WM0011

General Purpose Low-Power Audio DSP

DESCRIPTION

WM0011 Audio DSP provides Wolfson HD audio quality, with a power-budget targeted at handheld battery-powered audio devices.

WM0011 combines the advanced Tensilica HiFi EP™ audio DSP with an I/O and peripheral set optimized for flexible integration into smartphones, tablets and other portable consumer electronics devices. WM0011 is ideal for extremely power-efficient implementations of advanced voice enhancement, telephony noise reduction, voice and music CODECs and general audio enhancement.

A very wide range of audio CODECs, voice CODECs and third-party algorithms from such companies as Waves Audio, SRS Labs and Dolby are available, providing a rich portfolio of audio-processing options that can be integrated into a device with no additional software development.

WM0011 comes in a space-saving 3x3mm 49-ball W-CSP package with 0.4mm pitch.

APPLICATIONS

- Wireless audio devices – headsets, microphones, speakerphones
- Portable media devices
- Automotive
- General purpose digital signal processor for consumer audio applications
- Smartphones

FEATURES

- 260MHz Tensilica HiFi EP™ 24-bit audio digital signal processor
  - C-programmable with advanced debugging and profiling tool set
  - 256kB local RAM memory
  - 36kB Instruction / Data cache memory
  - 384kB general-purpose system RAM
  - Flexible boot options with 32kB boot ROM
  - 32 Channel DMA
  - XTAL or CMOS clock input
  - Low-power programmable PLL
- Security
  - Support for HW Authentication
  - Random Number Generator (RNG) to assist security algorithms
- Peripherals
  - SPI Master / Slave interface
  - 3 x multi-channel AIF interfaces, including I^2S and TDM
  - UART
  - I^2C Master
  - I^2C Slave
  - 3 x 32-bit general-purpose timer modules
  - Watchdog timer
  - On-chip JTAG debug unit and trace buffer
  - GPIO
- Software-defined standby modes for extended battery life

BLOCK DIAGRAM

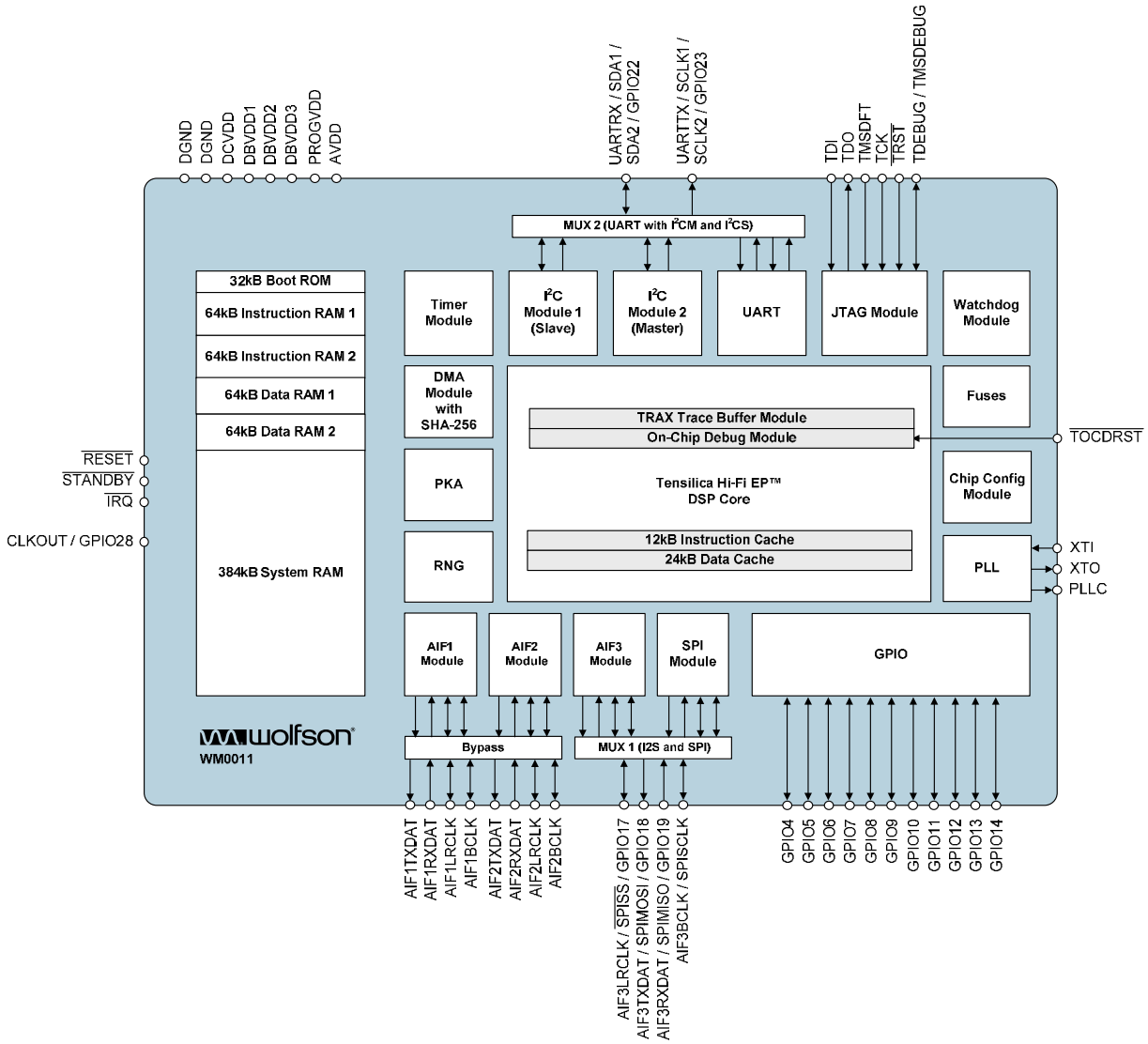
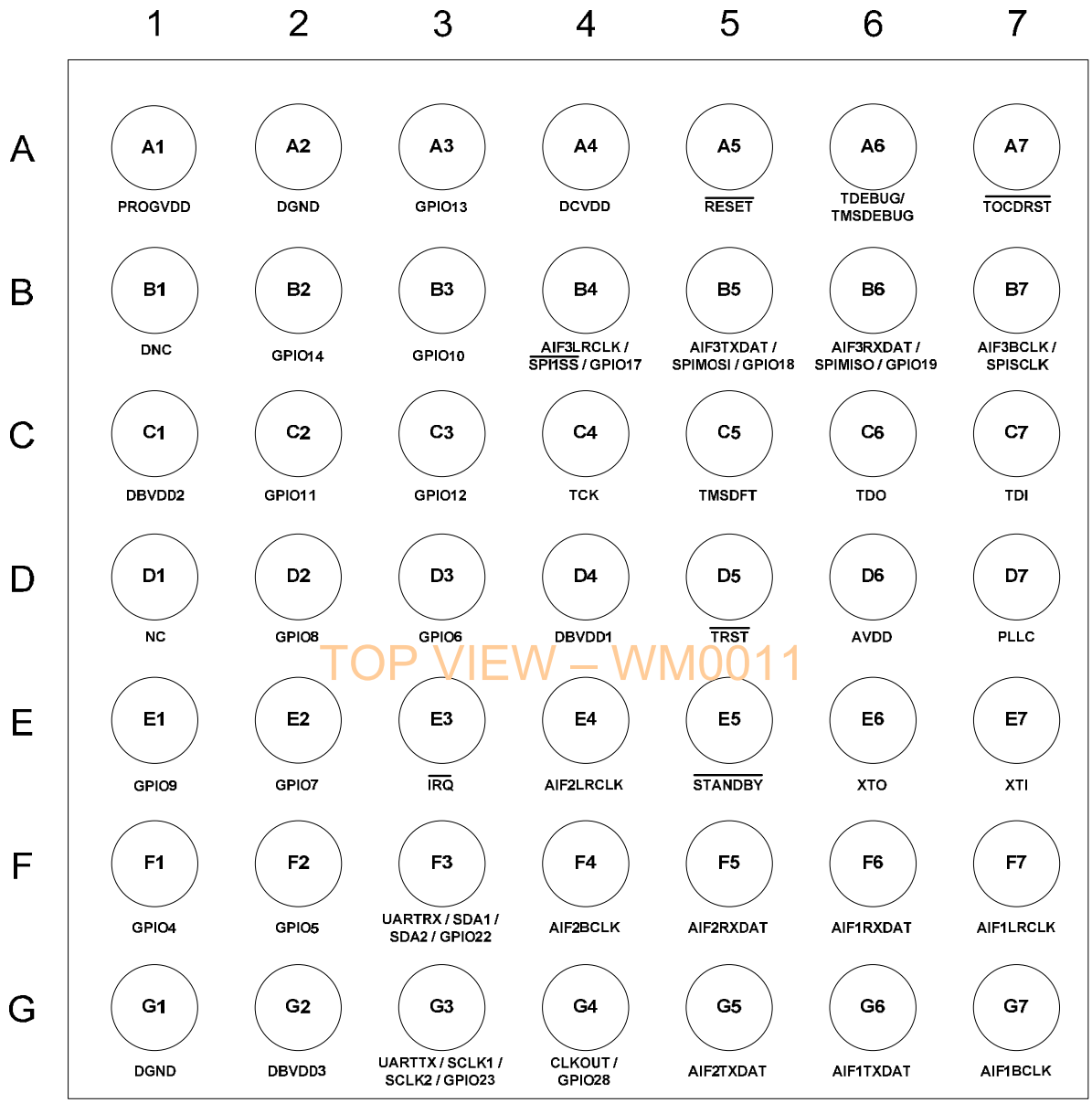


TABLE OF CONTENTS

PIN CONFIGURATION



ORDERING INFORMATION

DEVICE	CUSTOM FUSES	TEMPERATURE RANGE	PACKAGE	MOISTURE SENSITIVITY LEVEL	PEAK SOLDERING TEMPERATURE
WM0011ECS/R	Un-programmed	-40 to +85°C	49-ball W-CSP (Pb-free, Tape and reel)	MSL1	260°C
WM0011xxxECS/R	Custom-Programmed	-40 to +85°C	49-ball W-CSP (Pb-free, Tape and reel)	MSL1	260°C

Note:

Reel quantity = 5000

* xxx = Unique Custom Fuse part number

** Custom programmed minimum order quantity 50,000.

PIN DESCRIPTION

PIN NO	NAME	TYPE	PULL DEVICE	DESCRIPTION
Power and Ground Reference				
D4	DBVDD1	Supply	-	I/O supply (except GPIO pins 4...14)
A4	DCVDD	Supply	-	Core supply
A1	PROGVDD	Supply	-	Fuse programming supply. Connect to GND.
A2, G1	DGND	Supply	-	Ground
D6	AVDD	Supply	-	Analogue supply
D7	PLLC	Reference	-	PLL capacitor connection (0.1µF recommended)
C1	DBVDD2	Supply	-	I/O supply (GPIO10, GPIO11, GPIO12, GPIO13, GPIO14 pins)
G2	DBVDD3	Supply	-	I/O supply (GPIO4, GPIO5, GPIO6, GPIO7, GPIO8, GPIO9 pins)
Clock / Reset / Miscellaneous Interfaces				
E7	XTI	Input	-	Crystal connection or digital clock input
E6	XTO	Output	-	Crystal connection
A5	$\overline{\text{RESET}}$	Input	Pull-Up	Device reset
E5	$\overline{\text{STANDBY}}$	Input	Pull-Up	Standby input signal
E3	$\overline{\text{IRQ}}$	Output	Pull-Up	Interrupt output
G4	CLKOUT/GPIO28	Input / Output	Pull-Down	Reference clock output / GPIO pin
Audio Interface 1 (AIF1)				
G6	AIF1TXDAT	Output	Pull-Down	AIF1 data output
F6	AIF1RXDAT	Input	Pull-Down	AIF1 data input
F7	AIF1LRCLK	Input / Output	Pull-Down	AIF1 frame clock
G7	AIF1BCLK	Input / Output	Pull-Down	AIF1 bit clock
Audio Interface 2 (AIF2)				
G5	AIF2TXDAT	Output	Pull-Down	AIF2 data output
F5	AIF2RXDAT	Input	Pull-Down	AIF2 data input
E4	AIF2LRCLK	Input / Output	Pull-Down	AIF2 frame clock
F4	AIF2BCLK	Input / Output	Pull-Down	AIF2 bit clock
Audio Interface 3 (AIF3) / Control Interface (SPI)				
B5	AIF3TXDAT/SPIMOSI/GPIO18	Input / Output	Pull-Down	AIF3 data output / SPI Master Out Slave In / GPIO[1]
B6	AIF3RXDAT/SPIMISO/GPIO19	Input / Output	Pull-Down	AIF3 data input / SPI Master In Slave Out / GPIO[1]
B4	AIF3LRCLK/$\overline{\text{SPISS}}$/GPIO17	Input / Output	Pull-Up	AIF3 frame clock / SPI slave select / GPIO[1]

PIN NO	NAME	TYPE	PULL DEVICE	DESCRIPTION
B7	AIF3BCLK/SPISCLK	Input / Output	Pull-Down	AIF3 bit clock / SPI serial clock[1]
UART / I²C Master & Slave Interfaces				
F3	UARTRX/SDA1/SDA2/GPIO22	Input / Output	Pull-Down	UART RX / Serial data 1 (slave) / Serial data 2 (master) / GPIO[2]
G3	UARTTX/SCLK1/SCLK2/GPIO23	Input / Output	Pull-Down	UART TX / Serial clock 1 (slave) / Serial clock 2 (master) / GPIO[2]
GPIO				
F1	GPIO4	Input / Output	Pull-Up/Down	GPIO pin
F2	GPIO5	Input / Output	Pull-Up/Down	GPIO pin
D3	GPIO6	Input / Output	Pull-Up/Down	GPIO pin
E2	GPIO7	Input / Output	Pull-Up/Down	GPIO pin
D2	GPIO8	Input / Output	Pull-Up/Down	GPIO pin
E1	GPIO9	Input / Output	Pull-Up/Down	GPIO pin
B3	GPIO10	Input / Output	Pull-Up/Down	GPIO pin
C2	GPIO11	Input / Output	Pull-Up/Down	GPIO pin
C3	GPIO12	Input / Output	Pull-Up/Down	GPIO pin
A3	GPIO13	Input / Output	Pull-Up/Down	GPIO pin
B2	GPIO14	Input / Output	Pull-Up/Down	GPIO pin
Debug				
C4	TCK	Input	Pull-Up	JTAG clock
A6	TDEBUG/TMSDEBUG	Input / Output	Pull-Up	Test Mode Debug output / Test Mode Select input
C7	TDI	Input	Pull-Up	JTAG data input
C6	TDO	Output	Pull-Up	JTAG data output
C5	TMSDFT	Input	Pull-Up	JTAG mode select input
A7	$\overline{\text{TOCDRST}}$	Input	Pull-Up	Maskable chip reset from the debug tool
D5	$\overline{\text{TRST}}$	Input	Pull-Down	JTAG Test Access Port (TAP) block reset
Other				
B1	DNC			Do Not Connect
D1	NC			Not used - connect to GND.

Notes:

1. The SPI interface I/O pads are multiplexed with AIF3
2. The UART, I2C master and I2C slave signals are multiplexed into two I/O pads.
3. The I/O pad multiplexers are configured during the boot-up sequence, as determined by the Custom Fuse settings.

Table 1 identifies the default power-up condition of each of the input / output pins, assuming that the Custom Fuses are not programmed.

Application-specific parameters for configuring the input / output pins, and many other parameters, may be selected using the integrated one-time-programmable fuses. See “Boot Sequence Control” for further details.

PIN NO	NAME	DEFAULT FUNCTION / RESET CONDITION (FUSES NOT PROGRAMMED)		
E7	XTI	XTI	input	
E6	XTO	XTO	output	
A5	$\overline{\text{RESET}}$	$\overline{\text{RESET}}$	input	Pull-up enabled
E5	$\overline{\text{STANDBY}}$	$\overline{\text{STANDBY}}$	input	Pull-up enabled
E3	$\overline{\text{IRQ}}$	$\overline{\text{IRQ}}$	output	Pull-up enabled
G4	CLKOUT/GPIO28	CLKOUT	output	Pull-down enabled
G6	AIF1TXDAT	AIF1TXDAT	output	Pull-down enabled
F6	AIF1RXDAT	AIF1RXDAT	input	Pull-down enabled

PIN NO	NAME	DEFAULT FUNCTION / RESET CONDITION (FUSES NOT PROGRAMMED)		
F7	AIF1LRCLK	AIF1LRCLK	input	Pull-down enabled
G7	AIF1BCLK	AIF1BCLK	input	Pull-down enabled
G5	AIF2TXDAT	AIF2TXDAT	output	Pull-down enabled
F5	AIF2RXDAT	AIF2RXDAT	input	Pull-down enabled
E4	AIF2LRCLK	AIF2LRCLK	input	Pull-down enabled
F4	AIF2BCLK	AIF2BCLK	input	Pull-down enabled
B5	AIF3TXDAT/SPIMOSI/GPIO18	SPIMOSI	output	Pull-down enabled
B6	AIF3RXDAT/SPIMISO/GPIO19	SPIMISO	input	Pull-down enabled
B4	AIF3LRCLK/$\overline{\text{SPISS}}$/GPIO17	$\overline{\text{SPISS}}$	output	Pull-up enabled
B7	AIF3BCLK/SPISCLK	SPISCLK	output	Pull-down enabled
F3	UARTRX/SDA1/SDA2/GPIO22	UARTRX	input	Pull-down enabled
G3	UARTTX/SCLK1/SCLK2/GPIO23	UARTTX	output	Pull-down enabled whilst $\overline{\text{RESET}}$ is asserted. Pull-down is disabled after $\overline{\text{RESET}}$ is released. UARTTX is then actively driven.
F1	GPIO4	[Disabled]	input/output	Pull-down enabled
F2	GPIO5	[Disabled]	input/output	Pull-down enabled
D3	GPIO6	[Disabled]	input/output	Pull-down enabled
E2	GPIO7	[Disabled]	input/output	Pull-down enabled
D2	GPIO8	[Disabled]	input/output	Pull-down enabled
E1	GPIO9	[Disabled]	input/output	Pull-down enabled
B3	GPIO10	[Disabled]	input/output	Pull-down enabled whilst $\overline{\text{RESET}}$ is asserted. Pull-up is enabled after $\overline{\text{RESET}}$ is released.
C2	GPIO11	[Disabled]	input/output	Pull-down enabled
C3	GPIO12	[Disabled]	input/output	Pull-down enabled
A3	GPIO13	[Disabled]	input/output	Pull-down enabled
B2	GPIO14	[Disabled]	input/output	Pull-down enabled
C4	TCK	TCK	input	Pull-up enabled
A6	TDEBUG/TMSDEBUG	TDEBUG/TMSDEBUG		Pull-up enabled
C7	TDI	TDI	input	Pull-up enabled
C6	TDO	TDO	output	Pull-down enabled
C5	TMSDFT	TMSDFT	input	Pull-up enabled
A7	$\overline{\text{TOCDRST}}$	$\overline{\text{TOCDRST}}$	input	Pull-up enabled
D5	$\overline{\text{TRST}}$	$\overline{\text{TRST}}$	input	Pull-down enabled

Table 1 Default Pin Conditions (assuming Fuses are not programmed)

ABSOLUTE MAXIMUM RATINGS

Absolute Maximum Ratings are stress ratings only. Permanent damage to the device may be caused by continuously operating at or beyond these limits. Device functional operating limits and guaranteed performance specifications are given under Electrical Characteristics at the test conditions specified.

ESD Sensitive Device. This device is manufactured on a CMOS process. It is therefore generically susceptible to damage from excessive static voltages. Proper ESD precautions must be taken during handling and storage of this device.

Wolfson tests its package types according to IPC/JEDEC J-STD-020 for Moisture Sensitivity to determine acceptable storage conditions prior to surface mount assembly. These levels are:

MSL1 = unlimited floor life at <30°C / 85% Relative Humidity. Not normally stored in moisture barrier bag.
MSL2 = out of bag storage for 1 year at <30°C / 60% Relative Humidity. Supplied in moisture barrier bag.
MSL3 = out of bag storage for 168 hours at <30°C / 60% Relative Humidity. Supplied in moisture barrier bag.

The Moisture Sensitivity Level for each package type is specified in Ordering Information.

CONDITION	MIN	MAX
Supply voltage (DCVDD)	DGND - 0.3V	1.6V
Supply voltage (DBVDD1, DBVDD2, DBVDD3, AVDD, PROGVDD)	DGND - 0.3V	5.0V
Voltage range digital inputs (DBVDD1 domain)	DGND - 0.3V	DBVDD1 + 0.3V
Voltage range digital inputs (DBVDD2 domain)	DGND - 0.3V	DBVDD2 + 0.3V
Voltage range digital inputs (DBVDD3 domain)	DGND - 0.3V	DBVDD3 + 0.3V
Operating temperature range, T_A	-40ºC	+85ºC
Junction temperature, T_J	-40ºC	+125ºC
Storage temperature after soldering	-65ºC	+150ºC

RECOMMENDED OPERATING CONDITIONS

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Digital core supply range	DCVDD	1.14	1.2	1.32	V
Digital I/O supply range	DBVDD1	1.62	1.8	1.98	V
Digital I/O supply range (GPIO10, GPIO11, GPIO12, GPIO13, GPIO14)	DBVDD2	1.62		3.63	V
Digital I/O supply range (GPIO4, GPIO5, GPIO6, GPIO7, GPIO8, GPIO9)	DBVDD2	1.62		3.63	V
PLL supply range	AVDD	1.14	1.2	1.32	V
Fuse programming supply	PROGVDD		0		V
Ground	DGND		0		V
Operating temperature range	T_A	-40		+85	ºC

Notes:

1. All supplies are independent of each other (i.e. not internally connected)
2. PROGVDD must be tied to 0V during normal operation
3. The WM0011 can operate with DBVDD2 tied to 0V, but GPIO10, GPIO11, GPIO12, GPIO13, GPIO14 functionality is not supported in this case
4. The WM0011 can operate with DBVDD3 tied to 0V, but GPIO4, GPIO5, GPIO6, GPIO7, GPIO8, GPIO9 functionality is not supported in this case

THERMAL PERFORMANCE

Thermal analysis should be performed in the intended application to prevent the WM0011 from exceeding maximum junction temperature. Several contributing factors affect thermal performance most notably the physical properties of the mechanical enclosure, location of the device on the PCB in relation to surrounding components and the number of PCB layers. Connecting the GND pin through thermal vias and into a large ground plane will aid heat extraction.

Three main heat transfer paths exist to surrounding air as illustrated below in:

- Package top to air (radiation)
- Package bottom to PCB (radiation)
- Package pins to PCB (conduction)

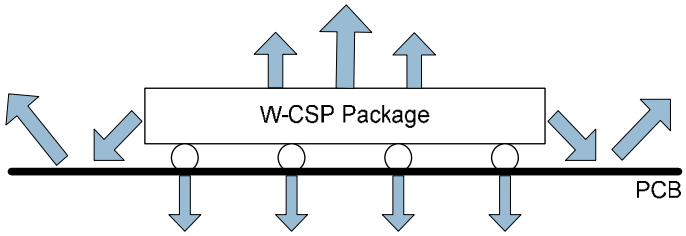


Figure 1 Heat Transfer Paths

The temperature rise T_R is given by $T_R = P_D * \Theta_{JA}$

- P_D is the power dissipated in the device.
- Θ_{JA} is the thermal resistance from the junction of the die to the ambient temperature and is therefore a measure of heat transfer from the die to surrounding air. Θ_{JA} is determined with reference to JEDEC standard JESD51-9.

The junction temperature T_J is given by $T_J = T_A + T_R$, where T_A is the ambient temperature.

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Ambient Temperature	T_A	-40		+85	°C
Junction Temperature	T_J	-40		+125	°C
Thermal Resistance	Θ_{JA}		58		°C/W

Note:

1. Junction temperature is a function of ambient temperature and of the device operating conditions. The ambient temperature limits and junction temperature limits must both be observed.
2. Thermal resistance (Θ_{JA}) is measured using JESD51-2 methodology

ELECTRICAL CHARACTERISTICS

Test Conditions

DCVDD=AVDD=1.2V, DBVDD1=DBVDD2=DBVDD3=1.8V, T_A = +25°C

PARAMETER	SYMBOL	TEST CONDITIONS	MIN	TYP	MAX	UNIT
Digital Input / Output						
Input HIGH Level, GPIO[4..9] pads	V_{IH}		0.65 x V_{DBVDD3}			V
Input LOW Level, GPIO[4..9] pads	V_{IL}				0.35 x V_{DBVDD3}	V
Input HIGH Level, GPIO[10..14] pads	V_{IH}		0.65 x V_{DBVDD2}			V
Input LOW Level, GPIO[10..14] pads	V_{IL}				0.35 x V_{DBVDD2}	V
Input HIGH Level, All other digital pads	V_{IH}		0.65 x V_{DBVDD1}			V
Input LOW Level, All other digital pads	V_{IL}				0.35 x V_{DBVDD1}	V
Output HIGH Level, GPIO[4..9] pads	V_{OH}	I_{OH} = 5mA Full strength output drive (*_DS = 1)	0.75 x V_{DBVDD3}			V
Output LOW Level, GPIO[4..9] pads	V_{OL}	I_{OL} = -5mA Full strength output drive (*_DS = 1)			0.25 x V_{DBVDD3}	V
Output HIGH Level, GPIO[10..14] pads	V_{OH}	I_{OH} = 5mA Full strength output drive (*_DS = 1)	0.75 x V_{DBVDD2}			V
Output LOW Level, GPIO[10..14] pads	V_{OL}	I_{OL} = -5mA Full strength output drive (*_DS = 1)			0.25 x V_{DBVDD2}	V
Output HIGH Level, All other digital pads	V_{OH}	I_{OH} = 1mA Full strength output drive (*_DS = 1)	0.75 x V_{DBVDD1}			V
Output LOW Level, All other digital pads	V_{OL}	I_{OL} = -1mA Full strength output drive (*_DS = 1)			0.25 x V_{DBVDD1}	V
Input Capacitance	C_{IN}				2.8	pF
Input Leakage			−10		+10	µA
Pull-up resistance, GPIO[4..14] pads		Pull-Up enabled for the respective pad (*_PU = 1)		61		kΩ
Pull-down resistance, GPIO[4..14] pads		Pull-Down enabled for the respective pad (*_PD = 1)		61		kΩ
Pull-up resistance, All other digital pads		Pull-Up enabled for the respective pad (*_PU = 1)		38		kΩ
Pull-down resistance, All other digital pads		Pull-Down enabled for the respective pad (*_PD = 1)		40		kΩ

Selectable output drive strength control is provided on the digital output pads, using the *_DS register bits. The reduced drive strength option may be used at lower clock speeds, if preferred. Specific characteristic data for reduced drive strength is not available.

TYPICAL POWER CONSUMPTION

Typical power consumption data is provided below for a number of different operating conditions.

Test Conditions:
DCVDD = AVDD = 1.2V, DBVDD1 = 1.8V, DBVDD2 = DBVDD3 = 0V, T_A = +25ºC

OPERATING MODE	TEST CONDITIONS	I_{DCVDD}	I_{DBVDD1}	I_{AVDD}	TOTAL
Reset	$\overline{\text{RESET}}$ asserted CLKIN = 0MHz	0.2mA	0.03mA	0.05mA	0.35mW
BootROM (awaiting code download)	$\overline{\text{RESET}}$ de-asserted CLKIN = 24.576MHz	8.78mA	0.48mA	0.05mA	11.46mW
Sleep Mode	$\overline{\text{RESET}}$ de-asserted SLP_ENA=1 (CCM_WKUP_CTRL register) CLKIN = 0MHz DSPCLK disabled, AHBCLK disabled	0.25mA	0.02mA	0.05mA	0.40mW
	$\overline{\text{RESET}}$ de-asserted SLP_ENA=1 (CCM_WKUP_CTRL register) CLKIN = 24.576MHz DSPCLK disabled, AHBCLK disabled RAM & IRQC modules enabled	0.91mA	0.47mA	0.05mA	2.00mW
Sleep Mode AIF Bypass enabled	$\overline{\text{RESET}}$ de-asserted SLP_ENA=1 (CCM_WKUP_CTRL register) CLKIN = 0MHz DSPCLK disabled, AHBCLK disabled AIF Bypass Mode A enabled	0.27mA	0.16mA	0.05mA	0.67mW
	$\overline{\text{RESET}}$ de-asserted SLP_ENA=1 (CCM_WKUP_CTRL register) CLKIN = 24.576MHz DSPCLK disabled, AHBCLK disabled AIF Bypass Mode A enabled	0.95mA	0.60mA	0.05mA	2.28mW
Run Mode (full processor load)	$\overline{\text{RESET}}$ de-asserted SLP_ENA=0 (CCM_WKUP_CTRL register) CLKIN = 24.576MHz PLLOUT = 259.2MHz All peripherals enabled Processor fully loaded	90mA	0.60mA	0.10mA	109.2mW

The WM0011 supports a low-power Sleep mode, as referenced above. Note that, when the WM0011 is not in use, the Sleep mode (not the Reset mode) is recommended for typical applications. The Sleep mode allows the full processor functionality to be resumed at any time, without needing to re-load the software code. The Sleep mode also enables AIF Bypass modes to be selected.

SIGNAL TIMING REQUIREMENTS

SYSTEM CLOCK & PHASE LOCKED LOOP (PLL)

Test Conditions
DCVDD=AVDD=1.2V, DBVDD1=DBVDD2=DBVDD3=1.8V, T_A = +25°C

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
External Clock Timing					
Chip Clock Input	CLKIN			26	MHz
Alternate Clock Input	ALTCLK			26	MHz
Timer Clock Trigger	TMRCLK			26	MHz
Input Clock duty cycle		40		60	%
Phase Locked Loop (PLL)					
PLL input frequency	CLKIN	5		26	MHz
PLL input duty cycle		40		60	%
PLL output frequency	PLLOUT	6.25		260	MHz
PLL lock time				2	ms
Internal Clock Timing					
DSP Core Clock	DSPCLK			260	MHz
AHB Bus Clock	AHBCLK			130	MHz
APB Bus Clock	APBCLK			130	MHz

Table 2 System Clock and Phase Locked Loop (PLL)

The WM0011 incorporates a 2-stage cascaded PLL circuit; the PLL timing parameters above refer to the 2-stage circuit in its entirety. Note that the specified frequency limits are not applicable to the internal reference points within the cascaded PLL circuits.

AUDIO INTERFACE (AIF) TIMING

DIGITAL AUDIO INTERFACE – MASTER MODE

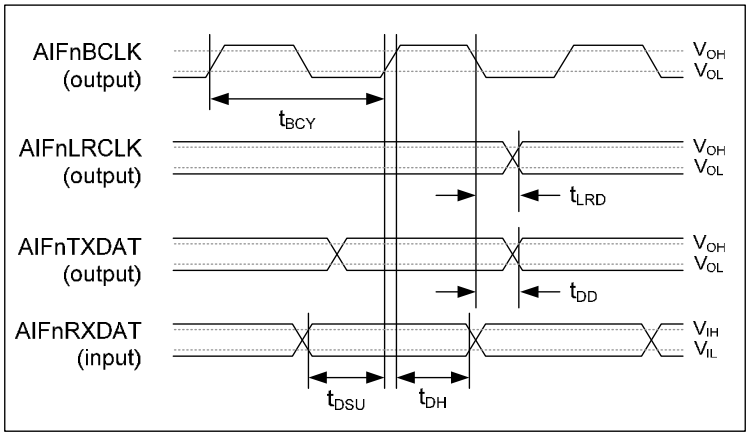


Figure 2 AIF Interface Timing – Master Mode

Test Conditions

DCVDD=AVDD=1.2V, DBVDD1=DBVDD2=DBVDD3=1.8V, T_A = +25°C, C_{LOAD}=5pF (output pins)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Audio Interface Timing - Master Mode					
AIFnBCLK cycle time	t_{BCY}	80			ns
AIFnBCLK duty cycle		40		60	%
AIFnLRCLK propagation delay from AIFnBCLK falling edge	t_{LRD}	0		15	ns
AIFnTXDAT propagation delay from AIFnBCLK falling edge	t_{DD}	0		15	ns
AIFnRXDAT setup time to AIFnBCLK rising edge	t_{DSU}	16.3			ns
AIFnRXDAT hold time from AIFnBCLK rising edge	t_{DH}	16.3			ns

Table 3 AIF Master Mode Timing Values

Note the timing figures quoted in the table above are for full drive strength outputs; these timings are not guaranteed for reduced drive strength.

DIGITAL AUDIO INTERFACE – SLAVE MODE

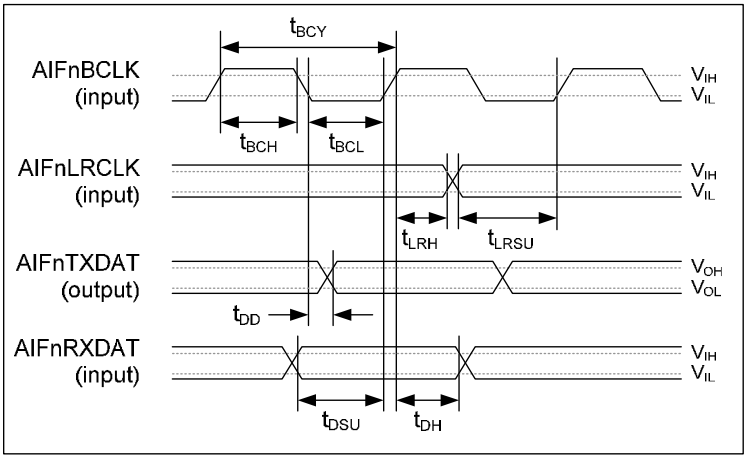


Figure 3 AIF Interface Timing – Slave Mode

Test Conditions

DCVDD=AVDD=1.2V, DBVDD1=DBVDD2=DBVDD3=1.8V, T_A = +25°C, C_{LOAD}=5pF (output pins)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Audio Interface Timing - Slave Mode					
AIFnBCLK cycle time	t_{BCY}	80			ns
AIFnBCLK duty cycle		35		65	%
AIFnLRCLK set-up time to AIFnBCLK rising edge	t_{LRSU}	16.3			ns
AIFnLRCLK hold time from AIFnBCLK rising edge	t_{LRH}	7.5			ns
AIFnRXDAT hold time from AIFnBCLK rising edge	t_{DH}	10			ns
AIFnTXDAT propagation delay from AIFnBCLK falling edge	t_{DD}	0		12	ns
AIFnRXDAT set-up time to AIFnBCLK rising edge	t_{DSU}	16.3			ns

Table 4 AIF Slave Mode Timing Values

Note the timing figures quoted in the table above are for full drive strength outputs; these timings are not guaranteed for reduced drive strength.

SPI INTERFACE TIMING

SPI INTERFACE – MASTER MODE

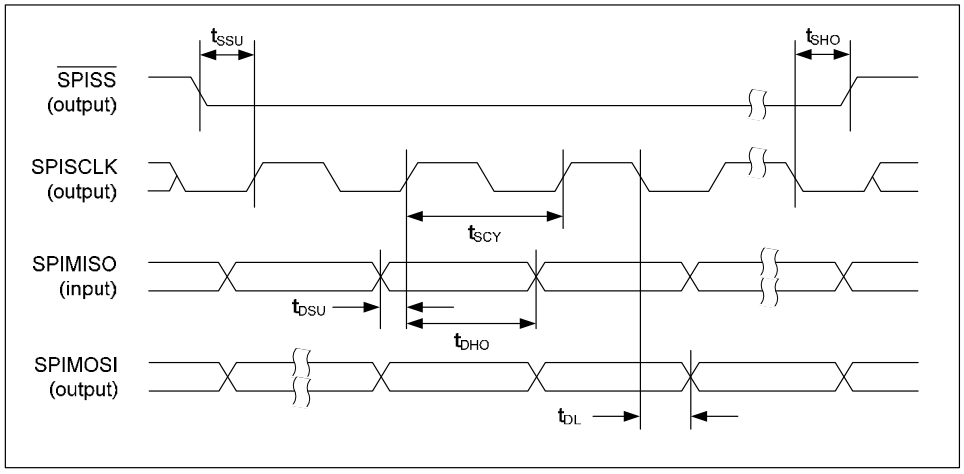


Figure 4 SPI Master Mode Timing

Note this diagram shows the mode where incoming data (SPIMISO) is sampled on the rising edge of SPISCLK, and outgoing data (SPIMOSI) transitions on the falling edge of SPISCLK.

Test Conditions

AVDD=DCVDD=1.2V, DBVDD1=DBVDD2=DBVDD3=1.8V, T_A = +25°C, C_{LOAD}=5pF (output pins), unless otherwise stated.

PARAMETER		SYMBOL	MIN	TYP	MAX	UNIT
SPI Interface Timing - Master Mode						
$\overline{SPISS}$ set-up time to SPISCLK rising edge		t_{SSU}	25			ns
$\overline{SPISS}$ hold time from SPISCLK falling edge		t_{SHO}	25			ns
SPISCLK pulse cycle time		t_{SCY}	61.6			ns
In SPI Master mode, the maximum SPISCLK frequency is 16.25MHz. It is also required that $F_{SPISCLK} \leq F_{AHBCLK}/8$.						
SPISCLK duty cycle			40		60	%
SPIMISO set-up time to SPISCLK rising edge		t_{DSU}	10.5			ns
SPIMISO hold time from SPISCLK rising edge		t_{DHO}	2.0			ns
SPIMOSI propagation delay from SPISCLK falling edge	5pF, reduced drive strength	t_{DL}			5.1	ns
	5pF, full drive strength				4.7	
	25pF, reduced drive strength				6.3	
	25pF, full drive strength				8.7	

Table 5 SPI Master Mode Timing Values

Note the timing figures quoted in the table above are for full drive strength outputs (except where otherwise stated); these timings are not guaranteed for reduced drive strength.

SPI INTERFACE – SLAVE MODE

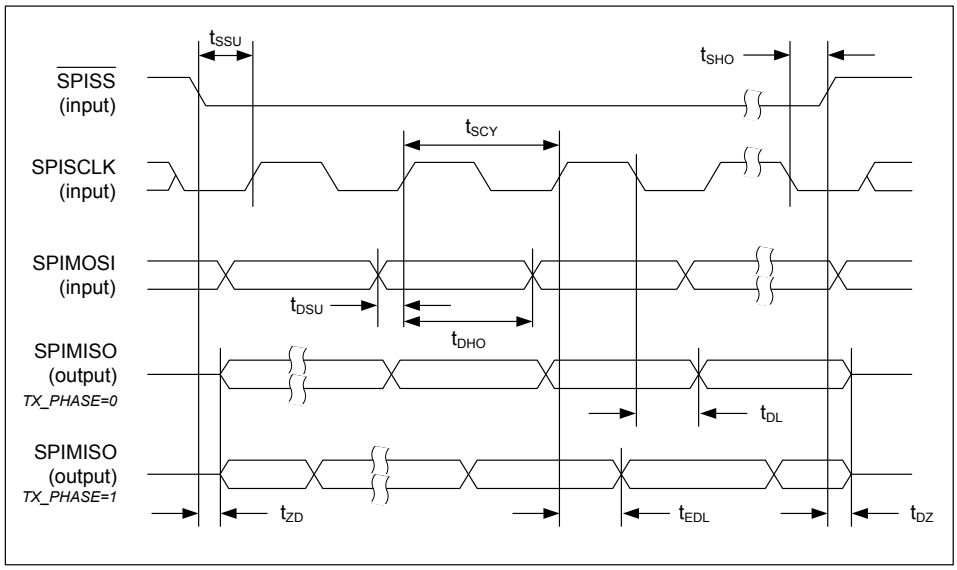


Figure 5 SPI Slave Mode Timing

Note this diagram shows the mode where incoming data (SPIMOSI) is sampled on the rising edge of SPISCLK. By default, the outgoing data (SPIMISO) transitions on the falling edge of SPISCLK. When 'Early Transmit Data Phase' mode is selected (TX_PHASE=1), the outgoing data (SPIMISO) transitions on the rising edge of SPISCLK.

Test Conditions

AVDD=DCVDD=1.2V, DBVDD1=DBVDD2=DBVDD3=1.8V, T_A = +25°C, C_{LOAD}=5pF (output pins), unless otherwise stated.

PARAMETER		SYMBOL	MIN	TYP	MAX	UNIT
SPI Interface Timing - Slave Mode						
$\overline{SPISS}$ set-up time to SPISCLK rising edge		t_{SSU}	t_{AHBCLK} + 1.0			ns
$\overline{SPISS}$ hold time from SPISCLK falling edge		t_{SHO}	2.0			ns
SPISCLK pulse cycle time		t_{SCY}	38.5			ns
In SPI Slave mode, the maximum SPISCLK frequency is 26MHz. It is also required that $F_{SPISCLK} < F_{AHBCLK}$.						
SPISCLK duty cycle			40		60	%
SPIMOSI set-up time to SPISCLK rising edge		t_{DSU}	2.0			ns
SPIMOSI hold time from SPISCLK rising edge		t_{DHO}	2.0			ns
SPIMISO propagation delay from SPISCLK falling edge	C_{LOAD}=25pF	t_{DL}			12.1	ns
	C_{LOAD}=5pF				9.3	
SPIMISO propagation delay from SPISCLK rising edge (early TX data mode)	C_{LOAD}=25pF	t_{EDL}			14.1	ns
	C_{LOAD}=5pF				11.3	
SPIMISO enable from $\overline{SPISS}$ falling edge		t_{ZD}			13.6	ns
SPIMISO disable from $\overline{SPISS}$ rising edge		t_{DZ}			7.8	ns

Table 6 SPI Slave Mode Timing Values

Note the timing figures quoted in the table above are for full drive strength outputs; these timings are not guaranteed for reduced drive strength.

CONTROL INTERFACE (I2C) TIMING

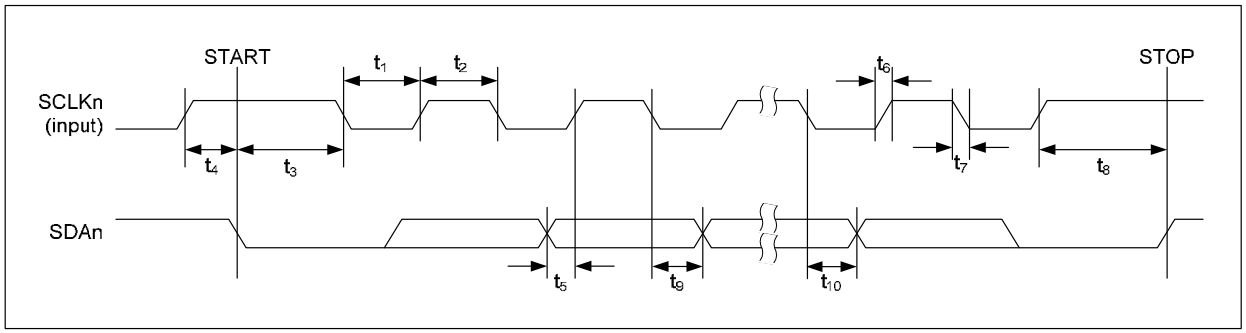


Figure 6 I²C Control Interface Timing

Test Conditions

AVDD= DCVDD=1.2V, DBVDD1=DBVDD2=DBVDD3=1.8V, T_A = +25°C, unless otherwise stated.

PARAMETER		SYMBOL	MIN	TYP	MAX	UNIT
SCLKn Frequency					1000	kHz
SCLKn Low Pulse-Width		t_1	500			ns
SCLKn High Pulse-Width		t_2	260			ns
Hold Time (Start Condition)	Pulse filter OFF	t_3	260			ns
	Pulse filter ON		275			
Setup Time (Start Condition)		t_4	260			ns
SDAn, SCLKn Rise Time		t_6			120	ns
SDAn, SCLKn Fall Time		t_7			120	ns
Setup Time (Stop Condition)		t_8	260			ns
SDAn Setup Time (data/ACK input)		t_5	50			ns
SDAn Hold Time (data/ACK input)		t_9	0			ns
SDAn Valid Time (data/ACK output)		t_{10}			450	ns
Pulse width of spikes that will be suppressed		t_{ps}	0		50	ns

Table 7 I²C Timing Values

DEVICE DESCRIPTION

INTRODUCTION

The WM0011 is an audio DSP designed for smartphones and other high performance audio applications. The architecture is optimised for multi-channel audio processing such as software CODECs, equalisation, compression and echo cancellation.

BLOCK DIAGRAM

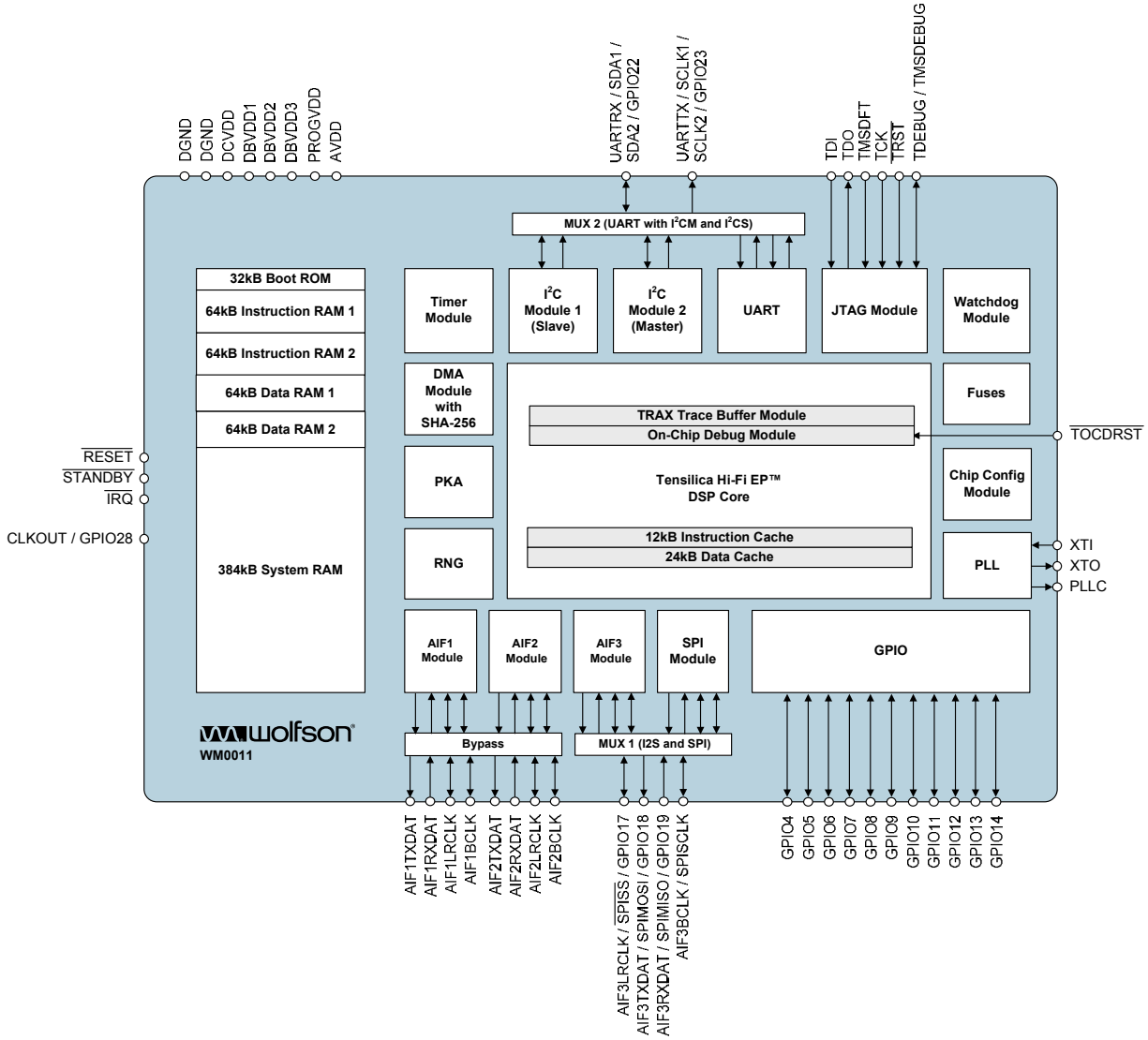


Figure 7 WM0011 Block Diagram

DESCRIPTION OF MODULES

BOOT ROM

The 32kB boot ROM allows the WM0011 to boot from a variety of sources. These are listed in the table below.

MODULE	BOOTABLE FROM
SPI Slave	External host processor
SPI Master	SST25WFxxx SPI Serial Flash

TIGHTLY-COUPLED MEMORY (TCM) RAM

The DSP's primary memory comprises 64-bit wide, zero-latency tightly coupled memory.

- 128kB of instruction RAM
- 128kB of data RAM

MULTI-PURPOSE RAM

The 384kB system RAM is connected to the DSP via the system bus. This RAM can be used for storing either instructions or data. Both data and instructions can be transferred in and out of TCM by DMA.

TENSILICA HIFI EP™ DSP CORE

The core combines a 24-bit audio DSP engine that has been optimised for highly efficient high-resolution audio processing, with a GCC-compatible (GNU Compiler Collection) general purpose RISC instruction set. It includes logic to interface with the AHB bus and the TRAX Trace Buffer, and to the JTAG TAP controller in order to provide support for On-Chip Debug (OCD). The HiFi EP™ features:

- 7-stage instruction pipeline
- One load-store unit
- 12kB 3-way Instruction Cache (64-bit width), and corresponding TAG memory
- 24kB 3-way Data Cache (64-bit width), and corresponding TAG memory
- Pre-fetch buffering for slow external RAM support
- TCM Core Instruction RAM (64-bit width)
- PIF-to-AHB-Lite Bridge, Synchronous, 64-bit width
- Three general purpose timers
- Interrupt controller, with sixteen external and five internal interrupt inputs
- On-Chip Debug (OCD) support
- Trace port and corresponding 1kB TRAX trace RAM (32-bit width)
- Floating point accelerator

For more detailed information on the Tensilica HiFi EP™ core, refer to the 'HiFi EP Audio Engine Instruction Set Architecture Reference Manual' (HIFIEP-ISA-rm.pdf), available from Tensilica.

TIMER MODULE

Three 32-bit general-purpose timers are provided. The timers can be configured as up-counters or as down-counters. The Timer block features include:

- Free-running counter operation (triggered internally or externally)
- Event counter operation (externally triggered)
- One-shot operation from either an external or an internal trigger

WATCHDOG MODULE

A watchdog timer block is provided as a means to reset the WM0011 chip in the case of a software failure. A timeout of the watchdog produces a Warm Reset (maskable) that includes a reset of most registers and state machines, and of the PLL.

GPIO MODULE

There is one GPIO controller controlling seventeen multiplexed GPIO inputs. Two of these inputs can be selected as an interrupt to HiFi EP™, or one can be selected to be used as an input to the IRQC controller.

IRQC MODULE

The IRQC controller provides fine control of interrupts (edge control, etc). It also enables wake-up, and controls the external $\overline{\text{IRQ}}$ output pin. Table 8 shows the IRQC assignments.

IRQC BIT	DIRECTION	DESCRIPTION
15	Output	Software interrupt – HiFi EP™ interrupt input
14	Output	Software interrupt – HiFi EP™ interrupt input
13	Input	Reserved
12	Input	Reserved
11	Input	Timer 2 interrupt
10	Input	Timer 1 interrupt
9	Input	DMA interrupt
8	Input	Watchdog interrupt
7	Input	$\overline{\text{STANDBY}}$ pin (Active Low) Note this input is inverted internally, and is therefore Active High at the input to the IRQC module.
6	Input	I2C interrupt
5	Input	AIF 2 interrupt
4	Input	AIF 1 interrupt
3	Input	UART interrupt
2	Input	SPI interrupt
1	Input	Cascaded interrupt input from the GPIO controller (Active Low)
0	Output	$\overline{\text{IRQ}}$ pin (Active Low)

Table 8 IRQC Interrupt assignment

I²C MASTER AND SLAVE MODULE

The I²C module provides two independent I²C buses. These are configured as one master and one slave. External pins are multiplexed such that only one of the I²C Master, the I²C slave, or the UART can be configured at any one time.

I²C Master:

- 100kHz, 400kHz and 1MHz operation
- Single master

I²C Slave:

- 100kHz, 400kHz and 1MHz operation
- Clock Stretching

PLL MODULE

An integrated cascade PLL can synthesise all internal clocks from a CMOS external reference clock or a directly-connected crystal. The two-stage PLL can generate accurate standard audio sampling frequencies from a wide range of reference frequencies. The cascade PLL provides a single lock indicator.

AUDIO INTERFACE (AIF) MODULES

The Audio Interface module transmits and receives a wide range of commonly used serial digital audio formats, including I²S and multi-channel TDM. It has two independent serial data lines with a shared bit clock and a shared frame clock for transmit and receive.

Data is typically transferred between the AIF modules and memory by DMA.

SPI MODULE

The SPI control interface block features support for:

- 4-wire SPI protocol up to 26 MHz
- Master and slave mode operation
- Selectable 8, 16, 24, 32 and 64-bit data word transfer

FUSE MODULE

The fuse memory is a small area of non-volatile, one-time programmable (OTP) memory that controls:

- Access to the JTAG port, for security on production devices
- Port selection for program download following reset
- Start-up (default) register settings
- Security configuration

For custom-programmed devices, the fuses are configured during manufacture, according to application-specific requirements.

The WM0011 is also available as an un-programmed device. Note that fuse programming by users is not supported.

DMA MODULE

The DMA module automates the movement of data between memory and key peripherals, or between different memory locations. Features of the DMA controller include:

- 32 independent channels
- DMA requests can be assigned to either a high or a low priority arbitration group, with each group being arbitrated separately
- Low priority arbitration group requests use a master transfer type of either Single or Burst
- Software transfer trigger per channel
- Each DMA channel is configurable for 64-bit, 32-bit, 16-bit or 8-bit transfers
- Programmable transfer length
- DMA chaining capability via Linked List descriptor
- Programmable byte-swapping function
- DMA striding

TRAX TRACE BUFFER MODULE

The Tensilica HiFi EP™ core has a trace capture unit that records the program execution flow to a circular trace buffer. Interrupts, exceptions and branches taken are all recorded in the trace capture file, which can be later used with the OCD module and Tensilica software tools for debugging real-time events or errors.

JTAG MODULE

The WM0011 features an IEEE 1149.1 JTAG Test Access Port (TAP) controller module for chip boundary scanning. The JTAG module also provides access to the On-Chip Debug (OCD) functions for the DSP core. A de-bug server connects to the TAP through a host TAP interface, which is typically an external device such as the USB2Demon™ from Macraigor Systems. All supported JTAG probes are shown on the Tensilica website at http://www.tensilica.com/partners/jtag-probes/. Using the JTAG TAP controller, users can access and control the software-visible state of the processor, including:

- Generate an interrupt to put the processor in the debug mode
- Gain control of the processor upon any debug exception
- Read and write any software-visible register and/or memory location
- Resume normal mode of operation

The JTAG interface can be disabled on custom-programmed devices, to ensure device security. When the JTAG module is disabled, the WM0011 will only execute software code that has been securely authenticated.

The TAP interface consists of five signals listed below.

PIN NAME	DIRECTION	DESCRIPTION
TCK	Input	TAP clock
TMSDEBUG	Input	Input to TAP controller state machine
TMSDFT	Input	JTAG mode select input
$\overline{\text{TRST}}$	Input	Reset input (Active low) for initialisation of the TAP controller
TDI	Input	Selected serial instruction/data shift register input
TDO	Output	Selected serial instruction/data shift register tri-state output

Table 9 IEEE 1149.1 TAP Signals

For more detailed information, refer to the 'Tensilica On-Chip Debugging Guide' (onchip_debug_guide.pdf).

ON-CHIP DEBUG MODULE

The Tensilica HiFi EP core has an On-Chip Debug (OCD) function that is accessed by the JTAG module.

The OCD module may be reset by the JTAG debugger probe by asserting the $\overline{\text{TOCDRST}}$ signal. This signal may also optionally generate a warm reset of the chip.

For further details on the on-chip debug module, please refer to the Tensilica user guide for the on-chip debug, 'onchip_debug_guide.pdf'.

POWER-ON AND RESET CONTROL

The WM0011 incorporates a number of different Reset mechanisms, which are summarised below.

Hardware Reset - this is controlled by the $\overline{\text{RESET}}$ input pin. When the $\overline{\text{RESET}}$ pin is asserted, the chip is held in its reset condition, with all modules disabled and registers set to default. When the $\overline{\text{RESET}}$ pin is de-asserted, the WM0011 will commence the boot sequence.

Warm Reset - this is controlled by the $\overline{\text{TOCDRST}}$ input pin, or by internal functions (Watchdog timeout, PLL Lock status, or the Wake-Up FSM). Each of these triggers can be masked individually. If any of the Warm Reset conditions is asserted (and unmasked), the Warm Reset will reset the core functions and peripheral modules.

Software Reset - this function comprises individual reset control fields for each peripheral module.

POWER ON RESET

There is no Power-On Reset (POR) circuit for initialising the chip on power-up.

It is required that the $\overline{\text{RESET}}$ input pin is asserted (logic '0') during power-up, and must remain asserted until the power supply rails are within recommended operating conditions, and the CLKIN reference is stable.

The WM0011 boot sequence will commence after the $\overline{\text{RESET}}$ pin has been de-asserted. When the WM0011 is ready to commence software/configuration download, the $\overline{\text{IRQ}}$ output pin will be asserted (logic '0').

See "Boot Sequence Control" for details of the WM0011 boot sequence. Note that, on completion of the boot sequence, the $\overline{\text{IRQ}}$ output pin will be de-asserted (logic '1').

Note that, under default start-up conditions, the CLKIN input is selected as the clock source. The Custom fuse settings, and/or PLL Configuration download, can be used to select the start-up clocking configuration for different applications.

The Power-On Reset sequence is illustrated in Figure 8.

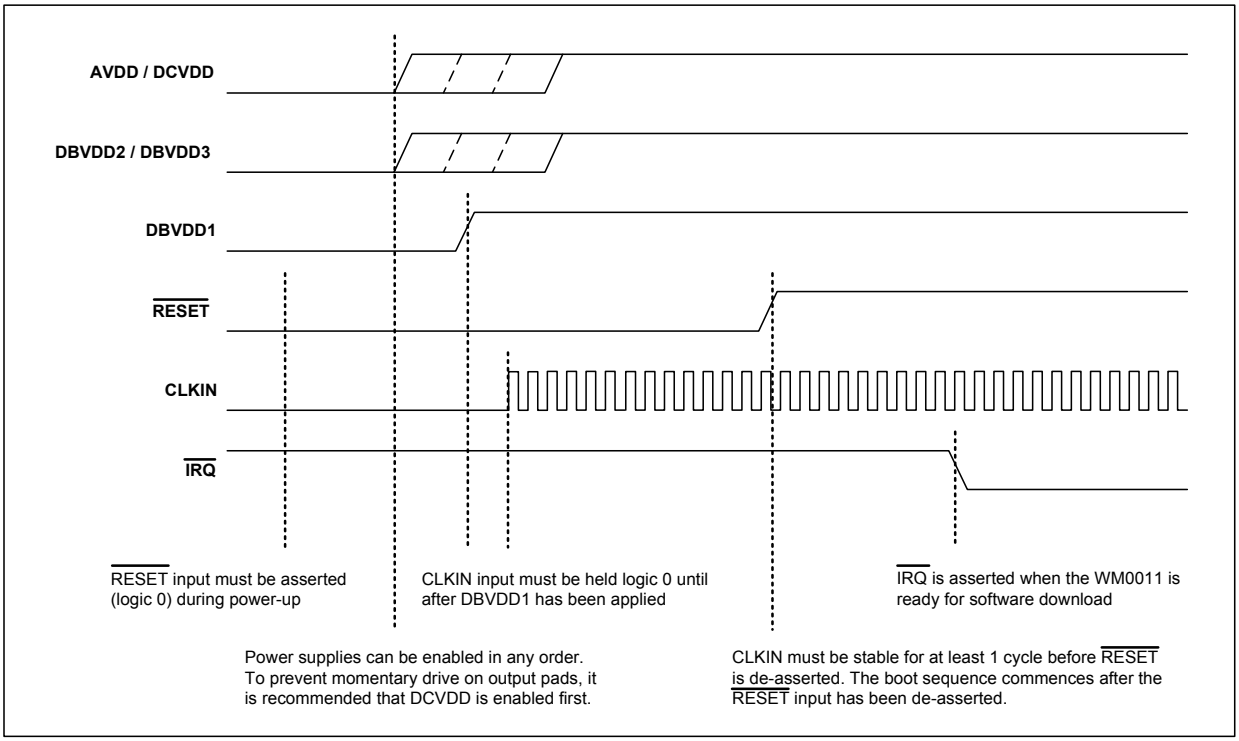


Figure 8 Power-On Reset Sequence

HARDWARE RESET

The Hardware Reset is triggered by asserting the $\overline{\text{RESET}}$ input pin. This pin is an 'active low' input; the Hardware Reset is asserted by applying a logic '0'. The Hardware Reset will reset the core functions and peripheral modules.

The WM0011 boot sequence will commence after the $\overline{\text{RESET}}$ pin has been de-asserted. When the WM0011 is ready to commence software/configuration download, the $\overline{\text{IRQ}}$ output pin will be asserted (logic '0').

See "Boot Sequence Control" for details of the WM0011 boot sequence. Note that, on completion of the boot sequence, the $\overline{\text{IRQ}}$ output pin will be de-asserted (logic '1').

Note that, under default start-up conditions, the CLKIN input is selected as the clock source. The Custom fuse settings, and/or PLL Configuration download, can be used to select the start-up clocking configuration for different applications.

The Hardware Reset sequence is illustrated in Figure 9.

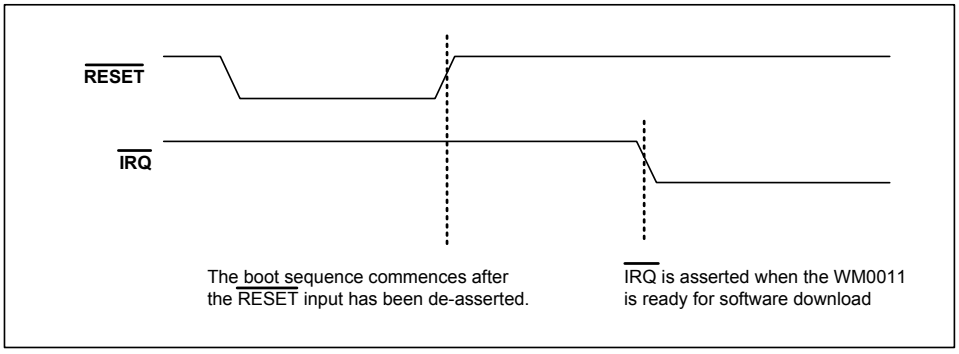


Figure 9 Hardware Reset Sequence

WARM RESET

The conditions that will initiate a Warm Reset are listed below. Each condition can be individually enabled or masked, to control whether a Warm Reset is triggered by the respective condition. The Warm Reset will reset the peripheral modules. Note that the Core Configuration Module (CCM) settings and the RAM contents are not affected by the Warm Reset (except where overwritten by the associated boot sequence).

$\overline{\text{TOCDRST}}$ input - this pin is provided for use as an input from the debug tool. Under default conditions, asserting this pin (logic '0') will trigger a Warm Reset. This can be masked using the OCD_MSK bit.

Watchdog timeout - the Watchdog Timeout condition can trigger a Warm Reset. This is disabled by default, and must be enabled in the Watchdog Timer (WDT) module using the WDT_RST_ENA bit if required. The Warm Reset can be masked within the Chip Configuration module using the WDT_MSK bit.

PLL Lock - the 'out-of-lock' condition in the PLL can trigger a Warm Reset. Under default conditions, the 'out-of-lock' condition will trigger a Warm Reset. This can be masked using the PLL_MSK bit.

Wake-Up condition - the device wake-up is triggered by the FIRQ_N signal from the Interrupt Controller (IRQC) module to the Wake-Up FSM. The Wake-Up event can be enabled as a Warm Reset condition using the WKUP_RST_ENA bit.

When a Warm Reset is triggered as part of a Wake-Up transition, software execution will commence at the code address determined by the STATIC_VECT_SEL register field (see Table 25). If the primary reset vector is selected, then the code execution will be equivalent to a Hardware Reset. The alternate reset vector allows application-specific reset behaviour to be configured. See "Memory Map" for details of the reset vector addresses.

The Warm Reset logic is shown in Figure 10. The illustration includes a number of latching status registers that are associated with the Warm Reset conditions.

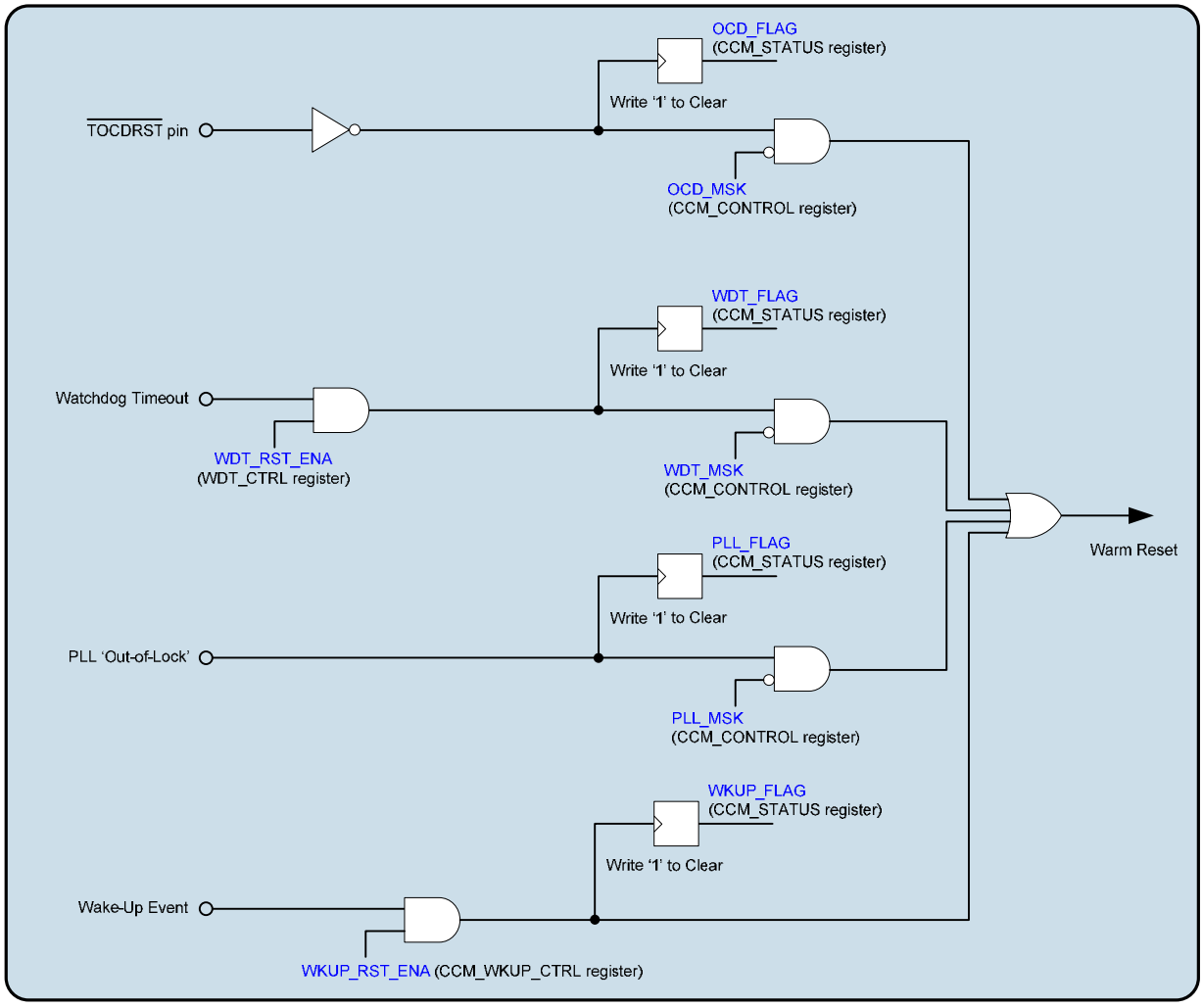


Figure 10 Warm Reset Control

SOFTWARE RESET

The Software Reset function comprises individual reset control fields for each peripheral module. Setting these bits to '0' will reset the respective module to its start-up condition. (These bits must be set to '1' for normal operation of the module.)

The Software Reset control bits are located in the CCM_SOFTRST register (see Table 24).

JTAG RESET

The JTAG interface controller is reset by asserting the $\overline{\text{TRST}}$ input pin. This pin is an 'active low' input; the JTAG Reset is asserted by applying a logic '0'.

Note that the JTAG interface is not affected by the WM0011 Warm Reset or Software Reset functions.

The JTAG interface can be disabled on custom-programmed devices, to ensure device security.

BOOT SEQUENCE CONTROL

Following Power-Up or Hardware Reset, the WM0011 executes the integrated ROM boot code, which starts up the chip from the reset condition. Following a short self-test routine, the $\overline{IRQ}$ pin is asserted (logic 0), indicating the WM0011 is ready to commence software/configuration download.

The boot-up behaviour is configurable using internal, one-time-programmable fuses. The fuse data controls which interface will be used for software/configuration download. The fuses also allow the start-up condition of certain control registers to be configured.

Note that the fuse data capability is supported on custom-programmed devices only. Un-programmed devices do not support these options. Fuse programming by users is not supported.

The software/configuration download is described later in this section. See also “Fuse Memory” for details of the programmable fuses.

As part of the boot sequence, the WM0011 will determine whether the Custom fuses have been programmed.

If the Custom fuses are not programmed, then the WM0011 will await a software/configuration download via the SPI (Slave) port.

If the Custom fuses have been programmed, then the fuse data will select the desired clocking configuration, and also select the desired boot method for software/configuration download. The available download options are SPI Slave, or SPI Master (eg. Flash Memory).

If SPI Master is selected, then the boot download is automatically initiated by the WM0011. If SPI Slave is selected, then the boot download is controlled by an external device. In all cases, the software will automatically execute on completion of a successful code download.

For normal operation, the software/configuration download must include executable code for the WM0011 DSP Core. The download may, optionally, include PLL settings codes for setting the desired clocking configuration.

The supported download actions are described later in this section. The ‘Code Packet’ format, also described, is used in each case.

The device clocking configuration can be selected via Custom fuse data or via PLL Configuration download. Note that, if the Custom fuses have been programmed, then the associated clocking configuration details will be superseded by the PLL Configuration data, if this is subsequently received.

Note that the Custom fuse data and PLL Configuration download include parameters that are held in the WM0011 control registers. The fuse settings and PLL download will determine the start-up values of the corresponding registers, but these can be updated during normal operation later if required.

The $\overline{IRQ}$ pin is asserted (logic 0) shortly after Power-Up or Hardware Reset, indicating the WM0011 is ready to commence software/configuration download. The $\overline{IRQ}$ pin remains asserted until valid application software is fully downloaded; the $\overline{IRQ}$ output is then de-asserted (logic 1).

SOFTWARE / CONFIGURATION DOWNLOAD

The software/configuration download following power-up or hardware reset will comprise one or more of the following operations:

- Software Header download
- Software Data download
- Phase Locked Loop (PLL) configuration

A standard “Code Packet Format” data transfer mode is used in all cases, as described below.

CODE PACKET FORMAT

The Code Packet Format comprises 4 data blocks, as described in Table 10.

NAME	SIZE	DESCRIPTION
CMD	1 byte	Command field, describing the function of the packet
LEN	3 bytes	Length (in bytes) of the DAT portion of this code packet
ADDR	4 bytes	Memory Address associated with the packet
DAT	0 to 8184 bytes	Data words

Table 10 Code Packet Format

The total size of the Code Packet (LEN) is required to be a multiple of 8 bytes.

The ADDR field must also be 64-bit aligned (ie. a multiple of 8 bytes).

All multi-byte data fields in the packet must the formatted in 'little endian' (Least Significant Byte first) format.

One or more code packets may be downloaded to the WM0011 in order to configure the device for the required application. The Code Packet Format is illustrated in Figure 11.

Figure 11 Code Packet Format

CODE HEADER DOWNLOAD

The Software Code download operation requires multiple Code Packets to be sent to the WM0011. The download may be achieved via the SPI Slave or SPI Master (eg. Flash Memory) interfaces. If the Custom fuses are not programmed, then only the SPI (Slave) download method is possible.

The Software Code download comprises one Code Header packet, followed by multiple Code Data packets.

The Code Packet definition for the Code Header is:

- CMD = 0x02
- LEN = 0x00_0108
- ADDR = Start Address for code execution
- DAT = Data words

In the DAT portion of the code packet, the first 32-bit data word will contain the total length (in bytes) of the code image. This is followed by a 32-bit filler word, followed by the 256-byte image signature (SHA-256).

For custom-programmed devices, the WM0011 supports PKA-encryption of the image signature. This is not supported on un-programmed devices.

On receipt of a valid Code Header packet, the WM0011 will expect to receive the associated Code Data packets, thus completing the software code download.

Boot status and error codes are reported via the UART interface, and via the SPI interface during the Code Header download - see "Boot Status and Error Reporting".

CODE DATA DOWNLOAD

The Software Code download comprises one Code Header packet (as described above), followed by multiple Code Data packets.

The Code Packet definition for the Code Data is:

- CMD = 0x03
- LEN = Data Length (in bytes)
- ADDR = Start Address for code data
- DAT = Data words

On completion of the full set of Code Data packet downloads, the $\overline{IRQ}$ output is de-asserted and the WM0011 will commence execution of the downloaded software.

Note that completion of the Code Data packets is determined by the code image length that is contained within the Code Header packet (DAT). Software execution commences at the start address (ADDR) - also contained in the Code Header packet.

Boot status and error codes are reported via the UART interface, and via the SPI interface during the Code Data download - see "Boot Status and Error Reporting".

PLL CONFIGURATION DOWNLOAD

The PLL Configuration download operation requires a single Code Packet to be sent to the WM0011. The download may be achieved via the SPI Slave or SPI Master (eg. Flash Memory) interfaces. If the Custom fuses are not programmed, then only the SPI (Slave) download method is possible.

The Code Packet definition for PLL Configuration download is:

- CMD = 0x04
- LEN = 0x00_0018
- ADDR = 0x0000_0000
- DAT = PLL Configuration Data

The "DAT" portion must comprise 24 bytes, corresponding to the intended contents of the clocking configuration registers listed below. The CCM_CLK_CTRL1 register is transmitted first.

- CCM_CLK_CTRL1 (4 bytes, see Table 19)
- CCM_CLK_CTRL2 (4 bytes, see Table 20)
- CCM_CLK_CTRL3 (4 bytes, see Table 21)
- CCM_PLL_LOCK_CTRL (4 bytes, see Table 22)
- UART_BAUD_LSW (1 byte, see Table 118)
- UART_BAUD_MSW (1 byte, see Table 119)
- Padding (2 bytes)
- SPI_SCLKDIV (4 bytes, see Table 123)

On receipt of a valid PLL Configuration packet, the control registers noted above will be updated with the received data, and the new clocking configuration will become effective.

Note that the SPI_SCLKDIV register on the WM0011 is only updated if the selected boot method is SPI Master. In all other cases, the SPI_SCLKDIV portion of the PLL download is ignored and discarded. Note that the SPI Master boot method is only possible via the Custom fuse data settings.

Boot status and error codes are reported via the UART interface, and via the SPI interface during the PLL Configuration download - see "Boot Status and Error Reporting".

BOOT STATUS AND ERROR REPORTING

During boot-up, the WM0011 generates status and error codes for external monitoring of the start-up process. These status codes are reported via the UART interface, and also via the SPI interface (in SPI Slave mode only).

The SPI output comprises a 32-bit code for each status code. The status reporting on the SPI interface is only supported in SPI Slave mode. A maximum of one status/error code can be reported per Code Packet received. The applicable code will be transmitted for the duration of the next Code Packet that follows after the Code Packet to which the status/error code relates. Accordingly, it should be noted that some codes may be applicable but are not transmitted on the SPI interface.

The UART output is in the form of a single ASCII character code for each condition. The applicable code(s) are reported immediately after receipt of the Code Packet to which they relate.

The SPI interface can report all of the defined status/error codes; the UART interface only supports a reduced set of codes, as noted in Table 11.

The UART data output format is: 8 data bits, stop bit, no parity. If the Custom fuses are not programmed, then the assumed clock rate (24.5MHz) gives 115,200bps data output. Other clock frequencies and data bit rates are possible using the Custom fuse or PLL Configuration options.

The boot status and error report codes are defined in Table 11.

MESSAGE NAME	UART (ASCII)	SPI SLAVE	DESCRIPTION
ROM_DBG_CODE_START		0x0FED0000	The C startup code has finished and ROM application starting
ROM_DBG_FUSE_CLR		0x0FED0001	The contents of the fuse array are entirely blank
ROM_DBG_CUST_FUSE_CLR		0x0FED0002	The custom portion of the fuse array is blank
ROM_DBG_COMM_ENABLED	A	0x0FED0003	Communication port enabled and ROM ready for image download
ROM_DBG_FUSE_DL_SUCCESS	B	0x0FED0004	Fuse Data received successfully.
ROM_DBG_FUSE_PROGRAMMED	C	0x0FED0005	Fuses are programmed, rebooting immediately.
ROM_DBG_GOOD_PKT	D	0x0FED0006	A valid data packet received
ROM_DBG_CODE_HDR_VALID	E	0x0FED0007	Valid Code Header Packet received
ROM_DBG_CODE_PKT_VALID	F	0x0FED0008	Valid Code Data packet received
ROM_DBG_CODE_DL_COMPLETE	G	0x0FED0009	An entire code image has been downloaded
ROM_DBG_CODE_SECURE_MATCH	H	0x0FED000A	The decrypted image header matches the SHA result
ROM_DBG_CODE_UNSECURE_MATCH	I	0x0FED000B	The raw image header matches the SHA result, with JTAG enabled
ROM_DBG_APP_START	J	0x0FED000C	This is the final message sent by the ROM prior to starting the User Application.
ROM_DBG_WAITING_PLL_LOCK	K	0x0FED000D	This is sent as soon as the PLL is enabled while waiting for LOCK to be asserted
ROM_DBG_PLL_PACKET_SUCCESS	L	0x0FED000E	The PLL packet was received and clocks have been successfully changed
ROM_DBG_FUSE_INVALID	a	0x0FED0023	A CRC mismatch in the custom fuses detected
ROM_DBG_FUSE_DL_FAIL_BAD_LEN	b	0x0FED0024	The Custom Fuse image length is incorrect.
ROM_DBG_IMG_TOO_LONG	c	0x0FED0025	Final Data Packet exceeds total length specified in Header Packet
ROM_DBG_PKT_OVERFLOW	d	0x0FED0026	Data packet received prior to previous packet finished processing possible overflow
ROM_DBG_CODE_UNSECURE_MISMATCH	e	0x0FED0027	The raw image header does NOT match the SHA result w/ JTAG enabled

MESSAGE NAME	UART (ASCII)	SPI SLAVE	DESCRIPTION
ROM_DBG_CODE_SECURE_MISMATCH	f	0x0FED0028	The decrypted image header does NOT match the SHA result
ROM_DBG_CODE_ILLEGAL_DOWNLOAD	g	0x0FED0029	The USER has attempted a Code Download when fuses are blank and JTAG is disabled
ROM_DBG_CODE_DL_FAIL	h	0x0FED002A	The code download has failed and will restart
ROM_DBG_BAD_SPI_FUSE_PKT	i	0x0FED002B	Invalid packet received when JTAG fuse is blank
ROM_DBG_BAD_SPI_PKT	j	0x0FED002C	SPI XFR length does NOT match packet length
N/A	k	0x0FED002D	Not Currently Used
N/A	l	0x0FED002E	Not Currently Used
N/A	m	0x0FED002F	Not Currently Used
N/A	n	0x0FED0030	Not Currently Used
N/A	o	0x0FED0031	Not Currently Used
ROM_DBG_SPI_ERR_RDOFL	p	0x0FED0032	SPI Read Overflow error reported by IP Packet is discarded
ROM_DBG_SPI_ERR_UCLK	q	0x0FED0033	SPI Underclock error is reported by IP, Packet is discarded
ROM_DBG_BAD_HDR_PKT	r	0x0FED0034	Header Packet does not contain correct # of bytes.
ROM_DBG_INVALID_PKT_TYPE	s	0x0FED0035	An unsupported packet type is received.
ROM_DBG_DATA_BEFORE_HDR	t	0x0FED0036	A Data packet is received without first receiving the Header packet
ROM_DBG_FUSE_DL_FAIL_FUSES_PROGD	u	0x0FED0037	A Custom Fuse image is received when fuses are already programmed.
ROM_DBG_FUSE_DL_FAIL_BAD_CRC	v	0x0FED0038	Computed CRC in downloaded Fuse Packet is incorrect.
ROM_DBG_INVALID_PLL_PKT	w	0x0FED0039	PLL packet with incorrect data length received.
ROM_DBG_CLEARING_PLL_OVERRIDE	x	0x0FED003A	PLL Unlock caused warm reset clearing error before switching to PLL clock source
ROM_DBG_DATA_PACKET_ALIGN_ERR	y	0x0FED003B	Code Download packet has a length or address that is not double word aligned

Table 11 Boot Status and Error Reporting

BOOT SEQUENCE FLOW DIAGRAMS

Figure 12, Figure 13 and Figure 14 illustrate the top level boot flow (Figure 12), boot packet processing (Figure 13), and application validation (Figure 14).

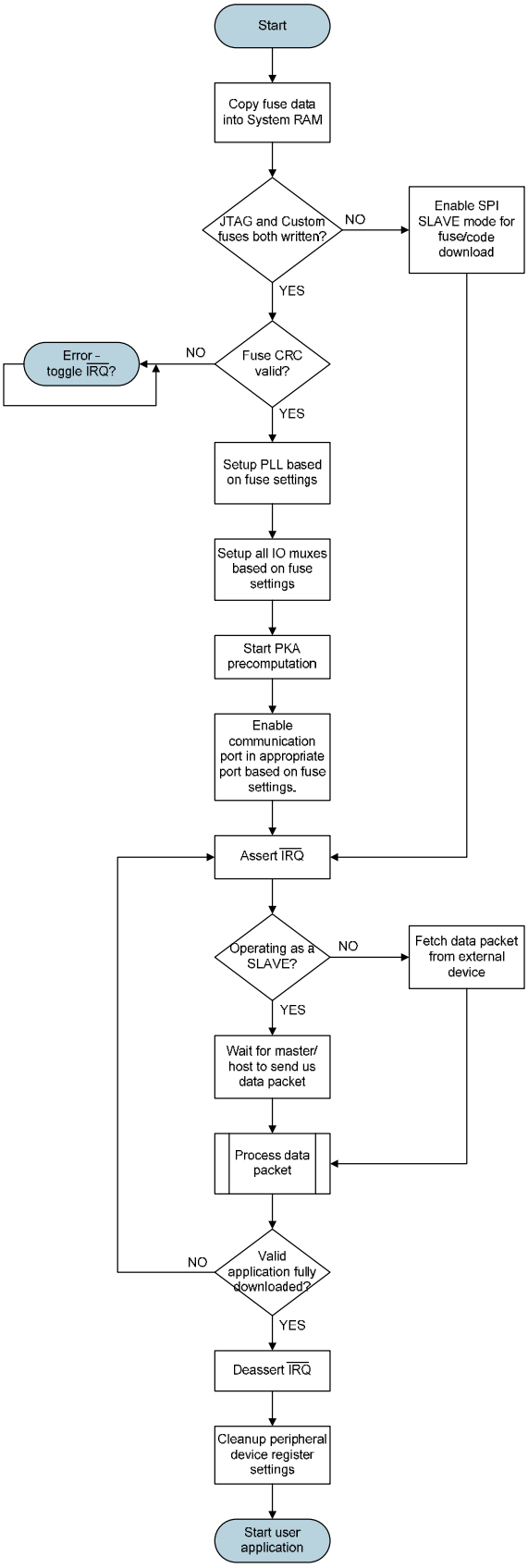


Figure 12 Top Level Boot Flow

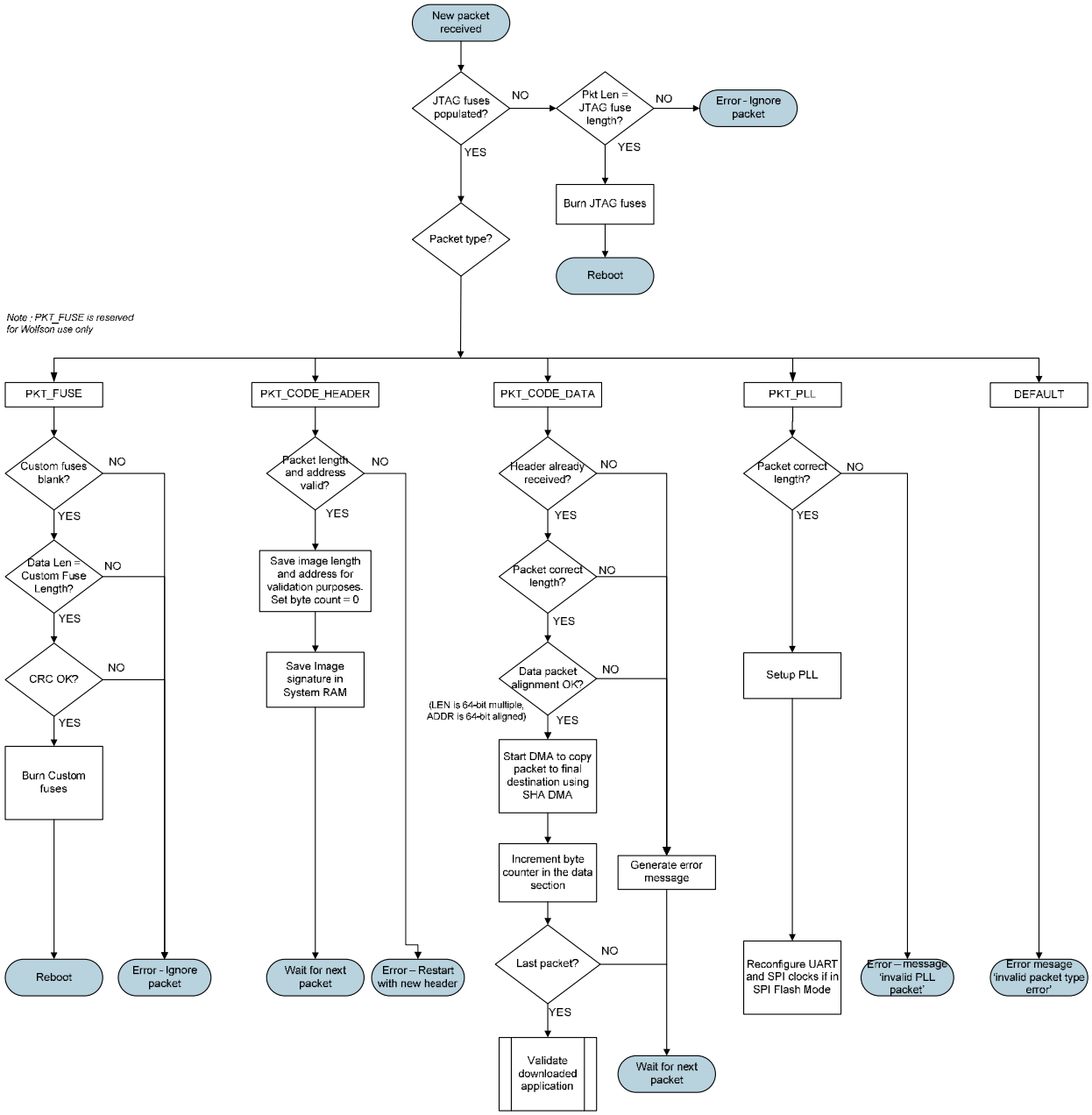


Figure 13 Boot Packet Processing

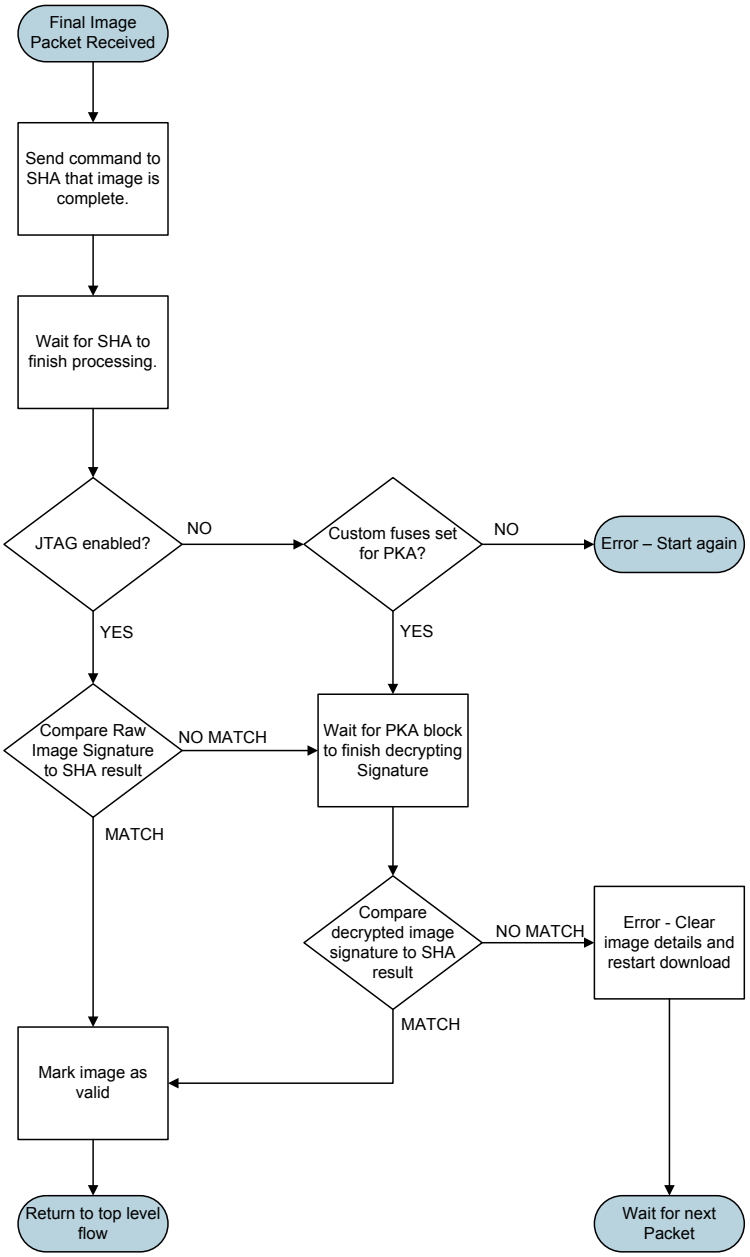


Figure 14 Application Validation

INTERRUPTS

There are a number of different Interrupt levels on the WM0011. The overall Interrupt scheme is illustrated in Figure 15.

GPIO pins that are configured as inputs are handled by a dedicated GPIO circuit. This provides readback of the GPIO status, and configurable edge/level detection, giving rise to a single GPIO_INT interrupt. See "General Purpose Input/Output (GPIO) Module" for further details.

Most of the peripheral modules generate one interrupt each, feeding into the WM0011 Interrupt module. The GPIO_INT signal described above is one such input. The $\overline{\text{STANDBY}}$ pin also provides input directly to the Interrupt module. See "Interrupt Controller (IRQC) Module" for further details.

The HiFi2 EP™ DSP core has its own Interrupt functionality also. The inputs to the HiFi2 EP™ DSP core comprise the IRQ_N and FIRQ_N outputs from the WM0011 Interrupt module, combined with direct inputs from most of the peripheral modules. Two GPIO inputs may be selected (via multiplexers) as HiFi2 EP™ interrupts. The $\overline{\text{STANDBY}}$ pin and a number of HiFi2 EP™ internal signals make up the remaining inputs.

The $\overline{\text{IRQ}}$ pin output is controlled by the WM0011 Interrupt module.

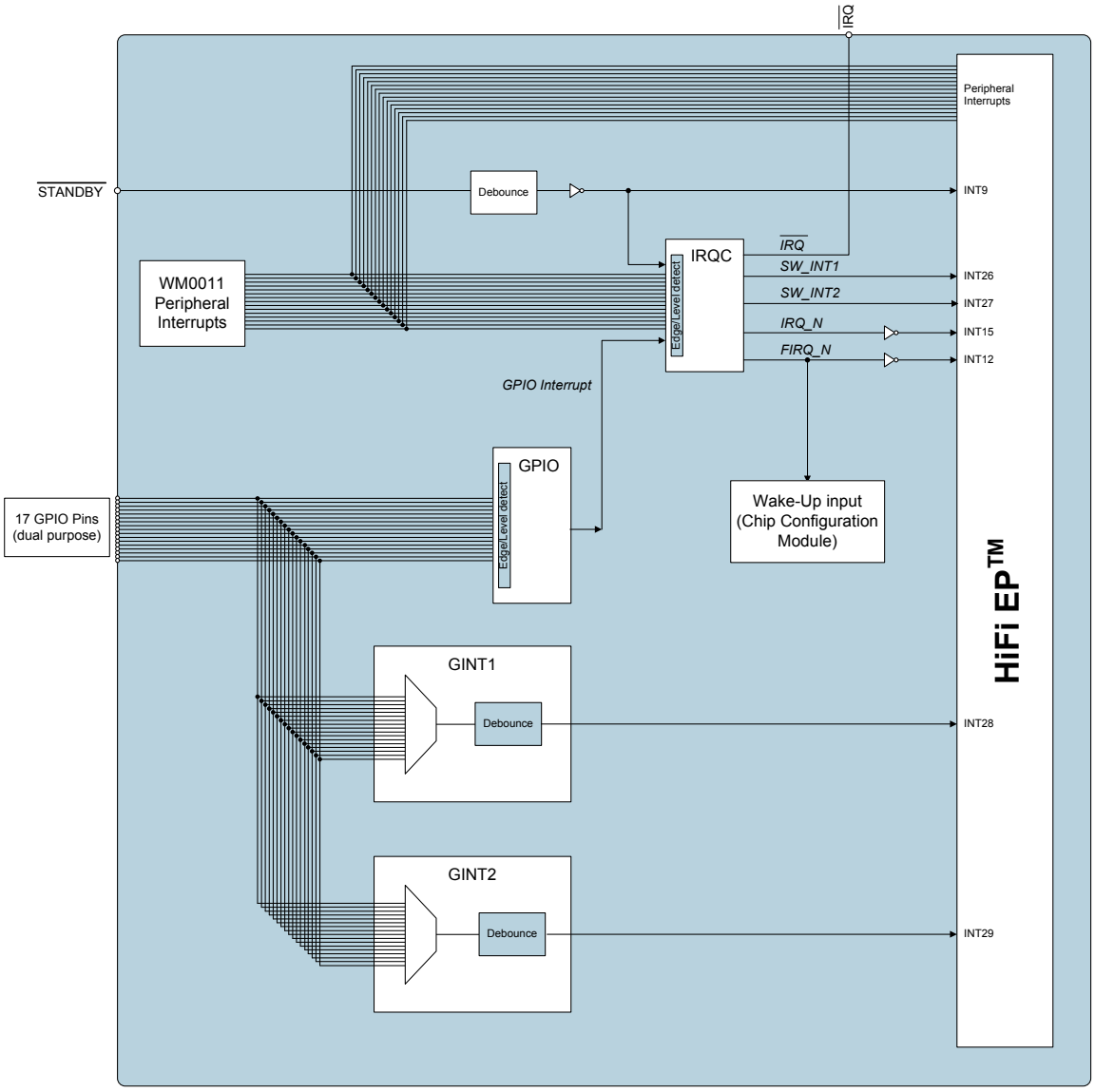


Figure 15 External and Internal Interrupts to HiFi EP™

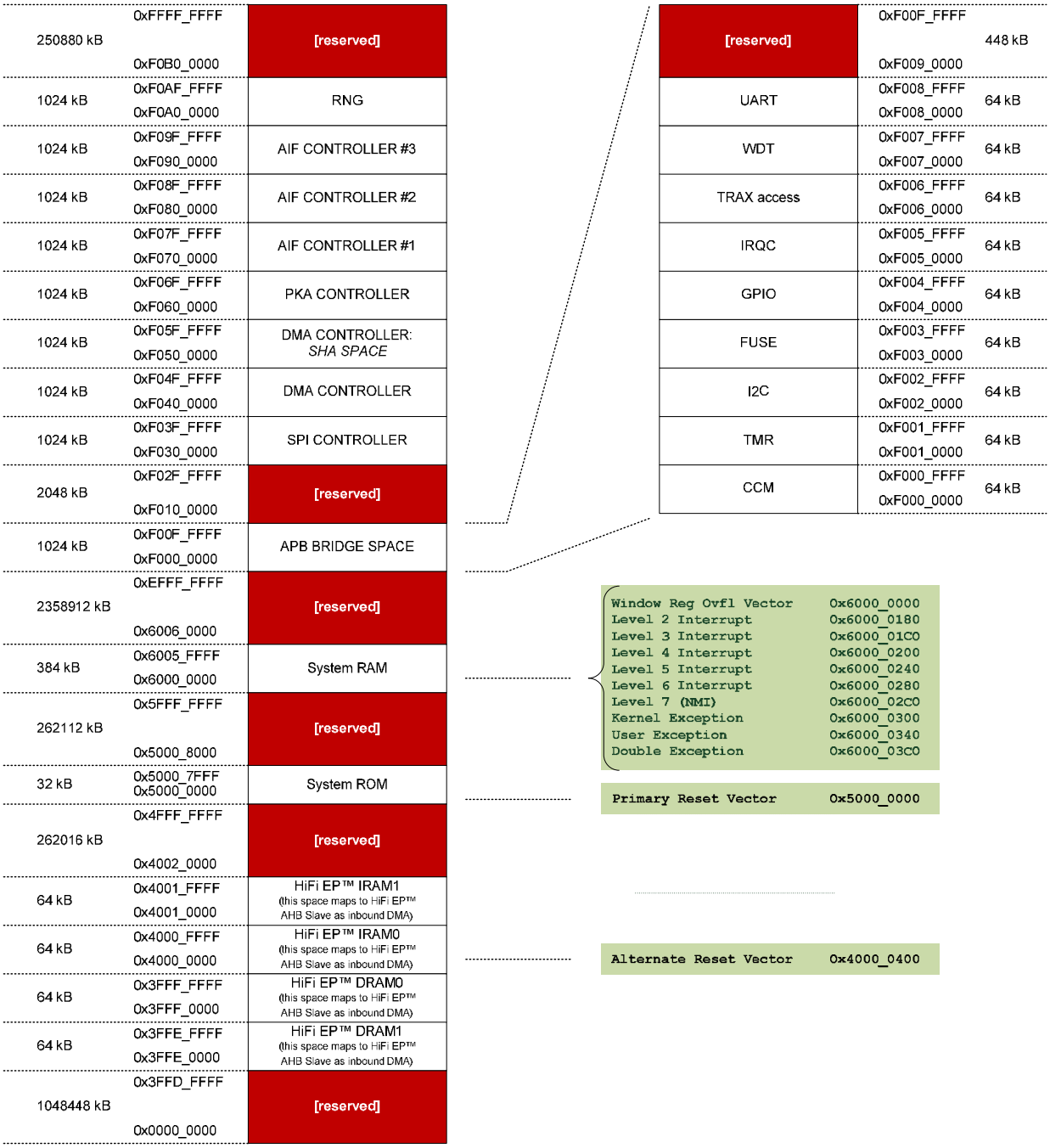
The HiFi EP™ DSP core interrupts are described in Table 12.

All of these interrupts are Active High at the input to the HiFi EP™ DSP core.

HIFI EP™ DSP INTERRUPT (TYPE/PRIORITY)	DESCRIPTION	SOURCE
Int0 (Level/1)	SPI interrupt	SPI controller
Int1 (Level/1)	UART interrupt	UART controller
Int2 (Level/2)	Reserved	
Int3 (Level/1)	WDT interrupt	Watchdog Timer
Int4 (Level/1)	I2C interrupt	I2C controller
Int5 (Level/1)	PKA interrupt	PKA controller
Int6 (TMR/1)	HiFi EP™ Timer0	Internal to HiFi EP™ core
Int7 (Software/1)	HiFi EP™ Software	Internal to HiFi EP™ core
Int8 (Level/2)	Reserved	
Int9 (Level/3)	STANDBY input pin	$\overline{\text{STANDBY}}$ input pin
Int10 (Timer/3)	HiFi EP™ Timer1	Internal to HiFi EP™ core
Int11 (Software/3)	HiFi EP™ Software	Internal to HiFi EP™ core
Int12 (Level/4)	FIRQ_N interrupt	IRQC module
Int13 (TMR/5)	HiFi EP™ Timer2	Internal to HiFi EP™ core
Int14 (NMI/7)	Non-Maskable Interrupt	
Int15 (Level/1)	IRQ_N interrupt	IRQC module
Int16 (Level/1)	AIF 1 interrupt	AIF controller #1
Int17 (Level/1)	AIF 2 interrupt	AIF controller #2
Int18 (Level/1)	AIF 3 interrupt	AIF controller #3
Int19 (Level/1)	DMA interrupt	DMA controller
Int20 (Level/1)	Reserved	
Int21 (Level/1)	TMR 1 interrupt	TIMER 1 module
Int22 (Level/1)	TMR 2 interrupt	TIMER 2 module
Int23 (Level/1)	TMR 3 interrupt	TIMER 3 module
Int24 (Level/1)	Reserved	
Int25 (Level/1)	Reserved	
Int26 (Level/1)	Software interrupt 15	IRQC module
Int27 (Level/1)	Software interrupt 14	IRQC module
Int28 (Level/1)	GINT1	GPIO pin (selected using GINT1_SEL - CCM_CONTROL register)
Int29 (Level/1)	GINT2	GPIO pin (selected using GINT2_SEL - CCM_CONTROL register)
Int30 (Level/1)	Reserved	
Int31 (WriteErr)	AHB bus error	Internal to HiFi EP™ core

Table 12 DSP Core Interrupts

MEMORY MAP


0xFFFF_FFFF
250880 kB
0xF0B0_0000
[reserved]
0xF0AF_FFFF
1024 kB
0xF0A0_0000
RNG
0xF09F_FFFF
1024 kB
0xF090_0000
AIF CONTROLLER #3
0xF08F_FFFF
1024 kB
0xF080_0000
AIF CONTROLLER #2
0xF07F_FFFF
1024 kB
0xF070_0000
AIF CONTROLLER #1
0xF06F_FFFF
1024 kB
0xF060_0000
PKA CONTROLLER
0xF05F_FFFF
1024 kB
0xF050_0000
DMA CONTROLLER: SHA SPACE
0xF04F_FFFF
1024 kB
0xF040_0000
DMA CONTROLLER
0xF03F_FFFF
1024 kB
0xF030_0000
SPI CONTROLLER
0xF02F_FFFF
2048 kB
0xF010_0000
[reserved]
0xF00F_FFFF
1024 kB
0xF000_0000
APB BRIDGE SPACE
0xEFFF_FFFF
2358912 kB
0x6006_0000
[reserved]
0x6005_FFFF
384 kB
0x6000_0000
System RAM
0x5FFF_FFFF
262112 kB
0x5000_8000
[reserved]
0x5000_7FFF
32 kB
0x5000_0000
System ROM
0x4FFF_FFFF
262016 kB
0x4002_0000
[reserved]
0x4001_FFFF
64 kB
0x4001_0000
HiFi EP™ IRAM1
(this space maps to HiFi EP™ AHB Slave as inbound DMA)
0x4000_FFFF
64 kB
0x4000_0000
HiFi EP™ IRAM0
(this space maps to HiFi EP™ AHB Slave as inbound DMA)
0x3FFF_FFFF
64 kB
0x3FFF_0000
HiFi EP™ DRAM0
(this space maps to HiFi EP™ AHB Slave as inbound DMA)
0x3FFE_FFFF
64 kB
0x3FFE_0000
HiFi EP™ DRAM1
(this space maps to HiFi EP™ AHB Slave as inbound DMA)
0x3FFD_FFFF
1048448 kB
0x0000_0000
[reserved]
[reserved]
0xF00F_FFFF
448 kB
0xF009_0000
UART
0xF008_FFFF
64 kB
0xF008_0000
WDT
0xF007_FFFF
64 kB
0xF007_0000
TRAX access
0xF006_FFFF
64 kB
0xF006_0000
IRQC
0xF005_FFFF
64 kB
0xF005_0000
GPIO
0xF004_FFFF
64 kB
0xF004_0000
FUSE
0xF003_FFFF
64 kB
0xF003_0000
I2C
0xF002_FFFF
64 kB
0xF002_0000
TMR
0xF001_FFFF
64 kB
0xF001_0000
CCM
0xF000_FFFF
64 kB
0xF000_0000
Window Reg Ovfl Vector 0x6000_0000
Level 2 Interrupt 0x6000_0180
Level 3 Interrupt 0x6000_01C0
Level 4 Interrupt 0x6000_0200
Level 5 Interrupt 0x6000_0240
Level 6 Interrupt 0x6000_0280
Level 7 (NMI) 0x6000_02C0
Kernel Exception 0x6000_0300
User Exception 0x6000_0340
Double Exception 0x6000_03C0
Primary Reset Vector 0x5000_0000
Alternate Reset Vector 0x4000_0400

CLOCKING

The WM0011 requires a clock reference for its internal functions, and to provide clocking for external interfaces when Master mode is selected on the respective module(s).

The external clock reference is connected via the XTI pin; this may be either a digital logic input, or may be provided using an external crystal. A two-stage PLL is provided, allowing a high frequency internal clock to be generated from the XTI clock input reference.

The clocking architecture is illustrated in Figure 16. The CLKIN reference (direct from the XTI pin) can provide clocking to all modules directly, and is also used as the input clock to the PLL. An alternate clock (ALTCLK) can also be configured using a GPIO pin as input.

The clock source for most of the WM0011 functions is selected using the CLK_SEL multiplexer; this provides a glitch-free switchover between the CLKIN, PLLOUT or ALTCLK signals. Note that, if a Warm Reset is triggered due to the PLL 'out-of-lock' condition, then the CLK_SEL multiplexer forces the selection of CLKIN as the system clock source. This override must be cleared before any other clock source can be selected.

The clock reference selected by CLK_SEL is processed by configurable dividers to generate the following system clocks:

- DSPCLK - clock reference for the HiFi2 EP™ DSP core
- AHBCLK - clock reference for selected peripherals
- APBCLK - clock reference for selected peripherals

The main clocking options are summarised as follows:

- Under initial start-up conditions, CLKIN is selected as the clock source.
- High-speed clocking is possible when the PLL is configured, and PLLOUT is selected as the clock source.
- The alternative clock source, ALTCLK provides the option of a low-speed clocking configuration; this could be used for a low-power operating mode, or if CLKIN was unsuitable or unavailable.

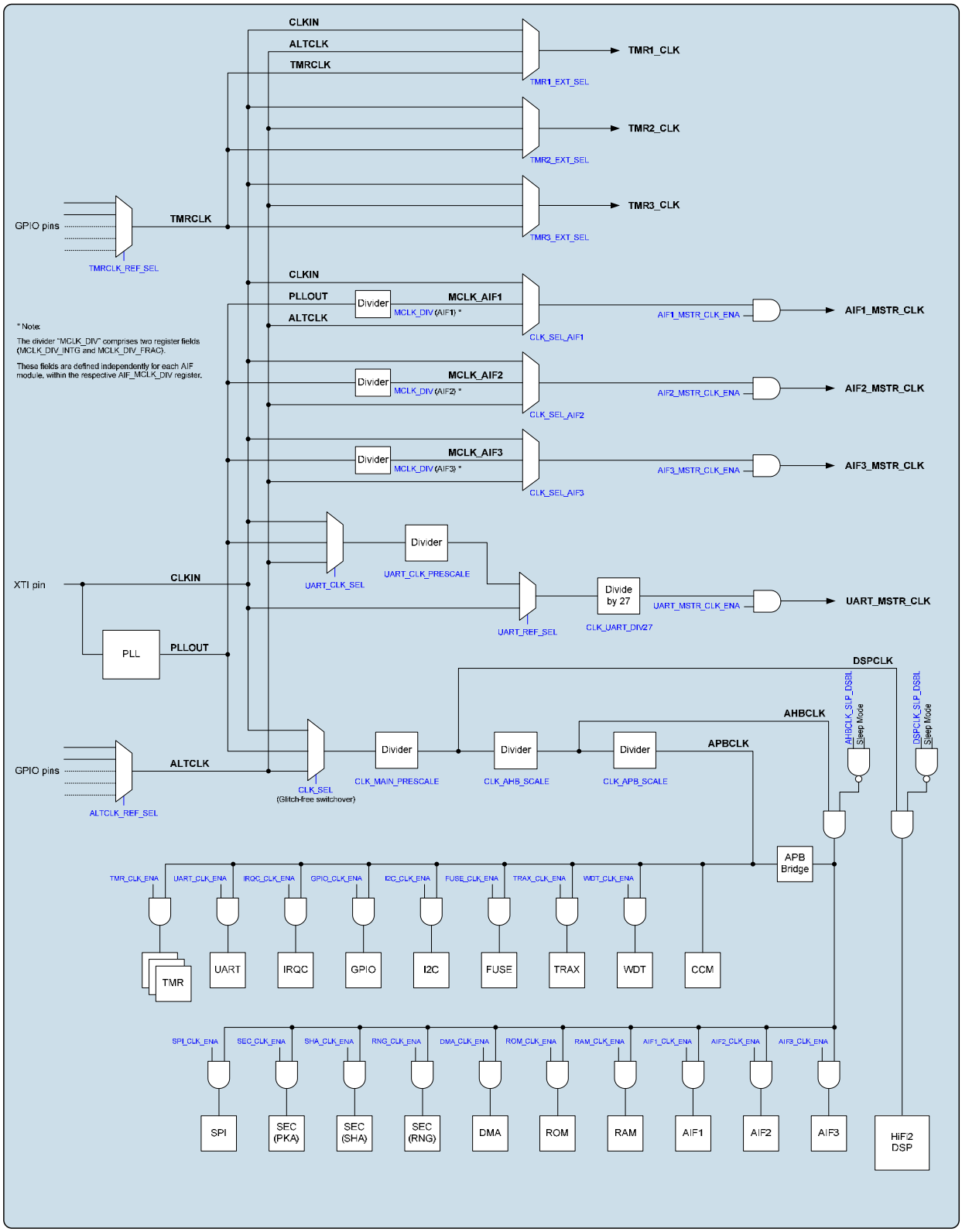


Figure 16 Clocking Architecture

CRYSTAL OSCILLATOR

The external clock reference connected to the XTI pin may be either a digital logic input, or may be provided using an external crystal. The typical connection details for an external crystal are illustrated in Figure 17.

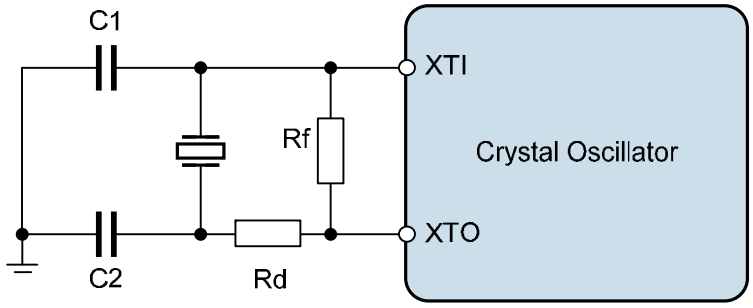


Figure 17 Crystal Oscillator External Components

Selection of the correct external components for the crystal oscillator is important. Recommended guidelines are provided below. Users should also refer to the crystal component datasheet for applicable guidelines.

The feedback resistor (Rf) biases the internal inverter in the high gain region. A typical resistance of 1MΩ is recommended.

The damping resistor (Rd) increases stability, and reduces power consumption, suppressing the high frequency gain. Note that, if this resistance is too large, the loop could fail to oscillate. Some circuits may omit the Rd resistor altogether.

The load capacitors C1 and C2 should be selected according to the recommended load capacitance, C_L of the crystal, which is given by the following equation:

$$\text{Load Capacitance } C_L = \frac{C1 \times C2}{C1 + C2} + C_{STRAY}$$

Assuming C1 = C2 and C_{STRAY} = 2.75pF (typical pad i/o capacitance), then:

$$C1 = C2 = 2 \times (C_L - 2.75pF).$$

For example, if the crystal has a recommended load capacitance C_L = 9pF, then C1 = C2 = 12.5pF.

Table 13 shows the recommended load capacitance and maximum ESR values for a range of suitable WM0011 clocking frequencies.

FREQUENCY	CAPACITANCE (C_L)	MAXIMUM ESR
2MHz to 6MHz	20pF	1000Ω
6MHz to 10MHz	16pF	160Ω
10MHz to 20 MHz	12pF	90Ω
20MHz to 30MHz	8pF	40Ω

Table 13 Crystal Selection Guide

PHASE LOCKED LOOP (PLL)

The WM0011 incorporates a 2-stage Phase Locked Loop (PLL), which can generate the internal high-speed clock reference for the DSP core and other peripheral modules.

The PLL input reference is derived from CLKIN, which may be either a digital logic input, or crystal-generated, as described earlier.

Each PLL can be configured independently. The PLL is reset using the PLLn_RST bits; the PLL is bypassed using the PLLn_BYPASS bits, where 'n' is 1 or 2 for the respective PLL. Note that, if only a single-stage PLL is required, then PLL1 should be bypassed, and PLL2 used.

The PLL loop filter is configured using the PLLn_FRANGE_MSK register; this should be set according to the reference frequency, F_{REF}, of the respective PLL. (Note that the reference frequency is the input frequency, after division by the PLLn_INDIV register setting.)

The frequency conversion ratio of the PLL is configured using PLLn_FRATIO. A divider is provided in the input path and output path of each PLL; these are adjusted using the PLLn_INDIV and PLLn_OUTDIV registers.

The PLL configuration registers are illustrated in Figure 18. The frequency limits for F_{REF} and F_{VCO} are also noted. The two PLLs are cascaded in series; the same frequency limits apply in each case.

The PLLs should be disabled whenever changes are made to the PLL configuration registers. Note that a valid system clock must be maintained when disabling the PLLs; the system clock multiplexer (CLK_SEL) must select a valid clock source (CLKIN or ALTCLK) before disabling the PLLs.

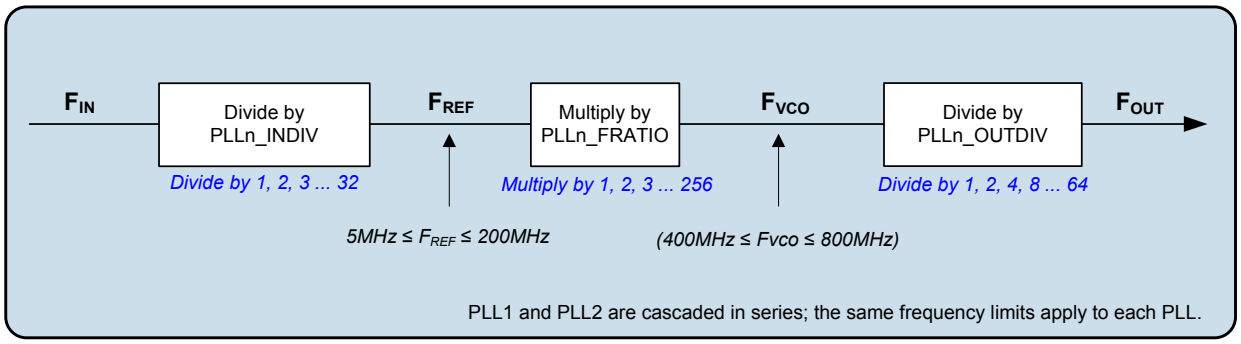


Figure 18 PLL Configuration

The PLL Lock status can be read from the PLL_RAW_LOCK register bit in the CCM_STATUS register (see Table 17). It is recommended that the PLL output is not selected as the clock source until PLL_RAW_LOCK indicates that PLL Lock has been achieved.

A configurable PLL 'out-of-lock' detection circuit is also provided; this is enabled and configured using the PLL_LOCKDET_ENA and PLL_LOCKDET_MODE registers, as described in Table 22. It is recommended that this function is not enabled until PLL_RAW_LOCK indicates that PLL Lock has been achieved.

The PLL lock detection is derived by checking the ratio of the PLL2 output frequency with respect to the PLL1 input frequency; a count is maintained of instances when the ratio is outside the limits set by PLL_LOCKDET_MIN and PLL_LOCKDET_MAX.

When setting the PLL_LOCKDET_MIN and PLL_LOCKDET_MAX thresholds, it should be noted that the input and output clock counters are not synchronised; an error margin should be incorporated into the thresholds to avoid incorrect triggering of the out-of-lock detection.

If the count of the number of frequency ratio exceptions exceeds the thresholds set by PLL_UNDERFLOW_LIMIT or PLL_OVERFLOW_LIMIT, then the PLL 'out-of-lock' condition is asserted.

The PLL 'out-of-lock' condition is indicated via the PLL_FLAG and PLL_UNLOCK register bits in the CCM_STATUS register (see Table 17).

The PLL 'out-of-lock' condition can trigger a Warm Reset, as described in the "Power-on and Reset Control" section. This is selectable using the PLL_MSK bit.

If a Warm Reset is triggered, due to the PLL 'out-of-lock' condition, then the CLK_SEL multiplexer is overridden to force the selection of CLKIN as the system clock source. This override condition is indicated via the PLL_OVERRIDE_FLAG in the CCM_STATUS register. The PLL_OVERRIDE_FLAG must be cleared before any other clock source can be selected.

The 2-stage PLL configuration is illustrated in Figure 19. Example PLL settings for typical use cases are described in Table 14.

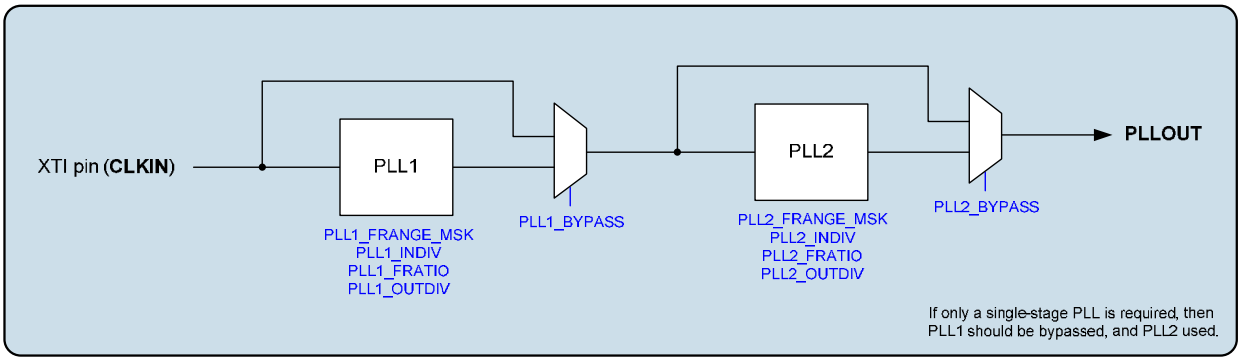


Figure 19 2-stage Cascade PLL Architecture

	PLL1 CONFIGURATION					PLL2 CONFIGURATION				
PLL1 INPUT (MHz)	**PLL1_ FRANGE _MSK**	**PLL1_ INDIV**	**PLL1_ FRATIO**	**PLL1_ OUTDIV**	**PLL1 OUTPUT (MHz)**	**PLL2_ FRANGE _MSK**	**PLL2_ INDIV**	**PLL2_ FRATIO**	**PLL2_ OUTDIV**	**PLL2 OUTPUT (MHz)**
6.144	(PLL1 bypass)				6.144	1h	00h	53h	01h	258.048
12.288	(PLL1 bypass)				12.288	2h	00h	29h	01h	258.048
24.576	(PLL1 bypass)				24.576	2h	01h	29h	01h	258.048
5.6448	(PLL1 bypass)				5.6448	1h	00h	5Bh	01h	259.6608
11.2896	(PLL1 bypass)				11.2896	2h	00h	2Dh	01h	259.6608
22.5792	(PLL1 bypass)				22.5792	2h	01h	2Dh	01h	259.6608
26	(PLL1 bypass)				26	3h	00h	13h	01h	260
19.2	(PLL1 bypass)				19.2	1h	01h	35h	01h	259.2
26	3h	03h	41h	01h	214.5	2h	12h	2Dh	01h	259.6579
Note that the values shown for PLLn_FRANGE_MSK, PLLn_INDIV, PLLn_FRATIO and PLLn_OUTDIV are the register values. See Table 20 and Table 21 for the coding of these registers.										

Table 14 Example PLL Configurations

CORE DEVICE PERIPHERALS

The following sections describe each of the peripheral modules in turn. Each section comprises a descriptive overview, and the detailed definition of the associated control registers.

Note that the following definitions apply for the “S/W Access” data relating to the control register fields:

- RO: Read-Only register bit. Writes to these bits have no effect.
- WO: Write-Only register bit. The read value has no meaning.
- RW: Read/Write register bit.
- R/W1C: Read / Write 1 to Clear bit. Supports Read and Write operations. Writing a ‘1’ clears the bit; Writing a ‘0’ has no effect.
- R/WC: Read / Write to Clear bit. Supports Read and Write operations. Writing any value clears the bit.
- RC: Read to Clear bit. The bit is cleared (set to 0) when it is Read.

CCM - CHIP CONFIGURATION MODULE

BASE ADDRESS 0xF000_0000

CCM FEATURES

This Chip Configuration Module section covers the internal chip configuration, core peripherals, and low power modes of operation.

This 32-bit APB slave contains user-programmable control registers to gate various peripheral clocks, force various peripheral resets, control power management, and control other miscellaneous functions.

The CCM implements the following functions:

- Clocking control/enable registers, and clock generation
- Reset control/enable registers, and Reset generation
- Main control and status registers
- GPIO / $\overline{\text{STANDBY}}$ de-bounce
- I/O buffer control registers (programmable drive strength, pull enables, etc.)
- Scratchpad registers
- Sleep/Wake-up control registers and Wake-up state machine (FSM)

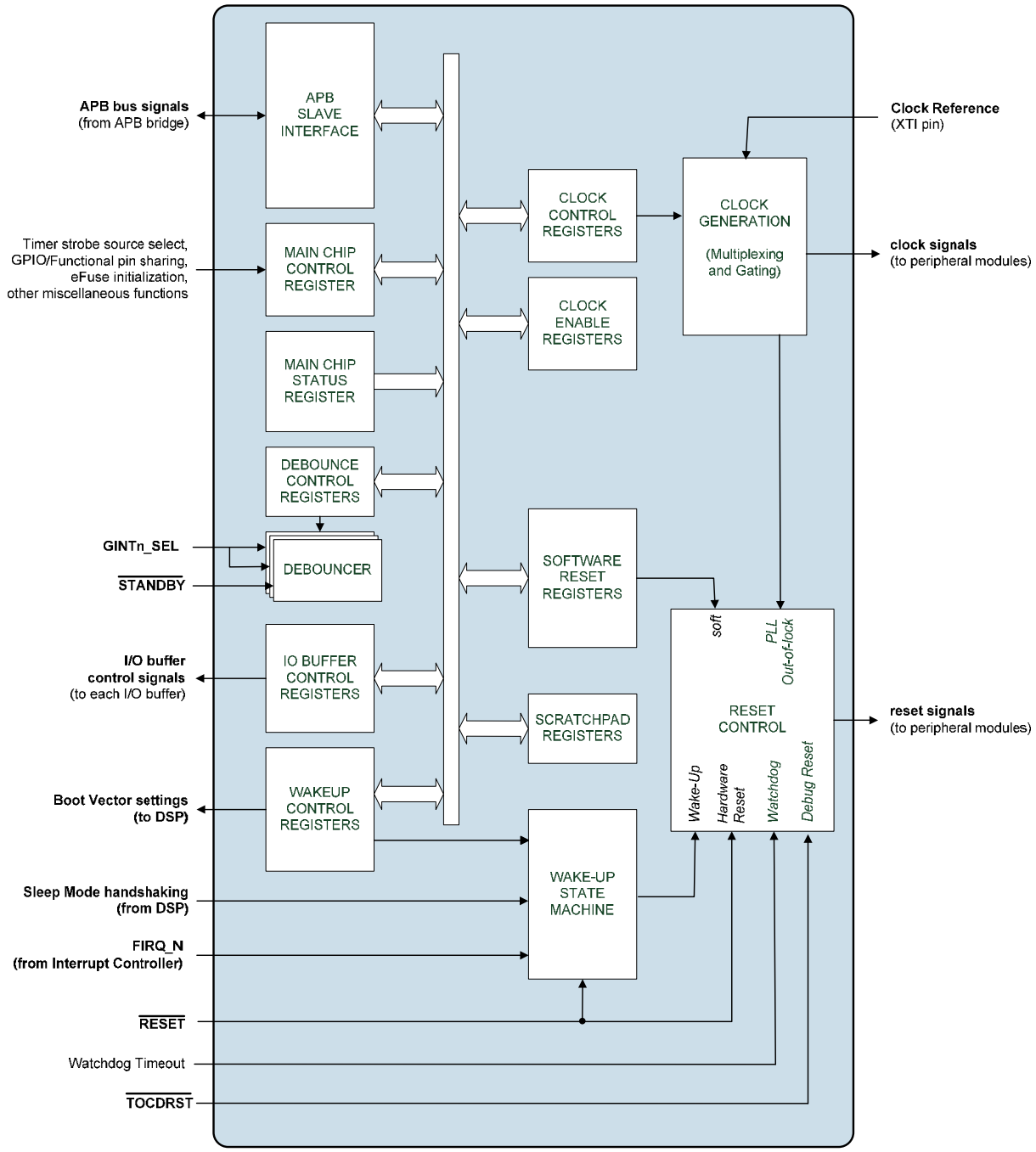


Figure 20 Chip Configuration Module (CCM) Block Diagram

CLOCKING CONTROL

The CCM registers allow full configuration of the WM0011 clocking options, including clock dividers, clock multiplexers and the 2-stage Phase Locked Loop (PLL). Individual clock enable registers are provided for each peripheral module.

RESET CONTROL

The CCM registers allow flexible control of the Warm Reset functions. The Warm Reset conditions are individually maskable, and status readback is also provided. Software Reset control registers allow each peripheral to be reset individually.

INTERFACE PORT SELECTION

The I2C and UART interfaces are supported via multiplexed input/output pins. The SPI and AIF3 interfaces are similarly multiplexed. These ports are configured using the PORTn_SEL fields in the CCM_CONTROL register.

Each of the GPIO pins is multiplexed with one or more serial interface pin function. These pins are configured using the control bits in the CCM_GPIO_SEL register.

GPIO / STANDBY DE-BOUNCE

A maximum of two GPIO pins can be selected as interrupts directly to the HiFi2 EP™ DSP core. The applicable GPIOs are selected using the GINTn_SEL register fields.

The $\overline{\text{STANDBY}}$ pin is an input to the Interrupt Controller module, and also to the HiFi2 EP™ DSP core.

De-bouncing of these inputs to the DSP core can be configured using the control fields in the CCM_DB_STBY, CCM_DB_GINT1 and CCM_DB_GINT2 registers.

I/O BUFFER CONTROL

The CCM provides full control of the input/output enables, drive strength and pull-up/pull-down configuration of the I/O buffer pins. Note that the Pull-Up / Pull-Down capabilities of the I/O pins are noted in the “Pin Description” section.

SLEEP / WAKE-UP CONTROL

The WM0011 supports a ‘Sleep’ mode, suitable for low-power standby and similar requirements.

Note that the application software must ensure that the WM0011 (and associated functions) are configured as required before selecting the Sleep mode.

Sleep mode is commanded by writing ‘1’ to the SLP_ENA bit in the CCM_WKUP_CTRL register.

Note that the SLP_ENA bit does not have any effect on the HiFi2 EP™ DSP core operation; the DSP core sleep state is selected when the core executes a “WAITI” command. The WAIT_HIFI_SLP_ENA bit selects whether to wait for the WAITI to complete before proceeding with the Sleep sequence.

The DSPCLK_SLP_DSBL bit selects whether to disable the DSPCLK in Sleep mode.

The AHBCLK_SLP_DSBL bit selects whether to disable the AHBCLK in Sleep mode.

Note that, if DSPCLK or AHBCLK is disabled in Sleep mode, then the WAIT_HIFI_SLP_ENA bit must be set to ‘1’. This ensures that the DSP core functions are suspended before the clocking is disabled.

Note that, if DSPCLK is disabled in Sleep mode, then the AHBCLK must also be disabled in Sleep.

The AIF_BYP_SEL field controls whether one of the AIF Bypass Modes is selected in Sleep mode. Details of the AIF Bypass modes are provided later in this section.

On completion of the steps described above, the WM0011 will be in Sleep mode.

The trigger for Wake-Up is the FIRQ_N output from the Interrupt Controller (IRQC) module. The $\overline{\text{STANDBY}}$ pin, GPIO pins, and Interrupt signals from the peripheral modules are all inputs to the IRQC module, and may be configured to trigger the WM0011 Wake-Up sequence.

Note that, if DSPCLK or AHBCLK is disabled in Sleep mode, then the $\overline{\text{STANDBY}}$ pin is the only signal that can trigger the Wake-Up sequence.

The AIF_BYP_AUTO_EXIT bit selects whether to exit the AIF Bypass mode (if applicable) as part of the Wake-Up sequence.

The AHBCLK and DSPCLK clocks are re-enabled as part of the Wake-Up sequence.

The WKUP_RST_ENA bit selects whether a Warm Reset is triggered as part of the Wake-Up.

On completion of the steps described above, the WM0011 will be in its normal operating state.

The Sleep and Wake-Up sequences are illustrated in Figure 21.

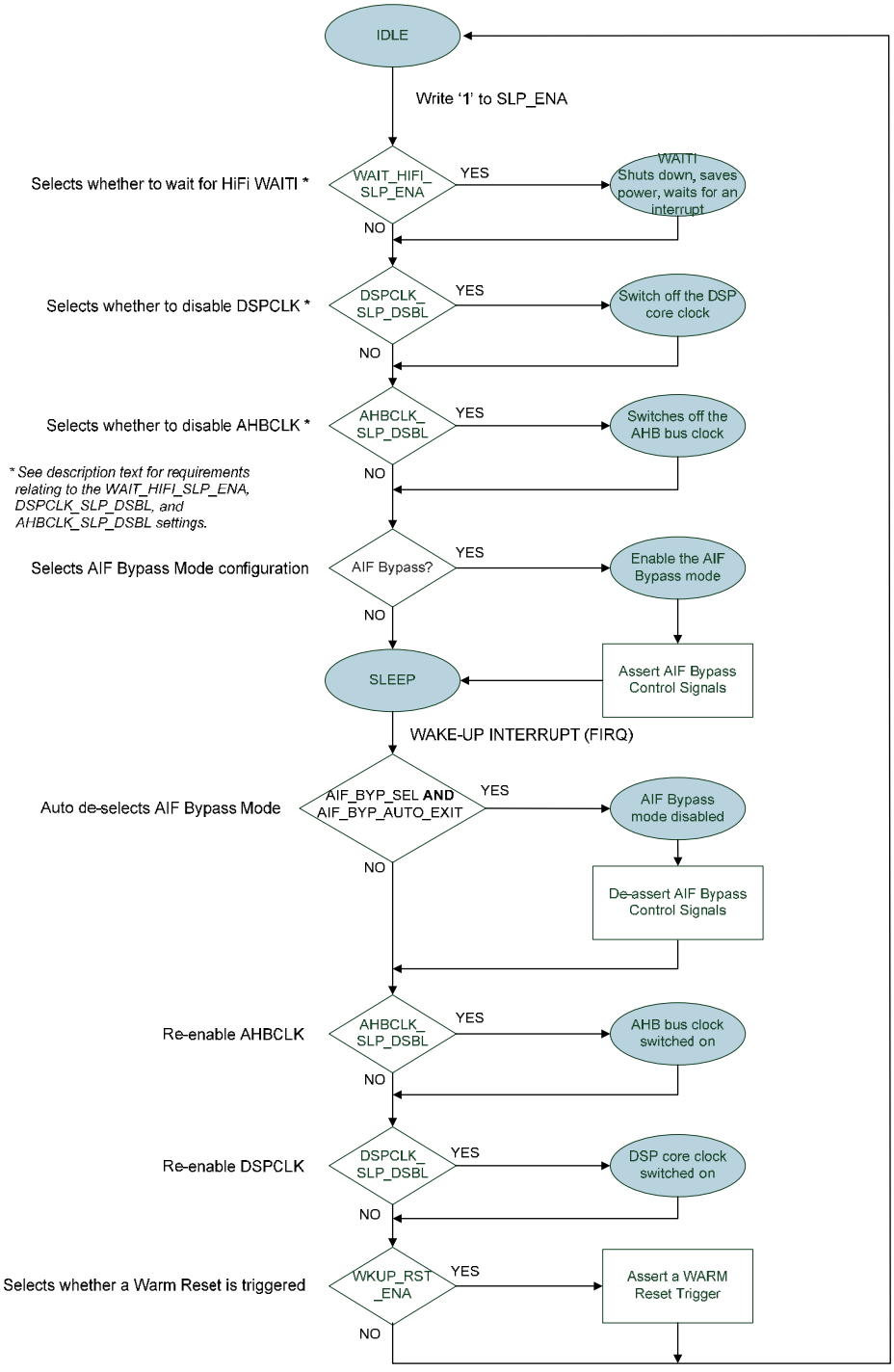


Figure 21 CCM Wake-up

AIF BYPASS MODE

When the WM0011 is in the Sleep mode, the AIF inputs/outputs can be configured in a Bypass mode, allowing AIF data to be looped through the device, with the AIF modules disabled.

The AIF Bypass modes are illustrated in Figure 22.

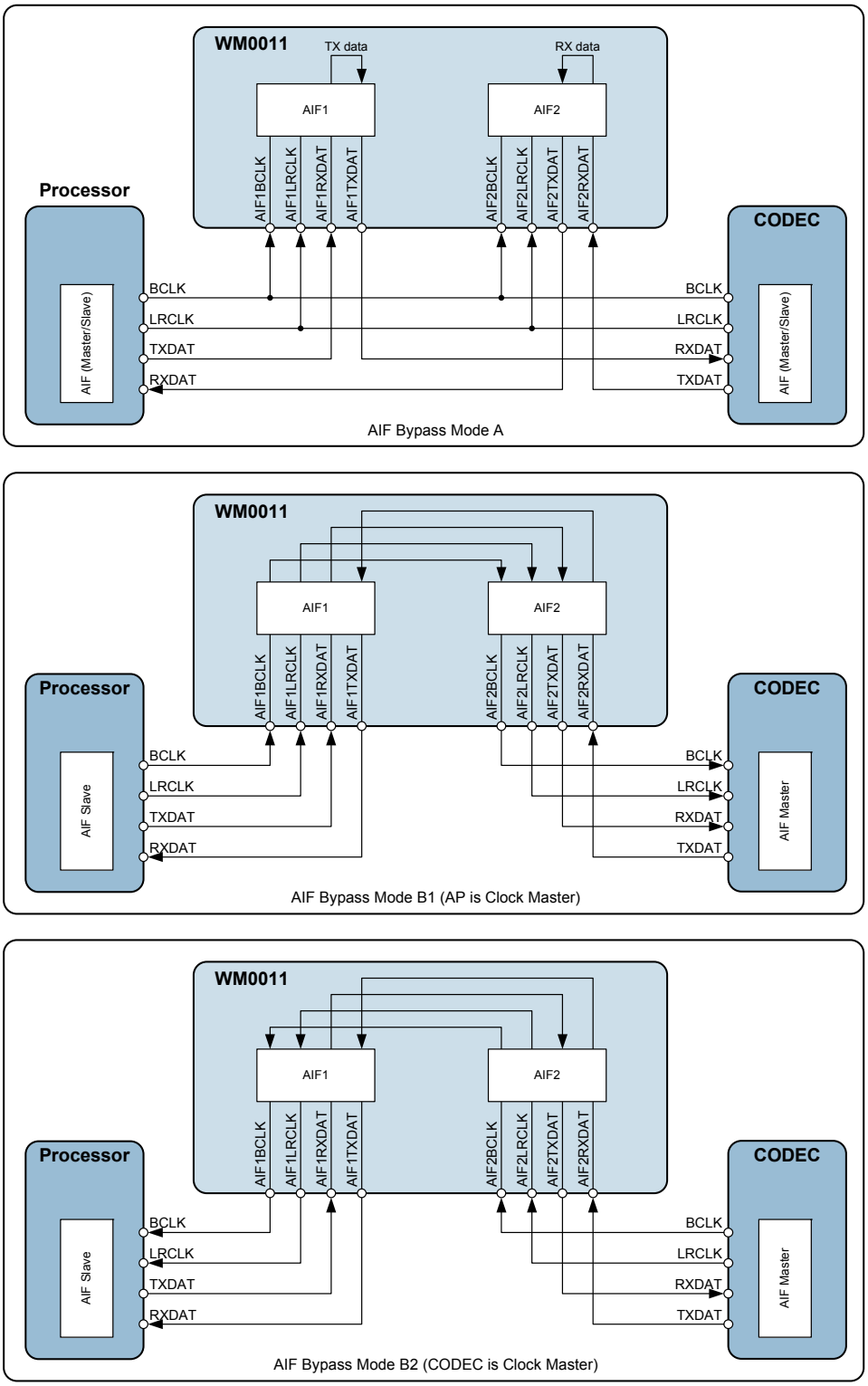


Figure 22 AIF Bypass Modes (Sleep Mode only)

Although the AIF Bypass Modes are intended for use when the WM0011 (including the HiFi2 EP™ DSP core) are in Sleep mode, it is also possible to select AIF Bypass with the DSP core still enabled. This can be achieved using the Sleep and Wake-Up sequences, as described below.

Writing '1' to SLP_ENA will command the WM0011 Sleep mode, as described above. If the DSP "WAITI" command is not executed, and WAIT_HIFI_SLP_ENA, DSPCLK_SLP_DSBL, and AHBCLK_SLP_DSBL are all set to '0', then AIF Bypass can be achieved without interrupting the HiFi2 EP™ DSP core operation. (The desired AIF Bypass mode is selected using the AIF_BYP_SEL field.)

If the Wake-Up sequence is triggered, and AIF_BYP_AUTO_EXIT=0, then the WM0011 will return to normal operation, with the AIF Bypass mode unchanged. Writing '1' to the AIF_BYP_FORCE_EXIT bit will de-select AIF Bypass mode.

CCM REGISTER MAP

The register map of the CCM module is illustrated in Table 15.

ADDRESS	REGISTER	DESCRIPTION	RESET VALUE
Base + 0x00	CCM_CONTROL	General Control	0x001E_1E00
Base + 0x04	CCM_STATUS	General Status	0x8000_0000
Base + 0x0C	CCM_GPIO_SEL	Port Select	0x0000_0000
Base + 0x10	CCM_CLK_CTRL1	Clock Control 1	0x0000_0011
Base + 0x14	CCM_CLK_CTRL2	Clock Control 2	0x03DE_0000
Base + 0x18	CCM_CLK_CTRL3	Clock Control 3	0x0000_0000
Base + 0x1C	CCM_PLL_LOCK_CTRL	PLL Lock Detect Control	0x3312_0E00
Base + 0x24	CCM_CLK_ENA	Clock Enable	0x02BA_187F
Base + 0x28	CCM_SOFTRST	Software Reset	0x00BA_087F
Base + 0x30	CCM_WKUP_CTRL	Chip Wakeup Control	0x0001_0000
Base + 0x44	CCM_DB_STBY	Standby De-bounce Control	0x0000_0000
Base + 0x48	CCM_DB_GINT1	GINT1 De-bounce Control	0x0000_0000
Base + 0x4C	CCM_DB_GINT2	GINT2 De-bounce Control	0x0000_0000
Base + 0x50	CCM_SCRATCH1	Scratchpad 1	0x0000_0000
Base + 0x54	CCM_SCRATCH2	Scratchpad 2	0x0000_0000
Base + 0x58	CCM_SCRATCH3	Scratchpad 3	0x0000_0000
Base + 0x5C	CCM_SCRATCH4	Scratchpad 4	0x0000_0000
Base + 0x60	CCM_IOCTRL1	I/O Control 1	0x7777_0000
Base + 0x64	CCM_IOCTRL2	I/O Control 2	0xFF7F_FF7F
Base + 0x68	CCM_IOCTRL3	I/O Control 3	0x7777_7777
Base + 0x6C	CCM_IOCTRL4	I/O Control 4	0x7777_7700
Base + 0x70	CCM_IOCTRL5	I/O Control 5	0x7D77_7777
Base + 0x74	CCM_IOCTRL6	I/O Control 6	0x7777_7770
Base + 0x78	CCM_IOCTRL7	I/O Control 7	0x0707_0707
Base + 0x7C	CCM_IOCTRL8	I/O Control 8	0x0F07_0700
Base + 0x84	CCM_IOCTRL10	I/O Control 10	0x0707_0707
Base + 0x88	CCM_IOCTRL11	I/O Control 11	0x0F07_0700

Table 15 CCM Register Definition

CCM_CONTROL – GENERAL CONTROL REGISTER

The CCM_CONTROL register contains control fields relating to Warm Reset, Timer (TMR) control sources, Interface port selections and GINTn Interrupts.

See “Power-on and Reset Control” for more details of the Warm Reset function.

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

CCM_CONTROL
GENERAL CONTROL REGISTER

Address = 0xF000_0000 **Default value = 0x001E_1E00**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	PLL_MSK	RW	0x0	Selects whether PLL ‘out-of-lock’ triggers a Warm Reset 0 = PLL ‘out-of-lock’ triggers a Warm Reset 1 = PLL ‘out-of-lock’ does not trigger a Warm Reset
30	OCD_MSK	RW	0x0	Selects whether $\overline{\text{TOCDRST}}$ triggers a Warm Reset 0 = $\overline{\text{TOCDRST}}$ triggers a Warm Reset 1 = $\overline{\text{TOCDRST}}$ does not trigger a Warm Reset
29	WDT_MSK	RW	0x0	Selects whether Watchdog Timeout triggers a Warm Reset 0 = Watchdog Timeout triggers a Warm Reset 1 = Watchdog Timeout does not trigger a Warm Reset
28	Reserved		0x0	
27:26	TMR3_EXT_SEL	RW	0x0	External trigger select for Timer Module (TMR3) 00 = CLKIN 01 = ALTCLK 10 = TMRCLK 11 = Reserved
25:24	TMR2_EXT_SEL	RW	0x0	External trigger select for Timer Module (TMR2) 00 = CLKIN 01 = ALTCLK 10 = TMRCLK 11 = Reserved
23:22	TMR1_EXT_SEL	RW	0x0	External trigger select for Timer Module (TMR1) 00 = CLKIN 01 = ALTCLK 10 = TMRCLK 11 = Reserved
21	Reserved		0x0	
20:16	GINT2_SEL	RW	0x1E	Selects the GPIO pin used as the GINT2 interrupt to the DSP core. 00h to 1Dh = Register coding follows the bit assignments of the CCM_GPIO_SEL register (see Table 18). 1Eh = Constant ‘0’ 1Fh = Constant ‘1’
15:13	Reserved		0x0	
12:8	GINT1_SEL	RW	0x1E	Selects the GPIO pin used as the GINT1 interrupt to the DSP core. 00h to 1Dh = Register coding follows the bit assignments of the CCM_GPIO_SEL register (see Table 18). 1Eh = Constant ‘0’ 1Fh = Constant ‘1’
7:6	PORT2_SEL	RW	0x0	Port 2 function select 00 = UART 01 = Reserved 10 = I2C Slave 11 = I2C Master

CCM_CONTROL GENERAL CONTROL REGISTER																															
Address = 0xF000_0000																	**Default value = 0x001E_1E00**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
5	PORT1_SEL	RW	0x0	Port 1 function select 0 = SPI 1 = AIF3 (I2S/TDM)
4	Reserved		0x0	
3	FUSE_INIT_STS	RO	0x0	Reserved for Wolfson use only
2	FUSE_INIT	WO	0x0	Reserved for Wolfson use only This bit should not be changed from the default value
1	FUSE_PGM_STS	RO	0x0	Reserved for Wolfson use only
0	FUSE_PGM_ENA	RW	0x0	Reserved for Wolfson use only This bit should not be changed from the default value

Table 16 CCM_CONTROL Register

CCM_STATUS – GENERAL STATUS REGISTER

The CCM_STATUS register contains general status bits relating to Warm Reset and PLL 'out-of-lock' conditions. See "Power-on and Reset Control" for more details of the Warm Reset function.

CCM_STATUS GENERAL STATUS REGISTER																															
Address = 0xF000_0004																	**Default value = 0x8000_0000**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	PLL_FLAG	R/W1C	0x1	PLL 'out-of-lock' indicator. This bit is set when the PLL 'out-of-lock' conditions are met. The 'out-of-lock' detection must be enabled using PLL_LOCKDET_ENA (see Table 22). This bit is set regardless of whether the PLL_MSK bit selects a Warm Reset. The bit is latched once set; Write '1' to clear. 0 = No PLL Out-of-Lock detected 1 = PLL Out-of-Lock detected
30	OCD_FLAG	R/W1C	0x0	$\overline{\text{TOCDRST}}$ input indicator. This bit is set when the $\overline{\text{TOCDRST}}$ is asserted. This bit is set regardless of whether the OCD_MSK bit selects a Warm Reset. The bit is latched once set; Write '1' to clear. 0 = $\overline{\text{TOCDRST}}$ has not been asserted 1 = $\overline{\text{TOCDRST}}$ has been asserted
29	WDT_FLAG	R/W1C	0x0	Watchdog Timeout indicator. This bit is set when the Watchdog Timer (WDT) module asserts the timeout indication. This bit is set regardless of whether the WDT_MSK bit selects a Warm Reset. The bit is latched once set; Write '1' to clear. 0 = WDT Timeout has not been asserted 1 = WDT Timeout has been asserted

CCM_STATUS GENERAL STATUS REGISTER																															
Address = 0xF000_0004																Default value = 0x8000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
28	PLL_OVERRIDE_FLAG	R/W1C	0x0	PLL Override indicator. This bit indicates when the CLK_SEL multiplexer has been overridden. The CLK_SEL multiplexer will be overridden, and the CLKIN input selected, following a Warm Reset that was caused by the PLL 'out-of-lock' condition. The bit is latched once set; Write '1' to clear. 0 = CLK_SEL has not been overridden 1 = CLK_SEL has been overridden
27	WKUP_FLAG	R/W1C	0x0	Wakeup Reset indicator. This bit indicates when a Warm Reset has been triggered as part of a Wake-Up transition. The bit is latched once set; Write '1' to clear. 0 = Wake-Up Reset has not been triggered 1 = Wake-Up Reset has been triggered
26:2	Reserved		0x0	
1	PLL_UNLOCK	RO	0x0	PLL 'out-of-lock' indicator This is the output of the configurable 'out-of-lock' detection circuit. The function must be enabled using PLL_LOCKDET_ENA (see Table 22). 0 = PLL second stage is locked 1 = PLL second stage is out-of-lock
0	PLL_RAW_LOCK	RO	0x0	PLL Lock indicator This is the raw indication of the PLL Lock status. Note that the PLL output should not be selected as clock source until this bit indicates the PLL is locked. 0 = PLL is out-of-lock 1 = PLL is locked

Table 17 CCM_STATUS Register

CCM_GPIO_SEL – PORT SELECT REGISTER

The CCM_GPIO_SEL register contains configuration bits for selecting the function of the GPIO pins. Note that the PORTn_SEL fields in the CCM_CONTROL register also determine the pin functionality in some cases.

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

CCM_GPIO_SEL PORT SELECT REGISTER																															
Address = 0xF000_000C																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	Reserved		0x0	
28	GPIO28_CLKOUT	RW	0x0	Pin function select: 0 = CLKOUT 1 = GPIO28
27:24	Reserved		0x0	

CCM_GPIO_SEL
PORT SELECT REGISTER

Address = 0xF000_000C | **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
23	GPIO23_UARTTX	RW	0x0	Pin function select: 0 = UARTTX 1 = GPIO23 (Note that PORT2_SEL must be set to 00.) When PORT2_SEL=10, function is SCLK1, regardless of this bit. When PORT2_SEL=11, function is SCLK2, regardless of this bit.
22	GPIO22_UARTRX	RW	0x0	Pin function select: 0 = UARTRX 1 = GPIO22 (Note that PORT2_SEL must be set to 00.) When PORT2_SEL=10, function is SDA1, regardless of this bit. When PORT2_SEL=11, function is SDA2, regardless of this bit.
21:20	Reserved		0x0	
19	GPIO19_SPIMISO	RW	0x0	Pin function select: 0 = SPIMISO 1 = GPIO19 (Note that PORT1_SEL must be set to 0.) When PORT1_SEL=1, function is AIF3RXDAT, regardless of this bit.
18	GPIO18_SPIMOSI	RW	0x0	Pin function select: 0 = SPIMOSI 1 = GPIO18 (Note that PORT1_SEL must be set to 0.) When PORT1_SEL=1, function is AIF3TXDAT, regardless of this bit.
17	GPIO17_SPISS	RW	0x0	Pin function select: 0 = $\overline{\text{SPISS}}$ 1 = GPIO17 (Note that PORT1_SEL must be set to 0.) When PORT1_SEL=1, function is AIF3LRCLK, regardless of this bit.
16:15	Reserved		0x0	
14	GPIO14_SEL	RW	0x0	Pin function select: 0 = GPIO14 disabled 1 = GPIO14 enabled Note that the I/O configuration settings in the CCM_IOCTRL10 register (eg. pull-up/down) are valid at all times, regardless of GPIO14_SEL.
13	GPIO13_SEL	RW	0x0	Pin function select: 0 = GPIO13 disabled 1 = GPIO13 enabled Note that the I/O configuration settings in the CCM_IOCTRL10 register (eg. pull-up/down) are valid at all times, regardless of GPIO13_SEL.
12	GPIO12_SEL	RW	0x0	Pin function select: 0 = GPIO12 disabled 1 = GPIO12 enabled Note that the I/O configuration settings in the CCM_IOCTRL10 register (eg. pull-up/down) are valid at all times, regardless of GPIO12_SEL.
11	GPIO11_SEL	RW	0x0	Pin function select: 0 = GPIO11 disabled 1 = GPIO11 enabled Note that the I/O configuration settings in the CCM_IOCTRL10 register (eg. pull-up/down) are valid at all times, regardless of GPIO11_SEL.
10	GPIO10_SEL	RW	0x0	Pin function select: 0 = GPIO10 disabled 1 = GPIO10 enabled Note that the I/O configuration settings in the CCM_IOCTRL11 register (eg. pull-up/down) are valid at all times, regardless of GPIO10_SEL.

CCM_GPIO_SEL PORT SELECT REGISTER																															
Address = 0xF000_000C																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
9	GPIO9_SEL	RW	0x0	Pin function select: 0 = GPIO9 disabled 1 = GPIO9 enabled Note that the I/O configuration settings in the CCM_IOCTRL8 register (eg. pull-up/down) are valid at all times, regardless of GPIO9_SEL.
8	GPIO8_SEL	RW	0x0	Pin function select: 0 = GPIO8 disabled 1 = GPIO8 enabled Note that the I/O configuration settings in the CCM_IOCTRL7 register (eg. pull-up/down) are valid at all times, regardless of GPIO8_SEL.
7	GPIO7_SEL	RW	0x0	Pin function select: 0 = GPIO7 disabled 1 = GPIO7 enabled Note that the I/O configuration settings in the CCM_IOCTRL7 register (eg. pull-up/down) are valid at all times, regardless of GPIO7_SEL.
6	GPIO6_SEL	RW	0x0	Pin function select: 0 = GPIO6 disabled 1 = GPIO6 enabled Note that the I/O configuration settings in the CCM_IOCTRL7 register (eg. pull-up/down) are valid at all times, regardless of GPIO6_SEL.
5	GPIO5_SEL	RW	0x0	Pin function select: 0 = GPIO5 disabled 1 = GPIO5 enabled Note that the I/O configuration settings in the CCM_IOCTRL7 register (eg. pull-up/down) are valid at all times, regardless of GPIO5_SEL.
4	GPIO4_SEL	RW	0x0	Pin function select: 0 = GPIO4 disabled 1 = GPIO4 enabled Note that the I/O configuration settings in the CCM_IOCTRL8 register (eg. pull-up/down) are valid at all times, regardless of GPIO4_SEL.
3:0	Reserved		0x0	

Table 18 CCM_GPIO_SEL Register

CCM_CLK_CTRL1 – CLOCK CONTROL 1 REGISTER

The CCM_CLK_CTRL1 register contains clocking configuration registers. See “Clocking” for more details of the Clocking architecture.

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

Note that this register can be written and updated prior to software download using the “PLL Configuration Download” code packet.

CCM_CLK_CTRL1
CLOCK CONTROL 1 REGISTER

Address = 0xF000_0010 | **Default value = 0x0000_0011**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:30	CLK_SEL_AIF3	RW	0x0	Clock source select for AIF3_MSTR_CLK 00 = CLKIN 01 = MCLK_AIF3 10 = ALTCLK 11 = Reserved The ALTCLK source is selected using ALTCLK_REF_SEL. The MCLK_AIF3 clock is derived from PLLOUT, via a configurable divider (MCLK_DIV) in the AIF3 module.
29:28	CLK_SEL_AIF2	RW	0x0	Clock source select for AIF2_MSTR_CLK 00 = CLKIN 01 = MCLK_AIF2 10 = ALTCLK 11 = Reserved The ALTCLK source is selected using ALTCLK_REF_SEL. The MCLK_AIF2 clock is derived from PLLOUT, via a configurable divider (MCLK_DIV) in the AIF2 module.
27:26	CLK_SEL_AIF1	RW	0x0	Clock source select for AIF1_MSTR_CLK 00 = CLKIN 01 = MCLK_AIF1 10 = ALTCLK 11 = Reserved The ALTCLK source is selected using ALTCLK_REF_SEL. The MCLK_AIF1 clock is derived from PLLOUT, via a configurable divider (MCLK_DIV) in the AIF1 module.
25:24	UART_CLK_SEL	RW	0x0	Clock source select for UART_MSTR_CLK 00 = CLKIN 01 = PLLOUT 10 = ALTCLK 11 = Reserved The ALTCLK source is selected using ALTCLK_REF_SEL. Only valid when UART_REF_SEL=0.
23:22	Reserved		0x0	
21:16	UART_CLK_PRESCALE	RW	0x00	Pre-scaler for UART_MSTR_CLK. The UART_MSTR_CLK source is selected using UART_CLK_SEL. The pre-scale division is controlled by this register. 00h = Divide by 1 01h = Divide by 2 …. 3Fh = Divide by 64 Only valid when UART_REF_SEL=0.
15:8	CLK_MAIN_PRESCALE	RW	0x00	Pre-scaler for DSPCLK. The clock source is selected using CLK_SEL. The pre-scale division is controlled by this register. 00h = Divide by 1 01h = Divide by 2 …. FFh = Divide by 256
7	Reserved		0x0	

CCM_CLK_CTRL1 CLOCK CONTROL 1 REGISTER																															
Address = 0xF000_0010																Default value = 0x0000_0011															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
6:4	CLK_APB_SCALE	RW	0x1	APBCLK Clock Division. Sets the APBCLK frequency with respect to AHBCLK frequency. 000 = Divide by 1 001 = Divide by 2 010 = Divide by 4 011 = Divide by 8 100 = Divide by 16 101 = Divide by 32 11X = Reserved
3:2	Reserved		0x0	
1:0	CLK_AHB_SCALE	RW	0x1	AHBCLK Clock Division. Sets the AHBCLK frequency with respect to DSPCLK frequency. 00 = Divide by 1 01 = Divide by 2 10 = Divide by 4 11 = Reserved

Table 19 CCM_CLK_CTRL1 Register

CCM_CLK_CTRL2 – CLOCK CONTROL 2 REGISTER

The CCM_CLK_CTRL2 register contains clocking configuration registers, including some of the PLL controls. See “Clocking” for more details of the Clocking architecture.

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

Note that this register can be written and updated prior to software download using the “PLL Configuration Download” code packet.

CCM_CLK_CTRL2 CLOCK CONTROL 2 REGISTER																															
Address = 0xF000_0014																Default value = 0x03DE_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	Reserved		0x0	
30	CLK_UART_DIV27	RW	0x0	UART_MSTR_CLK division The UART Master clock source is selected using the UART_REF_SEL and UART_CLK_SEL fields. This register enables ‘Divide by 27’ function. 0 = No division 1 = Divide by 27
29	Reserved		0x0	
28	UART_REF_SEL	RW	0x0	UART_MSTR_CLK source select 0 = Clock source is selected by UART_CLK_SEL 1 = CLKIN direct (note - the UART Clock pre-scaler is bypassed)
27:26	Reserved		0x0	

CCM_CLK_CTRL2
CLOCK CONTROL 2 REGISTER

Address = 0xF000_0014 **Default value = 0x03DE_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
25:21	TMRCLK_REF_SEL	RW	0x1E	Selects the GPIO pin used as the TMRCLK source. The TMRCLK signal can be selected as an external trigger for the Timer (TMR) modules. 00h to 1Dh = Register coding follows the bit assignments of the CCM_GPIO_SEL register (see Table 18). 1Eh = Constant '0' 1Fh = Constant '1'
20:16	ALTCLK_REF_SEL	RW	0x1E	Selects the GPIO pin used as the ALTCLK source. The ALTCLK signal can be selected as the reference clock for one or more modules. 00h to 1Dh = Register coding follows the bit assignments of the CCM_GPIO_SEL register (see Table 18). 1Eh = Constant '0' 1Fh = Constant '1'
15	Reserved		0x0	
14:12	PLL2_FRANGE_MSK	RW	0x0	PLL2 Filter Range select Configures the 2nd stage PLL for the required frequency range. The register should select the highest valid range for the PLL2 reference frequency. Note that the reference frequency is the input frequency, after division by the PLL2_INDIV register setting. 000 = Bypass 001 = 5MHz to 10MHz 010 = 10MHz to 16MHz 011 = 16MHz to 26MHz 100 = 26MHz to 42MHz 101 = 42MHz to 68MHz 110 = 68MHz to 108MHz 111 = 108MHz to 200MHz
11	Reserved		0x0	
10:8	PLL1_FRANGE_MSK	RW	0x0	PLL1 Filter Range select Configures the 1st stage PLL for the required frequency range. The register should select the highest valid range for the PLL1 reference frequency. Note that the reference frequency is the input frequency, after division by the PLL1_INDIV register setting. 000 = Bypass 001 = 5MHz to 10MHz 010 = 10MHz to 16MHz 011 = 16MHz to 26MHz 100 = 26MHz to 42MHz 101 = 42MHz to 68MHz 110 = 68MHz to 108MHz 111 = 108MHz to 200MHz
7	PLL2_BYPASS	RW	0x0	PLL2 Bypass 0 = Do not bypass PLL2 1 = Bypass PLL2
6	PLL1_BYPASS	RW	0x0	PLL1 Bypass 0 = Do not bypass PLL1 1 = Bypass PLL1
5	PLL2_RST	RW	0x0	PLL2 Reset 0 = No Reset 1 = Reset PLL2

CCM_CLK_CTRL2 CLOCK CONTROL 2 REGISTER																															
Address = 0xF000_0014																**Default value = 0x03DE_0000**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
4	PLL1_RST	RW	0x0	PLL1 Reset 0 = No Reset 1 = Reset PLL1
3	Reserved		0x0	
2:0	CLK_SEL	RW	0x0	Clock source select for the main system clock. Note that a glitch-free switchover between CLK_SEL settings is implemented. 000 = CLKIN 001 = PLLOUT 010 = ALTCLK All other settings are Reserved. The ALTCLK source is selected using ALTCLK_REF_SEL.

Table 20 CCM_CLK_CTRL2 Register

CCM_CLK_CTRL3 – CLOCK CONTROL 3 REGISTER

The CCM_CLK_CTRL3 register contains PLL configuration fields. See “Clocking” for more details of the PLL.

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

Note that this register can be written and updated prior to software download using the “PLL Configuration Download” code packet.

CCM_CLK_CTRL3 CLOCK CONTROL 3 REGISTER																															
Address = 0xF000_0018																**Default value = 0x0000_0000**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:24	PLL2_FRATIO	RW	0x00	PLL2 frequency ratio Sets the ratio of Fvco/Fref for PLL2 00h = 1 01h = 2 …. FFh = 256
23:19	PLL2_INDIV	RW	0x00	PLL2 input divider. 00h = Divide by 1 01h = Divide by 2 …. 1Fh = Divide by 32

CCM_CLK_CTRL3 CLOCK CONTROL 3 REGISTER																															
Address = 0xF000_0018																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
18:16	PLL2_OUTDIV	RW	0x0	PLL2 output divider 0h = Divide by 1 1h = Divide by 2 2h = Divide by 4 3h = Divide by 8 4h = Divide by 16 5h = Divide by 32 6h = Divide by 64 7h = Reserved
15:8	PLL1_FRATIO	RW	0x00	PLL1 frequency ratio Sets the ratio of Fvco/Fref for PLL1 00h = 1 01h = 2 …. FFh = 256
7:3	PLL1_INDIV	RW	0x00	PLL1 input divider. 00h = Divide by 1 01h = Divide by 2 …. 1Fh = Divide by 32
2:0	PLL1_OUTDIV	RW	0x0	PLL1 output divider 0h = Divide by 1 1h = Divide by 2 2h = Divide by 4 3h = Divide by 8 4h = Divide by 16 5h = Divide by 32 6h = Divide by 64 7h = Reserved

Table 21 CCM_CLK_CTRL3 Register

CCM_PLL_LOCK_CTRL – PLL LOCK DETECT CONTROL REGISTER

The CCM_PLL_LOCK_CTRL register contains fields that control the PLL 'out-of-lock' detection function. See "Clocking" for more details of the PLL.

The PLL out-of-lock detection function is enabled using PLL_LOCKDET_ENA.

The PLL lock detection is derived by checking the ratio of the PLL2 output frequency with respect to the PLL1 input frequency; a count is maintained of instances when the ratio is outside the limits set by PLL_LOCKDET_MIN and PLL_LOCKDET_MAX.

If the frequency ratio exceeds PLL_LOCKDET_MAX, this is counted as an Overflow condition.

If the frequency ratio is below PLL_LOCKDET_MIN, this is counted as an Underflow condition.

In Absolute Mode (PLL_LOCKDET_MODE=0), the Overflow and Underflow occurrences are counted cumulatively (no distinction is made between Overflow and Underflow conditions); the PLL 'out-of-lock' condition is asserted if the count exceeds PLL_OVERFLOW_LIMIT.

In Plus/Minus Mode (PLL_LOCKDET_MODE=1), the difference in the number of overflows vs underflows is counted. In this mode, an Overflow detection effectively 'cancels out' an earlier Underflow detection, and vice versa. The PLL 'out-of-lock' condition is asserted if either of the count limits is reached.

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

Note that this register can be written and updated prior to software download using the "PLL Configuration Download" code packet.

CCM_PLL_LOCK_CTRL
PLL LOCK DETECT CONTROL REGISTER

Address = 0xF000_001C | **Default value = 0x3312_0E00**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:28	PLL_UNDERFLOW_LIMIT	RW	0x3	PLL Underflow Limit Valid in Plus/Minus mode only (PLL_LOCKDET_MODE=1). Number of Underflows permitted before 'out-of-lock' is indicated. 0h = 16 underflows 1h = 15 underflows 2h = 14 underflows 3h = 13 underflows … Dh = 3 underflows Eh = 2 underflows Fh = 1 underflow Note this value should not be changed while the PLL is running.
27:24	PLL_OVERFLOW_LIMIT	RW	0x3	PLL Overflow Limit In Plus/Minus mode (PLL_LOCKDET__MODE=1), this sets the number of Overflows permitted before 'out-of-lock' is indicated. In Absolute mode (PLL_LOCKDET_MODE=0), this sets the cumulative number of Underflows or Overflows permitted before 'out-of-lock- is indicated. 0h = 0 overflows 1h = 1 overflow 2h = 2 overflows 3h = 3 overflows … Dh = 13 overflows Eh = 14 overflows Fh = 15 overflows Note this value should not be changed while the PLL is running.
23	Reserved		0x0	
22:16	PLL_LOCKDET_MIN	RW	0x12	PLL Minimum Detection ratio An Underflow is detected if the output/input frequency ratio is below this threshold.
15	Reserved		0x0	
14:8	PLL_LOCKDET_MAX	RW	0x0E	PLL Maximum Detection ratio An Overflow is detected if the output/input frequency ratio is above this threshold.
7:2	Reserved		0x00	

CCM_PLL_LOCK_CTRL PLL LOCK DETECT CONTROL REGISTER																															
Address = 0xF000_001C																Default value = 0x3312_0E00															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
1	PLL_LOCKDET_MODE	RW	0x0	PLL Lock Detector Mode 0 = Absolute mode 1 = Plus/minus mode In 'absolute mode', the number of overflows and underflows are counted cumulatively. The 'out-of-lock' condition is asserted if the count exceeds PLL_OVERFLOW_LIMIT. In 'plus/minus mode', the difference in the number of overflows vs underflows is counted. The 'out-of-lock' condition is asserted if the PLL_OVERFLOW_LIMIT or PLL_UNDERFLOW_LIMIT threshold is exceeded.
0	PLL_LOCKDET_ENA	RW	0x0	PLL Lock Detector Enable 0 = Disabled 1 = Enabled Note that this bit must be the last bit written on this register. It must be written as a separate write action after all the other bits have been written.

Table 22 CCM_PLL_LOCK_CTRL Register

CCM_CLK_ENA – CLOCK ENABLE REGISTER

The CCM_CLK_ENA register contains the enable bits for the clock signals to each peripheral module. The DSPCLK_SLP_DSBL and AHBCLK_SLP_DSBL bit select whether the respective clock is enabled in Sleep mode. See “Clocking” for more details of the Clocking architecture.

CCM_CLK_ENA
CLOCK ENABLE REGISTER

Address = 0xF000_0024 | **Default value = 0x02BA_187F**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	DSPCLK_SLP_DSBL	RW	0x0	DSPCLK control in Sleep mode 0 = DSPCLK enabled in Sleep mode 1 = DSPCLK disabled in Sleep mode
30	AHBCLK_SLP_DSBL	RW	0x0	AHBCLK control in Sleep mode 0 = AHBCLK enabled in Sleep mode 1 = AHBCLK disabled in Sleep mode
29:26	Reserved		0x0	
25	UART_MSTR_CLK_ENA	RW	0x1	UART master clock (UART_MSTR_CLK) enable 0 = Disabled 1 = Enabled
24	Reserved		0x0	
23	UART_CLK_ENA	RW	0x1	Enable APBCLK to UART module 0 = Disabled 1 = Enabled
22	TRAX_CLK_ENA	RW	0x0	Enable APBCLK to TRAX module 0 = Disabled 1 = Enabled
21	TMR_CLK_ENA	RW	0x1	Enable APBCLK to TMR modules (TMR1, TMR2 and TMR3) 0 = Disabled 1 = Enabled
20	WDT_CLK_ENA	RW	0x1	Enable APBCLK to WDT module 0 = Disabled 1 = Enabled
19	IRQC_CLK_ENA	RW	0x1	Enable APBCLK to IRQC module 0 = Disabled 1 = Enabled
18	GPIO_CLK_ENA	RW	0x0	Enable APBCLK to GPIO module 0 = Disabled 1 = Enabled
17	FUSE_CLK_ENA	RW	0x1	Enable APBCLK to FUSE module 0 = Disabled 1 = Enabled
16	I2C_CLK_ENA	RW	0x0	Enable APBCLK to I2C module 0 = Disabled 1 = Enabled
15	AIF3_MSTR_CLK_ENA	RW	0x0	AIF3 master clock (AIF3_MSTR_CLK) enable 0 = Disabled 1 = Enabled
14	AIF2_MSTR_CLK_ENA	RW	0x0	AIF3 master clock (AIF3_MSTR_CLK) enable 0 = Disabled 1 = Enabled
13	AIF1_MSTR_CLK_ENA	RW	0x0	AIF3 master clock (AIF3_MSTR_CLK) enable 0 = Disabled 1 = Enabled
12	Reserved		0x1	

CCM_CLK_ENA CLOCK ENABLE REGISTER																															
Address = 0xF000_0024																Default value = 0x02BA_187F															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
11	SEC_CLK_ENA	RW	0x1	Enable AHBCLK to SEC module 0 = Disabled 1 = Enabled
10	RNG_CLK_ENA	RW	0x0	Enable AHBCLK to RNG module 0 = Disabled 1 = Enabled
9	AIF3_CLK_ENA	RW	0x0	Enable AHBCLK to AIF3 module 0 = Disabled 1 = Enabled
8	AIF2_CLK_ENA	RW	0x0	Enable AHBCLK to AIF2 module 0 = Disabled 1 = Enabled
7	AIF1_CLK_ENA	RW	0x0	Enable AHBCLK to AIF1 module 0 = Disabled 1 = Enabled
6	SHA_CLK_ENA	RW	0x1	Enable AHBCLK to SHA module 0 = Disabled 1 = Enabled
5:4	Reserved		0x3	
3	ROM_CLK_ENA	RW	0x1	Enable AHBCLK to ROM module 0 = Disabled 1 = Enabled
2	RAM_CLK_ENA	RW	0x1	Enable AHBCLK to RAM module 0 = Disabled 1 = Enabled
1	SPI_CLK_ENA	RW	0x1	Enable AHBCLK to SPI module 0 = Disabled 1 = Enabled
0	DMA_CLK_ENA	RW	0x1	Enable AHBCLK to DMA module 0 = Disabled 1 = Enabled

Table 23 CCM_CLK_ENA Register

CCM_SOFTRST – SOFTWARE RESET REGISTER

The CCM_SOFTRST register contains the software reset bits for each peripheral module. See "Power-on and Reset Control" for more details of the Software Reset function.

CCM_SOFTRST
CHIP SOFTWARE RESET REGISTER

Address = 0xF000_0028 | **Default value = 0x00BA_087F**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:24	Reserved		0x00	
23	UART_SOFTRST_N	RW	0x1	UART Software Reset control 0 = Reset 1 = Not reset
22	TRAX_SOFTRST_N	RW	0x0	TRAX Software Reset control 0 = Reset 1 = Not reset
21	TMR_SOFTRST_N	RW	0x1	TMR1, TMR2, TMR3 Software Reset control 0 = Reset 1 = Not reset
20	WDT_SOFTRST_N	RW	0x1	WDT Software Reset control 0 = Reset 1 = Not reset
19	IRQC_SOFTRST_N	RW	0x1	IRQC Software Reset control 0 = Reset 1 = Not reset
18	GPIO_SOFTRST_N	RW	0x0	GPIO Software Reset control 0 = Reset 1 = Not reset
17	FUSE_SOFTRST_N	RW	0x1	FUSE Software Reset control 0 = Reset 1 = Not reset
16	I2C_SOFTRST_N	RW	0x0	I2C Software Reset control 0 = Reset 1 = Not reset
15:12	Reserved		0x0	
11	SEC_SOFTRST_N	RW	0x1	SEC Software Reset control 0 = Reset 1 = Not reset
10	RNG_SOFTRST_N	RW	0x0	RNG Software Reset control 0 = Reset 1 = Not reset
9	AIF3_SOFTRST_N	RW	0x0	AIF3 Software Reset control 0 = Reset 1 = Not reset
8	AIF2_SOFTRST_N	RW	0x0	AIF2 Software Reset control 0 = Reset 1 = Not reset
7	AIF1_SOFTRST_N	RW	0x0	AIF1 Software Reset control 0 = Reset 1 = Not reset
6	SHA_SOFTRST_N	RW	0x1	SHA Software Reset control 0 = Reset 1 = Not reset
5:4	Reserved		0x3	

CCM_SOFTRST CHIP SOFTWARE RESET REGISTER			
Address = 0xF000_0028			Default value = 0x00BA_087F
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0			
BITS	**FIELD NAME**	**S/W ACCESS**	**RESET VALUE**	**FIELD DESCRIPTION**
3	ROM_SOFTRST_N	RW	0x1	ROM Software Reset control 0 = Reset 1 = Not reset
2	RAM_SOFTRST_N	RW	0x1	RAM Software Reset control 0 = Reset 1 = Not reset
1	SPI_SOFTRST_N	RW	0x1	SPI Software Reset control 0 = Reset 1 = Not reset
0	DMA_SOFTRST_N	RW	0x1	DMA Software Reset control 0 = Reset 1 = Not reset

Table 24 CCM_SOFTRST Register

CCM_WKUP_CTRL – CHIP WAKEUP CONTROL REGISTER

The CCM_WKUP_CTRL register provides control of the Sleep and Wake-Up mode transitions. See "AIF Interface Modules" for details of the AIF Bypass modes.

The STATIC_VECT_SEL register bit selects the boot vector address for code execution following a Warm Reset (see "Power-on and Reset Control").

CCM_WKUP_CTRL CHIP WAKEUP CONTROL REGISTER			
Address = 0xF000_0030			Default value = 0x0001_0000
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0			
BITS	**FIELD NAME**	**S/W ACCESS**	**RESET VALUE**	**FIELD DESCRIPTION**
31	SLP_ENA	WO	0x0	Sleep Mode select Write '1' to select Sleep mode.
30:8	Reserved		0x00_0000	
7	AIF_BYP_FORCE_EXIT	WO	0x0	AIF Bypass Mode Forced Exit Write '1' to de-select the AIF Bypass mode. (This is typically used if when AIF_BYP_AUTO_EXIT=0, and the AIF module remains in Bypass Mode after Wake-Up.)
6	AIF_BYP_AUTO_EXIT	RW	0x0	AIF Bypass Mode Auto Exit Selects whether the AIF Bypass mode is de-selected on Wake-Up 0 = AIF Bypass not de-selected on Wake-Up 1 = AIF Bypass de-selected on Wake-Up
5:4	AIF_BYP_SEL	RW	0x0	AIF Bypass Mode control Selects whether one of the AIF Bypass Modes is selected when entering Sleep mode. 00 = No bypass 01 = Bypass Mode A 10 = Bypass Mode B1 11 = Bypass Mode B2

CCM_WKUP_CTRL CHIP WAKEUP CONTROL REGISTER																															
Address = 0xF000_0030																Default value = 0x0001_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
3	WAIT_HIFI_SLP_ENA	RW	0x0	HiFi EP™ Sleep WAITI control 0 = Do not wait for HiFi WAITI to complete before entering Sleep mode 1 = Wait for HiFi WAITI to complete before entering Sleep mode
2	WKUP_RST_ENA	RW	0x0	Selects whether a Wake-Up (exit from Sleep) triggers a Warm Reset 0 = Wake-Up does not trigger a Warm Reset 1 = Wake-Up triggers a Warm Reset
0	STATIC_VECT_SEL	RW	0x0	Selects the DSP core boot vector following a Wake-Up transition. Only valid when WKUP_RST_ENA=1. 0 = Boot to the primary static vector 1 = Boot to the alternate static vector

Table 25 CCM_WKUP_CTRL Register

CCM_DB_STBY – STANDBY DE-BOUNCE CONTROL REGISTER

The CCM_DB_STBY register configures the de-bounce circuit for the $\overline{\text{STANDBY}}$ input pin. The de-bounced $\overline{\text{STANDBY}}$ is an input to the IRQC module, and also one of the DSP core interrupt inputs. See “Interrupts” for details of the DSP core interrupts.

CCM_DB_STBY STANDBY DE-BOUNCE CONTROL REGISTER																															
Address = 0xF000_0044																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	STBY_DB_BYP	RW	0x0	$\overline{\text{STANDBY}}$ de-bounce select 0 = De-bounce enabled 1 = De-bounce disabled (bypass)
30:24	Reserved		0x0	
23:0	STBY_DB_CNT	RW	0x00_0000	$\overline{\text{STANDBY}}$ de-bounce time - sets the number of APBCLK clock cycles for de-bouncing the $\overline{\text{STANDBY}}$ input pin.

Table 26 CCM_DB_STBY Register

CCM_DB_GINT1 – GINT1 DE-BOUNCE CONTROL REGISTER

The CCM_DB_GINT1 register configures the de-bounce circuit for the GINT1 interrupt signal. The GPIO pin used as the GINT1 source is selected by the GINT1_SEL register. The de-bounced GINT1 is one of the DSP core interrupt inputs. See “Interrupts” for details of the DSP core interrupts.

CCM_DB_GINT1
GINT1 DE-BOUNCE CONTROL REGISTER

Address = 0xF000_0048 **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	GINT1_DB_BYP	RW	0x0	GINT1 de-bounce select 0 = De-bounce enabled 1 = De-bounce disabled (bypass)
30:24	Reserved		0x0	
23:0	GINT1_DB_CNT	RW	0x00_0000	GINT1 de-bounce time - sets the number of APBCLK clock cycles for de-bouncing the GINT1 input. The GPIO pin used as the GINT1 source is selected by GINT1_SEL (see Table 16).

Table 27 CCM_DB_GINT1 Register

CCM_DB_GINT2 – GINT2 DE-BOUNCE CONTROL REGISTER

The CCM_DB_GINT2 register configures the de-bounce circuit for the GINT2 interrupt signal. The GPIO pin used as the GINT2 source is selected by the GINT2_SEL register. The de-bounced GINT2 is one of the DSP core interrupt inputs. See “Interrupts” for details of the DSP core interrupts

CCM_DB_GINT2
GINT2 DE-BOUNCE CONTROL REGISTER

Address = 0xF000_004C **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	GINT2_DB_BYP	RW	0x0	GINT2 de-bounce select 0 = De-bounce enabled 1 = De-bounce disabled (bypass)
30:24	Reserved		0x0	
23:0	GINT2_DB_CNT	RW	0x00_0000	GINT2 de-bounce time - sets the number of APBCLK clock cycles for de-bouncing the GINT2 input. The GPIO pin used as the GINT2 source is selected by GINT2_SEL (see Table 16).

Table 28 CCM_DB_GINT2 Register

CCM_SCRATCH1 – SCRATCHPAD 1 REGISTER

The CCM_SCRATCH1 registers is a general-purpose scratchpad register. Note that the default value (the value of this register at reset) may vary, depending on the applicable boot process.

CCM_SCRATCH1 SCRATCHPAD 1 REGISTER																															
Address = 0xF000_0050																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	SCRATCH1	RW	0x0FED_0003	Scratchpad 1

Table 29 CCM_SCRATCH1 Register

CCM_SCRATCH2 – SCRATCHPAD 2 REGISTER

The CCM_SCRATCH2 registers is a general-purpose scratchpad register. Note that the default value (the value of this register at reset) may vary, depending on the applicable boot process.

CCM_SCRATCH2 SCRATCHPAD 2 REGISTER																															
Address = 0xF000_0054																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	SCRATCH2	RW	0x0000_0000	Scratchpad 2

Table 30 CCM_SCRATCH2 Register

CCM_SCRATCH3 – SCRATCHPAD 3 REGISTER

The CCM_SCRATCH3 registers is a general-purpose scratchpad register. Note that the default value (the value of this register at reset) may vary, depending on the applicable boot process.

CCM_SCRATCH3 SCRATCHPAD 3 REGISTER																															
Address = 0xF000_0058																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	SCRATCH3	RW	0x0000_0000	Scratchpad 3

Table 31 CCM_SCRATCH3 Register

CCM_SCRATCH4 – SCRATCHPAD 4 REGISTER

The CCM_SCRATCH4 registers is a general-purpose scratchpad register. Note that the default value (the value of this register at reset) may vary, depending on the applicable boot process.

CCM_SCRATCH4 SCRATCHPAD 4 REGISTER																															
Address = 0xF000_005C																	Default value = 0x0000_0000														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	SCRATCH4	RW	0x0000_0000	Scratchpad 4

Table 32 CCM_SCRATCH4 Register

CCM_IOCTRL1 – I/O CONTROL 1 REGISTER

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

CCM_IOCTRL1 I/O CONTROL 1 REGISTER																															
Address = 0xF000_0060																	Default value = 0x7777_0000														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	Reserved			
30	RESET_DS	RW	0x1	$\overline{\text{RESET}}$ drive strength 0 = Reduced strength 1 = Full strength Note - $\overline{\text{RESET}}$ is input only; this bit has no effect
29	RESET_PU	RW	0x1	$\overline{\text{RESET}}$ pull-up control 0 = Disabled 1 = Enabled
28	RESET_IE	RW	0x1	$\overline{\text{RESET}}$ input enable 0 = Disabled 1 = Enabled Note - $\overline{\text{RESET}}$ is always enabled for input; this bit has no effect
27	Reserved		0x0	
26	STANDBY_DS	RW	0x1	$\overline{\text{STANDBY}}$ drive strength 0 = Reduced strength 1 = Full strength Note - $\overline{\text{STANDBY}}$ is input only; this bit has no effect
25	STANDBY_PU	RW	0x1	$\overline{\text{STANDBY}}$ pull-up control 0 = Disabled 1 = Enabled
24	STANDBY_IE	RW	0x1	$\overline{\text{STANDBY}}$ input enable 0 = Disabled 1 = Enabled Note - $\overline{\text{STANDBY}}$ is always enabled for input; this bit has no effect
23	CLKOUT_OE	RW	0x0	CLKOUT/GPIO28 output enable 0 = Disabled (tri-state) 1 = Enabled

CCM_IOCTRL1 I/O CONTROL 1 REGISTER																															
Address = 0xF000_0060																**Default value = 0x7777_0000**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
22	CLKOUT_DS	RW	0x1	CLKOUT/GPIO28 output drive strength 0 = Reduced strength 1 = Full strength
21	CLKOUT_PD	RW	0x1	CLKOUT/GPIO28 pull-down control 0 = Disabled 1 = Enabled
20	CLKOUT_IE	RW	0x1	CLKOUT/GPIO28 input enable 0 = Disabled 1 = Enabled
19	Reserved		0x0	
18	IRQ_DS	RW	0x1	$\overline{\text{IRQ}}$ output drive strength 0 = Reduced strength 1 = Full strength
17	IRQ_PU	RW	0x1	$\overline{\text{IRQ}}$ pull-up control 0 = Disabled 1 = Enabled
16	IRQ_IE	RW	0x1	$\overline{\text{IRQ}}$ input enable 0 = Disabled 1 = Enabled Note - $\overline{\text{IRQ}}$ is output only; this bit has no effect
15:0	Reserved		0x0000	

Table 33 CCM_IOCTRL1 Register

CCM_IOCTRL2 – I/O CONTROL 2 REGISTER

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

CCM_IOCTRL2 I/O CONTROL 2 REGISTER																															
Address = 0xF000_0064																**Default value = 0xFF7F_FF7F**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	AIF1BCLK_OE	RW	0x1	AIF1BCLK output enable 0 = Disabled (tri-state) 1 = Enabled Valid for Bypass Mode B2 only. Must be enabled in Bypass Mode B2. Note that in AIF Master mode, this bit has no effect.
30	AIF1BCLK_DS	RW	0x1	AIF1BCLK output drive strength 0 = Reduced strength 1 = Full strength
29	AIF1BCLK_PD	RW	0x1	AIF1BCLK pull-down control 0 = Disabled 1 = Enabled

CCM_IOCTRL2
I/O CONTROL 2 REGISTER

Address = 0xF000_0064 | **Default value = 0xFF7F_FF7F**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
28	AIF1BCLK_IE	RW	0x1	AIF1BCLK input enable 0 = Disabled 1 = Enabled Must be enabled in AIF1 Master or AIF1 Slave modes. (In AIF Master mode, this enables the BCLK as an input to the LRCLK generator.)
27	AIF1LRCLK_OE	RW	0x1	AIF1LRCLK output enable 0 = Disabled (tri-state) 1 = Enabled Valid for Bypass Mode B2 only. Must be enabled in Bypass Mode B2. Note that in AIF Master mode, this bit has no effect.
26	AIF1LRCLK_DS	RW	0x1	AIF1LRCLK output drive strength 0 = Reduced strength 1 = Full strength
25	AIF1LRCLK_PD	RW	0x1	AIF1LRCLK pull-down control 0 = Disabled 1 = Enabled
24	AIF1LRCLK_IE	RW	0x1	AIF1LRCLK input enable 0 = Disabled 1 = Enabled Must be enabled in AIF1 Master or AIF1 Slave modes. (In AIF Master mode, this enables the LRCLK as an input to the TXDAT generator.)
23	Reserved		0x0	
22	AIF1RXDAT_DS	RW	0x1	AIF1RXDAT drive strength 0 = Reduced strength 1 = Full strength Note - AIF1RXDAT is input only; this bit has no effect
21	AIF1RXDAT_PD	RW	0x1	AIF1RXDAT pull-down control 0 = Disabled 1 = Enabled
20	AIF1RXDAT_IE	RW	0x1	AIF1RXDAT input enable 0 = Disabled 1 = Enabled
19	AIF1TXDAT_OE	RW	0x1	AIF1TXDAT output enable 0 = Disabled (tri-state) 1 = Enabled
18	AIF1TXDAT_DS	RW	0x1	AIF1TXDAT output drive strength 0 = Reduced strength 1 = Full strength
17	AIF1TXDAT_PD	RW	0x1	AIF1TXDAT pull-down control 0 = Disabled 1 = Enabled
16	AIF1TXDAT_IE	RW	0x1	AIF1TXDAT input enable 0 = Disabled 1 = Enabled Note - AIF1TXDAT is output only; this bit has no effect
15	AIF2BCLK_OE	RW	0x1	AIF2BCLK output enable 0 = Disabled (tri-state) 1 = Enabled Valid for Bypass Mode B1 only. Must be enabled in Bypass Mode B1. Note that in AIF Master mode, this bit has no effect.

CCM_IOCTRL2
I/O CONTROL 2 REGISTER

Address = 0xF000_0064 | **Default value = 0xFF7F_FF7F**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
14	AIF2BCLK_DS	RW	0x1	AIF2BCLK output drive strength 0 = Reduced strength 1 = Full strength
13	AIF2BCLK_PD	RW	0x1	AIF2BCLK pull-down control 0 = Disabled 1 = Enabled
12	AIF2BCLK_IE	RW	0x1	AIF2BCLK input enable 0 = Disabled 1 = Enabled Must be enabled in AIF2 Master or AIF2 Slave modes. (In AIF Master mode, this enables the BCLK as an input to the LRCLK generator.)
11	AIF2LRCLK_OE	RW	0x1	AIF2LRCLK output enable 0 = Disabled (tri-state) 1 = Enabled Valid for Bypass Mode B1 only. Must be enabled in Bypass Mode B1. Note that in AIF Master mode, this bit has no effect.
10	AIF2LRCLK_DS	RW	0x1	AIF2LRCLK output drive strength 0 = Reduced strength 1 = Full strength
9	AIF2LRCLK_PD	RW	0x1	AIF2LRCLK pull-down control 0 = Disabled 1 = Enabled
8	AIF2LRCLK_IE	RW	0x1	AIF2LRCLK input enable 0 = Disabled 1 = Enabled Must be enabled in AIF2 Master or AIF2 Slave modes. (In AIF Master mode, this enables the LRCLK as an input to the TXDAT generator.)
7	Reserved		0x0	
6	AIF2RXDAT_DS	RW	0x1	AIF2RXDAT drive strength 0 = Reduced strength 1 = Full strength Note - AIF2RXDAT is input only; this bit has no effect
5	AIF2RXDAT_PD	RW	0x1	AIF2RXDAT pull-down control 0 = Disabled 1 = Enabled
4	AIF2RXDAT_IE	RW	0x1	AIF2RXDAT input enable 0 = Disabled 1 = Enabled
3	AIF2TXDAT_OE	RW	0x1	AIF2TXDAT output enable 0 = Disabled (tri-state) 1 = Enabled
2	AIF2TXDAT_DS	RW	0x1	AIF2TXDAT output drive strength 0 = Reduced strength 1 = Full strength
1	AIF2TXDAT_PD	RW	0x1	AIF2TXDAT pull-down control 0 = Disabled 1 = Enabled

CCM_IOCTRL2 I/O CONTROL 2 REGISTER																															
Address = 0xF000_0064																												**Default value = 0xFF7F_FF7F**			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
0	AIF2TXDAT_IE	RW	0x1	AIF2TXDAT input enable 0 = Disabled 1 = Enabled Note - AIF2TXDAT is output only; this bit has no effect

Table 34 CCM_IOCTRL2 Register

CCM_IOCTRL3 – I/O CONTROL 3 REGISTER

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

CCM_IOCTRL3 I/O CONTROL 3 REGISTER																															
Address = 0xF000_0068																												**Default value = 0x7777_7777**			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	Reserved		0x0	
30	AIF3BCLK_DS	RW	0x1	AIF3BCLK output drive strength 0 = Reduced strength 1 = Full strength
29	AIF3BCLK_PD	RW	0x1	AIF3BCLK pull-down control 0 = Disabled 1 = Enabled
28	AIF3BCLK_IE	RW	0x1	AIF3BCLK input enable 0 = Disabled 1 = Enabled Must be enabled in AIF3 Master or AIF3 Slave modes. (In AIF Master mode, this enables the BCLK as an input to the LRCLK generator.)
27	Reserved		0x0	
26	AIF3LRCLK_DS	RW	0x1	AIF3LRCLK output drive strength 0 = Reduced strength 1 = Full strength
25	AIF3LRCLK_PU	RW	0x1	AIF3LRCLK pull-up control 0 = Disabled 1 = Enabled
24	AIF3LRCLK_IE	RW	0x1	AIF3LRCLK input enable 0 = Disabled 1 = Enabled Must be enabled in AIF3 Master or AIF3 Slave modes. (In AIF Master mode, this enables the LRCLK as an input to the TXDAT generator.)
23	Reserved		0x0	
22	AIF3RXDAT_DS	RW	0x1	AIF3RXDAT drive strength 0 = Reduced strength 1 = Full strength Note - AIF3RXDAT is input only; this bit has no effect

CCM_IOCTRL3
I/O CONTROL 3 REGISTER

Address = 0xF000_0068 | **Default value = 0x7777_7777**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
21	AIF3RXDAT_PD	RW	0x1	AIF3RXDAT pull-down control 0 = Disabled 1 = Enabled
20	AIF3RXDAT_IE	RW	0x1	AIF3RXDAT input enable 0 = Disabled 1 = Enabled
19	Reserved		0x0	
18	AIF3TXDAT_DS	RW	0x1	AIF3TXDAT output drive strength 0 = Reduced strength 1 = Full strength
17	AIF3TXDAT_PD	RW	0x1	AIF3TXDAT pull-down control 0 = Disabled 1 = Enabled
16	AIF3TXDAT_IE	RW	0x1	AIF3TXDAT input enable 0 = Disabled 1 = Enabled Note - AIF3TXDAT is output only; this bit has no effect
15	Reserved		0x0	
14	SCLK2_DS	RW	0x1	SCLK2 output drive strength 0 = Reduced strength 1 = Full strength
13	SCLK2_PD	RW	0x1	SCLK2 pull-down control 0 = Disabled 1 = Enabled
12	SCLK2_IE	RW	0x1	SCLK2 input enable 0 = Disabled 1 = Enabled Note - SCLK2 is output only; this bit has no effect
11	Reserved		0x0	
10	SDA2_DS	RW	0x1	SDA2 output drive strength 0 = Reduced strength 1 = Full strength
9	SDA2_PD	RW	0x1	SDA2 pull-down control 0 = Disabled 1 = Enabled
8	SDA2_IE	RW	0x1	SDA2 input enable 0 = Disabled 1 = Enabled
7	Reserved		0x0	
6	SCLK1_DS	RW	0x1	SCLK1 drive strength 0 = Reduced strength 1 = Full strength Note - SCLK1 is input only; this bit has no effect
5	SCLK1_PD	RW	0x1	SCLK1 pull-down control 0 = Disabled 1 = Enabled
4	SCLK1_IE	RW	0x1	SCLK1 input enable 0 = Disabled 1 = Enabled

CCM_IOCTRL3 I/O CONTROL 3 REGISTER																															
Address = 0xF000_0068																Default value = 0x7777_7777															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
3	Reserved		0x0	
2	SDA1_DS	RW	0x1	SDA1 output drive strength 0 = Reduced strength 1 = Full strength
1	SDA1_PD	RW	0x1	SDA1 pull-down control 0 = Disabled 1 = Enabled
0	SDA1_IE	RW	0x1	SDA1 input enable 0 = Disabled 1 = Enabled

Table 35 CCM_IOCTRL3 Register

CCM_IOCTRL4 – I/O CONTROL 4 REGISTER

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

CCM_IOCTRL4 I/O CONTROL 4 REGISTER																															
Address = 0xF000_006C																Default value = 0x7777_7700															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	Reserved		0x0	
30	SPISCLK_DS	RW	0x1	SPISCLK output drive strength 0 = Reduced strength 1 = Full strength
29	SPISCLK_PD	RW	0x1	SPISCLK pull-down control 0 = Disabled 1 = Enabled
28	SPISCLK_IE	RW	0x1	SPISCLK input enable 0 = Disabled 1 = Enabled
27	Reserved		0x0	
26	SPIMOSI_DS	RW	0x1	SPIMOSI/GPIO18 output drive strength 0 = Reduced strength 1 = Full strength
25	SPIMOSI_PD	RW	0x1	SPIMOSI/GPIO18 pull-down control 0 = Disabled 1 = Enabled
24	SPIMOSI_IE	RW	0x1	SPIMOSI/GPIO18 input enable 0 = Disabled 1 = Enabled
23	Reserved		0x0	
22	SPIMISO_DS	RW	0x1	SPIMISO/GPIO19 output drive strength 0 = Reduced strength 1 = Full strength

CCM_IOCTRL4 I/O CONTROL 4 REGISTER																															
Address = 0xF000_006C																**Default value = 0x7777_7700**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
21	SPIMISO_PD	RW	0x1	SPIMISO/GPIO19 pull-down control 0 = Disabled 1 = Enabled
20	SPIMISO_IE	RW	0x1	SPIMISO/GPIO19 input enable 0 = Disabled 1 = Enabled
19	Reserved		0x0	
18	SPISS_DS	RW	0x1	$\overline{\text{SPISS}}$/GPIO17 output drive strength 0 = Reduced strength 1 = Full strength
17	SPISS_PU	RW	0x1	$\overline{\text{SPISS}}$/GPIO17 pull-up control 0 = Disabled 1 = Enabled
16	SPISS_IE	RW	0x1	$\overline{\text{SPISS}}$/GPIO17 input enable 0 = Disabled 1 = Enabled
15:0	Reserved		0x7700	

Table 36 CCM_IOCTRL4 Register

CCM_IOCTRL5 – I/O CONTROL 5 REGISTER

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

CCM_IOCTRL5 I/O CONTROL 5 REGISTER																															
Address = 0xF000_0070																**Default value = 0x7D77_7777**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	Reserved		0x0	
30	UARTRX_DS	RW	0x1	UARTRX/GPIO22 output drive strength 0 = Reduced strength 1 = Full strength
29	UARTRX_PD	RW	0x1	UARTRX/GPIO22 pull-down control 0 = Disabled 1 = Enabled
28	UARTRX_IE	RW	0x1	UARTRX/GPIO22 input enable 0 = Disabled 1 = Enabled
27	UARTTX_OE	RW	0x1	UARTTX/GPIO23 output enable 0 = Disabled (tri-state) 1 = Enabled
26	UARTTX_DS	RW	0x1	UARTTX/GPIO23 output drive strength 0 = Reduced strength 1 = Full strength

CCM_IOCTRL5 I/O CONTROL 5 REGISTER																															
Address = 0xF000_0070																														**Default value = 0x7D77_7777**	
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
25	UARTTX_PD	RW	0x0	UARTTX/GPIO23 pull-down control 0 = Disabled 1 = Enabled
24	UARTTX_IE	RW	0x1	UARTTX/GPIO23 input enable 0 = Disabled 1 = Enabled
23:0	Reserved		0x77_7777	

Table 37 CCM_IOCTRL5 Register

CCM_IOCTRL6 – I/O CONTROL 6 REGISTER

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

CCM_IOCTRL6 I/O CONTROL 6 REGISTER																															
Address = 0xF000_0074																														**Default value = 0x7777_7770**	
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	TDEBUG_OE	RW	0x0	TDEBUG/TMSDEBUG output enable 0 = Disabled (tri-state) 1 = Enabled
30	TDEBUG_DS	RW	0x1	TDEBUG/TMSDEBUG output drive strength 0 = Reduced strength 1 = Full strength
29	TDEBUG_PU	RW	0x1	TDEBUG/TMSDEBUG pull-up control 0 = Disabled 1 = Enabled
28	TDEBUG_IE	RW	0x1	TDEBUG/TMSDEBUG input enable 0 = Disabled 1 = Enabled
27	Reserved		0x0	
26	TOCDRST_DS	RW	0x1	$\overline{\text{TOCDRST}}$ drive strength 0 = Reduced strength 1 = Full strength Note - $\overline{\text{TOCDRST}}$ is input only; this bit has no effect
25	TOCDRST_PU	RW	0x1	$\overline{\text{TOCDRST}}$ pull-up control 0 = Disabled 1 = Enabled
24	TOCDRST_IE	RW	0x1	$\overline{\text{TOCDRST}}$ input enable 0 = Disabled 1 = Enabled
23	Reserved		0x0	

CCM_IOCTRL6
I/O CONTROL 6 REGISTER

Address = 0xF000_0074 | **Default value = 0x7777_7770**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
22	TCK_DS	RW	0x1	TCK drive strength 0 = Reduced strength 1 = Full strength Note - TCK is input only; this bit has no effect
21	TCK_PU	RW	0x1	TCK pull-up control 0 = Disabled 1 = Enabled
20	TCK_IE	RW	0x1	TCK input enable 0 = Disabled 1 = Enabled Note - TCK is always enabled for input; this bit has no effect
19	Reserved		0x0	
18	TMSDFT_DS	RW	0x1	TMSDFT drive strength 0 = Reduced strength 1 = Full strength Note - TMSDFT is input only; this bit has no effect
17	TMSDFT_PU	RW	0x1	TMSDFT pull-up control 0 = Disabled 1 = Enabled
16	TMSDFT_IE	RW	0x1	TMSDFT input enable 0 = Disabled 1 = Enabled Note - TMSDFT is always enabled for input; this bit has no effect
15	Reserved		0x0	
14	TDI_DS	RW	0x1	TDI drive strength 0 = Reduced strength 1 = Full strength Note - TDI is input only; this bit has no effect
13	TDI_PU	RW	0x1	TDI pull-up control 0 = Disabled 1 = Enabled
12	TDI_IE	RW	0x1	TDI input enable 0 = Disabled 1 = Enabled Note - TDI is always enabled for input; this bit has no effect
11	Reserved		0x0	
10	TRST_DS	RW	0x1	$\overline{\text{TRST}}$ drive strength 0 = Reduced strength 1 = Full strength Note - $\overline{\text{TRST}}$ is input only; this bit has no effect
9	TRST_PD	RW	0x1	$\overline{\text{TRST}}$ pull-down control 0 = Disabled 1 = Enabled
8	TRST_IE	RW	0x1	$\overline{\text{TRST}}$ input enable 0 = Disabled 1 = Enabled Note - $\overline{\text{TRST}}$ is always enabled for input; this bit has no effect
7	Reserved		0x0	

CCM_IOCTRL6 I/O CONTROL 6 REGISTER																															
Address = 0xF000_0074																Default value = 0x7777_7770															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
6	TDO_DS	RW	0x1	TDO output drive strength 0 = Reduced strength 1 = Full strength
5	TDO_PD	RW	0x1	TDO pull-down control 0 = Disabled 1 = Enabled
4	TDO_IE	RW	0x1	TDO input enable 0 = Disabled 1 = Enabled Note - TDO is output only; this bit has no effect
3:0	Reserved		0x0	Must be set to 0x0 for normal operation

Table 38 CCM_IOCTRL6 Register

CCM_IOCTRL7 – I/O CONTROL 7 REGISTER

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

CCM_IOCTRL7 I/O CONTROL 7 REGISTER																															
Address = 0xF000_0078																Default value = 0x0707_0707															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	Reserved		0x0	
28	GPIO8_OP_CFG	RW	0x0	GPIO8 output configuration 0 = CMOS 1 = Open Drain
27	GPIO8_PULL_DIR	RW	0x0	GPIO8 pull-up/pull-down select 0 = Pull-down 1 = Pull-up
26	GPIO8_DS	RW	0x1	GPIO8 output drive strength 0 = Reduced strength 1 = Full strength
25	GPIO8_PULL_ENA	RW	0x1	GPIO8 pull-up/pull-down enable 0 = Disabled 1 = Enabled
24	GPIO8_IE	RW	0x1	GPIO8 input enable 0 = Disabled 1 = Enabled
23:21	Reserved		0x0	
20	GPIO7_OP_CFG	RW	0x0	GPIO7 output configuration 0 = CMOS 1 = Open Drain
19	GPIO7_PULL_DIR	RW	0x0	GPIO7 pull-up/pull-down select 0 = Pull-down 1 = Pull-up

CCM_IOCTRL7 I/O CONTROL 7 REGISTER																															
Address = 0xF000_0078																Default value = 0x0707_0707															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
18	GPIO7_DS	RW	0x1	GPIO7 output drive strength 0 = Reduced strength 1 = Full strength
17	GPIO7_PULL_ENA	RW	0x1	GPIO7 pull-up/pull-down enable 0 = Disabled 1 = Enabled
16	GPIO7_IE	RW	0x1	GPIO7 input enable 0 = Disabled 1 = Enabled
15:13	Reserved		0x0	
12	GPIO6_OP_CFG	RW	0x0	GPIO6 output configuration 0 = CMOS 1 = Open Drain
11	GPIO6_PULL_DIR	RW	0x0	GPIO6 pull-up/pull-down select 0 = Pull-down 1 = Pull-up
10	GPIO6_DS	RW	0x1	GPIO6 output drive strength 0 = Reduced strength 1 = Full strength
9	GPIO6_PULL_ENA	RW	0x1	GPIO6 pull-up/pull-down enable 0 = Disabled 1 = Enabled
8	GPIO6_IE	RW	0x1	GPIO6 input enable 0 = Disabled 1 = Enabled
7:5	Reserved		0x0	
4	GPIO5_OP_CFG	RW	0x0	GPIO5 output configuration 0 = CMOS 1 = Open Drain
3	GPIO5_PULL_DIR	RW	0x0	GPIO5 pull-up/pull-down select 0 = Pull-down 1 = Pull-up
2	GPIO5_DS	RW	0x1	GPIO5 output drive strength 0 = Reduced strength 1 = Full strength
1	GPIO5_PULL_ENA	RW	0x1	GPIO5 pull-up/pull-down enable 0 = Disabled 1 = Enabled
0	GPIO5_IE	RW	0x1	GPIO5 input enable 0 = Disabled 1 = Enabled

Table 39 CCM_IOCTRL7 Register

CCM_IOCTRL8 – I/O CONTROL 8 REGISTER

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

CCM_IOCTRL8 I/O CONTROL 8 REGISTER																															
Address = 0xF000_007C																Default value = 0x0F07_0700															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	Reserved		0x0	
28	GPIO4_OP_CFG	RW	0x0	GPIO4 output configuration 0 = CMOS 1 = Open Drain
27	GPIO4_PULL_DIR	RW	0x1	GPIO4 pull-up/pull-down select 0 = Pull-down 1 = Pull-up
26	GPIO4_DS	RW	0x1	GPIO4 output drive strength 0 = Reduced strength 1 = Full strength
25	GPIO4_PULL_ENA	RW	0x1	GPIO4 pull-up/pull-down enable 0 = Disabled 1 = Enabled
24	GPIO4_IE	RW	0x1	GPIO4 input enable 0 = Disabled 1 = Enabled
23:14	Reserved		0x01C	
13	GPIO9_OE	RW	0x0	GPIO9 output enable 0 = Disabled (tri-state) 1 = Enabled
12	GPIO9_OP_CFG	RW	0x0	GPIO9 output configuration 0 = CMOS 1 = Open Drain
11	GPIO9_PULL_DIR	RW	0x0	GPIO9 pull-up/pull-down select 0 = Pull-down 1 = Pull-up
10	GPIO9_DS	RW	0x1	GPIO9 output drive strength 0 = Reduced strength 1 = Full strength
9	GPIO9_PULL_ENA	RW	0x1	GPIO9 pull-up/pull-down enable 0 = Disabled 1 = Enabled
8	GPIO9_IE	RW	0x1	GPIO9 input enable 0 = Disabled 1 = Enabled
7:0	Reserved		0x00	

Table 40 CCM_IOCTRL8 Register

CCM_IOCTRL9 – I/O CONTROL 9 REGISTER

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

It is recommended to configure the default value of this register only.

CCM_IOCTRL9 I/O CONTROL 9 REGISTER																															
Address = 0xF000_0080																														Default value = 0x0707_0700	
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	Reserved		0x0707_0700	

Table 41 CCM_IOCTRL9 Register

CCM_IOCTRL10 – I/O CONTROL 10 REGISTER

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

CCM_IOCTRL10 I/O CONTROL 10 REGISTER																															
Address = 0xF000_0084																														Default value = 0x0707_0707	
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	Reserved		0x0	
28	GPIO14_OP_CFG	RW	0x0	GPIO14 output configuration 0 = CMOS 1 = Open Drain
27	GPIO14_PULL_DIR	RW	0x0	GPIO14 pull-up/pull-down select 0 = Pull-down 1 = Pull-up
26	GPIO14_DS	RW	0x1	GPIO14 output drive strength 0 = Reduced strength 1 = Full strength
25	GPIO14_PULL_ENA	RW	0x1	GPIO14 pull-up/pull-down enable 0 = Disabled 1 = Enabled
24	GPIO14_IE	RW	0x1	GPIO14 input enable 0 = Disabled 1 = Enabled
23:21	Reserved		0x0	
20	GPIO13_OP_CFG	RW	0x0	GPIO13 output configuration 0 = CMOS 1 = Open Drain
19	GPIO13_PULL_DIR	RW	0x0	GPIO13 pull-up/pull-down select 0 = Pull-down 1 = Pull-up
18	GPIO13_DS	RW	0x1	GPIO13 output drive strength 0 = Reduced strength 1 = Full strength

CCM_IOCTRL10
I/O CONTROL 10 REGISTER

Address = 0xF000_0084 | **Default value = 0x0707_0707**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
17	GPIO13_PULL_ENA	RW	0x1	GPIO13 pull-up/pull-down enable 0 = Disabled 1 = Enabled
16	GPIO13_IE	RW	0x1	GPIO13 input enable 0 = Disabled 1 = Enabled
15:13	Reserved		0x0	
12	GPIO12_OP_CFG	RW	0x0	GPIO12 output configuration 0 = CMOS 1 = Open Drain
11	GPIO12_PULL_DIR	RW	0x0	GPIO12 pull-up/pull-down select 0 = Pull-down 1 = Pull-up
10	GPIO12_DS	RW	0x1	GPIO12 output drive strength 0 = Reduced strength 1 = Full strength
9	GPIO12_PULL_ENA	RW	0x1	GPIO12 pull-up/pull-down enable 0 = Disabled 1 = Enabled
8	GPIO12_IE	RW	0x1	GPIO12 input enable 0 = Disabled 1 = Enabled
7:5	Reserved		0x0	
4	GPIO11_OP_CFG	RW	0x0	GPIO11 output configuration 0 = CMOS 1 = Open Drain
3	GPIO11_PULL_DIR	RW	0x0	GPIO11 pull-up/pull-down select 0 = Pull-down 1 = Pull-up
2	GPIO11_DS	RW	0x1	GPIO11 output drive strength 0 = Reduced strength 1 = Full strength
1	GPIO11_PULL_ENA	RW	0x1	GPIO11 pull-up/pull-down enable 0 = Disabled 1 = Enabled
0	GPIO11_IE	RW	0x1	GPIO11 input enable 0 = Disabled 1 = Enabled

Table 42 CCM_IOCTRL10 Register

CCM_IOCTRL11 – I/O CONTROL 11 REGISTER

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

CCM_IOCTRL11
I/O CONTROL 11 REGISTER

Address = 0xF000_0088 | **Default value = 0x0F07_0700**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	Reserved		0x0	
28	GPIO10_OP_CFG	RW	0x0	GPIO10 output configuration 0 = CMOS 1 = Open Drain
27	GPIO10_PULL_DIR	RW	0x1	GPIO10 pull-up/pull-down select 0 = Pull-down 1 = Pull-up
26	GPIO10_DS	RW	0x1	GPIO10 output drive strength 0 = Reduced strength 1 = Full strength
25	GPIO10_PULL_ENA	RW	0x1	GPIO10 pull-up/pull-down enable 0 = Disabled 1 = Enabled
24	GPIO10_IE	RW	0x1	GPIO10 input enable 0 = Disabled 1 = Enabled
23:0	Reserved		0x07_0700	

Table 43 CCM_IOCTRL11 Register

TIMER (TMR) MODULES

TIMER 1 - BASE ADDRESS 0xF001_0000

TIMER 2 - BASE ADDRESS 0xF001_0020

TIMER 3 - BASE ADDRESS 0xF001_0040

TIMER DESCRIPTION

The WM0011 provides three timers, which count up from 0, or count down from TMR_MAX_CNT. The counters are enabled using the TMR_ENA bit, and count direction is selected using the TMR_DIR bit (see Table 49).

The number of APBCLK clock cycles per count is determined by the TMR_PRESCALE register. When TMR_PRESCALE = 00h, the module will count at the APBCLK clock rate.

The TMR_MODE bit enables a selectable external trigger to be used to start the timer count. The TMC_INC bit configures the external trigger either as a 'start' trigger or as an alternate 'clock' signal. When TMR_MODE=1 and TMR_INC=0, the count rate is controlled only by the external trigger (ie. not by APBCLK).

The TMR_1SHOT bit selects whether the Timer automatically re-starts after the 'end of count' condition has been reached.

Note that the timer clock enable bit (TMR_CLK_ENA) is on the CCM_CLK_ENA register, and the timer reset bit (TMR_SOFTRST_N) is in the CCM_SOFTRST register. Note that these signals are common to all three Timer (TMR) modules.

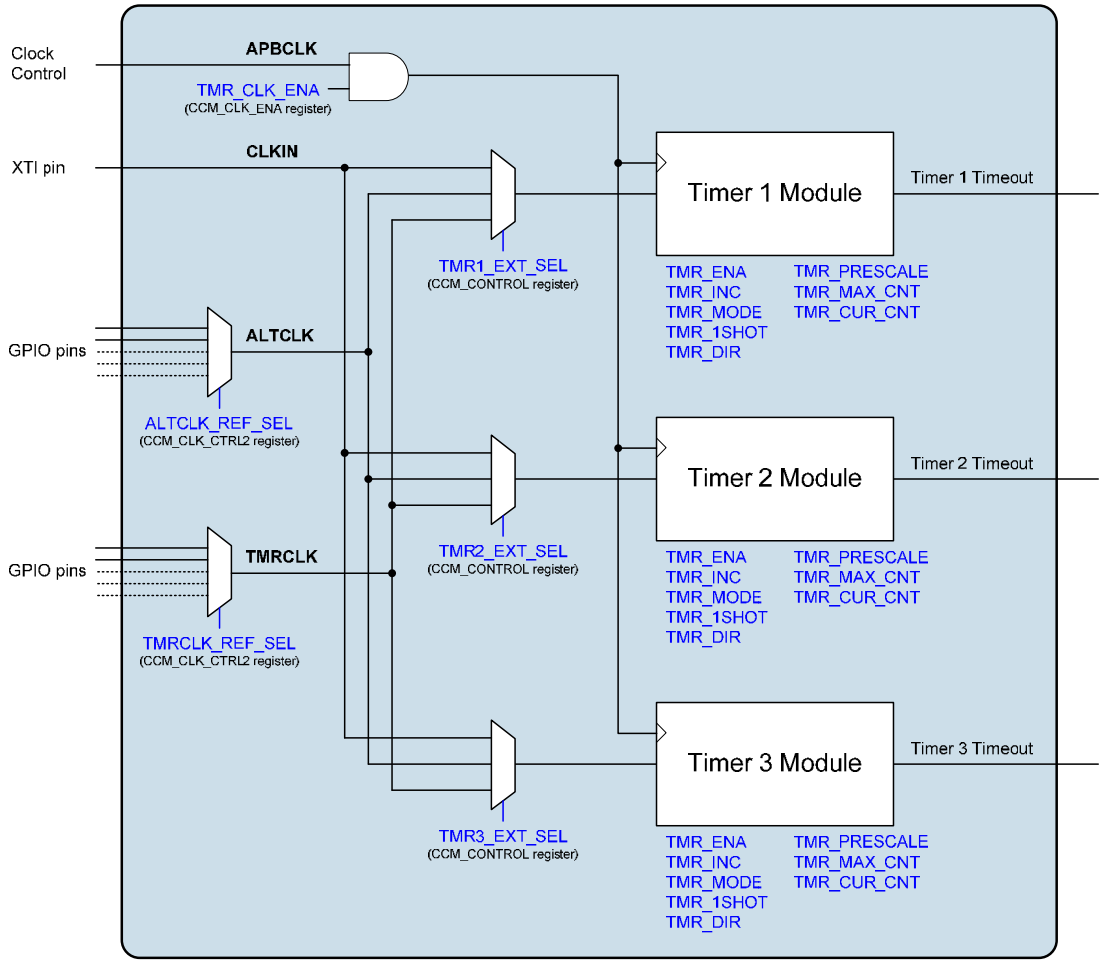
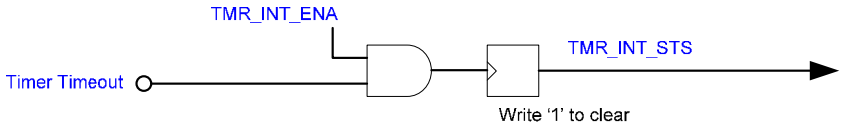


Figure 23 Timer (TMR) Modules Block Diagram

TIMER INTERRUPTS

The Timer module can generate an interrupt when the 'end of count' condition occurs.

The Timer module interrupt control registers are illustrated in Figure 24.



The interrupt control functions are replicated for each of the 3 Timer modules.

Figure 24 Timer Interrupts

TIMER REGISTER MAP

The register map of the Timer module is illustrated in Table 44.

ADDRESS	REGISTER	DESCRIPTION	RESET VALUE
Base + 0x00	TMR_PRESCALE	Timer Prescale	0x0000_0000
Base + 0x04	TMR_MAX_CNT	Timer Maximum Count	0x0000_00FF
Base + 0x0C	TMR_CUR_CNT	Timer Current Count	0x0000_0000
Base + 0x10	TMR_CTRL	Timer Control	0x0000_0000
Base + 0x14	TMR_INT_STATUS	Timer Interrupt Status	0x0000_0000

Table 44 Timer Register Definition

TMR_PRESCALE – TIMER PRESCALE REGISTER

TMR_PRESCALE
TIMER PRESCALE REGISTER

Address = 0xF001_0000 (Timer 1)
Address = 0xF001_0020 (Timer 2)
Address = 0xF001_0040 (Timer 3)

Default value = 0x0000_0000

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:8	Reserved		0x00_0000	
7:0	TMR_PRESCALE	RW	0x00	Timer Prescaler The prescaler is used as a divider to set the TMR count frequency according to the following calculation: TMR count frequency = [APBCLK] / (TMR_PRESCALE + 1) Note that, when TMR_MODE=1 and TMR_INC=0, the count rate is controlled only by the external trigger (ie. not by APBCLK). TMR_PRESCALE has no effect in this case.

Table 45 TMR_PRESCALE Register

TMR_MAX_CNT – TIMER MAXIMUM COUNT REGISTER

TMR_MAX_CNT TIMER MAXIMUM COUNT REGISTER																															
Address = 0xF001_0004 (Timer 1) **Address = 0xF001_0024 (Timer 2)** **Address = 0xF001_0044 (Timer 3)**																**Default value = 0x0000_00FF**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:24	Reserved		0x00	
23:0	TMR_MAX_CNT	RW	0x00_00FF	Timer End Value This register indicates the maximum count number for the timer. When configured as an up counter, the counter will increment from 0 to TMR_MAX_CNT value. When configured as a down counter, the counter will decrement from TMR_MAX_CNT to 0.

Table 46 TMR_MAX_CNT Register

TMR_CUR_CNT – TIMER CURRENT COUNT REGISTER

TMR_CUR_CNT TIMER CURRENT COUNT REGISTER																															
Address = 0xF001_000C (Timer 1) **Address = 0xF001_002C (Timer 2)** **Address = 0xF001_004C (Timer 3)**																**Default value = 0x0000_0000**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:24	Reserved		0x00	
23:0	TMR_CUR_CNT	RO	0x00_0000	Timer Current Count Value The value of the current counter. This value represents the value of the counter with one APBCLK delay.

Table 47 TMR_CUR_CNT Register

TMR_CTRL – TIMER CONTROL REGISTER

This Timer Control register provides control of the timing function.

When using the down-counter (TMR_DIR = 1), it is important to set the register bits in the sequence described in Table 48 in order to guarantee correct operation:

The counter value (TMR_CUR_CNT described in Table 47) must be set to a non-zero value before asserting the timer interrupt bit (TMR_INT_ENA). If TMR_CUR_CNT = 0 at the point that TMR_INT_ENA is asserted, an interrupt event will be generated immediately as the interrupt detection logic interprets the zero value as 'end of down-count sequence'. The correct sequence of events is summarised below in Table 48:

STEP	REGISTER SETTINGS	DESCRIPTION
Step 1	TMR_DIR = 1	Set the timer direction to 'down'
	TMR_ENA = 0	Disable the counter, and load the initial count value into TMR_CUR_CNT
Step 2	TMR_INT_ENA = 1	Enable the timer interrupt. Note that, as TMR_CUR_CNT has already been set to a non-zero value in Step 1, an interrupt event will not be generated immediately.
Step 3	TMR_ENA = 1	Enable the counter and start counting.

Table 48 Setting the Downtimer

TMR_CTRL
TIMER CONTROL REGISTER

Address = 0xF001_0010 (Timer 1)
Address = 0xF001_0030 (Timer 2)
Address = 0xF001_0050 (Timer 3)

Default value = 0x0000_0000

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:7	Reserved		0x000_0000	
6	TMR_ENA	RW	0x0	Timer Enable 0 = Count disabled and initial count reloaded 1 = Count enabled
5	TMR_INC	RW	0x0	Timer Increment control Only valid when TMR_MODE=1. 0 = Timer increments by one count at every rising signal edge of the external trigger. 1 = Timer starts incrementing on the first rising signal edge of the external trigger. (Count rate is set by APBCLK and the TMR_PRESCALE register.) The external trigger is selected using the TMRn_EXT_SEL bit in the CCM_CONTROL register (see Table 16).
4	TMR_MODE	RW	0x0	Timer Mode select 0 = Timer starts counting when TMR_ENA=1. 1 = Timer behaviour is controlled using TMR_INC. An external trigger signal is enabled as a control input to the Timer.
3	Reserved		0x0	
2	TMR_1SHOT	RW	0x0	Timer One-Shot Mode select 0 = Disabled (Timer automatically re-starts when the 'end of count' condition is reached). 1 = Enabled (Timer stops when the 'end of count' condition is reached).

TMR_CTRL TIMER CONTROL REGISTER	
Address = 0xF001_0010 (Timer 1) **Address = 0xF001_0030 (Timer 2)** **Address = 0xF001_0050 (Timer 3)**	**Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
1	TMR_DIR	RW	0x0	Timer Count Direction 0 = Count Up 1 = Count Down When set to 1, the counter counts down from TMR_CUR_CNT (see Table 47) to 0. When set to 0, the counter counts up from 0.
0	TMR_INT_ENA	RW	0x0	Timer Interrupt Enable 0 = Disabled 1 = Enabled When set, this bit allows registration of an interrupt event when the 'end of count' condition is reached. When set to 0, neither the TMR_INT_STS bit, nor the Timer Interrupt signal will assert.

Table 49 TMR_CTRL Register

TMR_INT_STATUS – TIMER INTERRUPT STATUS REGISTER

TMR_INT_STATUS TIMER INTERRUPT STATUS REGISTER	
Address = 0xF001_0014 (Timer 1) **Address = 0xF001_0034 (Timer 2)** **Address = 0xF001_0054 (Timer 3)**	**Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:1	Reserved		0x0000_0000	
0	TMR_INT_STS	R/W1C	0x0	Timer Interrupt Status 0 = Timer Interrupt is de-asserted 1 = Timer Interrupts is asserted Write 1 to clear.

Table 50 TMR_INT_STATUS Register

I²C INTERFACE MODULE

BASE ADDRESS 0xF002_0000

I2C FEATURES

The I²C Controller Module is an APB peripheral with two independent I²C buses. These are configured as one master and one slave. Note that the external pins are multiplexed such that only one of the I²C Master, the I²C slave, or the UART can be configured at any one time.

The following I²C specification features are supported by the I²C Controller Module.

I²C Master:

- Normal (100kHz) and Fast Mode (400kHz and 1MHz) operation
- Both Single and Multi-master

I²C Slave:

- Normal (100kHz) and Fast Mode (400kHz and 1MHz) operation
- Clock Stretching

I2C TRANSFERS

The I²C Controller supports Read and Write functions on the Master and Slave interfaces.

Typical protocols for these transfers are described in Figure 25 through to Figure 29. Note that only a high-level description is provided here; further details of each of the I2C control registers are provided later in this section.

Note that, in a typical implementation, the Master and Slave devices should both have prior knowledge of the number of bytes to be transferred. This is especially relevant to the Slave device in I²C Read operations, as the Slave must provide the required number of data bytes as expected by the Master.

A typical protocol for an I²C Master Write is illustrated and summarised in Figure 25.

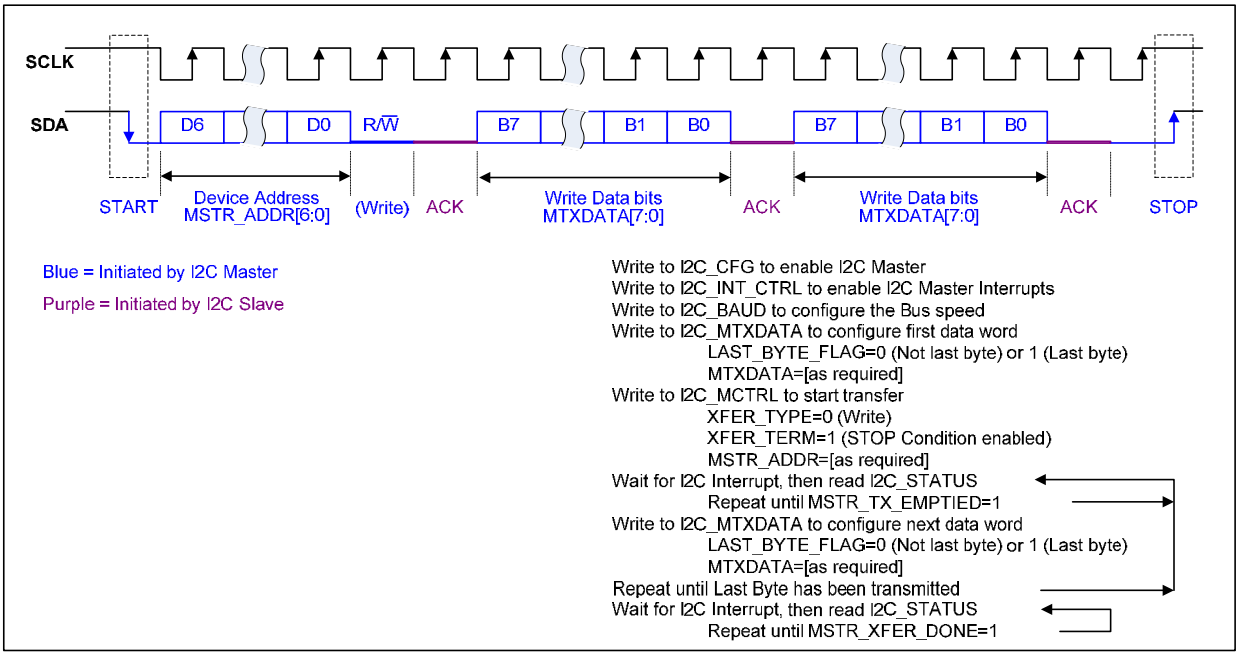


Figure 25 I2C Master Write

A typical protocol for an I²C Master Read is illustrated and summarised in Figure 26.

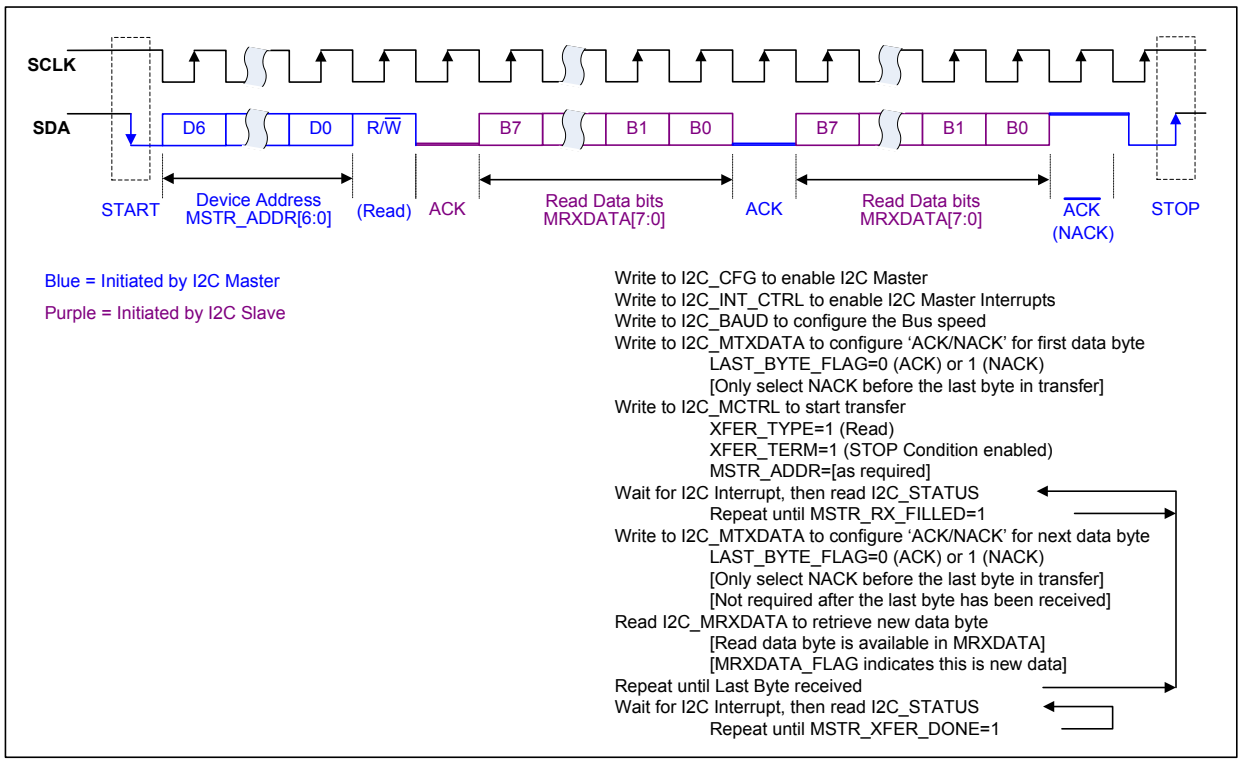


Figure 26 I2C Master Read

A typical protocol for an I²C Master Write, followed by Master Read is illustrated in Figure 27. Note that this transfer makes use of the "Repeated START Condition" before the Master Read.

The implementation of this transfer is as described above for the Write and Read actions, except for setting XFER_TERM=0 for the I²C Write - this configuration selects the "Rpt START" at the end of the I²C Write.

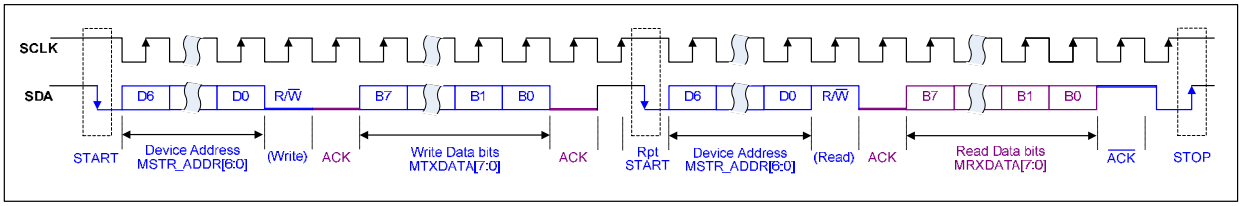


Figure 27 I2C Master Write & Read

A typical protocol for an I²C Slave Write is illustrated and summarised in Figure 28.

Note that the 'I²C Write' transfer describes a data transfer from the Master to the Slave. Accordingly, this transfer relates to Received (RX) data in the Slave module.

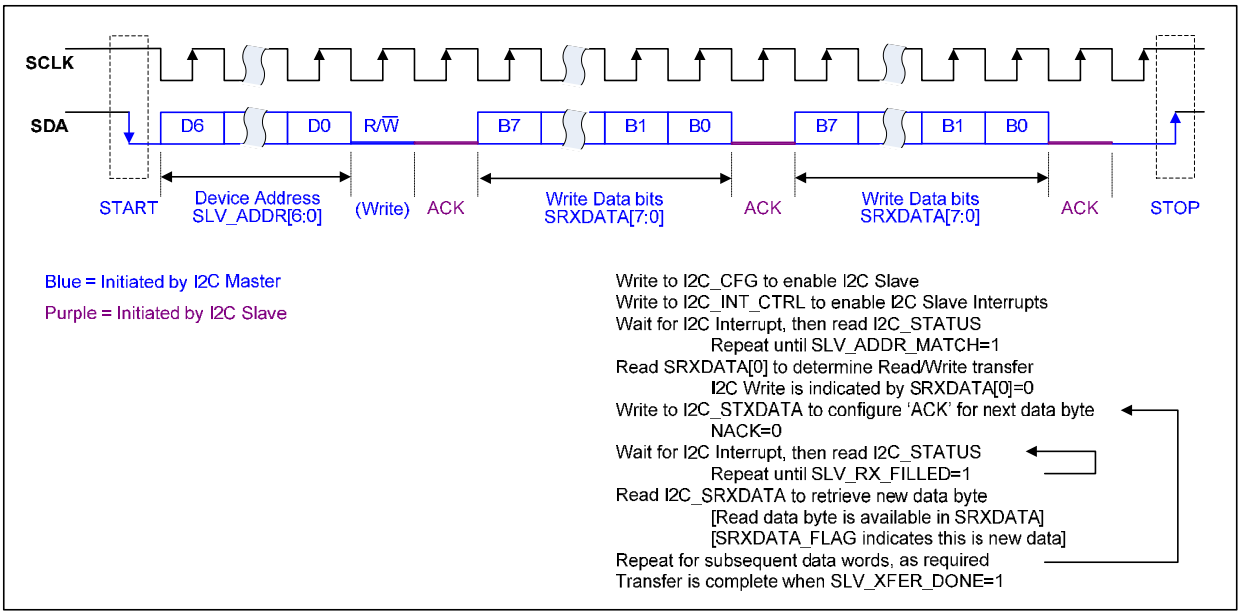


Figure 28 I2C Slave Write

A typical protocol for an I²C Slave Read is illustrated and summarised in Figure 29.

Note that the 'I²C Read' transfer describes a data transfer from the Slave to the Master. Accordingly, this transfer relates to Transmit (TX) data in the Slave module.

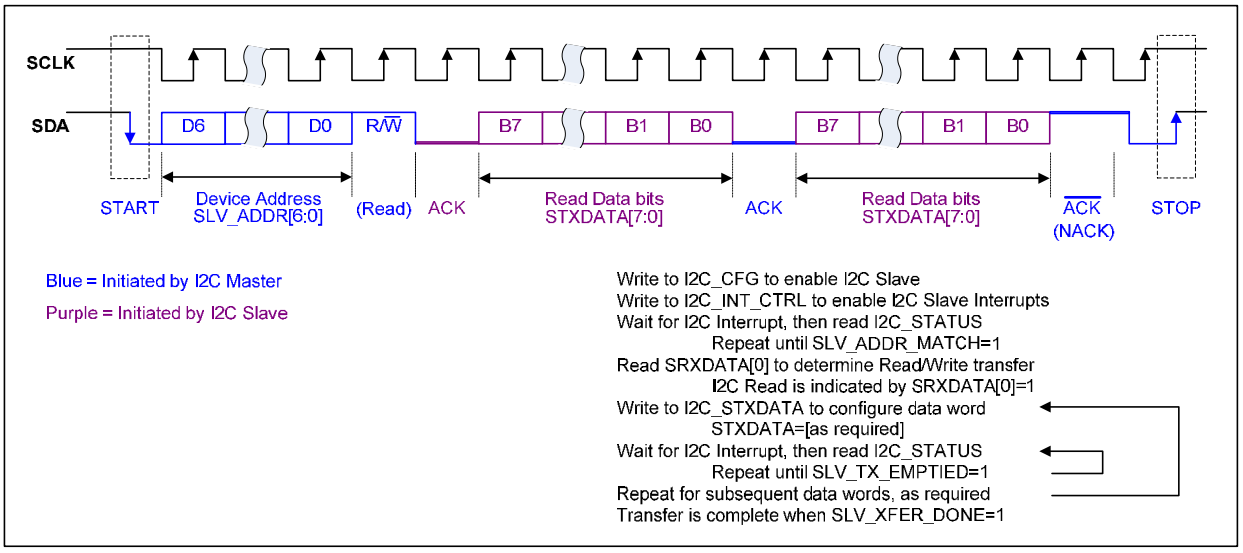


Figure 29 I2C Slave Read

I2C INTERRUPTS

The I2C module can generate an interrupt when any of the conditions described in the I2C_STATUS register occurs. The interrupt conditions provide status indications of the I2C bus transactions.

The I2C interrupt control registers are illustrated in Figure 30.

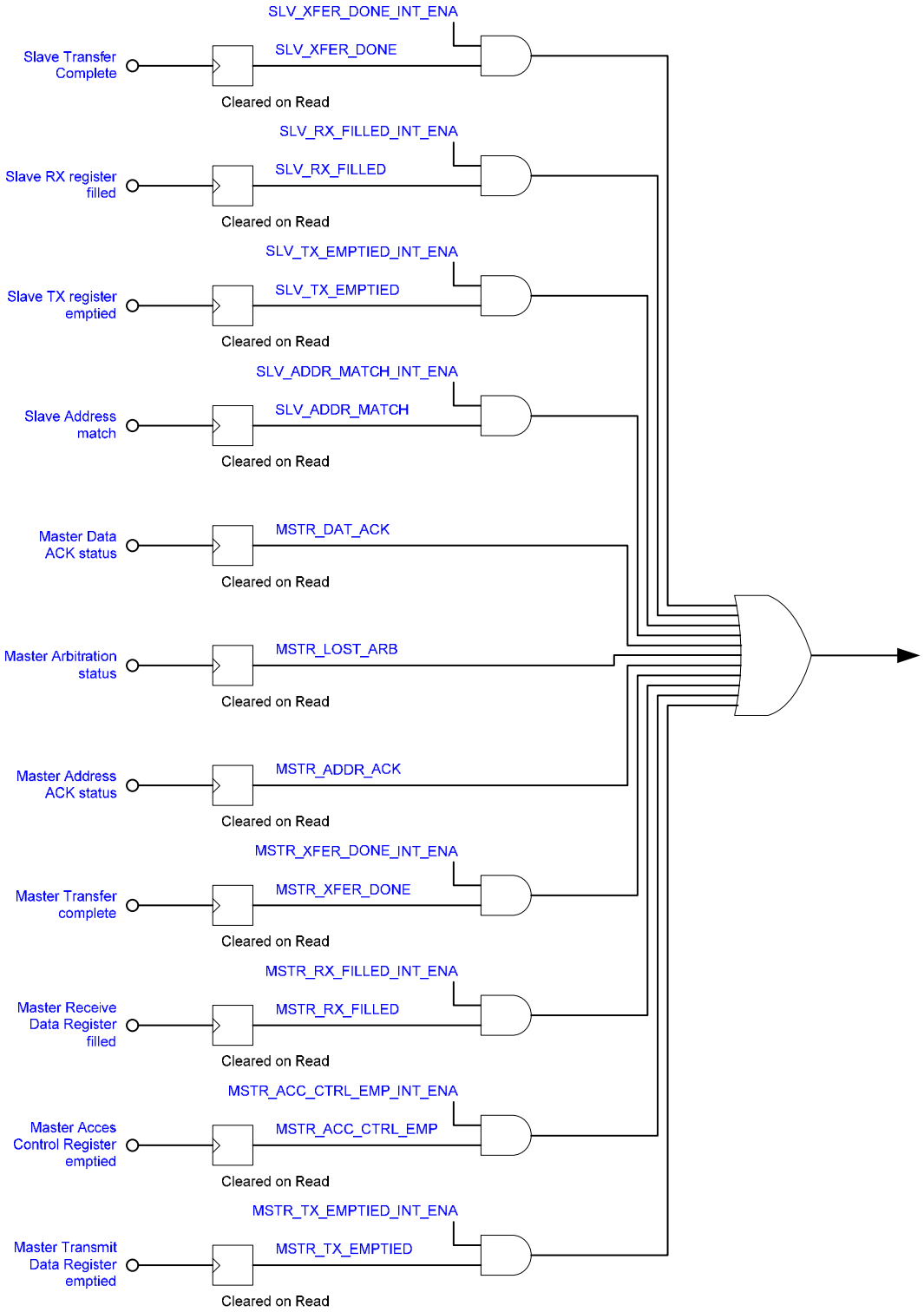


Figure 30 I2C Interrupts

I2C REGISTER MAP

The register map of the I2C module is illustrated in Table 51.

ADDRESS	REGISTER	DESCRIPTION	RESET VALUE
Base + 0x00	I2C_CFG	I2C Configuration	0x0000_0020
Base + 0x04	I2C_STATUS	I2C Status	0x0000_0000
Base + 0x08	I2C_INT_CTRL	I2C Interrupt Control	0x0000_0000
Base + 0x30	I2C_MCTRL	I2C Master Access Control	0x0000_0000
Base + 0x34	I2C_MRXDATA	I2C Master Receive Data	0x0000_0000
Base + 0x38	I2C_MTXDATA	I2C Master Transmit Data	0x0000_0000
Base + 0x40	I2C_BAUD	I2C Baud Rate	0x01EC_01EC
Base + 0x70	I2C_SRXDATA	I2C Slave Receive Data	0x0000_0000
Base + 0x74	I2C_STXDATA	I2C Slave Transmit Data	0x0000_0000
Base + 0x78	I2C_SLV_ADDR	I2C Slave Address	0x0000_0000

Table 51 I2C Register Definition

I2C_CFG – I²C CONFIGURATION REGISTER

A pulse filter is provided to remove spikes on the I²C bus input signal. The operation of the pulse filter is dependent on the frequency of the main controller clock (APBCLK), with wider pulses being filtered out on slower running clocks. Pulses shorter than 50ns are always filtered, in accordance with I²C standards. Pulses shorter than the minimum SCLK High Pulse-Width (identified as t_2 in Figure 6) are always filtered.

Note that, when setting the I2C baud rate (see Table 59), the I2C_BAUD register must take account of the Pulse Filter selection.

The I²C controller has two independent buses; these are enabled using the MSTR_ENA and SLV_ENA fields, as described in Table 52.

The external pins are multiplexed such that only one of the I²C Master, the I²C slave, or the UART can be configured at any one time. (The applicable function is selected using the PORT2_SEL field in the CCM_CONTROL register - see Table 16). The MSTR_ENA and SLV_ENA fields should be set consistent with the PORT2_SEL selection.

Note that the I²C clock enabling bit I2C_CLK_ENA is on the CCM_CLK_ENA register (see Table 23).

I2C_CFG
I2C CONFIGURATION REGISTER

Address = 0xF002_0000 | **Default value = 0x0000_0020**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:6	Reserved		0x00_0000	
5	PULSE_FILTER_DSBL	RW	0x1	Pulse filter select 0 = Pulse filter enabled 1 = Pulse filter disabled
4	Reserved		0x0	
3:2	MSTR_ENA	RW	0x0	I²C master bus-enable 00 = Disabled 01 = Reserved 10 = Reserved 11 = Enabled
1:0	SLV_ENA	RW	0x0	I²C slave bus-enable 00 = Disabled 01 = Enabled 10 = Reserved 11 = Reserved

Table 52 I2C_CFG Register

I2C_STATUS – I2C STATUS REGISTER

The active bits in this register indicate the status of most I^2C operations. All bits are cleared on read unless otherwise stated. Note that this register must be cleared (by reading) after every I^2C transfer; subsequent I^2C transfers will be inhibited if this register is not clear.

The I2C_STATUS register provides a status indication for each data byte transmitted or received via the relevant TX/RX register. Additional status bits indicate when the full I2C transfer is complete. Interrupt flags, corresponding to each of these conditions, can be enabled using the control fields in the I2C_INT_CTRL register (see Table 54).

On the Slave I2C interface, the SLV_ADDR_MATCH field indicates receipt of a Device Address that matches the 7-bit SLV_ADDR (see Table 63).

On the Master I2C interface, the Device Address contained in the MSTR_ADDR field (see Table 56) is transmitted at the start of an I2C transfer. The MSTR_ACC_CTRL_EMP field indicates that the Device Address has been transmitted.

On the Master I2C interface, the remote (Slave) device must acknowledge each byte received. (This includes the receipt of the Device Address for Read or Write operations.) The received ACK/NACK status is contained in the MSTR_ADDR_ACK and MSTR_DAT_ACK bits.

I2C_STATUS
I2C STATUS REGISTER

Address = 0xF002_0004 | **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:25	Reserved		0x00	
24	SLV_XFER_DONE	RC	0x0	Slave transfer transmission status This bit is set on successful completion of a Slave transfer. This bit is cleared when the register is read. 0 = I2C transfer not complete 1 = I2C transfer complete Only valid when I2C Slave is enabled.
23:19	Reserved		0x00	
18	SLV_RX_FILLED	RC	0x0	Slave Receive Data Register full. This bit is asserted when the Slave Receive Data Register I2C_SRXDATA (see Table 60) has been filled with data from the I^2C bus. This bit is cleared when the register is read. 0 = No new I2C data received since last status register read 1 = I2C_SRXDATA has new data since last status register read Only valid when I2C Slave is enabled.
17	SLV_TX_EMPTIED	RC	0x0	Slave Transmit Data Register empty. This bit is asserted when the Slave Transmit Data Register I2C_STXDATA (see Table 61) has been emptied by the I^2C controller, for transmission on the I^2C bus. This bit is cleared when the register is read. 0 = No new I2C data transmitted since last status register read 1 = I2C_STXDATA emptied since last status register read Only valid when I2C Slave is enabled.
16	SLV_ADDR_MATCH	RC	0x0	Slave Address Match status This bit is set on successful receipt of a Device Address which matches the SLV_ADDR field. This bit is cleared when the register is read. 0 = no match 1 = match detected Only valid when I2C Slave is enabled.
15:12	Reserved		0x0	

I2C_STATUS I2C STATUS REGISTER																															
Address = 0xF002_0004																**Default value = 0x0000_0000**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
11	MSTR_DAT_ACK	RC	0x0	Master data ACK or NACK status 0 = ACK response received 1 = ACK not received to Master Data byte Only valid when I2C Master is enabled.
10	MSTR_LOST_ARB	RC	0x0	Master arbitration error status 0 = No error 1 = Master lost arbitration Only valid when I2C Master is enabled.
9	MSTR_ADDR_ACK	RC	0x0	Master address ACK or NACK status 0 = ACK response received 1 = ACK not received to Master Device Address Only valid when I2C Master is enabled.
8	MSTR_XFER_DONE	RC	0x0	Master transfer transmission status This bit is set on successful transmission of a STOP condition, or by loss of arbitration where the current master lost the arbitration. Note that this bit is not set on transmission of a Repeated START condition. This bit is cleared when the register is read. 0 = I2C transfer not complete 1 = I2C transfer complete Only valid when I2C Master is enabled.
7:3	Reserved		0x0	
2	MSTR_RX_FILLED	RC	0x0	Master Receive Data Register full. This bit is asserted when the Master Receive Data Register I2C_MRXDATA (see Table 57) has been filled with data from the I^2C bus. This bit is cleared when the register is read. 0 = No new I2C data received since last status register read 1 = I2C_MRXDATA has new data since last status register read Only valid when I2C Master is enabled.
1	MSTR_ACC_CTRL_EMP	RC	0x0	Master Access Control Register cleared. This bit is asserted when the I2C_MCTRL register is cleared (following final ACK/NACK of the I2C transfer). This bit is cleared when the register is read. 0 = Master Access Control Register (I2C_MCTRL) not cleared since last status register read 1 = Master Access Control Register (I2C_MCTRL) cleared since last status register read Only valid when I2C Master is enabled.
0	MSTR_TX_EMPTIED	RC	0x0	Master Transmit Data Register empty. This bit is asserted when the Master Transmit Data Register I2C_MTXDATA (see Table 58) has been emptied by the I^2C controller, for transmission on the I^2C bus. This bit is cleared when the register is read. 0 = No new I2C data transmitted since last status register read 1 = I2C_MTXDATA emptied since last status register read Only valid when I2C Master is enabled.

Table 53 I2C_STATUS Register

I2C_INT_CTRL – I²C INTERRUPT CONTROL REGISTER

There are eight I²C Controller interrupts which can be enabled or disabled with the bits on this I2C_INT_CTRL register.

Note that bits [11:9] of the I2C Status register (I2C_STATUS) will always generate an I2C Interrupt; there are no enable control bits corresponding to these signals.

I2C_INT_CTRL
I2C INTERRUPT CONTROL REGISTER

Address = 0xF002_0008 **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:25	Reserved		0x00	
24	SLV_XFER_DONE_INT_ENA	RW	0x0	Slave Transfer Complete interrupt enable 0 = disabled 1 = enabled
23:19	Reserved		0x00	
18	SLV_RX_FILLED_INT_ENA	RW	0x0	Slave Receive Data Register Full interrupt enable 0 = disabled 1 = enabled
17	SLV_TX_EMPTIED_INT_ENA	RW	0x0	Slave Transmit Data Register Empty interrupt enable 0 = disabled 1 = enabled
16	SLV_ADDR_MATCH_INT_ENA	RW	0x0	Slave Address Match interrupt enable 0 = disabled 1 = enabled
15:9	Reserved		0x00	
8	MSTR_XFER_DONE_INT_ENA	RW	0x0	Master Transfer Complete interrupt enable 0 = disabled 1 = enabled
7:3	Reserved		0x00	
2	MSTR_RX_FILLED_INT_ENA	RW	0x0	Master Receive Data Register Full interrupt enable 0 = disabled 1 = enabled
1	MSTR_ACC_CTRL_EMP_INT_ENA	RW	0x0	Master Access Control Register Empty interrupt enable 0 = disabled 1 = enabled
0	MSTR_TX_EMPTIED_INT_ENA	RW	0x0	Master Transmit Data Register Empty interrupt enable 0 = disabled 1 = enabled

Table 54 I2C_INT_CTRL Register

I2C_MCTRL – I²C MASTER ACCESS CONTROL REGISTER

The Master Access Control Register configures the I2C Master module for Read or Write operations, and is used to initiate an I²C Master transfer.

Writing to the I2C_MCTRL register initiates an I2C Master transfer. The MODULE_BUSY bit indicates that the I²C transfer has been correctly configured, and that the I²C controller will initiate the transfer. Note that the I2C_STATUS register must be cleared (by reading) before initiating an I2C Master transfer.

XFER_TYPE determines whether the I²C action to be performed is a read or a write. XFER_TERM specifies whether a STOP Condition should be issued at the end of the current transfer.

The I²C address of the target Slave device is held in the MSTR_ADDR register. (This identifies the remote device that the I2C transfer is intended for.)

Note that MSTR_ADDR is the 7-bit Device Address, and does not include the Read/Write bit. The equivalent 8-bit Device Address comprises the 7 bits of MSTR_ADDR, and the R/W bit in the LSB position. This is illustrated by an example in Table 55.

I²C ACTION	MSTR_ADDR	8-BIT DEVICE ADDRESS
Write	0x38 (hex), 011 1000 (binary)	0x70 (hex), 0111 0000 (binary)
Read		0x71 (hex), 0111 0001 (binary)

Table 55 Illustration of 7-bit MSTR_ADDR compared with 8-bit Device Address

The I2C_MCTRL register is cleared (to default) after the final ACK/NACK of the I²C Master transfer. The MSTR_ACC_CTRL_EMP bit in the I2C_STATUS register (see Table 53) indicates when the I2C_MCTRL register has been cleared. A corresponding interrupt can also be enabled if required.

The I2C_MCTRL register will be cleared (and the MSTR_ACC_CTRL_EMP bit set) as described above. Note that the MSTR_XFER_DONE bit, indicating completion of the I²C Master transfer, will be set at approximately the same time if XFER_TERM=1. The MSTR_XFER_DONE bit will not be set if XFER_TERM=0 for the current transfer.

I2C_MCTRL
I2C MASTER ACCESS CONTROL REGISTER

Address = 0xF002_0030 **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:14	Reserved		0x0_0000	
13	MODULE_BUSY	RO	0x0	This bit is asserted on any write to the I2C_MCTRL register, and indicates that the hardware is initiating an I2C Master transfer 0 = Register cleared (following final ACK/NACK of the I2C transfer) 1 = Module busy (I2C transfer initiated) Note that this bit is read only
12	Reserved		0x0	
11	XFER_TYPE	RW	0x0	Selects Read or Write for I2C Master transfer 0 = Write 1 = Read
10	XFER_TERM	RW	0x0	Transfer termination Selects a STOP Condition at the end of the current transfer 0 = No STOP Condition. (Next transfer uses a Repeated START.) 1 = STOP Condition issued at end of current transfer
9:7	Reserved		0x0	
6:0	MSTR_ADDR	RW	0x00	This is the 7 bit I²C address of the target Slave device, identifying the remote device that the I2C transfer is intended for. (Note this does not include the R/W bit.)

Table 56 I2C_MCTRL Register

I2C_MRXDATA – I²C MASTER RECEIVE DATA REGISTER

The Master Receive Data Register (I2C_MRXDATA) holds the byte of data received (MRXDATA) in the I²C Master Read operation.

The MRXDATA_FLAG bit indicates when a new byte of data has been successfully received. This bit is cleared when the register is read.

Note that the MSTR_RX_FILLED bit in the I2C_STATUS register (see Table 53) also indicates when new data has been received. The associated interrupt may also be enabled.

Each received data byte must be acknowledged by the I²C Master module, using the ACK/NACK response. The response is configured by writing to the I2C_MTXDATA register (see Table 58), setting the LAST_BYTE_FLAG bit to 0 or 1 (for ACK/NACK respectively). For further details, see the following section, describing the I2C_MTXDATA register.

The last byte of the I²C Master Read operation is acknowledged with the NACK response from the I²C Master. An I²C STOP Condition will normally be transmitted after the final NACK, thus completing the I²C Read. Note that the action taken on termination of the transfer is configurable, using the XFER_TERM bit in the I2C_MCTRL register (see Table 56).

The MSTR_XFER_DONE bit in the I2C_STATUS register indicates when an I²C Master transfer has completed.

I2C_MRXDATA
I2C MASTER RECEIVE DATA REGISTER

Address = 0xF002_0034 | **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:9	Reserved		0x00_0000	
8	MRXDATA_FLAG	RC	0x0	This is asserted to indicate that the 8-bit byte of data has been received 0 = Register empty 1 = Register full (new data received) Note that this bit is cleared on read
7:0	MRXDATA	RO	0x00	Master Data byte received Note that this is a read only field

Table 57 I2C_MRXDATA Register

I2C_MTXDATA – I²C MASTER TRANSMIT DATA REGISTER

The Master Transmit Data Register (I2C_MTXDATA) holds the byte of data (MTXDATA) for transmission in the I²C Master Write operation. The I2C_MTXDATA register also controls the ACK/NACK response in I²C Master Read operations.

The MTXDATA_FLAG bit indicates when a byte of data has been loaded, ready for transmission. This bit is set to '0' after the byte has been transmitted, indicating that the register has been emptied and is ready for the next data byte to be loaded. The MTXDATA_FLAG is a Read-Only bit.

Note that the MXTR_TX_EMPTIED bit in the I2C_STATUS register (see Table 53) also indicates when the I2C_MTXDATA register has been emptied. The associated interrupt may also be enabled.

The first byte to be transmitted must be written to MTXDATA before the Master Write transfer is initiated. Each subsequent data byte for transmission should be written to MTXDATA as soon as possible after the MSTR_TX_EMPTIED bit is asserted.

For I²C Master Write operations, the LAST_BYTE_FLAG bit indicates whether the associated data byte is the last byte to be transferred. For I²C Master Read operations, the ACK/NACK response is configured by writing to the LAST_BYTE_FLAG bit.

Each transmitted word must be acknowledged by the remote (Slave) device. This includes the transmission of the Device Address for Read or Write operations. The received ACK/NACK status is provided in the I2C_STATUS register (see Table 53).

For I^2C Master Write, setting LAST_BYTE_FLAG=1 will complete the transfer after the associated data byte has been transmitted.

For I^2C Master Read, the I2C_MTXDATA register must be written to configure the ACK/NACK response for each received data word. Setting LAST_BYTE_FLAG=0 will configure the 'ACK', indicating readiness for more data bytes. Setting LAST_BYTE_FLAG=1 will configure the 'NACK', completing the transfer, and indicating that no further data is expected.

For I^2C Master Read, the ACK/NACK response for the first data byte must be written to MTXDATA before the Master Read transfer is initiated. For subsequent data bytes, the LAST_BYTE_FLAG should be written following each received data byte (MSTR_RX_FILLED=1); writing to the LAST_BYTE_FLAG bit configures the ACK/NACK status for the next word that is received.

Note that the LAST_BYTE_FLAG bit must be written before reading the received data byte from I2C_MRXDATA.

The ACK/NACK status is configured in advance of each byte received, as described above. Accordingly, there is no requirement to write to the LAST_BYTE_FLAG bit after receipt of the last byte of the transfer.

The last byte of the I^2C Master Write operation is acknowledged with the ACK response from the I^2C Slave. An I^2C STOP Condition will normally be transmitted after the final ACK, thus completing the I^2C Write. Note that the action taken on termination of the transfer is configurable, using the XFER_TERM bit in the I2C_MCTRL register (see Table 56).

The MSTR_XFER_DONE bit in the I2C_STATUS register indicates when an I^2C Master transfer has completed.

I2C_MTXDATA
I2C MASTER TRANSMIT DATA REGISTER

Address = 0xF002_0038 | **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:10	Reserved		0x00_0000	
9	LAST_BYTE_FLAG	RW	0x0	Indicates the last byte of a transfer from either an I^2C read or an I^2C write 0 = Not last byte 1 = Last Byte
8	MTXDATA_FLAG	RO	0x0	Indicates when the register has been loaded and is ready for transmission. 0 = Register empty 1 = Register full (ready for transmit) Note that this bit is read only
7:0	MTXDATA	RW	0x00	Master Data byte for transmission (I^2C write)

Table 58 I2C_MTXDATA Register

I2C_BAUD – I²C BAUD RATE REGISTER

The SCLK output frequency must be configured when using the I2C Master module.

The INP_CLK_HIGH_DIV and INP_CLK_LOW_DIV register fields set the number of APBCLK cycles in the SCLK High Phase and SCLK Low Phase respectively.

For example, if 100kHz SCLK is required, and the APBCLK frequency is 100MHz, this is a frequency ratio of 1000, ie. the total number of APBCLK cycles in the SCLK High / Low phases is 1000. Assuming 50% duty cycle, INP_CLK_HIGH_DIV and INP_CLK_LOW_DIV should each be set to 500 cycles, which is coded as 0x1F3 (note the -1 offset).

If the I2C Pulse Filter is enabled (see the I2C_CFG register, described in Table 52), then the INP_CLK_HIGH_DIV and INP_CLK_LOW_DIV should be adjusted by subtracting 7 from each.

In the above example, if the I2C Pulse Filter is enabled, then INP_CLK_HIGH_DIV and INP_CLK_LOW_DIV should each be set to 0x1EC.

The I2C_BAUD register settings must be consistent with the Signal Timing Requirements illustrated in Figure 6.

I2C_BAUD
I2C BAUD RATE REGISTER

Address = 0xF002_0040 | **Default value = 0x01EC_01EC**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:27	Reserved		0x00	
26:16	INP_CLK_HIGH_DIV	RW	0x1EC	Clock Divisor for SCLK High Phase. This field sets the number of APBCLK cycles in the SCLK (High) Phase for I2C Master Mode output. 000h = 1 clock cycle 001h = 2 clock cycles 002h = 3 clock cycles … FFFh = 4096 clock cycles Note that, when the I2C Pulse Filter is enabled, then INP_CLK_HIGH_DIV should be compensated by subtracting 7 from the required pulse length. (eg. 500 clock cycles = 0x1F3; subtract 7 = 0x1EC).
15:11	Reserved		0x00	
10:0	INP_CLK_LOW_DIV	RW	0x1EC	Clock Divisor for SCLK Low Phase. This field sets the number of APBCLK cycles in the SCLK (Low) Phase for I2C Master Mode output. 000h = 1 clock cycle 001h = 2 clock cycles 002h = 3 clock cycles … FFFh = 4096 clock cycles Note that, when the I2C Pulse Filter is enabled, then INP_CLK_LOW_DIV should be compensated by subtracting 7 from the required pulse length. (eg. 500 clock cycles = 0x1F3; subtract 7 = 0x1EC)

Table 59 I2C_BAUD Register

I2C_SRXDATA – I²C SLAVE RECEIVE DATA REGISTER

The Slave Receive Data Register (I2C_SRXDATA) holds the byte of data received (SRXDATA) in the I²C Slave Write operation. (The I²C Master initiates the Write operation, and the associated data is received by the I²C Slave.)

The Device Address and I²C Read/Write bit is also received in the SRXDATA field for I²C Write and I²C Read operations. The I²C Read/Write bit (in the LSB position) indicates whether a Write or a Read will follow.

The SRXDATA_FLAG bit indicates when a new byte of data has been successfully received. This bit is cleared when the register is read.

Note that the SLV_RX_FILLED bit in the I2C_STATUS register (see Table 53) also indicates when new data has been received. The associated interrupt may also be enabled.

Each received word must be acknowledged (ACK) by the I²C Slave module. The ACK response is configured by writing to the I2C_STXDATA register (see Table 61), setting the NACK bit to 0.

Note that the ACK response must be configured (by writing to the I2C_STXDATA) before reading the received data byte from I2C_SRXDATA.

For further details, see the following section, describing the I2C_STXDATA register.

New data is only indicated/available following receipt of a matching Device Address (SLV_ADDR). The SLV_ADDR_MATCH bit in the I2C_STATUS register indicates when a matching Device Address is detected; an associated interrupt may also be enabled.

The SLV_XFER_DONE bit in the I2C_STATUS register indicates when an I²C Slave transfer has completed.

I2C_SRXDATA
I2C SLAVE RECEIVE DATA REGISTER

Address = 0xF002_0070 | **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:9	Reserved		0x00_0000	
8	SRXDATA_FLAG	RC	0x0	This is asserted to indicate that the 8-bit byte of data has been received. It is also asserted to indicate that a matched Device Address has been received. 0 = Register empty 1 = Register full (new data received) Note that this bit is cleared on read
7:0	SRXDATA	RO	0x00	Slave Data byte or matched Device Address received. When a data byte is received (SLV_RX_FILLED=1), this field holds the 8-bit data word. When a matched Device Address is received (SLV_ADDR_MATCH=1), this field holds the 7-bit Device Address (bits [7:1]) and the Read/Write bit (bit [0]). The Read/Write bit indicates whether a Write or Read transfer will follow. Note that this is a read only field

Table 60 I²C_SRXDATA Register

I2C_STXDATA – I²C SLAVE TRANSMIT DATA REGISTER

The Slave Transmit Data Register (I2C_STXDATA) holds the byte of data (STXDATA) for transmission in the I²C Slave Read operation. (The I²C Master initiates the Read operation, and the associated data is transmitted by the I²C Slave.) The I2C_STXDATA register also controls the ACK response in I²C Slave Write operations.

The STXDATA_FLAG bit indicates when a byte of data has been loaded, ready for transmission. This bit is set to '0' after the byte has been transmitted, indicating that the register has been emptied and is ready for the next data byte to be loaded. The STXDATA_FLAG is a Read-Only bit.

Note that the SLV_TX_EMPTIED bit in the I2C_STATUS register (see Table 53) also indicates when the I2C_STXDATA register has been emptied. The associated interrupt may also be enabled.

The next data byte for transmission should be written to STXDATA as soon as possible after the SLV_TX_EMPTIED bit is asserted. Note that, in normal I²C Read operations, the Slave device must have prior knowledge of the number of bytes to be transferred. (This is because the STOP Condition, indicating completion of the transfer, occurs later than the time at which the next data bit would otherwise be transmitted.)

For I²C Slave Write transfers, the I2C_STXDATA register must be written to configure the ACK response for each received data byte. Setting NACK=0 will configure the 'ACK' response for the next received data byte.

The NACK bit should be written following a matching Device Address (SLV_ADDR_MATCH=1), and following a received data byte (SLV_RX_FILLED=1); writing to the NACK bit configures the ACK status for the next word that is received.

Note that the NACK bit must be written before reading the received data byte from I2C_SRXDATA.

The Device Address is acknowledged automatically by the I²C Slave module (if the received address matches the SLV_ADDR field). The NACK bit is configured in advance of each byte received, as described above. Accordingly, there is no requirement to write to the NACK bit after receipt of the last byte of the transfer.

Data is only transmitted following receipt of a matching Device Address (SLV_ADDR). The SLV_ADDR_MATCH bit in the I2C_STATUS register indicates when a matching Device Address is detected; an associated interrupt may also be enabled.

The SLV_XFER_DONE bit in the I2C_STATUS register indicates when an I²C Slave transfer has completed.

I2C_STXDATA
I2C SLAVE TRANSMIT DATA REGISTER

Address = 0xF002_0074 | **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:10	Reserved		0x00_0000	
9	STXDATA_FLAG	RO	0x0	Indicates when the register has been loaded and is ready for transmission. 0 = Register empty 1 = Register full (ready to transmit) Note that this bit is read only
8	NACK	RW	0x0	ACK or NACK configuration for I²C Slave Write transfers 0 = ACK 1 = NACK
7:0	STXDATA	RW	0x00	Slave Data byte for transmission (I²C read)

Table 61 I2C_STXDATA Register

I2C_SLV_ADDR – I²C SLAVE ADDRESS REGISTER

The I²C slave address is held in the SLV_ADDR register. Note that this is the 7-bit Device Address, and does not include the Read/Write bit. The equivalent 8-bit Device Address comprises the 7 bits of SLV_ADDR, and the R/W bit in the LSB position. This is illustrated by an example in Table 62.

I²C ACTION	SLV_ADDR	8-BIT DEVICE ADDRESS
Write	0x38 (hex), 011 1000 (binary)	0x70 (hex), 0111 0000 (binary)
Read		0x71 (hex), 0111 0001 (binary)

Table 62 Illustration of 7-bit SLV_ADDR compared with 8-bit Device Address

I2C_SLV_ADDR
I2C SLAVE RECEIVE DATA REGISTER

Address = 0xF002_0078 **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:7	Reserved		0x000_0000	
6:0	SLV_ADDR[6:0]	RW	0x00	This field holds the 7 bit I²C slave address. (Note this does not include the R/W bit.)

Table 63 I2C_SLV_ADDR Register

FUSE MEMORY

The WM0011 incorporates a one-time-programmable (OTP) fuse memory. The fuse data can be used to select which interface will be used for software/configuration download. The fuses also allow the start-up condition of selected control registers to be configured.

Note that the fuse data capability is supported on custom-programmed devices only. Un-programmed devices do not support these options. Fuse programming by users is not supported.

As part of the boot sequence, the WM0011 will determine whether the Custom fuses have been programmed. If the Custom fuses have been programmed, then the fuse data will select the desired clocking configuration, and also select the desired boot method for software/configuration download.

Note that the Custom fuse data includes parameters that are held in the WM0011 control registers. The fuse settings will be loaded as initial start-up values of the corresponding registers, but these can be updated during normal operation later if required.

See “Boot Sequence Control” for further details of the WM0011 boot-up process.

The integrated fuse memory holds 512 bytes of configuration data, including:

- Software Authentication Key
- Boot-up & Clocking configuration
- Cyclic Redundancy Check (CRC)
- Custom-defined data
- JTAG de-bug module configuration

A full definition of the Custom fuse memory contents is provided in Table 64.

See below for further details of the CRC and the Software Authentication Key.

FUSE MEMORY DEFINITION

The memory map of the Custom fuse region is provided in Table 64. See below for further details of the CRC value, and the Software Authentication key fields that are held within the fuse memory.

BYTE ADDRESS	SIZE (BYTES)	DESCRIPTION	COMMENT
0x000	256	PUB_KEY_MOD	Public Key (Modulus), for Software Authentication
0x100	4	PUB_KEY_EXP	Public Key (Exponent), for Software Authentication
0x104	1	BOOT_SRC	Selects the boot method: 0h = SPI (Slave) port 1h = SPI (Master) port, eg. SST Flash Memory All other codes are Reserved
0x105	1	UART_DLL	UART Divisor (LSW), for debug messaging. Setting UART_DLL and UART_DLH to 0xFF will disable debug messaging.
0x106	1	UART_DLH	UART Divisor (MSW), for debug messaging. Setting UART_DLL and UART_DLH to 0xFF will disable debug messaging.
0x107	1	Reserved	
0x108	4	CCM_CONTROL	General Control Register - see Table 16
0x10C	4	CCM_GPIO_SEL	Port Select Register - see Table 18
0x110	4	CCM_CLK_CTRL1	Clock Control 1 Register - see Table 19
0x114	4	CCM_CLK_CTRL2	Clock Control 2 Register - see Table 20
0x118	4	CCM_CLK_CTRL3	Clock Control 3 Register - see Table 21
0x11C	4	CCM_PLL_LOCK_CTRL	PLL Lock Detect Control Register - see Table 22
0x120	4	CCM_IOCTRL1	I/O Control 1 Register - see Table 33
0x124	4	CCM_IOCTRL2	I/O Control 2 Register - see Table 34
0x128	4	CCM_IOCTRL3	I/O Control 3 Register - see Table 35
0x12C	4	CCM_IOCTRL4	I/O Control 4 Register - see Table 36
0x130	4	CCM_IOCTRL5	I/O Control 5 Register - see Table 37
0x134	4	CCM_IOCTRL6	I/O Control 6 Register - see Table 38
0x138	4	CCM_IOCTRL7	I/O Control 7 Register - see Table 39
0x13C	4	CCM_IOCTRL8	I/O Control 8 Register - see Table 40
0x140	4	CCM_IOCTRL9	I/O Control 9 Register - see Table 41Table 41
0x144	4	CCM_IOCTRL10	I/O Control 10 Register - see Table 42
0x148	4	CCM_IOCTRL11	I/O Control 11 Register - see Table 43
0x14C to 0x18C	68	Reserved	
0x190	4	SPI_SCLKDIV	SPI Clock Division Register - see Table 123 Selects the SCLK frequency for SPI (Master) boot-up. The maximum clock rate of 40MHz must not be exceeded.
0x194 to 0x1A3	16	Reserved	
0x1A4	4	CRC	CRC value, calculated over byte addresses 0x000 to 0x1A3.
0x1A8	80	(undefined)	Available for custom use. This region is not used by the boot process, and is not protected by the CRC.

Table 64 Fuse Memory Definition

CYCLIC REDUNDANCY CHECK (CRC)

The Custom fuse memory is protected by a CRC, which is calculated using the IEEE 802.3 polynomial over byte addresses 0x000 to 0x1A3.

An error condition will be detected during start-up if the Custom fuse CRC value does not match the CRC value calculated by the WM0011.

SOFTWARE AUTHENTICATION

Software Code authentication is implemented, to ensure that the WM0011 will only execute code that has been supplied by an approved vendor.

All Software Code downloads must include an authentication signature. The signature (SHA-256) is contained within the Software Header download. The WM0011 will check the downloaded signature against an internally-generated signature, and will only execute the code if the signatures match.

As an option, the authentication signature may be encrypted. In this case, the code will only be executed if the correct Public Key data is configured in the Custom fuse memory. Secure authentication ensures that the WM0011 will only execute code that has been supplied by an approved vendor with the correct Private Key required to encrypt the signature.

Note that PKA-encryption of the image signature can only be supported on custom-programmed devices. This is not supported on un-programmed devices.

If the JTAG function has been disabled (custom-programmed devices only), then the WM0011 will only execute code that includes a PKA-encrypted (secure) signature.

The supported options for software authentication are shown in Table 65.

DEVICE TYPE	JTAG CONFIGURATION	UNENCRYPTED SIGNATURE	ENCRYPTED SIGNATURE
Custom	Disabled		✓
Custom	Enabled	✓	✓
Un-programmed	Enabled	✓	

Table 65 Software Authentication support

GENERAL PURPOSE INPUT/OUTPUT (GPIO) MODULE

BASE ADDRESS 0xF004_0000

GPIO FEATURES

- 17 configurable GPIO pins (multiplexed with other functions)
- Configurable Interrupt logic using edge or level detection
- Individual Mask control for each GPIO
- Interrupt output (to IRQC module)

The GPIO module supports 17 configurable GPIO pins. These can be configured as Input or Output. Any pins configured as Input may be selected as interrupt sources for the GPIO module. The interrupt sources can be edge or level sensitive; the active polarity is also selectable. A priority-encoded readback is available on the occurrence of a GPIO interrupt.

INPUT / OUTPUT CONTROL

Each bit may serve as either a programmed input or programmed output in this mode.

The GPIO_DIR register configures each bit as an input or an output. The system powers up with each input configured as an input (the register bits are cleared). By setting the associated bit in GPIO_DIR, the bit is controlled as an output.

The logic level each input is observable after de-metastability logic and inversion logic by reading GPIO_IN. When input inversion is selected (using GPIO_INV), value read from GPIO_IN will be the opposite logic level from that appearing at the external pin.

When any GPIO is configured as an output, the logic level at the pad will be controlled by the respective GPIO_OUT register bit. Note that GPIO output is not affected by the GPIO_INV bits.

LEVEL/EDGE INTERRUPT CONTROL

The interrupt inputs are configured as level sensitive or edge sensitive using the GPIO_EDGE1 and GPIO_EDGE0 registers.

In Level-sensitive configuration, the interrupt is asserted when the applicable logic level is detected. Active High is selected when GPIO_INV=0; Active Low is selected when GPIO_INV=1. Note that the interrupt cannot be cleared whilst the Active logic level is present (and unmasked) on the respective GPIO.

In Edge-sensitive configuration, the interrupt is asserted when the applicable logic transition is detected. This may be the rising (leading) edge, falling (trailing) edge, or both edges. The active edge(s) will cause the respective GPIO_INT_STS bit to be set. Note that the active edge(s) are inverted when GPIO_INV=1.

In each case, the interrupt status bits in the GPIO_INT_STS register are latching bits, and are only cleared when a ‘1’ is written to the respective bit in the GPIO_INT_CLR register. To observe successive interrupts, the GPIO_INT_STS bit must be cleared before another interrupt event can be registered.

To avoid false interrupts, the input signals must be in their respective de-asserted logic states when the interrupts are enabled. Note that the input inversion must be considered when determining the de-asserted logic state.

When a rising-edge interrupt is enabled and unmasked, the corresponding interrupt status will be asserted if the relevant input signal is logic 1. Similarly, when a falling-edge interrupt is enabled and unmasked, the corresponding interrupt status will be asserted if the relevant input is logic 0. In other words, the behaviour is effectively level-triggered at the point when the interrupt is initially configured.

If necessary, the interrupt service routines should take account of the behaviour described above, and should clear the respective interrupt(s) immediately after they are enabled, before they are unmasked.

The control sequence below is recommended to ensure false interrupts are avoided.

- Mask the interrupt using GPIOn_INT_MSK=1
- Configure the GPIO interrupt registers (including GPIOn_INT_ENA=1)
- Clear the interrupt using GPIOn_INT_CLR=1
- Unmask the interrupt using GPIOn_INT_MSK=0

GPIO INTERRUPTS

An input is considered part of the interrupt system when the associated enable bit in GPIO_INT_CTRL is set. The register is cleared at reset. Consequently, no input bits are considered interrupt sources at reset.

The interrupt inputs are configured as level sensitive or edge sensitive using the GPIO_EDGE1 and GPIO_EDGE0 registers.

Each GPIO may be individually masked from the interrupt structure by setting the corresponding GPIO_INT_MSK bit. The Mask bits are set by default, so the interrupt structure is disabled until the corresponding bit is enabled (using GPIO_INT_CTRL) and unmasked (using GPIO_INT_MSK).

When a valid level or edge is detected on an interrupt input, the corresponding GPIO_INT_STS bit is set (provided that the corresponding input is enabled and unmasked). These bits are latching bits, and are only cleared when a '1' is written to the respective bit in the GPIO_INT_CLR register.

The GPIO_INT_STS register provides readback of all the enabled and unmasked GPIO interrupts. The GPIO_INT_VECT register provides a readback of the single, highest priority unmasked & asserted GPIO interrupt. Highest priority is implemented as the interrupt in the lowest-numbered bit position of the GPIO_INT_STS register. For example, the GPIO 4 Interrupt (in bit [4]), is given higher priority than the GPIO5 Interrupt (in bit 5).

When one or more bit in the GPIO_INT_STS register is set, the GPIO Interrupt input to the IRQC Module is asserted. Note that the GPIO Interrupt input to the IRQC Module is Active Low.

The GPIO interrupt control registers are illustrated in Figure 31.

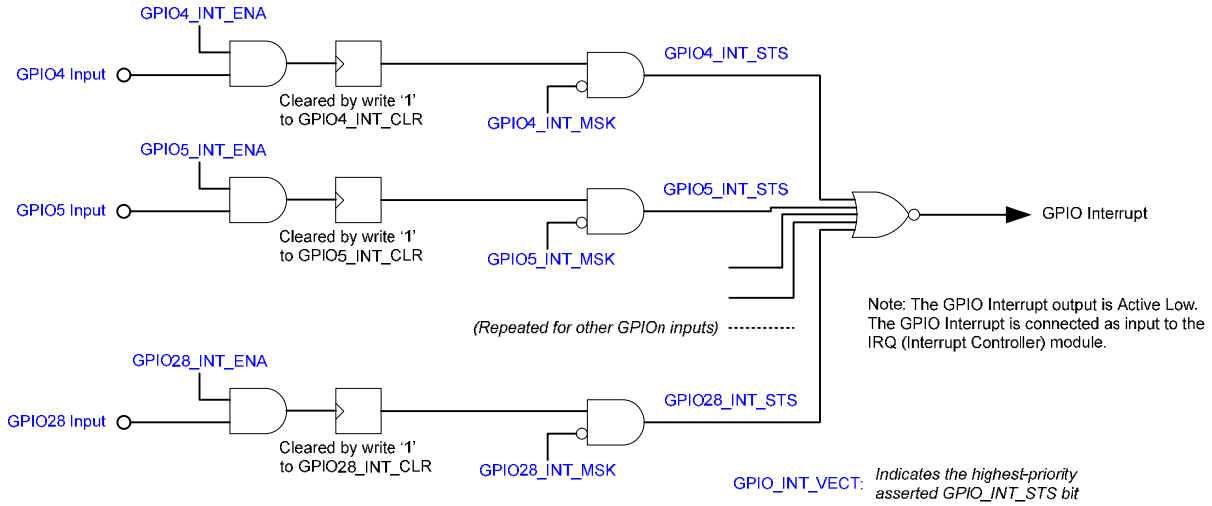


Figure 31 GPIO Interrupts

GPIO REGISTER MAP

ADDRESS	REGISTER	DESCRIPTION	RESET VALUE
Base + 0x00	GPIO_OUT	GPIO Output	0x0000_0000
Base + 0x04	GPIO_IN	GPIO Input	Undefined
Base + 0x08	GPIO_DIR	GPIO Direction	0x0000_0000
Base + 0x0C	GPIO_INV	GPIO Inversion	0x0000_0000
Base + 0x10	GPIO_EDGE0	GPIO Edge Detection 0	0x0000_0000
Base + 0x14	GPIO_EDGE1	GPIO Edge Detection 1	0x0000_0000
Base + 0x18	GPIO_INT_CTRL	GPIO Interrupt Control	0x0000_0000
Base + 0x1C	GPIO_INT_CLR	GPIO Interrupt Clear	0x0000_0000
Base + 0x20	GPIO_INT_MSK	GPIO Interrupt Mask	0xFFFF_FFFF
Base + 0x24	GPIO_INT_VECT	GPIO Interrupt Vector	0x0000_0000
Base + 0x28	GPIO_INT_STS	GPIO Interrupt Status Register	0x0000_0000

Table 66 GPIO Register Definition

GPIO_OUT – GPIO OUTPUT REGISTER

GPIO_OUT
GPIO OUTPUT REGISTER

Address = 0xF004_0000 | **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	Reserved		0x0	
28	GPIO28_OUT	RW	0x0	Controls the logic level of GPIO28 when configured as output
27:24	Reserved		0x0	
23	GPIO23_OUT	RW	0x0	Controls the logic level of GPIO23 when configured as output
22	GPIO22_OUT	RW	0x0	Controls the logic level of GPIO22 when configured as output
21:20	Reserved		0x0	
19	GPIO19_OUT	RW	0x0	Controls the logic level of GPIO19 when configured as output
18	GPIO18_OUT	RW	0x0	Controls the logic level of GPIO18 when configured as output
17	GPIO17_OUT	RW	0x0	Controls the logic level of GPIO17 when configured as output
16:15	Reserved		0x0	
14	GPIO14_OUT	RW	0x0	Controls the logic level of GPIO14 when configured as output
13	GPIO13_OUT	RW	0x0	Controls the logic level of GPIO13 when configured as output
12	GPIO12_OUT	RW	0x0	Controls the logic level of GPIO12 when configured as output
11	GPIO11_OUT	RW	0x0	Controls the logic level of GPIO11 when configured as output
10	GPIO10_OUT	RW	0x0	Controls the logic level of GPIO10 when configured as output
9	GPIO9_OUT	RW	0x0	Controls the logic level of GPIO9 when configured as output
8	GPIO8_OUT	RW	0x0	Controls the logic level of GPIO8 when configured as output
7	GPIO7_OUT	RW	0x0	Controls the logic level of GPIO7 when configured as output
6	GPIO6_OUT	RW	0x0	Controls the logic level of GPIO6 when configured as output
5	GPIO5_OUT	RW	0x0	Controls the logic level of GPIO5 when configured as output
4	GPIO4_OUT	RW	0x0	Controls the logic level of GPIO4 when configured as output
3:0	Reserved		0x0	

Table 67 GPIO_OUT Register

GPIO_IN – GPIO INPUT REGISTER

GPIO_IN GPIO INPUT REGISTER																															
Address = 0xF004_0004																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	Reserved		0x0	
28	GPIO28_IN	RO	0x0	Indicates the logic level of GPIO28 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
27:24	Reserved		0x0	
23	GPIO23_IN	RO	0x0	Indicates the logic level of GPIO23 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
22	GPIO22_IN	RO	0x0	Indicates the logic level of GPIO22 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
21:20	Reserved		0x0	
19	GPIO19_IN	RO	0x0	Indicates the logic level of GPIO19 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
18	GPIO18_IN	RO	0x0	Indicates the logic level of GPIO18 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
17	GPIO17_IN	RO	0x0	Indicates the logic level of GPIO17 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
16:15	Reserved		0x0	
14	GPIO14_IN	RO	0x0	Indicates the logic level of GPIO14 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
13	GPIO13_IN	RO	0x0	Indicates the logic level of GPIO13 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
12	GPIO12_IN	RO	0x0	Indicates the logic level of GPIO12 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
11	GPIO11_IN	RO	0x0	Indicates the logic level of GPIO11 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
10	GPIO10_IN	RO	0x0	Indicates the logic level of GPIO10 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
9	GPIO9_IN	RO	0x0	Indicates the logic level of GPIO9 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
8	GPIO8_IN	RO	0x0	Indicates the logic level of GPIO8 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
7	GPIO7_IN	RO	0x0	Indicates the logic level of GPIO7 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
6	GPIO6_IN	RO	0x0	Indicates the logic level of GPIO6 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
5	GPIO5_IN	RO	0x0	Indicates the logic level of GPIO5 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
4	GPIO4_IN	RO	0x0	Indicates the logic level of GPIO4 when configured as input. Readback is after de-metastability logic and GPIO_INV inversion (if applicable).
3:0	Reserved		0x0	

Table 68 GPIO_IN Register

GPIO_DIR – GPIO DIRECTION REGISTER

GPIO_DIR GPIO DIRECTION REGISTER																															
Address = 0xF004_0008																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	Reserved		0x0	
28	GPIO28_DIR	RW	0x0	Selects GPIO28 direction. 0 = input. 1 = output.
27:24	Reserved		0x0	
23	GPIO23_DIR	RW	0x0	Selects GPIO23 direction. 0 = input. 1 = output.
22	GPIO22_DIR	RW	0x0	Selects GPIO22 direction. 0 = input. 1 = output.
21:20	Reserved		0x0	
19	GPIO19_DIR	RW	0x0	Selects GPIO19 direction. 0 = input. 1 = output.
18	GPIO18_DIR	RW	0x0	Selects GPIO18 direction. 0 = input. 1 = output.
17	GPIO17_DIR	RW	0x0	Selects GPIO17 direction. 0 = input. 1 = output.
16:15	Reserved		0x0	
14	GPIO14_DIR	RW	0x0	Selects GPIO14 direction. 0 = input. 1 = output.
13	GPIO13_DIR	RW	0x0	Selects GPIO13 direction. 0 = input. 1 = output.
12	GPIO12_DIR	RW	0x0	Selects GPIO12 direction. 0 = input. 1 = output.
11	GPIO11_DIR	RW	0x0	Selects GPIO11 direction. 0 = input. 1 = output.
10	GPIO10_DIR	RW	0x0	Selects GPIO10 direction. 0 = input. 1 = output.
9	GPIO9_DIR	RW	0x0	Selects GPIO9 direction. 0 = input. 1 = output.
8	GPIO8_DIR	RW	0x0	Selects GPIO8 direction. 0 = input. 1 = output.
7	GPIO7_DIR	RW	0x0	Selects GPIO7 direction. 0 = input. 1 = output.
6	GPIO6_DIR	RW	0x0	Selects GPIO6 direction. 0 = input. 1 = output.
5	GPIO5_DIR	RW	0x0	Selects GPIO5 direction. 0 = input. 1 = output.
4	GPIO4_DIR	RW	0x0	Selects GPIO4 direction. 0 = input. 1 = output.
3:0	Reserved		0x0	

Table 69 GPIO_DIR Register

GPIO_INV – GPIO INVERSION REGISTER

GPIO_INV GPIO INVERSION REGISTER																															
Address = 0xF004_000C																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	Reserved		0x0	
28	GPIO28_INV	RW	0x0	Selects GPIO28 input inversion. 0 = no inversion. 1 = inversion.
27:24	Reserved		0x0	
23	GPIO23_INV	RW	0x0	Selects GPIO23 input inversion. 0 = no inversion. 1 = inversion.
22	GPIO22_INV	RW	0x0	Selects GPIO22 input inversion. 0 = no inversion. 1 = inversion.
21:20	Reserved		0x0	
19	GPIO19_INV	RW	0x0	Selects GPIO19 input inversion. 0 = no inversion. 1 = inversion.
18	GPIO18_INV	RW	0x0	Selects GPIO18 input inversion. 0 = no inversion. 1 = inversion.

GPIO_INV GPIO INVERSION REGISTER																															
Address = 0xF004_000C																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
17	GPIO17_INV	RW	0x0	Selects GPIO17 input inversion. 0 = no inversion. 1 = inversion.
16:15	Reserved		0x0	
14	GPIO14_INV	RW	0x0	Selects GPIO14 input inversion. 0 = no inversion. 1 = inversion.
13	GPIO13_INV	RW	0x0	Selects GPIO13 input inversion. 0 = no inversion. 1 = inversion.
12	GPIO12_INV	RW	0x0	Selects GPIO12 input inversion. 0 = no inversion. 1 = inversion.
11	GPIO11_INV	RW	0x0	Selects GPIO11 input inversion. 0 = no inversion. 1 = inversion.
10	GPIO10_INV	RW	0x0	Selects GPIO10 input inversion. 0 = no inversion. 1 = inversion.
9	GPIO9_INV	RW	0x0	Selects GPIO9 input inversion. 0 = no inversion. 1 = inversion.
8	GPIO8_INV	RW	0x0	Selects GPIO8 input inversion. 0 = no inversion. 1 = inversion.
7	GPIO7_INV	RW	0x0	Selects GPIO7 input inversion. 0 = no inversion. 1 = inversion.
6	GPIO6_INV	RW	0x0	Selects GPIO6 input inversion. 0 = no inversion. 1 = inversion.
5	GPIO5_INV	RW	0x0	Selects GPIO5 input inversion. 0 = no inversion. 1 = inversion.
4	GPIO4_INV	RW	0x0	Selects GPIO4 input inversion. 0 = no inversion. 1 = inversion.
3:0	Reserved		0x0	

Table 70 GPIO_INV Register

EDGE DETECTION

The interrupt inputs are configured as level sensitive or edge sensitive using the GPIO_EDGE1 and GPIO_EDGE0 registers, as described in Table 71.

GPIO_EDGE1	GPIO_EDGE0	DESCRIPTION
0	0	Level Sensitive
0	1	Leading Edge
1	0	Trailing Edge
1	1	Dual Edge Interrupt

Table 71 GPIO Edge Detection Control

In Level-sensitive configuration, the interrupt is asserted when the applicable logic level is detected. Active High is selected when GPIO_INV=0; Active Low is selected when GPIO_INV=1. Note that the interrupt cannot be cleared whilst the Active logic level is present (and unmasked) on the respective GPIO.

In Edge-sensitive configuration, the interrupt is asserted when the applicable logic transition is detected. This may be the rising (leading) edge, falling (trailing) edge, or both edges. The active edge(s) will cause the respective GPIO_INT_STS bit to be set. Note that the active edge(s) are inverted when GPIO_INV=1.

GPIO_EDGE0 – GPIO EDGE DETECTION 0 REGISTER

GPIO_EDGE0 GPIO EDGE DETECTION 0 REGISTER																															
Address = 0xF004_0010																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	Reserved		0x0	
28	GPIO28_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
27:24	Reserved		0x0	
23	GPIO23_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
22	GPIO22_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
21:20	Reserved		0x0	
19	GPIO19_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
18	GPIO18_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
17	GPIO17_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
16:15	Reserved		0x0	
14	GPIO14_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
13	GPIO13_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
12	GPIO12_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
11	GPIO11_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
10	GPIO10_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
9	GPIO9_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
8	GPIO8_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
7	GPIO7_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
6	GPIO6_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
5	GPIO5_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
4	GPIO4_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
3:0	Reserved		0x0	

Table 72 GPIO_EDGE0 Register

GPIO_EDGE1 – GPIO EDGE DETECTION 1 REGISTER

GPIO_EDGE1 GPIO EDGE DETECTION 1 REGISTER																															
Address = 0xF004_0014																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	Reserved		0x0	
28	GPIO28_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
27:24	Reserved		0x0	
23	GPIO23_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
22	GPIO22_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
21:20	Reserved		0x0	
19	GPIO19_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
18	GPIO18_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0

GPIO_EDGE1 GPIO EDGE DETECTION 1 REGISTER																															
Address = 0xF004_0014																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
17	GPIO17_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
16:15	Reserved		0x0	
14	GPIO14_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
13	GPIO13_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
12	GPIO12_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
11	GPIO11_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
10	GPIO10_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
9	GPIO9_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
8	GPIO8_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
7	GPIO7_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
6	GPIO6_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
5	GPIO5_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
4	GPIO4_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
3:0	Reserved		0x0	

Table 73 GPIO_EDGE1 Register

GPIO_INT_CTRL – GPIO INTERRUPT CONTROL REGISTER

GPIO_INT_CTRL GPIO INTERRUPT CONTROL REGISTER																															
Address = 0xF004_0018																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	Reserved		0x0	
28	GPIO28_INT_ENA	RW	0x0	Enables GPIO28 as an input to the GPIO Interrupt logic
27:24	Reserved		0x0	
23	GPIO23_INT_ENA	RW	0x0	Enables GPIO23 as an input to the GPIO Interrupt logic
22	GPIO22_INT_ENA	RW	0x0	Enables GPIO22 as an input to the GPIO Interrupt logic
21:20	Reserved		0x0	
19	GPIO19_INT_ENA	RW	0x0	Enables GPIO19 as an input to the GPIO Interrupt logic
18	GPIO18_INT_ENA	RW	0x0	Enables GPIO18 as an input to the GPIO Interrupt logic
17	GPIO17_INT_ENA	RW	0x0	Enables GPIO17 as an input to the GPIO Interrupt logic
16:15	Reserved		0x0	
14	GPIO14_INT_ENA	RW	0x0	Enables GPIO14 as an input to the GPIO Interrupt logic
13	GPIO13_INT_ENA	RW	0x0	Enables GPIO13 as an input to the GPIO Interrupt logic
12	GPIO12_INT_ENA	RW	0x0	Enables GPIO12 as an input to the GPIO Interrupt logic
11	GPIO11_INT_ENA	RW	0x0	Enables GPIO11 as an input to the GPIO Interrupt logic
10	GPIO10_INT_ENA	RW	0x0	Enables GPIO10 as an input to the GPIO Interrupt logic
9	GPIO9_INT_ENA	RW	0x0	Enables GPIO9 as an input to the GPIO Interrupt logic
8	GPIO8_INT_ENA	RW	0x0	Enables GPIO8 as an input to the GPIO Interrupt logic
7	GPIO7_INT_ENA	RW	0x0	Enables GPIO7 as an input to the GPIO Interrupt logic
6	GPIO6_INT_ENA	RW	0x0	Enables GPIO6 as an input to the GPIO Interrupt logic

GPIO_INT_CTRL GPIO INTERRUPT CONTROL REGISTER																															
Address = 0xF004_0018																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
5	GPIO5_INT_ENA	RW	0x0	Enables GPIO5 as an input to the GPIO Interrupt logic
4	GPIO4_INT_ENA	RW	0x0	Enables GPIO4 as an input to the GPIO Interrupt logic
3:0	Reserved		0x0	

Table 74 GPIO_INT_CTRL Register

GPIO_INT_CLR – GPIO INTERRUPT CLEAR REGISTER

GPIO_INT_CLR GPIO INTERRUPT CLEAR REGISTER																															
Address = 0xF004_001C																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	Reserved		0x0	
28	GPIO28_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO28 interrupt (GPIO_INT28_STS)
27:24	Reserved		0x0	
23	GPIO23_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO23 interrupt (GPIO_INT23_STS)
22	GPIO22_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO22 interrupt (GPIO_INT22_STS)
21:20	Reserved		0x0	
19	GPIO19_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO19 interrupt (GPIO19_INT_STS)
18	GPIO18_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO18 interrupt (GPIO18_INT_STS)
17	GPIO17_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO17 interrupt (GPIO17_INT_STS)
16:15	Reserved		0x0	
14	GPIO14_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO14 interrupt (GPIO14_INT_STS)
13	GPIO13_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO13 interrupt (GPIO13_INT_STS)
12	GPIO12_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO12 interrupt (GPIO12_INT_STS)
11	GPIO11_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO11 interrupt (GPIO11_INT_STS)
10	GPIO10_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO10 interrupt (GPIO10_INT_STS)
9	GPIO9_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO9 interrupt (GPIO9_INT_STS)
8	GPIO8_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO8 interrupt (GPIO8_INT_STS)
7	GPIO7_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO7 interrupt (GPIO7_INT_STS)
6	GPIO6_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO6 interrupt (GPIO6_INT_STS)
5	GPIO5_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO5 interrupt (GPIO5_INT_STS)
4	GPIO4_INT_CLR	WO	0x0	Write ‘1’ to clear GPIO4 interrupt (GPIO4_INT_STS)
3:0	Reserved		0x0	

Table 75 GPIO_INT_CLR Register

GPIO_INT_MSK – GPIO INTERRUPT MASK REGISTER

GPIO_INT_MSK
GPIO INTERRUPT MASK REGISTER

Address = 0xF004_0020 | **Default value = 0xFFFF_FFFF**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	Reserved		0x7	
28	GPIO28_INT_MSK	RW	0x1	Selects whether GPIO28 interrupt is masked. A masked interrupt will not trigger the GPIO28_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.
27:24	Reserved		0xF	
23	GPIO23_INT_MSK	RW	0x1	Selects whether GPIO23 interrupt is masked. A masked interrupt will not trigger the GPIO23_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.
22	GPIO22_INT_MSK	RW	0x1	Selects whether GPIO22 interrupt is masked. A masked interrupt will not trigger the GPIO22_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.
21:20	Reserved		0x3	
19	GPIO19_INT_MSK	RW	0x1	Selects whether GPIO19 interrupt is masked. A masked interrupt will not trigger the GPIO19_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.
18	GPIO18_INT_MSK	RW	0x1	Selects whether GPIO18 interrupt is masked. A masked interrupt will not trigger the GPIO18_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.
17	GPIO17_INT_MSK	RW	0x1	Selects whether GPIO17 interrupt is masked. A masked interrupt will not trigger the GPIO17_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.
16:15	Reserved		0x3	
14	GPIO14_INT_MSK	RW	0x1	Selects whether GPIO14 interrupt is masked. A masked interrupt will not trigger the GPIO14_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.
13	GPIO13_INT_MSK	RW	0x1	Selects whether GPIO13 interrupt is masked. A masked interrupt will not trigger the GPIO13_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.
12	GPIO12_INT_MSK	RW	0x1	Selects whether GPIO12 interrupt is masked. A masked interrupt will not trigger the GPIO12_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.
11	GPIO11_INT_MSK	RW	0x1	Selects whether GPIO11 interrupt is masked. A masked interrupt will not trigger the GPIO11_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.
10	GPIO10_INT_MSK	RW	0x1	Selects whether GPIO10 interrupt is masked. A masked interrupt will not trigger the GPIO10_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.

GPIO_INT_MSK GPIO INTERRUPT MASK REGISTER																															
Address = 0xF004_0020																Default value = 0xFFFF_FFFF															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
9	GPIO9_INT_MSK	RW	0x1	Selects whether GPIO9 interrupt is masked. A masked interrupt will not trigger the GPIO9_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.
8	GPIO8_INT_MSK	RW	0x1	Selects whether GPIO8 interrupt is masked. A masked interrupt will not trigger the GPIO8_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.
7	GPIO7_INT_MSK	RW	0x1	Selects whether GPIO7 interrupt is masked. A masked interrupt will not trigger the GPIO7_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.
6	GPIO6_INT_MSK	RW	0x1	Selects whether GPIO6 interrupt is masked. A masked interrupt will not trigger the GPIO6_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.
5	GPIO5_INT_MSK	RW	0x1	Selects whether GPIO5 interrupt is masked. A masked interrupt will not trigger the GPIO5_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.
4	GPIO4_INT_MSK	RW	0x1	Selects whether GPIO4 interrupt is masked. A masked interrupt will not trigger the GPIO4_INT_STS bit, and is disabled from the GPIO_INT_VECT logic. 0 = Enabled; 1 = Masked.
3:0	Reserved		0xF	

Table 76 GPIO_INT_MSK Register

GPIO_INT_VECT – GPIO INTERRUPT VECTOR REGISTER

GPIO_INT_VECT GPIO INTERRUPT VECTOR REGISTER																															
Address = 0xF004_0024																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:5	Reserved		0x000_0000	
4:0	GPIO_INT_VECT	RO	0x00	Indicates which highest priority interrupt is currently asserted in the GPIO_INT_STS register. Highest priority is implemented as the interrupt in the lowest-numbered bit position of the GPIO_INT_STS register. The same priority is represented in the coding of GPIO_INT_VECT - the GPIO4 Interrupt is highest priority, and the GPIO28 Interrupt is lowest priority. 0x00 = No interrupt 0x04 = GPIO4 0x05 = GPIO5 etc.

Table 77 GPIO_INT_VECT Register

GPIO_INT_STS – GPIO INTERRUPT STATUS REGISTER

GPIO_INT_STS
GPIO INTERRUPT STATUS REGISTER

Address = 0xF004_0028 | **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	Reserved		0x0	
28	GPIO28_INT_STS	RO	0x0	GPIO28 Interrupt Status
27:24	Reserved		0x0	
23	GPIO23_INT_STS	RO	0x0	GPIO23 Interrupt Status
22	GPIO22_INT_STS	RO	0x0	GPIO22 Interrupt Status
21:20	Reserved		0x0	
19	GPIO19_INT_STS	RO	0x0	GPIO19 Interrupt Status
18	GPIO18_INT_STS	RO	0x0	GPIO18 Interrupt Status
17	GPIO17_INT_STS	RO	0x0	GPIO17 Interrupt Status
16:15	Reserved		0x0	
14	GPIO14_INT_STS	RO	0x0	GPIO14 Interrupt Status
13	GPIO13_INT_STS	RO	0x0	GPIO13 Interrupt Status
12	GPIO12_INT_STS	RO	0x0	GPIO12 Interrupt Status
11	GPIO11_INT_STS	RO	0x0	GPIO11 Interrupt Status
10	GPIO10_INT_STS	RO	0x0	GPIO10 Interrupt Status
9	GPIO9_INT_STS	RO	0x0	GPIO9 Interrupt Status
8	GPIO8_INT_STS	RO	0x0	GPIO8 Interrupt Status
7	GPIO7_INT_STS	RO	0x0	GPIO7 Interrupt Status
6	GPIO6_INT_STS	RO	0x0	GPIO6 Interrupt Status
5	GPIO5_INT_STS	RO	0x0	GPIO5 Interrupt Status
4	GPIO4_INT_STS	RO	0x0	GPIO4 Interrupt Status
3:0	Reserved		0x0	

Table 78 GPIO_INT_STS Register

INTERRUPT CONTROLLER (IRQC) MODULE

BASE ADDRESS 0xF005_0000

INTERRUPT CONTROLLER (IRQC) FEATURES

- 10 interrupt inputs from peripheral modules, including cascaded GPIO input
- De-bounced input from the $\overline{\text{STANDBY}}$ pin
- Register control of the $\overline{\text{IRQ}}$ output pin
- 2 register-controlled software interrupts
- Configurable interrupt logic using edge or level detection
- Individual Mask control for each interrupt
- Configurable FIRQ_N output to the Wake-Up FSM
- Configurable IRQ_N and FIRQ_N outputs to the Wake-Up FSM and HiFi EP™ DSP Core

The IRQC module supports 11 inputs, comprising Interrupt signals from peripheral modules (eg. I2C Module), the cascaded input from the GPIO module, and also the de-bounced input from the $\overline{\text{STANDBY}}$ pin.

Any of the inputs may be selected as interrupt sources for the IRQC module, and used to generate the IRQ_N and FIRQ_N outputs to the HiFi EP™. A priority-encoded readback is available on the occurrence of an IRQ_N or FIRQ_N interrupt.

The FIRQ_N ('Fast Interrupt) signal is also an input to the CCM module, providing a configurable 'Wakeup' control signal.

Note that many of the peripheral module interrupt signals are also independently provided as direct inputs to the HiFi EP™.

The IRQC module provides software capability to generate user-defined interrupts to the HiFi EP™ and also to directly control the $\overline{\text{IRQ}}$ output pin logic level.

The inputs and outputs of the IRQC module are illustrated in Figure 15.

INPUT / OUTPUT CONTROL

Each signal described in the IRQC_DIR must be configured as an input or as an output. The software interrupts (bits [15:14] and the $\overline{\text{IRQ}}$ output (bit [0]) should be configured as outputs. All other bits should be configured as inputs.

The logic level each input is observable after de-metastability logic and inversion logic by reading IRQC_IN. When input inversion is selected (using IRQC_INV), value read from IRQC_IN will be the opposite logic level from the signal source.

In the case of output signals, these are controlled by the respective IRQC_OUT register bits. Note that these outputs are not affected by the IRQC_INV bits.

LEVEL/EDGE INTERRUPT CONTROL

The interrupt inputs are configured as level sensitive or edge sensitive using the IRQC_EDGE1 and IRQC_EDGE0 registers.

In Level-sensitive configuration, the interrupt is asserted when the applicable logic level is detected. Active High is selected when IRQC_INV=0; Active Low is selected when IRQC_INV=1. Note that the interrupt cannot be cleared whilst the Active logic level is present (and unmasked) on the respective IRQC input.

In Edge-sensitive configuration, the interrupt is asserted when the applicable logic transition is detected. This may be the rising (leading) edge, falling (trailing) edge, or both edges. The active edge(s) will cause the respective IRQC_IRQ_STS and/or IRQC_FIRQ_STS bit to be set (as controlled by the respective mask bits). Note that the active edge(s) are inverted when IRQC_INV=1.

In each case, the interrupt status bits in the IRQC_IRQ_STS and IRQC_FIRQ_STS registers are latching bits, and are only cleared when a '1' is written to the respective bit in the IRQC_INT_CLR register. To observe successive interrupts, the IRQC_IRQ_STS and/or IRQC_FIRQ_STS bits must be cleared before another interrupt event can be registered.

To avoid false interrupts, the input signals must be in their respective de-asserted logic states when the interrupts are enabled. Note that the input inversion must be considered when determining the de-asserted logic state.

When a rising-edge interrupt is enabled and unmasked, the corresponding interrupt status will be asserted if the relevant input signal is logic 1. Similarly, when a falling-edge interrupt is enabled and unmasked, the corresponding interrupt status will be asserted if the relevant input is logic 0. In other words, the behaviour is effectively level-triggered at the point when the interrupt is initially configured.

If necessary, the interrupt service routines should take account of the behaviour described above, and should clear the respective interrupt(s) immediately after they are enabled, before they are unmasked.

The control sequence below is recommended to ensure false interrupts are avoided.

- Mask the interrupt using the mask bits in IRQC_IRQ_MSK and IRQC_FIRQ_MSK
- Configure the IRQC interrupt registers (including the enable bits in IRQC_INT_CTRL)
- Clear the interrupt using the IRQC_INT_CLR register
- Unmask the interrupt using IRQC_IRQ_MSK and IRQC_FIRQ_MSK

IRQC MODULE INTERRUPTS

An input is considered part of the interrupt system when the associated enable bit in IRQC_INT_CTRL is set. The register is cleared at reset. Consequently, no input signals are considered interrupt sources at reset.

The interrupt inputs are configured as level sensitive or edge sensitive using the IRQC_EDGE1 and IRQC_EDGE0 registers.

Each input may be individually masked from the IRQ_N interrupt structure by setting the corresponding IRQC_IRQ_MSK bit. The Mask bits are set by default, so the IRQ_N interrupt structure is disabled until the corresponding bit is enabled (using IRQC_INT_CTRL) and unmasked (using IRQC_IRQ_MSK).

Each input may be individually masked from the FIRQ_N interrupt structure by setting the corresponding IRQC_FIRQ_MSK bit. The Mask bits are set by default, so the FIRQ_N interrupt structure is disabled until the corresponding bit is enabled (using IRQC_INT_CTRL) and unmasked (using IRQC_FIRQ_MSK).

When a valid level or edge is detected on an interrupt input, the corresponding IRQC_IRQ_STS and/or IRQC_FIRQ_STS bit is set (provided that the corresponding input is enabled and unmasked on the respective IRQ_N or FIRQ_N structure). The status (_STS) bits are latching bits, and are only cleared when a '1' is written to the respective bit in the IRQC_INT_CLR register.

The IRQC_IRQ_STS register provides readback of all the enabled and unmasked interrupts that are unmasked on IRQ_N structure. The IRQC_IRQ_VECT register provides a readback of the single, highest priority unmasked & asserted IRQ_N interrupt.

The IRQC_FIRQ_STS register provides readback of all the enabled and unmasked interrupts that are unmasked on FIRQ_N structure. The IRQC_FIRQ_VECT register provides a readback of the single, highest priority unmasked & asserted FIRQ_N interrupt.

Note that 'highest priority' is implemented as the interrupt in the lowest-numbered bit position of the IRQC_IRQ_STS or IRQC_FIRQ_STS register. For example, the GPIO Interrupt (in bit [1]), is given higher priority than the SPI Interrupt (in bit [2]).

When one or more bit in the IRQC_IRQ_STS register is set, the IRQ_N input to the HiFi EP™ is asserted. When one or more bit in the IRQC_FIRQ_STS register is set, the FIRQ_N input to the HiFi EP™ is asserted. The IRQ_N and FIRQ_N signals are Active Low; these signals are inverted (Active High) as inputs to the HiFi EP™ Core.

The IRQC interrupt control registers are illustrated in Figure 32.

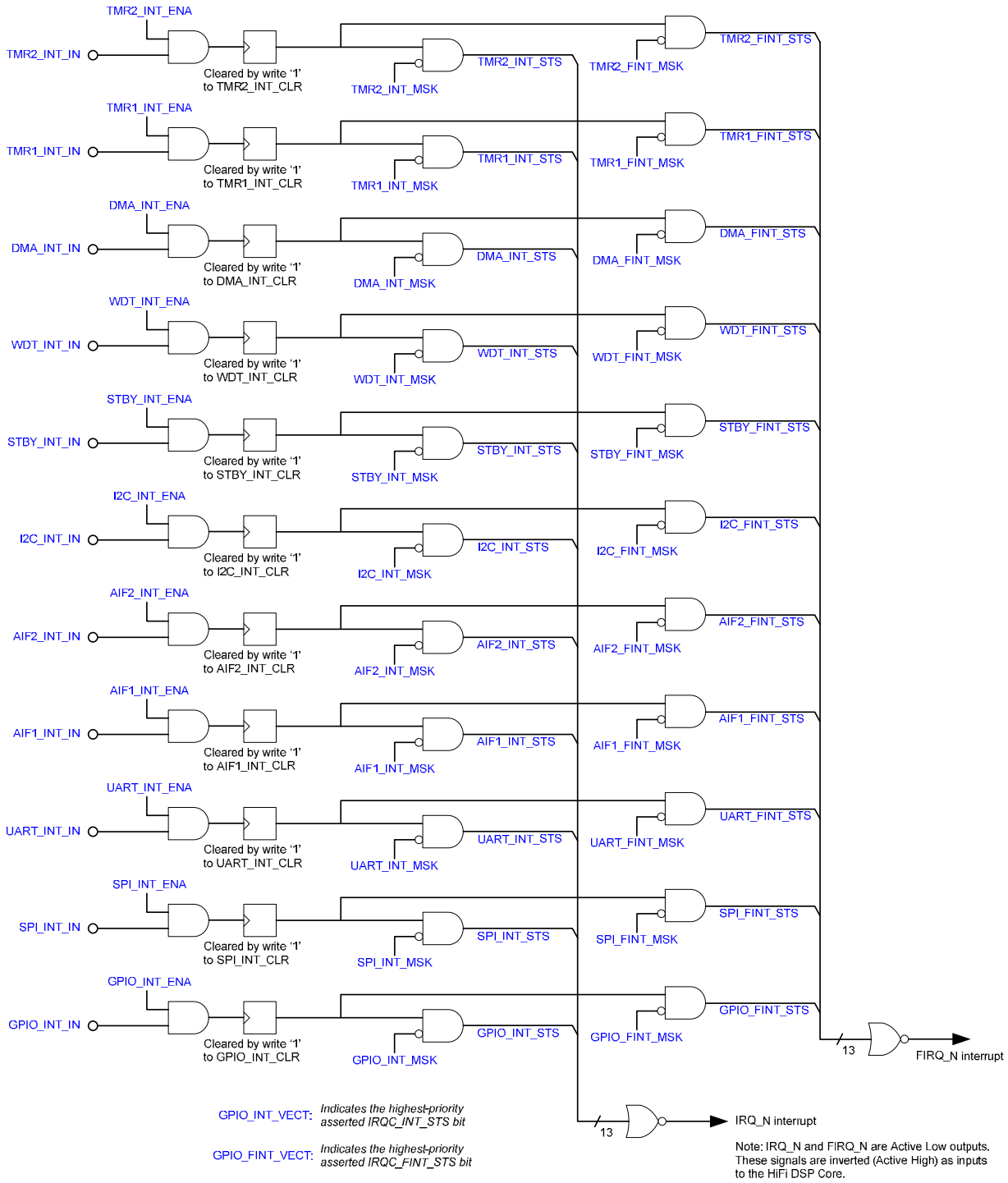


Figure 32 IRQC Interrupts

IRQC MODULE REGISTER MAP

ADDRESS	REGISTER	DESCRIPTION	RESET VALUE
Base + 0x00	IRQC_OUT	IRQ Output	0x0000_0000
Base + 0x04	IRQC_IN	IRQ Input	Undefined
Base + 0x08	IRQC_DIR	IRQ Direction	0x0000_0000
Base + 0x0C	IRQC_INV	IRQ Inversion	0x0000_0000
Base + 0x10	IRQC_EDGE0	IRQ Edge Detection 0	0x0000_0000
Base + 0x14	IRQC_EDGE1	IRQ Edge Detection 1	0x0000_0000
Base + 0x18	IRQC_INT_CTRL	IRQ Interrupt Control	0x0000_0000
Base + 0x1C	IRQC_INT_CLR	IRQ Interrupt Clear	0x0000_0000
Base + 0x20	IRQC_IRQ_MSK	IRQ Interrupt Mask	0xFFFF_FFFF
Base + 0x24	IRQC_IRQ_VECT	IRQ Interrupt Vector	0x0000_0000
Base + 0x28	IRQC_IRQ_STS	IRQ Interrupt Status	0x0000_0000
Base + 0x2C	IRQC_FIRQ_MSK	IRQ Fast Interrupt Mask	0xFFFF_FFFF
Base + 0x30	IRQC_FIRQ_VECT	IRQ Fast Interrupt Vector	0x0000_0000
Base + 0x34	IRQC_FIRQ_STS	IRQ Fast Interrupt Status	0x0000_0000

Table 79 IRQC Register Definition

IRQC_OUT – IRQ OUTPUT REGISTER

IRQC_OUT
IRQ OUTPUT REGISTER

Address = 0xF005_0000 **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:16	Reserved		0x0	
15	SW_INT2_OUT	RW	0x0	Controls the logic level of the SW_INT2 signal
14	SW_INT1_OUT	RW	0x0	Controls the logic level of the SW_INT1 signal
13		RW	0x0	Reserved - set to 0 only
12		RW	0x0	Reserved - set to 0 only
11		RW	0x0	Reserved - set to 0 only
10		RW	0x0	Reserved - set to 0 only
9		RW	0x0	Reserved - set to 0 only
8		RW	0x0	Reserved - set to 0 only
7		RW	0x0	Reserved - set to 0 only
6		RW	0x0	Reserved - set to 0 only
5		RW	0x0	Reserved - set to 0 only
4		RW	0x0	Reserved - set to 0 only
3		RW	0x0	Reserved - set to 0 only
2		RW	0x0	Reserved - set to 0 only
1		RW	0x0	Reserved - set to 0 only
0	IRQ_INT_OUT	RW	0x0	Controls the logic level of the $\overline{\text{IRQ}}$signal (external pin)

Table 80 IRQC_OUT Register

IRQC_IN – IRQ INPUT REGISTER

IRQC_IN IRQ INPUT REGISTER																															
Address = 0xF005_0004																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:16	Reserved		0x0	
15		RO	0x0	Reserved - reads back 0 only
14		RO	0x0	Reserved - reads back 0 only
13	Reserved		0x0	
12	Reserved		0x0	
11	TMR2_INT_IN	RO	0x0	Indicates the TMR2 (Timer 2) Interrupt logic level. Readback is after de-metastability logic and IRQC_INV inversion (if applicable).
10	TMR1_INT_IN	RO	0x0	Indicates the TMR1 (Timer 1) Interrupt logic level. Readback is after de-metastability logic and IRQC_INV inversion (if applicable).
9	DMA_INT_IN	RO	0x0	Indicates the DMA Interrupt logic level. Readback is after de-metastability logic and IRQC_INV inversion (if applicable).
8	WDT_INT_IN	RO	0x0	Indicates the WDT (Watchdog Timer) Interrupt logic level. Readback is after de-metastability logic and IRQC_INV inversion (if applicable).
7	STBY_INT_IN	RO	0x0	Indicates the $\overline{\text{STANDBY}}$ Interrupt logic level. Readback is after de-metastability logic and IRQC_INV inversion (if applicable).
6	I2C_INT_IN	RO	0x0	Indicates the I2C Interrupt logic level. Readback is after de-metastability logic and IRQC_INV inversion (if applicable).
5	AIF2_INT_IN	RO	0x0	Indicates the AIF2 Interrupt logic level. Readback is after de-metastability logic and IRQC_INV inversion (if applicable).
4	AIF1_INT_IN	RO	0x0	Indicates the AIF1 Interrupt logic level. Readback is after de-metastability logic and IRQC_INV inversion (if applicable).
3	UART_INT_IN	RO	0x0	Indicates the UART Interrupt logic level. Readback is after de-metastability logic and IRQC_INV inversion (if applicable).
2	SPI_INT_IN	RO	0x0	Indicates the SPI Interrupt logic level. Readback is after de-metastability logic and IRQC_INV inversion (if applicable).
1	GPIO_INT_IN	RO	0x0	Indicates the GPIO Interrupt logic level. Readback is after de-metastability logic and IRQC_INV inversion (if applicable).
0		RO	0x0	Reserved - reads back 0 only

Table 81 IRQC_IN Register

IRQC_DIR – IRQ DIRECTION REGISTER

IRQC_DIR IRQ DIRECTION REGISTER																															
Address = 0xF005_0008																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:16	Reserved		0x0	
15	SW_INT2_DIR	RW	0x0	Selects the SW_INT2 signal direction. 1 = output.
14	SW_INT1_DIR	RW	0x0	Selects the SW_INT1 signal direction. 1 = output.
13	Reserved		0x0	

IRQC_DIR IRQ DIRECTION REGISTER																															
Address = 0xF005_0008																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
12	Reserved		0x0	
11	TMR2_INT_DIR	RW	0x0	Selects the TMR2 (Timer 2) Interrupt direction. 0 = input.
10	TMR1_INT_DIR	RW	0x0	Selects the TMR1 (Timer 1) Interrupt direction. 0 = input.
9	DMA_INT_DIR	RW	0x0	Selects the DMA Interrupt direction. 0 = input.
8	WDT_INT_DIR	RW	0x0	Selects the WDT (Watchdog Timer) Interrupt direction. 0 = input.
7	STBY_INT_DIR	RW	0x0	Selects the $\overline{\text{STANDBY}}$ Interrupt direction. 0 = input.
6	I2C_INT_DIR	RW	0x0	Selects the I2C Interrupt direction. 0 = input.
5	AIF2_INT_DIR	RW	0x0	Selects the AIF2 Interrupt direction. 0 = input.
4	AIF1_INT_DIR	RW	0x0	Selects the AIF1 Interrupt direction. 0 = input.
3	UART_INT_DIR	RW	0x0	Selects the UART Interrupt direction. 0 = input.
2	SPI_INT_DIR	RW	0x0	Selects the SPI Interrupt direction. 0 = input.
1	GPIO_INT_DIR	RW	0x0	Selects the GPIO Interrupt direction. 0 = input.
0	IRQ_INT_DIR	RW	0x0	Selects the $\overline{\text{IRQ}}$ output pin direction. 1 = output.

Table 82 IRQC_DIR Register

IRQC_INV – IRQ INVERSION REGISTER

IRQC_INV IRQ INVERSION REGISTER																															
Address = 0xF005_000C																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:16	Reserved		0x0	
15		RW	0x0	Reserved - set to 0 only
14		RW	0x0	Reserved - set to 0 only
13	Reserved		0x0	
12	Reserved		0x0	
11	TMR2_INT_INV	RW	0x0	Selects the TMR2 (Timer 2) Interrupt input inversion. 0 = no inversion. 1 = inversion.
10	TMR1_INT_INV	RW	0x0	Selects the TMR1 (Timer 1) Interrupt input inversion. 0 = no inversion. 1 = inversion.
9	DMA_INT_INV	RW	0x0	Selects the DMA Interrupt input inversion. 0 = no inversion. 1 = inversion.
8	WDT_INT_INV	RW	0x0	Selects the WDT (Watchdog Timer) Interrupt input inversion. 0 = no inversion. 1 = inversion.
7	STBY_INT_INV	RW	0x0	Selects the $\overline{\text{STANDBY}}$ Interrupt input inversion. 0 = no inversion. 1 = inversion.
6	I2C_INT_INV	RW	0x0	Selects the I2C Interrupt input inversion. 0 = no inversion. 1 = inversion.
5	AIF2_INT_INV	RW	0x0	Selects the AIF2 Interrupt input inversion. 0 = no inversion. 1 = inversion.
4	AIF1_INT_INV	RW	0x0	Selects the AIF1 Interrupt input inversion. 0 = no inversion. 1 = inversion.

IRQC_INV IRQ INVERSION REGISTER																															
Address = 0xF005_000C																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
3	UART_INT_INV	RW	0x0	Selects the UART Interrupt input inversion. 0 = no inversion. 1 = inversion.
2	SPI_INT_INV	RW	0x0	Selects the SPI Interrupt input inversion. 0 = no inversion. 1 = inversion.
1	GPIO_INT_INV	RW	0x0	Selects the GPIO Interrupt input inversion. 0 = no inversion. 1 = inversion.
0		RW	0x0	Reserved - set to 0 only

Table 83 IRQC_INV Register

EDGE DETECTION

The interrupt inputs are configured as level sensitive or edge sensitive using the IRQC_EDGE1 and IRQC_EDGE0 registers, as described in Table 84.

IRQC_EDGE1	IRQC_EDGE0	DESCRIPTION
0	0	Level Sensitive
0	1	Leading Edge
1	0	Trailing Edge
1	1	Dual Edge Interrupt

Table 84 IRQC Edge Detection Control

In Level-sensitive configuration, the interrupt is asserted when the applicable logic level is detected. Active High is selected when IRQC_INV=0; Active Low is selected when IRQC_INV=1. Note that the interrupt cannot be cleared whilst the Active logic level is present (and unmasked) on the respective IRQC input.

In Edge-sensitive configuration, the interrupt is asserted when the applicable logic transition is detected. This may be the rising (leading) edge, falling (trailing) edge, or both edges. The active edge(s) will cause the respective IRQC_IRQ_STS and/or IRQC_FIRQ_STS bit to be set (as controlled by the respective mask bits). Note that the active edge(s) are inverted when IRQC_INV=1.

IRQC_EDGE0 – IRQ EDGE DETECTION 0 REGISTER

IRQC_EDGE0 IRQ EDGE DETECTION 0 REGISTER																															
Address = 0xF005_0010																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:16	Reserved		0x0	
15		RW	0x0	Reserved - set to 0 only
14		RW	0x0	Reserved - set to 0 only
13	Reserved		0x0	
12	Reserved		0x0	
11	TMR2_INT_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
10	TMR1_INT_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
9	DMA_INT_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
8	WDT_INT_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
7	STBY_INT_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
6	I2C_INT_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
5	AIF2_INT_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
4	AIF1_INT_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
3	UART_INT_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
2	SPI_INT_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
1	GPIO_INT_EDGE0	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE1
0		RW	0x0	Reserved - set to 0 only

Table 85 IRQC_EDGE0 Register

IRQC_EDGE1 – IRQ EDGE DETECTION 1 REGISTER

IRQC_EDGE1 IRQ EDGE DETECTION 1 REGISTER																															
Address = 0xF005_0014																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:16	Reserved		0x0	
15		RW	0x0	Reserved - set to 0 only
14		RW	0x0	Reserved - set to 0 only
13	Reserved		0x0	
12	Reserved		0x0	
11	TMR2_INT_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
10	TMR1_INT_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
9	DMA_INT_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
8	WDT_INT_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
7	STBY_INT_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
6	I2C_INT_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
5	AIF2_INT_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
4	AIF1_INT_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0

IRQC_EDGE1
IRQ EDGE DETECTION 1 REGISTER

Address = 0xF005_0014 **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
3	UART_INT_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
2	SPI_INT_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
1	GPIO_INT_EDGE1	RW	0x0	Selects Level or Edge interrupt detection, depending on *EDGE0
0		RW	0x0	Reserved - set to 0 only

Table 86 IRQC_EDGE1 Register

IRQC_INT_CTRL – IRQ INTERRUPT CONTROL REGISTER

IRQC_INT_CTRL
IRQ INTERRUPT CONTROL REGISTER

Address = 0xF005_0018 **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:16	Reserved		0x0	
15		RW	0x0	Reserved - set to 0 only
14		RW	0x0	Reserved - set to 0 only
13	Reserved		0x0	
12	Reserved		0x0	
11	TMR2_IRQC_ENA	RW	0x0	Enables TMR2 (Timer 2) as an input to the IRQC Interrupt logic
10	TMR1_IRQC_ENA	RW	0x0	Enables TMR1 (Timer 1) as an input to the IRQC Interrupt logic
9	DMA_IRQC_ENA	RW	0x0	Enables DMA as an input to the IRQC Interrupt logic
8	WDT_IRQC_ENA	RW	0x0	Enables WDT (Watchdog Timer) as an input to the IRQC Interrupt logic
7	STBY_IRQC_ENA	RW	0x0	Enables $\overline{\text{STANDBY}}$ as an input to the IRQC Interrupt logic
6	I2C_IRQC_ENA	RW	0x0	Enables I2C as an input to the IRQC Interrupt logic
5	AIF2_IRQC_ENA	RW	0x0	Enables AIF2 as an input to the IRQC Interrupt logic
4	AIF1_IRQC_ENA	RW	0x0	Enables AIF1 as an input to the IRQC Interrupt logic
3	UART_IRQC_ENA	RW	0x0	Enables UART as an input to the IRQC Interrupt logic
2	SPI_IRQC_ENA	RW	0x0	Enables SPI as an input to the IRQC Interrupt logic
1	GPIO_IRQC_ENA	RW	0x0	Enables GPIO as an input to the IRQC Interrupt logic
0		RW	0x0	Reserved - set to 0 only

Table 87 IRQC_INT_CTRL Register

IRQC_INT_CLR – IRQ INTERRUPT CLEAR REGISTER

IRQC_INT_CLR IRQ INTERRUPT CLEAR REGISTER																															
Address = 0xF005_001C																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:16	Reserved		0x0	
15		WO	0x0	Reserved - set to 0 only
14		WO	0x0	Reserved - set to 0 only
13	Reserved		0x0	
12	Reserved		0x0	
11	TMR2_IRQC_CLR	WO	0x0	Write ‘1’ to clear TMR2 (Timer 2) Interrupt (TMR2_IRQ_STS)
10	TMR1_IRQC_CLR	WO	0x0	Write ‘1’ to clear TMR1 (Timer 1) Interrupt (TMR1_IRQ_STS)
9	DMA_IRQC_CLR	WO	0x0	Write ‘1’ to clear DMA Interrupt (DMA_IRQ_STS)
8	WDT_IRQC_CLR	WO	0x0	Write ‘1’ to clear WDT (Watchdog Timer) Interrupt (WDT_IRQ_STS)
7	STBY_IRQC_CLR	WO	0x0	Write ‘1’ to clear $\overline{\text{STANDBY}}$ Interrupt (STBY_IRQ_STS)
6	I2C_IRQC_CLR	WO	0x0	Write ‘1’ to clear I2C Interrupt (I2C_IRQ_STS)
5	AIF2_IRQC_CLR	WO	0x0	Write ‘1’ to clear AIF2 Interrupt (AIF2_IRQ_STS)
4	AIF1_IRQC_CLR	WO	0x0	Write ‘1’ to clear AIF1 Interrupt (AIF1_IRQ_STS)
3	UART_IRQC_CLR	WO	0x0	Write ‘1’ to clear UART Interrupt (UART_IRQ_STS)
2	SPI_IRQC_CLR	WO	0x0	Write ‘1’ to clear SPI Interrupt (SPI_IRQ_STS)
1	GPIO_IRQC_CLR	WO	0x0	Write ‘1’ to clear GPIO Interrupt (GPIO_IRQ_STS)
0		WO	0x0	Reserved - set to 0 only

Table 88 IRQC_INT_CLR Register

IRQC_IRQ_MSK – IRQ INTERRUPT MASK REGISTER

IRQC_IRQ_MSK IRQ INTERRUPT MASK REGISTER																															
Address = 0xF005_0020																Default value = 0xFFFF_FFFF															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:16	Reserved		0x7	
15		RW	0x1	Reserved - set to 1 only
14		RW	0x1	Reserved - set to 1 only
13	Reserved		0x1	
12	Reserved		0x1	
11	TMR2_IRQ_MSK	RW	0x1	Selects whether TMR2 (Timer 2) Interrupt is masked. A masked interrupt will not trigger the TMR2_IRQ_STS bit, and is disabled from the IRQC_IRQ_VECT logic. 0 = Enabled; 1 = Masked.
10	TMR1_IRQ_MSK	RW	0x1	Selects whether TMR1 (Timer 1) Interrupt is masked. A masked interrupt will not trigger the TMR1_IRQ_STS bit, and is disabled from the IRQC_IRQ_VECT logic. 0 = Enabled; 1 = Masked.

IRQC_IRQ_MSK
IRQ INTERRUPT MASK REGISTER

Address = 0xF005_0020 **Default value = 0xFFFF_FFFF**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
9	DMA_IRQ_MSK	RW	0x1	Selects whether DMA Interrupt is masked. A masked interrupt will not trigger the DMA_IRQ_STS bit, and is disabled from the IRQC_IRQ_VECT logic. 0 = Enabled; 1 = Masked.
8	WDT_IRQ_MSK	RW	0x1	Selects whether WDT (Watchdog Timer) Interrupt is masked. A masked interrupt will not trigger the WDT_IRQ_STS bit, and is disabled from the IRQC_IRQ_VECT logic. 0 = Enabled; 1 = Masked.
7	STBY_IRQ_MSK	RW	0x1	Selects whether $\overline{\text{STANDBY}}$ Interrupt is masked. A masked interrupt will not trigger the STBY_IRQ_STS bit, and is disabled from the IRQC_IRQ_VECT logic. 0 = Enabled; 1 = Masked.
6	I2C_IRQ_MSK	RW	0x1	Selects whether I2C Interrupt is masked. A masked interrupt will not trigger the I2C_IRQ_STS bit, and is disabled from the IRQC_IRQ_VECT logic. 0 = Enabled; 1 = Masked.
5	AIF2_IRQ_MSK	RW	0x1	Selects whether AIF2 Interrupt is masked. A masked interrupt will not trigger the AIF2_IRQ_STS bit, and is disabled from the IRQC_IRQ_VECT logic. 0 = Enabled; 1 = Masked.
4	AIF1_IRQ_MSK	RW	0x1	Selects whether AIF1 Interrupt is masked. A masked interrupt will not trigger the AIF1_IRQ_STS bit, and is disabled from the IRQC_IRQ_VECT logic. 0 = Enabled; 1 = Masked.
3	UART_IRQ_MSK	RW	0x1	Selects whether UART Interrupt is masked. A masked interrupt will not trigger the UART_IRQ_STS bit, and is disabled from the IRQC_IRQ_VECT logic. 0 = Enabled; 1 = Masked.
2	SPI_IRQ_MSK	RW	0x1	Selects whether SPI Interrupt is masked. A masked interrupt will not trigger the SPI_IRQ_STS bit, and is disabled from the IRQC_IRQ_VECT logic. 0 = Enabled; 1 = Masked.
1	GPIO_IRQ_MSK	RW	0x1	Selects whether GPIO Interrupt is masked. A masked interrupt will not trigger the GPIO_IRQ_STS bit, and is disabled from the IRQC_IRQ_VECT logic. 0 = Enabled; 1 = Masked.
0		RW	0x1	Reserved - set to 1 only

Table 89 IRQC_IRQ_MSK Register

IRQC_IRQ_VECT – IRQ INTERRUPT VECTOR REGISTER

IRQC_IRQ_VECT
IRQ INTERRUPT VECTOR REGISTER

Address = 0xF005_0024 **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:5	Reserved		0x000_0000	
4:0	IRQC_IRQ_VECT	RO	0x00	Indicates which highest priority interrupt is currently asserted in the IRQC_IRQ_STS register. Highest priority is implemented as the interrupt in the lowest-numbered bit position of the IRQC_IRQ_STS register. The same priority is represented in the coding of IRQC_IRQ_VECT - the GPIO Interrupt is highest priority, and the TMR2 (Timer 2) Interrupt is lowest priority. 0x00 = No interrupt 0x01 = GPIO Interrupt 0x02 = SPI Interrupt 0x03 = UART Interrupt 0x04 =AIF1 Interrupt 0x05 =AIF2 Interrupt 0x06 =I2C Interrupt 0x07 = $\overline{\text{STANDBY}}$ Interrupt 0x08 = WDT (Watchdog Timer) Interrupt 0x09 = DMA Interrupt 0x0A = TMR1 (Timer 1) Interrupt 0x0B = TMR2 (Timer 2) Interrupt

Table 90 IRQC_IRQ_VECT Register

IRQC_IRQ_STS – IRQ INTERRUPT STATUS REGISTER

IRQC_IRQ_STS
IRQ INTERRUPT STATUS REGISTER

Address = 0xF005_0028 **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:16	Reserved		0x0	
15		RO	0x0	Reserved - reads back 0 only
14		RO	0x0	Reserved - reads back 0 only
13	Reserved		0x0	
12	Reserved		0x0	
11	TMR2_IRQ_STS	RO	0x0	TMR2 (Timer 2) Interrupt Status
10	TMR1_IRQ_STS	RO	0x0	TMR1 (Timer 1) Interrupt Status
9	DMA_IRQ_STS	RO	0x0	DMA Interrupt Status
8	WDT_IRQ_STS	RO	0x0	WDT (Watchdog Timer) Interrupt Status
7	STBY_IRQ_STS	RO	0x0	$\overline{\text{STANDBY}}$ Interrupt Status
6	I2C_IRQ_STS	RO	0x0	I2C Interrupt Status
5	AIF2_IRQ_STS	RO	0x0	AIF2 Interrupt Status
4	AIF1_IRQ_STS	RO	0x0	AIF1 Interrupt Status

IRQC_IRQ_STS IRQ INTERRUPT STATUS REGISTER																															
Address = 0xF005_0028																**Default value = 0x0000_0000**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
3	UART_IRQ_STS	RO	0x0	UART Interrupt Status
2	SPI_IRQ_STS	RO	0x0	SPI Interrupt Status
1	GPIO_IRQ_STS	RO	0x0	GPIO Interrupt Status
0		RO	0x0	Reserved - reads back 0 only

Table 91 IRQC_IRQ_STS Register

IRQC_FIRQ_MSK – IRQ FAST INTERRUPT MASK REGISTER

IRQC_FIRQ_MSK IRQ FAST INTERRUPT MASK REGISTER																															
Address = 0xF005_002C																**Default value = 0xFFFF_FFFF**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:16	Reserved		0x7	
15		RW	0x1	Reserved - set to 1 only
14		RW	0x1	Reserved - set to 1 only
13	Reserved		0x1	
12	Reserved		0x1	
11	TMR2_FIRQ_MSK	RW	0x1	Selects whether TMR2 (Timer 2) Interrupt is masked. A masked interrupt will not trigger the TMR2_FIRQ_STS bit, and is disabled from the IRQC_FIRQ_VECT logic. 0 = Enabled; 1 = Masked.
10	TMR1_FIRQ_MSK	RW	0x1	Selects whether TMR1 (Timer 1) Interrupt is masked. A masked interrupt will not trigger the TMR1_FIRQ_STS bit, and is disabled from the IRQC_FIRQ_VECT logic. 0 = Enabled; 1 = Masked.
9	DMA_FIRQ_MSK	RW	0x1	Selects whether DMA Interrupt is masked. A masked interrupt will not trigger the DMA_FIRQ_STS bit, and is disabled from the IRQC_FIRQ_VECT logic. 0 = Enabled; 1 = Masked.
8	WDT_FIRQ_MSK	RW	0x1	Selects whether WDT (Watchdog Timer) Interrupt is masked. A masked interrupt will not trigger the WDT_FIRQ_STS bit, and is disabled from the IRQC_FIRQ_VECT logic. 0 = Enabled; 1 = Masked.
7	STBY_FIRQ_MSK	RW	0x1	Selects whether $\overline{\text{STANDBY}}$ Interrupt is masked. A masked interrupt will not trigger the STBY_FIRQ_STS bit, and is disabled from the IRQC_FIRQ_VECT logic. 0 = Enabled; 1 = Masked.
6	I2C_FIRQ_MSK	RW	0x1	Selects whether I2C Interrupt is masked. A masked interrupt will not trigger the I2C_FIRQ_STS bit, and is disabled from the IRQC_FIRQ_VECT logic. 0 = Enabled; 1 = Masked.

IRQC_FIRQ_MSK IRQ FAST INTERRUPT MASK REGISTER																															
Address = 0xF005_002C																Default value = 0xFFFF_FFFF															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
5	AIF2_FIRQ_MSK	RW	0x1	Selects whether AIF2 Interrupt is masked. A masked interrupt will not trigger the AIF2_FIRQ_STS bit, and is disabled from the IRQC_FIRQ_VECT logic. 0 = Enabled; 1 = Masked.
4	AIF1_FIRQ_MSK	RW	0x1	Selects whether AIF1 Interrupt is masked. A masked interrupt will not trigger the AIF1_FIRQ_STS bit, and is disabled from the IRQC_FIRQ_VECT logic. 0 = Enabled; 1 = Masked.
3	UART_FIRQ_MSK	RW	0x1	Selects whether UART Interrupt is masked. A masked interrupt will not trigger the UART_FIRQ_STS bit, and is disabled from the IRQC_FIRQ_VECT logic. 0 = Enabled; 1 = Masked.
2	SPI_FIRQ_MSK	RW	0x1	Selects whether SPI Interrupt is masked. A masked interrupt will not trigger the SPI_FIRQ_STS bit, and is disabled from the IRQC_FIRQ_VECT logic. 0 = Enabled; 1 = Masked.
1	GPIO_FIRQ_MSK	RW	0x1	Selects whether GPIO Interrupt is masked. A masked interrupt will not trigger the GPIO_FIRQ_STS bit, and is disabled from the IRQC_FIRQ_VECT logic. 0 = Enabled; 1 = Masked.
0		RW	0x1	Reserved - set to 1 only

Table 92 IRQC_FIRQ_MSK Register

IRQC_FIRQ_VECT – IRQ FAST INTERRUPT VECTOR REGISTER

IRQC_FIRQ_VECT IRQ FAST INTERRUPT VECTOR REGISTER																															
Address = 0xF005_0030																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:5	Reserved		0x000_0000	
4:0	IRQC_FIRQ_VECT	RO	0x00	Indicates which highest priority interrupt is currently asserted in the IRQC_FIRQ_STS register. Highest priority is implemented as the interrupt in the lowest-numbered bit position of the IRQC_FIRQ_STS register. The same priority is represented in the coding of IRQC_FIRQ_VECT - the GPIO Interrupt is highest priority, and the TMR2 (Timer 2) Interrupt is lowest priority. 0x00 = No interrupt 0x01 = GPIO Interrupt 0x02 = SPI Interrupt 0x03 = UART Interrupt 0x04 = AIF1 Interrupt 0x05 = AIF2 Interrupt 0x06 = I2C Interrupt 0x07 = $\overline{\text{STANDBY}}$ Interrupt 0x08 = WDT (Watchdog Timer) Interrupt 0x09 = DMA Interrupt 0x0A = TMR1 (Timer 1) Interrupt 0x0B = TMR2 (Timer 2) Interrupt

Table 93 IRQC_FIRQ_VECT Register

IRQC_FIRQ_STS – IRQ FAST INTERRUPT STATUS REGISTER

IRQC_FIRQ_STS IRQ FAST INTERRUPT STATUS REGISTER																															
Address = 0xF005_0034																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:16	Reserved		0x0	
15		RO	0x0	Reserved - reads back 0 only
14		RO	0x0	Reserved - reads back 0 only
13	Reserved		0x0	
12	Reserved		0x0	
11	TMR2_FIRQ_STS	RO	0x0	TMR2 (Timer 2) Interrupt Status
10	TMR1_FIRQ_STS	RO	0x0	TMR1 (Timer 1) Interrupt Status
9	DMA_FIRQ_STS	RO	0x0	DMA Interrupt Status
8	WDT_FIRQ_STS	RO	0x0	WDT (Watchdog Timer) Interrupt Status
7	STBY_FIRQ_STS	RO	0x0	$\overline{\text{STANDBY}}$ Interrupt Status
6	I2C_FIRQ_STS	RO	0x0	I2C Interrupt Status
5	AIF2_FIRQ_STS	RO	0x0	AIF2 Interrupt Status
4	AIF1_FIRQ_STS	RO	0x0	AIF1 Interrupt Status
3	UART_FIRQ_STS	RO	0x0	UART Interrupt Status
2	SPI_FIRQ_STS	RO	0x0	SPI Interrupt Status
1	GPIO_FIRQ_STS	RO	0x0	GPIO Interrupt Status
0		RO	0x0	Reserved - reads back 0 only

Table 94 IRQC_FIRQ_STS Register

TRAX TRACE BUFFER MODULE

BASE ADDRESS 0xF006_0000

The TRAX module is a software debug facility. The associated Trace Memory (1024 x 32bit words) stores a record of HiFi2 EP™ DSP core instructions executed. Read/Write access to the debug data is supported via either the JTAG or APB interfaces.

The TRAX module implements the Program Trace using Traditional Branch Messaging (BTM). Specifically, it implements the following Nexus Public messages:

- Indirect Branch Message
- Synchronisation Message
- Indirect Branch with Synchronisation Message
- Correlation Message

The Trace Memory data is accessed via the TRAX_DATA register. The applicable memory address is selected using the TRAX_ADDR register, which is automatically incremented on each access, and wraps around when the last address is reached. Wrap-around status bits are also provided, in the event of the Trace Memory being filled. The TRAX_DATA register can only be accessed when the Trace function is inactive. The current memory address and the associated status bits are reset each time the Trace function is started.

The Trace function is enabled using TR_ENA. In a typical use case, it is stopped using a configurable function dependent on the Program Counter. The Trace function can be configured to continue recording events after a stop trigger condition; the number of additional events is configurable using the TRAX_DLY_CNT register.

A number of status flags provide readback of the TRAX module status. Note that the TRAX module does not generate any WM0011 Interrupt signals.

TRAX REGISTER MAP

The register map of the TRAX module is illustrated in Table 95.

ADDRESS	REGISTER	DESCRIPTION	RESET VALUE
Base + 0x00	TRAX_CONFIG	TRAX Trace Buffer Configuration	0x0000_0000
Base + 0x04	TRAX_CTRL	TRAX Control	0x0000_0C00
Base + 0x08	TRAX_STS	TRAX Status	0x0000_0C00
Base +0x0C	TRAX_DATA	TRAX Data	0x0000_0000
Base + 0x10	TRAX_ADDR	TRAX Address	0x0000_0000
Base + 0x14	TRAX_TRIG_PC	TRAX PC Match Trigger	0x0000_0000
Base + 0x18	TRAX_PC_MATCH	TRAX PC Match Control	0x0000_0000
Base + 0x1C	TRAX_DLY_CNT	TRAX Post-Trigger Delay Count	0x0000_0000

Table 95 TRAX Register Definition

TRAX_CONFIG – TRAX TRACE BUFFER CONFIGURATION REGISTER

The TRAX Trace Buffer can be accessed via the JTAG interface, or via the internal APB interface. Only one of these can be supported at any time - the selected method is determined by the TRAX_MODE bit.

When APB mode is selected, the APB_RST bit can be used to reset the TRAX module. When JTAG mode is selected, the TRAX module can be reset using the $\overline{\text{TRST}}$ pin.

Note that the TRAX clock enable bit (TRAX_CLK_ENA) is on the CCM_CLK_ENA register.

TRAX_CONFIG
TRAX TRACE BUFFER CONFIGURATION REGISTER

Address = 0xF0006_0000 **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:2	Reserved			
1	TRAX_MODE	RW	0x0	TRAX Access mode select 0 = JTAG control 1 = APB control
0	APB_RST	RW	0x0	TRAX Module Reset 0 = Not reset 1 = Reset Only valid if APB mode is selected (TRAX_MODE=1)

Table 96 TRAX_CONFIG Register

TRAX_CTRL – TRAX CONTROL REGISTER

The Trace function is enabled when the TR_ENA bit transitions from 0 to 1.

The Trace function can stopped using the configurable 'stop' trigger derived from the PC Match function (see below). If a 'stop' trigger is enabled and asserted, then the Trace function will either stop immediately, or will continue for a 'post-stop-trigger' period, configured via the CNTU control bit and the TRAX_DLY_CNT register (see Table 103).

Writing '0' to the TR_ENA bit before a 'stop' trigger has been asserted will disable the Trace function immediately. If a 'stop' trigger has been asserted, and the TR_ENA bit is set to '0' during the 'post-stop-trigger' period, then the Trace will continue until completion.

Note that the TR_ENA bit is not automatically reset when the Trace function stops. This bit must be set to 0 prior to initiating a new Trace.

The TRAX Control register allows configuration of the PC Match function.

The contents of the TRAX Control register can be read at any time. When the Trace function is active (indicated via the TRACT bit in the TRAX_STS register), then only the TR_ENA bit can be written to.

TRAX_CTRL
TRAX CONTROL REGISTER

Address = 0xF006_0004 | **Default value = 0x0000_0C00**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:15	Reserved		0x0_0000	
14:12	SMPER	RW	0x0	Synchronization Message Frequency Control Specifies the rate at which synchronization messages are recorded in the output trace. When 0, no periodic synchronization messages are recorded. 000 = No synchronization messages 001 = 1 synch. message every 256 messages 010 = 1 synch. message every 128 messages 011 = 1 synch. message every 64 messages 100 = 1 synch. message every 32 messages 101 = 1 synch. message every 16 messages 110 = 1 synch. message every 8 messages 111 = Reserved
11:10	Reserved		0x3	
9	CNTU	RW	0x0	Post-Stop-Trigger Count Control Selects which type(s) of Trace data will cause the CAPTURE_SIZE field (see Table 103) to decrement, 0 = decrement by 1 for every word written to trace memory 1 = decrement by 1 for every processor instruction executed, and for each exception and interrupt
8:3	Reserved		0x00	
2	PCM_ENA	RW	0x0	PC Match Trigger Control Enables the PC Match function as a Trace Stop Event trigger. 0 = Disabled 1 = Enabled Note that the PC Match function, when enabled, is also configured via the TRAX_TRIG_PC and TRAX_PC_MATCH control registers.
1	Reserved		0x0	
0	TR_ENA	RW	0x0	Trace Memory Enable The Trace function is enabled when TR_ENA transitions from 0 to 1. Writing '0' to this bit before a stop trigger has been asserted will disable the Trace function immediately. If this bit is set to '0' after a stop trigger has been asserted, then the Trace will continue until completion of the post-stop-trigger period (configured via the CNTU control bit and the TRAX_DLY_CNT register). Note this bit does not automatically reset when the Trace function stops; this bit must be set to 0 prior to initiating a new Trace. 0 = Disabled 1 = Enabled

Table 97 TRAX_CTRL Register

TRAX_STS – TRAX STATUS REGISTER

The TRAX Status register provides readback of the Trace module status.

The PCMTG bit provides readback indicating whether the PC Match function has generated a Trace Stop event. The TRIG bit provides an indication that one (or more) of the enabled Trace Stop events has been triggered.

The TRACT bit indicates the current status of the Trace function. Note that read/write access to the Trace memory and to most of the TRAX configuration bits is only possible when the Trace function is inactive (TRACT=0).

TRAX_STS TRAX STATUS REGISTER				
Address = 0xF006_0008				Default value = 0x0000_0C00
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0				
BITS	**FIELD NAME**	**S/W ACCESS**	**RESET VALUE**	**FIELD DESCRIPTION**
31:13	Reserved		0x0_0000	
12:8	MEMSZ	RO	0x0C	Trace Memory Size 0Ch = 4kB Note that although the memory size is measured in bytes, the trace memory is only accessible as 32-bit words Note that this is a READ ONLY register
7:3	Reserved		0x00	
2	PCMTG	RO	0x0	Program Count (PC) Match Trace Stop Event status 0 = Trace Stop Event not triggered by PC Match 1 = Trace Stop Event triggered by PC Match This bit is reset to 0 when TR_ENA transitions from 0 to 1.
1	TRIG	RO	0x0	Trace Stop Trigger status This bit is set whenever a Trace Stop condition is detected. Possible Trace Stop conditions include PC Match, or setting TR_ENA=0. This bit is reset 0 when TR_ENA transitions from 0 to 1.
0	TRACT	RO	0x0	Trace Active status TRACT is set to 1 (active) when TR_ENA transitions from 0 to 1. TRACT is set to 0 (inactive) following a stop condition. Note that the transition to 'inactive' may not be immediate, depending on the 'post-stop-trigger' period configuration. Read/write access to the Trace memory and to most of the TRAX configuration bits is only possible when the Trace function is inactive (TRACT=0). 0 = Trace is inactive 1 = Trace is active

Table 98 TRAX_STS Register

TRAX_DATA – TRAX DATA REGISTER

The Trace Memory is accessed via the TRAX_DATA register, which represents the 32-bit data word indexed by the TADDR field (in the TRAX_ADDR register).

Read or Write access to the TRAX_DATA register is only possible when the Trace function is inactive (TRACT=0).

Note that TADDR is auto-incremented after each Read or Write access to the TRAX_DATA register.

TRAX_DATA
TRAX DATA REGISTER

Address = 0xF006_000C																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	TRAX_DATA	RW	0x0000_0000	32-bit Trace RAM word, indexed by the TRAX_ADDR address register

Table 99 TRAX_DATA Register

TRAX_ADDR – TRAX ADDRESS REGISTER

The TADDR field within the TRAX_ADDR register controls the address within the Trace Memory for the next Read/Write access to the TRAX_DATA register.

The TADDR field is reset to 000h when TR_ENA transitions from 0 to 1, and is auto-incremented after each Read or Write access to the TRAX_DATA register.

If the TADDR field reaches its maximum value, then it will wrap-around to 000h after the next Read or Write access to TRAX_DATA. The number of wrap-arounds can be read from the TWRAP field.

If the TWRAP field reaches its maximum value, then it will also wrap-around to 0000h on the next increment. In this event, the TWSAT bit will be set, indicating saturation of the TWRAP field.

The TADDR and TWSAT fields are reset to 0 when TR_ENA transitions from 0 to 1.

The TRAX_ADDR register can be read at any time, but Write access is only possible when the Trace function is inactive (TRACT=0).

When a Trace stops, the TADDR register will indicate the next trace memory word to be written (ie. one greater than the last-written word). If no wrap-around has occurred (ie. TWRAP=0000h and TWSAT=0), then the captured trace comprises the Trace Memory words from index 000h up to TADDR-1 inclusive. If a wrap-around has occurred, then the captured trace comprises the Trace Memory words from TADDR to 3FFFh, followed by words from index 000h up to TADDR-1 inclusive.

TRAX_ADDR TRAX ADDRESS REGISTER																															
Address = 0xF006_0010																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	TWSAT	RW	0x0	Trace Memory Warp-Around Saturation Indicates that the number of Trace Memory wrap-arounds exceeds the maximum value of TWRAP. This field is reset 0 when TR_ENA transitions from 0 to 1.
30:24	Reserved		0x00	
23:10	TWRAP	RW	0x0000	Trace Memory Wrap-Around Count Indicates how many TADDR wrap-arounds have occurred during the current Trace. If the number of wrap-arounds exceeds the maximum value (3FFFh), then TWRAP will also wraps-around to 0000h on the next increment, and the TWSAT bit will be set. This field is reset 0000h when TR_ENA transitions from 0 to 1.
9:0	TADDR	RW	0x000	Trace Memory Address Index Controls the address within the Trace Memory for the next Read/Write access to the TRAX_DATA register. The index value is expressed in 32-bit words (not bytes). This field is reset to 000h when TR_ENA transitions from 0 to 1, and is auto-incremented after each Read or Write access to the TRAX_DATA register.

Table 100 TRAX_ADDR Register

TRAX_TRIG_PC – TRAX PC MATCH TRIGGER REGISTER

When the PC Match function is enabled as a Trace Stop Event trigger (PCM_ENA=1), the program count (PC) of the HiFi2 EP™ DSP core processor is monitored, and is used to generate a Stop condition for the Trace function.

The processor PC value is compared against the address held in the STOP_PC register. If a match is detected between the PC value (corresponding to the instruction about to be executed) and the STOP_PC value, then a Trace Stop condition will be triggered (TRIG=1, PCMTG=1).

Note that the PC Match function is also configurable using the control bits in the TRAX_PC_MATCH register (see Table 102). The configurable options allow some of the least significant bits of the PC value to be ignored, and enable a Stop condition to be generated when the PC value does not match STOP_PC.

TRAX_TRIG_PC TRAX PC MATCH TRIGGER REGISTER																															
Address = 0xF006_0014																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	STOP_PC	RW	0x0000_0000	Trace Program Count Match value This is the 32-bit address used to trigger a Trace Stop event when the Trace PC Match function is enabled (PCM_ENA=1).

Table 101 TRAX_TRIG_PC Register

TRAX_PC_MATCH – TRAX PC MATCH CONTROL REGISTER

When the PC Match function is enabled as a Trace Stop Event trigger (PCM_ENA=1), the processor PC value is compared against the address held in the STOP_PC register, and is used to generate a Stop condition for the Trace function.

Under default conditions, the PC Match stop condition is asserted when the PC value (corresponding to the instruction about to be executed) is equal to the STOP_PC register value.

The PCML field selects how many of the least significant bits of the PC value and STOP_PC value are ignored when identifying a PC Match condition.

The PCMS bit allows the matching logic to be inverted; it selects whether a Stop condition is generated when the PC value and STOP_PC values are the same, or when they are not the same. (Note that, in both cases, the match condition is also governed by the LSB mask function controlled by PCML.)

TRAX_PC_MATCH
TRAX PC MATCH CONTROL REGISTER

Address = 0xF006_0018 **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	PCMS	RW	0x0	Trace Program Count Match inversion Selects whether a PC Match Stop condition is triggered when the PC value and STOP_PC value are the same, or when they are not the same. (In both cases, the match condition is also subject to the mask length, PCML.) 0 = PC Match when PC value and STOP_PC are equal 1 = PC Match when PC value and STOP_PC are not equal
30:5	Reserved		0x000_0000	
4:0	PCML	RW	0x00	Trace Program Count Match mask length Selects how many least significant bits of the PC value and STOP_PC value are ignored when identifying a PC Match condition.

Table 102 TRAX_PC_MATCH Register

TRAX_DLY_CNT – TRAX POST-TRIGGER DELAY COUNT REGISTER

The CAPTURE_SIZE field controls how many trace events are recorded from the occurrence of a valid Stop Event trigger until the Trace function completes.

The CAPTURE_SIZE field should be set to the desired value prior to enabling the Trace. When a valid Stop Event trigger is detected, the CAPTURE_SIZE field will decrement as subsequent Trace data is recorded, until the CAPTURE_SIZE counter reaches zero.

The CNTU field in the TRAX_CTRL register (see Table 97) selects which type(s) of Trace data will cause the CAPTURE_SIZE counter to decrement during the 'post-stop-trigger' period.

At the end of the Trace, a final synchronisation message is recorded, and all internally buffered messages are flushed to the Trace RAM.

The TRAX_DLY_CNT register can be read at any time, but Write access is normally only possible when the Trace function is inactive (TRACT=0).

Writing to the TRAX_DLY_CNT is possible while the Trace function is active, but only when setting the CAPTURE_SIZE value to 0. If a valid Stop Event trigger has occurred, and the Trace function is executing the 'post-stop-trigger' phase, then writing 0x00_0000 to the CAPTURE_SIZE field will cause the Trace to stop immediately.

TRAX_DLY_CNT
TRAX POST-TRIGGER DELAY COUNT REGISTER

Address = 0xF006_001C | **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
30:24	Reserved		0x00	
23:0	CAPTURE_SIZE	RW	0x00 _0000	Trace Capture Size (Post-Stop-Trigger) Selects how many trace events are recorded from the occurrence of a valid Stop Event trigger until the Trace function completes. This field must be set to the desired value prior to enabling the Trace. Following a valid Stop Event trigger, the CAPTURE_SIZE field will decrement as subsequent Trace data is recorded. The Trace function stops when CAPTURE_SIZE reaches zero.

Table 103 TRAX_DLY_CNT Register

WATCHDOG TIMER (WDT) MODULE

BASE ADDRESS 0xF007_0000

WATCHDOG DESCRIPTION

The watchdog timer is enabled using WDT_ENA.

WDT_INT_ENA controls the assertion of Watchdog Timer Interrupt (to the Interrupt Module and to the HiFi2 EP™ DSP core) after the first occurrence of a watchdog timeout. WDT_INT_STS indicates that a watchdog timeout has caused an interrupt assertion.

WDT_RST_ENA controls the assertion of Watchdog Timer Reset signal (to the Reset controller) after the second occurrence of a watchdog timeout. The WDT_FLAG bit in the CCM_STATUS register (see Table 17) indicates that a watchdog timeout has occurred. Note that the Watchdog input to the Reset controller is selectable using the WDT_MSK bit, as described in the "Power-on and Reset Control" section.

Only the WDT_CTRL register (see Table 105) and the WDT_CNT_RESTART register (see Table 106) should be written while watchdog is running. No other watchdog registers should be written while watchdog is running.

All registers in the watchdog module are reset by a Warm Reset or a Hardware Reset unless otherwise stated.

Note that the watchdog clock enabling bit WDT_CLK_ENA is on the CCM_CLK_ENA register.

WATCHDOG TIMER INTERRUPT

The Watchdog Timer module can generate an interrupt when the timeout condition occurs.

The Watchdog Timer interrupt control registers are illustrated in Figure 33.

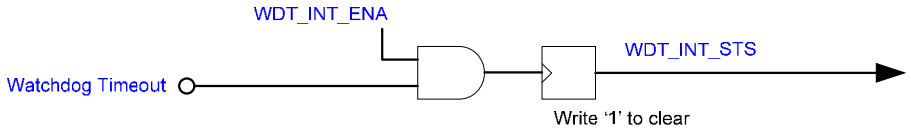


Figure 33 Watchdog Timer Interrupt

WATCHDOG REGISTER MAP

The register map of the Watchdog module is illustrated in Table 104.

ADDRESS	REGISTER	DESCRIPTION	RESET VALUE
Base + 0x00	WDT_CTRL	Watchdog Control	0x0000_0000
Base + 0x04	WDT_CNT_RESTART	Watchdog Counter Restart	0x0000_0000
Base + 0x08	WDT_MAX_CNT	Watchdog Maximum Count	0x0000_FFFF
Base + 0x0C	WDT_CUR_CNT	Watchdog Current Count	0x0000_FFFF
Base + 0x10	WDT_RST_LEN	Watchdog Reset Pulse Length	0x0000_00FF

Table 104 Watchdog Register Definition

WDT_CTRL – WATCHDOG CONTROL REGISTER

WDT_CTRL WATCHDOG CONTROL REGISTER																															
Address = 0xF007_0000																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:6	Reserved		0x000_0000	
5	WDT_INT_ENA	RW	0x0	Enables interrupt assertion after the first occurrence of a watchdog timeout 0 = Disabled 1 = Enabled
4	WDT_RST_ENA	RW	0x0	Enables watchdog reset assertion after the second occurrence of a watchdog timeout 0 = Disabled 1 = Enabled
3	Reserved		0x0	
2	WDT_INT_STS	R/W1C	0x0	Indicates when a watchdog timeout has caused a watchdog interrupt assertion 0 = No watchdog interrupt has been set 1 = Watchdog interrupt has been set by a watchdog timeout This bit is cleared by writing a '1'
1	Reserved		0x0	
0	WDT_ENA	RW	0x0	Watchdog Timer Enable 0 = Disabled 1 = Enabled

Table 105 WDT_CTRL Register

WDT_CNT_RESTART – WATCHDOG COUNTER RESTART REGISTER

WDT_CNT_RESTART WATCHDOG COUNTER RESTART REGISTER																															
Address = 0xF007_0004																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:1	Reserved		0x0000_0000	
0	WDT_RESTART	WO	0x0	Watchdog Timer Restart Writing a '1' restarts the watchdog timer

Table 106 WDT_CNT_RESTART Register

WDT_MAX_CNT – WATCHDOG MAXIMUM COUNT REGISTER

WDT_MAX_CNT holds the target count value (measured in APBCLK cycles). This represents the number of APBCLK cycles that are counted before watchdog times out.

WDT_MAX_CNT WATCHDOG MAXIMUM COUNT REGISTER																															
Address = 0xF007_0008																Default value = 0x0000_FFFF															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	WDT_MAX_CNT	RW	0x0000_FFFF	Count value (measured in APBCLK cycles) before watchdog times out.

Table 107 WDT_MAX_CNT Register

WDT_CUR_CNT – WATCHDOG CURRENT COUNT REGISTER

WDT_CUR_CNT WATCHDOG CURRENT COUNT REGISTER																															
Address = 0xF007_000C																Default value = 0x0000_FFFF															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	WDT_CUR_CNT	RO	0x0000_FFFF	The current counter value of the watchdog timer. When WDT_CUR_CNT reaches the value held in WDT_MAX_CNT, watchdog is activated. After de-assertion of $\overline{\text{RESET}}$ and whenever the watchdog timer is disabled (WDT_ENA = 0), the WDT_CUR_CNT value will reflect the value of the WDT_MAX_CNT register. This is a read only register

Table 108 WDT_CUR_CNT Register

WDT_RST_LEN – WATCHDOG RESET PULSE LENGTH REGISTER

WDT_RST_LEN controls the duration (pulse length) of the Watchdog Reset signal. This field represents the number of APBCLK cycles for which the Watchdog Reset signal is asserted (Active Low output to the Reset Controller).

WDT_RST_LEN WATCHDOG RESET PULSE LENGTH REGISTER																															
Address = 0xF007_0010																Default value = 0x0000_00FF															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:8	Reserved		0x00_0000	
7:0	WDT_RST_LEN	RW	0x0000_00FF	Number of APBCLK cycles for which the Watchdog Timer Reset signal is asserted (Active Low output to the Reset Controller).

Table 109 WDT_RST_LEN Register

UART MODULE

BASE ADDRESS 0xF008_0000

UART FEATURES

- Separate Transmit / Receive data buffers
- Buffer status flags and interrupts
- RX data error detection and interrupts
- Selectable Baud rate, derived from APBCLK
- Selectable parity, stop bit, word length configuration
- Loopback test function
- Boot Status and Error reporting

Data transmitted and received via the UART_DAT register. When FIFO mode is enabled, a 16-word buffer is enabled in the UART TX and UART RX paths. (Note that separate TX/RX buffers are implemented.)

Data transmission is selected by simply writing to the UART_DAT register. The UART TX buffer provides a TX_BUF_EMPTY flag, which indicates when the buffer is empty. An Interrupt function is also supported, indicating the TX buffer status.

Received data can be read from the UART_DAT register. The RX_BUF_STS flag indicates when the buffer contains new data. An Interrupt function indicates when the RX buffer status exceeds a configurable threshold. Buffer overflow and data error indications are also provided.

During boot-up, the WM0011 generates status and error codes for external monitoring of the start-up process. These status codes are reported via the UART interface, in the form of a single ASCII character code for each condition. See “Boot Sequence Control” for further details.

UART INTERRUPTS

The UART module can generate an interrupt in response to TX or RX Data Buffer conditions, and also in response to RX Error conditions.

The UART Line Status and Interrupt Control registers are illustrated in Figure 34.

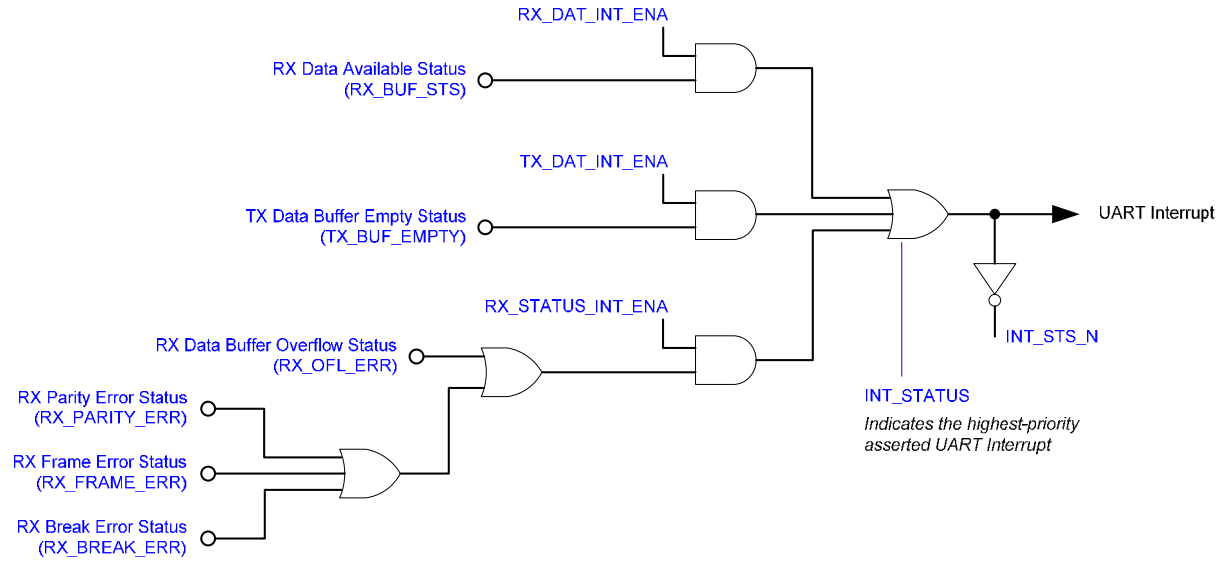


Figure 34 UART Interrupts

UART REGISTER MAP

This table illustrates the address map of the UART module.

ADDRESS	REGISTER	DESCRIPTION	RESET VALUE
Base + 0x00	UART_DAT	UART Data Register	0x00
Base + 0x04	UART_INT_CTRL	UART Interrupt Control Register	0x00
Base + 0x08 (Write)	UART_FIFO_CTRL	UART FIFO Control Register	0x00
Base + 0x08 (Read)	UART_INT_STATUS	UART Interrupt Status Register	0x00
Base + 0x0C	UART_LINE_CTRL	UART Line Control Register	0x00
Base + 0x10	UART_LOOPBACK_CTRL	UART Loopback Control Register	0x00
Base + 0x14	UART_LINE_STS	UART Line Status Register	0x00
Base + 0x00 (see note)	UART_BAUD_LSW	UART Baud LSW Register	0x01
Base + 0x04 (see note)	UART_BAUD_MSW	UART Baud MSW Register	0x00

Table 110 UART Register Definition

The UART_FIFO_CTRL and UART_INT_STATUS registers both exist at the same address; the applicable description depends on whether the register action is a Read or a Write operation.

The UART_BAUD_LSW and UART_BAUD_MSW registers are supported (instead of UART_DAT and UART_INT_CTRL respectively) when enabled using bit [7] of the UART_LINE_CTRL register.

UART_DAT - UART DATA REGISTER

UART_DAT
UART DATA REGISTER

Address = 0xF008_0000				Default value = 0x00
	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
7:0	UART_DAT	RW	0x00	READ - received data word from the RX data buffer. WRITE - write data word to be transmitted via the TX data buffer.

Table 111 UART_DAT Register

UART_INT_CTRL - UART INTERRUPT CONTROL REGISTER

UART_INT_CTRL
UART INTERRUPT CONTROL REGISTER

Address = 0xF008_0004				Default value = 0x00
	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
7:3	Reserved			
2	RX_STATUS_INT_ENA	RW	0x0	Enables the RX Line Status Interrupt. The RX Status Interrupt is triggered whenever an RX Overflow, Parity, Framing, or Transmission Break error is detected (RX_DAT_ERR=1 or RX_OFL_ERR=1) 0 = Disabled 1 = Enabled

UART_INT_CTRL UART INTERRUPT CONTROL REGISTER											
Address = 0xF008_0004										Default value = 0x00	
						7	6	5	4	3	2	1	0
BITS	**FIELD NAME**	**S/W ACCESS**	**RESET VALUE**	**FIELD DESCRIPTION**							
1	TX_DAT_INT_ENA	RW	0x0	Enables the TX Buffer Empty Interrupt. The TX Buffer Empty Interrupt is triggered whenever the TX data buffer is empty (TX_BUF_EMPTY=1). 0 = Disabled 1 = Enabled							
0	RX_DAT_INT_ENA	RW	0x0	Enables the RX Data Available Interrupt. The RX Data Available Interrupt is triggered whenever data in the RX FIFO reaches the threshold set by RX_FIFO_LIMIT. 0 = Disabled, 1 = Enabled							

Table 112 UART_INT_CTRL Register

UART_FIFO_CTRL - UART FIFO CONTROL REGISTER

The UART_FIFO_CTRL and UART_INT_STATUS registers both exist at the same address; the applicable description depends on whether the register action is a Read or a Write operation.

The UART_FIFO_CTRL register is defined in Table 113. Note that this definition is valid for register Write operations only.

UART_FIFO_CTRL UART FIFO CONTROL REGISTER											
Address = 0xF008_0008										Default value = 0x00	
						7	6	5	4	3	2	1	0
BITS	**FIELD NAME**	**S/W ACCESS**	**RESET VALUE**	**FIELD DESCRIPTION**							
7:6	RX_FIFO_LIMIT	W	0x0	Sets the RX FIFO limit at which the RX Data Available Interrupt is asserted. 0h = 1 word 1h = 4 words 2h = 8 words 3h = 14 words Only valid in FIFO mode (FIFO_ENA=1). Note that the RX FIFO buffer will hold a maximum of 16 words.							
5:3	Reserved	W									
2	TX_FIFO_FLUSH	W	0x0	Flushes the TX FIFO buffer 0 = Normal TX FIFO operation 1 = Flush TX FIFO Only valid in FIFO mode (FIFO_ENA=1). Note that the FIFO is automatically flushed whenever FIFO_ENA is changed.							
1	RX_FIFO_FLUSH	W	0x0	Flushes the RX FIFO buffer 0 = Normal RX FIFO operation 1 = Flush RX FIFO Only valid in FIFO mode (FIFO_ENA=1). Note that the FIFO is automatically flushed whenever FIFO_ENA is changed.							

UART_FIFO_CTRL UART FIFO CONTROL REGISTER										
Address = 0xF008_0008				Default value = 0x00						
				7	6	5	4	3	2	1	0
BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION							
0	FIFO_ENA	W	0x0	FIFO mode enable 0 = Disabled 1 = Enabled When FIFO mode is enabled, a 16-word buffer is provided in the UART TX and RX data paths. When FIFO mode is disabled, a 1-word buffer is implemented in the UART TX and RX data paths.							

Table 113 UART_FIFO_CTRL Register

UART_INT_STATUS - UART INTERRUPT STATUS REGISTER

The UART_FIFO_CTRL and UART_INT_STATUS registers both exist at the same address; the applicable description depends on whether the register action is a Read or a Write operation.

The UART_INT_STATUS register is defined in Table 114. Note that this definition is valid for register Read operations only.

UART_INT_STATUS UART INTERRUPT STATUS REGISTER										
Address = 0xF008_0008				Default value = 0x00						
				7	6	5	4	3	2	1	0
BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION							
7:6	FIFO_ENA_STS	R	0x0	UART FIFO Enable status 00 = Disabled 11 = Enabled All other codes are Reserved							
5:4	Reserved	R									
3:1	INT_STATUS	R	0x0	UART Interrupt Status Description 0h = Modem Status Change Interrupt (Priority 4) 1h = TX Buffer Empty Interrupt (Priority 3) 2h = RX Data Available Interrupt (Priority 2a) 3h = RX Line Status Interrupt (Priority 1) 6h = RX Timeout Interrupt (Priority 2b) - see note below Only valid when INT_STS_N=0. This field provides an indication of the highest-priority UART Interrupt. Priority '1' is highest priority. The RX Timeout Interrupt occurs if received data is not read from the UART_DAT register within a timeout period (equal to 4 x UART Character Period).							
0	INT_STS_N	R	0x0	UART Interrupt Status 0 = UART Interrupt is asserted 1 = UART Interrupt is not asserted Note that, when a UART Interrupt is asserted (INT_STS_N=0), the INT_STATUS field provides an indication of the highest priority enabled and asserted UART Interrupt.							

Table 114 UART_INT_STATUS Register

UART_LINE_CTRL - UART LINE CONTROL REGISTER

UART_LINE_CTRL UART LINE CONTROL REGISTER				
Address = 0xF008_000C				**Default value = 0x00**
				7 6 5 4 3 2 1 0
BITS	**FIELD NAME**	**S/W ACCESS**	**RESET VALUE**	**FIELD DESCRIPTION**
7	BAUD_REGS_CTRL	RW	0x0	UART Baud Rate register control 0 = Disabled 1 = Enabled This bit must be set to '1' in order to access the UART_BAUD_LSW and UART_BAUD_MSW registers.
6	TX_BREAK_ENA	RW	0x0	UART TX Break control 0 = Disabled 1 = Enabled (forces TX output low)
5:3	PARITY_BITS	RW	0x0	UART Parity select 0h = No parity 1h = Odd parity 3h = Even parity 5h = Parity bit set to '1' 7h = Parity bit set to '0'
2	STOP_BITS	RW	0x0	UART Stop Bit select 0 = 1 stop bit 1 = 1.5 stop bits (5 bit mode) or 2 stop bits (other modes)
1:0	WORD_LEN	RW	0x0	UART Word Length control 0h = 5 bits 1h = 6 bits 2h = 7 bits 3h = 8 bits

Table 115 UART_LINE_CTRL Register

UART_LOOPBACK_CTRL - UART LOOPBACK CONTROL REGISTER

UART_LOOPBACK_CTRL UART LOOPBACK CONTROL REGISTER				
Address = 0xF008_0010				**Default value = 0x00**
				7 6 5 4 3 2 1 0
BITS	**FIELD NAME**	**S/W ACCESS**	**RESET VALUE**	**FIELD DESCRIPTION**
7:5	Reserved			
4	UART_LOOPBACK_ENA	RW	0x0	Loopback mode control 0 = Disabled 1 = Enabled When Loopback mode is enabled, the UART TX data is not transmitted externally, but is copied internally to the UART RX data path instead. This is a test function.
3:0	Reserved			

Table 116 UART_LOOPBACK_CTRL Register

UART_LINE_STS - UART LINE STATUS REGISTER

The UART_LINE_STS register contains status bits indicating TX or RX Data Buffer conditions, and RX Error conditions. Many of these bits are inputs to the UART Interrupt function, as illustrated in Figure 34.

Note that, if an RX Error condition is detected, then the associated data word will be discarded. The applicable Error Status bit(s) will be set, and will remain set until a subsequent data word is successfully received.

UART_LINE_STS
UART LINE STATUS REGISTER

Address = 0xF008_0014 **Default value = 0x00**

					7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
7	RX_DAT_ERR	RO	0x0	RX Data Error Status 0 = No Error 1 = RX Parity, Framing, or Transmission Break error This bit is set to '1' when an RX Break Error, RX Frame Error, or RX Parity Error is detected. It can only be cleared to '0' when a subsequent data word is successfully received.
6	TX_IDLE_STS	RO	0x0	TX Idle Status 0 = TX Data buffer not empty and UART output is active 1 = TX Data buffer is empty and UART output is idle
5	TX_BUF_EMPTY	RO	0x0	TX Data Buffer Status 0 = TX Data buffer not empty 1 = TX Data buffer is empty
4	RX_BREAK_ERR	RO	0x0	RX Break Error Status 0 = No Error 1 = RX Break Error This bit is set to '1' when an RX Break Error is detected. It can only be cleared to '0' when a subsequent data word is successfully received.
3	RX_FRAME_ERR	RO	0x0	RX Framing Error Status 0 = No Error 1 = RX Framing Error This bit is set to '1' when an RX Frame Error is detected. It can only be cleared to '0' when a subsequent data word is successfully received.
2	RX_PARITY_ERR	RO	0x0	RX Parity Error Status 0 = No Error 1 = RX Parity Error This bit is set to '1' when an RX Parity Error is detected. It can only be cleared to '0' when a subsequent data word is successfully received.
1	RX_OFL_ERR	RO	0x0	RX Data Overflow Status 0 = No Error 1 = RX Overflow Error This bit is set to '1' when an RX Data Overflow Error is detected. It can only be cleared to '0' when a subsequent data word is successfully received.
0	RX_BUF_STS	RO	0x0	RX Data Buffer Status 0 = No RX data to read 1 = RX Data is available to read

Table 117 UART_LINE_STS Register

UART_BAUD_LSW - UART BAUD LSW REGISTER

Note that the Address of this register is the same as the UART_DAT register. The UART_BAUD_LSW register is only accessible when BAUD_REGS_CTRL=1 (see UART_LINE_CTRL register).

UART_BAUD_LSW UART BAUD LSW REGISTER				
Address = 0xF008_0000				Default value = 0x01
		7 6 5 4 3 2 1 0		
BITS	**FIELD NAME**	**S/W ACCESS**	**RESET VALUE**	**FIELD DESCRIPTION**
7:0	UART_BAUD [7:0]	RW	0x01	Least Significant Word (LSW) of the UART Baud Rate divisor. UART Baud Rate = [APBCLK frequency] / 16 x UART_BAUD.

Table 118 UART_BAUD_LSW Register

UART_BAUD_MSW - UART BAUD MSW REGISTER

Note that the Address of this register is the same as the UART_INT_CTRL register. The UART_BAUD_LSW register is only accessible when BAUD_REGS_CTRL=1 (see UART_LINE_CTRL register).

UART_BAUD_MSW UART BAUD MSW REGISTER				
Address = 0xF008_0004				Default value = 0x00
		7 6 5 4 3 2 1 0		
BITS	**FIELD NAME**	**S/W ACCESS**	**RESET VALUE**	**FIELD DESCRIPTION**
7:0	UART_BAUD [15:8]	RW	0x00	Most Significant Word (MSW) of the UART Baud Rate divisor. UART Baud Rate = [APBCLK frequency] / 16 x UART_BAUD.

Table 119 UART_BAUD_MSW Register

SERIAL PERIPHERAL INTERFACE (SPI) MODULE

BASE ADDRESS 0xF030_0000

SPI FEATURES

- Configurable Data/Clock phase and Clock polarity
- Data word length can be on 8, 16, 24, 32 or 64 bits
- Selectable data bit ordering (LSB first or MSB first)
- Polarity selection for the Slave Select ($\overline{\text{SPISS}}$) signal
- Programmable soft reset capability
- Selectable “auto-retransmit” mode
- Selectable “early-tx-data transition” mode
- Byte-packing options
- Multiple Transfer mode allowing multiple data words per $\overline{\text{SPISS}}$ assertion
- Master Mode Slave Select “shaping” (configurable $\overline{\text{SPISS}}$ set-up, hold and wait times)

SPI MASTER MODE

The SPI_MISO pin direction is Input.

The $\overline{\text{SPISS}}$, SPISCLK, and SPIMOSI pins are driven as Outputs, but only during an actual data transfer. After a master data transfer has completed, these signals are tri-stated. This allows for lower power usage, and for usage in a multi-master SPI scenario.

Note that the above behaviour can be adjusted using the SPI_MM_MODE register (see Table 121), which allows constant driving of these master mode output signals whenever the SPI block is enabled.

The SPI Master mode is selected by setting SPI_MODE=0. The user should configure the desired SPISCLK, $\overline{\text{SPISS}}$, and MISO/MOSI parameters, and lastly set SPI_ENA=1 to enable the SPI module.

The SPI module will then be in Master mode, and will initiate a SPI data transfer when data is written to the SPI_DAT data register. The outgoing SPI_DAT data is double-buffered, allowing for the queuing of the “next word” to be transferred, while the current word is being shifted out.

SPI SLAVE MODE

The $\overline{\text{SPISS}}$, SPISCK, and SPIMOSI pin direction is input.

The SPIMISO pin is driven as Output, but only during an actual data transfer. After a slave data transfer has completed (i.e. de-assertion of $\overline{\text{SPISS}}$ by the master), this signal is tri-stated. This allows for usage in a multi-slave SPI scenario.

The SPI Slave mode is selected by setting SPI_MODE=1. The user should configure the desired SPISCLK, $\overline{\text{SPISS}}$, and MISO/MOSI parameters, and lastly set SPI_ENA=1 to enable the SPI module.

The SPI module will then be in Slave mode, and will wait for a SPI data transfer from an external master. Once initiated, the incoming data bits are shifted in until one word is received. The incoming data word is placed in a holding register, allowing for the reception of the serial bits of a “new current word”, while the previous word is being queued for transfer to the AHB system side.

SPISCLK (CLOCK) CONFIGURATION

In SPI Master mode, the SPI Clock Divisor register SPI_SCLKDIV is used to control the frequency of SPISCLK. The register stores a 16-bit parameter that supplies the initial value for the clock generator counter. The derived frequency for SPISCLK is:

[AHBCLK frequency] / (SPI_SCLKDIV+1) * 2

In SPI Master mode, the maximum supported SPISCLK frequency is [AHBCLK frequency] / 8.

In SPI Slave mode, there is an asynchronous clock domain crossing between the incoming SPISCLK clock and the AHB clock, as well as $\overline{\text{SPISS}}$ detection that is synchronized to the AHB clock. The synchronization to the AHB clock domain places a restriction on the maximum rate of SPISCLK, and set-up and hold requirements on $\overline{\text{SPISS}}$.

In SPI Slave mode, the AHBCLK frequency must be faster than the SPISCLK frequency.

Software can select the SPICLK phase and polarity using the register bits MSTR_CLK_POL and MSTR_CLK_PHASE in Master mode, or SLV_CLK_POL and SLV_CLK_PHASE in Slave mode. These can be found in the SPI_CFG register.

The selectable options are illustrated in Figure 35 and Figure 36, showing an 8-bit SPI transfer.

When SLV_CLK_PHASE=0, the Slave begins sourcing the first bit of data as soon as $\overline{\text{SPISS}}$ is driven active. When MSTR_CLK_PHASE=0, the Master will drive the first bit out at the beginning of the clock cycle. The receiving device should sample the first data bit on the first transition of the clock.

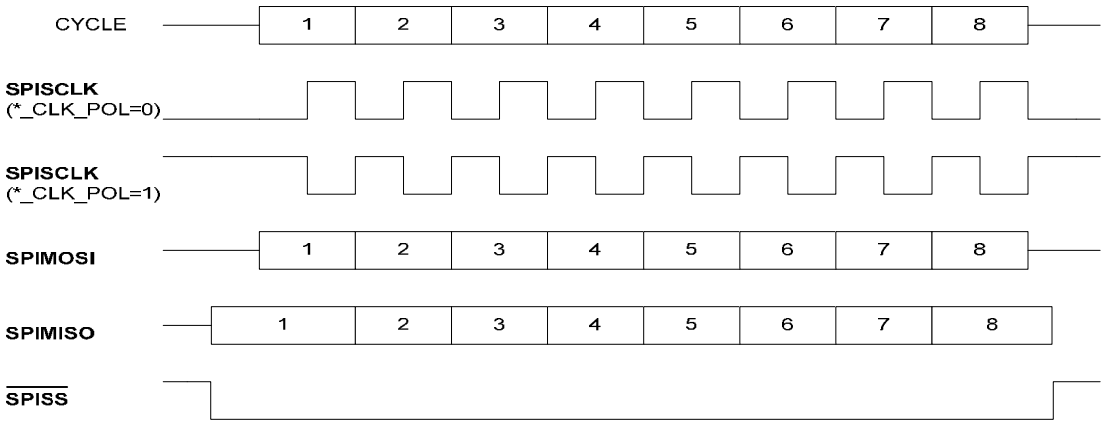


Figure 35 SPI Protocol with CLK_PHASE=0

When SLV_CLK_PHASE=1, the Slave begins sourcing the data as soon as $\overline{\text{SPISS}}$ is driven active. When MSTR_CLK_PHASE=1, the Master will drive the first bit out at the beginning of the cycle, corresponding to the first transition of the clock. The receiving device should sample the first data bit on the second transition of the clock.

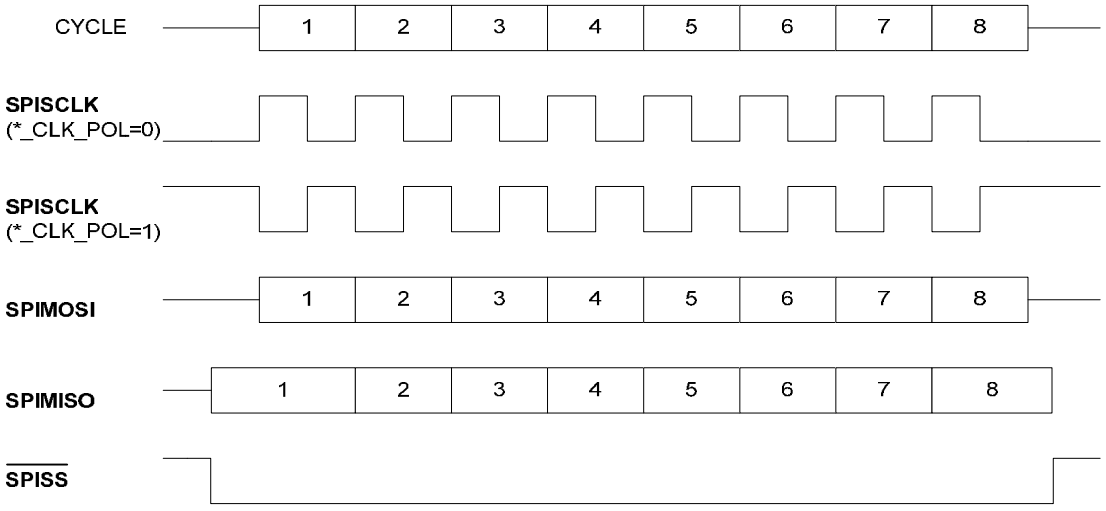


Figure 36 SPI Protocol with CLK_PHASE=1

MISO/MOSI (DATA) CONFIGURATION

The SPI_CFG configuration register contains control parameters for the MOSI/MISO data.

The BIT_ORDER register selects whether data is transmitted MSB-first or LSB-first.

The WL_1 [1:0] and WL_2 fields select the data word length. The word length can be 8, 16, 24, 32 or 64 bits. The transmission of WL bits is one "transfer". The SPI Slave Select ($\overline{\text{SPISS}}$) remains asserted for the entire word transfer length.

Note that 64-bit mode (WL_2=1) is only valid for DMA transfers.

SPISS (SLAVE SELECT) PROTOCOL

The SPI module supports two different $\overline{\text{SPISS}}$ protocols. These are the 'Single-Word' transfer mode and the 'Multiple Word' transfer modes.

In Master mode, the SPI_MT_ENA bit controls whether the $\overline{\text{SPISS}}$ signal de-asserts between each word transfer (single-word transfers), remains asserted over multiple word transfers.

Setting SPI_MT_ENA=1 selects the function of the SPI module continually asserting $\overline{\text{SPISS}}$ over multiple word transfers. When SPI_MT_ENA=0, the SPI module will treat each data transfer as a separate sequence of $\overline{\text{SPISS}}$ assertion, SPISCLK/MOSI data transfer, $\overline{\text{SPISS}}$ de-assertion.

Note that, in Slave mode, there is no unique concept of Multiple-Transfer mode (multiple data transfers per single $\overline{\text{SPISS}}$ assertion). The SPI module will simply accommodate whatever is presented on the SPI bus, and move a data word as soon as all the bits are received, regardless of whether $\overline{\text{SPISS}}$ is de-asserted between words.

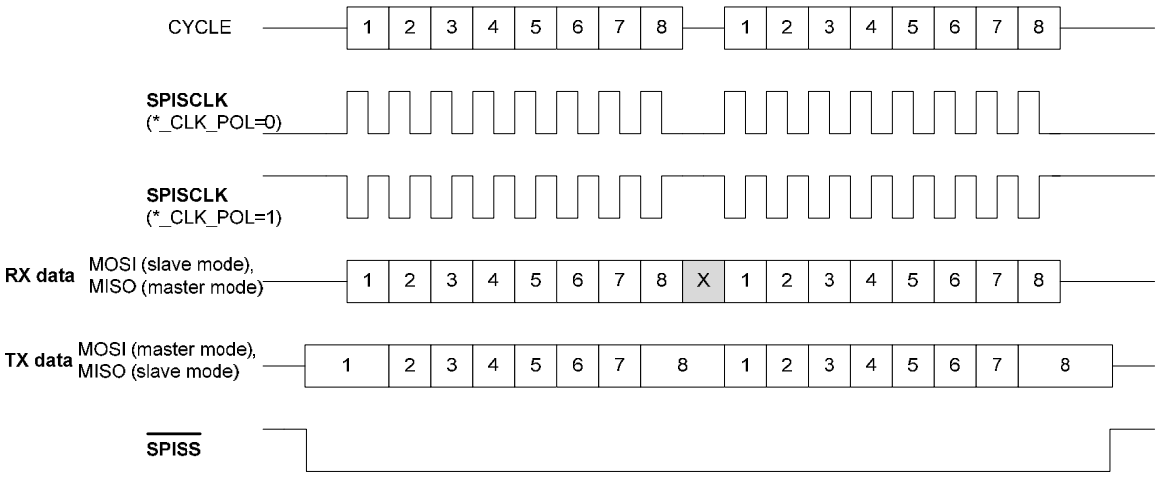
When 'Multiple Word' transfer mode is selected in Master mode, the $\overline{\text{SPISS}}$ signal will assert upon the first data transmission (same as in single-word mode). After completion of the first data word transmission, $\overline{\text{SPISS}}$ remains asserted. Subsequent writes to the SPI_DAT register simply perform further data word transmissions (SPISCLK/MOSI data transfers), and the SPI module remains in this state indefinitely.

When 'Multiple Word' transfer mode is selected in Master mode, there are three methods by which the $\overline{\text{SPISS}}$ line may be de-asserted:

1. De-select Multiple Transfer mode (set SPI_MT_ENA=0). Slave Select ($\overline{\text{SPISS}}$) is de-asserted and the Master Mode state machine is returned to 'Idle'. Subsequent SPI transfers will be single-word, unless the SPI_MT_ENA bit is once again set to '1'.
2. Re-arm the Multiple Transfer mode by writing a '1' to SPI_MT_IDLE. Slave Select ($\overline{\text{SPISS}}$) is de-asserted and the master mode state machine is returned to 'Idle'. This effectively does the same thing as (1) above, but without ever leaving the Multiple Transfer mode.
3. Select automatic re-arming of the Multiple Transfer mode each time the transfer count (SPI_BP_CNT) is reached. This is valid for byte-packing mode only, and must always be selected when Multiple Transfer mode and Byte-Packing modes are both enabled. This effectively does the same thing as (2) above, but without having to write to the SPI_CTRL register to de-assert $\overline{\text{SPISS}}$. This function is controlled via register bit BP_MT_ENA.

Note that the request to exit the Multiple Transfer mode is queued, and not immediate. If there is a current transfer in progress, or more outgoing data queued, then $\overline{\text{SPISS}}$ will remain asserted until the outgoing data transmission has completed. The request to exit or re-arm Multiple Transfer mode will occur after transmission of the queued data has completed.

The Single-Word transfer protocol is illustrated in Figure 35 and Figure 36. The Multiple-Word transfer protocol is illustrated in Figure 37.



*Note: The incoming (RX) And outgoing (TX) data is shown following the standard protocol of transitioning on 'Launch' edges of SPISCLK. This is configurable using the *_CLK_PHASE bits.*

Figure 37 Multiple Transfer Mode

SPISS (SLAVE SELECT) CONFIGURATION AND TIMING CONTROL

The SPI_SS_CFG register is used to control the $\overline{\text{SPISS}}$ signal protocol, allowing user selection of the $\overline{\text{SPISS}}$ signal polarity, set-up and hold timing between $\overline{\text{SPISS}}$ and SPISCLK, and the wait periods between back-to-back transfers.

SS_POL selects the polarity, which may be either Active-High or Active-Low $\overline{\text{SPISS}}$ assertion.

SS_SETUP determines the minimum wait-time from assertion of $\overline{\text{SPISS}}$ to the first SPISCLK transition. Note that the minimum setup time is also constrained as described in the "Signal Timing Requirements" section.

SS_HOLD determines the wait-time between the last SPISCLK transition and the de-assertion of $\overline{\text{SPISS}}$. Note that the minimum wait time time is also constrained as described in the "Signal Timing Requirements" section.

SS_WAIT determines the wait-time between successive data transfers in single-transfer mode (SPI_MT_ENA = 0). This parameter allows insertion of a chip select pause, to allow downstream slaves to offload their recently-received data.

SCLK_WAIT determines the wait-time between successive data transfers in multiple-transfer mode (SPI_MT_ENA = 1). Note that the $\overline{\text{SPISS}}$ signal is not de-asserted during the SCLK_WAIT period. This parameter allows insertion of a clock pause, to allow downstream slaves to offload their recently received data.

The $\overline{\text{SPISS}}$ signal control timings are measured in numbers of SPISCLK clock cycles, and are illustrated in Figure 38.

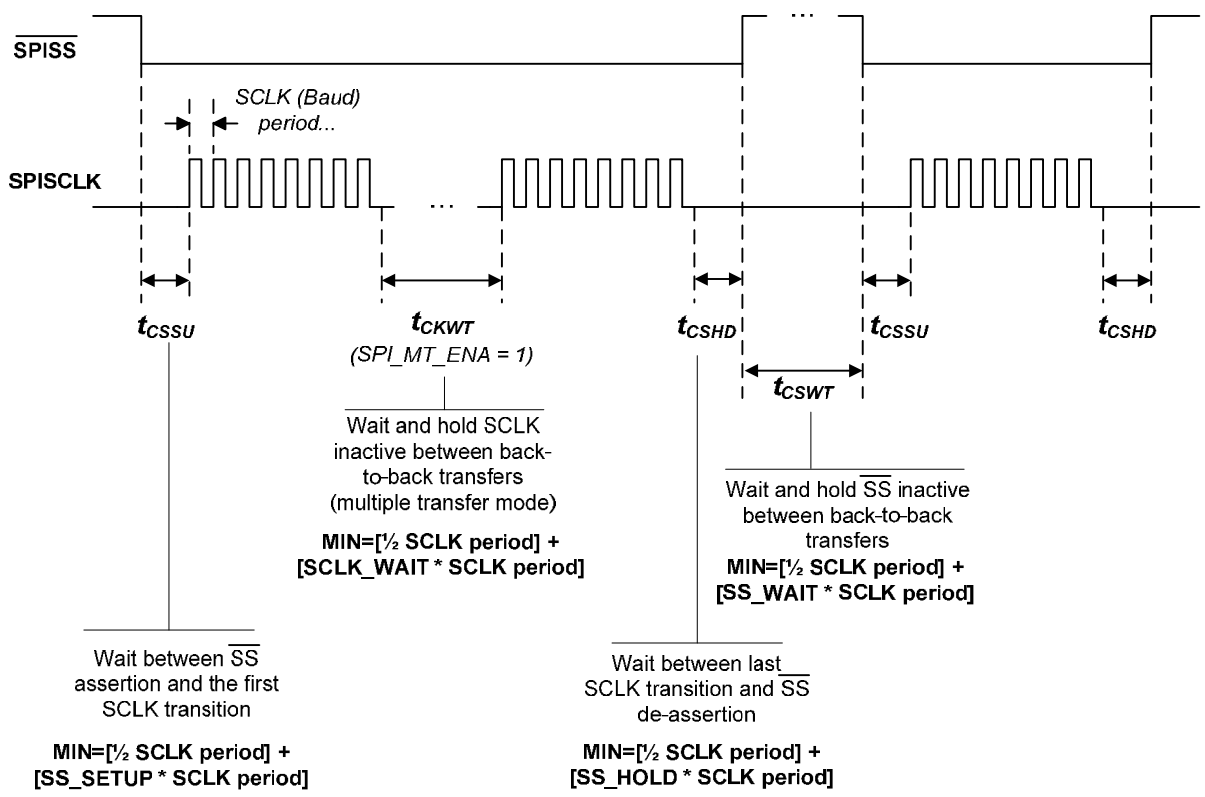


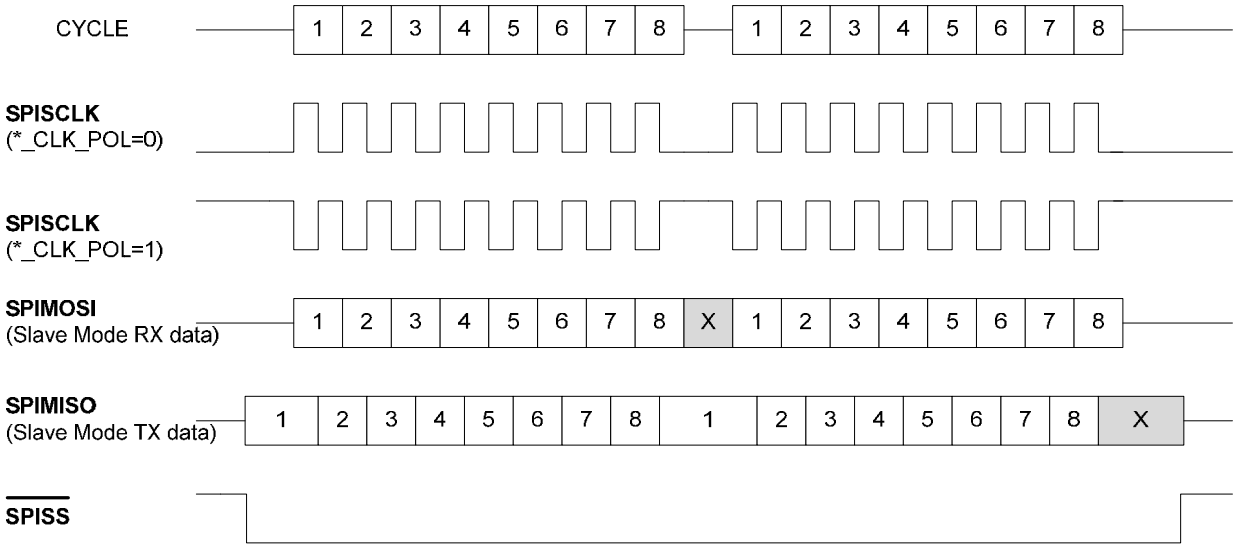
Figure 38 SPISS (Slave Select) Timing Diagram

EARLY TRANSMIT DATA PHASE

For circumstances where the SPI port is to be run at a high speed, and there is possibility of long delays between the launch edge and transition of transmitted data, an 'Early Transmit Data Phase' mode is provided.

The Early Transmit Data Phase mode is enabled by setting the TX_PHASE bit in the SPI_CFG register. This control bit affects the transmitted data (SPIMISO) in SPI Slave Mode. Note that the Early Transmit Data Phase mode is supported in SPI Slave Mode only.

When Early Transmit Data Phase mode is enabled, the effect is that the transmitted data transitions half an SPISCLK period early. This allows for a full period of setup time to the 'capture' SPISCLK edge, instead of only half a period of set-up time. The gain in set-up margin is countered by a loss in hold margin. Users should ensure appropriate setup and hold constraints at the ASIC level if this mode is to be used.



Note: The incoming (RX) data is shown following the standard protocol of transitioning on 'Launch' edges of SPISCLK. The outgoing (TX) data transitions half an SPISCLK cycle early, on the 'Capture' edges of SPISCLK.

Figure 39 Early Transmit Data Mode (SPI Slave Mode only)

AUTOMATED RE-TRANSMISSION OF DATA WORD

Upon detection of an Underclock (UCLK_ERR) error (see Table 124), the default behavior of the SPI module is to reset the bit counters and the transmit side holding buffers, assuming that software must re-load the word that did not complete transmission due to the UCLK_ERR error. Note that the receive side holding buffers are not reset, and contain the data word received from the last good transfer.

An optional mode is provided by setting the SPI_UCLK_MODE bit in the SPI_CTRL register. When set, the reset of the transmit side holding buffers due to UCLK_ERR error is disabled, and the word that did not complete transmission remains queued for transmit.

DOUBLE-BUFFERED TRANSMIT

A double-buffered transmit feature is provided, allowing support for slower SPISCLK rates, helping to ensure there is enough time for the transmit buffer architecture to queue up each word for transmission. This effectively makes two final-stage shift-register buffers, actively shifting one buffer while queuing data in the other.

This feature is controlled by the TX_DBL_BUF_ENA register bit (see Table 122).

SPI BYTE-PACKING

The SPI module interface to the AHB bus is 64-bit width. The external SPI data bus format typically uses smaller data widths (down to 8-bit size). To allow more efficient use of AHB bus bandwidth in cases where the SPI bus word size is small compared to the AHB bus width, a byte-packing feature is provided.

The byte packing process employs a data buffering stage, in which the 64-bit AHB bus width is filled with smaller-width SPI words. The precise packing format depends upon the applicable SPI word length.

In the case of SPI Receive (RX) path, SPI words are loaded into a buffer, which is transferred onto the AHB bus when sufficient SPI words have been 'packed'.

For SPI Transmit (TX), the 64-bit AHB data is loaded into a buffer, for transmission in smaller-sized blocks via the SPI protocol.

The SPI byte-packing feature is enabled by setting the BP_ENA control bit. When byte-packing is enabled, the 64-bit AHB bus width is packed as shown in Figure 40, according to the applicable SPI word length.

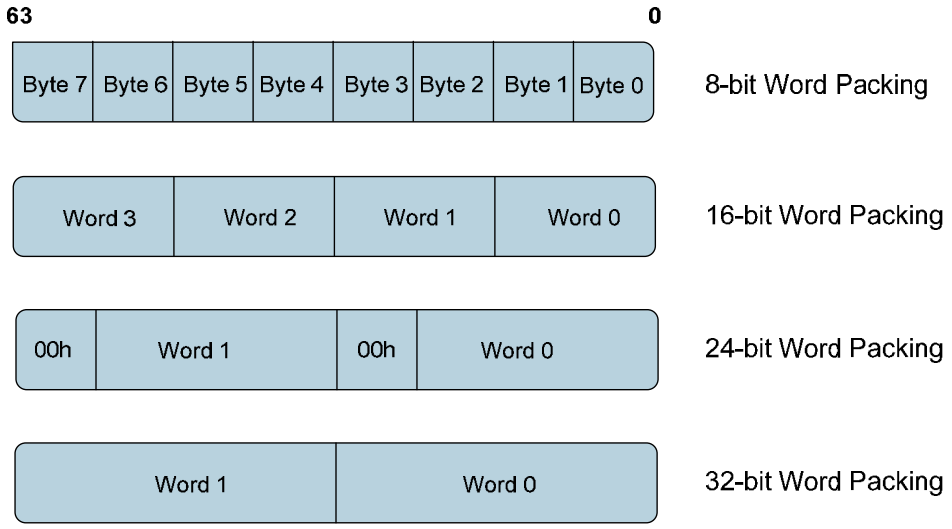


Figure 40 Byte Packing for different SPI Word Lengths

The Byte Packing State Machine will handle and control movement of data between the Byte Packing holding registers and the SPI holding registers. It also generates a specific Byte-Packing Interrupt and manipulates the DMA handshake signaling such that Interrupt requests and DMA requests are synchronized to the larger-capacity Byte Packing holding registers.

The state machine also handles instances where the total number of words to be packed does not fit precisely into full 64-bit AHB width; the user does not need to make any specific provision for this.

In SPI Master mode, the SPI_BP_CNT register is used to specify the total number of words to be transferred in Byte-Packed format. Note that, to avoid a lock-up, the number of words must be known and configured before the transfer commences.

In SPI Slave mode, the SPI_BP_CNT register provides readback of the number of words that have been transferred. The readback is only valid after the byte-packed transfer has completed, which is detected when $\overline{\text{SPISS}}$ is de-asserted.

In Master and Slave modes, the SPI_BP_CNT_RAW register provides readback of the number of words that have been transferred during the active transfer; the register can be read at any time during the transfer.

Note that, when Multiple Transfer mode is enabled in SPI Master mode, and Byte Packing is also enabled, the BP_MT_ENA register bit must be set to 1.

SPI DMA OPERATIONS

DMA operations associated with the SPI interface are controlled by the SPI_DMA_CTRL register.

For DMA handshake in Master or Slave modes, the SPI_DMA_CTRL register bits must be set for the desired operation:

The WR_RQST_ENA bit enables the DMA Write request handshake, which indicates the transmit buffer is empty and ready for more data.

The RD_RQST_ENA bit enables the DMA Read request handshake, which indicates the receive buffer is full and needs to be read.

In the case where byte-packing is disabled (BP_EN=0), the DMA requests are based on the normal buffer status (empty, full) – ie. mimics the function of the CYC_DONE status.

When byte-packing is enabled (BP_EN=1), the DMA requests are based on the packed BP 64-bit buffer status (empty, full), OR on the determination that the byte-packed transfer is done – ie. it mimics the function of the BP_DONE status.

The CYC_DONE and BP_DONE registers are held within the SPI_STATUS register.

SPI CONTROL SEQUENCES

Typical control sequences for SPI data transmission are illustrated on the following pages.

Note that the different figures illustrate the configurable handling of the Underclock Error (UCLK_ERR) condition, which is selectable as described in Table 124.

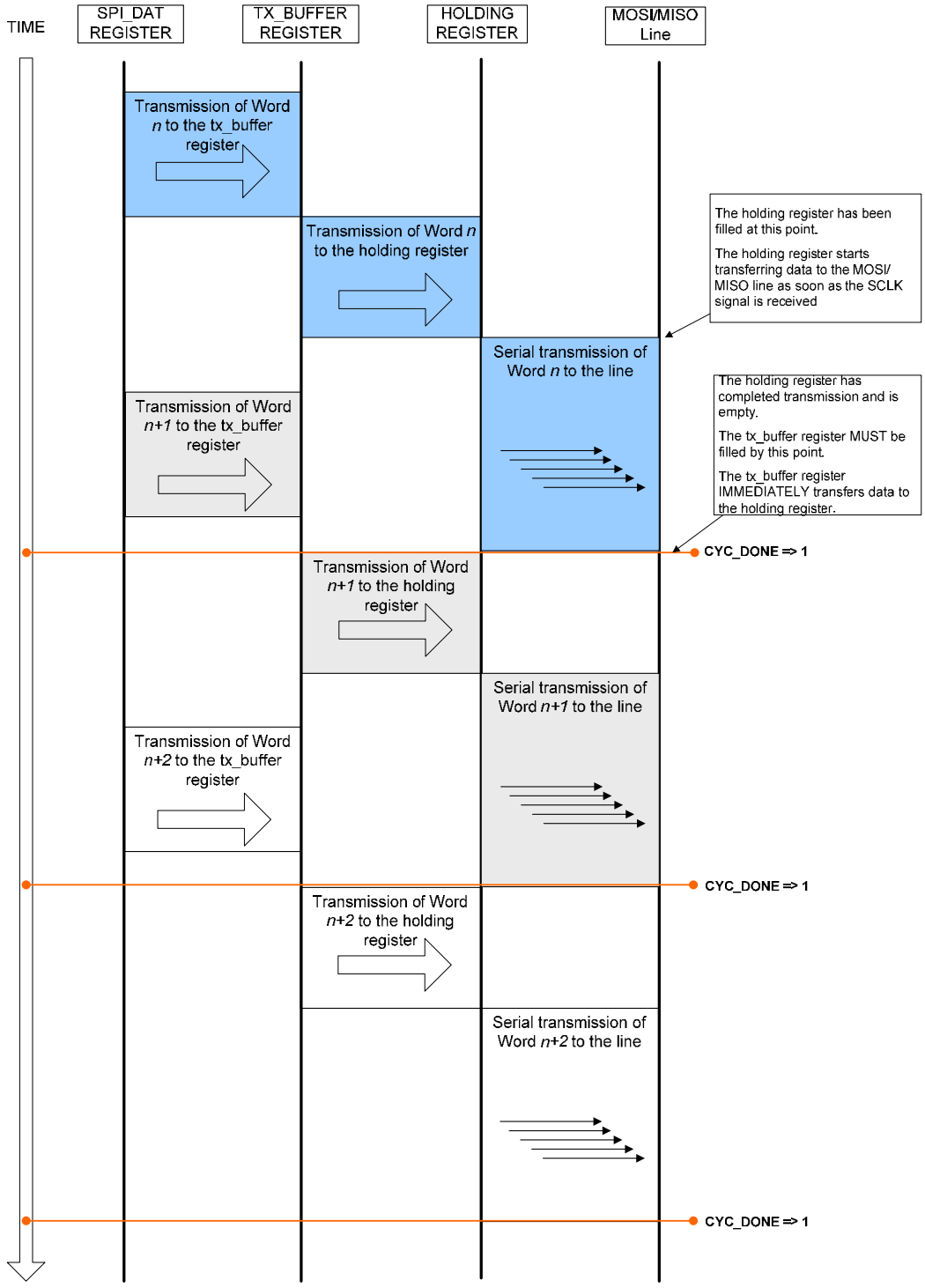
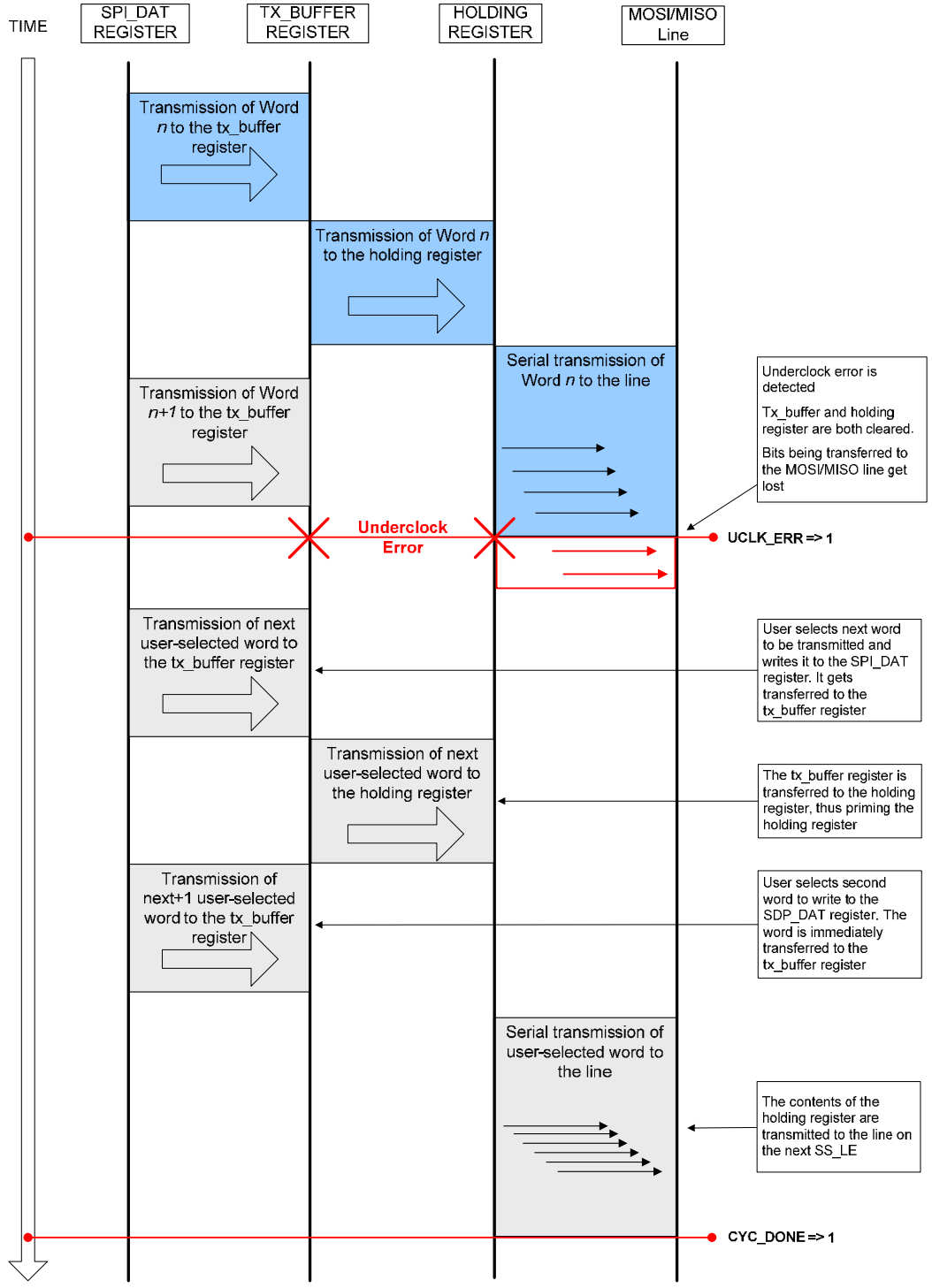


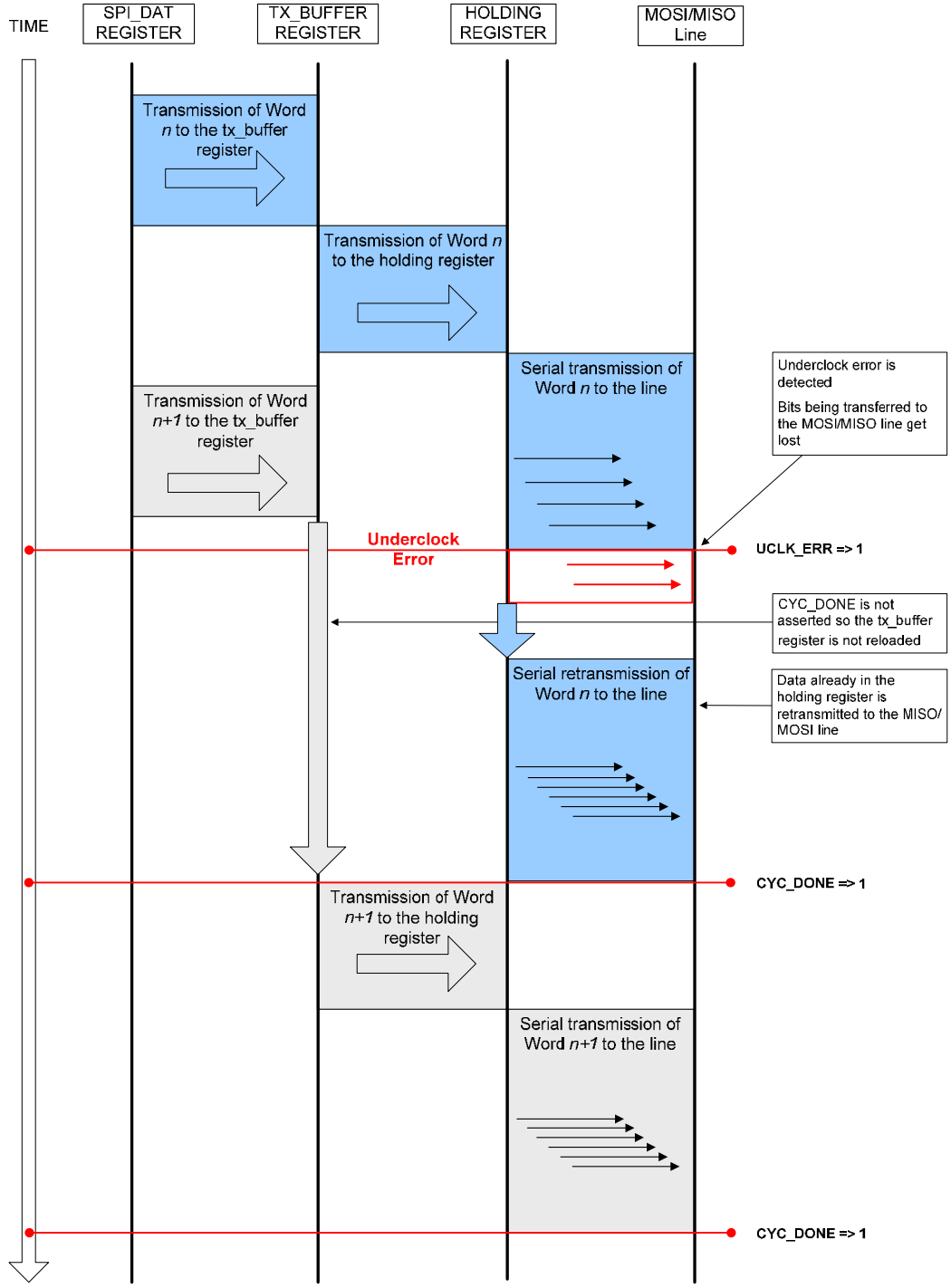
Figure 41 Normal SPI Transmission – No Errors



Underclock error - **UCLK_ERR = 0**

Following an Underclock Error, the user determines what data to send next. Both tx_buffer and the holding register must be primed before the next CYC_DONE. It is recommended that this is done during the UCLK_ERR interrupt.

Figure 42 SPI Transmission with Underclock Error, UCLK_ERR=0



Underclock error - **UCLK_ERR = 1**

Following an Underclock Error, the data in both the tx_buffer register and the holding register are retained, and the word being transmitted at the time of the Underclock error is re-transmitted to the MOSI/MISO output.

Figure 43 SPI Transmission with Underclock Error, UCLK_ERR=1

SPI INTERRUPTS

The SPI module can generate an interrupt when any of the conditions described in the SPI_STATUS register occurs. The interrupt conditions provide status indications of the SPI bus transactions, and are summarised below.

- TX_UFL_ERR (Write Underflow Error): the outgoing data buffer did not get loaded with new data since the last transmission, and another transmission began.
- RX_OFL_ERR (Read Overflow Error): the incoming data buffer did not get off-loaded since the last reception, and was overwritten with another incoming data word.
- SS_LE (Leading Edge): the assertion of $\overline{\text{SPISS}}$ was detected.
- SS_TE (Trailing Edge): the de-assertion of $\overline{\text{SPISS}}$ was detected.
- CYC_DONE: a transfer cycle of one word (WL bits) has completed.
- UCLK_ERR (Underclock Error): the de-assertion of $\overline{\text{SPISS}}$ occurred with fewer than WL bits sent/received.
- BP_DONE: a transfer of 'n' words in a byte-packed transfer has completed, indicating that the Byte Packing holding register is full (RX) or empty (TX).

The SPI_INT_STS bit is the logical OR of the enabled status bits. For the interrupt to propagate (to the Interrupt Module and to the HiFi2 EP™ DSP core), the SPI_INT_ENA bit must also be set.

The SPI interrupt control registers are illustrated in Figure 44.

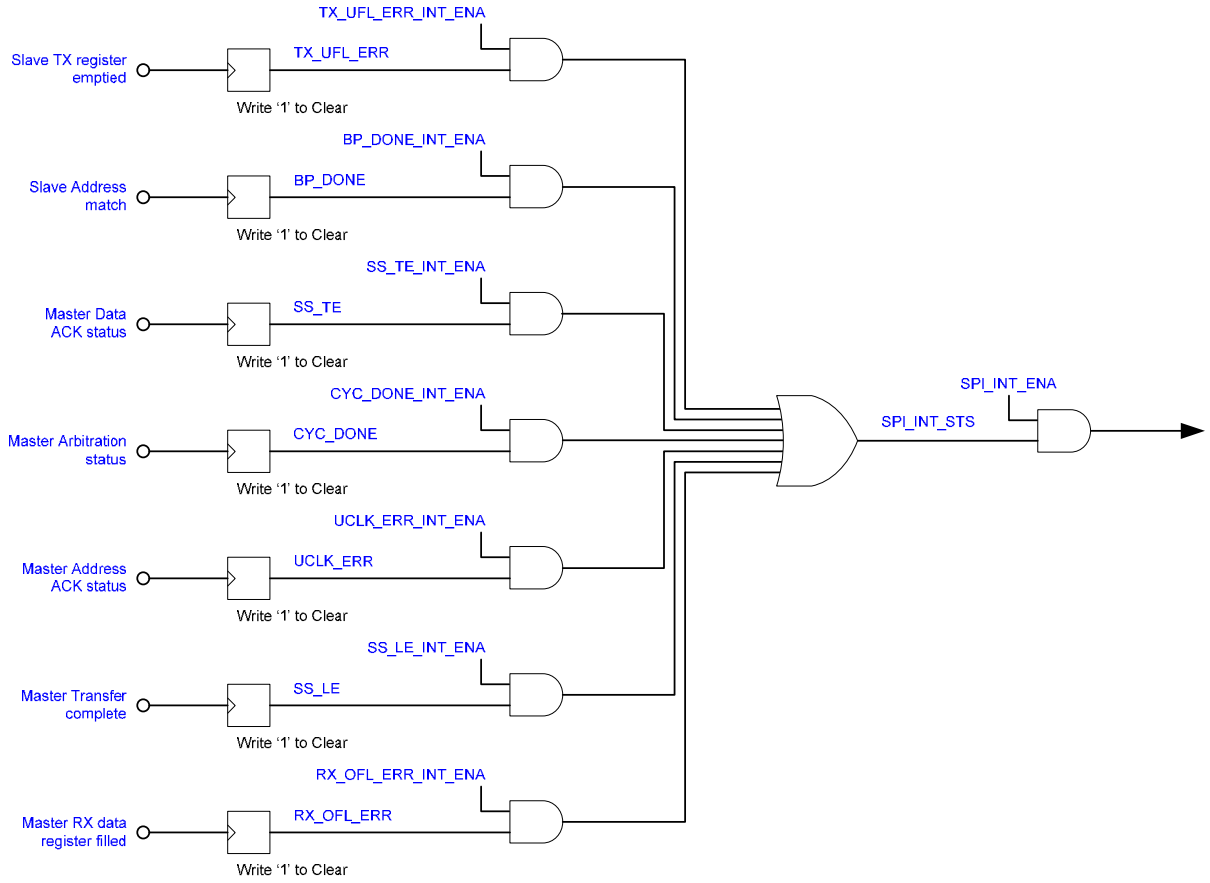


Figure 44 SPI Interrupts

SPI REGISTER MAP

This table illustrates the address map of the AHB SPI module

ADDRESS	REGISTER	DESCRIPTION	RESET VALUE
Base + 0x00	SPI_CTRL	SPI Control	0x0000_0000
Base + 0x08	SPI_CFG	SPI Configuration	0x0000_0200
Base + 0x10	SPI_SCLKDIV	SPI Clock Division	0x0000_0008
Base + 0x28	SPI_STATUS	SPI Status	0x0000_0000
Base + 0x30	SPI_SS_CFG	SPI Slave Select Configuration	0x0000_0000
Base + 0x38	SPI_DAT	SPI Data	0x0000_0000
Base + 0x40	SPI_INT_CTRL	SPI Interrupt Control	0x0000_0034
Base + 0x48	SPI_DMA_CTRL	SPI DMA Control	0x0000_0000
Base + 0x50	SPI_BP_CNT	SPI Byte Pack Word Count	0x0000_0000
Base + 0x58	SPI_BP_CNT_RAW	SPI Byte Pack Raw Word Count	0x0000_0000

Table 120 SPI Register Definition

SPI_CTRL – SPI CONTROL REGISTER

SPI_CTRL
SPI CONTROL REGISTER

Address = 0xF030_0000 | **Default value = 0x0000_0000**

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

Bits	Field Name	S/W Access	Reset Value	Field Description
31:9	Reserved		0	
8	SPI_UCLK_MODE	RW	0x0	Selects the response to an Underclock Error condition: 0 = On a UCLK_ERR error, the transmit holding register is reset, and the word that was transmitting when the UCLK error occurred is lost. 1 = On a UCLK_ERR error, the transmit holding register does not get reset, so the word that was transmitting when the UCLK error occurred remains queued.
7	SPI_MM_MODE	RW	0x0	Multi-Master mode (Valid in SPI Master Mode only): 0 = Multi-Master mode, always tri-state the $\overline{\text{SPISS}}$, SPISCLK and SPIMOSI lines when a transfer is complete. 1 = Single-Master mode, always drive the $\overline{\text{SPISS}}$, SPISCLK and SPIMOSI lines
6	SPI_MT_ENA	RW	0x0	Multiple Transfer Mode (Valid in SPI Master Mode only): 0 = Generate single transfers (de-assert $\overline{\text{SPISS}}$ after each transfer). 1 = Generate multiple transfers within a single $\overline{\text{SPISS}}$ assertion.
5	SPI_MT_IDLE	WO	0x0	Multiple Transfer re-arm (Valid in SPI Master Mode only): Writing a '1' to this bit re-arms the Multiple Transfer operation by de-asserting the $\overline{\text{SPISS}}$ Slave Select and returning the master mode state machine to Idle.
4	Reserved		0x0	
3	SPI_MODE	RW	0x0	SPI Mode select: 0 = Master SPI mode 1 = Slave SPI mode
2	SPI_LOOPBACK_ENA	RW	0x0	Internal Loopback Mode: 0 = Normal operation. 1 = Serial input is linked to serial output line (internal signaling; does not traverse the chip I/O bidirectional buffers).

SPI_CTRL SPI CONTROL REGISTER																															
Address = 0xF030_0000																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Bits	Field Name	S/W Access	Reset Value	Field Description
1	SPI_INT_ENA	RW	0x0	SPI Interrupt Enable; selects whether SPI_INT_STS will cause an interrupt, or not. 0 = Disable the interrupt line. 1 = Enable the interrupt line to the CPU.
0	SPI_ENA	RW	0x0	SPI Module Enable 0 = Disabled 1 = Enabled

Table 121 SPI_CTRL Register

SPI_CFG – SPI CONFIGURATION REGISTER

SPI_CFG SPI CONFIGURATION REGISTER																															
Address = 0xF030_0008																Default value = 0x0000_0200															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:14	Reserved		0	
13	BP_MT_ENA	RW	0x0	Byte-packing in Multiple Transfer mode 0 = Disabled 1 = Enabled This bit must be set to 1 when Byte Packing and Multiple Transfer modes are both enabled in SPI Master Mode. (Byte Packing mode is selected using BP_ENA; Multiple Transfer mode is selected using SPI_MT_ENA; SPI Master mode is selected using SPI_MODE.)
12	BP_ENA	RW	0x0	Byte-packing mode 0 = Disabled 1 = Enabled
11	Reserved		0x0	
10	TX_DBL_BUF_ENA	RW	0x0	Transmit Double-Buffer mode: 0 = Disable double-buffer; single shift register buffer and single queuing buffer 1 = Enable double-buffer; “ping-pong” on shift register transmit output path (keep up with back-to-back words at faster HCLK:SCK ratios)
9	TX_PHASE	RW	0x1	Early Transmit Data Phase: 0 = Disabled - normal transmit data phase, transitions are on the launch edge of SPISCLK 1 = Enabled - output data transitions occur half an SPISCLK period sooner than the normal protocol (i.e. transition on capture edge instead of launch edge) Note that Early Transmit mode is only supported in SPI Slave Mode. This bit should be set to 0 in SPI Master Mode.

SPI_CFG SPI CONFIGURATION REGISTER																															
Address = 0xF030_0008																	Default value = 0x0000_0200														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
8	WL_2	RW	0x0	Word Length (64-bit extension), use with WL_1 [1:0]: 0 = Use WL_1 [1:0] to determine word length 1 = Select 64-bit word length Note that 64-bit mode (WL_2=1) is only valid for DMA transfers.
7	SLV_CLK_PHASE	RW	0x0	Slave Mode Clock Phase: Selects the timing of the SPIMOSI and SPIMISO data bits in SPI Slave mode. The definition describes the timing of the first data bit only; subsequent data bits are aligned with the corresponding edge (ie. rising or falling) of successive SPISCLK cycles. 0 = Valid on the first SPISCLK transition after SPISS asserted 1 = Valid on the second SPISCLK transition after SPISS asserted
6	SLV_CLK_POL	RW	0x0	Slave Mode Clock Polarity: 0 = SPISCLK is Low (0) in its inactive state 1 = SPISCLK is High (1) in its inactive state
5	MSTR_CLK_PHASE	RW	0x0	Master Mode Clock Phase: Selects the timing of the SPIMOSI and SPIMISO data bits in SPI Master mode. The definition describes the timing of the first data bit only; subsequent data bits are aligned with the corresponding edge (ie. rising or falling) of successive SPISCLK cycles. 0 = Valid on the first SPISCLK transition after SPISS asserted 1 = Valid on the second SPISCLK transition after SPISS asserted
4	MSTR_CLK_POL	RW	0x0	Master Mode Clock Polarity: 0 = SPISCLK is Low (0) in its inactive state 1 = SPISCLK is High (1) in its inactive state
3	BIT_ORDER	RW	0x0	Least Bit First: 0 = Data is MSB-first 1 = Data is LSB-first
2	SPI_RESET	RW	0x0	Module Reset: 0 = Normal operation 1 = Force soft reset of module. The internal state machines are reset; Status register is cleared; However, the soft reset doesn't affect control register values.
1:0	WL_1	RW	0x0	Word Length select: 00 = 8-bit word length 01 = 16-bit word length 10 = 24-bit word length 11 = 32-bit word length

Table 122 SPI_CFG Register

SPI_SCLKDIV – SPI CLOCK DIVISION REGISTER

For custom-programmed devices, the start-up (default) contents of this register are configured in the fuse memory, and can be set according to the application requirements.

Note that this register can be written and updated prior to software download using the "PLL Configuration Download" code packet.

SPI_SCLKDIV
SPI CLOCK DIVISION REGISTER

Address = 0xF030_0010 **Default value = 0x0000_0008**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:16	Reserved		0	
15:0	SPI_SCLKDIV	RW	0x0008	Clock Divisor for master mode SPISCLK generation. This field sets the number of AHBCLK cycles in each phase of the SPISCLK output. 0000h = 1 clock cycle 0001h = 2 clock cycles 0002h = 3 clock cycles … FFFFh = 65536 clock cycles AHBCLK : SPISCLK frequency ratio = (SPI_SCLKDIV+1) * 2 This field must be set to 3 or higher.

Table 123 SPI_SCLKDIV Register

SPI_STATUS – SPI STATUS REGISTER

The SPI_STATUS register is defined in Table 124.

Four status bits in the SPI_STATUS register give an indication of the status of each word transferred. The table below shows some typically expected status, assuming that the status bits are cleared to '0' by software after being read (i.e. reset the status for each transfer).

SS_LE	UCLK_ERR	CYC_DONE	SS_TE	INTERPRETATION
1	0	1	0	First word of a transfer completed – it may be a multiple-word transfer, or it may be a single-word transfer and the chip select has yet to be de-asserted.
1	0	1	1	Single-word transfer completed and chip select has already de-asserted
0	0	0	1	Chip select was de-asserted.
1	1	0	1	Underclock error, single-word transfer (not enough bits sent prior to de-assertion of chip select)
0	0	1	0	Subsequent word of a multiple-word transfer completed – there may be more words to come, or it may be the last word and the chip select has yet to be de-asserted.
0	0	1	1	Last word of a multiple-word transfer completed and chip select de-asserted
0	1	0	1	Underclock error on subsequent or last word of a multiple-word transfer
x	1	1	1	Underclock error on the 'next' word transfer, before s/w could clear the CYC_DONE status of the previous word. This could occur because Underclock detection is possible within one bit time of the previous successful transfer.

SPI_STATUS
SPI STATUS REGISTER

Address = 0xF030_0028 | **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:15	Reserved		0	
14:12	SPI_CURRENT_STS	RO	0x0	Raw status of the SPI master state machine's "current_state" register: 000 = IDLE 001 = CSSETUP 010 = TRANSFER 011 = CSHOLD 100 = CSWAIT 101 = CKWAIT 110 = MTRANS 111 = CSBEGIN
11	Reserved		0x0	
10	RX_BUF_FULL	RO	0x0	Raw indicator of Rx incoming holding register status 0 = No data in holding register 1 = Holding register contains a valid data word
9	TX_BUF_FULL	RO	0x0	Raw indicator of Tx outgoing holding register status 0 = Holding register ready for new data word 1 = Tx Buffer is full
8	TX_UFL_ERR	R/W1C	0x0	Write Underflow Error indication: indicates that the outgoing data buffer did not get loaded with new data since the last transmission, and another transmission began. 0 = No Write Underflow since this bit was cleared 1 = Write Underflow detected This bit is cleared by writing a '1' to it. This bit is a sticky status bit, and will set upon meeting the condition regardless of the state of its corresponding interrupt enable TX_UFL_ERR_INT_ENA.
7	BP_DONE	R/W1C	0x0	Byte Packing Transfer Done: indicates a request for more packed data. This bit is set when the Byte Packing Holding Register is full (RX) and empty (TX), or when the current transfer of multiple byte-packed words is complete. 0 = Byte Packing Transfer is not complete 1 = Byte Packing Transfer is complete Only valid if byte packing is enabled (BP_EN = 1). This bit is cleared by writing a '1' to it. This bit is a sticky status bit, and will set upon meeting the condition regardless of the state of its corresponding interrupt enable BP_DONE_INT_ENA.
6	SS_TE	R/W1C	0x0	Slave Select Trailing Edge Detect: 0 = no $\overline{\text{SPISS}}$ de-assertion detected since this bit was cleared 1 = the $\overline{\text{SPISS}}$ de-assertion has been detected This bit is cleared by writing a '1' to it. This bit is a sticky status bit, and will set upon meeting the condition regardless of the state of its corresponding interrupt enable SS_TE_INT_ENA.
5	CYC_DONE	R/W1C	0x0	Cycle Done: this bit will set when the current transfer of word-length "WL" bits is complete. It indicates that "WL" bits were sent on the transmit port and "WL" bits were sampled on the receive port. This bit is cleared by writing a '1' to it. This bit is a sticky status bit, and will set upon meeting the condition regardless of the state of its corresponding interrupt enable CYC_DONE_INT_ENA.

SPI_STATUS SPI STATUS REGISTER																															
Address = 0xF030_0028																	Default value = 0x0000_0000														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
4	UCLK_ERR	R/W1C	0x0	Underclock Condition: set if the current transfer received less than word-length “WL” clocks on the SPISCLK line prior to $\overline{\text{SPISS}}$ de-assertion. 0 = No Underclock condition detected since this bit was cleared. 1 = Underclock condition detected This bit is cleared by writing a ‘1’ to it. This bit is a sticky status bit, and will set upon meeting the condition regardless of the state of its corresponding interrupt enable UCLK_ERR_INT_ENA.
3	SS_LE	R/W1C	0x0	Slave Select Leading Edge Detect: 0 = no $\overline{\text{SPISS}}$assertion detected since this bit was cleared 1 = a $\overline{\text{SPISS}}$assertion was detected This bit is cleared by writing a ‘1’ to it. This bit is a sticky status bit, and will set upon meeting the condition regardless of the state of its corresponding interrupt enable SS_LE_INT_ENA.
2	RX_OFL_ERR	R/W1C	0x0	Read Buffer Overflow: indicates that the incoming data buffer did not get off-loaded since the last reception, and was overwritten with another incoming data word. 0 = No Read Overflow since this bit was cleared 1 = Read Overflow detected This bit is cleared by writing a ‘1’ to it. This bit is a sticky status bit, and will set upon meeting the condition regardless of the state of its corresponding interrupt enable RX_OFL_ERR_INT_ENA.
1	Reserved		0x0	
0	SPI_INT_STS	RO	0x0	SPI Interrupt: Logical OR of the raw status bit TX_UFL_ERR, RX_OFL_ERR, UCLK_ERR, BP_DONE, CYC_DONE, SS_TE and SS_LE, qualified with each corresponding *_INT_ENA enable. Note that if the corresponding *_INT_ENA of those status bits is reset to ‘0’, those status bits themselves will still assert upon meeting the condition, but will not contribute to the assertion of SPI_INT_STS. The status bits are true “raw” status bits, and the corresponding *_INT_ENA simply allows them to cause an interrupt.

Table 124 SPI_STATUS Register

SPI_SS_CFG– SPI SLAVE SELECT CONFIGURATION REGISTER

SPI_SS_CFG SPI SLAVE SELECT CONFIGURATION REGISTER																															
Address = 0xF030_0030																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:21	Reserved		0	
23:20	SCLK_WAIT	RW	0x0	SPI Clock Wait (Valid in SPI Master Mode only): determines the wait time between successive data transfers in multiple-transfer mode (SPI_MT_ENA=1). The register value sets the minimum number of SPISCLK clock cycles between back-to-back transfers. Note that, during this wait time, $\overline{\text{SPISS}}$ remains active, but SPISCLK does not toggle.
19:16	SS_WAIT	RW	0x0	Slave Select Wait (Valid in SPI Master Mode only): determines the wait time between successive data transfers in single-transfer mode (SPI_MT_ENA=0). The register value sets the minimum number of SPISCLK clock cycles between the de-assertion of $\overline{\text{SPISS}}$ and the re-assertion of $\overline{\text{SPISS}}$.
15:12	SS_HOLD	RW	0x0	Slave Select Hold (Valid in SPI Master Mode only): The register value sets the minimum number of SPISCLK clock cycles to wait from the last SPISCLK transition to de-assertion of $\overline{\text{SPISS}}$.
11:8	SS_SETUP	RW	0x0	Slave Select Setup (Valid in SPI Master Mode only): The register value sets the minimum number of SPISCLK clock cycles to wait from the assertion of $\overline{\text{SPISS}}$to the first SPISCLK transition.
7:3	Reserved		0x00	
2	SS_POL	RW	0x0	Slave Select active level select (Master or Slave modes): 0 = Active low 1 = Active high
1:0	Reserved		0x0	

Table 125 SPI_SS_CFG Register

Note that setting (or resetting) the SS_POL bit may cause SS_TE, SS_LE and CYC_DONE to be set in the status register; these may need to be cleared out before operations begin. It is recommended to set SPI_ENA=0 whilst configuring the SPI module. SPI_ENA should be set to 1 after the SPI_SS_CFG register (and other registers) have been set to the desired values.

SPI_DAT– SPI DATA REGISTER

SPI_DAT SPI DATA REGISTER																															
Address = 0xF030_0038																Default value = 0x0000_0000_0000_0000															
63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
63:0	SPI_DAT	RW	0	READ - data word received from the RX data buffer. WRITE - data word to be transmitted via the TX data buffer.

Table 126 SPI_DAT Register

SPI_INT_CTRL – SPI INTERRUPT CONTROL REGISTER

SPI_INT_CTRL SPI INTERRUPT CONTROL REGISTER																															
Address = 0xF030_0040																Default value = 0x0000_0034															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:9	Reserved		0	
8	TX_UFL_ERR_INT_ENA	RW	0x0	Controls whether the TX_UFL_ERR bit asserts the SPI Interrupt 0 = Disabled 1 = Enabled
7	BP_DONE_INT_ENA	RW	0x0	Controls whether the BP_DONE bit asserts the SPI Interrupt 0 = Disabled 1 = Enabled
6	SS_TE_INT_ENA	RW	0x0	Controls whether the SS_TE bit asserts the SPI Interrupt 0 = Disabled 1 = Enabled
5	CYC_DONE_INT_ENA	RW	0x1	Controls whether the CYC_DONE bit asserts the SPI Interrupt 0 = Disabled 1 = Enabled
4	UCLK_ERR_INT_ENA	RW	0x1	Controls whether the UCLK_ERR bit asserts the SPI Interrupt 0 = Disabled 1 = Enabled
3	SS_LE_INT_ENA	RW	0x0	Controls whether the SS_LE bit asserts the SPI Interrupt 0 = Disabled 1 = Enabled
2	RX_OFL_ERR_INT_ENA	RW	0x1	Controls whether the RX_OFL_ERR bit asserts the SPI Interrupt 0 = Disabled 1 = Enabled
1:0	Reserved		0x0	

Table 127 SPI_INT_CTRL Register

SPI_DMA_CTRL– SPI DMA CONTROL REGISTER

SPI_DMA_CTRL SPI DMA CONTROL REGISTER																															
Address = 0xF030_0048																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:2	Reserved		0	
1	WR_RQST_ENA	RW	0x0	DMA Handshake Enable for SPI TX 0 = Disabled 1 = Enabled This bit must be set during a DMA transfer to the SPI_DAT register
0	RD_RQST_ENA	RW	0x0	DMA Handshake Enable for SPI RX 0 = Disable 1 = Enabled This bit must be set during a DMA transfer from the SPI_DAT register

Table 128 SPI_DMA_CTRL Register

SPI_BP_CNT – SPI BYTE PACK WORD COUNT REGISTER

SPI_BP_CNT SPI BYTE PACK WORD COUNT REGISTER																															
Address = 0xF030_0050																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	SPI_BP_CNT	RW	0x0000_0000	Master Mode (R/W) - Sets the total number of "WL"-bit words for a byte-packed transfer. The transfer completes (and SPISS is de-asserted) when the raw count (SPI_BP_CNT_RAW) reaches this value. Slave Mode (RO) - Indicates the total number of "WL"-bit words transferred for a byte-packed transfer. The value is captured from the raw count at the end of a byte-packed transfer; the readback value is thus invalid until completion of the transfer.

Table 129 SPI_BP_CNT Register

SPI_BP_CNT_RAW – SPI BYTE PACK RAW WORD COUNT REGISTER

SPI_BP_CNT_RAW SPI BYTE PACK RAW WORD COUNT REGISTER																															
Address = 0xF030_0058																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	SPI_BP_CNT_RAW	RW	0x0000_0000	Master Mode - This register is reset to zero whenever the user loads the SPI_BP_CNT register. Increments for each WL word transferred, until the raw count reaches the value loaded in SPI_BP_CNT. Slave Mode - This register is reset to zero whenever a trailing edge SS_TE (de-assertion of $\overline{\text{SPISS}}$) is detected. Before being reset, the contents of this raw count is captured in the SPI_BP_CNT register.

Table 130 SPI_BP_CNT_RAW Register

DMA CONTROLLER MODULE

BASE ADDRESS 0xF040_0000

DMA FEATURES

- 32 Independent Channels
- Fixed-priority 'fairness arbitration' algorithm for all enabled channels
- DMA requests can be assigned to a high priority or low priority arbitration group; each group is arbitrated separately
- Configurable level or edge detection of DMA request signals per channel
- Software Transfer Trigger per channel
- Automated double buffer configurable to load a new transfer set of Source/Destination/Transfer Length registers upon completion per channel
- Support for 64-bit, 32-bit, 16-bit or 8-bit transfers per channel
- Burst Transfer mode for transfers in Low priority arbitration group
- Programmable transfer length (ie. number of bytes)
- Static or incrementing Source and Destination Address per channel
- Maskable Error, Terminal Count, Watermark, Null Link interrupts per channel
- Programmable handshaking
- DMA chaining capability via Linked List descriptors
- Programmable Endian byte-swapping function
- DMA striding
- Write access to SHA module via dedicated FIFO buffer

DMA CHANNEL CONTROL

The following control attributes are provided for each DMA channel:

- Transfer size – the overall number of bytes to transfer
- Transfer arbitration priority, burst mode/size, chaining mode, watermark threshold, endian swap mode, other per channel modes
- Base Addresses for both Source and Destination, and a buffer set of these registers
- Address mode for both Source and Destination (eg. fixed addressing for accessing FIFOs, incrementing addressing for accessing memory)
- Flow Control - can be hardware (ACK) controlled for peripheral modules, or can be software controlled for memory transfers

Some modules are supported on specific DMA channels only, as noted in Table 131. The required handshake (ACK) configurations for the associated TX/RX functions must also be observed.

Note that the restriction applies to the particular module, not to the DMA channel.

MODULE / PATH	DMA CHANNEL	HANDSHAKE (ACK) REQUIREMENTS	
SPI RX	Channel 4	(No specific requirements)	
SPI TX	Channel 5	(No specific requirements)	
AIF1 RX	Channel 6	Source Data Phase ACK	DMA_SRC_ACK_CTRL=1 DMA_ADP_ACK_CTRL=0
AIF2 RX	Channel 7	Source Data Phase ACK	DMA_SRC_ACK_CTRL=1 DMA_ADP_ACK_CTRL=0
AIF1 TX	Channel 8	Destination Data Phase ACK	DMA_SRC_ACK_CTRL=0 DMA_ADP_ACK_CTRL=0
AIF2 TX	Channel 9	Destination Data Phase ACK	DMA_SRC_ACK_CTRL=0 DMA_ADP_ACK_CTRL=0
AIF3 RX	Channel 10	Source Data Phase ACK	DMA_SRC_ACK_CTRL=1 DMA_ADP_ACK_CTRL=0
AIF3 TX	Channel 11	Destination Data Phase ACK	DMA_SRC_ACK_CTRL=0 DMA_ADP_ACK_CTRL=0
Note: the Handshake (ACK) configuration is selected using the DMA_SRC_ACK_CTRL and DMA_ADP_ACK_CTRL control fields in the DMA Control 1 Register (DMA_CTRL1_n)			

Table 131 DMA Channel Assignments

The Handshake (ACK) configuration is selected using the DMA_SRC_ACK_CTRL and DMA_ADP_ACK_CTRL control fields in the DMA Control 1 Register (DMA_CTRL1_n).

Note that the DMA Handshake must also be enabled in the respective path for the applicable module(s). In the case of the SPI module and AIF modules, refer to the SPI_DMA_CTRL and AIF_INT_CTRL registers respectively,

DMA CHANNEL ARBITRATION

Channel requests may be assigned by configuration to either the high or low priority group. The high priority channels (as programmed by the DMA_CH_PRI_LOW_ENA bits) are arbitrated separately from the low priority channels.

The arbiters receive requests from each DMA channel. Each DMA channel receives an arbitration slot; further requests from this channel reaching the priority encoder are disabled until all current requesting channels have been serviced within that priority group.

Channels within each group have a fixed-priority where lower numbered channels have the higher priority (i.e. channel 0 is highest priority, followed by channel 1, etc.). However, the servicing of all channels within a high or low priority group is ensured by the disabling of granted requests until all requesting channels in the group have had an opportunity to be serviced.

The arbitration result will be selected from the low priority channels only if there are no high priority requests.

Low priority channels for which the burst write portion of a DMA transfer is pre-empted retain the pre-empted status for the next time no high priority channels are selected by the arbiter. Only one channel may be pre-empted at a time.

In the event of conflicting demands for accessing the AHB bus, the priority selection is determined by the DMA_AHB_ARB_SET bit. Priority is given either to the DSP core, or else to the DMA controller. Care should be taken when selecting DMA priority, as this can cause the rest of the system to be locked out until the DMA activity completes. This concern is only applicable for ‘memory to memory’ transfers, where there is no external I/O interface constraining the transfer speed.

NORMAL DMA OPERATION

The DMA copies data between memory addresses and/or peripheral modules. The DMA can support 64-bit, 32-bit, 16-bit, or 8-bit data word sizes; the word size is selected using the DMA_SRC_HSIZE and DMA_DST_HSIZE register fields.

Note that, for DMA transfers to/from the AIF modules, the data word size must be 32 bits (DMA_DST_HSIZE=0x2).

For normal DMA operation, the Primary set of Source/Destination/Transfer Length registers are configured for the applicable DMA channel. When the channel is enabled (DMA_CH_ENA=1), these registers direct the DMAs to proceed until the transfer count (DMA_CNT) equals the Transfer Length (DMA_PRI_LEN); the DMA activity for the channel then stops. The Terminal Count Status bit (and Terminal Count Interrupt, if un-masked) will also be asserted at this time. The channel must then be re-enabled for further DMAs to occur.

By default, the Source & Destination addresses increment after each data transfer; this is selectable using DMA_SRC_NINC and DMA_DST_NINC. The default increment step is equal to the number of bytes selected as the Source & Destination word size (DMA_SRC_HSIZE, DMA_DST_HSIZE). Other increments can be configured using the Stride function.

Hardware handshake (ACK) control must be configured for DMA transfers to/from the SPI or AIF modules. This is configured using the DMA_SRC_ACK_CTRL and DMA_ADP_ACK_CTRL control bits. Note that some modules have specific ACK requirements, as noted in Table 131.

Software transfer control is used (instead of the ACK handshake control) for ‘memory to memory’ transfers. Software transfer control is selected using the DMA_SOFT_XFER_ENA bit.

The DMA_LOCAL_DST_ADDR and DMA_LOCAL_SRC_ADDR bits select a local address (internal to the DMA controller) for the data destination or source. Local addresses are undefined, and implemented as Null (Write) or ‘0’ (Read) values.

SHA MODULE DATA INPUT

Data transferred by the DMA module can be enabled as input to the SHA module by setting the DMA_SHA_XFER_ENA bit for the respective DMA channel.

SHA data input is implemented via a FIFO buffer within the DMA module. The destination address (ie. within the SHA module) is configured automatically.

Note that, when the SHA data transfer is enabled, this is additional to any 'normal' transfer configured using the Destination registers (eg. DMA_PRI_DST_n). To transfer data to the SHA alone, and not to any other destination, the DMA_LOCAL_DST_ADDR bit should be used to select a 'Null' destination address, as described above.

The DMA_FIFO_STATUS register provides an indication of the SHA FIFO buffer status. This can be read at any time, or selected as an input to the Interrupt controller.

Note that the SHA data transfer (via the FIFO) may take longer than the transfer to the DMA_PRI_DST_n destination. In this event, the SHA FIFO buffer status will indicate data in the buffer after the DMA channel has disabled.

DOUBLE-BUFFER DMA OPERATION

Normal DMA operation, using the Primary set of Source/Destination/Transfer Length registers, is described above.

When double-buffered operation is enabled, a Secondary (buffer) set of Source/Destination/Transfer Length registers are available to automatically extend DMA activity without incurring any gap between one transfer and the next. Double-buffered operation is enabled by setting the register bit DMA_DWB_ENA = 1. The Secondary (buffer) register set is loaded into the Primary register set upon the Terminal Count interrupt being set. The Secondary (buffer) register set is only used when Double Buffer operation is enabled.

Normally, the DMA_CH_ENA bit in the DMA_CTRL1 register is set to 0 upon reaching the Terminal Count, and would stay cleared. When Double-buffered Operation is enabled, the DMA_CH_ENA will be set automatically after the primary registers have been updated, allowing DMA processing to continue.

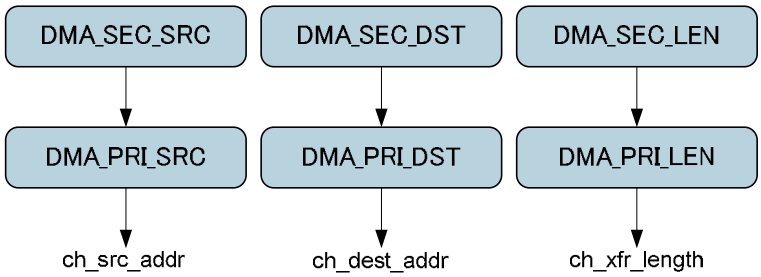


Figure 45 Double Buffer Register Structure

After the Terminal Count interrupt, the DMA_DWB_ENA bit is cleared to 0 by the hardware, indicating that the Secondary Buffer registers are empty. The Secondary Buffer registers can then be re-loaded, and DMA_DWB_ENA set to 1, to configure the next transfer. Note that the Primary register set should not be modified unless DMA_CH_ENA and DMA_DWB_ENA are both cleared.

Double-buffer operation is extended by loading the next buffer register set and re-enabling DMA_DWB_ENA, as described above. If the current transfer completes before the Secondary Buffer registers have been loaded (ie. before the DMA_DWB_ENA bit has been set to 1), then the DMA channel will be disabled, and must be enabled again by setting DMA_CH_ENA=1.

Note that, depending on the length of each DMA transfer operation, there may be only a short time window between the Terminal Count Interrupt (DMA_DWB_ENA=0) and completion of the next transfer. For continuous DMA operation, the secondary buffers must be loaded for the next transfer before the previous one completes.

Further operations are automatically disabled after the DMA_DWB_ENA bit is set to 0 by the hardware. This can be re-enabled by setting DMA_DWB_ENA=1.

Multiple DMA processes can be chained in a defined sequence using the DMA_LINK_ADDR register, as described below.

LINKED LIST DMA CHAINING

The Linked List feature enables multiple DMA processes to be chained in a defined sequence. This feature is enabled using the DMA_LINK_ENA bit.

Linked List DMA chaining is supported for High Priority channels only; Burst Data transfers are not supported with the Linked List feature.

Linked List DMA chaining is supported with 32-bit data word size only (DMA_SRC_HSIZE=0x2, DMA_DST_HSIZE=0x2). DMA chaining makes use of the DMA double buffer mechanism, which must be enabled by setting DMA_DWB_ENA=1.

When a DMA is initiated, and DMA_LINK_ENA is set, the primary register set is invalid except for the DMA_LINK_ADDR register, which indicates where the first descriptor is located in memory. When the channel is enabled (with DMA_LINK_ENA also enabled), the DMA controller fetches the initial descriptor and writes it to the Secondary (buffer) set of Source / Destination / Transfer Length / Next Link Address registers. The newly-loaded buffer register set is then loaded into the primary register set, and the DMA will proceed as directed by the initial parameters. When the DMA reaches its Terminal Count, the subsequent descriptor is fetched and loaded (as directed by the DMA_LINK_ADDR register), and the process continues. If the link address DMA_LINK_ADDR is set to 0x0000_0000, the chaining will terminate.

Note that the fetch of the next descriptor from memory is itself a DMA operation. The source address is the next link address from the primary set, and the destination address is base address of the buffer set of registers within the DMA controller.

The DMA descriptor format is arranged in linked list memory as described below:

DESCRIPTION	ADDRESS
Source Address	= DMA_LINK_ADDR
Destination Address	= DMA_LINK_ADDR + 0x4
Transfer Length	= DMA_LINK_ADDR + 0x8
Next Linked descriptor Source Address	= DMA_LINK_ADDR + 0xC

Table 132 Linked List Memory Addressing

Note that Linked List DMA chaining is supported with either hardware handshake (ACK) or software controlled transfers. The required transfer control (selected by DMA_SOFT_XFR_ENA) depends upon the peripheral type(s) associated with the transfer. The DMA_SOFT_XFR_ENA bit is described in Table 156.

DMA transfers can be configured to generate an interrupt on every Terminal Count. Linked DMA transfers can be configured to assert the interrupt on every terminal count, or on just the final terminal count of the DMA transfer set. This is selected using the DMA_LINK_INT register bit.

Linked DMA transfers may also be configured to interrupt when the next field in the fetched descriptor is 0x0000_0000 (NULL) - indicating that the fetched DMA descriptor is for the last DMA transfer set in the chain. The 'Link Null' status can be read from the DMA_LINKNUL_STS register; the un-masked status bits are used to trigger the DMA_LINKNUL_INT_STS Interrupt Status.

On reaching the end of the Linked List DMA chain, the DMA_DWB_ENA bit is cleared to 0 by the hardware. The Linked List chain can then be disabled. Note that the Linked List function must be disabled (by setting DMA_LINK_ENA=0) before a subsequent Linked List is enabled.

An example Linked List DMA Chain operation is described in the "DMA Program Examples" section.

DMA STRIDING

Under default conditions, the Source & Destination addresses increment after each data transfer; the address increment is equal to the number of bytes selected as the Source & Destination word size (DMA_SRC_HSIZE, DMA_DST_HSIZE). The DMA controller also supports a DMA striding feature whereby the next AHB address may stride forward (increment) by a selectable multiple of the default step size, and can also access a number of interleaved sets of address registers.

Typical applications for DMA striding include sorting different channels of received data into separate buffers, or combining multiple buffers of audio data for interleaved transmission.

Striding is configured using the DMA_STRIDE register. The stride feature can be enabled for source addresses and/or destination addresses using the DMA_STRIDE_SRC_ENA and DAM_STRIDE_DST_ENA fields. Note that this feature is limited to non-burst DMAs only.

Note that, when striding is enabled for source addresses, the DMA_SRC_NINC bit must set be 0. When striding is enabled for destination addresses, the DMA_DST_NINC bit must set be 0.

The magnitude of the stride step is configured via the DMA_STRIDE_LEN field. Setting DMA_STRIDE_LEN = 0x3 sets the stride step as 4 x the DMA_SRC_HSIZE number of bytes. If DMA_SRC_HSIZE = 0x2 (32-bits, 4 bytes), then the stride step size is 4 x 4 = 16 bytes in this case.

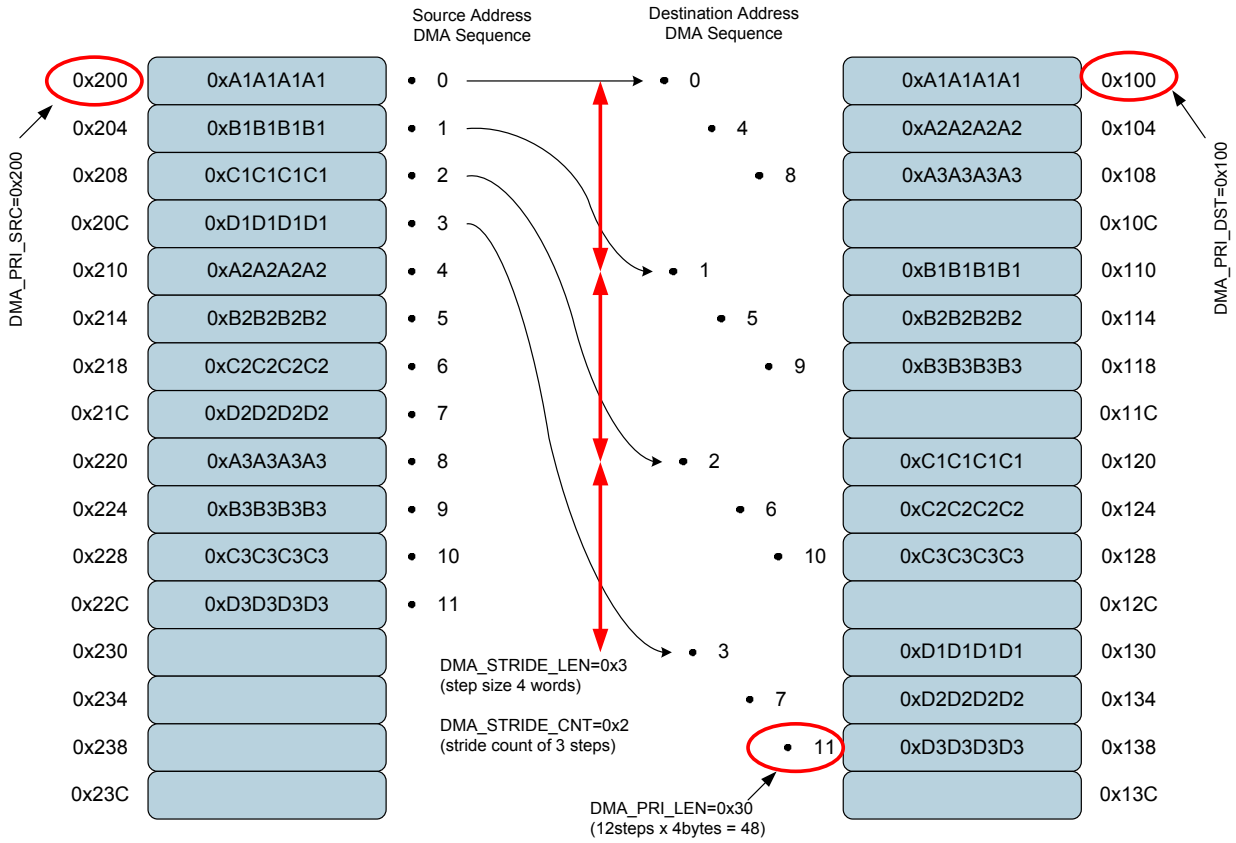
The number (count) of stride steps taken before beginning a next set of stride steps is configured via the DMA_STRIDE_CNT field. Setting DMA_STRIDE_CNT=0x2 selects 3 steps to be taken before the selecting the next set of memory addresses.

The first set of stride steps begins with the configured source or destination AHB address for the channel. Subsequent sets of stride steps begin at the initial address of the previous set of stride steps incremented by the number of bytes indicated by DMA_SRC_HSIZE. The striding process thus selects interleaved values of source and/or destination data addresses. Successive strides will be executed until the total number of bytes transferred reaches the transfer size (DMA_PRI_LEN).

An example DMA striding sequence is described in Figure 46. This illustrates how the DMA stride function could be used to convert an interleaved set of data into separate buffers.

Note that the DMA stride function could also be used to perform the reverse function, ie. combining multiple buffers of data into an interleaved set.

In the example shown, the data word size is 32-bits (DMA_SRC_HSIZE = 0x2). The stride sequence is configured using the DMA_STRIDE_LEN and DMA_STRIDE_CNT registers, as shown.



Register configuration for illustrated DMA transfer:

DMA_PRI_SRC = 0x200	Initial Source address = 0x200
DMA_PRI_DST = 0x100	Initial Destination address = 0x100
DMA_PRI_LEN = 0x30	Transfer length = 48 bytes
DMA_STRIDE_SRC_ENA = 0x0	Striding disabled for Source addresses
DMA_STRIDE_DST_ENA = 0x1	Striding enabled for Destination addresses
DMA_STRIDE_LEN = 0x3	Stride step size of 4 x 32-bit words, ie. 0x10 bytes
DMA_STRIDE_CNT = 0x2	Stride count of 3 steps before next set of strides

Figure 46 DMA Striding Example

BURST DATA TRANSFERS

To improve the efficiency of low-priority DMA operations, these channels are configured as Burst Data transfers. This is a mechanism where multiple data words are transferred in a single 64-bit AHB operation.

Burst Data transfers are enabled by setting DMA_AHB_BURST_ENA=1. The Burst Data transfer must be enabled for Low-Priority DMA channels, and must be disabled for High-Priority DMA channels. Accordingly, DMA_AHB_BURST_ENA and DMA_CH_PRI_LOW_ENA must always be set to the same value.

The maximum burst size/type is configured using the DMA_AHB_MAX_BURST field. It is recommended that the highest setting (10) is selected in all cases.

The auto-increment option must be enabled for Source addresses and Destination addresses (DMA_SRC_NINC=0 and DMA_DST_NINC=0) when Burst Data transfer is enabled. As with other DMA transfers, the Source and Destination addresses must be aligned with the data word size (DMA_SRC_HSIZE, DMA_DST_HSIZE).

The Burst Data transfer mode cannot be used with an I/O or FIFO-type device (ie. cannot be used for DMA transfers to/from the SPI or AIF modules).

Note that, once a DMA Burst Data transfer has been commanded, there is no provision to cancel the transfer.

The Burst Data controller automatically handles instances where the transfer length (LEN) does not align exactly with the 64-bit AHB width. Burst Data transactions are limited to 1kB address boundaries; the burst controller automatically handles any transfers that cross over these limits.

After a Burst Data transfer has been initiated, it is possible that a high-priority DMA channel may be subsequently enabled before the write portion of the Burst Data transfer. In this case (known as 'pre-emption'), the high-priority DMA channel will be serviced, and the Burst Data transfer is deferred.

DMA BYTE SWAP

The DMA_BYTEx_SRC fields within the DMA_CTRL2 register provide a universal byte swap feature. The source byte for each byte within the destination register may be selected independently.

The register defaults are set so that no byte swapping occurs for a DMA transfer for the AHB write data bytes relative to the AHB read data bytes. The byte swapping is illustrated in Figure 47. The byte swap is universal since each byte of the destination AHB write data word may be selected from any byte in the AHB source read data word.

Note that, when Endian Byte Swap is enabled (DMA_ENDIAN_SWAP_ENA=1), then the DMA_BYTEx_SRC fields are ignored.

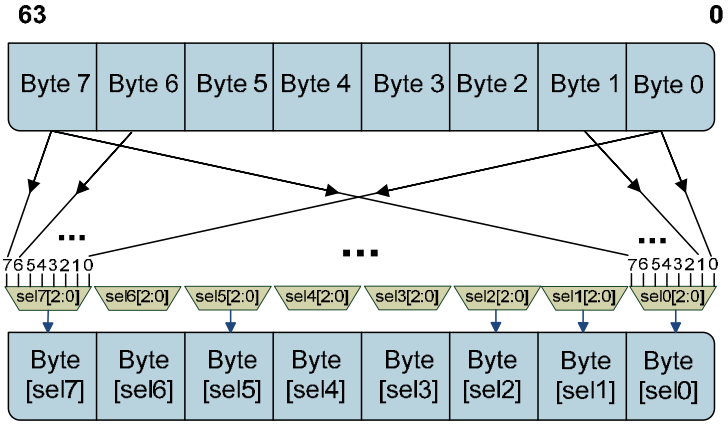


Figure 47 DMA Byte Swap

When 32-bit word size is selected, the DMA_BYTE[7,6,5,4]_SRC registers must be set to the same values as the respective DMA_BYTE[3,2,1,0]_SRC register.

When 16-bit word size is selected, the DMA_BYTE[7,5,3]_SRC registers must all be set to the same value as the DMA_BYTE1_SRC register. The DMA_BYTE[6,4,2]_SRC registers must all be set to the same value as the DMA_BYTE0_SRC register.

If any of the DMA_BYTEn_SRC fields select a byte that is outside the selected data word size (eg. selecting Byte 4 when the word size is 32-bits), then a Modulus function will adjust the selection to a valid setting for the applicable data word size. This ensures that the default register settings will always result in 'no swap', regardless of the data word size.

ENDIAN BYTE SWAP

An Endian Byte Swap function is provided, which is enabled using the DMA_ENDIAN_SWAP_ENA control bit. The Endian Byte Swap is designed to support 64-bit, 32-bit, 24-bit or 16-bit application word sizes, as selected by DMA_ENDIAN_SWAP_LEN.

In the case of packed data words, 2 or more application words may be arranged within the DMA data word, as shown in Figure 48. The swap is typically configured so that the position of each packed word is unchanged by the swap, but the associated bytes are reverse-ordered.

For 24-bit word data, a padding byte (0x00) is included in the word definition; this may be either in the Most Significant or Least Significant Byte position.

Packed data formats are illustrated in Figure 48. Note that byte packing is implemented outside the DMA controller, within the SPI module only.

It is recommended to ensure that the selected swap is compatible with the DMA data word size (and packing configuration, if applicable).

For example, if the DMA data word size is 16-bit, then the 32-bit word Endian Byte Swap (DMA_ENDIAN_SWAP_LEN=3h) should not be selected. The DMA data word size may be larger than the application word size, but cannot be smaller.

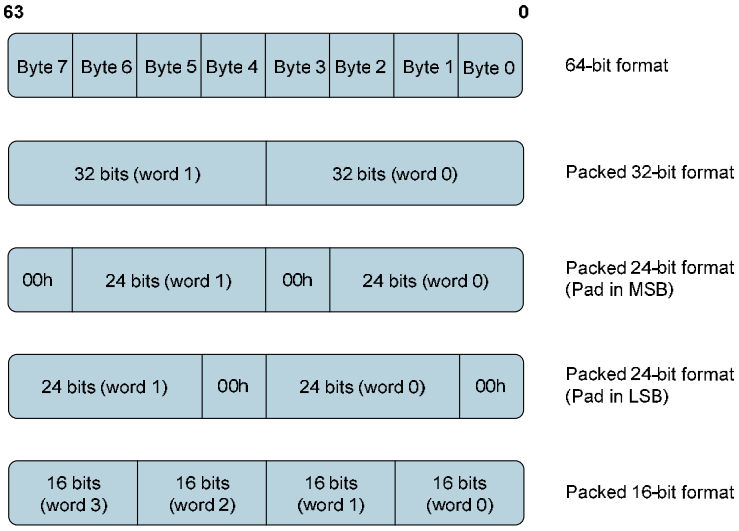


Figure 48 Data Word Packing

Figure 49 illustrates the Endian Byte Swap options, selectable by DMA_ENDIAN_SWAP_LEN. Note that the data within each byte is not affected by this function.

- The 64-bit application word swap is a reverse ordering of the 8 bytes.
- The Packed 32-bit application word swap is a reverse ordering of each 4-byte word.
- Two different swaps are supported for Packed 24-bit Application Word swaps, with the padding bytes either in the Most Significant or Least Significant Byte position. In each case, the padding bytes are unchanged, and the swap is a reverse ordering of each 3-byte word.
- The 16-bit application word swap is a reverse ordering of each 2-byte pair.

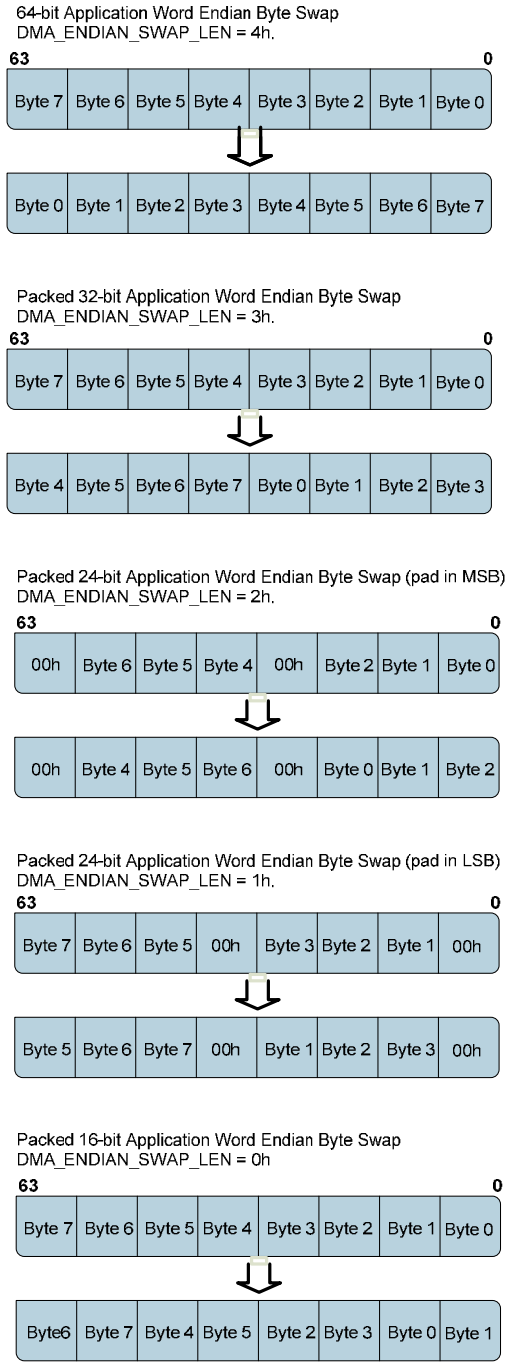


Figure 49 Endian Byte Swap

DMA INTERRUPTS

DMA interrupts may be triggered by any of the following events:

- Terminal Count being reached
- Watermark Threshold being reached or exceeded
- Next Link address for a Linked DMA transfer is NULL
- SHA transfer FIFO status
- Error conditions

Whenever an Error or Terminal Count condition occurs, the corresponding DMA_CH_ENA bit will be reset to 0, disabling further transfers on that channel. Note that the channel will be disabled regardless of whether the Error or Terminal Count interrupts are masked for the channel. Watermark and 'Link Null' conditions will also cause interrupts, but do not disable transfers.

When Double-buffered Operation is enabled, the DMA_CH_ENA will be set automatically after the primary registers have been updated, allowing DMA processing to continue.

Note that a SHA data transfer (via the FIFO) may take longer than the transfer to the DMA_PRI_DST_n destination. In this event, the SHA FIFO buffer status will indicate data in the buffer after the Terminal Count is reached, and after the DMA channel has been disabled.

The status bits relating to each interrupt condition are latched, and are held high once set. The latched values are available to be read via the DMA_TC_STS, DMA_WMARK_STS, DMA_LINKNUL_STS, DMA_FIFO_STATUS and DMA_ERR_STS registers. Individual bits may be cleared by writing a '1' to the respective bit. Each condition may be individually masked from contributing to the DMA interrupt via the associated *_INT_MSK bits.

The DMA interrupt output signal (when enabled using DMA_INT_ENA=1) is the "OR" of all the unmasked interrupt status register bits.

The DMA interrupt control registers are illustrated in Figure 50.

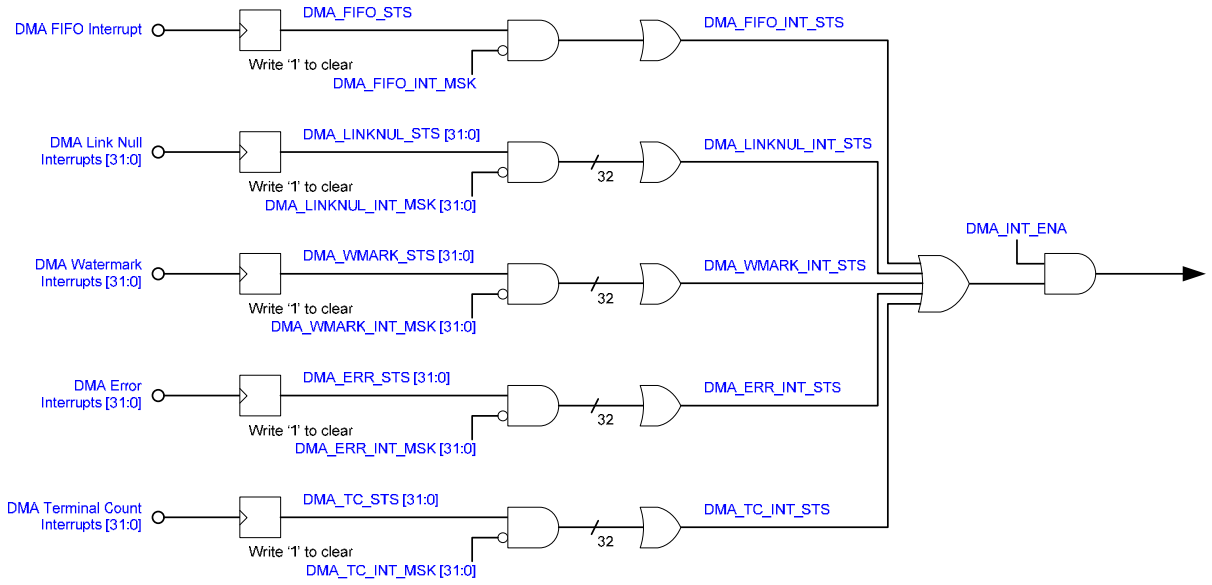


Figure 50 DMA Interrupts

DMA REGISTER MAP

This table illustrates the address map of the DMA module.

ADDRESS	REGISTER	DESCRIPTION	RESET VALUE
Base + 0x00	DMA_GLB_CTRL	DMA Global Control	0x0000_0000
Base + 0x04	DMA_INT_STS	DMA Interrupt Status	0x0000_0000
Base + 0x08	DMA_TC_INT_MSK	DMA Terminal Count Interrupt Mask (Channels 31:0)	0x0000_0000
Base + 0x10	DMA_ERR_INT_MSK	DMA Error Interrupt Mask (Channels 31:0)	0x0000_0000
Base + 0x18	DMA_WMARK_INT_MSK	DMA Watermark Interrupt Mask (Channels 31:0)	0x0000_0000
Base + 0x20	DMA_LINKNUL_INT_MSK	DMA Link Null Interrupt Mask (Channels 31:0)	0x0000_0000
Base + 0x28	DMA_TC_STS	DMA Terminal Count Status (Channels 31:0)	0x0000_0000
Base + 0x30	DMA_ERR_STS	DMA Error Status (Channels 31:0)	0x0000_0000
Base + 0x38	DMA_WMARK_STS	DMA Watermark Status (Channels 31:0)	0x0000_0000
Base + 0x40	DMA_LINKNUL_STS	DMA Link Null Status (Channels 31:0)	0x0000_0000
Base + 0x48	DMA_FIFO_INT_MASK	DMA FIFO Interrupt Mask	0x0000_0000
Base + 0x4C	DMA_FIFO_STATUS	DMA FIFO Status	0x0000_0000
Base + 0x50	DMA_AHB_SLAVE_ADDR	DMA AHB Slave Address	0x0000_0000
Base + n*0x40 + 0x100	DMA_PRI_SRC_n	DMA Primary Source Address (Channel ‘n’)	0x0000_0000
Base + n*0x40 + 0x104	DMA_PRI_DST_n	DMA Primary Destination Address (Channel ‘n’)	0x0000_0000
Base + n*0x40 + 0x108	DMA_PRI_LEN_n	DMA Primary Transfer Length (Channel ‘n’)	0x0000_0000
Base + n*0x40 + 0x10C	DMA_LINK_ADDR_n	DMA Link Address (Channel ‘n’)	0x0000_0000
Base + n*0x40 + 0x110	DMA_SEC_SRC_n	DMA Secondary Source Address (Channel ‘n’)	0x0000_0000
Base + n*0x40 + 0x114	DMA_SEC_DST_n	DMA Secondary Destination Address (Channel ‘n’)	0x0000_0000
Base + n*0x40 + 0x118	DMA_SEC_LEN_n	DMA Secondary Transfer Length (Channel ‘n’)	0x0000_0000
Base + n*0x40 + 0x120	DMA_COUNT_n	DMA Transfer Count (Channel ‘n’)	0x0000_0000
Base + n*0x40 + 0x124	DMA_WMARK_CNT_n	DMA Watermark Count (Channel ‘n’)	0x0000_0000
Base + n*0x40 + 0x128	DMA_CTRL1_n	DMA Control 1 (Channel ‘n’)	0x0000_0000
Base + n*0x40 + 0x12C	DMA_CTRL2_n	DMA Control 2 (Channel ‘n’)	0x0000_0000
Base + n*0x40 + 0x130	DMA_SOFT_ABORT_n	DMA Software Abort (Channel ‘n’)	0x0000_0000
Base + n*0x40 + 0x134	DMA_STRIDE_n	DMA Stride (Channel ‘n’)	0x0000_0000
Note that, in the above descriptions, ‘n’ represents the DMA channel number, ie. 0, 1, 2 … 31.			

Table 133 DMA Register Definition

DMA_GLB_CTRL REGISTER

DMA_GLB_CTRL
DMA GLOBAL CONTROL REGISTER

Address = 0xF040_0000 **Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:2	Reserved		0x0000_0000	
1	DMA_INT_ENA	RW	0x0	DMA Interrupt Enable - selects whether a DMA Interrupt is raised when the DMA_INT_STS register is non-zero. 0 = Disabled 1 = Enabled

DMA_GLB_CTRL DMA GLOBAL CONTROL REGISTER																															
Address = 0xF040_0000																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
0	DMA_ENA	RW	0x0	DMA Module Enable 0 = Disabled 1 = Enabled

Table 134 DMA_GLB_CTRL Register

DMA_INT_STS REGISTER

DMA_INT_STS DMA INTERRUPT STATUS REGISTER																															
Address = 0xF040_0004																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	DMA_FIFO_INT_STS	RO	0x0	This bit is the logical OR of the unmasked bits in the DMA_FIFO_STATUS register. This bit is cleared only when the DMA_FIFO_STATUS bits are cleared or masked; there is no explicit method to clear DMA_FIFO_INT_STS directly.
30:13	Reserved		0x0000	
12	DMA_LINKNUL_INT_STS	RO	0x0	These bits are the logical OR of the unmasked bits in the DMA_LINKNUL_STS register. This bit is cleared only when the DMA_LINKNUL_STS bits are cleared or masked; there is no explicit method to clear DMA_LINKNUL_INT_STS directly.
11:9	Reserved		0x0	
8	DMA_WMARK_INT_STS	RO	0x0	These bits are the logical OR of the unmasked bits in the DMA_WMARK_STS register. This bit is cleared only when the DMA_WMARK_STS bits are cleared or masked; there is no explicit method to clear DMA_WMARK_INT_STS directly.
7:5	Reserved		0x0	
4	DMA_ERR_INT_STS	RO	0x0	These bits are the logical OR of the unmasked bits in the DMA_ERR_STS register. This bit is cleared only when the DMA_ERR_STS bits are cleared or masked; there is no explicit method to clear DMA_ERR_INT_STS directly.
3:1	Reserved		0x0	
0	DMA_TC_INT_STS	RO	0x0	These bits are the logical OR of the unmasked bits in the DMA_TC_STS register. This bit is cleared only when the DMA_TC_STS bits are cleared or masked; there is no explicit method to clear DMA_TC_INT_STS directly.

Table 135 DMA_INT_STS Register

DMA_TC_INT_MSK REGISTER

DMA_TC_INT_MSK
DMA TERMINAL COUNT INTERRUPT MASK REGISTER

Address = 0xF040_0008																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_TC_INT_MSK	RW	0x0000_0000	Terminal Count Interrupt Mask. Each bit masks the Terminal Count status interrupt for the corresponding channel. Bit 31 corresponds to DMA Channel 31 … Bit 0 corresponds to DMA Channel 0 Each bit is coded as: 0 = Enabled; 1 = Masked.

Table 136 DMA_TC_INT_MSK Register

DMA_ERR_INT_MSK REGISTER

DMA_ERR_INT_MSK
DMA ERROR INTERRUPT MASK REGISTER

Address = 0xF040_0010																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_ERR_INT_MSK	RW	0x0000_0000	Error Interrupt Mask. Each bit masks the Error status interrupt for the corresponding channel. Bit 31 corresponds to DMA Channel 31 … Bit 0 corresponds to DMA Channel 0 Each bit is coded as: 0 = Enabled; 1 = Masked.

Table 137 DMA_ERR_INT_MSK Register

DMA_WMARK_INT_MSK REGISTER

DMA_WMARK_INT_MSK
DMA WATERMARK INTERRUPT MASK REGISTER

Address = 0xF040_0018																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_WMARK_INT_MSK	RW	0x0000_0000	Watermark Interrupt Mask. Each bit masks the Watermark status interrupt for the corresponding channel. Bit 31 corresponds to DMA Channel 31 … Bit 0 corresponds to DMA Channel 0 Each bit is coded as: 0 = Enabled; 1 = Masked.

Table 138 DMA_WMARK_INT_MSK Register

DMA_LINKNUL_INT_MSK REGISTER

DMA_LINKNUL_INT_MSK DMA LINK NULL INTERRUPT MASK REGISTER																															
Address = 0xF040_0020																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_LINKNUL_INT_MSK	RW	0x0000_0000	Link Null Interrupt Mask. Each bit masks the Link Null status interrupt for the corresponding channel. Bit 31 corresponds to DMA Channel 31 … Bit 0 corresponds to DMA Channel 0 Each bit is coded as: 0 = Enabled; 1 = Masked.

Table 139 DMA_LINKNUL_INT_MSK Register

DMA_TC_STS REGISTER

DMA_TC_STS DMA TERMINAL COUNT STATUS REGISTER																															
Address = 0xF040_0028																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_TC_STS	R/W1C	0x0000_0000	Terminal Count Status. Each bit is asserted (logic 1) when the Terminal Count has been reached for the corresponding channel. When a DMA_TC_STS bit is set, the corresponding DMA_CH_ENA bit in the DMA_CTRL1 register will be reset. This disables further transfers on this channel. These bits are cleared by writing '1' to the respective bit. Bit 31 corresponds to DMA Channel 31 … Bit 0 corresponds to DMA Channel 0

Table 140 DMA_TC_STS Register

DMA_ERR_STS REGISTER

DMA_ERR_STS DMA ERROR STATUS REGISTER																															
Address = 0xF040_0030																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_ERR_STS	R/W1C	0x0000_0000	Error Status. Each bit is asserted (logic 1) when an Error response is received during a Read or Write transfer on the corresponding channel. When a DMA_ERR_STS bit is set, the corresponding DMA_CH_ENA bit in the DMA_CTRL1 register will be reset. This disables further transfers on this channel. These bits are cleared by writing '1' to the respective bit. Bit 31 corresponds to DMA Channel 31 … Bit 0 corresponds to DMA Channel 0

Table 141 DMA_ERR_STS Register

DMA_WMARK_STS REGISTER

DMA_WMARK_STS DMA WATERMARK STATUS REGISTER																															
Address = 0xF040_0038																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_WMARK_STS	R/W1C	0x0000_0000	Watermark Status. Each bit is asserted (logic 1) when the Watermark Count has been reached for the corresponding channel. These bits are cleared by writing '1' to the respective bit. Bit 31 corresponds to DMA Channel 31 … Bit 0 corresponds to DMA Channel 0

Table 142 DMA_WMARK_STS Register

DMA_LINKNUL_STS REGISTER

DMA_LINKNUL_STS DMA LINK NULL STATUS REGISTER																															
Address = 0xF040_0040																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_LINKNUL_STS	R/W1C	0x0000_0000	Link Null Status. Each bit is asserted (logic 1) if a 'NULL' link was fetched for the corresponding channel. These bits are cleared by writing '1' to the respective bit. Bit 31 corresponds to DMA Channel 31 … Bit 0 corresponds to DMA Channel 0

Table 143 DMA_LINKNUL_STS Register

DMA_FIFO_INT_MASK REGISTER

DMA_FIFO_INT_MASK DMA FIFO INTERRUPT MASK REGISTER																															
Address = 0xF040_0048																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:1	Reserved		0x0000_0000	
0	DMA_FIFO_INT_MSK	RW	0x0	FIFO Status Interrupt Mask. This bit masks the FIFO status interrupt. 0 = Enabled; 1 = Masked.

Table 144 DMA_FIFO_INT_MASK Register

DMA_FIFO_STATUS REGISTER

DMA_FIFO_STATUS DMA FIFO STATUS REGISTER																															
Address = 0xF040_004C																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:1	Reserved		0x0000_0000	
0	DMA_FIFO_STS	R/W1C	0x0	FIFO Status. This bit is asserted (logic 1) if there is data in the SHA transfer FIFO. The bit is cleared by writing '1'.

Table 145 DMA_FIFO_STATUS Register

DMA_AHB_SLAVE_ADDR REGISTER

DMA_AHB_SLAVE_ADDR DMA AHB SLAVE ADDRESS REGISTER																															
Address = 0xF040_0050																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_AHB_SLAVE_ADDR	RW	0x0000_0000	DMA AHB Slave Address. This field must be set to 0xF040_0000 for correct operation of the Linked List DMA function.

Table 146 DMA_AHB_SLAVE_ADDR Register

DMA_PRI_SRC REGISTER

DMA_PRI_SRC_n DMA PRIMARY SOURCE ADDRESS REGISTER																															
Address = 0xF040_0100 + (n * 0x40) (n = DMA Channel, valid from 0 to 31)																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_PRI_SRC_n	RW	0x0000_0000	DMA Source Address (Channel 'n') Each register holds base address for the source of the DMA Transfer for its respective channel. Unaligned Address bits for all SRC, DST, and LINK_ADDR registers are ignored. For example, when DMA_SRC_HSIZE=0x3, bits [2:0] are ignored. Not to be written while Linked List chaining is enabled (DMA_LINK_ENA=1).
Note that 'n' represents the DMA channel number, ie. 0, 1, 2 … 31.				

Table 147 DMA_PRI_SRC_n Register

DMA_PRI_DST REGISTER

DMA_PRI_DST_n DMA PRIMARY DESTINATION ADDRESS REGISTER																															
Address = 0xF040_0104 + (n * 0x40) (n = DMA Channel, valid from 0 to 31)																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_PRI_DST_n	RW	0x0000_0000	DMA Destination Address (Channel 'n') Each register holds base address for the destination of the DMA Transfer for its respective channel. Unaligned Address bits for all SRC, DST, and LINK_ADDR registers are ignored. For example, when DMA_SRC_HSIZE=0x3, bits [2:0] are ignored. Not to be written while Linked List chaining is enabled (DMA_LINK_ENA=1).
Note that 'n' represents the DMA channel number, ie. 0, 1, 2 … 31.				

Table 148 DMA_PRI_DST_n Register

DMA_PRI_LEN REGISTER

DMA_PRI_LEN_n DMA PRIMARY TRANSFER LENGTH																															
Address = 0xF040_0108 + (n * 0x40) (n = DMA Channel, valid from 0 to 31)																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_PRI_LEN_n	RW	0x0000_0000	DMA Transfer Length (Channel 'n') Contains the required number of transfer bytes for the channel. The value in this register is not affected by the transfer occurring. The number of bytes must be aligned to the programmed DMA_SRC_HSIZE ("unaligned" bits are ignored). Not to be written while Linked List chaining is enabled (DMA_LINK_ENA=1).
Note that 'n' represents the DMA channel number, ie. 0, 1, 2 … 31.				

Table 149 DMA_PRI_LEN_n Register

DMA_LINK_ADDR REGISTER

DMA_LINK_ADDR_n DMA LINK ADDRESS REGISTER																															
Address = 0xF040_010C + (n * 0x40) (n = DMA Channel, valid from 0 to 31)																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_LINK_ADDR_n	RW	0x0000_0000	DMA Link Address (Channel 'n') This is the AHB Address that contains the next (linked) DMA Source Address register value. The next (linked) DST, LEN and LINK_ADDR registers are held at the adjacent addresses DMA_LINK_ADDR_n + 0x4, DMA_LINK_ADDR_n + 0x8, and DMA_LINK_ADDR_n + 0xC respectively, defining the next (linked) DMA Transfer for its respective channel. Setting this register to 0x0000_0000 indicates that there is no linked DMA Transfer - this is used to terminate the chaining. Note that bits [1:0] are unimplemented and are reserved for future use (DMA_SRC_HSIZE is fixed at 32-bits for Linked List descriptor fetches). Unaligned Address bits for all SRC, DST, and LINK_ADDR registers are ignored.
Note that 'n' represents the DMA channel number, ie. 0, 1, 2 … 31.				

Table 150 DMA_LINK_ADDR_n Register

DMA_SEC_SRC REGISTER

DMA_SEC_SRC_n DMA SECONDARY SOURCE ADDRESS REGISTER																															
Address = 0xF040_0110 + (n * 0x40) (n = DMA Channel, valid from 0 to 31)																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_SEC_SRC_n	RW	0x0000_0000	DMA Buffer Source Address (Channel 'n') When Double Buffer Control is enabled (DMA_DWB_ENA=1), then the contents of this register will be placed into the Source Address register (DMA_PRI_SRC_n) upon reaching terminal count.
Note that 'n' represents the DMA channel number, ie. 0, 1, 2 … 31.				

Table 151 DMA_SEC_SRC_n Register

DMA_SEC_DST REGISTER

DMA_SEC_DST_n DMA SECONDARY DESTINATION ADDRESS REGISTER																															
Address = 0xF040_0114 + (n * 0x40) **(n = DMA Channel, valid from 0 to 31)**																**Default value = 0x0000_0000**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_SEC_DST_n	RW	0x0000_0000	DMA Buffer Destination Address (Channel 'n') When Double Buffer Control is enabled (DMA_DWB_ENA=1), then the contents of this register will be placed into the Destination Address register (DMA_PRI_DST_n) upon reaching terminal count.
Note that 'n' represents the DMA channel number, ie. 0, 1, 2 … 31.				

Table 152 DMA_SEC_DST_n Register

DMA_SEC_LEN REGISTER

DMA_SEC_LEN_n DMA SECONDARY TRANSFER LENGTH REGISTER																															
Address = 0xF040_0118 + (n * 0x40) **(n = DMA Channel, valid from 0 to 31)**																**Default value = 0x0000_0000**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_SEC_LEN_n	RW	0x0000_0000	DMA Buffer Transfer Length (Channel 'n') When Double Buffer Control is enabled (DMA_DWB_ENA=1), then the contents of this register will be placed into the Transfer Length register (DMA_PRI_LEN_n) upon reaching terminal count.
Note that 'n' represents the DMA channel number, ie. 0, 1, 2 … 31.				

Table 153 DMA_SEC_LEN_n Register

DMA_COUNT REGISTER

DMA_COUNT_n DMA TRANSFER COUNT REGISTER																															
Address = 0xF040_0120 + (n * 0x40) (n = DMA Channel, valid from 0 to 31)																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	DMA_TYPE_n	R/WC	0	DMA Transfer Type (Channel 'n') This bit indicates the type of the DMA transfer 0 = Normal DMA 1 = Fetch (Link) DMA
31:0	DMA_CNT_n	R/WC	0x0000_0000	DMA Transfer Count (Channel 'n') Contains the current transfer count (in bytes). This is a read-only register, and reflects the current transfer counter value. The register is reset to 0 upon reaching terminal count or upon writing any value to the register. The register must be cleared manually if an error occurs.
Note that 'n' represents the DMA channel number, ie. 0, 1, 2 … 31.				

Table 154 DMA_COUNT_n Register

DMA_WMARK_CNT REGISTER

DMA_WMARK_CNT_n DMA WATERMARK COUNT																															
Address = 0xF040_0124 + (n * 0x40) (n = DMA Channel, valid from 0 to 31)																Default value = 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	DMA_WMARK_CNT_n	RW	0x0000_0000	DMA Transfer Watermark (Channel 'n') When the DMA Transfer Count (DMA_CNT_n) equals or exceeds the Watermark Count (DMA_WMARK_CNT_n), a watermark interrupt will be generated for this channel, provided that the interrupt is unmasked and enabled.
Note that 'n' represents the DMA channel number, ie. 0, 1, 2 … 31.				

Table 155 DMA_WMARK_CNT_n Register

DMA_CTRL1 REGISTER

DMA_CTRL1_n DMA CONTROL 1 REGISTER																															
Address = 0xF040_0128 + (n * 0x40) (n = DMA Channel, valid from 0 to 31)																													Default value = 0x0000_0000		
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:28	Reserved		0x0	
27:26	DMA_DST_HSIZE	RO	0x0	DMA Destination data word size. The DMA uses the programmed DMA_DST_HSIZE for all AHB write transfers for this channel. 00 = 8 bits 01 = 16 bits 10 = 32 bits 11 = 64 bits Must be the same as DMA_SRC_HSIZE. For DMA transfers to/from the AIF modules, the data word size must be 32 bits (DMA_DST_HSIZE=10). When Linked List DMA chaining is enabled (DMA_LINK_ENA=1), then the data word size must be 32 bits (DMA_DST_HSIZE=10).
25	Reserved		0x0	
24:23	DMA_SRC_HSIZE	RW	0x0	DMA Source data word size. The DMA uses the programmed DMA_SRC_HSIZE for all AHB read transfers for this channel. 00 = 8 bits 01 = 16 bits 10 = 32 bits 11 = 64 bits Must be the same as DMA_DST_HSIZE. For DMA transfers to/from the AIF modules, the data word size must be 32 bits (DMA_DST_HSIZE=10). When Linked List DMA chaining is enabled (DMA_LINK_ENA=1), then the data word size must be 32 bits (DMA_SRC_HSIZE=10).
22	DMA_LOCAL_DST_ADDR	RW	0x0	Indicates the destination address is local (internal to the DMA Controller) when set to a 1. Write half of the DMA transfer completes without AHB cycles. Un-decoded destinations result in a DMA write to "null". In the current implementation all local addresses are undefined and un-decoded. The definition of local addresses is reserved for future implementations.
21	DMA_LOCAL_SRC_ADDR	RW	0x0	Indicates the source address is local (internal to the DMA Controller) when set to a 1. Read half of the DMA transfer completes without AHB cycles. Un-decoded sources result in a DMA null read of all 0s. In the current implementation, all local addresses are undefined and un-decoded. The definition of local addresses is reserved for future implementations.
20	DMA_LINK_INT	RW	0x0	For linked DMA transfers (DMA_LINK_ENA=1), this bit controls whether the Terminal Count Status (DMA_TC_STS, bit [n]) is set every time the Terminal Count is reached, or is set only at the Terminal Count of the final DMA transfer in a chain (final DMA transfer in a chain for which DMA_LINK_ADDR_n=0x0000_0000, ie. 'NULL'). 0 = Set DMA_TC_STS bit [n] at the Terminal Count for each transfer 1 = Set DMA_TC_STS bit [n] at the Terminal Count of the last transfer only

DMA_CTRL1_n
DMA CONTROL 1 REGISTER

Address = 0xF040_0128 + (n * 0x40)
(n = DMA Channel, valid from 0 to 31)

Default value = 0x0000_0000

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
19	DMA_LINK_ENA	RW	0x0	Enable Linked List DMA chaining. 0 = Disabled 1 = Enabled When enabled, the DMA_LINK_ADDR register is used to direct the DMA to the SRC, DST, LEN and LINK_ADDR registers for each linked transfer. The linked list terminates when it reaches a LINK_ADDR equal to 0x0000_0000. Note that, when Linked List DMA chaining is enabled, the data word size must be 32 bits, and Double-Buffering must be enabled (DMA_DWB_ENA=1).
18:17	DMA_AHB_MAX_BURST	RW	0x0	Maximum burst size/type used by the DMA master controller when Burst Data transfer is enabled (DMA_AHB_BURST_ENA=1). 00 = SINGLE 01 = INCR4 10 = INCR8 11 = Reserved The highest setting (10) is recommended in all cases.
16	DMA_AHB_BURST_ENA	RW	0x0	Enable AHB burst transfers. 0 = Disabled 1 = Enabled Burst Data transfers must be enabled for Low-Priority DMA channels. Burst Data transfers must be disabled for High-Priority DMA channels. Accordingly, DMA_AHB_BURST_ENA and DMA_CH_PRI_LOW_ENA must always be set to the same value.
15	DMA_SHA_XFER_ENA	RW	0x0	Enable SHA data transfer. 0 = Disabled 1 = Enabled Note that the SHA data transfer is via a dedicated FIFO. The SHA data transfer is enabled in addition to any 'normal' transfer to the DMA_PRI_DST_n address.
14	Reserved		0x0	
13:11	DMA_ENDIAN_SWAP_LEN	RW	0x0	Endian Byte Swap select. 000 = 16-bit word size 001 = 24-bit word size (pad LS Byte) 010 = 24-bit word size (pad MS Byte) 011 = 32-bit word size 100 = 64-bit word size 101 to 111 = Reserved for future implementations Only valid when DMA_ENDIAN_SWAP_ENA=1
10	Reserved		0x0	
9	DMA_ENDIAN_SWAP_ENA	RW	0x0	Endian Byte Swap enable 0 = Disabled 1 = Enabled
8	DMA_AHB_ARB_SET	RW	0x0	AHB Master priority select Controls which module has priority, in the event of conflicting demands for accessing the AHB bus. 0 = DMA controller has higher priority 1 = DSP Core has higher priority

DMA_CTRL1_n
DMA CONTROL 1 REGISTER

Address = 0xF040_0128 + (n * 0x40)
(n = DMA Channel, valid from 0 to 31)

Default value = 0x0000_0000

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
7	DMA_CH_PRI_LOW_ENA	RW	0x0	Channel Priority group selection. 0= High priority 1= Low priority Burst Data transfers must be enabled for Low-Priority DMA channels. Burst Data transfers must be disabled for High-Priority DMA channels. Accordingly, DMA_AHB_BURST_ENA and DMA_CH_PRI_LOW_ENA must always be set to the same value.
6	DMA_SRC_ACK_CTRL	RW	0x0	ACK control 0 = ACK asserted during Destination Address or Data phase 1 = ACK asserted during Source Address or Data phase In each case, the applicable Address or Data phase is selected by the DMA_ADP_ACK_CTRL bit. When Software Transfer control is enabled (DMA_SOFT_XFER_ENA=1), then this ACK control field is ignored.
5	DMA_ADP_ACK_CTRL	RW	0x0	ACK control 0 = ACK is associated with Data phase (Source or Destination) 1 = ACK is associated with Address phrase (Source or Destination) The DMA_SRC_ACK_CTRL bit determines whether the Source phase or Destination phase is applicable. When Software Transfer control is enabled (DMA_SOFT_XFER_ENA=1), then this ACK control field is ignored.
4	DMA_DWB_ENA	RW	0x0	Double Buffer Control 0 = Disabled 1 = Enabled When Double-Buffering is enabled, the Secondary SRC, DST, LEN registers are used to define the next DMA transfer. These registers are copied into the Primary SRC, DST, LEN registers upon reaching Terminal Count, and the corresponding DMA_CH_ENA bit is set to 1, allowing a new transfer to begin. The DMA_DWB_ENA bit resets to 0 when the Secondary Buffers are empty. The Secondary Buffers must be re-loaded (and DMA_DWB_ENA set to 1) to configure the next transfer. If the current transfer completes before DMA_DWB_ENA has been enabled, then the DMA channel will be disabled and must be enabled again by setting DMA_CH_ENA=1. When Linked List DMA chaining is enabled (DMA_LINK_ENA=1), then Double-Buffering must be enabled (DMA_DWB_ENA=1).
3	DMA_DST_NINC	RW	0x0	Destination Address Increment Control 0 = Destination Address is incremented for each data transfer 1 = Destination Address is not incremented When striding is enabled for destination addresses (DMA_STRIDE_DST_ENA=1), then DMA_DST_NINC must set be 0.
2	DMA_SRC_NINC	RW	0x0	Source Address Increment Control 0 = Source Address is incremented for each data transfer 1 = Source Address is not incremented When striding is enabled for source addresses (DMA_STRIDE_SRC_ENA=1), then DMA_SRC_NINC must set be 0.

DMA_CTRL1_n
DMA CONTROL 1 REGISTER

Address = 0xF040_0128 + (n * 0x40)
(n = DMA Channel, valid from 0 to 31)

Default value = 0x0000_0000

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
1	DMA_SOFT_XFR_ENA	RW	0x0	Selects Software Transfer control 0 = Hardware handshake (ACK) control Hardware handshake control must be selected for DMA transfers to/from SPI or AIF modules. 1 = Software transfer control. (ACK control settings are ignored.) Software transfer must be selected for 'memory-to-memory' transfers.
0	DMA_CH_ENA	RW	0x0	DMA Channel Enable 0 = Disabled 1 = Enabled This bit will automatically reset to 0 upon the channel reaching Terminal Count, or under Error conditions. If Double-Buffering is enabled (DMA_DWB_ENA=1), the channel will then automatically re-enable for the next transfer.
Note that 'n' represents the DMA channel number, ie. 0, 1, 2 … 31.				

Table 156 DMA_CTRL1_n Register

DMA_CTRL2 REGISTER

DMA_CTRL2_n
DMA CONTROL 2 REGISTER

Address = 0xF040_012C + (n * 0x40)
(n = DMA Channel, valid from 0 to 31)

Default value = 0xFAC6_8800

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	DMA_BYTE7_SRC	RW	0x7	Source byte selection for destination byte 7 (bits [63:56])
28:26	DMA_BYTE6_SRC	RW	0x6	Source byte selection for destination byte 6 (bits [55:48])
25:23	DMA_BYTE5_SRC	RW	0x5	Source byte selection for destination byte 5 (bits [47:40])
22:20	DMA_BYTE4_SRC	RW	0x4	Source byte selection for destination byte 4 (bits [39:32])
19:17	DMA_BYTE3_SRC	RW	0x3	Source byte selection for destination byte 3 (bits [31:24])
16:14	DMA_BYTE2_SRC	RW	0x2	Source byte selection for destination byte 2 (bits [23:16])
13:11	DMA_BYTE1_SRC	RW	0x1	Source byte selection for destination byte 1 (bits [15:8])
10:8	DMA_BYTE0_SRC	RW	0x0	Source byte selection for destination byte 0 (bits [7:0])
7:0	Reserved		0x00	Reserved - Do Not Change from 0x00
Note that 'n' represents the DMA channel number, ie. 0, 1, 2 … 31. When Endian Byte Swap is enabled (DMA_ENDIAN_SWAP_ENA=1), then the DMA_BYTEn_SRC registers are ignored.				

Table 157 DMA_CTRL2_n Register

DMA_SOFT_ABORT REGISTER

DMA_SOFT_ABORT_n
DMA SOFTWARE ABORT

Address = 0xF040_0130 + (n * 0x40)
(n = DMA Channel, valid from 0 to 31)

Default value = 0x0000_0000

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:1	Reserved		0x0000_0000	
0	DMA_SW_ABORT_n	RW	0x0	Software abort. Write any value to this register to initiate the abort. The DMA_SW_ABORT_n bit will read back '1' whilst the abort is executed. The bit is cleared to '0' after the abort is complete.

Note that 'n' represents the DMA channel number, ie. 0, 1, 2 … 31.

Table 158 DMA_SOFT_ABORT_n Register

DMA_STRIDE REGISTER

DMA_STRIDE_n
DMA STRIDE REGISTER

Address = 0xF040_0134 + (n * 0x40)
(n = DMA Channel, valid from 0 to 31)

Default value = 0x0000_0000

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	DMA_STRIDE_SRC_ENA	RW	0x0	DMA Stride Control (Source Addresses) 0 = Striding of Source Addresses disabled 1 = Striding of Source Addresses enabled Note that the Burst transfer mode must be disabled (DMA_AHB_BURST_ENA=0) when using the DMA Stride function.
30	DMA_STRIDE_DST_ENA	RW	0x0	DMA Stride Control (Destination Addresses) 0 = Striding of Destination Addresses disabled 1 = Striding of Destination Addresses enabled Note that the Burst transfer mode must be disabled (DMA_AHB_BURST_ENA=0) when using the DMA Stride function.
29:22	Reserved		0x00	
21:16	DMA_STRIDE_CNT	RW	0x00	DMA Stride Count Selects the number of strides to take (AHB transfers + 1) prior to beginning a new set of strides at a base address that is incremented by the # of bytes indicated by DMA_SRC_HSIZE for each set of strides. The base address begins with the configured DMA_PRI_SRC / DMA_PRI_DST address and is incremented each set of strides. A stride is an increment of the AHB address by the amount indicated by DMA_PRI_LEN. 0 = Take 1 stride prior to beginning a new set 1 = Take 2 strides prior to beginning a new set … 63 = Take 64 strides prior to beginning a new set
15:12	Reserved		0x0	

DMA_STRIDE_n DMA STRIDE REGISTER																															
Address = 0xF040_0134 + (n * 0x40) **(n = DMA Channel, valid from 0 to 31)**																**Default value = 0x0000_0000**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
11:0	DMA_STRIDE_LEN	RW	0x000	DMA Stride Length Selects the number of bytes by which to increment the AHB source/destination address(es) when striding is enabled. 0 = 1 times the # of bytes indicated by DMA_SRC_HSIZE 1 = 2 times the # of bytes indicated by DMA_SRC_HSIZE … 4095 = 4096 times the # of bytes indicated by DMA_SRC_HSIZE
Note that ‘n’ represents the DMA channel number, ie. 0, 1, 2 … 31.				

Table 159 DMA_STRIDE_n Register

DMA PROGRAM EXAMPLES

EXAMPLE 1: PERIPHERAL TRANSFERS USING DMA

This example describes a mechanism to transfer 32-bit data words from an I/O peripheral into memory. It is assumed that 64 words (256 bytes) are to be transferred from the AIF1 module into memory.

The data words will be read from a fixed address in the AIF1 module; the AIF_RX_DAT register address for AIF1 is 0xF070_0000. The data will be written to a 256-word block of System RAM memory, starting at 0x6000_0000.

The AIF1 RX path requires the use of DMA channel 6. The handshake configuration must be Source Data Phase ACK. (These requirements are described in Table 131.)

The required register settings for this transfer are noted below. Note that the default setting is assumed for any register fields that are not quoted here.

REGISTER / FIELD NAME	VALUE	DESCRIPTION
DMA_GLB_CTRL		
DMA_ENA	0x1	Enables the DMA module
DMA_PRI_SRC_6		
DMA_PRI_SRC_6	0xF070_0000	Selects AIF_RX_DAT as the source
DMA_PRI_DST_6		
DMA_PRI_DST_6	0x6000_0000	Selects 0x6000_0000 as the destination
DMA_PRI_LEN_6		
DMA_PRI_LEN_6	0x100	Selects a transfer length of 256 bytes
DMA_CTRL1_6		
DMA_DST_HSIZE	0x2	Selects 32-bit word size
DMA_SRC_HSIZE	0x2	Selects 32-bit word size
DMA_SRC_ACK_CTRL	0x1	Selects Source Address ACK
DMA_ADP_ACK_CTRL	0x0	Selects Data Phase ACK
DMA_DST_NINC	0x0	Selects Incrementing Destination addresses
DMA_SRC_NINC	0x1	Selects Non-Incrementing Source addresses
DMA_SOFT_XFER_ENA	0x0	Selects Hardware handshake (ACK) control
DMA_CH_ENA	0x1	Enables the DMA Channel
Note that the default setting is assumed for any DMA Controller register fields that are not quoted.		

Table 160 DMA Example 1

The register settings described in Table 160 will initiate a 64-word (256 byte) transfer from the AIF1 module into a block of memory.

The DMA channel is automatically disabled (DMA_CH_ENA=0) on completion of the transfer.

On completion, the DMA controller will also assert bit [6] in the DMA Terminal Count Status (DMA_TC_STS) register, indicating that the Terminal Count for DMA channel 6 has been reached. When unmasked and enabled, this bit can be used to signal an Interrupt Event to the CCM module.

Note that the Terminal Count status bit should be reset (by writing '1' to the respective bit) in order to allow subsequent DMA transfers to be signalled.

EXAMPLE 2: MEMORY TO MEMORY TRANSFERS USING DMA

This example describes a mechanism to transfer 64-bit data words from one memory block to another. It is assumed that 512 words (4096 bytes) are to be transferred.

The data words will be read from a base address of 0x6000_0000, and will be written to a base address of 0x6000_4000.

Software transfer control will be used, as is required for 'memory-to-memory' transfers. In this example, DMA channel 12 will be used. (Note that SRAM-SRAM transfers can be supported on all DMA channels.)

The required register settings for this transfer are noted below. Note that the default setting is assumed for any register fields that are not quoted here.

REGISTER / FIELD NAME	VALUE	DESCRIPTION
DMA_GLB_CTRL		
DMA_ENA	0x1	Enables the DMA module
DMA_PRI_SRC_12		
DMA_PRI_SRC_12	0x6000_0000	Selects 0x6000_0000 as the source
DMA_PRI_DST_12		
DMA_PRI_DST_12	0x6000_4000	Selects 0x6000_4000 as the destination
DMA_PRI_LEN_12		
DMA_PRI_LEN_12	0x1000	Selects a transfer length of 4096 bytes
DMA_CTRL1_12		
DMA_DST_HSIZE	0x3	Selects 64-bit word size
DMA_SRC_HSIZE	0x3	Selects 64-bit word size
DMA_DST_NINC	0x0	Selects Incrementing Destination addresses
DMA_SRC_NINC	0x0	Selects Incrementing Source addresses
DMA_SOFT_XFER_ENA	0x1	Selects Software transfer control
DMA_CH_ENA	0x1	Enables the DMA Channel
Note that the default setting is assumed for any DMA Controller register fields that are not quoted.		

Table 161 DMA Example 2

The register settings described in Table 161 will initiate a 256-word (1024 byte) transfer from base address 0x6000_0000 to base address 0x6000_4000.

The DMA channel is automatically disabled (DMA_CH_ENA=0) on completion of the transfer.

On completion, the DMA controller will also assert bit [12] in the DMA Terminal Count Status (DMA_TC_STS) register, indicating that the Terminal Count for DMA channel 12 has been reached. When unmasked and enabled, this bit can be used to signal an Interrupt Event to the CCM module.

Note that the Terminal Count status bit should be reset (by writing '1' to the respective bit) in order to allow subsequent DMA transfers to be signalled.

EXAMPLE 3: LINKED LIST DMA OPERATION

This example describes a mechanism to transfer 3 packets of 32-bit data words from memory to the AIF2 module. The packets are defined in a list of descriptor registers, with the first packet descriptor at memory address 0x6007_0000.

The first packet comprises 256 words (1024 bytes) read from base address 0x6000_0000. The DMA descriptors for this part of the transfer are located at address 0x6007_0000.

The second packet comprises 256 words (1024 bytes) read from base address 0x6001_0000. The DMA descriptors for this part of the transfer are located at address 0x6007_0010.

The third packet comprises 512 words (2048 bytes) read from base address 0x6002_0000. The DMA descriptors for this part of the transfer are located at address 0x6007_0020.

The DMA descriptors for each of the packet transfers are contained in the memory configuration described in Table 162.

ADDRESS	VALUE	DESCRIPTION
Packet 1 definition		
0x6007_0000	0x6000_0000	Selects 0x6000_0000 as the source
0x6007_0004	0xF080_0020	Selects AIF_TX_DAT as the destination
0x6007_0008	0x400	Selects a transfer length of 256 bytes
0x6007_000C	0x6007_0010	Identifies the next packet descriptors address
Packet 2 definition		
0x6007_0010	0x6001_0000	Selects 0x6001_0000 as the source
0x6007_0014	0xF080_0020	Selects AIF_TX_DAT as the destination
0x6007_0018	0x400	Selects a transfer length of 256 bytes
0x6007_001C	0x6007_0020	Identifies the next packet descriptors address
Packet 3 definition		
0x6007_0020	0x6002_0000	Selects 0x6002_0000 as the source
0x6007_0024	0xF080_0020	Selects AIF_TX_DAT as the destination
0x6007_0028	0x1000	Selects a transfer length of 512 bytes
0x6007_002C	0x0000_0000	Terminates the Linked List chain

Table 162 DMA Example 3 - Linked List Memory configuration

The data words will be written to a fixed address in the AIF2 module; the AIF_TX_DAT register address for AIF2 is 0xF080_0020.

The AIF2 TX path requires the use of DMA channel 9. The handshake configuration must be Destination Data Phase ACK. (These requirements are described in Table 131.)

The required register settings for this transfer are noted in Table 163. Note that the default setting is assumed for any register fields that are not quoted here.

REGISTER / FIELD NAME	VALUE	DESCRIPTION
DMA_GLB_CTRL		
DMA_ENA	0x1	Enables the DMA module
DMA_AHB_SLAVE_ADDR		
DMA_AHB_SLAVE_ADDR	0xF040_0000	Defines the AHB Slave Address of the DMA module
DMA_LINK_ADDR_9		
DMA_LINK_ADDR_9	0x6007_0000	Defines the address of the DMA descriptors for the first transfer packet
DMA_CTRL1_9		
DMA_DST_HSIZE	0x2	Selects 32-bit word size
DMA_SRC_HSIZE	0x2	Selects 32-bit word size
DMA_LINK_INT	0x1	Configures the Terminal Count Interrupt to assert on completion of the final packet transfer.
DMA_LINK_ENA	0x1	Enables Linked List DMA function
DMA_SRC_ACK_CTRL	0x0	Selects Destination Address ACK
DMA_ADP_ACK_CTRL	0x0	Selects Data Phase ACK
DMA_DWB_ENA	0x1	Enabled Double-Buffer operation
DMA_DST_NINC	0x1	Selects Non-Incrementing Destination addresses
DMA_SRC_NINC	0x0	Selects Incrementing Source addresses
DMA_SOFT_XFER_ENA	0x0	Selects Hardware handshake (ACK) control
DMA_CH_ENA	0x1	Enables the DMA Channel
Note that the default setting is assumed for any DMA Controller register fields that are not quoted.		

Table 163 DMA Example 3 - DMA Register settings

The memory configuration described in Table 162, and the register settings described in Table 163 will initiate a sequence of 3 transfers from memory to the AIF TX port.

The DMA channel is automatically disabled (DMA_CH_ENA=0) on completion of the transfer.

On completion, the DMA controller will also assert bit [9] in the DMA Terminal Count Status (DMA_TC_STS) register, indicating that the Terminal Count for DMA channel 9 has been reached. When unmasked and enabled, this bit can be used to signal an Interrupt Event to the CCM module.

The Terminal Count Interrupt status is configurable for Linked List chains - it can be used to indicate completion of each packet, or else completion of the final packet only. In the example settings above, the DMA_LINK_INT bit configures the DMA channel to indicate only the final packet transfer.

Note that the Terminal Count status bit should be reset (by writing '1' to the respective bit) in order to allow subsequent DMA transfers to be signalled.

AIF INTERFACE MODULES

AIF1 - BASE ADDRESS 0xF070_0000

AIF2 - BASE ADDRESS 0xF080_0000

AIF3 - BASE ADDRESS 0xF090_0000

AIF FEATURES

The AIF Interface modules provide the following features:

- Runtime configurable multi-channel TDM format
- Runtime configurable serial audio format: I2S, Left-Justified or Right-Justified
- Supports all commonly used sample rates (8kHz to 192 kHz)
- Supports any audio sample sizes to 32 bits
- Reports status number of samples in FIFO
- Runtime configurable FIFO thresholds: an interrupt is asserted when the number of samples in the FIFO is greater and or lower than the applicable limit
- Reports loss of channel order (FIFO error conditions)
- Supports slave or master modes
- Supports up to 64 TDM audio channels

An overview of the AIF module is illustrated in Figure 51.

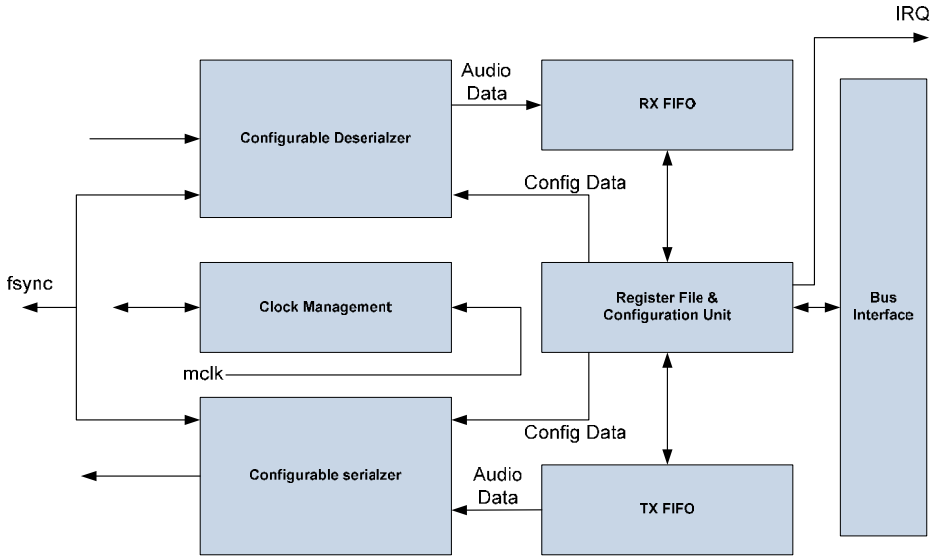


Figure 51 AIF Block Diagram

The RX path de-serializer can be configured to convert the incoming serial audio stream to a parallel interface. If the FIFO is full, the newly arrived samples are dropped until there is space in the FIFO. The RX path de-serializer should be reset before a stable serial audio signal is present at the input.

The TX path serializer reads the audio samples from the FIFO and converts the parallel audio stream interface into the desired output format. If the FIFO is empty, this module can be configured either to repeat the last sample present in the FIFO or to transmit zeros.

The clock management block provides the BCLK and LRCLK generator functions.

The RX FIFO and TX FIFO decouple the AIF clock domain from the host system clock domain. Each FIFO holds a maximum of 64 samples. The status and number of samples in each FIFO are

accessible by means of memory mapped registers and ports. The back-end interface supports blocking transactions.

AIF INTERFACE FORMATS

The AIF digital audio interface ports comprise 4 external connections:

- AIFnTXDAT - Data output
- AIFnRX_DAT - Data input
- AIFnLRCLK - Left/Right frame alignment clock
- AIFnBCLK - Bit clock, for data synchronisation

In Master mode, the clock signals BCLK and LRCLK are outputs from the WM0011. In Slave mode, these signals are inputs.

The AIF data format is highly configurable, using the AIF_DATA_CFG and AIF_CLK_CFG registers (see Table 173 and Table 174). The AIF modules support I2S, Left-Justified, Right-Justified, DSP Mode-A, DSP Mode-B formats, and many others. Typical configurations are described and illustrated below.

In I^2S mode, the MSB is available on the second rising edge of BCLK following a LRCLK transition. The other bits up to the LSB are then transmitted in order. Depending on the Channel length and Sample length configuration, there may be unused BCLK cycles between the LSB of one sample and the MSB of the next.

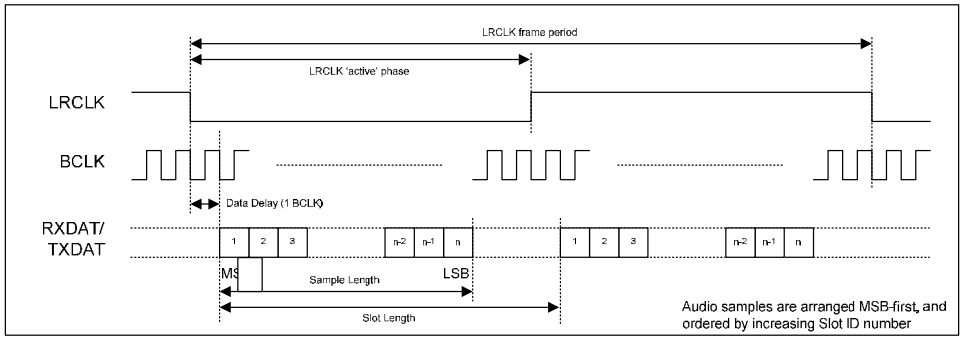


Figure 52 I2S Justified Audio Interface

In Left Justified mode, the MSB is available on the first rising edge of BCLK following a LRCLK transition. The other bits up to the LSB are then transmitted in order. Depending on the Channel length and Sample length configuration, there may be unused BCLK cycles before each LRCLK transition.

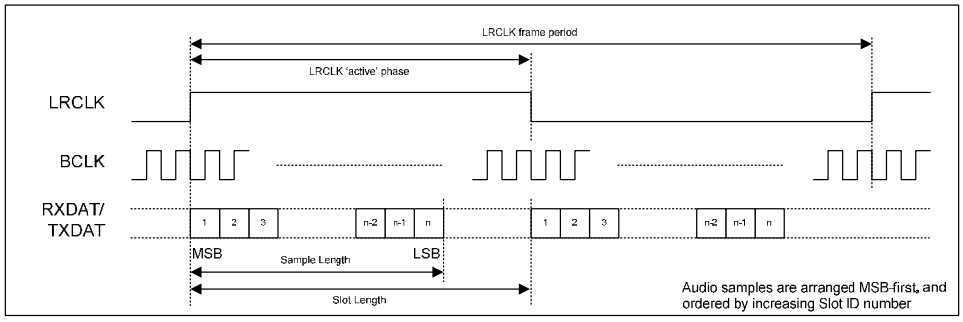


Figure 53 Left Justified Audio Interface

In Right Justified mode, the LSB is available on the last rising edge of BCLK before a LRCLK transition. All other bits are transmitted before (MSB first). Depending on the Channel length and Sample length configuration, there may be unused BCLK cycles after each LRCLK transition.

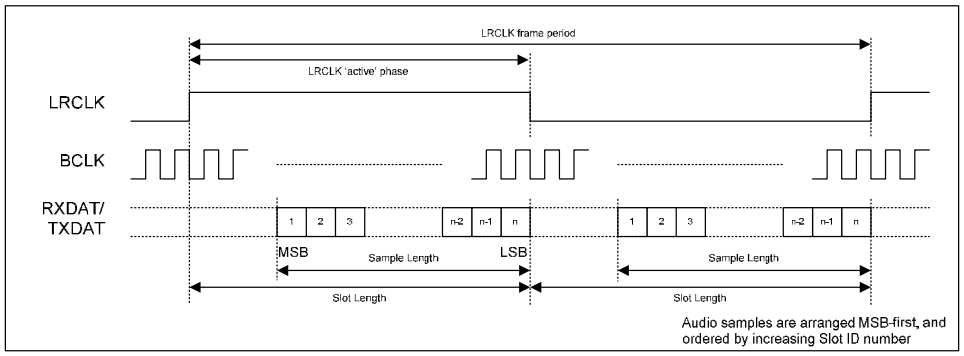


Figure 54 Right Justified Audio Interface

Many other AIF data formats can also be defined, supporting two or more channels of audio data. Dual phase mode can also be selected, allowing mixed-configuration sample slots for each channel. As an example, Figure 55 shows a format comprising 2 x 24-bit samples (Phase 1), followed by 4 x 16-bit samples (Phase 2). The first sample is delayed by 1 x BCLK cycle relative to the leading edge of the Frame Sync (LRCLK) signal.

Refer to the AIF_DATA_CFG and AIF_CLK_CFG register descriptions (Table 173 and Table 174) for further details on how to configure the AIF data format.

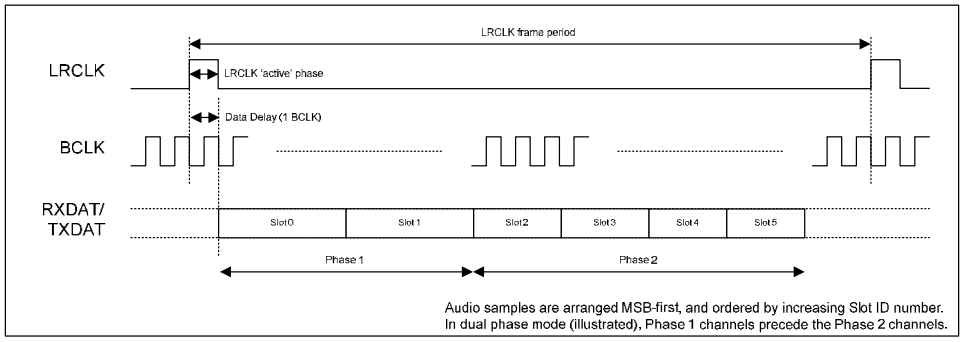
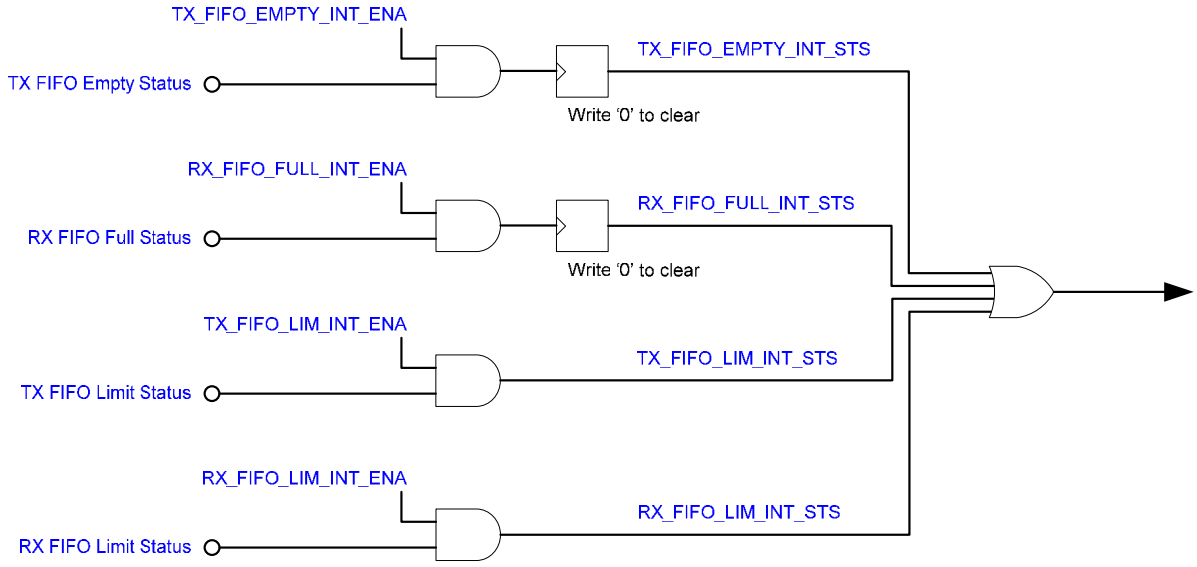


Figure 55 Multi-Channel Audio Interface

AIF INTERRUPTS

The AIF module can generate an interrupt when any of the conditions described in the AIF_INT_CTRL register occurs. The interrupt conditions provide status indications of the AIF TX and RX data buffers.

The AIF interrupt control registers are illustrated in Figure 56.



The interrupt control functions are replicated for each of the 3 AIF modules.

Figure 56 AIF Interrupts

AIF REGISTER MAP

The register map of the AIF module is illustrated in Table 164.

ADDRESS	REGISTER	DESCRIPTION	RESET VALUE
Base + 0x00	AIF_RX_DAT	AIF Receive Data	0x0000_0000
Base + 0x04	AIF_RX_CH_ID	AIF Receive Channel ID	0x0000_0000
Base + 0x08	AIF_RX_STS	AIF Receive FIFO Status	0x0000_0000
Base + 0x10	AIF_RX_LIMIT	AIF Receive FIFO Upper Limit	0x0000_FFFF
Base + 0x20	AIF_TX_DAT	AIF Transmit Data	0x0000_0000
Base + 0x24	AIF_TX_CH_ID	AIF Transmit Channel ID	0x0000_0000
Base + 0x28	AIF_TX_STS	AIF Transmit FIFO Status	0x0000_0000
Base + 0x30	AIF_TX_LIMIT	AIF Transmit FIFO Lower Limit	0x0000_0000
Base + 0x40	AIF_DATA_CFG	AIF Data Configuration	0x01AC_01A4
Base + 0x44	AIF_CLK_CFG	AIF Serial Clocking Configuration	0x01F1_03F0
Base + 0x48	AIF_CTRL	AIF Control	0x0000_0022
Base + 0x4C	AIF_INT_CTRL	AIF Interrupt Control	0x0000_0000
Base + 0x60	AIF_MCLK_DIV	AIF MCLK Divider	0x0000_0000

Table 164 AIF Register Definition

AIF_RX_DAT – AIF RECEIVE DATA REGISTER

This register contains the received data from the RX FIFO. The audio samples have their MSB in bit 31, regardless of the number of bits per sample and the left/right justification being used.

The AIF_RX_DAT register can only be accessed when the RX FIFO is enabled using the AIF_CTRL register (see Table 175). For read access to the AIF_RX_DAT register, it is required that AIF_RX_ENA=1 and AIF_RX_RST=0.

The AIF_RX_DAT register cannot be read when the RX FIFO is empty. The RX FIFO status can be checked using the RX_EMPTY_STS bit in the AIF_RX_STS register (see Table 167).

Note that any attempt to read AIF_RX_DAT when the conditions described above do not support access may cause incorrect device behaviour. The restrictions noted also apply when accessing the register via the JTAG debug interface.

AIF_RX_DAT
AIF RECEIVE DATA REGISTER

Address = 0xF070_0000 (AIF 1)
Address = 0xF080_0000 (AIF 2)
Address = 0xF090_0000 (AIF 3)

Default value = 0x0000_0000

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	AIF_RX_DAT	RO	0x0000 0000	Rx Data. The audio samples have their MSB in bit 31, regardless of the number of bits per sample and the left/right justification being used.

Table 165 AIF_RX_DAT Register

AIF_RX_CH_ID – AIF RECEIVE CHANNEL ID REGISTER

This register indicates the channel number of the last audio sample read from the AIF_RX_DAT register.

AIF_RX_CH_ID
AIF RECEIVE CHANNEL ID REGISTER

Address = 0xF070_0004 (AIF 1)
Address = 0xF080_0004 (AIF 2)
Address = 0xF090_0004 (AIF 3)

Default value = 0x0000_0000

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:8	Reserved		0x00_0000	
7:0	RX_SLOT_ID	RO	0x00	Slot ID RX Slots are identified by an integer from 0 to [N-1]

Table 166 AIF_RX_CH_ID Register

AIF_RX_STS – AIF RECEIVE FIFO STATUS REGISTER

This register indicates the number of samples currently in the RX FIFO.

AIF_RX_STS
AIF RECEIVE FIFO STATUS REGISTER

Address = 0xF070_0008 (AIF 1)
Address = 0xF080_0008 (AIF 2)
Address = 0xF090_0008 (AIF 3)

Default value = 0x0000_0000

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:29	Reserved		0x0	
28	RX_EMPTY_STS	RO	0x0	RX FIFO Empty/Full indication. 0 = not Empty, 1= Empty.
27:0	RX_FIFO_SAMPLES	RO	0x000 0000	Number of samples in the RX FIFO

Table 167 AIF_RX_STS Register

AIF_RX_LIMIT – AIF RECEIVE FIFO UPPER LIMIT REGISTER

This register holds the RX FIFO Upper Limit value.

When the number of samples in the RX FIFO exceeds the Upper Limit value, the RX_FIFO_LIM_INT_STS interrupt will be asserted (if enabled by the RX_FIFO_LIM_INT_ENA bit in the AIF_INT_CTRL register).

The DMA handshake to the RX FIFO is also triggered by the same Upper Limit value (when enabled by RX_FIFO_LIM_DMA_ENA). The DMA operation will not execute while the number of samples in the buffer is less than or equal to AIF_RX_LIMIT.

The RX FIFO buffer size is 64 samples. Therefore, to support the functionality described above, the AIF_RX_LIMIT is valid from 0 to 63.

AIF_RX_LIMIT
AIF RECEIVE FIFO UPPER LIMIT REGISTER

Address = 0xF070_0010 (AIF 1)
Address = 0xF080_0010 (AIF 2)
Address = 0xF090_0010 (AIF 3)

Default value = 0x0000_FFFF

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	AIF_RX_LIMIT	RW	0xFFFF	RX FIFO Upper Limit value. When the number of samples in the RX FIFO exceeds the Upper Limit value, the RX_FIFO_LIM_INT_STS interrupt and RX_FIFO_LIM_DMA_STS handshake will be asserted (if enabled by the respective bits in the AIF_INT_CTRL register). To support Interrupt or DMA Handshake functionality, AIF_RX_LIMIT is valid from 0 to 63.

Table 168 AIF_RX_LIMIT Register

AIF_TX_DAT – AIF TRANSMIT DATA REGISTER

This register contains the data to be transmitted via the TX FIFO. The audio samples have their MSB in bit 31, regardless of the number of bits per sample and the left/right justification being used.

AIF_TX_DAT
AIF TRANSMIT DATA REGISTER

Address = 0xF070_0020 (AIF 1)
Address = 0xF080_0020 (AIF 2)
Address = 0xF090_0020 (AIF 3)

Default value = 0x0000_0000

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	AIF_TX_DAT	WO	0x0000_0000	Tx Data. The audio samples have their MSB in bit 31, regardless of the number of bits per sample and the left/right justification being used.

Table 169 AIF_TX_DAT Register

AIF_TX_CH_ID – AIF TRANSMIT CHANNEL ID REGISTER

This register indicates the channel number of the next audio sample that will be written to the AIF_TX_DAT register.

AIF_TX_CH_ID
AIF TRANSMIT CHANNEL ID REGISTER

Address = 0xF070_0024 (AIF 1)
Address = 0xF080_0024 (AIF 2)
Address = 0xF090_0024 (AIF 3)

Default value = 0x0000_0000

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:8	Reserved		0x00 0000	
7:0	TX_SLOT_ID	RW	0x00	Slot ID TX Slots are identified by an integer from 0 to [N-1]

Table 170 AIF_TX_CH_ID Register

AIF_TX_STS – AIF TRANSMIT FIFO STATUS REGISTER

This register holds the number of samples currently in the TX FIFO.

AIF_TX_STS
AIF TRANSMIT FIFO STATUS REGISTER

Address = 0xF070_0028 (AIF 1)
Address = 0xF080_0028 (AIF 2)
Address = 0xF090_0028 (AIF 3)

Default value = 0x0000_0000

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:30	Reserved		0x0	
29	TX_FULL_STS	RO	0x0	TX FIFO Empty/Full indication. 0 = not Full, 1= Full.
28	Reserved		0x0	
27:0	TX_FIFO_SAMPLES	RO	0x000 0000	Number of samples in the TX FIFO

Table 171 AIF_TX_STS Register

AIF_TX_LIMIT – AIF TRANSMIT FIFO LOWER LIMIT REGISTER

This register holds the TX FIFO Lower Limit value.

When the number of samples in the TX FIFO is less than the Lower Limit value, the TX_FIFO_LIM_INT_STS interrupt will be asserted (if enabled by the TX_FIFO_LIM_INT_ENA bit in the AIF_INT_CTRL register).

The DMA handshake to the TX FIFO is also triggered by the same Lower Limit value (when enabled by TX_FIFO_LIM_DMA_ENA). The DMA operation will not execute while the number of samples in the buffer is greater than or equal to AIF_TX_LIMIT.

The TX FIFO buffer size is 64 samples. Therefore, to support the functionality described above, the AIF_TX_LIMIT is valid from 1 to 64.

AIF_TX_LIMIT
AIF TRANSMIT FIFO LOWER LIMIT REGISTER

Address = 0xF070_0030 (AIF 1) **Address = 0xF080_0030 (AIF 2)** **Address = 0xF090_0030 (AIF 3)**	**Default value = 0x0000_0000**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:0	AIF_TX_LIMIT	RW	0x00	TX FIFO Lower Limit value. When the number of samples in the TX FIFO is less than the Lower Limit value, the TX_FIFO_LIM_INT_STS interrupt and TX_FIFO_LIM_DMA_STS handshake will be asserted (if enabled by the respective bits in the AIF_INT_CTRL register). To support Interrupt or DMA Handshake functionality, AIF_TX_LIMIT is valid from 1 to 64.

Table 172 AIF_TX_LIMIT Register

AIF_DATA_CFG – AIF DATA CONFIGURATION REGISTER

The AIF data format comprises a sequence of data words corresponding to as many data slots as are configured. The number of slots, number of bits per slot, and audio sample size are configurable. Each audio sample may be Left or Right justified within the allocated time slots. Each audio sample is transmitted/received MSB-first. The first sample can be delayed relative to the leading edge of the Frame Sync (LRCLK) signal using the AIF_DATA_DLY control field.

In Dual-Phase mode (AIF_DUAL_PHASE=1), the sequence comprises two phases, where each phase is independently configurable. This allows, for example, 'n' channels of 24-bit samples to be followed by 'm' channels of 16-bit samples in an efficient manner. Phase 1 is transmitted/received before Phase 2.

The timing and polarity of the Frame Sync (LRCLK) signal is configurable, as described in the AIF_CLK_CFG register (see Table 174).

The AIF data format is highly flexible, supporting I2S, Left-Justified, Right-Justified, DSP Mode-A, DSP Mode-B formats, and many others.

AIF_DATA_CFG
AIF DATA CONFIGURATION REGISTER

Address = 0xF070_0040 (AIF 1)
Address = 0xF080_0040 (AIF 2)
Address = 0xF090_0040 (AIF 3)

Default value = 0x01AC_01A4

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31	AIF_DUAL_PHASE	RW	0x0	0 = Single Phase frame (Phase 1 only) 1 = Dual Phase frame (Phase 1 & Phase 2)
30:24	SLOT_CNT_PH2	RW	0x01	Frame Length (number of slots) in phase 2 00h = 1 Slot 01h = 2 Slots … 7Fh = 128 Slots Only valid when AIF_DUAL_PHASE=1
23:21	SLOT_LEN_PH2	RW	0x5	Slot Length (number of bits per slot) in phase 2 0h = 8 bits 1h = 12 bits 2h = 16 bits 3h = 20 bits 4h = 24 bits 5h = 32 bits 6h = Reserved 7h = Reserved Only valid when AIF_DUAL_PHASE=1
20:19	AIF_DATA_DLY	RW	0x1	Data Delay select 0h = 0-bit data delay 1h = 1-bit data delay 2h = 2-bit data delay 3h = Reserved
18:16	SAMPLE_LEN_PH2	RW	0x4	Sample Length (sample length per slot) in phase 2 0h = 8 bits 1h = 12 bits 2h = 16 bits 3h = 20 bits 4h = 24 bits 5h = 32 bits 6h = Reserved 7h = Reserved Note that, if the Slot Length > Sample Length, then each Slot will be Left-Justified or Right-Justified depending on the AIF_FORMAT bit. Only valid when AIF_DUAL_PHASE=1
15	Reserved		0x0	
14:8	SLOT_CNT_PH1	RW	0x01	Frame Length (number of slots) in phase 1 00h = 1 Slot 01h = 2 Slots … 7Fh = 128 Slots

AIF_DATA_CFG
AIF DATA CONFIGURATION REGISTER

Address = 0xF070_0040 (AIF 1)
Address = 0xF080_0040 (AIF 2)
Address = 0xF090_0040 (AIF 3)

Default value = 0x01AC_01A4

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
7:5	SLOT_LEN_PH1	RW	0x5	Slot Length (number of bits per slot) in phase 1 0h = 8 bits 1h = 12 bits 2h = 16 bits 3h = 20 bits 4h = 24 bits 5h = 32 bits 6h = Reserved 7h = Reserved
4	Reserved		0x0	
3	AIF_FORMAT	RW	0x0	Audio Sample Justification 0 = Left Justified 1 = Right Justified
2:0	SAMPLE_LEN_PH1	RW	0x4	Sample Length (sample length per slot) in phase 1 0h = 8 bits 1h = 12 bits 2h = 16 bits 3h = 20 bits 4h = 24 bits 5h = 32 bits 6h = Reserved 7h = Reserved Note that, if the Slot Length > Sample Length, then each Slot will be Left-Justified or Right-Justified depending on the AIF_FORMAT bit.

Table 173 AIF_DATA_CFG Register

AIF_CLK_CFG – AIF SERIAL CLOCKING CONFIGURATION REGISTER

This register selects AIF Master or Slave mode, and defines the timing and polarity of the LRCLK signal. The sample edge for the RX and TX data can also be configured.

AIF_CLK_CFG
AIF SERIAL CLOCKING CONFIGURATION REGISTER

Address = 0xF070_0044 (AIF 1)
Address = 0xF080_0044 (AIF 2)
Address = 0xF090_0044 (AIF 3)

Default value = 0x01F1_03F0

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:28	Reserved			
27:20	AIF_LRCLK_LEN	RW	0x1F	Sets the length of the LRCLK active phase at the start of each frame. 00h = 1 BCLK cycle 01h = 2 BCLK cycles 02h = 3 BCLK cycles …etc. Default is 1Fh (32 BCLK cycles)
19	AIF_RX_EDGE	RW	0x0	Receive Data clock edge select 0 = AIFnRXDAT is sampled at the rising edge of AIFnBCLK 1 = AIFnRXDAT is sampled at the falling edge of AIFnBCLK
18	AIF_MSTR	RW	0x0	Master/Slave configuration 0 = Slave mode (BCLK and LRCLK are external inputs) 1 = Master mode (BCLK and LRCLK are generated as outputs)
17	AIF_TX_EDGE	RW	0x0	Transmit Data clock edge select 0 = AIFnTXDAT is valid at the rising edge of AIFnBCLK 1 = AIFnTXDAT is valid at the falling edge of AIFnBCLK
16	AIF_LRCLK_INV	RW	0x1	LRCLK Polarity select 0 = LRCLK is active high 1 = LRCLK is active low
15:4	AIF_LRCLK_PERIOD	RW	0x03F	Sets the duration of the LRCLK frame. 000h = 1 BCLK cycle 001h = 2 BCLK cycles 002h = 3 BCLK cycles …etc. Default is 03Fh (64 BCLK cycles)
3:1	Reserved			
0	AIF_TX_DAT_ENA	RW	0x0	AIFnTXDAT output enable 0 = Disabled 1 = Enabled Note that the AIFnTXDAT output pin is also controlled via the CCM I/O Control registers. It is recommended to set AIF_TX_DAT_ENA=1 at all times.

Table 174 AIF_CLK_CFG Register

AIF_CTRL – AIF CONTROL REGISTER

This register contains reset / enable control bits for the AIF modules.

AIF_CTRL AIF CONTROL REGISTER	
Address = 0xF070_0048 (AIF 1) **Address = 0xF080_0048 (AIF 2)** **Address = 0xF090_0048 (AIF 3)**	**Default value = 0x0000_0022**

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:6	Reserved			
5	AIF_RX_RST	RW	0x1	AIF RX Reset 0 = Do nothing 1 = Reset the RX registers and the TX/RX common registers. Flushes the RX FIFO
4	AIF_RX_ENA	RW	0x0	AIF RX Enable Controls whether RX data is written to the RX FIFO. 0 = Disabled 1 = Enabled
3:2	Reserved			
1	AIF_TX_RST	RW	0x1	AIF TX Reset 0 = Do nothing 1 = Reset the TX registers and the TX/RX common registers. Flushes the TX FIFO
0	AIF_TX_ENA	RW	0x0	AIF TX Enable Controls whether TX data is output from the TX FIFO. 0 = Disabled 1 = Enabled

Table 175 AIF_CTRL Register

AIF_INT_CTRL – AIF INTERRUPT CONTROL REGISTER

The AIF module can generate interrupts to indicate the TX and RX FIFO buffer status, as described in Table 176. Note that the Interrupt Status fields (bits [19:16]) can only be asserted when the respective Interrupt Enable bit is set.

The AIF Interrupt output to the Interrupt module is asserted when any of the enabled AIF interrupts are asserted.

The handshake (ACK) function for DMA transfers to/from the TX/RX FIFO buffers is controlled using the TX_FIFO_LIM_DMA_ENA and RX_FIFO_LIM_DMA_ENA fields. These must be enabled when using a DMA transfer of data to/from the respective buffer.

AIF_INT_CTRL
AIF INTERRUPT CONTROL REGISTER

Address = 0xF070_004C (AIF 1)
Address = 0xF080_004C (AIF 2)
Address = 0xF090_004C (AIF 3)

Default value = 0x0000_0000

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

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:22	Reserved		0x000	
21	TX_FIFO_LIM_DMA_STS	RO	0x0	TX FIFO Lower Limit DMA Handshake status 0 = TX FIFO Lower Limit has not been reached 1 = TX FIFO Lower Limit has been reached This bit automatically de-asserts when the Lower Limit condition is no longer met.
20	RX_FIFO_LIM_DMA_STS	RO	0x0	RX FIFO Upper Limit DMA Handshake status 0 = RX FIFO Upper Limit has not been reached 1 = RX FIFO Upper Limit has been reached This bit automatically de-asserts when the Upper Limit condition is no longer met.
19	TX_FIFO_EMPTY_INT_STS	RC	0x0	TX FIFO Empty Interrupt This bit asserted (logic '1') to indicate a TX FIFO read attempt when the TX buffer was empty. Write '1' to clear.
18	RX_FIFO_FULL_INT_STS	RC	0x0	RX FIFO Full Interrupt This bit asserted (logic '1') to indicate a RX FIFO write attempt when the RX buffer was full. Write '1' to clear.
17	TX_FIFO_LIM_INT_STS	RO	0x0	TX FIFO Lower Limit Interrupt 0 = TX FIFO Lower Limit has not been reached 1 = TX FIFO Lower Limit has been reached This Interrupt automatically de-asserts when the Lower Limit condition is no longer met.
16	RX_FIFO_LIM_INT_STS	RO	0x0	RX FIFO Upper Limit Interrupt 0 = RX FIFO Upper Limit has not been reached 1 = RX FIFO Upper Limit has been reached This Interrupt automatically de-asserts when the Upper Limit condition is no longer met.
15:6	Reserved		0x000	
5	TX_FIFO_LIM_DMA_ENA	RW	0x0	TX FIFO Lower Limit DMA Handshake Enable 0 = Disabled 1 = Enabled This bit must be set during a DMA transfer to the TX FIFO buffer.
4	RX_FIFO_LIM_DMA_ENA	RW	0x0	RX FIFO Upper Limit DMA Handshake Enable 0 = Disabled 1 = Enabled This bit must be set during a DMA transfer from the RX FIFO buffer.
3	TX_FIFO_EMPTY_INT_ENA	RW	0x0	TX FIFO Empty Interrupt Enable 0 = Disabled 1 = Enabled
2	RX_FIFO_FULL_INT_ENA	RW	0x0	RX FIFO Full Interrupt Enable 0 = Disabled 1 = Enabled
1	TX_FIFO_LIM_INT_ENA	RW	0x0	TX FIFO Lower Limit Interrupt Enable 0 = Disabled 1 = Enabled

AIF_INT_CTRL AIF INTERRUPT CONTROL REGISTER																															
Address = 0xF070_004C (AIF 1) **Address = 0xF080_004C (AIF 2)** **Address = 0xF090_004C (AIF 3)**																**Default value = 0x0000_0000**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
0	RX_FIFO_LIM_INT_ENA	RW	0x0	RX FIFO Upper Limit Interrupt Enable 0 = Disabled 1 = Enabled

Table 176 AIF_INT_CTRL Register

AIF_MCLK_DIV – AIF MCLK DIVIDER REGISTER

In AIF Master mode the AIF module generates the BCLK and LRCLK signals as outputs from the WM0011.

The BCLK output is generated as AIFn_MSTR_CLK (see Figure 16). The clock source is selected via a multiplexer, using the CLK_SEL_AIFn bits (see Table 19). The MCLK_AIFn signal, derived from PLLOUT, is one of the inputs to this multiplexer, and is configured as described below. See "Clocking" for further details.

The LRCLK output is derived from BCLK; the polarity, pulse length and frame period are configured using the AIF_CLK_CFG register (see Table 174).

The AIF_MCLK_DIV registers define the ratio of the PLLOUT frequency to the MCLK_AIFn frequency. MCLK_DIV_INTG defines the integer portion of the frequency ratio; MCLK_DIV_FRAC defines the fractional portion.

For example, if PLLOUT = 125MHz and the required BCLK frequency = 12.288MHz, the frequency ratio is approximately 10.172526. The corresponding register settings would be MCLK_DIV_INTG=0x00Ah, MCLK_DIV_FRAC=0x2C2AA.

When MCLK_AIFn is selected as the clock source (CLK_SEL_AIFn=1h), then the PLLOUT frequency ratio (MCLK_DIV) must be set to 4.0 or higher.

Note that the BCLK frequency can be calculated as Sample Frequency * AIF_LRCLK_PERIOD.

AIF_MCLK_DIV AIF MCLK DIVIDER REGISTER																															
Address = 0xF070_0060 (AIF 1) **Address = 0xF080_0060 (AIF 2)** **Address = 0xF090_0060 (AIF 3)**																**Default value = 0x0000_0000**														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	FIELD NAME	S/W ACCESS	RESET VALUE	FIELD DESCRIPTION
31:20	MCLK_DIV_INTG	RW	0x000	Integer portion of PLLOUT / BCLK ratio (Coded as LSB = 1) When MCLK_AIFn is the clock source (CLK_SEL_AIFn=1h), then MCLK_DIV_INTG must be set to 4 or higher.
19:0	MCLK_DIV_FRAC	RW	0x0 _0000	Fractional portion of PLLOUT / BCLK ratio (Coded as MSB = 0.5)

Table 177 AIF_MCLK_DIV Register

JTAG (JTAG) MODULE

For further details on the JTAG module please refer to the documentation available from Tensilica (www.tensilica.com).

CROSS-TRIGGER MODULE (CTM)

For further details on the Cross-Trigger module please refer to the documentation available from Tensilica (www.tensilica.com).

ON-CHIP DEBUG (OCD) MODULE

For further details on the On-Chip Debug module please refer to the documentation available from Tensilica (www.tensilica.com).

APPLICATIONS INFORMATION

To achieve a reasonable level of long term jitter, it is vital to deliver an analogue-grade power supply to the PLL via AVDD.

Board layout around the capacitor and the path from there to the AVDD and PLLC pins is critical. It is vital that the AVDD and power are treated as sensitive analogue signals.

The power (AVDD) path must be a single wire from the DSP pin to the capacitor, and then through the series resistor to board power (VDD). The distance from the DSP pin to the capacitor should be as short as possible.

Similarly, the ground (PLLC) path should be from the IC pin to the capacitor, with the distance from IC pin to capacitor being very short. This DSP has the PLL ground connection made on-chip, so the external PLLC connection must not be connected to PCB ground.

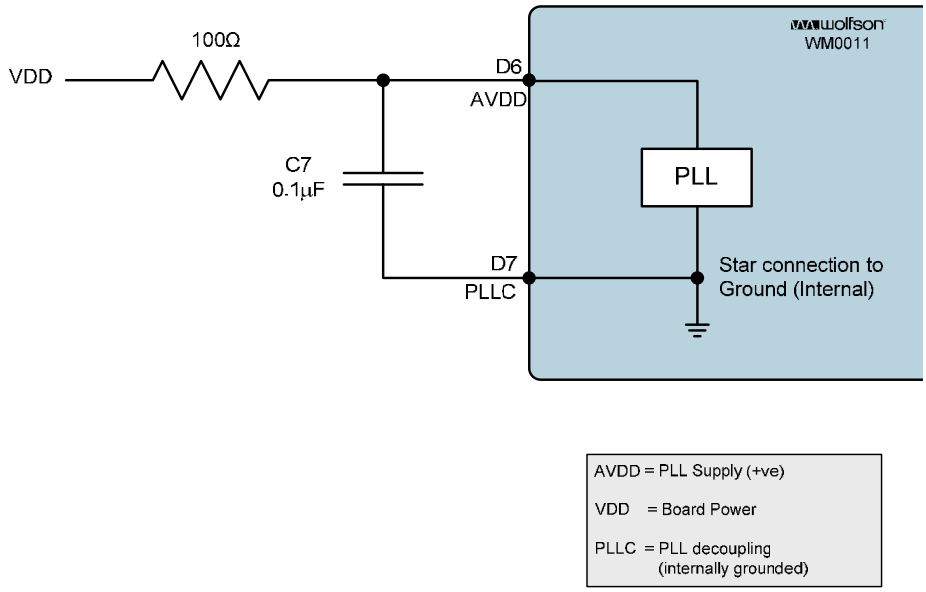
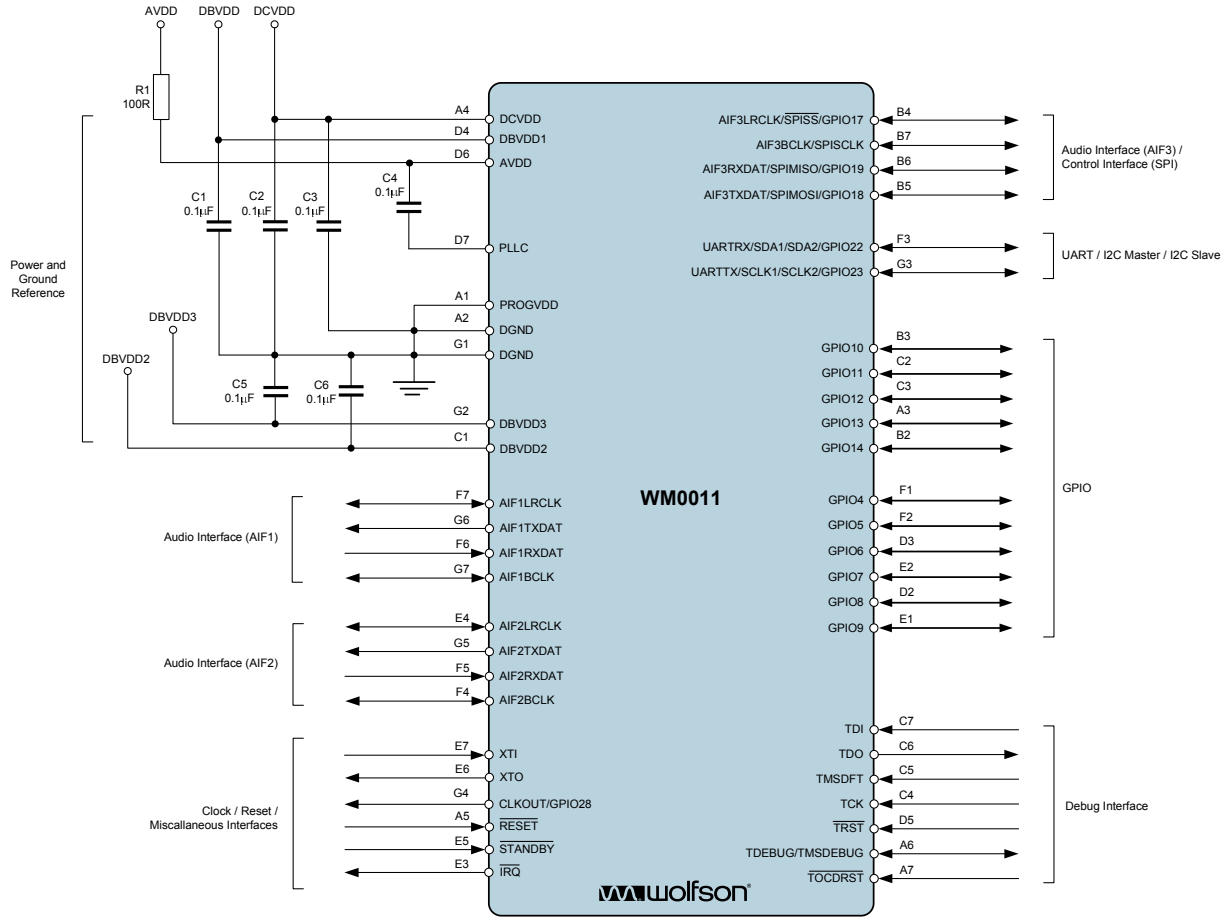


Figure 57 Recommended Filter Circuit for the PLL

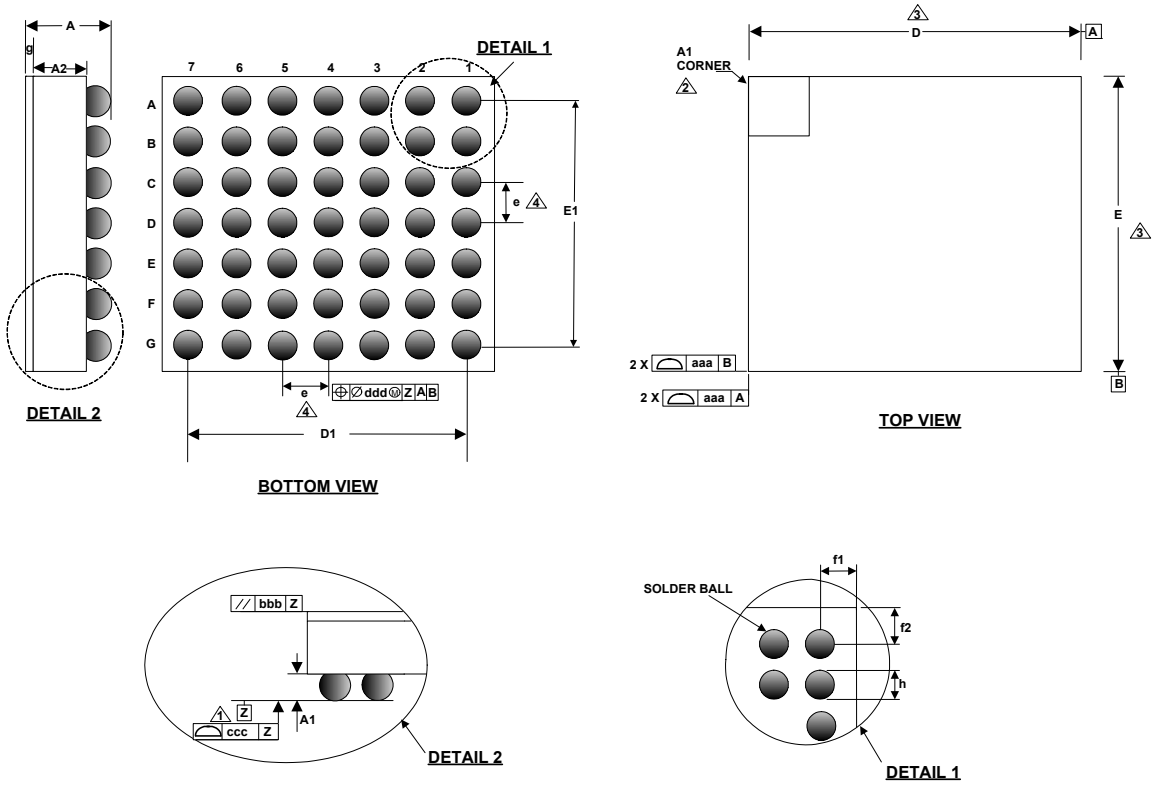
CONNECTIVITY DIAGRAM



Notes:
1. Decoupling track layout is critical. DCVDD and AVDD/PLLC decoupling must be placed as close to the device as possible.
2. Do not connect the PLLC to any ground source on the PCB. The PLL DC ground connection is made on chip, so the external ground connection must not be connected to PCB ground.
3. Observe best power supply design practice to accommodate the power demand of DSP

PACKAGE DIMENSIONS

B: 49 BALL W-CSP PACKAGE 3.055 x 3.055 x 0.636mm BODY, 0.40 mm BALL PITCH **DM114.B**



Symbols	Dimensions (mm)			
	MIN	NOM	MAX	NOTE
A	0.592	0.636	0.681	
A1	0.175	0.190	0.205	
A2	0.381	0.406	0.432	
D	3.000	3.055	3.080	
D1		2.400 BSC		
E	3.000	3.055	3.080	
E1		2.400 BSC		
e		0.400 BSC		4
f1	0.300	0.328		Bump centre to die edge
f2	0.300	0.328		Bump centre to die edge
h	0.216	0.270	0.324	
g	0.036	0.040	0.044	
aaa		0.10		
bbb		0.10		
ccc		0.03		
ddd		0.015		

NOTES:
1. PRIMARY DATUM -Z- AND SEATING PLANE ARE DEFINED BY THE SPHERICAL CROWNS OF THE SOLDER BALLS.
2. A1 CORNER IS IDENTIFIED BY INK/LASER MARK ON TOP PACKAGE.
3. BILATERAL TOLERANCE ZONE IS APPLIED TO EACH SIDE OF THE PACKAGE BODY.
4. 'e' REPRESENTS THE BASIC SOLDER BALL GRID PITCH.
5. THIS DRAWING IS SUBJECT TO CHANGE WITHOUT NOTICE.
6. FOLLOWS JEDEC DESIGN GUIDE MO-211-C.

IMPORTANT NOTICE

Wolfson Microelectronics plc ("Wolfson") products and services are sold subject to Wolfson's terms and conditions of sale, delivery and payment supplied at the time of order acknowledgement.

Wolfson warrants performance of its products to the specifications in effect at the date of shipment. Wolfson reserves the right to make changes to its products and specifications or to discontinue any product or service without notice. Customers should therefore obtain the latest version of relevant information from Wolfson to verify that the information is current.

Testing and other quality control techniques are utilised to the extent Wolfson deems necessary to support its warranty. Specific testing of all parameters of each device is not necessarily performed unless required by law or regulation.

In order to minimise risks associated with customer applications, the customer must use adequate design and operating safeguards to minimise inherent or procedural hazards. Wolfson is not liable for applications assistance or customer product design. The customer is solely responsible for its selection and use of Wolfson products. Wolfson is not liable for such selection or use nor for use of any circuitry other than circuitry entirely embodied in a Wolfson product.

Wolfson's products are not intended for use in life support systems, appliances, nuclear systems or systems where malfunction can reasonably be expected to result in personal injury, death or severe property or environmental damage. Any use of products by the customer for such purposes is at the customer's own risk.

Wolfson does not grant any licence (express or implied) under any patent right, copyright, mask work right or other intellectual property right of Wolfson covering or relating to any combination, machine, or process in which its products or services might be or are used. Any provision or publication of any third party's products or services does not constitute Wolfson's approval, licence, warranty or endorsement thereof. Any third party trade marks contained in this document belong to the respective third party owner.

Reproduction of information from Wolfson datasheets is permissible only if reproduction is without alteration and is accompanied by all associated copyright, proprietary and other notices (including this notice) and conditions.Wolfson is not liable for any unauthorised alteration of such information or for any reliance placed thereon.

Any representations made, warranties given, and/or liabilities accepted by any person which differ from those contained in this datasheet or in Wolfson's standard terms and conditions of sale, delivery and payment are made, given and/or accepted at that person's own risk. Wolfson is not liable for any such representations, warranties or liabilities or for any reliance placed thereon by any person.

ADDRESS:

Wolfson Microelectronics plc

Westfield House

26 Westfield Road

Edinburgh

EH11 2QB

Tel :: +44 (0)131 272 7000

Fax :: +44 (0)131 272 7001

Email :: sales@wolfsonmicro.com

REVISION HISTORY

DATE	REV	DESCRIPTION OF CHANGES	PAGE	CHANGED BY
2/08/12	1.0	Initial draft		PH
19/10/12	2.0	Updates to all sections, including some pin/register names.		PH
22/11/12	2.1	Maximum recommended DCVDD increased to 1.32V	14	PH
22/01/13	2.2	Pin Description updated, incorporating pull-up/down capabilities Max AVDD updated Electrical Characteristics updated Signal Timing Requirements updated Clocking diagram updated to incorporate TMRn_CLK signals Updates to PLL description, registers, and configuration examples Miscellaneous updates to I/O Control Registers Clarifications and updates to I2C module description Updates to GPIO/IRQC edge detect control register descriptions Minor clarifications to SPI module description Additions and edits in DMA module description, including examples Minor clarifications to AIF module description UART module description added		PH
06/02/13	3.0	I2S TDM mode deleted I2C 10-bit address mode deleted		PH
07/02/13	3.0	Block diagram updated (CLK DIV now labeled as Chip Config Module) 10-bit I2C addressing deleted I2S TDM mode deleted Typical Power Consumption data added Miscellaneous minor clarifications and corrections		PH
20/03/13	4.0	TRAX module description added Correction to Memory Map definition (APB Bridge space)		PH
20/05/13	4.0	Pin Description updates (name changes only) Minor clarifications to Warm Reset, Sleep/Wake-Up, AIF Bypass and SPI module descriptions Significant clarifications to I2C module description Notes added for avoidance of false interrupts in GPIO and IRQC. Noted requirements for accessing AIF_RX_DAT register.		PH
21/08/13	4.1	Front page description updated	1	JMacD