode” on page 14.2.1.2. The data and control flow for a signal are shown in **Figure 15-2**.



N = 0 through GPIO_PWIDTH_X where "X" is A, B, C or D.

* These data are multiplexed onto prdata when a read from the gpio_ext_portx register occurs.

** This is a port signal. See Table 5-1 on page 50 for more information.

*** If configuration parameter GPIO_PORTN_SINGLE_CTL = 0, this remains a single bit as shown. If GPIO_PORTN_SINGLE_CTL = 1, this becomes a bus, gpio_swportx_ctlN.

15.2.1.1 Software Control Mode

When a signal is configured for software control, the data and direction control for the signal are sourced from the data register (gpio_swport*x*_dr) and direction control register (gpio_swport*x*_ddr), where *x* is either a, b, c, or d.

Under software control, the direction of the external I/O pad is controlled by a write to the Port*x* data direction register (gpio_swport*x*_ddr). The data written to this memory-mapped register gets mapped onto an output signal, gpio_port*x*_ddr, of the DW_apb_gpio peripheral. This output signal controls the direction of an external I/O pad.

The data written to the Port*x* data register (gpio_swport*x*_dr) drives the output buffer of the I/O pad. External data are input on the external data signal, gpio_ext_port*x*. Depending on whether gpio_ext_port*x* is configured as an input or output, the following occurs:

- Input – Reads the values on the signal
- Output – Reads the data register for Port*x*

The gpio_ext_port*x* register is read-only, meaning that it cannot be written from the APB software interface.

~~15.2.1.2 Hardware Control Mode~~

If a signal is configured for hardware control, it has external, auxiliary hardware signals controlling the data and the direction of Ports A through D. In this mode, the auxiliary data input signal (aux_port*x*_out) and direction control signal (aux_port*x*_en) are selected, where *x* is either a, b, c, or d.

The data direction of the external I/O pad, gpio_port*x*_ddr, is controlled through the auxiliary signal direction control signal, aux_port*x*_en.

For lines that are set to Output, the gpio_portx_dr and gpio_portx_ddr output signals drive the data and direction control onto the bidirectional pad that exists within the I/O ring of the SoC device. **Figure 15-2** on page 22 shows how the DW_apb_gpio peripheral controls the data and direction signals of an I/O PAD and data generation for the auxiliary source.

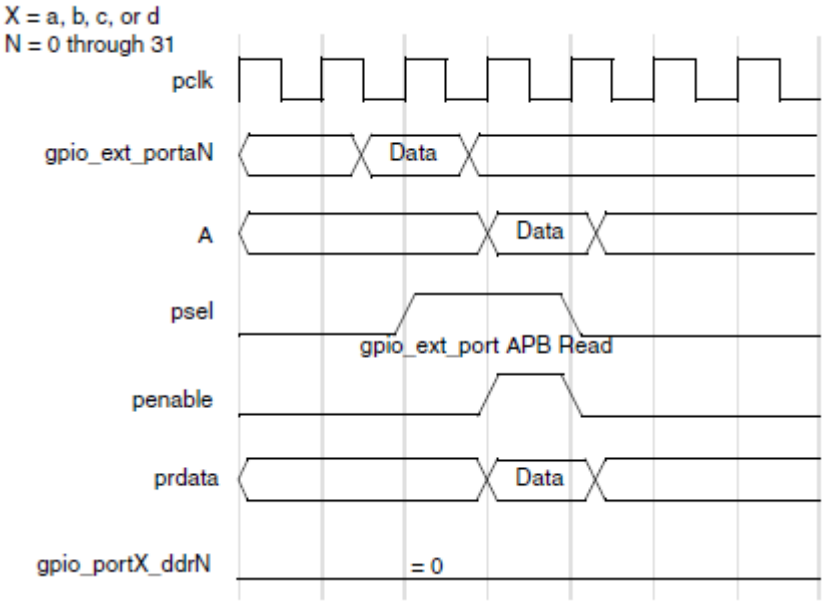
The gpio_swport*x*_ctl signal masks the value on the gpio_ext_portx external signal in order to generate aux_portx_in. The net result is that when hardware mode is selected, the value on aux_port_in output signal is equal to the value on the gpio_ext_port*x* input signal. When software mode is selected, the aux_port*x*_in output signal is always driven low. Setting bit 0 of gpio_swport*x*_ctl to 1 selects hardware mode for the entire signal if the parameter GPIO_PORT*X*_SINGLE_CTL is 1. If GPIO_PORT*X*_SINGLE_CTL is 0, setting bit *n* of gpio_swport*x*_ctl to 1 selects hardware mode for bit *n* of Port*x*. Setting bit 0 of gpio_swport*x*_ctl to 0 selects software mode for the entire signal if GPIO_PORT*X*_SINGLE_CTL is 1, while setting bit *n* of gpio_swport*x*_ctl to 0 selects software mode for bit *n* of Port*x* if GPIO_PORT*X*_SINGLE_CTL is 0.

15.2.1.3 Reading External Signals

The data on the external gpio signal is read by an APB read of the memory-mapped register, gpio_ext_port*x*. An APB read to the gpio_ext_port*x* register provides either the data on the gpio_ext_port*x* control lines or the contents of the gpio_swport*x*_dr, depending on the value of gpio_swportx_ddr.

~~**Figure 15-2 on page 22** shows how the hardware/software option is multiplexed, where the control lines for the multiplexing come from a memory-mapped register. It also shows the synchronization registers and the individual bit control of each data and data direction bit.~~

Figure 15-3 shows a timing diagram of a read to the gpio_ext_port*x* memory map registers when the direction is set to *Input* and the metastability registers are included.



~~#### 15.2.1.4 Reading External Signals~~

Synchronization of gpio_ext_port*X* to pclk prior to an APB read is enabled if the corresponding signal configuration parameter GPIO_P*x*_SYNC_EXT_DATA is set (*x* = A, B, C, or D); this is shown in **Figure 15-2 on page 22.**

15.2.2 Interrupts

Port A can be programmed to accept external signals as interrupt sources on any of the bits of the signal. The type of interrupt is programmable with one of the following settings:

- Active-high and level
  - Active-low and level
  - Rising edge
  - Falling edge

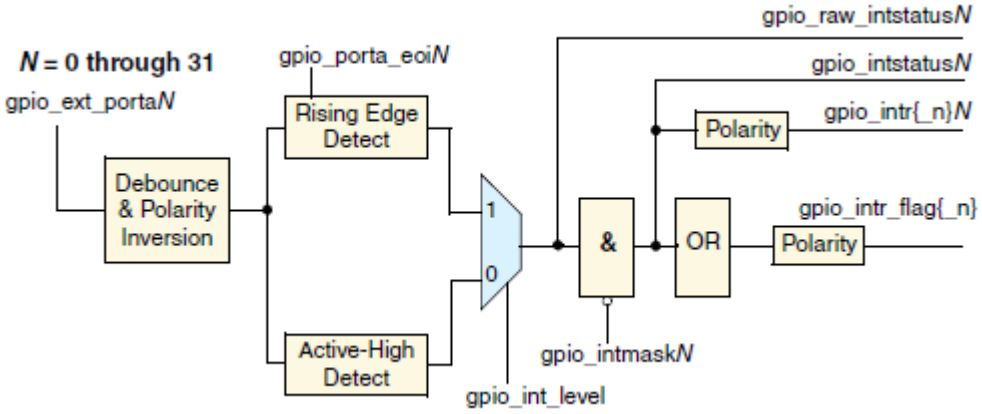
The interrupts can be masked by programming the gpio_intmask register. The interrupt status can be read before masking (called raw status) and after masking.

~~The interrupts are also combined into a single interrupt output signal, which has the same polarity as the individual interrupts. Either individual interrupts (gpio_intr or gpio_intr_n) or a single combined interrupt~~

~~(gpio_intr_flag or gpio_intr_flag_n) can be generated. In order to mask the combined interrupt, all individual interrupts have to be masked. The single combined interrupt does not have its own mask bit.~~

Whenever Port A is configured for interrupts, the data direction must be set to Input and the mode must be set to Software for interrupts to be latched. If the data direction register is reprogrammed to Output ~~or the mode register is programmed to enable Hardware mode,~~ then any pending interrupts are not lost. However, no new interrupts are generated.

Figure 15-4 illustrates how the interrupts are generated and how the data flows. The signal names in the diagram correspond to either I/O signals or memory-mapped registers.



Two interrupt request connection schemes are supported, and one scheme is chosen during configuration. The simplest connection scheme is where the combined interrupt gpio_intr_flag is generated by ORing together the bits of the gpio_intr bus. When only the combined interrupt request is used, then the gpio_status register must be read in the interrupt service routine (ISR) to find the source of the interrupt. When the individual interrupts lines are connected directly to the interrupt controller, then the gpio_status register does not have to be read by the ISR.

For edge-detected interrupts, the ISR can clear the interrupt by writing a 1 to the gpio_porta_eoi register for the corresponding bit to disable the interrupt. This write also clears the interrupt status and raw status registers. It is recommended that the interrupt source be cleared prior to writing to the gpio_porta_eoi register. Writing to the gpio_porta_eoi register has no effect on level-sensitive interrupts.

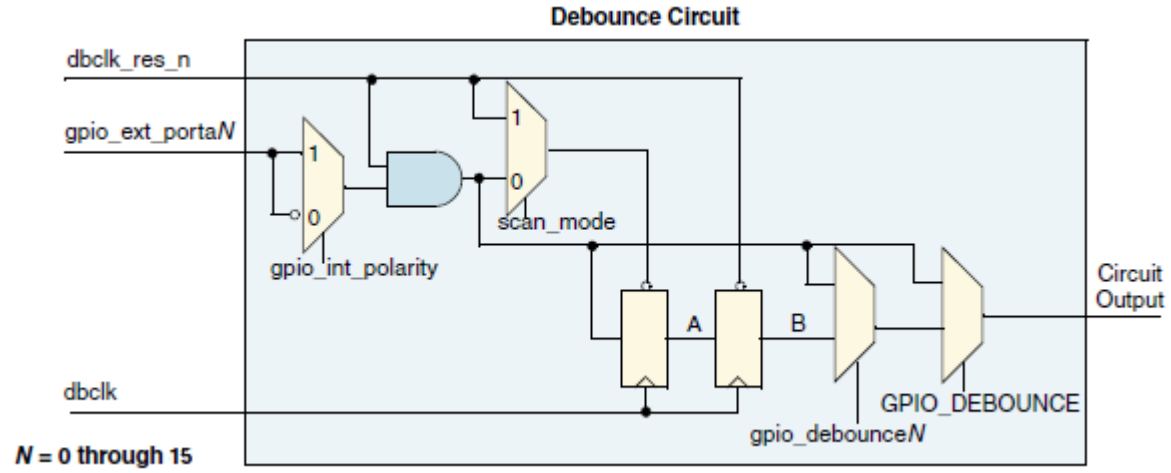
If level-sensitive interrupts cause the processor to interrupt, then the ISR can poll the gpio_rawint status register until the interrupt source disappears, or it can write to the gpio_intmask register to mask the interrupt before exiting the ISR. If the ISR exits without masking or disabling the interrupt prior to exiting, then the level-sensitive interrupt repeatedly requests an interrupt until the interrupt is cleared at the source.
If the gpio_intr_flag connection scheme is used and the interrupt service routine reads the gpio_intr_status register to find multiple pending interrupt requests, then it is up to the processor to prioritize these pending interrupt requests. There are no restrictions on the number of edge-detected interrupts that can be cleared simultaneously by writing multiple 1's to the gpio_porta_eoi register.

15.2.2.1 Debounce Operation

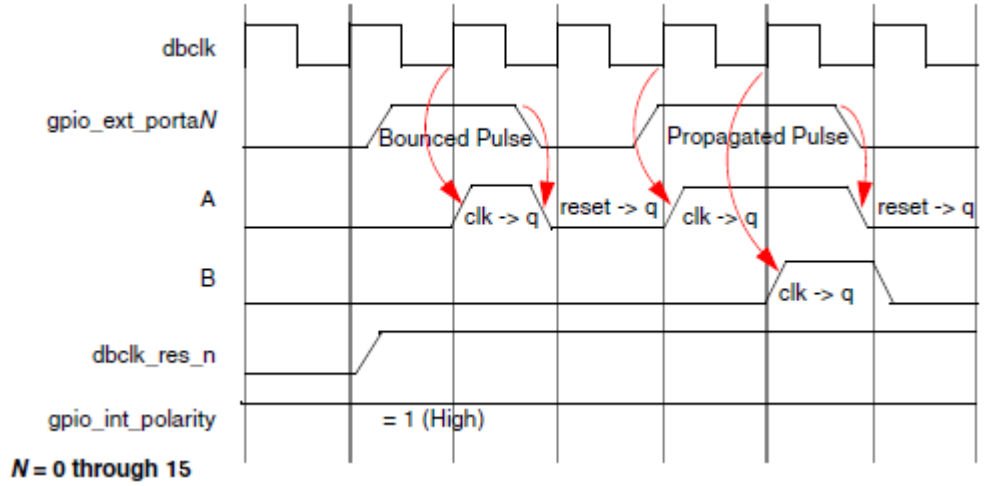
~~If the user has configured Port A to include the interrupt feature, the DW_apb_gpio can be configured to either include or exclude a debounce capability using the GPIO_DEBOUNCE parameter.~~

The external signal can be debounced to remove any spurious glitches that are less than one period of the external debouncing clock.

Figure 15-5 shows an RTL diagram of the debounce circuitry. The timing diagram shows an active-high input signal on gpio_ext_porta*N*. The polarity of the input signal detection is controlled by the memory-mapped signal, gpio_int_polarity. For a falling-edge- or active-low-sensitive input, the input is then inverted and the same debounce logic is used as for rising-edge or active-high level-sensitive interrupts.



A timing diagram of the debounce circuitry is shown in Figure 15-6.



When input interrupt signals are debounced using a debounce clock, the signals must be active for a minimum of two cycles of the debounce clock to guarantee that they are registered. Any input pulse widths less than a debounce clock period are bounced. A pulse width between one and two debounce clock widths may or may not propagate, depending on its phase relationship to the debounce clock. If the input pulse spans two rising edges of the debounce clock, it is registered. If it spans only one rising edge, it is not registered.

The timing diagram in **Figure 15-6** shows both cases: the input signal being bounced, and later, a propagated input signal. If the DW_apb_gpio supports debounce, then debouncing input signals on Port A can be enabled or disabled under software control.

The dbclk_res_n signal is asynchronously asserted and synchronously de-asserted to the debounce clock, dbclk. The system reset signal, presetn, is asynchronously asserted and synchronously de-asserted to pclk; synchronization must be external to the component. The pclk and dbclk signals are assumed to be asynchronous to each other.

The debounce circuitry works with only asynchronous reset flip-flops.

15.2.2.2 Synchronization of Interrupt Signals to the System Clock

Interrupt signals can be internally synchronized to a system clock, pclk_intr. Synchronization to pclk_intr must occur for edge-detect signals. Edge-detected interrupts to the processor are always synchronous to the system bus clock. With level-sensitive interrupts, synchronization is optional and under software control.

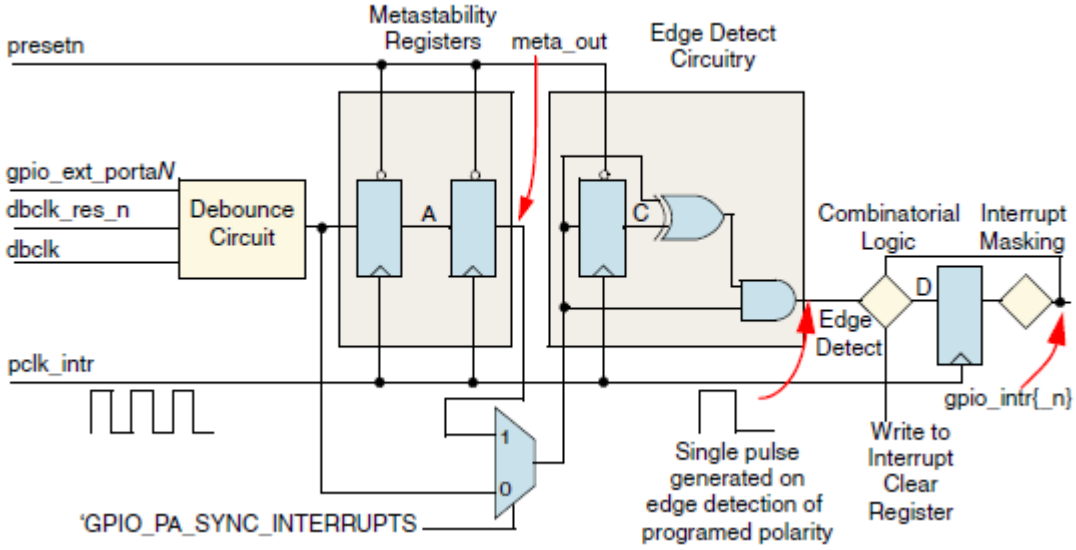
The pclk_intr signal is needed for systems that may have the DW_apb_gpio pclk bus clock gated off, but the system still wants to detect interrupts. It is assumed that this clock is synchronous to pclk. If interrupt detection is required only when pclk is running, then pclk_intr and pclk can be connected to the same clock

source. If the system employs a gated pclk to the DW_apb_gpio, pclk_intr needs to be running for interrupt detection to occur.

The gpio_intrclk_en output signal is asserted when either edge-sensitive interrupts or level-sensitive interrupts requiring synchronization are enabled in the DW_apb_gpio block. Both cases require a clock for detection. Therefore, this signal can cause the external clock generator block to generate pclk_intr.

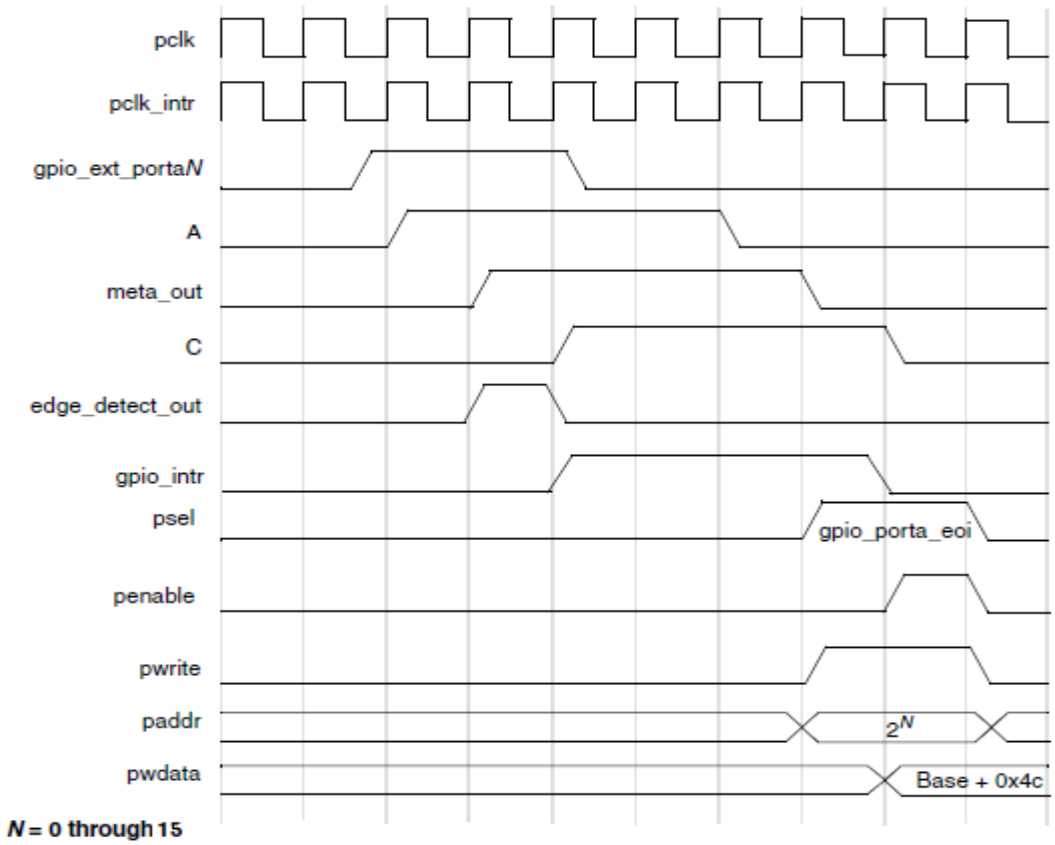
15.2.2.3 Interrupt Edge Dectecion

Figure 15-7 shows an RTL diagram of the synchronization and edge detection of interrupt sources on gpio_ext_porta*N* signals.



The MUX allows inclusion or exclusion of the metastability registers at configuration depending on the value of GPIO_PA_SYNC_INTERRUPTS. If this parameter is a 1, the registers are included.

Figure 15-8 shows a timing diagram in which an interrupt is generated on the rising edge of an input on Port A; this is where the debounce logic is disabled and Metastability registers are included. It also shows how an interrupt is cleared by a write to the interrupt clear register.



A case may arise where the Interrupt Service Routine (ISR) writes to the interrupt clear register in order to clear an existing interrupt during the same clock cycle in which a new interrupt is detected. In such a case, writing to the interrupt clear register clears only the first interrupt. The second interrupt is not lost, since setting an interrupt has a higher priority than clearing it.

Figure 15-9 shows a timing diagram similar to Figure 15-8, except that Metastability registers are removed. It also shows how an interrupt is cleared by a write to the interrupt clear register.

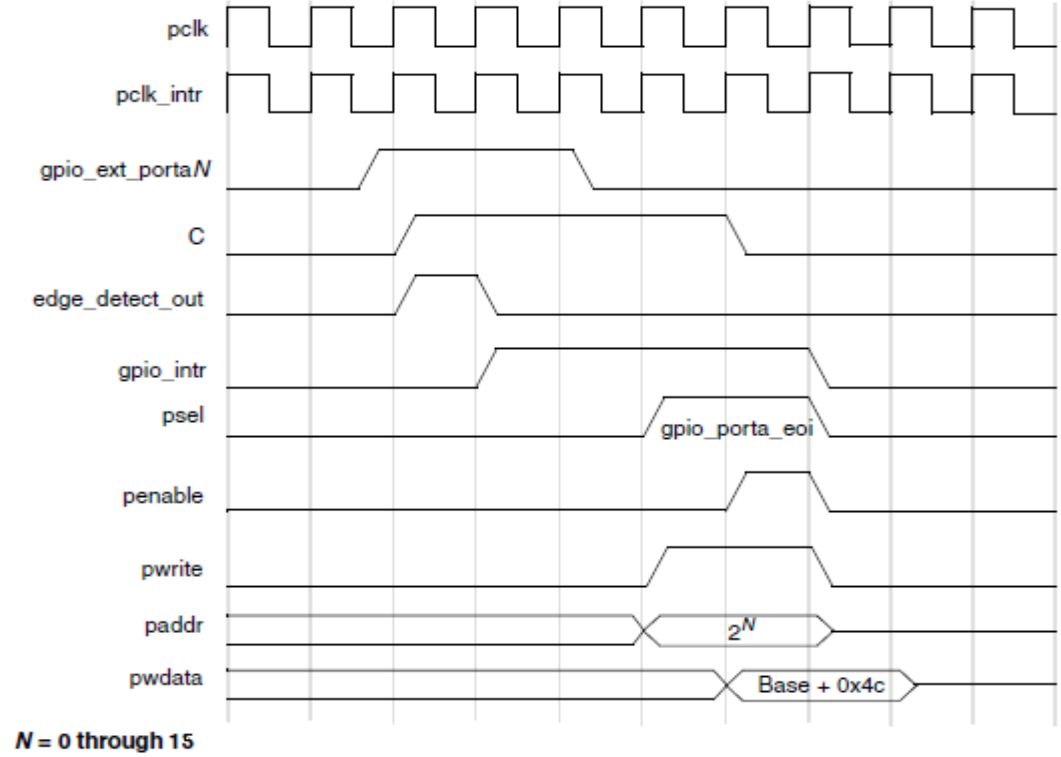
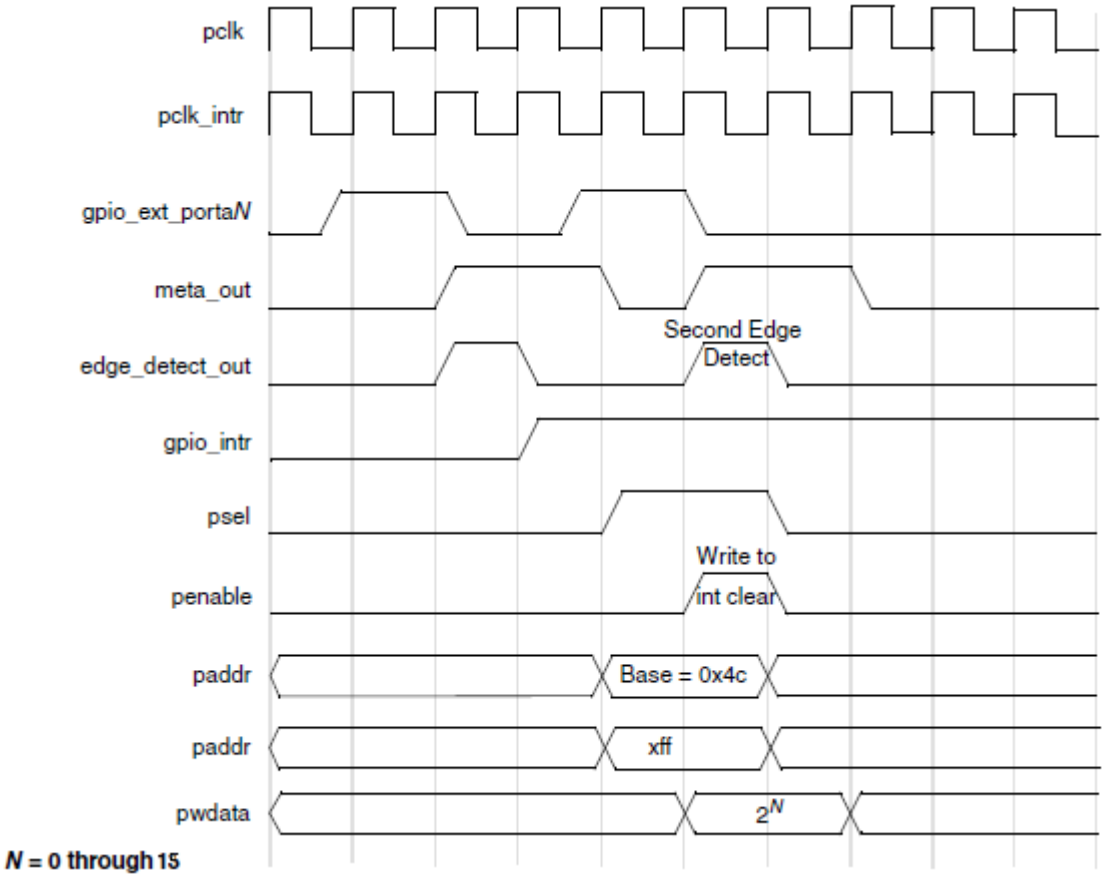


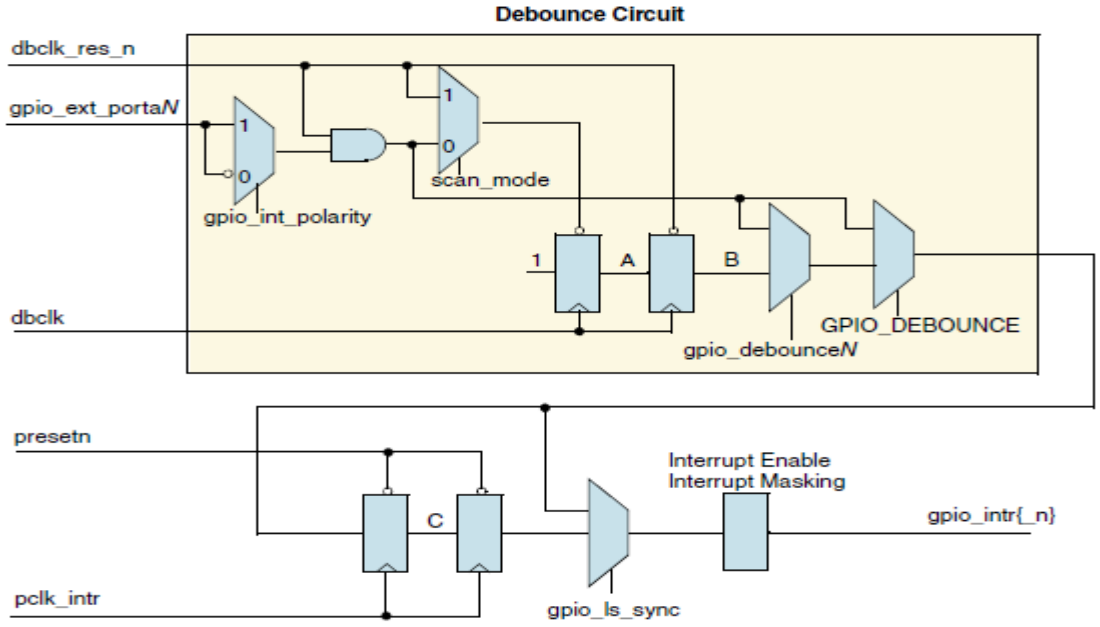
Figure 15-10 shows such a case where the debounce logic is unused. In this timing diagram, meta_out and edge_detect_out are the outputs of the second metastability register and the edge detect logic, respectively.

The second edge detection occurs on the same cycle as the write to the interrupt clear register. In this example, the write to the interrupt clear register does not clear the second interrupt, and the gpio_intr{_n} signal is not de-asserted.



15.2.2.4 Level-Sensitive Interrupts

Figure 15-11 shows the generation of level-sensitive interrupts. As for edge-detect interrupts, the user can configure DW_apb_gpio with or without debounce logic.

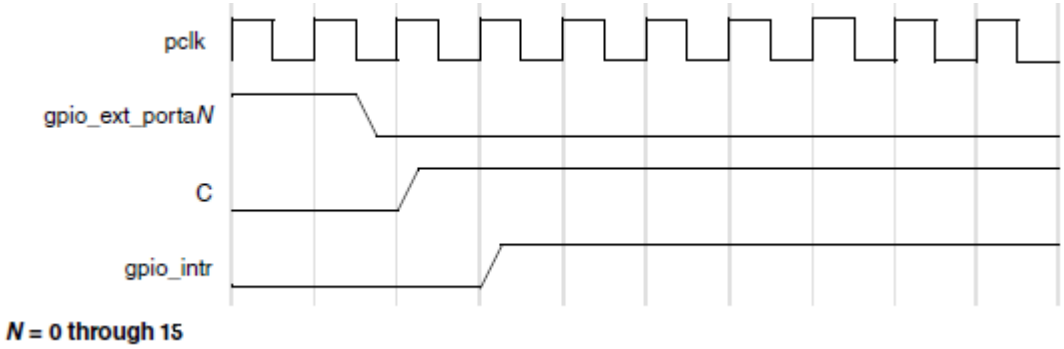


With level-sensitive interrupts, there is a choice of whether they are synchronized to the interrupt clock pclk_intr or are entirely combinational (aside from the debounce circuitry). The selection is done by programming the gpio_ls_sync (GPIO Level Sensitive Synchronous) register.

This is a memory-mapped bit that inserts two metastability registers clocked off of pclk_intr to synchronize the level-sensitive interrupts to pclk_intr. When gpio_ls_sync is not asserted, then there is no guarantee that the interrupt lines are synchronous to pclk_intr. A processor status register may need to be set to indicate asynchronous interrupts. When gpio_ls_sync is asserted, then the pclk_intr clock must be present to pass the interrupt to the interrupt controller block. The gpio_intrclk_en output signal is asserted when level-sensitive interrupts that are to be synchronized to pclk_intr are selected. The gpio_intrclk_en signal can be used in the clock generation block to turn on pclk_intr.

The input signal is inverted for active-low level-sensitive interrupts. The same detection logic is used here as is used for active-high level-sensitive interrupts.

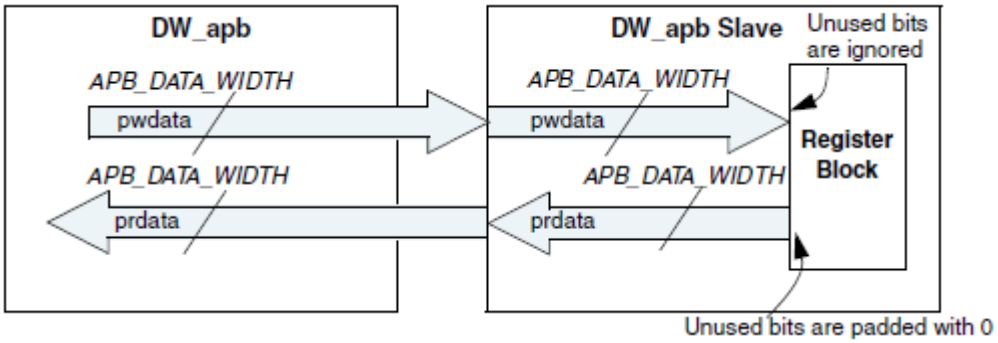
Figure 15-12 shows the generation of an active-low level-sensitive interrupt where the debounce circuitry is disabled.



15.3 GPIO Register Manual

15.3.1 Bus Interface

The DW_apb_gpio peripheral has a standard AMBA 2.0 APB interface for reading and writing the internal registers. This peripheral supports APB data bus widths of 8, 16, or 32 bits., which is set with the.



15.3.2 GPIO Register Mapping Summary

Name	Address Offset	Width	Type	Description
gpio_swporta_dr	0x00	32	R/W	Port A data register
gpio_swporta_ddr	0x04	32	R/W	Port A data direction register
~~gpio_swporta_ctl~~	~~0x08~~	~~32~~	~~R/W~~	~~Port A data source register~~
gpio_swportb_dr	0x0C	32	R/W	Port B data register
gpio_swportb_ddr	0x10	32	R/W	Port B data direction register
~~gpio_swportb_ctl~~	~~0x14~~	~~32~~	~~R/W~~	~~Port B data source register~~
gpio_swportc_dr	0x18	32	R/W	Port C data register

gpio_swportc_ddr	0x1C	32	R/W	Port C data direction register
~~gpio_swportc_ctl~~	~~0x20~~	~~32~~	~~R/W~~	~~Port C data source register~~
gpio_swportd_dr	0x24	32	R/W	Port D data register
gpio_swportd_ddr	0x28	32	R/W	Port D data direction register
~~gpio_swportd_ctl~~	~~0x2C~~	~~32~~	~~R/W~~	~~Port D data source register~~
gpio_inten	0x30	32	R/W	Interrupt enable register
gpio_intmask	0x34	32	R/W	Interrupt mask register
gpio_inttype_level	0x38	32	R/W	Interrupt level register
gpio_int_polarity	0x3C	32	R/W	Interrupt polarity register
gpio_intstatus	0x40	32	R/W	Interrupt status of Port A
gpio_raw_intstatus	0x44	32	R/W	Row Interrupt status of Port A
gpio_debounce	0x48	32	R/W	Debounce enable register
gpio_porta_eoi	0x4C	32	R/W	Port A clear interrupt register
gpio_ext_porta	0x50	32	R/W	Port A external port register
gpio_ext_portb	0x54	32	R/W	Port B external port register
gpio_ext_portc	0x58	32	R/W	Port C external port register
gpio_ext_portd	0x5C	32	R/W	Port D external port register
gpio_is_sync	0x60	32	R/W	Level-sensitive synchronization enable
~~gpio_id_code~~	~~0x64~~	~~32~~	~~R/W~~	~~ID code register~~
reserved	0x68	32	R/W	Reserved
~~gpio_ver_id_code~~	~~0x6C~~	~~32~~	~~R/W~~	~~Component Version register~~
gpio_config_reg1	0x70	32	R/W	Configuration Register 1
gpio_config_reg2	0x74	32	R/W	Configuration Register 2

15.3.3 Register and Field Descriptions

15.3.3.1 gpio_swporta_dr

- **Name:** Port A Data Register
- **Size:** ~~*GPIO_PWIDTH_A*~~*1 bit*
- **Address Offset:** 0x00
- **Read/write access:** read/write

Bits	Name	R/W	Description
31:1~~*GPIO_PWIDTH_A*~~	Reserved, read as zero		

Bits	Name	R/W	Description
~~*GPIO_PWIDTH_A*-1~~:0	Port A Data Register	R/W	Values written to this register are output on the I/O signals for Port A if the corresponding data direction bits for Port A are set to Output mode and the corresponding control bit for Port A is set to Software mode. The value read back is equal to the last value written to this register. **Reset Value:** ~~*GPIO_SWPORTA_RESET0x0*~~

15.3.3.2 gpio_swporta_ddr

- **Name:** Port A Data Direction Register
- **Size:** ~~*GPIO_PWIDTH_A*~~*1 bit*
- **Address Offset:** 0x04
- **Read/write access:** read/write

Bits	Name	R/W	Description
31:1~~*GPIO_PWIDTH_A*~~	Reserved, read as zero		
~~*GPIO_PWIDTH_A*-1~~:0	Port A Data Direction Register	R/W	Values written to this register independently control the direction of the corresponding data bit in Port A. The default direction can be configured as input or output after system reset through input ~~the GPIO_DFLT_DIR_A parameter.~~ 0 – Input (default) 1 – Output **Reset Value:** ~~*GPIO_DFLT_DIR_A0x0*~~

~~15.3.3.3 gpio_swporta_ctl~~

- **Name:** Port A Data Source
- **Size:**
  1 bit wide if GPIO_PORTA_SINGLE_CTL = 1
  GPIO_PWIDTH_A bits wide if GPIO_PORTA_SINGLE_CTL = 0
- **Address Offset:** 0x08
- **Read/write access:** read/write

Bits	Name	R/W	Description
0 -or- 0:*GPIO_ PWIDTH_A*-1 see above.	Port A Data Source	R/W	The data and control source for a signal can come from either software or hardware; this bit selects between them. The default source is configurable through the GPIO_DFLT_SRC_A configuration parameter. 0 – Software mode (default) 1 – Hardware mode If GPIO_PORTA_SINGLE_CTL = 0, the register will contain one bit for each bit of the signal. Upon reset in this case, the value of GPIO_DFLT_SRC_A is replicated across all bits of the signal so that all bits power up with the same operating mode. Furthermore, the default source of each bit of the signal can subsequently be changed by writing to the corresponding bit of this register. This register is not available unless GPIO_HW_PORTA = 1. **Reset Value:** If GPIO_PORTA_SINGLE_CTL = 1, then the reset value is *GPIO_DFLT_SRC_A.* If GPIO_PORTA_SINGLE_CTL = 0, then the reset value is {GPIO_PWIDTH_A{GPIO_DFLT_SRC_A in each bit}}.

15.3.3.4 gpio_swportb_dr

- **Name:** Port B Data Register
- **Size:** ~~*GPIO_PWIDTH_B1 bit*~~
- **Address Offset:** 0x0c
- **Read/write access:** read/write

Bits	Name	R/W	Description
31:1~~*GPIO_PWIDTH_B*~~	Reserved, read as zero		
~~*GPIO_PWIDTH_B*~~ ~~1~~:0	Port B Data Register	R/W	Values written to this register are output on the I/O signals for Port B if the corresponding data direction bits for Port B are set to Output mode and the corresponding control bit for Port B is set to Software mode. The value read back is equal to the last value written to this register. **Reset Value:** ~~GPIO_SWPORTB_RESET~~0x0

15.3.3.5 gpio_swportb_ddr

- **Name:** Port B Data Direction
- **Size:** ~~*GPIO_PWIDTH_B1 bit*~~
- **Address Offset:** 0x10
- **Read/write access:** read/write

Bits	Name	R/W	Description
31:1~~*GPIO_PWIDTH_B*~~	Reserved, read as zero		
~~*GPIO_PWIDTH_B-1:0*~~	Port B Data Direction	R/W	Values written to this register independently control the direction of the corresponding data bit in Port B. The default direction can be configured as input or output after system reset through input ~~the GPIO_DFLT_DIR_B parameter~~. 0 – Input (default) 1 – Output **Reset Value:** ~~*GPIO_DFLT_DIR_B0x0*~~

~~15.3.3.6 gpio_swportb_ctl~~

- **Name:** Port B Data Source
- **Size:**
  1 bit wide if GPIO_PORTB_SINGLE_CTL = 1
  GPIO_PWIDTH_B bits wide if GPIO_PORTB_SINGLE_CTL = 0
- **Address Offset:** 0x14
- **Read/write access:** read/write

Bits	Name	R/W	Description

Bits	Name	R/W	Description
0 -or- 0:*GPIO_ *PWIDTH_B*-1 see above.	Port B Data Source	R/W	The data and control source for a signal can come from either software or hardware; this bit selects between them. The default source is configurable through the GPIO_DFLT_SRC_B configuration parameter. 0 – Software mode (default) 1 – Hardware mode If GPIO_PORTB_SINGLE_CTL = 0, the register will contain one bit for each bit of the signal. Upon reset in this case, the value of GPIO_DFLT_SRC_B is replicated across all bits of the signal so that all bits power up with the same operating mode. Furthermore, the default source of each bit of the signal can subsequently be changed by writing to the corresponding bit of this register. This register is not available unless GPIO_HW_PORTB = 1. **Reset Value:** If GPIO_PORTB_SINGLE_CTL = 1, then the reset value is *GPIO_DFLT_SRC_B.* If GPIO_PORTB_SINGLE_CTL = 0, then the reset value is {GPIO_PWIDTH_B{GPIO_DFLT_SRC_B in each bit}}.

15.3.3.7 gpio_swportc_dr

- **Name:** Port C Data Register
- **Size:** ~~*GPIO_PWIDTH_C*~~*1 bit*
- **Address Offset:** 0x18
- **Read/write access:** read/write

Bits	Name	R/W	Description
31:1~~*GPIO_PWIDTH_C*~~	Reserved, read as zero		
~~*GPIO_PWIDTH_C–1:0*~~	Port C Data Register	R/W	Values written to this register are output on the I/O signals for Port C if the corresponding data direction bits for Port C are set to Output mode and the corresponding control bit for Port C is set to Software mode. The value read back is equal to the last value written to this register. **Reset Value:** ~~*GPIO_SWPORTC_RESET*~~*0x0*

15.3.3.8 gpio_swportc_ddr

- **Name:** Port C Data Direction
- **Size:** ~~*GPIO_PWIDTH_C*~~*1 bit*
- **Address Offset:** 0x1c
- **Read/write access:** read/write

Bits	Name	R/W	Description
31:~~*GPIO_PWIDTH_C1*~~	Reserved, read as zero		

Bits	Name	R/W	Description
~~*GPIO_PWIDTH_C -1:0*~~	Port C Data Direction	R/W	Values written to this register independently control the direction of the corresponding data bit in Port C. The default direction can be configured as input or output after system reset through ~~the~~ input~~GPIO_DFLT_DIR_C parameter.~~ 0 – Input (default) 1 – Output **Reset Value:** ~~*GPIO_DFLT_DIR_C*~~*0x0*

~~15.3.3.9 gpio_swportc_ctl~~

- **Name:** Port C Data Source
- **Size:**
  1 bit wide if GPIO_PORTC_SINGLE_CTL = 1
  GPIO_PWIDTH_C bits wide if GPIO_PORTC_SINGLE_CTL = 0
- **Address Offset:** 0x20
- **Read/write access:** read/write

Bits	Name	R/W	Description
0 -or- 0:*GPIO_ PWIDTH_C*-1 see above.	Port C Data Source	R/W	The data and control source for a signal can come from either software or hardware; this bit selects between them. The default source is configurable through the GPIO_DFLT_SRC_C configuration parameter. 0 – Software mode (default) 1 – Hardware mode If GPIO_PORTC_SINGLE_CTL = 0, the register will contain one bit for each bit of the signal. Upon reset in this case, the value of GPIO_DFLT_SRC_C is replicated across all bits of the signal so that all bits power up with the same operating mode. Furthermore, the default source of each bit of the signal can subsequently be changed by writing to the corresponding bit of this register. This register is not available unless GPIO_HW_PORTC = 1. **Reset Value:** If GPIO_PORTC_SINGLE_CTL = 1, then the reset value is *GPIO_DFLT_SRC_C.* If GPIO_PORTC_SINGLE_CTL = 0, then the reset value is {GPIO_PWIDTH_C{GPIO_DFLT_SRC_C in each bit}}.

15.3.3.10 gpio_swportd_dr

- **Name:** Port D Data Register
- **Size:** ~~*GPIO_PWIDTH_D*~~*1 bit*
- **Address Offset:** 0x24
- **Read/write access:** read/write

Bits	Name	R/W	Description
31:1~~*GPIO_PWIDTH_D*~~	Reserved, read as zero		

Bits	Name	R/W	Description
~~*GPIO_PWIDTH_D–1:0*~~	Port D Data Register	R/W	Values written to this register are output on the I/O signals for Port D if the corresponding data direction bits for Port D are set to Output mode and the corresponding control bit for Port D is set to Software mode. The value read back is equal to the last value written to this register. 0 – Input (default) 1 – Output **Reset Value:** ~~*GPIO_SWPORTD_RESET*~~*0x0*

15.3.3.11 gpio_swportd_ddr

- **Name:** Port D Data Direction
- **Size:** ~~*GPIO_PWIDTH_D*~~*1'b1*
- **Address Offset:** 0x28
- **Read/write access:** read/write

Bits	Name	R/W	Description
31:1~~*GPIO_PWIDTH_D*~~	Reserved, read as zero		
~~*GPIO_PWIDTH_D–1:0*~~	Port D Data Direction	R/W	Values written to this register independently control the direction of the corresponding data bit in Port D. The default direction can be configured as input or output after system reset through ~~the~~ input ~~GPIO_DFLT_DIR_D parameter.~~ 0 – Input (default) 1 – Output **Reset Value:** ~~*GPIO_DFLT_DIR_D*~~*0x0*

~~15.3.3.12 gpio_swportd_ctl~~

- **Name:** Port D Data Source
- **Size:**
  ~~1 bit wide if GPIO_PORTD_SINGLE_CTL = 1~~
  1'b1~~*GPIO_PWIDTH_D* bits wide if GPIO_PORTD_SINGLE_CTL = 0~~
- **Address Offset:** 0x2c
- **Read/write access:** read/write

Bits	Name	R/W	Description

Bits	Name	R/W	Description
0 ~~-or-~~ ~~0:*GPIO_PWIDTH_D-1*~~ ~~see above.~~	Port D Data Source	R/W	The data and control source for a signal can come from either software or hardware; this bit selects between them. The default source is configurable through the GPIO_DFLT_SRC_D configuration parameter. 0 – Software mode (default) 1 – Hardware mode If GPIO_PORTD_SINGLE_CTL = 0, the register will contain one bit for each bit of the signal. Upon reset in this case, the value of GPIO_DFLT_SRC_D is replicated across all bits of the signal so that all bits power up with the same operating mode. Furthermore, the default source of each bit of the signal can subsequently be changed by writing to the corresponding bit of this register. This register is not available unless GPIO_HW_PORTD = 1. **Reset Value:** ~~If GPIO_PORTD_SINGLE_CTL = 1, then the reset value is *GPIO_DFLT_SRC_D.*~~ ~~If GPIO_PORTD_SINGLE_CTL = 0, then the reset value is {GPIO_PWIDTH_D{GPIO_DFLT_SRC_D in each bit}}.~~1'b0

15.3.3.13 gpio_inten

- **Name:** Interrupt enable
- **Size:** ~~*GPIO_PWIDTH_A*~~1'b1
  ~~This register is available only if Port A is configured to generate interrupts (GPIO_PORTA_INTR = Include (1)).~~
- **Address Offset:** 0x30
- **Read/write access:** read/write

Bits	Name	R/W	Description
31:1~~*GPIO_PWIDTH_A*~~	Reserved, read as zero		
~~*GPIO_PWIDTH_A-1:0*~~	Interrupt enable	R/W	Allows each bit of Port A to be configured for interrupts. By default the generation of interrupts is disabled. Whenever a 1 is written to a bit of this register, it configures the corresponding bit on Port A to become an interrupt; otherwise, Port A operates as a normal GPIO signal. Interrupts are disabled on the corresponding bits of Port A if the corresponding data direction register is set to Output or if Port A mode is set to Hardware. 0 – Configure Port A bit as normal GPIO signal (default) 1 – Configure Port A bit as interrupt **Reset Value:** 0x0

15.3.3.14 gpio_intmask

- **Name:** Interrupt mask
- **Size:** ~~*GPIO_PWIDTH_A*~~1'b1
  ~~This register is available only if Port A is configured to generate interrupts (GPIO_PORTA_INTR = Include (1)).~~

- **Address Offset:** 0x34
- **Read/write access:** read/write

Bits	Name	R/W	Description
31:1~~*GPIO_PWIDTH_A*~~	Reserved, read as zero		
~~*GPIO_PWIDTH_A–1:0*~~	Interrupt mask	R/W	Controls whether an interrupt on Port A can create an interrupt for the interrupt controller by not masking it. By default, all interrupts bits are unmasked. Whenever a 1 is written to a bit in this register, it masks the interrupt generation capability for this signal; otherwise interrupts are allowed through. The unmasked status can be read as well as the resultant status after masking. 0 – Interrupt bits are unmasked (default) 1 – Mask interrupt **Reset Value:** 0x0

15.3.3.15 gpio_inttype_level

- **Name:** Interrupt level
- **Size:** ~~*GPIO_PWIDTH_A*~~*1'b1*
  ~~This register is available only if Port A is configured to generate interrupts (GPIO_PORTA_INTR = Include (1)).~~
- **Address Offset:** 0x38
- **Read/write access:** read/write

Bits	Name	R/W	Description
31:1~~*GPIO_PWIDTH_A*~~	Reserved, read as zero		
~~*GPIO_PWIDTH_A–1:0*~~	Interrupt level	R/W	Controls the type of interrupt that can occur on Port A Whenever a 0 is written to a bit of this register, it configures the interrupt type to be level-sensitive; otherwise, it is edge-sensitive. 0 – Level-sensitive (default) 1 – Edge-sensitive **Reset Value:** 0x0

15.3.3.16 gpio_int_polarity

- **Name:** Interrupt polarity
- **Size:** ~~*GPIO_PWIDTH_A*~~*1'b1*
  ~~This register is available only if Port A is configured to generate interrupts (GPIO_PORTA_INTR = Include (1)).~~
- **Address Offset:** 0x3c
- **Read/write access:** read/write

Bits	Name	R/W	Description
31:1~~*GPIO_PWIDTH_A*~~	Reserved, read as zero		
~~*GPIO_PWIDTH_A–1:0*~~	Interrupt polarity	R/W	Controls the polarity of edge or level sensitivity that can occur on input of Port A. Whenever a 0 is written to a bit of this register, it configures the interrupt type to falling-edge or active-low sensitive; otherwise, it is rising-edge or active-high sensitive. 0 – Active-low (default) 1 – Active-high **Reset Value:** 0x0

15.3.3.17 gpio_intstatus

- **Name:** Interrupt status

- **Size:** ~~*GPIO_PWIDTH_A1'b1*~~
  ~~This register is available only if Port A is configured to generate interrupts (GPIO_PORTA_INTR = Include (1)).~~
- **Address Offset:** 0x40
- **Read/write access:** read

Bits	Name	R/W	Description
31:1~~*GPIO_PWIDTH_A*~~	Reserved, read as zero		
~~*GPIO_PWIDTH_A-1:0*~~	Interrupt status	R/W	Interrupt status of Port A **Reset Value:** 0x0

15.3.3.18 gpio_raw_intstatus

- **Name:** Raw interrupt status
- **Size:** ~~*GPIO_PWIDTH_A1'b1*~~
  ~~This register is available only if Port A is configured to generate interrupts (GPIO_PORTA_INTR = Include (1)).~~
- **Address Offset:** 0x44
- **Read/write access:** read

Bits	Name	R/W	Description
31:1~~*GPIO_PWIDTH_A*~~	Reserved, read as zero		
~~*GPIO_PWIDTH_A-1:0*~~	Raw Interrupt status	R/W	Raw interrupt of status of Port A (premasking bits) **Reset Value:** 0x0

15.3.3.19 gpio_debounce

- **Name:** Debounce enable
- **Size:** ~~*GPIO_PWIDTH_A1'b1*~~
  ~~This register is available only if Port A is configured to generate interrupts (GPIO_PORTA_INTR = Include (1)) and when the debounce logic is included (GPIO_DEBOUNCE = Include (1)).~~
- **Address Offset:** 0x48
- **Read/write access:** read/write

Bits	Name	R/W	Description
31:1~~*GPIO_PWIDTH_A*~~	Reserved, read as zero		
~~*GPIO_PWIDTH_A-1:0*~~	Debounce enable	R/W	Controls whether an external signal that is the source of an interrupt needs to be debounced to remove any spurious glitches. Writing a 1 to a bit in this register enables the debouncing circuitry. A signal must be valid for two periods of an external clock before it is internally processed. 0 – No debounce (default) 1 – Enable debounce **Reset Value:** 0x0

15.3.3.20 gpio_porta_eoi

- **Name:** Clear interrupt
- **Size:** ~~*GPIO_PWIDTH_A1'b1*~~
  ~~This register is available only if Port A is configured to generate interrupts (GPIO_PORTA_INTR = Include (1)).~~
- **Address Offset:** 0x4c
- **Read/write access:** write

Bits	Name	R/W	Description
31:1~~*GPIO_PWIDTH_A*~~	Reserved, read as zero		

Bits	Name	R/W	Description
~~GPIO_PWIDTH_A-1:0~~	Clear interrupt	W	Controls the clearing of edge type interrupts from Port A. When a 1 is written into a corresponding bit of this register, the interrupt is cleared. All interrupts are cleared when Port A is not configured for interrupts. 0 – No interrupt clear (default) 1 – Clear interrupt **Reset Value:** 0x0

15.3.3.21 gpio_ext_porta

- **Name:** External Port A
- **Size:** *~~GPIO_PWIDTH_A~~1'b1*
- **Address Offset:** 0x50
- **Read/write access:** read

Bits	Name	R/W	Description
31:1*~~GPIO_PWIDTH_A~~*	Reserved, read as zero		
~~GPIO_PWIDTH_A-1:0~~	External Port A	R	When Port A is configured as Input, then reading this location reads the values on the signal. When the data direction of Port A is set as Output, reading this location reads the data register for Port A. **Reset Value:** 0x0

15.3.3.22 gpio_ext_portb

- **Name:** External Port B
- **Size:** *~~GPIO_PWIDTH_B~~1'b1*
- **Address Offset:** 0x54
- **Read/write access:** read

Bits	Name	R/W	Description
31:1*~~GPIO_PWIDTH_B~~*	Reserved, read as zero		
~~GPIO_PWIDTH_B-1:0~~	External Port B	R	When Port B is configured as Input, then reading this location reads the values on the signal. When the data direction of Port B is set as Output, reading this location reads the data register for Port B. **Reset Value:** 0x0

15.3.3.23 gpio_ext_portc

- **Name:** External Port C
- **Size:** *~~GPIO_PWIDTH_C~~1'b1*
- **Address Offset:** 0x58
- **Read/write access:** read

Bits	Name	R/W	Description
31:1*~~GPIO_PWIDTH_C~~*	Reserved, read as zero		
~~GPIO_PWIDTH_C-1:0~~	External Port C	R	When Port C is configured as Input, then reading this location reads the values on the signal. When the data direction of Port C is set as Output, reading this location reads the data register for Port C. **Reset Value:** 0x0

15.3.3.24 gpio_ext_portd

- **Name:** External Port D
- **Size:** *1'b1~~GPIO_PWIDTH_D~~*
- **Address Offset:** 0x5c
- **Read/write access:** read

Bits	Name	R/W	Description

Bits	Name	R/W	Description
31:1~~*GPIO_PWIDTH_D*~~	Reserved, read as zero		
~~*GPIO_PWIDTH_D-1:0*~~	External Port D	R	When Port D is configured as Input, then reading this location reads the values on the signal. When the data direction of Port D is set as Output, reading this location reads the data register for Port D. **Reset Value:** 0x0

15.3.3.25 gpio_is_sync

- **Name:** Synchronization level
- **Size:** 1 bit
- **Address Offset:** 0x60
- **Read/write access:**
  - read/write when Port A is configured to generate interrupts (GPIO_PORTA_INTR = 1)
  - ~~read-only when GPIO_PORTA_INTR = 0~~

Bits	Name	R/W	Description
0	Synchronization level	R/W	Writing a 1 to this register results in all level-sensitive interrupts being synchronized to pclk_intr. 0 – No synchronization to pclk_intr (default) 1 – Synchronize to pclk_intr **Reset Value:** 0x0

~~15.3.3.26 gpio_id_code~~

- ~~**Name:** GPIO ID code~~
- ~~**Size:** *GPIO_ID_WIDTH*~~
- ~~**Address Offset:** 0x64~~
- ~~**Read/write access:** read~~

~~Bits~~	~~Name~~	~~R/W~~	~~Description~~
~~31:*GPIO_ID_WIDTH*~~	~~Reserved, read as zero~~		
~~*GPIO_ID_WIDTH-1:0*~~	~~GPIO ID code~~	~~R~~	~~This is a user-specified code that a system can read. It can be used for chip identification, and so on.~~ ~~**Reset Value:** *GPIO_ID_NUM*~~

~~15.3.3.27 gpio_ver_id_code~~

- ~~**Name:** GPIO Component Version~~
- ~~**Size:** 32 bits~~
- ~~**Address Offset:** 0x6c~~
- ~~**Read/write access:** read~~

~~Bits~~	~~Name~~	~~R/W~~	~~Description~~
~~31:*0*~~	~~GPIO Component Version~~	~~R~~	~~ASCII value for each number in the version, followed by *.~~ ~~**Reset Value:** See the releases table in the *Release Notes*~~

15.3.3.28 gpio_config_reg1

- **Name:** GPIO Configuration Register 1
- **Size:** 32 bits
- **Address Offset:** 0x74
- **Read/write access:** read

~~This register is present when the configuration parameter GPIO_ADD_ENCODED_PARAMS is set to True. If this parameter is set to False, this register reads back zero (0).~~

Bits	Name	R/W	Description
31:*21*	Reserved	R	Reserved
20:16	ENCODED_ID_WIDTH	R	~~5'd31. The value of this register is equal to GPIO_ID_WIDTH-1.~~

Bits	Name	R/W	Description
15	GPIO_ID	R	~~The value of this register is derived from the GPIO_ID configuration parameter.~~ 0 = Exclude ~~1 = Include~~
14	ADD_ENCODED_PARAMS	R	~~The value of this register is derived from the GPIO_ADD_ENCODED_PARAMS configuration parameter.~~ ~~0 = False~~ 1 = True
13	DEBOUNCE	R	~~The value of this register is derived from the GPIO_DEBOUNCE configuration parameter.~~ ~~0 = Exclude~~ 1 = Include
12	PORTA_INTR	R	~~The value of this register is derived from the GPIO_PORTA_INTR configuration parameter.~~ ~~0 = Exclude~~ 1 = Include
11	HW_PORTD	R	~~The value of this register is derived from the GPIO_HW_PORTD configuration parameter.~~ 0 = Exclude ~~1 = Include~~
10	HW_PORTC	R	~~The value of this register is derived from the GPIO_HW_PORTC configuration parameter.~~ 0 = Exclude ~~1 = Include~~
9	HW_PORTB	R	~~The value of this register is derived from the GPIO_HW_PORTB configuration parameter.~~ 0 = Exclude ~~1 = Include~~
8	HW_PORTA	R	~~The value of this register is derived from the GPIO_HW_PORTA configuration parameter.~~ 0 = Exclude ~~1 = Include~~
7	PORTD_SINGLE_CTL	R	~~The value of this register is derived from the GPIO_PORTD_SINGLE_CTL configuration parameter.~~ ~~0 = False~~ ~~1 = True~~
6	PORTC_SINGLE_CTL	R	~~The value of this register is derived from the GPIO_PORTC_SINGLE_CTL configuration parameter.~~ ~~0 = False~~ ~~1 = True~~
5	PORTB_SINGLE_CTL	R	~~The value of this register is derived from the GPIO_PORTB_SINGLE_CTL configuration parameter.~~ ~~0 = False~~ ~~1 = True~~
4	PORTA_SINGLE_CTL	R	~~The value of this register is derived from the GPIO_PORTA_SINGLE_CTL configuration parameter.~~ ~~0 = False~~ ~~1 = True~~
3:2	NUM_PORTS	R	~~The value of this register is derived from the GPIO_NUM_PORT configuration parameter.~~ ~~0x0 =1~~ ~~0x1 = 2~~ ~~0x2 = 3~~ 0x3 = 4

Bits	Name	R/W	Description
1:0	APB_DATA_WIDTH	R	~~The value of this register is derived from the GPIO_APB_DATA_WIDTH configuration parameter.~~ ~~0x0 = 8 bits~~ ~~0x1 = 16 bits~~ 0x2 = 32 bits ~~0x3 = Reserved~~

15.3.3.29 gpio_config_reg2

- **Name:** GPIO Configuration Register 2
- **Size:** 32 bits
- **Address Offset:** 0x70
- **Read/write access:** read

Bits	Name	R/W	Description
31:20	Reserved	R	Reserved
19:15	ENCODED_ID_PWIDTH_D	R	5'h1.
14:10	ENCODED_ID_PWIDTH_C	R	5'h1.
9:5	ENCODED_ID_PWIDTH_B	R	5'h1..
4:0	ENCODED_ID_PWIDTH_A	R	5'h1.

16 Pin out

16.1 Pinout Overview
16.2 Pinout Environment
16.3 Pinout Integration
16.4 Pinout Function Description
16.5 Pinout Programming Model
16.6 Pinout Register

17 Initialization

17.1 Initialization Overview

This chapter provides an overview of the requirements for initializing an device from power-on through OS load and applications running. An overview of the overall initialization process is presented, including both hardware and software-related steps, a general overview of the boot ROM operational requirements, and the behavioral expectations.

A brief overview of the whole initialization process and its steps is illustrated.

Initialization of device consists of several steps:

- Preinitialization
- Ramp sequence for power, clocks, and resets
- Boot ROM
- Boot loader (Initial software launched by the ROM code during flash booting phase)
- OS start

Each of these steps, up to OS/applications running, is explained in detail in the following sections.

The first two steps in the initialization process are more hardware oriented, but do require a good understanding of the process of configuring those system interface pads (pins or balls on the device) that have software configurable functionality. Pad configuration is an essential part of chip configuration and is application-dependent. This chapter refers to those pads and the associated configuration registers that are vital for proper device initialization.

It is highly recommended that users refer to the *Pinout* and *Configuration* chapters for further details regarding the use of the pad configuration registers.

17.2 Pre-initialization

Certain hardware configuration settings must be made in order to accomplish a successful boot-up operation with an device. The clock, reset, and power connections, as well as pads involved in setting the boot memory space for the MPU, must be connected and driven properly for successful device initialization. This section details the specific requirements for the pre-initialization stage.

- Power Connections
  The device can be supplied by an external companion chip.
- Clock configuration
- Reset configuration
- Boot configuration
- Pin Multiplexing and Pull-up/Pull-down

17.3 Power, Clock, and Reset Sequence on Power-up

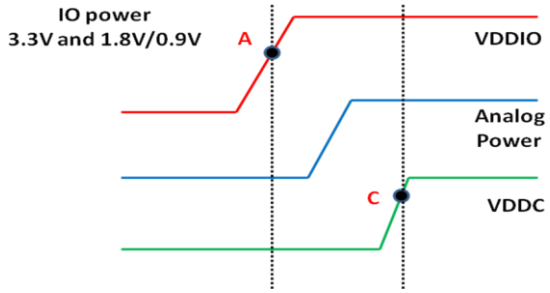
Power-up Sequence for the device

Device power supplies can be provided by external companion chips.

Below describes the power-on sequence for the device, including clocks, signals, and their relationship to the core and PLL voltages with the following steps:

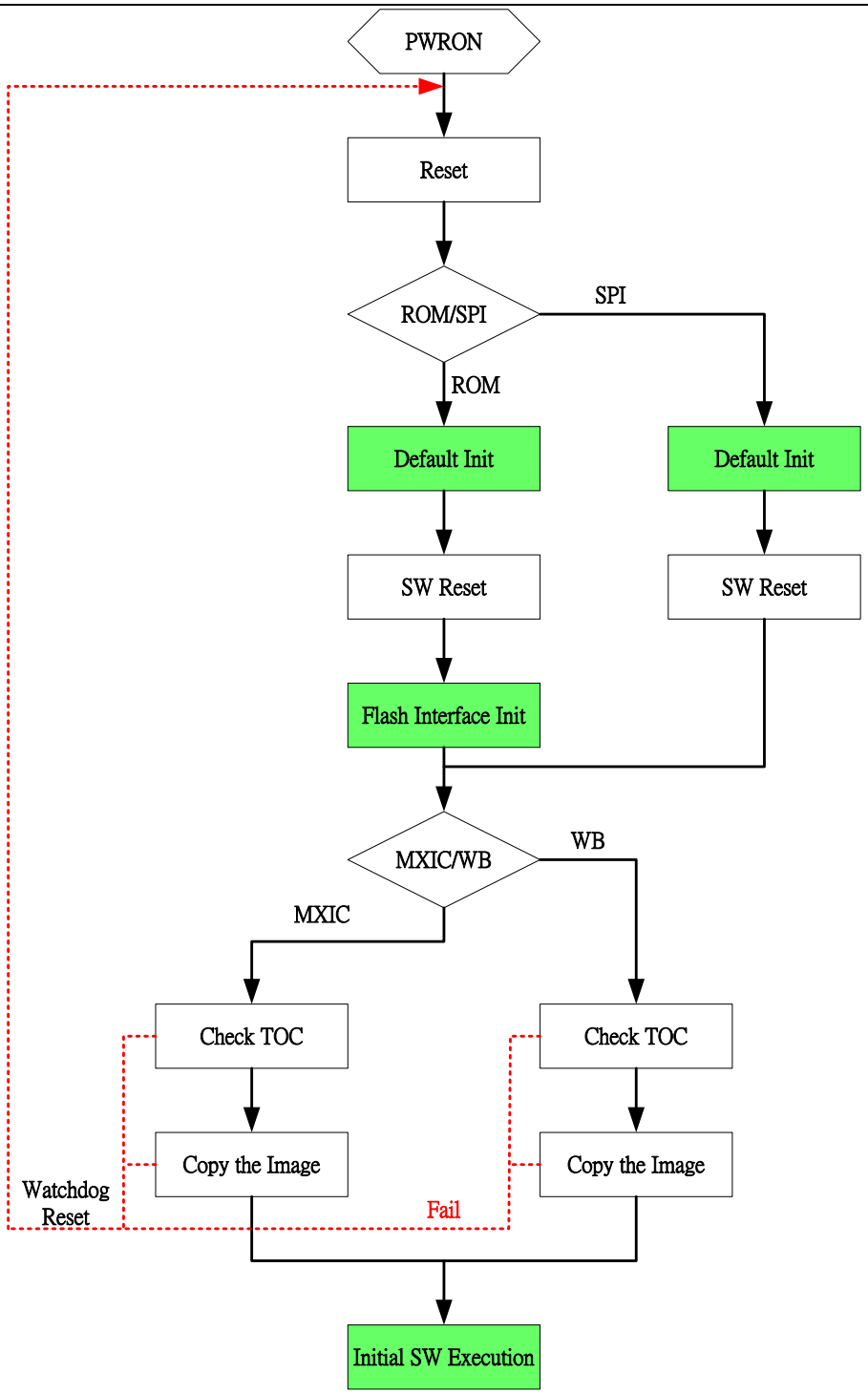
1) Keep RESET_N in low state.
2) Turn on I/O power. This includes 3.3V I/O power supply, DDR1.8V power supply and 0.9V reference voltage for DDR2 IO
3) Turn on 2.5V power to ADC/DAC/SNPS_PLL
4) Turn on 1.2V core power.
5) When the power supplies are stabilized, raise RESET_N from low to high and keep in high state during normal operation. And the CLK_25MHz and CLK_32MHz clock input is activated by external clock source.
6) Once the voltages and clock are ramped and stable, the sysbot pins are sampled and the device is ready to enable the boot ROM to run.

Refer to figure below. Point ‘A’ is VDDIO crossing point when I/O power on control (POC) circuit starts to take control. When POC control is active, all outputs are in high impedance state. Point ‘C’ is the 70% crossing point of core power. POC control turns off when point ‘C’ is reached.
As long as IO power is greater than core power by one diode voltage the SoC will power-up normally without inducing crowbar current. There is 10us minimum time requirement between point ‘A’ and point ‘C’. For best insurance, it is recommended to fully turn on VDDIO before turning on VDDC.



17.4 Device Booting by ROM code

This section describes the high-level booting concepts on the device and provides basic knowledge of booting on the device.
Booting is referred herein as the process of starting a bootstrap from the booting flash. The code executed during memory booting is usually used for OS start-up.
The ROM code finds the boot loader in external flash memory, copies it to external DDR, and executes it. The flow chart represents the ROM code structure.


PWRON
Reset
ROM/SPI
SPI
ROM
Default Init
Default Init
SW Reset
SW Reset
Flash Interface Init
MXIC/WB
WB
MXIC
Check TOC
Check TOC
Copy the Image
Copy the Image
Watchdog Reset
Fail
Initial SW Execution

18 Configuration

18.1 System Control Module Overview

18.2 Function Configuration Overview

18.3 I/O Multiplexing

18.4 Debug Module

19 Test Plan

UR0431A_PinList_0312.xls

20 Electrical Characteristics

21 Package Specifications

- Package Size (D)(E): 13*13 mm
- Package Type: LFBGA
- Ball count: 225
- Package height (A): 1.2 mm max.
- Substrate layer #: 4 layers
- Ball pitch (e): 0.8 mm
- Signal pin count: 140
- Pwr/Gnd pin count: 85
- Thermal: 1.5 W

22 Memory Usage

Memory	Size (Byte)	Usage
DCache	16K	Used for ARM data cache
ICache	16K	Used for ARM instruction cache
internal SRAM	64K	Used for internal buffer
Boot ROM	256x32bit	Used for ARM boot