CEC1702

Cryptographic Embedded Controller

- 3.3V and 1.8V Operation
- ACPI Compliant
- VTR (standby) and VBAT Power Planes
  - Low Standby Current in Sleep Mode
- ARM® Cortex®-M4 Processor Core
  - 32-Bit ARM v7-M Instruction Set Architecture
  - Hardware Floating Point Unit (FPU)
  - Single 4GByte Addressing Space (Von Neumann Model)
  - Little-Endian Byte Ordering
  - Bit-Banding Feature Included
  - NVIC Nested Vectored Interrupt Controller
    - Up to 240 Individually-Vectored Interrupt Sources Supported
    - 8 Levels of Priority, Individually Assignable By Vector
    - Chip-Level Interrupt Aggregator supported, to expand number of interrupt sources or reduce number of vectors
  - System Tick Timer
  - Complete ARM-Standard Debug Support
    - JTAG-Based DAP Port, Comprised of SWJ-DP and AHB-AP Debugger Access Functions
    - Full DWT Hardware Functionality: 4 Data Watchpoints and Execution Monitoring
    - Full FPB Hardware Breakpoint Functionality: 6 Execution Breakpoints and 2 Literal (Data) Breakpoints
  - Comprehensive ARM-Standard Trace Support
    - Full DWT Hardware Trace Functionality for Watchpoint and Performance Monitoring
    - Full ITM Hardware Trace Functionality for Instrumented Firmware Support and Profiling
    - Full TPIU Functionality for Trace Output Communication
  - MPU Feature
  - 1µS Delay Register
- Internal Memory
  - 64k Boot ROM
  - Two blocks of SRAM, totaling 480KB
    - Each block can be used for either program or data
  - 128 Bytes Battery Powered SRAM
- Battery Backed Resources
  - Power-Fail Status Register
  - 32 KHz Clock Generator
  - Week Alarm Timer Interface
  - Real Time Clock
  - VBAT-Powered Control Interface
    - Two Wake-up Input Signals
    - Optional Latching of Wake-up Inputs
  - VBAT-Backed 128 Byte Memory
- Four I^2C Host Controllers
  - Allows Master or Dual Slave Operation
  - Fully Operational on Standby Power
  - DMA-driven I^2C Network Layer Hardware
  - I^2C Datalink Compatibility Mode
  - Multi-Master Capable
  - Supports Clock Stretching
  - Programmable Bus Speed up to 1MHz
  - Hardware Bus Access “Fairness” Interface
  - SMBus Time-outs Interface
  - All Ports Assignable to Any Controller
  - All ports 1.8V-capable
- General Purpose Serial Peripheral Interface Controller
  - One 4-pin Full Duplex Serial Communication Interface
  - Flexible Clock Rates
  - SPI Burst Capable
- One Quad Serial Peripheral Interface (SPI) Controller
  - Master Only SPI Controller
  - Mappable to two ports (only 1 port active at a time)
  - Dual and Quad I/O Support
  - Flexible Clock Rates
  - SPI Burst Capable
  - SPI Controller Operates with Internal DMA Controller with CRC Generation
- 13 x 8 Interrupt Capable Multiplexed Keyboard Scan Matrix
  - Optional Push-Pull Drive for Fast Signal Switching
- Two Breathing/Blinking LED Interfaces
  - Supports three modes of operation:
    - Blinking Mode with Programmable Blink Rates
    - Breathing LED Output
    - 8-bit PWM
  - Breathing LED Supports Piecewise-linear Brightness Curves, Symmetric or Asymmetric
  - Supports Low Power Operation in Blinking and Breathing Modes
    - Operates on Standby Power
    - Operates in Chip's System Deepest Sleep State on 32kHz standby clock
    - Operational in EC Sleep State
  - Pin buffers capable of sinking up to 12 mA
- Two Resistor/Capacitor Identification Detection (RC_ID) ports
  - Single Pin Interface to External Inexpensive RC Circuit
  - Replacement for Multiple GPIO’s
  - Provides 8 Quantized States on One Pin
- General Purpose I/O Pins
  - Up to 65 GPIOs

  - Glitch protection on most GPIO pins
  - 1 Battery-powered General Purpose Outputs
  - All GPIOs can be powered by 1.8V
  - Programmable Drive Strength and Slew Rate on all GPIOs
- Programmable 16-bit Counter/Timer Interface
  - Four 16-bit Auto-reloading Counter/Timer Instances
  - Four Operating Modes per Instance: Timer, One-shot, Event and Measurement
    - 3 External Inputs
    - 2 External Outputs
- Hibernation Timer Interface
  - Two 32.768 KHz Driven 16-bit Timers
    - Programmable Wake-up from 0.5ms to 128 Minutes
  - One 32.768 KHz Driven 32-bit RTOS Timer
    - Programmable Wake-up from 30µS to 35 Hours
    - Auto Reload Option
- System Watch Dog Timer (WDT)
- Input Capture Timer
  - 32-bit Free-running timer
  - Four 32-bit Capture Registers
  - One Compare Timer with Optional Toggling Output
  - Capture Interrupts with Programmable Edge Detection
  - Compare Timer and Counter Overflow Interrupts
- Week Timer
  - Power-up Event Output
  - Week Alarm Interrupt with 1 Second to 8.5 Year Time-out
  - Sub-Week Alarm Interrupt with 0.50 Seconds - 72.67 hours time-out
  - 1 Second and Sub-second Interrupts
- Real Time Clock (RTC)
  - VBAT Powered
  - 32KHz Crystal Oscillator
  - Time-of-Day and Calendar Registers
  - Programmable Alarms
  - Supports Leap Year and Daylight Savings Time
- Pulse-Width Modulator Support
  - Seven Programmable PWM Outputs
    - Multiple Clock Rates
    - 16-Bit ‘On’ and 16-Bit ‘Off’ Counters
  - Optional Inverted Output
- FAN Support
  - Two Fan Tachometer Inputs
  - Two RPM-Based Fan Speed Controllers
    - Each includes one Tach input and one PWM output
    - 3% accurate from 500 RPM to 16k RPM
    - Automatic Tachometer feedback
    - Aging Fan or Invalid Drive Detection
    - Spin Up Routine
    - Ramp Rate Control
    - RPM-based Fan Speed Control Algorithm
- ADC Interface
  - 10-bit Conversion in 1µs
  - 5 Channels
  - Integral Non-Linearity of ±1.5 LSB; Differential Non-Linearity of ±1.0 LSB
- Two Standard 16C550 UARTs
  - Both UARTs with 4-pin Interface
  - Programmable Input/output Pin Polarity Inversion
  - Programmable Main Power or Standby Power Functionality
- Trace FIFO Debug Port (TFDP)
- Integrated Standby Power Reset Generator
  - Reset Input Pin
- Clock Generator
  - 32.768KHz Clock Source
    - Low power 32KHz crystal oscillator
    - Optional use of a crystal-free silicon oscillator with ±2% Accuracy
    - Optional use of 32.768 KHz input Clock
    - Operational on Suspend Power
  - Programmable Clock Power Management Control and Distribution
  - 48 MHz PLL
- Multi-purpose AES Cryptographic Engine
  - Hardware support for ECB, CTR, CBC and OFB AES modes
  - Support for 128-bit, 192-bit and 256-bit key length
  - DMA interface to SRAM, shared with Hash engine
- Cryptographic Hash Engine
  - Support for SHA-1, SHA-256, SHA-512
  - DMA interface to SRAM, shared with AES engine
- Public Key Cryptographic Engine
  - Hardware support for RSA and Elliptic Curve public key algorithms
  - RSA keys length from 1024 to 4096 bits
  - ECC Prime Field and Binary Field keys up to 640 bits
  - Microcoded support for standard public key algorithms
- Cryptographic Features
  - True Random Number Generator
    - 1K bit FIFO
  - Monotonic Counter
- Boot ROM Secure Boot Loader
  - Hardware Root of Trust (RoT) using Secure Boot and Immutable Code
  - Supports 2 Code Images in external SPI
- Flash (Primary and Fallback image)
  - Authenticates SPI Flash image before loading
  - Support AES-256 Encrypted SPI Flash images
- Package
  - 84 Pin WFBGA RoHS Compliant package

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@microchip.com**. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000000A is version A of document DS30000000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Web site; http://www.microchip.com
- Your local Microchip sales office (see last page)

When contacting a sales office, please specify which device, revision of silicon and data sheet (include -literature number) you are using.

Customer Notification System

Register on our web site at **www.microchip.com** to receive the most current information on all of our products.

Table of Contents

1.0 GENERAL DESCRIPTION

The CEC1702 is a family of embedded controller designs with strong cryptographic support, customized for Internet of Things (IOT) platforms. The family is a highly-configurable, mixed signal, advanced I/O controller architecture. The device incorporates a 32-bit ARM Cortex M4F Microcontroller core with a closely-coupled SRAM for code and data. A secure boot-loader is used to download the custom firmware image from the system's shared SPI Flash device, thereby allowing system designers to customize the device's behavior.

The CEC1702 is directly powered by a minimum of two separate suspend supply planes (VBAT and VTR). There are three voltage supply regions for all GPIO pins. Two regions may be either 3.3V or 1.8V.

The CEC1702 family of devices offer a software development system interface that includes a Trace FIFO Debug port and a JTAG/SWD debug interface.

1.1 Family Features

TABLE 1-1: CEC1702 FEATURE LIST BY PACKAGE

CEC1702 Product Family	CEC1702
Package	84 WFBGA
Device ID	31h
Boundary Scan JTAG ID	021F2445h
SRAM Block (Primary use: code)	416KB
SRAM Block (Primary use: data)	64KB
Battery Backed SRAM	128 bytes
Trace FIFO Debug Port	Yes
Internal DMA Channels	14
16-bit Basic Timer	4
32-bit Basic Timer	2
16-bit Counter/Timer	4
Capture Timer	4
Compare Timer	1
Watchdog Timer (WDT)	1
Hibernation Timer	2
Week Timer	1
RTC	1
Battery-Powered General Purpose Output (BGPO)	1
Active Low VBAT-Powered Control Interface (VCI)	2
Keyboard Matrix Scan Support	13x8
I2C Host Controllers	4
I2C Ports	6
GPIOs	65
Pass-through GPIOs	2
Blinking/Breathing PWM	2
General Purpose SPI Master Controller	1
Quad SPI Master Controller	1
10-bit ADC Channels	5
16-bit PWMs	7
16-bit TACHs	2
UARTs	2
AES Hardware Support	128-256 bit
SHA Hashing Support	SHA-1 to SHA-512

TABLE 1-1: CEC1702 FEATURE LIST BY PACKAGE (CONTINUED)

CEC1702 Product Family	CEC1702
Public Key Cryptography Support	RSA: 4K bit ECC: 640 bit
True Random Number Generator	1K bit
Root Of Trust	Yes
Secure Boot	Yes
Immutable code	Yes

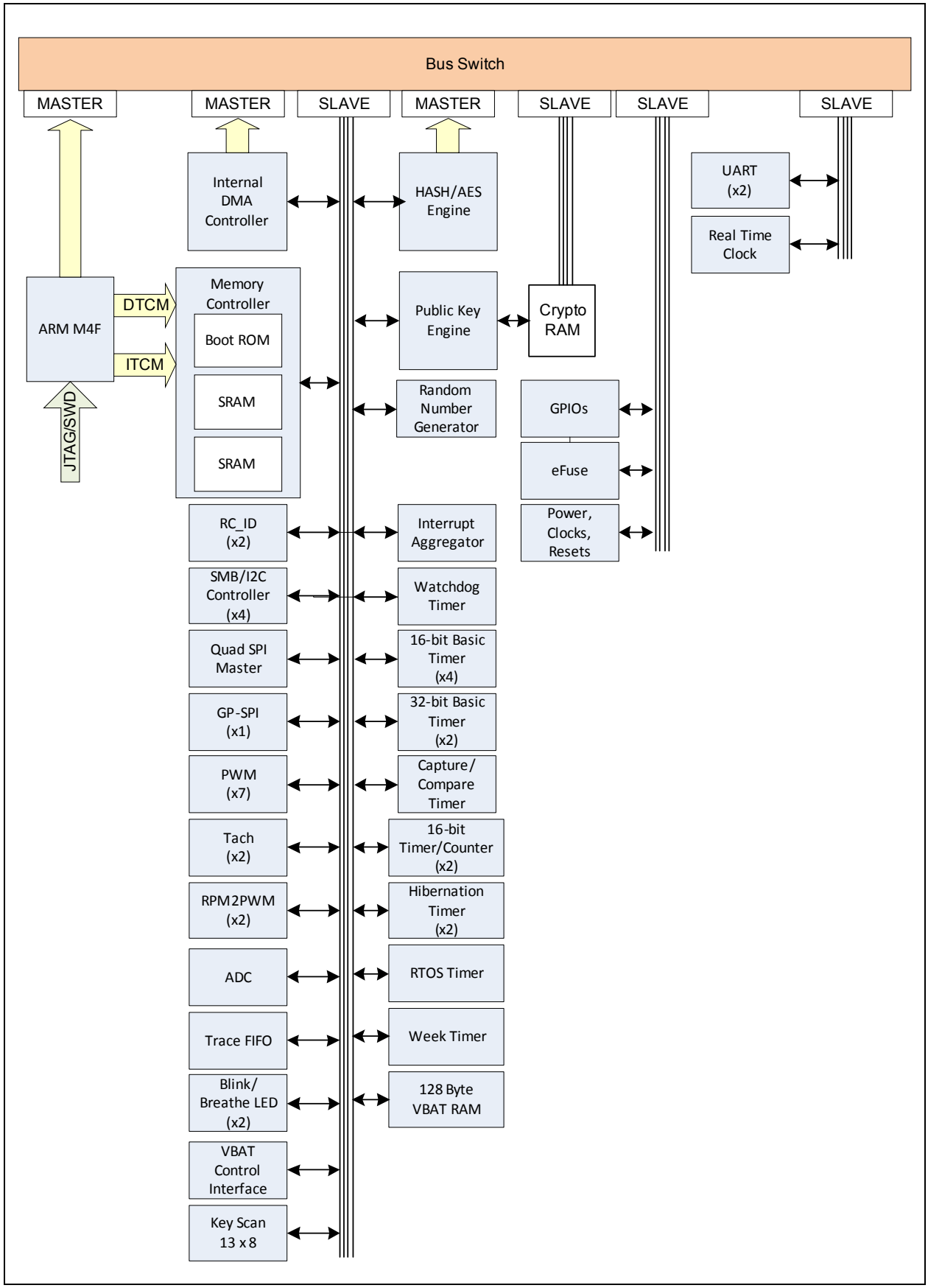
1.2 Boot ROM

Following the release of the RESET_EC signal, the processor will start executing code in the Boot ROM. The Boot ROM executes the SPI Flash Loader, which downloads User Code from an external SPI Flash and stores it in the internal Code RAM. Upon completion, the Boot ROM jumps into the User Code and starts executing as defined in the CEC1702 ROM Description Addendum.

The Boot ROM loads code from an external SPI Flash device. The interface supports SPI devices with dual and quad data rates, in addition to standard SPI devices. The downloaded code must configure the device's pins according to the platform's needs. After loading code, the Boot ROM leaves all pins in their default initial state.

1.3 CEC1702 Block Diagram

Note: Not all features shown are available on all devices. Refer to Table 1-1, "CEC1702 Feature List by Package" for a list of the features by device.


Bus Switch
MASTER
MASTER
SLAVE
MASTER
SLAVE
SLAVE
SLAVE
ARM M4F
DTCM
ITCM
JTAG/SWD
Internal DMA Controller
Memory Controller
Boot ROM
SRAM
SRAM
HASH/AES Engine
Public Key Engine
Random Number Generator
Crypto RAM
GPIOs
eFuse
Power, Clocks, Resets
UART (x2)
Real Time Clock
RC_ID (x2)
SMB/I2C Controller (x4)
Quad SPI Master
GP-SPI (x1)
PWM (x7)
Tach (x2)
RPM2PWM (x2)
ADC
Trace FIFO
Blink/ Breathe LED (x2)
VBAT Control Interface
Key Scan 13 x 8
Interrupt Aggregator
Watchdog Timer
16-bit Basic Timer (x4)
32-bit Basic Timer (x2)
Capture/ Compare Timer
16-bit Timer/Counter (x2)
Hibernation Timer (x2)
RTOS Timer
Week Timer
128 Byte VBAT RAM

2.0 PIN CONFIGURATION

2.1 Description

The Pin Configuration chapter includes Pin List By Pin Name, Signal Description by Signal, Notes for Tables in this Chapter, Pin Default State Through Power Transitions, and Package.

2.2 Terminology and Symbols for Pins/Buffers

2.2.1 BUFFER TERMINOLOGY

Term	Definition
#	The '#' sign at the end of a signal name indicates an active-low signal
n	The lowercase 'n' preceding a signal name indicates an active-low signal
PWR	Power
PIO	Programmable as Input, Output, Open Drain Output, Bi-directional or Bi-directional with Open Drain Output. Configurable drive strength from 2ma to12ma. **Note:** All GPIOs have programmable drive strength options of 2ma, 4ma, 8ma and 12ma. GPIO pin drive strength is determined by the DRIVE_STRENGTH field in the Pin Control 2 Register.
In	Input only - I Type Input Buffer.
O2ma	O-2 mA Type Buffer.

2.2.2 PIN NAMING CONVENTIONS

1. Pin Name is composed of the multiplexed options separated by '/'. E.g., GPIOxxxx/SignalA/SignalB.
2. Parenthesis '()' are used to list aliases or alternate functionality for a single mux option. E.g. GPIOxxx(Alias)/SignalA/SignalB. The Alias is the intended usage for a specific GPIO. E.g., GPIOxxx(ICSP_DATA) is intended to indicate that ICSP_DATA signal may come out on this pin when the Mux Control is set for GPIOxxx. In this case, enabling the test mode takes precedence over the Mux Control selection.
3. Signal Names appended with a numeric value indicates the Instance Number. E.g., PWM0, PWM1, etc. indicates that PWM0 is the PWM output for PWM Instance 0, PWM1 is the PWM output for PWM Instance 1, etc. Note that this same instance number is shown in the Register Base Address tables linking the specific PWM block instance to a specific signal on the pinout. The instance number may be omitted if there in only one instance of the IP block implemented.

2.3 Notes for Tables in this Chapter

Note	Description
Note 1	When the JTAG_RST# pin is not asserted (logic'1'), the pins for the signal functions in the JTAG/SWD interface are unconditionally routed to the interface; the Pin Control register for these pins has no effect. When the JTAG_RST# pin is asserted (logic'0'), the signal functions in the JTAG/SWD interface are not routed to the interface and the Pin Control Register for these pins controls the muxing. The pin control registers can not route the JTAG interface to the pins. System Board Designer should terminate this pin in all functional state using jumpers and pull-up or pull down resistors, etc.
Note 2	I2C/SMBus Port pins can be mapped to any I2C/SMB Controller. The number in the I2C/SMBus signal names (I2Cxx_DATA) indicates the port value. E.g. I2C01_DATA represents I2C/SMBus Data Port 1
Note 3	VCI_IN# function works even when configured as GPIO.
Note 4	The Voltage Regulator Capacitor (VR_CAP) pin requires an external 1uF capacitor and a voltage range of 1.08V (min) to 1.32V (max).

2.4 Pin List By Pin Name

2.4.1 DEFAULT STATE

The default state for analog pins is Input. The default state for all pins that default to a GPIO function is also input, with pull-up and pull-down resistors disabled. The default state for pins that differ is shown in the following table. Entries for the Default State column are

- O2ma-Low: Push-Pull output, Slow slew rate, 2ma drive strength, grounded
- O2ma-High Push-Pull output, Slow slew rate, 2ma drive strength, high output
- In-PU Input, with pull-up resistor enabled

CEC1702-84	Signal	Default (if not GPIO)	Default State (if not In)
B1	BGPO0	BGPO0	O2ma-Low
J6	GPIO001/PWM4		
J5	GPIO002/PWM5		
A3	GPIO003/I2C00_SDA/SPI0_CS#		
B2	GPIO004/I2C00_SCL/SPI0_MOSI		
B5	GPIO007/I2C03_SDA		
A8	GPIO010/I2C03_SCL		
K1	GPIO012/I2C07_SDA/TOUT3		
J2	GPIO013/I2C07_SCL/TOUT2		
K2	GPIO016/GPTP-IN7/QSPI0_IO3/ICT3		
J7	GPIO017/GPTP-IN5/KSI0		
J3	GPIO020/KSI1		
J4	GPIO021/KSI2		
K9	GPIO026/TIN1/KSI3		
J10	GPIO027/TIN2/KSI4		
G7	GPIO030/TIN3/KSI5		
J8	GPIO031/KSI6		
H5	GPIO032/KSI7		
G4	GPIO034/RC_ID1/SPI0_CLK		
F10	GPIO036/RC_ID2/SPI0_MISO		
K8	GPIO040/KSO00		
C5	GPIO045/KSO01		
C9	GPIO046/KSO02		
B8	GPIO047/KSO03		
F3	GPIO050/FAN_TACH0/GTACH0		
E2	GPIO051/FAN_TACH1/GTACH1		
K10	GPIO053/PWM0/GPWM0		
J9	GPIO054/PWM1/GPWM1		
K7	GPIO055/PWM2/QSPI0_CS#		
K6	GPIO056/PWM3/QSPI0_CLK		
D9	GPIO104/UART0_TX		
E9	GPIO105/UART0_RX		

CEC1702-84	Signal	Default (if not GPIO)	Default State (if not In)
H9	GPIO107/KSO04		
G9	GPIO112/KSO05		
G10	GPIO113/KSO06		
H10	GPIO120/KSO07		
D10	GPIO121/QSPI1_IO0/KSO08		
B10	GPIO122/QSPI1_IO1/KSO09		
C10	GPIO124/QSPI1_CS#/KSO11		
A10	GPIO125/GPTP-OUT5/QSPI1_CLK/KSO12		
C6	GPIO127/UART0_CTS#		
E10	GPIO134/PWM10/UART1_RTS#		
E7	GPIO135/UART1_CTS#		
H6	GPIO140/ICT5		
C2	GPIO145/I2C09_SDA/JTAG_TDI		
B6	GPIO146/I2C09_SCL/JTAG_TDO		
A7	GPIO147/I2C08_SDA/JTAG_CLK		
B3	GPIO150/I2C08_SCL/JTAG_TMS		
A9	GPIO154/I2C02_SDA		
B7	GPIO155/I2C02_SCL		
D7	GPIO156/LED0		
B9	GPIO157/LED1		
A5	GPIO162/VCI_IN1#	VCI_IN1#	
B4	GPIO163/VCI_IN0#	VCI_IN0#	
A6	GPIO165/32KHZ_IN/CTOUT0		
F9	GPIO170/TFCLK/UART1_TX		
F8	GPIO171/TFDATA/UART1_RX/(JTAG_STRAP)		In-PU
F2	GPIO200/ADC00	ADC00	
G2	GPIO201/ADC01	ADC01	
H2	GPIO202/ADC02	ADC02	
G1	GPIO203/ADC03	ADC03	
H1	GPIO204/ADC04	ADC04	
K5	GPIO223/QSPI0_IO0		
K3	GPIO224/QSPI0_IO1		
D6	GPIO225/UART0_RTS#		
K4	GPIO227/QSPI0_IO2		
E8	JTAG_RST#	JTAG_RST#	
D2	RESETI#	RESETI#	
D5	VBAT	VBAT	
A1	VCI_OUT	VCI_OUT	O2ma-High
F1	VR_CAP	VR_CAP	
E3	VREF_ADC	VREF_ADC	
E4	VSS1	VSS1	

CEC1702-84	Signal	Default (if not GPIO)	Default State (if not In)
F7	VSS2	VSS2	
J1	VSS_ADC	VSS_ADC	
D4	VSS_ANALOG	VSS_ANALOG	
D1	VFLT_PLL	VFLT_PLL	
G6	VTR1	VTR1	
G5	VTR2	VTR2	
F4	VTR_ANALOG	VTR_ANALOG	
C1	VTR_PLL	VTR_PLL	
E1	VTR_REG	VTR_REG	
A4	XTAL1	XTAL1	
A2	XTAL2	XTAL2	

2.5 Pin List by Pin Number

2.5.1 POWER RAIL

The Power Rail column defines the power pin that provides I/O power for the signal pin.

2.5.2 PAD TYPES

The Pad Type column defines the type of pad associated with each signal. Some pins have signals with two different pad types sharing the pin; in this case, the pin is shown with the Pin Name but no pad type, followed by rows showing the pad type for each of the signals that share the pin. Pad Types are defined in the Section 37.0, "Electrical Specifications," on page 372.

- I/O Pad Types are defined in Section 37.2.4, "DC Electrical Characteristics for I/O Buffers," on page 374.
- The abbreviation “PWR” is used to denote power pins. The power supplies are defined in Section 37.2.1, "Power Supply Operational Characteristics," on page 372

2.5.3 GLITCH PROTECTION

Pins with glitch protection are glitch-free tristate pins and will not drive out while their associated power rail is rising. These glitch-free tristate pins require either an external pull-up or pull-down to set the state of the pin high or low.

Note:	If the pin needs to default low, a 1M ohm (max) external pull-down is required.

Pins without glitch protection may be susceptible to transitory changes as the power rail is rising.

Note:	The power rail must rise monotonically in order for glitch protection to operate.

BGPO GLITCH PROTECTION

All BGPO pins are glitch protected while VBAT power is applied.

The BGPO outputs are glitch protected on VBAT power as well as VTR power. As VBAT rises from ground, the BGPOx output are not driven until the VBAT power rail reaches approximately 1V. Once the VBAT power rail reaches approximately 1V, the BGPO outputs drive low.

Note:	It is recommended that a pull-down resistor be added to the BGPO pins.

2.5.4 OVER-VOLTAGE PROTECTION

For pins that have Over-voltage protection and the VTR*x* power rail that is supplying the pin is 3.3V, the pin can tolerate an input voltage of up to 5.5V without causing an error.

Note:	Pins with over-voltage protection may be pulled up externally to 5V supply. It is recommended to select strong pull-up resistor values (less than 10k ohms) that keep the pull-up voltage on the pin less than 3.8V and above 4.5V. If the voltage is 3.8V ≤ x ≤ 4.5V the pad current will be higher (65ua -nominal)

For pins with Over-voltage protection and the VTR*x* power rail that is supplying the pin is 1.8V, the pin can tolerate an input voltage of up to 3.6V without causing an error.

An input level that exceeds 105% of the power rail on a pin without Over-voltage protection may cause errors in the logic and may additionally damage internal circuitry.

2.5.5 UNDER-VOLTAGE PROTECTION

Pins that are identified as having Under-voltage PROTECTION may be configured so the will not sink excess current if powered by 3.3V and externally pulled up to 1.8V. The following configuration requirements must be met.

- If the pad is an output only pad type and it is configured as either open drain or the output is disabled.
- If the pin is a GPIO pin with a PIO pad type then is must be configured as open drain output with the input disabled. The input is disabled by setting the GPIO POWER_GATING bits to 11b.

2.5.6 BACKDRIVE PROTECTION

Assuming that the external voltage on the pin is within the parameters defined for the specific pad type, the backdrive protected pin will not sink excess current when it is at a lower potential than the external circuit. There are two cases where this occurs:

- The pad power is off and the external circuit is powered
- The pad power is on and the external circuitry is pulled to a higher potential than the pad power. This may occur on 3.3V powered pads that are 5V tolerant or on 1.8V powered pads that are 3.6V tolerant.

2.5.7 CEC1702 84 WFBGA

CEC1702-84	Pin Name	Power Rail	Pad Type	Glitch Prot	Over-voltage Prot	Under-voltage Prot	Backdrive Prot
A1	VCI_OUT	VBAT	O2ma			X	X
A2	XTAL2	VBAT	I_AN				
A3	GPIO003/I2C00_SDA/SPI0_CS#	VTR1	PIO	X		X	X
A4	XTAL1	VBAT	I_AN				
A5	GPIO162/VCI_IN1#	VBAT	PIO	X		X	X
A6	GPIO165/32KHZ_IN/CTOUT0	VTR1	PIO	X		X	X
A7	GPIO147/I2C08_SDA/JTAG_CLK	VTR1	PIO	X		X	X
A8	GPIO010/I2C03_SCL	VTR1	PIO	X	X	X	X
A9	GPIO154/I2C02_SDA	VTR1	PIO	X	X	X	X
A10	GPIO125/GPTP-OUT5/QSPI1_CLK/KSO12	VTR1	PIO	X		X	X
B1	BGPO0	VBAT	O	X		X	X
B2	GPIO004/I2C00_SCL/SPI0_MOSI	VTR1	PIO	X		X	X
B3	GPIO150/I2C08_SCL/JTAG_TMS	VTR1	PIO	X		X	X
B4	GPIO163/VCI_IN0#	VBAT	PIO	X		X	X

CEC1702-84	Pin Name	Power Rail	Pad Type	Glitch Prot	Over-voltage Prot	Under-voltage Prot	Backdrive Prot
B5	GPIO007/I2C03_SDA	VTR1	PIO	X	X	X	X
B6	GPIO146/I2C09_SCL/JTAG_TDO	VTR1	PIO	X		X	X
B7	GPIO155/I2C02_SCL	VTR1	PIO	X	X	X	X
B8	GPIO047/KSO03	VTR1	PIO	X		X	X
B9	GPIO157/LED1	VTR1	PIO	X	X	X	X
B10	GPIO122/QSPI1_IO1/KSO09	VTR1	PIO	X		X	X
C1	VTR_PLL		PWR				
C2	GPIO145/I2C09_SDA/JTAG_TDI	VTR1	PIO	X		X	X
C5	GPIO045/KSO01	VTR1	PIO	X		X	X
C6	GPIO127/UART0_CTS#	VTR1	PIO	X		X	X
C9	GPIO046/KSO02	VTR1	PIO	X		X	X
C10	GPIO124/QSPI1_CS#/KSO11	VTR1	PIO	X		X	X
D1	VFLT_PLL		PWR				
D2	RESETI#	VTR1	Om	X		X	X
D4	VSS_ANALOG		PWR				
D5	VBAT		PWR				
D6	GPIO225/UART0_RTS#	VTR1	PIO	X		X	X
D7	GPIO156/LED0	VTR1	PIO	X	X	X	X
D9	GPIO104/UART0_TX	VTR1	PIO	X		X	X
D10	GPIO121/QSPI1_IO0/KSO08	VTR1	PIO	X		X	X
E1	VTR_REG		PWR				
E2	GPIO051/FAN_TACH1/GTACH1	VTR1	PIO	X	X	X	X
E3	VREF_ADC		PWR			X	
E4	VSS1		PWR				
E7	GPIO135/UART1_CTS#	VTR1	PIO	X		X	X
E8	JTAG_RST#	VTR1	In	X		X	X
E9	GPIO105/UART0_RX	VTR1	PIO	X		X	X
E10	GPIO134/PWM10/UART1_RTS#	VTR1	PIO	X		X	X
F1	VR_CAP		PWR				
F2	GPIO200/ADC00	VTR1					
	GPIO200		PIO	X		X	
	ADC00		I_AN	X		X	
F3	GPIO050/FAN_TACH0/GTACH0	VTR1	PIO	X	X	X	X
F4	VTR_ANALOG		PWR				
F7	VSS2		PWR				
F8	GPIO171/TFDATA/UART1_RX/(JTAG_STRAP)	VTR1	PIO	X		X	X
F9	GPIO170/TFCLK/UART1_TX	VTR1	PIO	X		X	X

CEC1702-84	Pin Name	Power Rail	Pad Type	Glitch Prot	Over-voltage Prot	Under-voltage Prot	Backdrive Prot
F10	GPIO036/RC_ID2/SPI0_MISO	VTR1					
	GPIO036/SPI0_MISO		PIO	X		X	
	RC_ID2		I_AN	X		X	
G1	GPIO203/ADC03	VTR1					
	GPIO203		PIO	X		X	
	ADC03		I_AN	X		X	
G2	GPIO201/ADC01	VTR1					
	GPIO201		PIO	X		X	
	ADC01		I_AN	X		X	
G4	GPIO034/RC_ID1/SPI0_CLK	VTR1					
	GPIO034/SPI0_CLK		PIO	X		X	
	RC_ID1		I_AN	X		X	
G5	VTR2		PWR				
G6	VTR1		PWR				
G7	GPIO030/TIN3/KSI5	VTR2	PIO	X		X	X
G9	GPIO112/KSO05	VTR2	PIO	X		X	X
G10	GPIO113/KSO06	VTR2	PIO	X		X	X
H1	GPIO204/ADC04	VTR1					
	GPIO204		PIO	X		X	
	ADC04		I_AN	X		X	
H2	GPIO202/ADC02	VTR1					
	GPIO202		PIO	X		X	
	ADC02		I_AN	X		X	
H5	GPIO032/KSI7	VTR2	PIO	X		X	X
H6	GPIO140/ICT5	VTR2	PIO	X		X	X
H9	GPIO107/KSO04	VTR2	PIO	X		X	X
H10	GPIO120/KSO07	VTR2	PIO	X		X	X
J1	VSS_ADC		PWR				
J2	GPIO013/I2C07_SCL/TOUT2	VTR2	PIO	X		X	X
J3	GPIO020/KSI1	VTR2	PIO	X		X	X
J4	GPIO021/KSI2	VTR2	PIO	X		X	X
J5	GPIO002/PWM5	VTR2	PIO	X		X	X
J6	GPIO001/PWM4	VTR2	PIO	X		X	X
J7	GPIO017/GPTP-IN5/KSI0	VTR2	PIO	X		X	X
J8	GPIO031/KSI6	VTR2	PIO	X		X	X
J9	GPIO054/PWM1/GPWM1	VTR2	PIO	X	X	X	X
J10	GPIO027/TIN2/KSI4	VTR2	PIO	X		X	X
K1	GPIO012/I2C07_SDA/TOUT3	VTR2	PIO	X		X	X
K2	GPIO016/GPTP-IN7/QSPI0_IO3/ICT3	VTR2	PIO	X		X	X

CEC1702-84	Pin Name	Power Rail	Pad Type	Glitch Prot	Over-voltage Prot	Under-voltage Prot	Backdrive Prot
K3	GPIO224/QSPI0_IO1	VTR2	PIO	X		X	X
K4	GPIO227/QSPI0_IO2	VTR2	PIO	X		X	X
K5	GPIO223/QSPI0_IO0	VTR2	PIO	X		X	X
K6	GPIO056/PWM3/QSPI0_CLK	VTR2	PIO	X		X	X
K7	GPIO055/PWM2/QSPI0_CS#	VTR2	PIO	X		X	X
K8	GPIO040/KSO00	VTR2	PIO	X		X	X
K9	GPIO026/TIN1/KSI3	VTR2	PIO	X		X	X
K10	GPIO053/PWM0/GPWM0	VTR2	PIO	X	X	X	X

2.6 Signal Description by Signal

EMULATED POWER WELL

Power well emulation for GPIOs and for signals that are multiplexed with GPIO signals is controlled by the POWER_-GATING field in the GPIO Pin Control Register. Power well emulation for signals that are not multiplexed with GPIO signals is defined by the entries in this column.

GATED STATE

This column defines the internal value of an input signal when either its emulated power well is inactive or it is not selected by the GPIO alternate function MUX. A value of “No Gate” means that the internal signal always follows the pin even when the emulated power well is inactive.

Note: Gated state is only meaningful to the operation of input signals. A gated state on an output pin defines the internal behavior of the GPIO MUX and does not imply pin behavior.

Signal	Emulated Power Rail	Gated State	Notes
ADC00	VTR	Low	
ADC01	VTR	Low	
ADC02	VTR	Low	
ADC03	VTR	Low	
ADC04	VTR	Low	
BGND		Low	
BGPO0	VTR	Low	
CTOUT0	VTR	Low	
FAN_TACH0	VTR	Low	
FAN_TACH1	VTR	Low	
GPIO001	VTR	No Gate	
GPIO002	VTR	No Gate	
GPIO003	VTR	No Gate	
GPIO004	VTR	No Gate	
GPIO007	VTR	No Gate	

Signal	Emulated Power Rail	Gated State	Notes
GPIO010	VTR	No Gate	
GPIO012	VTR	No Gate	
GPIO013	VTR	No Gate	
GPIO016	VTR	No Gate	
GPIO017	VTR	No Gate	
GPIO020	VTR	No Gate	
GPIO021	VTR	No Gate	
GPIO026	VTR	No Gate	
GPIO027	VTR	No Gate	
GPIO030	VTR	No Gate	
GPIO031	VTR	No Gate	
GPIO032	VTR	No Gate	
GPIO034	VTR	No Gate	
GPIO036	VTR	No Gate	
GPIO040	VTR	No Gate	
GPIO045	VTR	No Gate	
GPIO046	VTR	No Gate	
GPIO047	VTR	No Gate	
GPIO050	VTR	No Gate	
GPIO051	VTR	No Gate	
GPIO053	VTR	No Gate	
GPIO054	VTR	No Gate	
GPIO055	VTR	No Gate	
GPIO056	VTR	No Gate	
GPIO104	VTR	No Gate	
GPIO106	VTR	No Gate	
GPIO107	VTR	No Gate	
GPIO112	VTR	No Gate	
GPIO113	VTR	No Gate	
GPIO120	VTR	No Gate	
GPIO121	VTR	No Gate	
GPIO122	VTR	No Gate	
GPIO124	VTR	No Gate	
GPIO125	VTR	No Gate	
GPIO127	VTR	No Gate	
GPIO134	VTR	No Gate	
GPIO135	VTR	No Gate	
GPIO140	VTR	No Gate	
GPIO145	VTR	No Gate	
GPIO146	VTR	No Gate	
GPIO147	VTR	No Gate	
GPIO150	VTR	No Gate	
GPIO154	VTR	No Gate	

Signal	Emulated Power Rail	Gated State	Notes
GPIO155	VTR	No Gate	
GPIO156	VTR	No Gate	
GPIO157	VTR	No Gate	
GPIO162	VTR	No Gate	
GPIO163	VTR	No Gate	
GPIO165	VTR	No Gate	
GPIO170	VTR	No Gate	
GPIO171	VTR	No Gate	
GPIO200	VTR	No Gate	
GPIO201	VTR	No Gate	
GPIO202	VTR	No Gate	
GPIO203	VTR	No Gate	
GPIO204	VTR	No Gate	
GPIO223	VTR	No Gate	
GPIO224	VTR	No Gate	
GPIO225	VTR	No Gate	
GPIO227	VTR	No Gate	
GPTP-IN5	VTR	No Gate	
GPTP-OUT5	VTR	No Gate	
GPWM0	VTR	Low	
GPWM1	VTR	Low	
GTACH0	VTR	Low	
GTACH1	VTR	Low	
I2C00_SCL	VTR	High	Note 2
I2C00_SDA	VTR	High	Note 2
I2C02_SCL	VTR	High	Note 2
I2C02_SDA	VTR	High	Note 2
I2C03_SCL	VTR	High	Note 2
I2C03_SDA	VTR	High	Note 2
I2C07_SCL	VTR	High	Note 2
I2C07_SDA	VTR	High	Note 2
I2C08_SCL	VTR	High	Note 2
I2C08_SDA	VTR	High	Note 2
I2C09_SCL	VTR	High	Note 2
I2C09_SDA	VTR	High	Note 2
ICT3	VTR	Low	
ICT5	VTR	Low	
JTAG_CLK	VTR	Low	
JTAG_RST#	VTR	High	Note 1
JTAG_TDI	VTR	Low	
JTAG_TDO	VTR	Low	
JTAG_TMS	VTR	Low	
KSI0	VTR	Low	

Signal	Emulated Power Rail	Gated State	Notes
KSI1	VTR	Low	
KSI2	VTR	Low	
KSI3	VTR	Low	
KSI4	VTR	Low	
KSI5	VTR	Low	
KSI6	VTR	Low	
KSI7	VTR	Low	
KSO00	VTR	Low	
KSO01	VTR	Low	
KSO02	VTR	Low	
KSO03	VTR	Low	
KSO04	VTR	Low	
KSO05	VTR	Low	
KSO06	VTR	Low	
KSO07	VTR	Low	
KSO08	VTR	Low	
KSO09	VTR	Low	
KSO11	VTR	Low	
KSO12	VTR	Low	
LED0	VTR	Low	
LED1	VTR	Low	
PWM0	VTR	Low	
PWM1	VTR	Low	
PWM2	VTR	Low	
PWM3	VTR	Low	
PWM4	VTR	Low	
PWM5	VTR	Low	
PWM10	VTR	Low	
QSPI0_CLK	VTR	Low	
QSPI0_CS#	VTR	High	
QSPI0_IO0	VTR	Low	
QSPI0_IO1	VTR	Low	
QSPI0_IO2	VTR	Low	
QSPI0_IO3	VTR	Low	
QSPI1_CLK	VTR	Low	
QSPI1_CS#	VTR	High	
QSPI1_IO0	VTR	Low	
QSPI1_IO1	VTR	Low	
RC_ID1	VTR	Low	
RC_ID2	VTR	Low	
RESETI#	VTR	High	
SPI0_CLK	VTR	Low	
SPI0_CS#	VTR	High	

Signal	Emulated Power Rail	Gated State	Notes
SPI0_MISO	VTR	Low	
SPI0_MOSI	VTR	Low	
TFCLK	VTR	Low	
TFDATA	VTR	Low	
TIN1	VTR	Low	
TIN2	VTR	Low	
TIN3	VTR	Low	
TOUT2	VTR	Low	
TOUT3	VTR	Low	
UART0_CTS#	VTR	Low	
UART0_RTS#	VTR	Low	
UART0_RX	VTR	Low	
UART0_TX	VTR	Low	
UART1_CTS#	VTR	Low	
UART1_RTS#	VTR	Low	
UART1_RX	VTR	Low	
UART1_TX	VTR	Low	
VBAT			
VCI_IN0#	VTR	No Gate	Note 3
VCI_IN1#	VTR	No Gate	Note 3
VCI_OUT	VTR	Low	
VFLT_PLL			
VR_CAP			Note 4
VREF_ADC			
VSS_ADC			
VSS_ANALOG			
VSS_REG			
VSS1			
VSS2			
VTR_ANALOG			
VTR_PLL			
VTR_REG			
VTR1			
VTR2			
XTAL1			
XTAL2			

2.7 Signal Description by interface

CEC1702 84 WFBGA	Interface		Notes
16-Bit Counter/Timer Interface			
K9	TIN1	16-Bit Counter/Timer Input 2	
J10	TIN2	16-Bit Counter/Timer Input 3	
G7	TIN3	16-Bit Counter/Timer Input 4	
J2	TOUT2	16-Bit Counter/Timer Output 3	
K1	TOUT3	16-Bit Counter/Timer Output 4	
Analog Data Acquisition Interface			
F2	ADC00	ADC Channel 0	
G2	ADC01	ADC Channel 1	
H2	ADC02	ADC Channel 2	
G1	ADC03	ADC Channel 3	
H1	ADC04	ADC Channel 4	
E3	VREF_ADC	ADC Voltage Reference	
Capture Timer interface			
A6	CTOUT0	Compare Timer Output 0	
K2	ICT3	Input Capture Timer Input 3	
H6	ICT5	Input Capture Timer Input 5	
Fan PWM and Tachometer			
F3	FAN_TACH0	Fan Tachometer Input 0/Input Capture Timer Input 0	
E2	FAN_TACH1	Fan Tachometer Input 1/Input Capture Timer Input 1	
K10	GPWM0	PWM Output from RPM-based Fan Speed Control Algorithm, PWM 0	
J9	GPWM1	PWM Output from RPM-based Fan Speed Control Algorithm, PWM 1	
F3	GTACH0	Tach Input to RPM-based Fan Speed Control Algorithm, Tach 0	
E2	GTACH1	Tach Input to RPM-based Fan Speed Control Algorithm, Tach 1	
K10	PWM0	Pulse Width Modulator Output 0	
J9	PWM1	Pulse Width Modulator Output 1	
K7	PWM2	Pulse Width Modulator Output 2	
K6	PWM3	Pulse Width Modulator Output 3	
J6	PWM4	Pulse Width Modulator Output 4	
J5	PWM5	Pulse Width Modulator Output 5	
E10	PWM10	Pulse Width Modulator Output 10	
General Purpose Input/Outputs			
J6	GPIO001	General Purpose Input/Output Port	
J5	GPIO002	General Purpose Input/Output Port	
A3	GPIO003	General Purpose Input/Output Port	

CEC1702 84 WFBGA	Interface		Notes
B2	GPIO004	General Purpose Input/Output Port	
B5	GPIO007	General Purpose Input/Output Port	
A8	GPIO010	General Purpose Input/Output Port	
K1	GPIO012	General Purpose Input/Output Port	
J2	GPIO013	General Purpose Input/Output Port	
K2	GPIO016	General Purpose Input/Output Port	
J7	GPIO017	General Purpose Input/Output Port	
J3	GPIO020	General Purpose Input/Output Port	
J4	GPIO021	General Purpose Input/Output Port	
K9	GPIO026	General Purpose Input/Output Port	
J10	GPIO027	General Purpose Input/Output Port	
G7	GPIO030	General Purpose Input/Output Port	
J8	GPIO031	General Purpose Input/Output Port	
H5	GPIO032	General Purpose Input/Output Port	
G4	GPIO034	General Purpose Input/Output Port	
F10	GPIO036	General Purpose Input/Output Port	
K8	GPIO040	General Purpose Input/Output Port	
C5	GPIO045	General Purpose Input/Output Port	
C9	GPIO046	General Purpose Input/Output Port	
B8	GPIO047	General Purpose Input/Output Port	
F3	GPIO050	General Purpose Input/Output Port	
E2	GPIO051	General Purpose Input/Output Port	
K10	GPIO053	General Purpose Input/Output Port	
J9	GPIO054	General Purpose Input/Output Port	
K7	GPIO055	General Purpose Input/Output Port	
K6	GPIO056	General Purpose Input/Output Port	
D9	GPIO104	General Purpose Input/Output Port	
E9	GPIO105	General Purpose Input/Output Port	
H9	GPIO107	General Purpose Input/Output Port	
G9	GPIO112	General Purpose Input/Output Port	
G10	GPIO113	General Purpose Input/Output Port	
H10	GPIO120	General Purpose Input/Output Port	
D10	GPIO121	General Purpose Input/Output Port	
B10	GPIO122	General Purpose Input/Output Port	
C10	GPIO124	General Purpose Input/Output Port	
A10	GPIO125	General Purpose Input/Output Port	
C6	GPIO127	General Purpose Input/Output Port	
E10	GPIO134	General Purpose Input/Output Port	

CEC1702 84 WFBGA	Interface		Notes
E7	GPIO135	General Purpose Input/Output Port	
H6	GPIO140	General Purpose Input/Output Port	
C2	GPIO145	General Purpose Input/Output Port	
B6	GPIO146	General Purpose Input/Output Port	
A7	GPIO147	General Purpose Input/Output Port	
B3	GPIO150	General Purpose Input/Output Port	
A9	GPIO154	General Purpose Input/Output Port	
B7	GPIO155	General Purpose Input/Output Port	
D7	GPIO156	General Purpose Input/Output Port	
B9	GPIO157	General Purpose Input/Output Port	
A5	GPIO162	General Purpose Input/Output Port	
B4	GPIO163	General Purpose Input/Output Port	
A6	GPIO165	General Purpose Input/Output Port	
F9	GPIO170	General Purpose Input/Output Port	
F8	GPIO171	General Purpose Input/Output Port	
F2	GPIO200	General Purpose Input/Output Port	
G2	GPIO201	General Purpose Input/Output Port	
H2	GPIO202	General Purpose Input/Output Port	
G1	GPIO203	General Purpose Input/Output Port	
H1	GPIO204	General Purpose Input/Output Port	
K5	GPIO223	General Purpose Input/Output Port	
K3	GPIO224	General Purpose Input/Output Port	
D6	GPIO225	General Purpose Input/Output Port	
K4	GPIO227	General Purpose Input/Output Port	
General Purpose Pass-Through Ports			
J7	GPTP-IN5	General Purpose Pass Through Port Input 5	
K2	GPTP-IN7	General Purpose Pass Through Port Input 7	
A10	GPTP-OUT5	General Purpose Pass Through Port Output 5	
I2C Interface			
B2	I2C00_SCL	I2C Controller Port 0 Clock	Note 2
A3	I2C00_SDA	I2C Controller Port 0 Data	Note 2
B7	I2C02_SCL	I2C Controller Port 2 Clock	Note 2
A9	I2C02_SDA	I2C Controller Port 2 Data	Note 2
A8	I2C03_SCL	I2C Controller Port 3 Clock	Note 2
B5	I2C03_SDA	I2C Controller Port 3 Data	Note 2
J2	I2C07_SCL	I2C Controller Port 7 Clock	Note 2
K1	I2C07_SDA	I2C Controller Port 7 Data	Note 2
B3	I2C08_SCL	I2C Controller Port 8 Clock	Note 2

CEC1702 84 WFBGA	Interface		Notes
A7	I2C08_SDA	I2C Controller Port 8 Data	Note 2
B6	I2C09_SCL	I2C Controller Port 9 Clock	Note 2
C2	I2C09_SDA	I2C Controller Port 9 Data	Note 2
JTAG and Debug			
A7	JTAG_CLK	JTAG Test Clock. Also ARM SWDCLK	
E8	JTAG_RST#	JTAG Test Reset (active low)	Note 1
C2	JTAG_TDI	JTAG Test Data In	
B6	JTAG_TDO	JTAG Test Data Out. Also ARM SWO	
B3	JTAG_TMS	JTAG Test Mode Select. Also ARM SWDIO	
F9	TFCLK	Trace FIFO debug port - clock	
F8	TFDATA	Trace FIFO debug port - data	
Keyboard Scan Interface			
J7	KSI0	Keyboard Scan Matrix Input 0	
J3	KSI1	Keyboard Scan Matrix Input 1	
J4	KSI2	Keyboard Scan Matrix Input 1	
K9	KSI3	Keyboard Scan Matrix Input 3	
J10	KSI4	Keyboard Scan Matrix Input 4	
G7	KSI5	Keyboard Scan Matrix Input 5	
J8	KSI6	Keyboard Scan Matrix Input 6	
H5	KSI7	Keyboard Scan Matrix Input 7	
K8	KSO00	Keyboard Scan Matrix Output 0	
C5	KSO01	Keyboard Scan Matrix Output 1	
C9	KSO02	Keyboard Scan Matrix Output 2	
B8	KSO03	Keyboard Scan Matrix Output 3	
H9	KSO04	Keyboard Scan Matrix Output 4	
G9	KSO05	Keyboard Scan Matrix Output 5	
G10	KSO06	Keyboard Scan Matrix Output 6	
H10	KSO07	Keyboard Scan Matrix Output 7	
D10	KSO08	Keyboard Scan Matrix Output 8	
B10	KSO09	Keyboard Scan Matrix Output 9	
C10	KSO11	Keyboard Scan Matrix Output 11	
A10	KSO12	Keyboard Scan Matrix Output 12	
Master Clock Interface			
A6	32KHZ_IN	32.768 KHz Digital Input	
A4	XTAL1	32.768 KHz Crystal Input	
A2	XTAL2	32.768 KHz Crystal Output (single-ended clock input)	
Miscellaneous Functions			

CEC1702 84 WFBGA	Interface		Notes
D7	LED0	LED Output 0	
B9	LED1	LED Output 1	
G4	RC_ID1	RC Identification Detection 1	
F10	RC_ID2	RC Identification Detection 2	
D2	RESETI#	System Reset Input	
Power			
D5	VBAT	VBAT supply	
D1	VFLT_PLL	Filtered power input for PLL	
F1	VR_CAP	Internal Voltage Regulator Output (Capacitor Required)	Note 4
J1	VSS_ADC	Analog ADC VTR associated ground	
E4	VSS1	VTR I/O Ground pin region 1	
F7	VSS2	VTR I/O Ground pin region 2	
D4	VSS_ANALOG	VTR associated ground for Internal Analog Logic	
F4	VTR_ANALOG	VTR Power Supply for Internal Analog Logic	
C1	VTR_PLL	VTR associated power used for PLL	
E1	VTR_REG	VTR Internal Voltage Regulator Power Supply	
G6	VTR1	VTR I/O Power, pin region 1	
G5	VTR2	VTR I/O Power, pin region 2	
Serial Ports			
C6	UART0_CTS#	UART 0, Clear to Send Input	
D6	UART0_RTS#	UART 0, Request to Send Output	
E9	UART0_RX	UART 0, Receive Data	
D9	UART0_TX	UART 0, Transmit Data	
E7	UART1_CTS#	UART 1, Clear to Send Input	
E10	UART1_RTS#	UART 1, Request to Send Output	
F8	UART1_RX	UART 1, Receive Data	
F9	UART1_TX	UART 1, Transmit Data	
SPI Controllers Interface			
A10	QSPI1_CLK	Quad SPI Controller Clock, Port 1	
C10	QSPI1_CS#	Quad SPI Controller Chip Select, Port 1	
D10	QSPI1_IO0	Quad SPI Controller Data 0, Port 1	
B10	QSPI1_IO1	Quad SPI Controller Data 1, Port 1	
K6	QSPI0_CLK	Quad SPI Controller Clock, Port 0	
K7	QSPI0_CS#	Quad SPI Controller Chip Select, Port 0	
K5	QSPI0_IO0	Quad SPI Controller Data 0, Port 0	
K3	QSPI0_IO1	Quad SPI Controller Data 1, Port 0	
K4	QSPI0_IO2	Quad SPI Controller Data 2, Port 0	
K2	QSPI0_IO3	Quad SPI Controller Data 3, Port 0	

CEC1702 84 WFBGA	Interface		Notes
G4	SPI0_CLK	GP-SPI SPI Clock	
A3	SPI0_CS#	GP-SPI Chip Select	
F10	SPI0_MISO	GP-SPI SPI Output	
B2	SPI0_MOSI	GP-SPI SPI Input	
VBAT-Powered Control Interface			
B1	BGPO0	VBAT driven GPO	
B4	VCI_IN0#	Input can cause wakeup or interrupt event, active low	Note 3
A5	VCI_IN1#	Input can cause wakeup or interrupt event, active low	Note 3
A1	VCI_OUT	Output from combinational logic and/or EC	

2.8 Strapping Options

GPIO171 is used for the TAP Controller select strap. If any of the JTAG TAP controllers are used, GPIO171 must only be configured as an output to a VTR powered external function. GPIO171 may only be configured as an input when the JTAG TAP controllers are not needed or when an external driver does not violate the Slave Select Timing.See Section 35.2.1, "TAP Controller Select Strap Option".

TABLE 2-1: STRAPS AND MEANING

Pin	Function	Definition
GPIO171/TFDATA/ UART1_RX/(JTAG_STRAP)	JTAG Boundary Scan	1=Use the JAG TAP Controller for Boundary Scan 0=The JTAG TAP Controller is used for debug

2.9 Pin Default State Through Power Transitions

The power state and power state transitions illustrated in the following tables are defined in Section 5.0, "Power, Clocks, and Resets". Pin behavior in this table assumes no specific programming to change the pin state. All GPIO default pins that have the same behavior are described in the table generically as GPIOXXX.

TABLE 2-2: PIN DEFAULT STATE THROUGH POWER TRANSITIONS

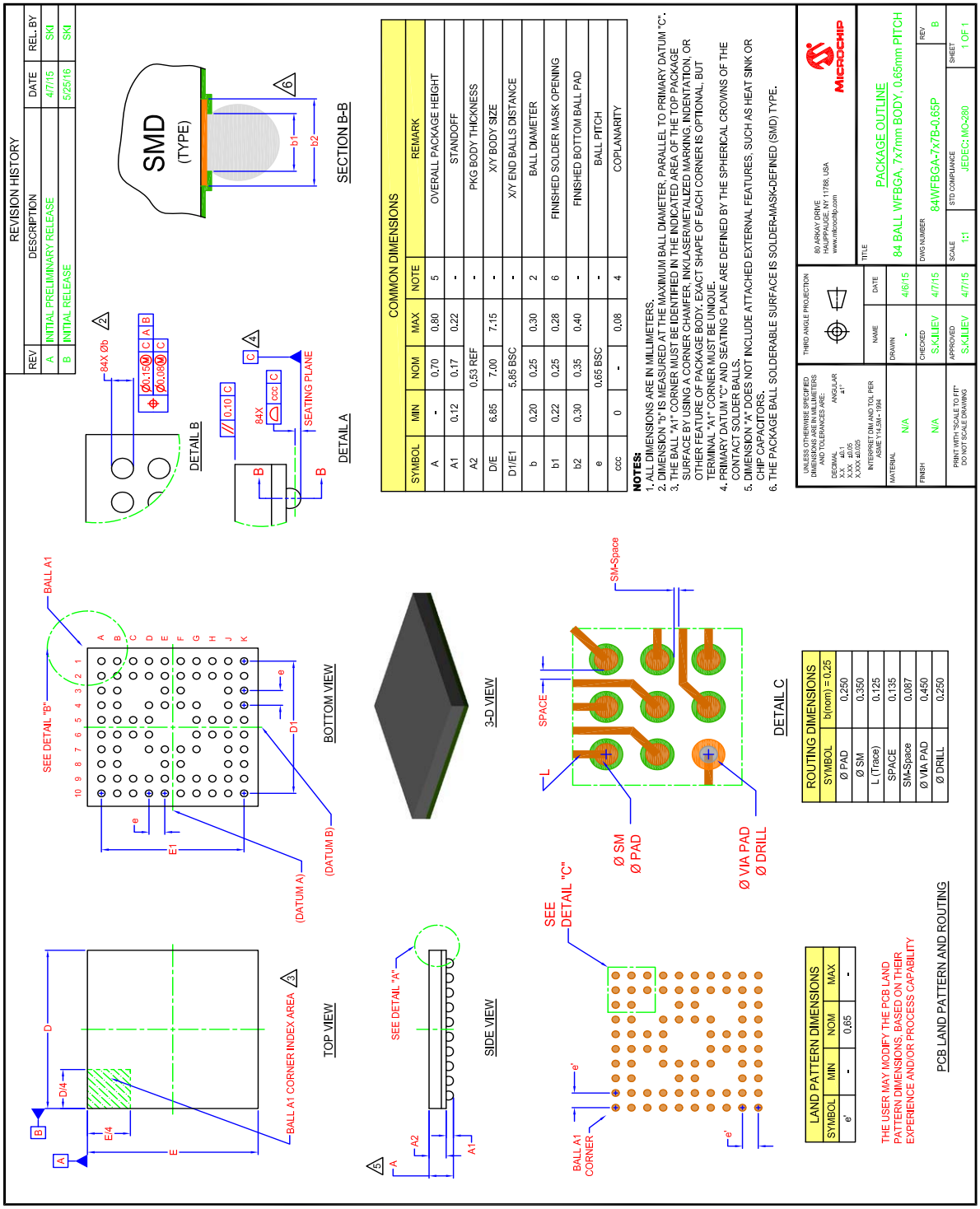
Signal	VBAT Applied	VBAT Stable	VTR Applied	RESET_SYS De-asserted	RESET_SYS Asserted	VTR Un-powered	VBAT Un-powered	Note
GPIOXXX	un-powered	un-powered	In	In	Z	glitch	un-powered	
BGPOx	Out=0	Out=0	Retain	Retain	Retain	Retain	un-powered	Note A
VCI_INx#	In	In	In	In	In	In	un-powered	
VCI_OUT	Out logic	Out logic	Out logic	Out logic	Out logic	Out logic	un-powered	Note B
XTAL1	Crystal In	Crystal In	Crystal In	Crystal In	Crystal In	Crystal In	Crystal In	
XTAL2	Crystal Out	Crystal Out	Crystal Out	Crystal Out	Crystal Out	Crystal Out	Crystal Out	

Legend	**Notes**	
(P) = I/O state is driven by protocol while power is applied.	Note A:	Pin is programmable by the EC and retains its value through a VTR power cycle.
Z = Tristate	Note B:	Pin is programmable by the EC and affected by other VBAT inputs pins.

2.10 Package

2.10.1 84 PIN WFBGA PACKAGE

Note: For the most current package drawings, see the Microchip Packaging Specification at http://www.microchip.com/packaging.



3.0 DEVICE INVENTORY

3.1 Conventions

Term	Definition
Block	Used to identify or describe the logic or IP Blocks implemented in the device.
Reserved	Reserved registers and bits defined in the following table are read only values that return 0 when read. Writes to these reserved registers have no effect.
TEST	Microchip Reserved locations which should not be modified from their default value. Changing a TEST register or a TEST field within a register may cause unwanted results.
b	The letter ‘b’ following a number denotes a binary number.
h	The letter ‘h’ following a number denotes a hexadecimal number.

Register access notation is in the form “Read / Write”. A Read term without a Write term means that the bit is read-only and writing has no effect. A Write term without a Read term means that the bit is write-only, and assumes that reading returns all zeros.

Register Field Type	Field Description
R	**Read:** A register or bit with this attribute can be read.
W	**Write:** A register or bit with this attribute can be written.
RS	**Read to Set:** This bit is set on read.
RC	**Read to Clear:** Content is cleared after the read. Writes have no effect.
WC	**Write One to Clear:** writing a one clears the value. Writing a zero has no effect.
WZC	**Write Zero to Clear:** writing a zero clears the value. Writing a one has no effect.
WS	**Write One to Set:** writing a one sets the value to 1. Writing a zero has no effect.
WZS	**Write Zero to Set:** writing a zero sets the value to 1. Writing a one has no effect.

3.2 GPIO Documentation Conventions

The GPIO registers and bits are allocated for the full GPIO complement. Therefore, even GPIOs that are not implemented in the package will appear in the GPIO register lists. Please refer to the pinout to determine which GPIOs are bonded out in the package. GPIOs that are not available in the package should not have their configuration register altered.

3.3 Block Overview and Base Addresses

Feature	Instance	Base Address
Watchdog Timer		4000_0000h
16-bit Basic Timer	0	4000_0C00h
16-bit Basic Timer	1	4000_0C20h
16-bit Basic Timer	2	4000_0C40h
16-bit Basic Timer	3	4000_0C60h
32-bit Basic Timer	0	4000_0C80h
32-bit Basic Timer	1	4000_0CA0h
16-bit Timer-Counter	0	4000_0D00h
16-bit Timer-Counter	1	4000_0D20h
16-bit Timer-Counter	2	4000_0D40h
16-bit Timer-Counter	3	4000_0D60h
Capture-Compare Timers		4000_1000h
RC-ID	1	4000_1480h
RC-ID	2	4000_1500h
DMA Controller		4000_2400h
I2C Controller	0	4000_4000h
I2C Controller	1	4000_4400h
I2C Controller	2	4000_4800h
I2C Controller	3	4000_4C00h
Quad Master SPI		4000_5400h
16-bit PWM	0	4000_5800h
16-bit PWM	1	4000_5810h
16-bit PWM	2	4000_5820h
16-bit PWM	3	4000_5830h
16-bit PWM	4	4000_5840h
16-bit PWM	5	4000_5850h
16-bit PWM	10	4000_58A0h
16-bit Tach	0	4000_6000h
16-bit Tach	1	4000_6010h
RTOS Timer		4000_7400h
ADC		4000_7C00h
Trace FIFO		4000_8C00h
GP-SPI	0	4000_9400h
Hibernation Timer	0	4000_9800h
Hibernation Timer	1	4000_9820h
Keyboard Matrix Scan		4000_9C00h
RPM to PWM Fan Controller	0	4000_A000h
RPM to PWM Fan Controller	1	4000_A080h
VBAT Register Bank		4000_A400h
VBAT Powered RAM		4000_A800h
Week Timer		4000_AC80h
VBAT-Powered Control Interface		4000_AE00h
Blinking-Breathing LED	0	4000_B800h
Blinking-Breathing LED	1	4000_B900h

Feature	Instance	Base Address
Interrupt Aggregator		4000_E000h
EC Subsystem Registers		4000_FC00h
JTAG		4008_0000h
Power, Clocks and Resets		4008_0100h
GPIOs		4008_1000h
eFuse		4008_2000h
UART	0	400F_2400h
UART	1	400F_2800h
Real Time Clock		400F_5000h
Global Configuration		400F_FF00h

3.4 Sleep Enable Register Bit Assignments

Block	Instance	Bit Position	Sleep Enable Register	Clock Required Register	Reset Enable Register
TEST		0	Sleep Enable 0	Clock Required 0	Reset Enable 0
eFuse		1	Sleep Enable 0	Clock Required 0	Reset Enable 0
Interrupt		0	Sleep Enable 1	Clock Required 1	Reset Enable 1
Tach	0	2	Sleep Enable 1	Clock Required 1	Reset Enable 1
PWM	0	4	Sleep Enable 1	Clock Required 1	Reset Enable 1
PMC		5	Sleep Enable 1	Clock Required 1	Reset Enable 1
DMA		6	Sleep Enable 1	Clock Required 1	Reset Enable 1
TFDP		7	Sleep Enable 1	Clock Required 1	Reset Enable 1
PROCESSOR		8	Sleep Enable 1	Clock Required 1	Reset Enable 1
WDT		9	Sleep Enable 1	Clock Required 1	Reset Enable 1
SMB	0	10	Sleep Enable 1	Clock Required 1	Reset Enable 1
Tach	1	11	Sleep Enable 1	Clock Required 1	Reset Enable 1
TEST		12	Sleep Enable 1	Clock Required 1	Reset Enable 1
PWM	1	20	Sleep Enable 1	Clock Required 1	Reset Enable 1
PWM	2	21	Sleep Enable 1	Clock Required 1	Reset Enable 1
PWM	3	22	Sleep Enable 1	Clock Required 1	Reset Enable 1
PWM	4	23	Sleep Enable 1	Clock Required 1	Reset Enable 1
PWM	5	24	Sleep Enable 1	Clock Required 1	Reset Enable 1
TEST		25	Sleep Enable 1	Clock Required 1	Reset Enable 1
TEST		26	Sleep Enable 1	Clock Required 1	Reset Enable 1
TEST		27	Sleep Enable 1	Clock Required 1	Reset Enable 1
EC Register Bank		29	Sleep Enable 1	Clock Required 1	Reset Enable 1
Basic Timer 16	0	30	Sleep Enable 1	Clock Required 1	Reset Enable 1
Basic Timer 16	1	31	Sleep Enable 1	Clock Required 1	Reset Enable 1
UART	0	1	Sleep Enable 2	Clock Required 2	Reset Enable 2
UART	1	2	Sleep Enable 2	Clock Required 2	Reset Enable 2
Global Configuration		12	Sleep Enable 2	Clock Required 2	Reset Enable 2
TEST	0	13	Sleep Enable 2	Clock Required 2	Reset Enable 2
TEST	1	14	Sleep Enable 2	Clock Required 2	Reset Enable 2
TEST		15	Sleep Enable 2	Clock Required 2	Reset Enable 2
RTC		18	Sleep Enable 2	Clock Required 2	Reset Enable 2
ADC		3	Sleep Enable 3	Clock Required 3	Reset Enable 3
GP-SPI	0	9	Sleep Enable 3	Clock Required 3	Reset Enable 3
Hibernation Timer	0	10	Sleep Enable 3	Clock Required 3	Reset Enable 3
Keyscan		11	Sleep Enable 3	Clock Required 3	Reset Enable 3
RPM2PWM	0	12	Sleep Enable 3	Clock Required 3	Reset Enable 3
I2C	1	13	Sleep Enable 3	Clock Required 3	Reset Enable 3
I2C	2	14	Sleep Enable 3	Clock Required 3	Reset Enable 3
I2C	3	15	Sleep Enable 3	Clock Required 3	Reset Enable 3
LED	0	16	Sleep Enable 3	Clock Required 3	Reset Enable 3
LED	1	17	Sleep Enable 3	Clock Required 3	Reset Enable 3
TEST		18	Sleep Enable 3	Clock Required 3	Reset Enable 3
TEST		20	Sleep Enable 3	Clock Required 3	Reset Enable 3

Block	Instance	Bit Position	Sleep Enable Register	Clock Required Register	Reset Enable Register
Basic Timer 16	2	21	Sleep Enable 3	Clock Required 3	Reset Enable 3
Basic Timer 16	3	22	Sleep Enable 3	Clock Required 3	Reset Enable 3
Basic Timer 32	0	23	Sleep Enable 3	Clock Required 3	Reset Enable 3
Basic Timer 32	1	24	Sleep Enable 3	Clock Required 3	Reset Enable 3
TEST		25	Sleep Enable 3	Clock Required 3	Reset Enable 3
PKE		26	Sleep Enable 3	Clock Required 3	Reset Enable 3
RNG		27	Sleep Enable 3	Clock Required 3	Reset Enable 3
AES-Hash		28	Sleep Enable 3	Clock Required 3	Reset Enable 3
Hibernation Timer	1	29	Sleep Enable 3	Clock Required 3	Reset Enable 3
Capture Compare Timer		30	Sleep Enable 3	Clock Required 3	Reset Enable 3
TEST		31	Sleep Enable 3	Clock Required 3	Reset Enable 3
PWM	10	0	Sleep Enable 4	Clock Required 4	Reset Enable 4
TEST		1	Sleep Enable 4	Clock Required 4	Reset Enable 4
16-bit Counter/Timer	0	2	Sleep Enable 4	Clock Required 4	Reset Enable 4
16-bit Counter/Timer	1	3	Sleep Enable 4	Clock Required 4	Reset Enable 4
16-bit Counter/Timer	2	4	Sleep Enable 4	Clock Required 4	Reset Enable 4
16-bit Counter/Timer	3	5	Sleep Enable 4	Clock Required 4	Reset Enable 4
RTOS Timer		6	Sleep Enable 4	Clock Required 4	Reset Enable 4
RPM2PWM	1	7	Sleep Enable 4	Clock Required 4	Reset Enable 4
Quad SPI Master		8	Sleep Enable 4	Clock Required 4	Reset Enable 4
RC_ID	1	11	Sleep Enable 4	Clock Required 4	Reset Enable 4
RC_ID	2	12	Sleep Enable 4	Clock Required 4	Reset Enable 4

3.5 Interrupt Aggregator Bit Assignments

Note: Interrupt Aggregator bits associated with GPIOs not present in the pinout for a particular device are Reserved.

Agg IRQ	Agg Bit	HWB Instance Name	Interrupt Event	Wake Event	Source Description	Agg NVIC	Direct NVIC
GIRQ8	0	GPIO140	GPIO Event	Yes	GPIO Interrupt Event	0	N/A
	1	GPIO141	GPIO Event	Yes	GPIO Interrupt Event		
	2	GPIO142	GPIO Event	Yes	GPIO Interrupt Event		
	3	GPIO143	GPIO Event	Yes	GPIO Interrupt Event		
	4	GPIO144	GPIO Event	Yes	GPIO Interrupt Event		
	5	GPIO145	GPIO Event	Yes	GPIO Interrupt Event		
	6	GPIO146	GPIO Event	Yes	GPIO Interrupt Event		
	7	GPIO147	GPIO Event	Yes	GPIO Interrupt Event		
	8	GPIO150	GPIO Event	Yes	GPIO Interrupt Event		
	9	GPIO151	GPIO Event	Yes	GPIO Interrupt Event		
	10	GPIO152	GPIO Event	Yes	GPIO Interrupt Event		
	11	GPIO153	GPIO Event	Yes	GPIO Interrupt Event		
	12	GPIO154	GPIO Event	Yes	GPIO Interrupt Event		
	13	GPIO155	GPIO Event	Yes	GPIO Interrupt Event		
	14	GPIO156	GPIO Event	Yes	GPIO Interrupt Event		
	15	GPIO157	GPIO Event	Yes	GPIO Interrupt Event		
	16	GPIO160	GPIO Event	Yes	GPIO Interrupt Event		
	17	GPIO161	GPIO Event	Yes	GPIO Interrupt Event		
	18	GPIO162	GPIO Event	Yes	GPIO Interrupt Event		
	19	GPIO163	GPIO Event	Yes	GPIO Interrupt Event		
	20	GPIO164	GPIO Event	Yes	GPIO Interrupt Event		
	21	GPIO165	GPIO Event	Yes	GPIO Interrupt Event		
	22	GPIO166	GPIO Event	Yes	GPIO Interrupt Event		
	23	GPIO167	GPIO Event	Yes	GPIO Interrupt Event		
	24	GPIO170	GPIO Event	Yes	GPIO Interrupt Event		
	25	GPIO171	GPIO Event	Yes	GPIO Interrupt Event		
	26	GPIO172	GPIO Event	Yes	GPIO Interrupt Event		
	27	GPIO173	GPIO Event	Yes	GPIO Interrupt Event		
	28	GPIO174	GPIO Event	Yes	GPIO Interrupt Event		
	29	GPIO175	GPIO Event	Yes	GPIO Interrupt Event		
	30	Reserved					
	31	Reserved					

Agg IRQ	Agg Bit	HWB Instance Name	Interrupt Event	Wake Event	Source Description	Agg NVIC	Direct NVIC
GIRQ9	0	GPIO100	GPIO Event	Yes	GPIO Interrupt Event	1	N/A
	1	GPIO101	GPIO Event	Yes	GPIO Interrupt Event		
	2	GPIO102	GPIO Event	Yes	GPIO Interrupt Event		
	3	GPIO103	GPIO Event	Yes	GPIO Interrupt Event		
	4	GPIO104	GPIO Event	Yes	GPIO Interrupt Event		
	5	GPIO105	GPIO Event	Yes	GPIO Interrupt Event		
	6	GPIO106	GPIO Event	Yes	GPIO Interrupt Event		
	7	GPIO107	GPIO Event	Yes	GPIO Interrupt Event		
	8	GPIO110	GPIO Event	Yes	GPIO Interrupt Event		
	9	GPIO111	GPIO Event	Yes	GPIO Interrupt Event		
	10	GPIO112	GPIO Event	Yes	GPIO Interrupt Event		
	11	GPIO113	GPIO Event	Yes	GPIO Interrupt Event		
	12	GPIO114	GPIO Event	Yes	GPIO Interrupt Event		
	13	GPIO115	GPIO Event	Yes	GPIO Interrupt Event		
	14	GPIO116	GPIO Event	Yes	GPIO Interrupt Event		
	15	GPIO117	GPIO Event	Yes	GPIO Interrupt Event		
	16	GPIO120	GPIO Event	Yes	GPIO Interrupt Event		
	17	GPIO121	GPIO Event	Yes	GPIO Interrupt Event		
	18	GPIO122	GPIO Event	Yes	GPIO Interrupt Event		
	19	GPIO123	GPIO Event	Yes	GPIO Interrupt Event		
	20	GPIO124	GPIO Event	Yes	GPIO Interrupt Event		
	21	GPIO125	GPIO Event	Yes	GPIO Interrupt Event		
	22	GPIO126	GPIO Event	Yes	GPIO Interrupt Event		
	23	GPIO127	GPIO Event	Yes	GPIO Interrupt Event		
	24	GPIO130	GPIO Event	Yes	GPIO Interrupt Event		
	25	GPIO131	GPIO Event	Yes	GPIO Interrupt Event		
	26	GPIO132	GPIO Event	Yes	GPIO Interrupt Event		
	27	GPIO133	GPIO Event	Yes	GPIO Interrupt Event		
	28	GPIO134	GPIO Event	Yes	GPIO Interrupt Event		
	29	GPIO135	GPIO Event	Yes	GPIO Interrupt Event		
	30	Reserved					
	31	Reserved					

Agg IRQ	Agg Bit	HWB Instance Name	Interrupt Event	Wake Event	Source Description	Agg NVIC	Direct NVIC
GIRQ10	0	GPIO040	GPIO Event	Yes	GPIO Interrupt Event	2	N/A
	1	GPIO041	GPIO Event	Yes	GPIO Interrupt Event		
	2	GPIO042	GPIO Event	Yes	GPIO Interrupt Event		
	3	GPIO043	GPIO Event	Yes	GPIO Interrupt Event		
	4	GPIO044	GPIO Event	Yes	GPIO Interrupt Event		
	5	GPIO045	GPIO Event	Yes	GPIO Interrupt Event		
	6	GPIO046	GPIO Event	Yes	GPIO Interrupt Event		
	7	GPIO047	GPIO Event	Yes	GPIO Interrupt Event		
	8	GPIO050	GPIO Event	Yes	GPIO Interrupt Event		
	9	GPIO051	GPIO Event	Yes	GPIO Interrupt Event		
	10	GPIO052	GPIO Event	Yes	GPIO Interrupt Event		
	11	GPIO053	GPIO Event	Yes	GPIO Interrupt Event		
	12	GPIO054	GPIO Event	Yes	GPIO Interrupt Event		
	13	GPIO055	GPIO Event	Yes	GPIO Interrupt Event		
	14	GPIO056	GPIO Event	Yes	GPIO Interrupt Event		
	15	GPIO057	GPIO Event	Yes	GPIO Interrupt Event		
	16	GPIO060	GPIO Event	Yes	GPIO Interrupt Event		
	17	GPIO061	GPIO Event	Yes	GPIO Interrupt Event		
	18	GPIO062	GPIO Event	Yes	GPIO Interrupt Event		
	19	GPIO063	GPIO Event	Yes	GPIO Interrupt Event		
	20	GPIO064	GPIO Event	Yes	GPIO Interrupt Event		
	21	GPIO065	GPIO Event	Yes	GPIO Interrupt Event		
	22	GPIO066	GPIO Event	Yes	GPIO Interrupt Event		
	23	GPIO067	GPIO Event	Yes	GPIO Interrupt Event		
	24	GPIO070	GPIO Event	Yes	GPIO Interrupt Event		
	25	GPIO071	GPIO Event	Yes	GPIO Interrupt Event		
	26	GPIO072	GPIO Event	Yes	GPIO Interrupt Event		
	27	GPIO073	GPIO Event	Yes	GPIO Interrupt Event		
	28	GPIO074	GPIO Event	Yes	GPIO Interrupt Event		
	29	GPIO075	GPIO Event	Yes	GPIO Interrupt Event		
	30	GPIO076	GPIO Event	Yes	GPIO Interrupt Event		
	31	Reserved					

Agg IRQ	Agg Bit	HWB Instance Name	Interrupt Event	Wake Event	Source Description	Agg NVIC	Direct NVIC
GIRQ11	0	GPIO000	GPIO Event	Yes	GPIO Interrupt Event	3	N/A
	1	GPIO001	GPIO Event	Yes	GPIO Interrupt Event		
	2	GPIO002	GPIO Event	Yes	GPIO Interrupt Event		
	3	GPIO003	GPIO Event	Yes	GPIO Interrupt Event		
	4	GPIO004	GPIO Event	Yes	GPIO Interrupt Event		
	5	GPIO005	GPIO Event	Yes	GPIO Interrupt Event		
	6	GPIO006	GPIO Event	Yes	GPIO Interrupt Event		
	7	GPIO007	GPIO Event	Yes	GPIO Interrupt Event		
	8	GPIO010	GPIO Event	Yes	GPIO Interrupt Event		
	9	GPIO011	GPIO Event	Yes	GPIO Interrupt Event		
	10	GPIO012	GPIO Event	Yes	GPIO Interrupt Event		
	11	GPIO013	GPIO Event	Yes	GPIO Interrupt Event		
	12	GPIO014	GPIO Event	Yes	GPIO Interrupt Event		
	13	GPIO015	GPIO Event	Yes	GPIO Interrupt Event		
	14	GPIO016	GPIO Event	Yes	GPIO Interrupt Event		
	15	GPIO017	GPIO Event	Yes	GPIO Interrupt Event		
	16	GPIO020	GPIO Event	Yes	GPIO Interrupt Event		
	17	GPIO021	GPIO Event	Yes	GPIO Interrupt Event		
	18	GPIO022	GPIO Event	Yes	GPIO Interrupt Event		
	19	GPIO023	GPIO Event	Yes	GPIO Interrupt Event		
	20	GPIO024	GPIO Event	Yes	GPIO Interrupt Event		
	21	GPIO025	GPIO Event	Yes	GPIO Interrupt Event		
	22	GPIO026	GPIO Event	Yes	GPIO Interrupt Event		
	23	GPIO027	GPIO Event	Yes	GPIO Interrupt Event		
	24	GPIO030	GPIO Event	Yes	GPIO Interrupt Event		
	25	GPIO031	GPIO Event	Yes	GPIO Interrupt Event		
	26	GPIO032	GPIO Event	Yes	GPIO Interrupt Event		
	27	GPIO033	GPIO Event	Yes	GPIO Interrupt Event		
	28	GPIO034	GPIO Event	Yes	GPIO Interrupt Event		
	29	GPIO035	GPIO Event	Yes	GPIO Interrupt Event		
	30	GPIO036	GPIO Event	Yes	GPIO Interrupt Event		
	31	Reserved					

Agg IRQ	Agg Bit	HWB Instance Name	Interrupt Event	Wake Event	Source Description	Agg NVIC	Direct NVIC
GIRQ12	0	GPIO200	GPIO Event	Yes	GPIO Interrupt Event	4	N/A
	1	GPIO201	GPIO Event	Yes	GPIO Interrupt Event		
	2	GPIO202	GPIO Event	Yes	GPIO Interrupt Event		
	3	GPIO203	GPIO Event	Yes	GPIO Interrupt Event		
	4	GPIO204	GPIO Event	Yes	GPIO Interrupt Event		
	5	GPIO205	GPIO Event	Yes	GPIO Interrupt Event		
	6	GPIO206	GPIO Event	Yes	GPIO Interrupt Event		
	7	GPIO207	GPIO Event	Yes	GPIO Interrupt Event		
	8	GPIO210	GPIO Event	Yes	GPIO Interrupt Event		
	9	GPIO211	GPIO Event	Yes	GPIO Interrupt Event		
	10	GPIO212	GPIO Event	Yes	GPIO Interrupt Event		
	11	GPIO213	GPIO Event	Yes	GPIO Interrupt Event		
	12	GPIO214	GPIO Event	Yes	GPIO Interrupt Event		
	13	GPIO215	GPIO Event	Yes	GPIO Interrupt Event		
	14	GPIO216	GPIO Event	Yes	GPIO Interrupt Event		
	15	GPIO217	GPIO Event	Yes	GPIO Interrupt Event		
	16	Reserved					
	17	GPIO221	GPIO Event	Yes	GPIO Interrupt Event		
	18	GPIO222	GPIO Event	Yes	GPIO Interrupt Event		
	19	GPIO223	GPIO Event	Yes	GPIO Interrupt Event		
	20	GPIO224	GPIO Event	Yes	GPIO Interrupt Event		
	21	GPIO225	GPIO Event	Yes	GPIO Interrupt Event		
	22	GPIO226	GPIO Event	Yes	GPIO Interrupt Event		
	23	GPIO227	GPIO Event	Yes	GPIO Interrupt Event		
	24	GPIO230	GPIO Event	Yes	GPIO Interrupt Event		
	25	GPIO231	GPIO Event	Yes	GPIO Interrupt Event		
	26	Reserved					
	27	GPIO233	GPIO Event	Yes	GPIO Interrupt Event		
	28	GPIO234	GPIO Event	Yes	GPIO Interrupt Event		
	29	GPIO235	GPIO Event	Yes	GPIO Interrupt Event		
	30	Reserved					
	31	Reserved					
GIRQ13	0	I2C Controller 0	I2C	No	I2C Controller 0 Interrupt Event	5	20
	1	I2C Controller 1	I2C	No	I2C Controller 1 Interrupt Event		21
	2	I2C Controller 2	I2C	No	I2C Controller 2 Interrupt Event		22
	3	I2C Controller 3	I2C	No	I2C Controller 3 Interrupt Event		23
	4-31	Reserved					

Agg IRQ	Agg Bit	HWB Instance Name	Interrupt Event	Wake Event	Source Description	Agg NVIC	Direct NVIC
GIRQ14	0	DMA Controller	DMA0	No	DMA Controller - Channel 0 Interrupt Event	6	24
	1	DMA Controller	DMA1	No	DMA Controller - Channel 1 Interrupt Event		25
	2	DMA Controller	DMA2	No	DMA Controller - Channel 2 Interrupt Event		26
	3	DMA Controller	DMA3	No	DMA Controller - Channel 3 Interrupt Event		27
	4	DMA Controller	DMA4	No	DMA Controller - Channel 4 Interrupt Event		28
	5	DMA Controller	DMA5	No	DMA Controller - Channel 5 Interrupt Event		29
	6	DMA Controller	DMA6	No	DMA Controller - Channel 6 Interrupt Event		30
	7	DMA Controller	DMA7	No	DMA Controller - Channel 7 Interrupt Event		31
	8	DMA Controller	DMA8	No	DMA Controller - Channel 8 Interrupt Event		32
	9	DMA Controller	DMA9	No	DMA Controller - Channel 9 Interrupt Event		33
	10	DMA Controller	DMA10	No	DMA Controller - Channel 10 Interrupt Event		34
	11	DMA Controller	DMA11	No	DMA Controller - Channel 11 Interrupt Event		35
	12	DMA Controller	DMA12	No	DMA Controller - Channel 12 Interrupt Event		36
	13	DMA Controller	DMA13	No	DMA Controller - Channel 13 Interrupt Event		37
	14-31	Reserved					
GIRQ15	0	UART 0	UART	No	UART Interrupt Event	7	40
	1	UART 1	UART	No	UART Interrupt Event		41
	2-23	Reserved					
	24	Test	Test	-	-		64
	25-31	Reserved					
GIRQ16	0	Public Key Engine	PKE ERROR	No	PKE core error detected	8	65
	1	Public Key Engine	PKE END	No	PKE completed processing		66
	2	Random Number Generator	RNG	No	RNG completed processing		67
	3	AES	AES	No	Interrupt from AES block		68
	4	Hash	HASH	No	Interrupt from SHA block		69
	5-31	Reserved					

Agg IRQ	Agg Bit	HWB Instance Name	Interrupt Event	Wake Event	Source Description	Agg NVIC	Direct NVIC
GIRQ17	0	Reserved				9	70
	1	TACH 0	TACH	No	Tachometer 0 Interrupt Event		71
	2	TACH 1	TACH	No	Tachometer 1 Interrupt Event		72
	3	Reserved					73
	4	RPM2PWM 0	FAN_FAIL	No	Failure to achieve target RPM		74
	5	RPM2PWM 0	FAN_STALL	No	Fan stall condition		75
	6	RPM2PWM 1	FAN_FAIL	No	Failure to achieve target RPM		76
	7	RPM2PWM 1	FAN_STALL	No	Fan stall condition		77
	8	ADC Controller	ADC_Single_Int	No	ADC Controller - Single-Sample ADC Conversion Event		78
	9	ADC Controller	ADC_Repeat_Int	No	ADC Controller - Repeat-Sample ADC Conversion Event		79
	10	Reserved					80
	11	RC-ID 1	RCID	No	0-1 transition of RC-ID done flag		81
	12	RC-ID 2	RCID	No	0-1 transition of RC-ID done flag		82
	13	Breathing LED 0	PWM_WDT	No	Blinking LED 0 Watchdog Event		83
	14	Breathing LED 1	PWM_WDT	No	Blinking LED 1 Watchdog Event		84
	15-24	Reserved					
	25	RTOS Timer	SWI_0	No	Soft Interrupt request 0		
	26	RTOS Timer	SWI_1	No	Soft Interrupt request 1		
	27	RTOS Timer	SWI_2	No	Soft Interrupt request 2		
	28	RTOS Timer	SWI_3	No	Soft Interrupt request 3		
	29-31	Reserved					
GIRQ18	0	Reserved				10	90
	1	Quad Master SPI Controller	QMSPI_INT	No	Master SPI Controller Requires Servicing		91
	2	GP-SPI 0	TXBE_STS	No	SPI TX buffer empty		92
	3	GP-SPI 0	RXBF_STS	No	SPI RX buffer full		93
	4-31	Reserved					
GIRQ19	0-31	Reserved				11	103
GIRQ20	0-8	Test	Test	-	-	12	N/A
	9-31	Reserved					

Agg IRQ	Agg Bit	HWB Instance Name	Interrupt Event	Wake Event	Source Description	Agg NVIC	Direct NVIC
GIRQ21	0	RTOS Timer	RTOS_TIMER	Yes	32-bit RTOS Timer Event	13	111
	1	Hibernation Timer 0	HTIMER	Yes	Hibernation Timer Event		112
	2	Hibernation Timer 1	HTIMER	Yes	Hibernation Timer Event		113
	3	Week Alarm	WEEK_ALARM_INT	Yes	Week Alarm Interrupt.		114
	4	Week Alarm	SUB_WEEK_ ALARM_INT	Yes	Sub-Week Alarm Interrupt		115
	5	Week Alarm	ONE_SECOND	Yes	Week Alarm - One Second Interrupt		116
	6	Week Alarm	SUB_SECOND	Yes	Week Alarm - Sub-second Interrupt		117
	7	Reserved					-
	8	RTC	RTC	Yes	Real Time Clock Interrupt		119
	9	RTC	RTC ALARM	Yes	Real Time Clock Alarm Interrupt		120
	10	Reserved					-
	11	VBAT-Powered Control Interface	VCI_IN0	Yes	VCI_IN0 Active-low Input Pin Interrupt		122
	12	VBAT-Powered Control Interface	VCI_IN1	Yes	VCI_IN1 Active-low Input Pin Interrupt		123
	13-24	Reserved					
	25	Keyscan	KSC_INT	Yes	Keyboard Scan Interface Runtime Interrupt		135
	26-31	Reserved					
GIRQ22	0	Reserved				N/A	N/A
	1	I2C Controller 0	I2C_WAKE_ONLY	Yes	Wake-Only Event (No Interrupt Generated) - I2C.0 START Detected		
	2	I2C Controller 1	I2C_WAKE_ONLY	Yes	Wake-Only Event (No Interrupt Generated) - I2C.1 START Detected		
	3	I2C Controller 2	I2C_WAKE_ONLY	Yes	Wake-Only Event (No Interrupt Generated) - I2C.2 START Detected		
	4	I2C Controller 3	I2C_WAKE_ONLY	Yes	Wake-Only Event (No Interrupt Generated) - I2C.3 START Detected		
	5-31	Reserved					

Agg IRQ	Agg Bit	HWB Instance Name	Interrupt Event	Wake Event	Source Description	Agg NVIC	Direct NVIC
GIRQ23	0	16-Bit Basic Timer 0	Timer_Event	No	Basic Timer Event	14	136
	1	16-Bit Basic Timer 1	Timer_Event	No	Basic Timer Event		137
	2	16-Bit Basic Timer 2	Timer_Event	No	Basic Timer Event		138
	3	16-Bit Basic Timer 3	Timer_Event	No	Basic Timer Event		139
	4	32-Bit Basic Timer 0	Timer_Event	No	Basic Timer Event		140
	5	32-Bit Basic Timer 1	Timer_Event	No	Basic Timer Event		141
	6	Counter/Timer 0	Timer_Event	No	16-bit Timer/Counter Event		142
	7	Counter/Timer 1	Timer_Event	No	16-bit Timer/Counter Event		143
	8	Counter/Timer 2	Timer_Event	No	16-bit Timer/Counter Event		144
	9	Counter/Timer 3	Timer_Event	No	16-bit Timer/Counter Event		145
	10	Capture Compare Timer	CAPTURE TIMER	No	CCT Counter Event		146
	11-13	Reserved					149
	14	Capture Compare Timer	CAPTURE 3	No	CCT Capture 3 Event		150
	15	Reserved					151
	16	Capture Compare Timer	CAPTURE 5	No	CCT Capture 5 Event		152
	17	Capture Compare Timer	COMPARE 0	No	CCT Compare 0 Event		153
	18-31	Reserved					

3.6 GPIO Register Assignments

TABLE 3-1: GPIO MULTIPLEXING

GPIO	MUX_CONTROL= 00b	MUX_CONTROL= 01b	MUX_CONTROL= 10b	MUX_CONTROL = 11b
GPIO001	GPIO001	PWM4		
GPIO002	GPIO002	PWM5	SPI0_CS#	
GPIO003	GPIO003	I2C00_SDA	SPI0_CS#	
GPIO004	GPIO004	I2C00_SCL	SPI0_MOSI	
GPIO007	GPIO007	I2C03_SDA		
GPIO010	GPIO010	I2C03_SCL		
GPIO012	GPIO012			
GPIO013	GPIO013			
GPIO016	GPIO016	GPTP-IN7	QSPI0_IO3	ICT3
GPIO017	GPIO017			KSI0
GPIO020	GPIO020			KSI1
GPIO021	GPIO021			KSI2
GPIO026	GPIO026	TIN1		KSI3
GPIO027	GPIO027	TIN2		KSI4
GPIO030	GPIO030	TIN3		KSI5
GPIO031	GPIO031			KSI6
GPIO032	GPIO032			KSI7
GPIO034	GPIO034	RC_ID1	SPI0_CLK	
GPIO036	GPIO036	RC_ID2	SPI0_MISO	
GPIO040	GPIO040			KSO00
GPIO045	GPIO045			KSO01
GPIO046	GPIO046			KSO02
GPIO047	GPIO047			KSO03
GPIO050	GPIO050	FAN_TACH0	GTACH0	
GPIO051	GPIO051	FAN_TACH1	GTACH1	
GPIO053	GPIO053	PWM0	GPWM0	
GPIO054	GPIO054	PWM1	GPWM1	
GPIO055	GPIO055	PWM2	QSPI0_CS#	
GPIO056	GPIO056	PWM3	QSPI0_CLK	
GPIO104	GPIO104	UART0_TX		
GPIO105	GPIO105	UART0_RX		
GPIO107	GPIO107			KSO04
GPIO112	GPIO112			KSO05
GPIO113	GPIO113			KSO06
GPIO120	GPIO120			KSO07
GPIO121	GPIO121		QSPI1_IO0	KSO08
GPIO122	GPIO122		QSPI1_IO1	KSO09
GPIO124	GPIO124		QSPI1_CS#	KSO11
GPIO125	GPIO125		QSPI1_CLK	KSO12
GPIO127	GPIO127		UART0_CTS#	

TABLE 3-1: GPIO MULTIPLEXING (CONTINUED)

GPIO	MUX_CONTROL= 00b	MUX_CONTROL= 01b	MUX_CONTROL= 10b	MUX_CONTROL = 11b
GPIO134	GPIO134	PWM10	UART1_RTS#	
GPIO135	GPIO135		UART1_CTS#	
GPIO140	GPIO140	I2C06_SCL	ICT5	
GPIO145	GPIO145	I2C09_SDA		
GPIO146	GPIO146	I2C09_SCL		
GPIO147	GPIO147	I2C08_SDA		
GPIO150	GPIO150	I2C08_SCL		
GPIO154	GPIO154	I2C02_SDA		
GPIO155	GPIO155	I2C02_SCL		
GPIO156	GPIO156	LED0		
GPIO162	GPIO162	VCI_IN1#		
GPIO163	GPIO163	VCI_IN0#		
GPIO165	GPIO165			
GPIO170	GPIO170	TFCLK	UART1_TX	
GPIO171	GPIO171	TFDATA	UART1_RX	
GPIO200	GPIO200	ADC0		
GPIO201	GPIO201	ADC1		
GPIO202	GPIO202	ADC2		
GPIO203	GPIO203	ADC3		
GPIO204	GPIO204	ADC4		
GPIO223	GPIO223		QSPI0_IO0	
GPIO224	GPIO224	GPTP-IN4	QSPI0_IO1	
GPIO225	GPIO225	UART0_RTS#		
GPIO227	GPIO227		QSPI0_IO2	

TABLE 3-2: GPIO PIN CONTROL REGISTERS DEFAULTS

GPIO Name	Pin Control Register Offset (Hex)	Pin Control Register Default Value (Hex)	Pin Control Register Default Function	Pin Control 2 Register Offset (Hex)	Pin Control 2 Register Default Value (Hex)
GPIO000	0000	00001040	GPIO000	500	000
GPIO001	0004	00000040	GPIO001	504	000
GPIO002	0008	00000040	GPIO002	508	000
GPIO003	000C	00000040	GPIO003	50C	000
GPIO004	0010	00000040	GPIO004	510	000
GPIO005	0014	00000040	GPIO005	514	000
GPIO006	0018	00000040	GPIO006	518	000
GPIO007	001C	00000040	GPIO007	51C	000
GPIO010	0020	00000040	GPIO010	520	000
GPIO011	0024	00000040	GPIO011	524	000
GPIO012	0028	00000040	GPIO012	528	000
GPIO013	002C	00000040	GPIO013	52C	000
GPIO014	0030	00000040	GPIO014	530	000

TABLE 3-2: GPIO PIN CONTROL REGISTERS DEFAULTS (CONTINUED)

GPIO Name	Pin Control Register Offset (Hex)	Pin Control Register Default Value (Hex)	Pin Control Register Default Function	Pin Control 2 Register Offset (Hex)	Pin Control 2 Register Default Value (Hex)
GPIO015	0034	00000040	GPIO015	534	000
GPIO016	0038	00000040	GPIO016	538	000
GPIO017	003C	00000040	GPIO017	53C	000
GPIO020	0040	00000040	GPIO020	540	000
GPIO021	0044	00000040	GPIO021	544	000
GPIO022	0048	00000040	GPIO022	548	000
GPIO023	004C	00000040	GPIO023	54C	000
GPIO024	0050	00000040	GPIO024	550	000
GPIO025	0054	00000040	GPIO025	554	000
GPIO026	0058	00000040	GPIO026	558	000
GPIO027	005C	00000040	GPIO027	55C	000
GPIO030	0060	00000040	GPIO030	560	000
GPIO031	0064	00000040	GPIO031	564	000
GPIO032	0068	00000040	GPIO032	568	000
GPIO033	006C	00000040	GPIO033	56C	000
GPIO034	0070	00000040	GPIO034	570	000
GPIO035	0074	00000040	GPIO035	574	000
GPIO036	0078	00000040	GPIO036	578	000
GPIO040	0080	00000040	GPIO040	580	000
GPIO041	0084	00000040	GPIO041	584	000
GPIO042	0088	00000040	GPIO042	588	000
GPIO043	008C	00000040	GPIO043	58C	000
GPIO044	0090	00000040	GPIO044	590	000
GPIO045	0094	00000040	GPIO045	594	000
GPIO046	0098	00000040	GPIO046	598	000
GPIO047	009C	00000040	GPIO047	59C	000
GPIO050	00A0	00000040	GPIO050	5A0	000
GPIO051	00A4	00000040	GPIO051	5A4	000
GPIO052	00A8	00000040	GPIO052	5A8	000
GPIO053	00AC	00000040	GPIO053	5AC	000
GPIO054	00B0	00000040	GPIO054	5B0	000
GPIO055	00B4	00000040	GPIO055	5B4	000
GPIO056	00B8	00000040	GPIO056	5B8	000
GPIO057	00BC	00001040	GPIO057	5BC	000
GPIO060	00C0	00000040	GPIO060	5C0	000
GPIO061	00C4	00000040	GPIO061	5C4	000
GPIO062	00C8	00000240	GPIO062	5C8	000
GPIO063	00CC	00000040	GPIO063	5CC	000
GPIO064	00D0	00001000	GPIO064	5D0	000
GPIO065	00D4	00000040	GPIO065	5D4	000
GPIO066	00D8	00000040	GPIO066	5D8	000
GPIO067	00DC	00000040	GPIO067	5DC	000

TABLE 3-2: GPIO PIN CONTROL REGISTERS DEFAULTS (CONTINUED)

GPIO Name	Pin Control Register Offset (Hex)	Pin Control Register Default Value (Hex)	Pin Control Register Default Function	Pin Control 2 Register Offset (Hex)	Pin Control 2 Register Default Value (Hex)
GPIO070	00E0	00000040	GPIO070	5E0	000
GPIO071	00E4	00000040	GPIO071	5E4	000
GPIO072	00E8	00000040	GPIO072	5E8	000
GPIO073	00EC	00000040	GPIO073	5EC	000
GPIO100	0100	00000040	GPIO100	600	000
GPIO101	0104	00000040	GPIO101	604	000
GPIO102	0108	00000040	GPIO102	608	000
GPIO103	010C	00000040	GPIO103	60C	000
GPIO104	0110	00000040	GPIO104	610	000
GPIO105	0114	00000040	GPIO105	614	000
GPIO106	0118	00000040	GPIO106	618	000
GPIO107	011C	00000040	GPIO107	61C	000
GPIO110	0120	00000040	GPIO110	620	000
GPIO111	0124	00000040	GPIO111	624	000
GPIO112	0128	00000040	GPIO112	628	000
GPIO113	012C	00000040	GPIO113	62C	000
GPIO114	0130	00000040	GPIO114	630	000
GPIO115	0134	00000040	GPIO115	634	000
GPIO120	0140	00000040	GPIO120	640	000
GPIO121	0144	00000040	GPIO121	644	000
GPIO122	0148	00000040	GPIO122	648	000
GPIO123	014C	00000040	GPIO123	64C	000
GPIO124	0150	00000040	GPIO124	650	000
GPIO125	0154	00000040	GPIO125	654	000
GPIO126	0158	00000040	GPIO126	658	000
GPIO127	015C	00000040	GPIO127	65C	000
GPIO130	0160	00000040	GPIO130	660	000
GPIO131	0164	00000040	GPIO131	664	000
GPIO132	0168	00000040	GPIO132	668	000
GPIO133	016C	00000040	GPIO133	66C	000
GPIO134	0170	00000040	GPIO134	670	000
GPIO135	0174	00000040	GPIO135	674	000
GPIO140	0180	00000040	GPIO140	680	000
GPIO141	0184	00000040	GPIO141	684	000
GPIO142	0188	00000040	GPIO142	688	000
GPIO143	018C	00000040	GPIO143	68C	000
GPIO144	0190	00000040	GPIO144	690	000
GPIO145	0194	00000040	GPIO145	694	000
GPIO146	0198	00000040	GPIO146	698	000
GPIO147	019C	00000040	GPIO147	69C	000
GPIO150	01A0	00000040	GPIO150	6A0	000
GPIO151	01A4	00000040	GPIO151	6A4	000

TABLE 3-2: GPIO PIN CONTROL REGISTERS DEFAULTS (CONTINUED)

GPIO Name	Pin Control Register Offset (Hex)	Pin Control Register Default Value (Hex)	Pin Control Register Default Function	Pin Control 2 Register Offset (Hex)	Pin Control 2 Register Default Value (Hex)
GPIO152	01A8	00000040	GPIO152	6A8	000
GPIO153	01AC	00000040	GPIO153	6AC	000
GPIO154	01B0	00000040	GPIO154	6B0	000
GPIO155	01B4	00000040	GPIO155	6B4	000
GPIO156	01B8	00000040	GPIO156	6B8	000
GPIO157	01BC	00000040	GPIO157	6BC	000
GPIO160	01C0	00000040	GPIO160	6C0	000
GPIO161	01C4	00001040	GPIO161	6C4	000
GPIO162	01C8	00001040	VCI_IN1#	6C8	000
GPIO163	01CC	00001040	VCI_IN0#	6CC	000
GPIO165	01D4	00000040	GPIO165	6D4	000
GPIO166	01D8	00000040	GPIO166	6D8	000
GPIO167	01DC	00001040	GPIO167	6DC	000
GPIO170	01E0	00000040	GPIO170	6E0	000
GPIO171	01E4	00000041	GPIO171	6E4	000
GPIO172	01E8	00000040	GPIO172	6E8	000
GPIO173	01EC	00000040	GPIO173	6EC	000
GPIO174	01F0	00000040	GPIO174	6F0	000
GPIO175	01F4	00000040	GPIO175	6F4	000
GPIO200	0200	00001040	ADC0	700	000
GPIO201	0204	00001040	ADC1	704	000
GPIO202	0208	00001040	ADC2	708	000
GPIO203	020C	00001040	ADC3	70C	000
GPIO204	0210	00001040	ADC4	710	000
GPIO205	0214	00001040	GPIO205	714	000
GPIO206	0218	00001040	GPIO206	718	000
GPIO207	021C	00001040	GPIO207	71C	000
GPIO210	0220	00001040	GPIO210	720	000
GPIO211	0224	00001040	GPIO211	724	000
GPIO212	0228	00001040	GPIO212	728	000
GPIO213	022C	00001040	GPIO213	72C	000
GPIO214	0230	00001040	GPIO214	730	000
GPIO215	0234	00001040	GPIO215	734	000
GPIO216	0238	00001040	GPIO216	738	000
GPIO217	023C	00001040	GPIO217	73C	000
GPIO221	0244	00000040	GPIO221	744	000
GPIO222	0248	00000040	GPIO222	748	000
GPIO223	024C	00000040	GPIO223	74C	000
GPIO224	0250	00000040	GPIO224	750	000
GPIO225	0254	00000040	GPIO225	754	000
GPIO226	0258	00000040	GPIO226	758	000
GPIO227	025C	00000040	GPIO227	75C	000

TABLE 3-3: GPIO INPUT AND OUTPUT REGISTERS

Offset	Register Name
300h	Input GPIO[036:000] Register
304h	Input GPIO[076:040] Register
308h	Input GPIO[136:100] Register
30Ch	Input GPIO[176:140] Register
310h	Input GPIO[236:200] Register
380h	Output GPIO[036:000] Register
384h	Output GPIO[076:040] Register
388h	Output GPIO[136:100] Register
38Ch	Output GPIO[176:140] Register
390h	Output GPIO[236:200] Register

3.7 Register Map

Block	Instance	Register	Register Address
Watchdog Timer	0	WDT Load Register	40000000h
Watchdog Timer	0	WDT Control Register	40000004h
Watchdog Timer	0	WDT Kick Register	40000008h
Watchdog Timer	0	WDT Count Register	4000000Ch
16-bit Basic Timer	0	Timer Count Register	40000C00h
16-bit Basic Timer	0	Timer Preload Register	40000C04h
16-bit Basic Timer	0	Timer Status Register	40000C08h
16-bit Basic Timer	0	Timer Int Enable Register	40000C0Ch
16-bit Basic Timer	0	Timer Control Register	40000C10h
16-bit Basic Timer	1	Timer Count Register	40000C20h
16-bit Basic Timer	1	Timer Preload Register	40000C24h
16-bit Basic Timer	1	Timer Status Register	40000C28h
16-bit Basic Timer	1	Timer Int Enable Register	40000C2Ch
16-bit Basic Timer	1	Timer Control Register	40000C30h
16-bit Basic Timer	2	Timer Count Register	40000C40h
16-bit Basic Timer	2	Timer Preload Register	40000C44h
16-bit Basic Timer	2	Timer Status Register	40000C48h
16-bit Basic Timer	2	Timer Int Enable Register	40000C4Ch
16-bit Basic Timer	2	Timer Control Register	40000C50h
16-bit Basic Timer	3	Timer Count Register	40000C60h
16-bit Basic Timer	3	Timer Preload Register	40000C64h
16-bit Basic Timer	3	Timer Status Register	40000C68h
16-bit Basic Timer	3	Timer Int Enable Register	40000C6Ch
16-bit Basic Timer	3	Timer Control Register	40000C70h
32-bit Basic Timer	0	Timer Count Register	40000C80h
32-bit Basic Timer	0	Timer Preload Register	40000C84h
32-bit Basic Timer	0	Timer Status Register	40000C88h
32-bit Basic Timer	0	Timer Int Enable Register	40000C8Ch
32-bit Basic Timer	0	Timer Control Register	40000C90h
32-bit Basic Timer	1	Timer Count Register	40000CA0h
32-bit Basic Timer	1	Timer Preload Register	40000CA4h
32-bit Basic Timer	1	Timer Status Register	40000CA8h
32-bit Basic Timer	1	Timer Int Enable Register	40000CACh
32-bit Basic Timer	1	Timer Control Register	40000CB0h
16-bit Counter Timer	0	Timer x Control Register	40000D00h
16-bit Counter Timer	0	Timer x Clock and Event Control Register	40000D04h
16-bit Counter Timer	0	Timer x Reload Register	40000D08h
16-bit Counter Timer	0	Timer x Count Register	40000D0Ch
16-bit Counter Timer	1	Timer x Control Register	40000D20h
16-bit Counter Timer	1	Timer x Clock and Event Control Register	40000D24h
16-bit Counter Timer	1	Timer x Reload Register	40000D28h

Block	Instance	Register	Register Address
16-bit Counter Timer	1	Timer x Count Register	40000D2Ch
16-bit Counter Timer	2	Timer x Control Register	40000D40h
16-bit Counter Timer	2	Timer x Clock and Event Control Register	40000D44h
16-bit Counter Timer	2	Timer x Reload Register	40000D48h
16-bit Counter Timer	2	Timer x Count Register	40000D4Ch
16-bit Counter Timer	3	Timer x Control Register	40000D60h
16-bit Counter Timer	3	Timer x Clock and Event Control Register	40000D64h
16-bit Counter Timer	3	Timer x Reload Register	40000D68h
16-bit Counter Timer	3	Timer x Count Register	40000D6Ch
Capture Compare Timer	0	Capture and Compare Timer Control Register	40001000h
Capture Compare Timer	0	Capture Control 0 Register	40001004h
Capture Compare Timer	0	Capture Control 1 Register	40001008h
Capture Compare Timer	0	Free Running Timer Register	4000100Ch
Capture Compare Timer	0	Capture 0 Register	40001010h
Capture Compare Timer	0	Capture 1 Register	40001014h
Capture Compare Timer	0	Capture 2 Register	40001018h
Capture Compare Timer	0	Capture 3 Register	4000101Ch
Capture Compare Timer	0	Capture 4 Register	40001020h
Capture Compare Timer	0	Capture 5 Register	40001024h
Capture Compare Timer	0	Compare 0 Register	40001028h
Capture Compare Timer	0	Compare 1 Register	4000102Ch
RC-ID	1	RC_ID Control Register	40001480h
RC-ID	1	RC_ID Data Register	40001484h
RC-ID	2	RC_ID Control Register	40001500h
RC-ID	2	RC_ID Data Register	40001504h
DMA Controller	0	DMA Main Control Register	40002400h
DMA Controller	0	DMA Data Packet Register	40002404h
DMA Controller	0	TEST	40002408h
DMA Channel	0	DMA Channel N Activate Register	40002440h
DMA Channel	0	DMA Channel N Memory Start Address Register	40002444h
DMA Channel	0	DMA Channel N Memory End Address Register	40002448h
DMA Channel	0	DMA Channel N Device Address	4000244Ch
DMA Channel	0	DMA Channel N Control Register	40002450h
DMA Channel	0	DMA Channel N Interrupt Status Register	40002454h
DMA Channel	0	DMA Channel N Interrupt Enable Register	40002458h
DMA Channel	0	TEST	4000245Ch
DMA Channel	0	Channel N CRC Enable Register	40002460h
DMA Channel	0	Channel N CRC Data Register	40002464h
DMA Channel	0	Channel N CRC Post Status Register	40002468h
DMA Channel	0	TEST	4000246Ch
DMA Channel	1	DMA Channel N Activate Register	40002480h
DMA Channel	1	DMA Channel N Memory Start Address Register	40002484h
DMA Channel	1	DMA Channel N Memory End Address Register	40002488h

Block	Instance	Register	Register Address
DMA Channel	1	DMA Channel N Device Address	4000248Ch
DMA Channel	1	DMA Channel N Control Register	40002490h
DMA Channel	1	DMA Channel N Interrupt Status Register	40002494h
DMA Channel	1	DMA Channel N Interrupt Enable Register	40002498h
DMA Channel	1	TEST	4000249Ch
DMA Channel	1	Channel N Fill Enable Register	400024A0h
DMA Channel	1	Channel N Fill Data Register	400024A4h
DMA Channel	1	Channel N Fill Status Register	400024A8h
DMA Channel	1	TEST	400024ACh
DMA Channel	2	DMA Channel N Activate Register	400024C0h
DMA Channel	2	DMA Channel N Memory Start Address Register	400024C4h
DMA Channel	2	DMA Channel N Memory End Address Register	400024C8h
DMA Channel	2	DMA Channel N Device Address	400024CCh
DMA Channel	2	DMA Channel N Control Register	400024D0h
DMA Channel	2	DMA Channel N Interrupt Status Register	400024D4h
DMA Channel	2	DMA Channel N Interrupt Enable Register	400024D8h
DMA Channel	2	TEST	400024DCh
DMA Channel	3	DMA Channel N Activate Register	40002500h
DMA Channel	3	DMA Channel N Memory Start Address Register	40002504h
DMA Channel	3	DMA Channel N Memory End Address Register	40002508h
DMA Channel	3	DMA Channel N Device Address	4000250Ch
DMA Channel	3	DMA Channel N Control Register	40002510h
DMA Channel	3	DMA Channel N Interrupt Status Register	40002514h
DMA Channel	3	DMA Channel N Interrupt Enable Register	40002518h
DMA Channel	3	TEST	4000251Ch
DMA Channel	4	DMA Channel N Activate Register	40002540h
DMA Channel	4	DMA Channel N Memory Start Address Register	40002544h
DMA Channel	4	DMA Channel N Memory End Address Register	40002548h
DMA Channel	4	DMA Channel N Device Address	4000254Ch
DMA Channel	4	DMA Channel N Control Register	40002550h
DMA Channel	4	DMA Channel N Interrupt Status Register	40002554h
DMA Channel	4	DMA Channel N Interrupt Enable Register	40002558h
DMA Channel	4	TEST	4000255Ch
DMA Channel	5	DMA Channel N Activate Register	40002580h
DMA Channel	5	DMA Channel N Memory Start Address Register	40002584h
DMA Channel	5	DMA Channel N Memory End Address Register	40002588h
DMA Channel	5	DMA Channel N Device Address	4000258Ch
DMA Channel	5	DMA Channel N Control Register	40002590h
DMA Channel	5	DMA Channel N Interrupt Status Register	40002594h
DMA Channel	5	DMA Channel N Interrupt Enable Register	40002598h
DMA Channel	5	TEST	4000259Ch
DMA Channel	6	DMA Channel N Activate Register	400025C0h
DMA Channel	6	DMA Channel N Memory Start Address Register	400025C4h

Block	Instance	Register	Register Address
DMA Channel	6	DMA Channel N Memory End Address Register	400025C8h
DMA Channel	6	DMA Channel N Device Address	400025CCh
DMA Channel	6	DMA Channel N Control Register	400025D0h
DMA Channel	6	DMA Channel N Interrupt Status Register	400025D4h
DMA Channel	6	DMA Channel N Interrupt Enable Register	400025D8h
DMA Channel	6	TEST	400025DCh
DMA Channel	7	DMA Channel N Activate Register	40002600h
DMA Channel	7	DMA Channel N Memory Start Address Register	40002604h
DMA Channel	7	DMA Channel N Memory End Address Register	40002608h
DMA Channel	7	DMA Channel N Device Address	4000260Ch
DMA Channel	7	DMA Channel N Control Register	40002610h
DMA Channel	7	DMA Channel N Interrupt Status Register	40002614h
DMA Channel	7	DMA Channel N Interrupt Enable Register	40002618h
DMA Channel	7	TEST	4000261Ch
DMA Channel	8	DMA Channel N Activate Register	40002640h
DMA Channel	8	DMA Channel N Memory Start Address Register	40002644h
DMA Channel	8	DMA Channel N Memory End Address Register	40002648h
DMA Channel	8	DMA Channel N Device Address	4000264Ch
DMA Channel	8	DMA Channel N Control Register	40002650h
DMA Channel	8	DMA Channel N Interrupt Status Register	40002654h
DMA Channel	8	DMA Channel N Interrupt Enable Register	40002658h
DMA Channel	8	TEST	4000265Ch
DMA Channel	9	DMA Channel N Activate Register	40002680h
DMA Channel	9	DMA Channel N Memory Start Address Register	40002684h
DMA Channel	9	DMA Channel N Memory End Address Register	40002688h
DMA Channel	9	DMA Channel N Device Address	4000268Ch
DMA Channel	9	DMA Channel N Control Register	40002690h
DMA Channel	9	DMA Channel N Interrupt Status Register	40002694h
DMA Channel	9	DMA Channel N Interrupt Enable Register	40002698h
DMA Channel	9	TEST	4000269Ch
DMA Channel	10	DMA Channel N Activate Register	400026C0h
DMA Channel	10	DMA Channel N Memory Start Address Register	400026C4h
DMA Channel	10	DMA Channel N Memory End Address Register	400026C8h
DMA Channel	10	DMA Channel N Device Address	400026CCh
DMA Channel	10	DMA Channel N Control Register	400026D0h
DMA Channel	10	DMA Channel N Interrupt Status Register	400026D4h
DMA Channel	10	DMA Channel N Interrupt Enable Register	400026D8h
DMA Channel	10	TEST	400026DCh
DMA Channel	11	DMA Channel N Activate Register	40002700h
DMA Channel	11	DMA Channel N Memory Start Address Register	40002704h
DMA Channel	11	DMA Channel N Memory End Address Register	40002708h
DMA Channel	11	DMA Channel N Device Address	4000270Ch
DMA Channel	11	DMA Channel N Control Register	40002710h

Block	Instance	Register	Register Address
DMA Channel	11	DMA Channel N Interrupt Status Register	40002714h
DMA Channel	11	DMA Channel N Interrupt Enable Register	40002718h
DMA Channel	11	TEST	4000271Ch
DMA Channel	12	DMA Channel N Activate Register	40002740h
DMA Channel	12	DMA Channel N Memory Start Address Register	40002744h
DMA Channel	12	DMA Channel N Memory End Address Register	40002748h
DMA Channel	12	DMA Channel N Device Address	4000274Ch
DMA Channel	12	DMA Channel N Control Register	40002750h
DMA Channel	12	DMA Channel N Interrupt Status Register	40002754h
DMA Channel	12	DMA Channel N Interrupt Enable Register	40002758h
DMA Channel	12	TEST	4000275Ch
DMA Channel	13	DMA Channel N Activate Register	40002780h
DMA Channel	13	DMA Channel N Memory Start Address Register	40002784h
DMA Channel	13	DMA Channel N Memory End Address Register	40002788h
DMA Channel	13	DMA Channel N Device Address	4000278Ch
DMA Channel	13	DMA Channel N Control Register	40002790h
DMA Channel	13	DMA Channel N Interrupt Status Register	40002794h
DMA Channel	13	DMA Channel N Interrupt Enable Register	40002798h
DMA Channel	13	TEST	4000279Ch
I2C	0	Control Register	40004000h
I2C	0	Status Register	40004000h
I2C	0	Own Address Register	40004004h
I2C	0	Data Register	40004008h
I2C	0	Master Command Register	4000400Ch
I2C	0	Slave Command Register	40004010h
I2C	0	PEC Register	40004014h
I2C	0	Repeated START Hold Time Register	40004018h
I2C	0	Completion Register	40004020h
I2C	0	Idle Scaling Register	40004024h
I2C	0	Configuration Register	40004028h
I2C	0	Bus Clock Register	4000402Ch
I2C	0	Block ID Register	40004030h
I2C	0	Revision Register	40004034h
I2C	0	Bit-Bang Control Register	40004038h
I2C	0	TEST	4000403Ch
I2C	0	Data Timing Register	40004040h
I2C	0	Time-Out Scaling Register	40004044h
I2C	0	Slave Transmit Buffer Register	40004048h
I2C	0	Slave Receive Buffer Register	4000404Ch
I2C	0	Master Transmit Buffer Register	40004050h
I2C	0	Master Receive Buffer Register	40004054h
I2C	0	TEST	40004058h
I2C	0	TEST	4000405Ch

Block	Instance	Register	Register Address
I2C	0	Wake Status Register	40004060h
I2C	0	Wake Enable Register	40004064h
I2C	0	TEST	40004068h
I2C	1	Control Register	40004400h
I2C	1	Status Register	40004400h
I2C	1	Own Address Register	40004404h
I2C	1	Data Register	40004408h
I2C	1	Master Command Register	4000440Ch
I2C	1	Slave Command Register	40004410h
I2C	1	PEC Register	40004414h
I2C	1	Repeated START Hold Time Register	40004418h
I2C	1	Completion Register	40004420h
I2C	1	Idle Scaling Register	40004424h
I2C	1	Configuration Register	40004428h
I2C	1	Bus Clock Register	4000442Ch
I2C	1	Block ID Register	40004430h
I2C	1	Revision Register	40004434h
I2C	1	Bit-Bang Control Register	40004438h
I2C	1	TEST	4000443Ch
I2C	1	Data Timing Register	40004440h
I2C	1	Time-Out Scaling Register	40004444h
I2C	1	Slave Transmit Buffer Register	40004448h
I2C	1	Slave Receive Buffer Register	4000444Ch
I2C	1	Master Transmit Buffer Register	40004450h
I2C	1	Master Receive Buffer Register	40004454h
I2C	1	TEST	40004458h
I2C	1	TEST	4000445Ch
I2C	1	Wake Status Register	40004460h
I2C	1	Wake Enable Register	40004464h
I2C	1	TEST	40004468h
I2C	2	Control Register	40004800h
I2C	2	Status Register	40004800h
I2C	2	Own Address Register	40004804h
I2C	2	Data Register	40004808h
I2C	2	Master Command Register	4000480Ch
I2C	2	Slave Command Register	40004810h
I2C	2	PEC Register	40004814h
I2C	2	Repeated START Hold Time Register	40004818h
I2C	2	Completion Register	40004820h
I2C	2	Idle Scaling Register	40004824h
I2C	2	Configuration Register	40004828h
I2C	2	Bus Clock Register	4000482Ch
I2C	2	Block ID Register	40004830h

Block	Instance	Register	Register Address
I2C	2	Revision Register	40004834h
I2C	2	Bit-Bang Control Register	40004838h
I2C	2	TEST	4000483Ch
I2C	2	Data Timing Register	40004840h
I2C	2	Time-Out Scaling Register	40004844h
I2C	2	Slave Transmit Buffer Register	40004848h
I2C	2	Slave Receive Buffer Register	4000484Ch
I2C	2	Master Transmit Buffer Register	40004850h
I2C	2	Master Receive Buffer Register	40004854h
I2C	2	TEST	40004858h
I2C	2	TEST	4000485Ch
I2C	2	Wake Status Register	40004860h
I2C	2	Wake Enable Register	40004864h
I2C	2	TEST	40004868h
I2C	3	Control Register	40004C00h
I2C	3	Status Register	40004C00h
I2C	3	Own Address Register	40004C04h
I2C	3	Data Register	40004C08h
I2C	3	Master Command Register	40004C0Ch
I2C	3	Slave Command Register	40004C10h
I2C	3	PEC Register	40004C14h
I2C	3	Repeated START Hold Time Register	40004C18h
I2C	3	Completion Register	40004C20h
I2C	3	Idle Scaling Register	40004C24h
I2C	3	Configuration Register	40004C28h
I2C	3	Bus Clock Register	40004C2Ch
I2C	3	Block ID Register	40004C30h
I2C	3	Revision Register	40004C34h
I2C	3	Bit-Bang Control Register	40004C38h
I2C	3	TEST	40004C3Ch
I2C	3	Data Timing Register	40004C40h
I2C	3	Time-Out Scaling Register	40004C44h
I2C	3	Slave Transmit Buffer Register	40004C48h
I2C	3	Slave Receive Buffer Register	40004C4Ch
I2C	3	Master Transmit Buffer Register	40004C50h
I2C	3	Master Receive Buffer Register	40004C54h
I2C	3	TEST	40004C58h
I2C	3	TEST	40004C5Ch
I2C	3	Wake Status Register	40004C60h
I2C	3	Wake Enable Register	40004C64h
I2C	3	TEST	40004C68h
QMSPI	0	QMSPI Mode Register	40005400h
QMSPI	0	QMSPI Control Register	40005404h

Block	Instance	Register	Register Address
QMSPI	0	QMSPI Execute Register	40005408h
QMSPI	0	QMSPI Interface Control Register	4000540Ch
QMSPI	0	QMSPI Status Register	40005410h
QMSPI	0	QMSPI Buffer Count Status Register	40005414h
QMSPI	0	QMSPI Interrupt Enable Register	40005418h
QMSPI	0	QMSPI Buffer Count Trigger Register	4000541Ch
QMSPI	0	QMSPI Transmit Buffer Register	40005420h
QMSPI	0	QMSPI Receive Buffer Register	40005424h
QMSPI	0	QMSPI Chip Select Timing Register	40005428h
QMSPI	0	QMSPI Description Buffer 0 Register	40005430h
QMSPI	0	QMSPI Description Buffer 1 Register	40005434h
QMSPI	0	QMSPI Description Buffer 2 Register	40005438h
QMSPI	0	QMSPI Description Buffer 3 Register	4000543Ch
QMSPI	0	QMSPI Description Buffer 4 Register	40005440h
16-bit PWM	0	PWMx Counter ON Time Register	40005800h
16-bit PWM	0	PWMx Counter OFF Time Register	40005804h
16-bit PWM	0	PWMx Configuration Register	40005808h
16-bit PWM	0	TEST	4000580Ch
16-bit PWM	1	PWMx Counter ON Time Register	40005810h
16-bit PWM	1	PWMx Counter OFF Time Register	40005814h
16-bit PWM	1	PWMx Configuration Register	40005818h
16-bit PWM	1	TEST	4000581Ch
16-bit PWM	2	PWMx Counter ON Time Register	40005820h
16-bit PWM	2	PWMx Counter OFF Time Register	40005824h
16-bit PWM	2	PWMx Configuration Register	40005828h
16-bit PWM	2	TEST	4000582Ch
16-bit PWM	3	PWMx Counter ON Time Register	40005830h
16-bit PWM	3	PWMx Counter OFF Time Register	40005834h
16-bit PWM	3	PWMx Configuration Register	40005838h
16-bit PWM	3	TEST	4000583Ch
16-bit PWM	4	PWMx Counter ON Time Register	40005840h
16-bit PWM	4	PWMx Counter OFF Time Register	40005844h
16-bit PWM	4	PWMx Configuration Register	40005848h
16-bit PWM	4	TEST	4000584Ch
16-bit PWM	5	PWMx Counter ON Time Register	40005850h
16-bit PWM	5	PWMx Counter OFF Time Register	40005854h
16-bit PWM	5	PWMx Configuration Register	40005858h
16-bit PWM	5	TEST	4000585Ch
16-bit PWM	10	PWMx Counter ON Time Register	400058A0h
16-bit PWM	10	PWMx Counter OFF Time Register	400058A4h
16-bit PWM	10	PWMx Configuration Register	400058A8h
16-bit PWM	10	TEST	400058ACh
16-bit Tach	0	TACHx Control Register	40006000h

Block	Instance	Register	Register Address
16-bit Tach	0	TACHx Status Register	40006004h
16-bit Tach	0	TACHx High Limit Register	40006008h
16-bit Tach	0	TACHx Low Limit Register	4000600Ch
16-bit Tach	1	TACHx Control Register	40006010h
16-bit Tach	1	TACHx Status Register	40006014h
16-bit Tach	1	TACHx High Limit Register	40006018h
16-bit Tach	1	TACHx Low Limit Register	4000601Ch
RTOS Timer	0	RTOS Timer Count Register	40007400h
RTOS Timer	0	RTOS Timer Preload Register	40007404h
RTOS Timer	0	RTOS Timer Control Register	40007408h
RTOS Timer	0	Soft Interrupt Register	4000740Ch
ADC	0	ADC Control Register	40007C00h
ADC	0	ADC Delay Register	40007C04h
ADC	0	ADC Status Register	40007C08h
ADC	0	ADC Single Register	40007C0Ch
ADC	0	ADC Repeat Register	40007C10h
ADC	0	ADC Channel 0 Reading Register	40007C14h
ADC	0	ADC Channel 1 Reading Register	40007C18h
ADC	0	ADC Channel 2 Reading Register	40007C1Ch
ADC	0	ADC Channel 3 Reading Register	40007C20h
ADC	0	ADC Channel 4 Reading Register	40007C24h
ADC	0	ADC Test Register	40007C78h
ADC	0	ADC Configuration Register	40007C7Ch
TFDP	0	Debug Data Register	40008C00h
TFDP	0	Debug Control Register	40008C04h
GP-SPI	0	SPI Enable Register	40009400h
GP-SPI	0	SPI Control Register	40009404h
GP-SPI	0	SPI Status Register	40009408h
GP-SPI	0	SPI TX_Data Register	4000940Ch
GP-SPI	0	SPI RX_Data Register	40009410h
GP-SPI	0	SPI Clock Control Register	40009414h
GP-SPI	0	SPI Clock Generator Register	40009418h
GP-SPI	0	TESET	40009420h
Hibernation Timer	0	HTimer Preload Register	40009800h
Hibernation Timer	0	HTimer Control Register	40009804h
Hibernation Timer	0	HTimer Count Register	40009808h
Hibernation Timer	1	HTimer Preload Register	40009820h
Hibernation Timer	1	HTimer Control Register	40009824h
Hibernation Timer	1	HTimer Count Register	40009828h
Keyscan	0	KSO Select Register	40009C04h
Keyscan	0	KSI INPUT Register	40009C08h
Keyscan	0	KSI STATUS Register	40009C0Ch
Keyscan	0	KSI INTERRUPT ENABLE Register	40009C10h

Block	Instance	Register	Register Address
Keyscan	0	Keyscan Extended Control Register	40009C14h
RPM2PWM	0	Fan Setting Register	4000A000h
RPM2PWM	0	Reserved	4000A001h
RPM2PWM	0	Fan Configuration 1 Register	4000A002h
RPM2PWM	0	Fan Configuration 2 Register	4000A003h
RPM2PWM	0	PWM Divide Register	4000A004h
RPM2PWM	0	Gain Register	4000A005h
RPM2PWM	0	Fan Spin Up Configuration Register	4000A006h
RPM2PWM	0	Fan Step Register	4000A007h
RPM2PWM	0	Fan Minimum Drive Register	4000A008h
RPM2PWM	0	Valid TACH Count Register	4000A009h
RPM2PWM	0	Fan Drive Fail Band Register	4000A00Ah
RPM2PWM	0	TACH Target Register	4000A00Ch
RPM2PWM	0	TACH Reading Register	4000A00Eh
RPM2PWM	0	PWM Driver Base Frequency Register	4000A010h
RPM2PWM	0	Fan Status Register	4000A011h
RPM2PWM	0	TEST	4000A012h
RPM2PWM	0	TEST	4000A014h
RPM2PWM	0	TEST	4000A015h
RPM2PWM	0	TEST	4000A016h
RPM2PWM	0	TEST	4000A017h
RPM2PWM	1	Fan Setting Register	4000A080h
RPM2PWM	1	PWM Divide Register	4000A081h
RPM2PWM	1	Fan Configuration 1 Register	4000A082h
RPM2PWM	1	Fan Configuration 2 Register	4000A083h
RPM2PWM	1	Reserved	4000A084h
RPM2PWM	1	Gain Register	4000A085h
RPM2PWM	1	Fan Spin Up Configuration Register	4000A086h
RPM2PWM	1	Fan Step Register	4000A087h
RPM2PWM	1	Fan Minimum Drive Register	4000A088h
RPM2PWM	1	Valid TACH Count Register	4000A089h
RPM2PWM	1	Fan Drive Fail Band Register	4000A08Ah
RPM2PWM	1	TACH Target Register	4000A08Ch
RPM2PWM	1	TACH Reading Register	4000A08Eh
RPM2PWM	1	PWM Driver Base Frequency Register	4000A090h
RPM2PWM	1	Fan Status Register	4000A091h
RPM2PWM	1	TEST	4000A092h
RPM2PWM	1	TEST	4000A094h
RPM2PWM	1	TEST	4000A095h
RPM2PWM	1	TEST	4000A096h
RPM2PWM	1	TEST	4000A097h
VBAT Register Bank	0	Power-Fail and Reset Status Register	4000A400h
VBAT Register Bank	0	TEST	4000A404h

Block	Instance	Register	Register Address
VBAT Register Bank	0	Clock Enable Register	4000A408h
VBAT Register Bank	0	TEST	4000A40Ch
VBAT Powered RAM	0	Registers	4000A800h
VBAT Register Bank	0	TEST	4000A410h
VBAT Register Bank	0	TEST	4000A414h
VBAT Register Bank	0	TEST	4000A418h
VBAT Register Bank	0	TEST	4000A41Ch
VBAT Register Bank	0	Monotonic Counter Register	4000A420h
VBAT Register Bank	0	Counter HiWord Register	4000A424h
VBAT Register Bank	0	TEST	4000A428h
VBAT Register Bank	0	TEST	4000A42Ch
Week Timer	0	Control Register	4000AC80h
Week Timer	0	Week Alarm Counter Register	4000AC84h
Week Timer	0	Week Timer Compare Register	4000AC88h
Week Timer	0	Clock Divider Register	4000AC8Ch
Week Timer	0	Sub-Second Programmable Interrupt Select Register	4000AC90h
Week Timer	0	Sub-Week Control Register	4000AC94h
Week Timer	0	Sub-Week Alarm Counter Register	4000AC98h
Week Timer	0	BGPO Data Register	4000AC9Ch
Week Timer	0	BGPO Power Register	4000ACA0h
Week Timer	0	BGPO Reset Register	4000ACA4h
VBAT-Powered Control Interface	0	VCI Register	4000AE00h
VBAT-Powered Control Interface	0	Latch Enable Register	4000AE04h
VBAT-Powered Control Interface	0	Latch Resets Register	4000AE08h
VBAT-Powered Control Interface	0	VCI Input Enable Register	4000AE0Ch
VBAT-Powered Control Interface	0	Holdoff Count Register	4000AE10h
VBAT-Powered Control Interface	0	VCI Polarity Register	4000AE14h
VBAT-Powered Control Interface	0	VCI Posedge Detect Register	4000AE18h
VBAT-Powered Control Interface	0	VCI Negedge Detect Register	4000AE1Ch
VBAT-Powered Control Interface	0	VCI Buffer Enable Register	4000AE20h
Blinking-Breathing PWM	0	LED Configuration Register	4000B800h
Blinking-Breathing PWM	0	LED Limits Register	4000B804h
Blinking-Breathing PWM	0	LED Delay Register	4000B808h
Blinking-Breathing PWM	0	LED Update Stepsize Register	4000B80Ch
Blinking-Breathing PWM	0	LED Update Interval Register	4000B810h
Blinking-Breathing PWM	0	LED Output Delay	4000B814h
Blinking-Breathing PWM	1	LED Configuration Register	4000B900h
Blinking-Breathing PWM	1	LED Limits Register	4000B904h
Blinking-Breathing PWM	1	LED Delay Register	4000B908h
Blinking-Breathing PWM	1	LED Update Stepsize Register	4000B90Ch
Blinking-Breathing PWM	1	LED Update Interval Register	4000B910h
Blinking-Breathing PWM	1	LED Output Delay	4000B914h

Block	Instance	Register	Register Address
Interrupt Aggregator	0	GIRQ8 Source Register	4000E000h
Interrupt Aggregator	0	GIRQ8 Enable Set Register	4000E004h
Interrupt Aggregator	0	GIRQ8 Result Register	4000E008h
Interrupt Aggregator	0	GIRQ8 Enable Clear Register	4000E00Ch
Interrupt Aggregator	0	GIRQ9 Source Register	4000E014h
Interrupt Aggregator	0	GIRQ9 Enable Set Register	4000E018h
Interrupt Aggregator	0	GIRQ9 Result Register	4000E01Ch
Interrupt Aggregator	0	GIRQ9 Enable Clear Register	4000E020h
Interrupt Aggregator	0	GIRQ10 Source Register	4000E028h
Interrupt Aggregator	0	GIRQ10 Enable Set Register	4000E02Ch
Interrupt Aggregator	0	GIRQ10 Result Register	4000E030h
Interrupt Aggregator	0	GIRQ10 Enable Clear Register	4000E034h
Interrupt Aggregator	0	GIRQ11 Source Register	4000E03Ch
Interrupt Aggregator	0	GIRQ11 Enable Set Register	4000E040h
Interrupt Aggregator	0	GIRQ11 Result Register	4000E044h
Interrupt Aggregator	0	GIRQ11 Enable Clear Register	4000E048h
Interrupt Aggregator	0	GIRQ12 Source Register	4000E050h
Interrupt Aggregator	0	GIRQ12 Enable Set Register	4000E054h
Interrupt Aggregator	0	GIRQ12 Result Register	4000E058h
Interrupt Aggregator	0	GIRQ12 Enable Clear Register	4000E05Ch
Interrupt Aggregator	0	GIRQ13 Source Register	4000E064h
Interrupt Aggregator	0	GIRQ13 Enable Set Register	4000E068h
Interrupt Aggregator	0	GIRQ13 Result Register	4000E06Ch
Interrupt Aggregator	0	GIRQ13 Enable Clear Register	4000E070h
Interrupt Aggregator	0	GIRQ14 Source Register	4000E078h
Interrupt Aggregator	0	GIRQ14 Enable Set Register	4000E07Ch
Interrupt Aggregator	0	GIRQ14 Result Register	4000E080h
Interrupt Aggregator	0	GIRQ14 Enable Clear Register	4000E084h
Interrupt Aggregator	0	GIRQ15 Source Register	4000E08Ch
Interrupt Aggregator	0	GIRQ15 Enable Set Register	4000E090h
Interrupt Aggregator	0	GIRQ15 Result Register	4000E094h
Interrupt Aggregator	0	GIRQ15 Enable Clear Register	4000E098h
Interrupt Aggregator	0	GIRQ16 Source Register	4000E0A0h
Interrupt Aggregator	0	GIRQ16 Enable Set Register	4000E0A4h
Interrupt Aggregator	0	GIRQ16 Result Register	4000E0A8h
Interrupt Aggregator	0	GIRQ16 Enable Clear Register	4000E0ACh
Interrupt Aggregator	0	GIRQ17 Source Register	4000E0B4h
Interrupt Aggregator	0	GIRQ17 Enable Set Register	4000E0B8h
Interrupt Aggregator	0	GIRQ17 Result Register	4000E0BCh
Interrupt Aggregator	0	GIRQ17 Enable Clear Register	4000E0C0h
Interrupt Aggregator	0	GIRQ18 Source Register	4000E0C8h
Interrupt Aggregator	0	GIRQ18 Enable Set Register	4000E0CCh
Interrupt Aggregator	0	GIRQ18 Result Register	4000E0D0h

Block	Instance	Register	Register Address
Interrupt Aggregator	0	GIRQ18 Enable Clear Register	4000E0D4h
Interrupt Aggregator	0	GIRQ19 Source Register	4000E0DCh
Interrupt Aggregator	0	GIRQ19 Enable Set Register	4000E0E0h
Interrupt Aggregator	0	GIRQ19 Result Register	4000E0E4h
Interrupt Aggregator	0	GIRQ19 Enable Clear Register	4000E0E8h
Interrupt Aggregator	0	GIRQ20 Source Register	4000E0F0h
Interrupt Aggregator	0	GIRQ20 Enable Set Register	4000E0F4h
Interrupt Aggregator	0	GIRQ20 Result Register	4000E0F8h
Interrupt Aggregator	0	GIRQ20 Enable Clear Register	4000E0FCh
Interrupt Aggregator	0	GIRQ21 Source Register	4000E104h
Interrupt Aggregator	0	GIRQ21 Enable Set Register	4000E108h
Interrupt Aggregator	0	GIRQ21 Result Register	4000E10Ch
Interrupt Aggregator	0	GIRQ21 Enable Clear Register	4000E110h
Interrupt Aggregator	0	GIRQ22 Source Register	4000E118h
Interrupt Aggregator	0	GIRQ22 Enable Set Register	4000E11Ch
Interrupt Aggregator	0	GIRQ22 Result Register	4000E120h
Interrupt Aggregator	0	GIRQ22 Enable Clear Register	4000E124h
Interrupt Aggregator	0	GIRQ23 Source Register	4000E12Ch
Interrupt Aggregator	0	GIRQ23 Enable Set Register	4000E130h
Interrupt Aggregator	0	GIRQ23 Result Register	4000E134h
Interrupt Aggregator	0	GIRQ23 Enable Clear Register	4000E138h
Interrupt Aggregator	0	GIRQ24 Source Register	4000E140h
Interrupt Aggregator	0	GIRQ24 Enable Set Register	4000E144h
Interrupt Aggregator	0	GIRQ24 Result Register	4000E148h
Interrupt Aggregator	0	GIRQ24 Enable Clear Register	4000E14Ch
Interrupt Aggregator	0	GIRQ25 Source Register	4000E154h
Interrupt Aggregator	0	GIRQ25 Enable Set Register	4000E158h
Interrupt Aggregator	0	GIRQ25 Result Register	4000E15Ch
Interrupt Aggregator	0	GIRQ25 Enable Clear Register	4000E160h
Interrupt Aggregator	0	GIRQ26 Source Register	4000E168h
Interrupt Aggregator	0	GIRQ26 Enable Set Register	4000E16Ch
Interrupt Aggregator	0	GIRQ26 Result Register	4000E170h
Interrupt Aggregator	0	GIRQ26 Enable Clear Register	4000E174h
Interrupt Aggregator	0	Block Enable Set Register	4000E200h
Interrupt Aggregator	0	Block Enable Clear Register	4000E204h
Interrupt Aggregator	0	Block IRQ Vector Register	4000E208h
EC Register Bank	0	TEST	4000FC00h
EC Register Bank	0	AHB Error Address Register	4000FC04h
EC Register Bank	0	TEST	4000FC08h
EC Register Bank	0	TEST	4000FC0Ch
EC Register Bank	0	TEST	4000FC10h
EC Register Bank	0	AHB Error Control Register	4000FC14h
EC Register Bank	0	Interrupt Control Register	4000FC18h

Block	Instance	Register	Register Address
EC Register Bank	0	ETM TRACE Enable Register	4000FC1Ch
EC Register Bank	0	Debug Enable Register	4000FC20h
EC Register Bank	0	OTP Lock Register	4000FC24h
EC Register Bank	0	WDT Event Count Register	4000FC28h
EC Register Bank	0	TEST	4000FC2Ch
EC Register Bank	0	TEST	4000FC30h
EC Register Bank	0	TEST	4000FC34h
EC Register Bank	0	Reserved	4000FC38h
EC Register Bank	0	TEST	4000FC3Ch
EC Register Bank	0	TEST	4000FC40h
EC Register Bank	0	TEST	4000FC44h
EC Register Bank	0	TEST	4000FC5Ch
EC Register Bank	0	TEST	4000FC60h
EC Register Bank	0	GPIO Bank Power Register	4000FC64h
EC Register Bank	0	TEST	4000FC68h
EC Register Bank	0	TEST	4000FC6Ch
EC Register Bank	0	JTAG Master Configuration Register	4000FC70h
EC Register Bank	0	JTAG Master Status Register	4000FC74h
EC Register Bank	0	JTAG Master TDO Register	4000FC78h
EC Register Bank	0	JTAG Master TDI Register	4000FC7Ch
EC Register Bank	0	JTAG Master TMS Register	4000FC80h
EC Register Bank	0	JTAG Master Command Register	4000FC84h
Power Clocks and Resets	0	System Sleep Control Register	40080100h
Power Clocks and Resets	0	Processor Clock Control Register	40080104h
Power Clocks and Resets	0	Slow Clock Control Register	40080108h
Power Clocks and Resets	0	Oscillator ID Register	4008010Ch
Power Clocks and Resets	0	PCR Power Reset Status Register	40080110h
Power Clocks and Resets	0	Power Reset Control Register	40080114h
Power Clocks and Resets	0	System Reset Register	40080118h
Power Clocks and Resets	0	TEST	4008011Ch
Power Clocks and Resets	0	TEST	40080120h
Power Clocks and Resets	0	Sleep Enable 0 Register	40080130h
Power Clocks and Resets	0	Sleep Enable 1 Register	40080134h
Power Clocks and Resets	0	Sleep Enable 2 Register	40080138h
Power Clocks and Resets	0	Sleep Enable 3 Register	4008013Ch
Power Clocks and Resets	0	Sleep Enable 4 Register	40080140h
Power Clocks and Resets	0	Clock Required 0 Register	40080150h
Power Clocks and Resets	0	Clock Required 1 Register	40080154h
Power Clocks and Resets	0	Clock Required 2 Register	40080158h
Power Clocks and Resets	0	Clock Required 3 Register	4008015Ch
Power Clocks and Resets	0	Clock Required 4 Register	40080160h
Power Clocks and Resets	0	Reset Enable 0 Register	40080170h
Power Clocks and Resets		Reset Enable 1 Register	40080174h

Block	Instance	Register	Register Address
Power Clocks and Resets		Reset Enable 2 Register	40080178h
Power Clocks and Resets		Reset Enable 3 Register	4008017Ch
Power Clocks and Resets		Reset Enable 4 Register	40080180h
GPIO	0	GPIO001 Pin Control Register	40081004h
GPIO	0	GPIO002 Pin Control Register	40081008h
GPIO	0	GPIO003 Pin Control Register	4008100Ch
GPIO	0	GPIO004 Pin Control Register	40081010h
GPIO	0	GPIO007 Pin Control Register	4008101Ch
GPIO	0	GPIO010 Pin Control Register	40081020h
GPIO	0	GPIO012 Pin Control Register	40081028h
GPIO	0	GPIO013 Pin Control Register	4008102Ch
GPIO	0	GPIO016 Pin Control Register	40081038h
GPIO	0	GPIO017 Pin Control Register	4008103Ch
GPIO	0	GPIO020 Pin Control Register	40081040h
GPIO	0	GPIO021 Pin Control Register	40081044h
GPIO	0	GPIO026 Pin Control Register	40081058h
GPIO	0	GPIO027 Pin Control Register	4008105Ch
GPIO	0	GPIO030 Pin Control Register	40081060h
GPIO	0	GPIO031 Pin Control Register	40081064h
GPIO	0	GPIO032 Pin Control Register	40081068h
GPIO	0	GPIO034 Pin Control Register	40081070h
GPIO	0	GPIO036 Pin Control Register	40081078h
GPIO	0	GPIO040 Pin Control Register	40081080h
GPIO	0	GPIO045 Pin Control Register	40081094h
GPIO	0	GPIO046 Pin Control Register	40081098h
GPIO	0	GPIO047 Pin Control Register	4008109Ch
GPIO	0	GPIO050 Pin Control Register	400810A0h
GPIO	0	GPIO051 Pin Control Register	400810A4h
GPIO	0	GPIO053 Pin Control Register	400810ACh
GPIO	0	GPIO054 Pin Control Register	400810B0h
GPIO	0	GPIO055 Pin Control Register	400810B4h
GPIO	0	GPIO056 Pin Control Register	400810B8h
GPIO	0	GPIO104 Pin Control Register	40081110h
GPIO	0	GPIO105 Pin Control Register	40081114h
GPIO	0	GPIO107 Pin Control Register	4008111Ch
GPIO	0	GPIO112 Pin Control Register	40081128h
GPIO	0	GPIO113 Pin Control Register	4008112Ch
GPIO	0	GPIO120 Pin Control Register	40081140h
GPIO	0	GPIO121 Pin Control Register	40081144h
GPIO	0	GPIO122 Pin Control Register	40081148h
GPIO	0	GPIO124 Pin Control Register	40081150h
GPIO	0	GPIO125 Pin Control Register	40081154h
GPIO	0	GPIO127 Pin Control Register	4008115Ch

Block	Instance	Register	Register Address
GPIO	0	GPIO134 Pin Control Register	40081170h
GPIO	0	GPIO135 Pin Control Register	40081174h
GPIO	0	GPIO140 Pin Control Register	40081180h
GPIO	0	GPIO145 Pin Control Register	40081194h
GPIO	0	GPIO146 Pin Control Register	40081198h
GPIO	0	GPIO147 Pin Control Register	4008119Ch
GPIO	0	GPIO150 Pin Control Register	400811A0h
GPIO	0	GPIO154 Pin Control Register	400811B0h
GPIO	0	GPIO155 Pin Control Register	400811B4h
GPIO	0	GPIO156 Pin Control Register	400811B8h
GPIO	0	GPIO157 Pin Control Register	400811BCh
GPIO	0	GPIO162 Pin Control Register	400811C8h
GPIO	0	GPIO163 Pin Control Register	400811CCh
GPIO	0	GPIO165 Pin Control Register	400811D4h
GPIO	0	GPIO170 Pin Control Register	400811E0h
GPIO	0	GPIO171 Pin Control Register	400811E4h
GPIO	0	GPIO200 Pin Control Register	40081200h
GPIO	0	GPIO201 Pin Control Register	40081204h
GPIO	0	GPIO202 Pin Control Register	40081208h
GPIO	0	GPIO203 Pin Control Register	4008120Ch
GPIO	0	GPIO204 Pin Control Register	40081210h
GPIO	0	GPIO223 Pin Control Register	4008124Ch
GPIO	0	GPIO224 Pin Control Register	40081250h
GPIO	0	GPIO225 Pin Control Register	40081254h
GPIO	0	GPIO227 Pin Control Register	4008125Ch
GPIO	0	Input GPIO[000:036]	40081300h
GPIO	0	Input GPIO[040:076]	40081304h
GPIO	0	Input GPIO[100:127]	40081308h
GPIO	0	Input GPIO[140:176]	4008130Ch
GPIO	0	Input GPIO[200:236]	40081310h
GPIO	0	Output GPIO[000:036]	40081380h
GPIO	0	Output GPIO[040:076]	40081384h
GPIO	0	Output GPIO[100:127]	40081388h
GPIO	0	Output GPIO[140:176]	4008138Ch
GPIO	0	Output GPIO[200:236]	40081390h
GPIO	0	GPIO001 Pin Control 2 Register	40081504h
GPIO	0	GPIO002 Pin Control 2 Register	40081508h
GPIO	0	GPIO003 Pin Control 2 Register	4008150Ch
GPIO	0	GPIO004 Pin Control 2 Register	40081510h
GPIO	0	GPIO007 Pin Control 2 Register	4008151Ch
GPIO	0	GPIO010 Pin Control 2 Register	40081520h
GPIO	0	GPIO012 Pin Control 2 Register	40081528h
GPIO	0	GPIO013 Pin Control 2 Register	4008152Ch

Block	Instance	Register	Register Address
GPIO	0	GPIO016 Pin Control 2 Register	40081538h
GPIO	0	GPIO017 Pin Control 2 Register	4008153Ch
GPIO	0	GPIO020 Pin Control 2 Register	40081540h
GPIO	0	GPIO021 Pin Control 2 Register	40081544h
GPIO	0	GPIO022 Pin Control 2 Register	40081548h
GPIO	0	GPIO026 Pin Control 2 Register	40081558h
GPIO	0	GPIO027 Pin Control 2 Register	4008155Ch
GPIO	0	GPIO030 Pin Control 2 Register	40081560h
GPIO	0	GPIO031 Pin Control 2 Register	40081564h
GPIO	0	GPIO032 Pin Control 2 Register	40081568h
GPIO	0	GPIO034 Pin Control 2 Register	40081570h
GPIO	0	GPIO036 Pin Control 2 Register	40081578h
GPIO	0	GPIO040 Pin Control 2 Register	40081580h
GPIO	0	GPIO045 Pin Control 2 Register	40081594h
GPIO	0	GPIO046 Pin Control 2 Register	40081598h
GPIO	0	GPIO047 Pin Control 2 Register	4008159Ch
GPIO	0	GPIO050 Pin Control 2 Register	400815A0h
GPIO	0	GPIO051 Pin Control 2 Register	400815A4h
GPIO	0	GPIO053 Pin Control 2 Register	400815ACh
GPIO	0	GPIO054 Pin Control 2 Register	400815B0h
GPIO	0	GPIO055 Pin Control 2 Register	400815B4h
GPIO	0	GPIO056 Pin Control 2 Register	400815B8h
GPIO	0	GPIO104 Pin Control 2 Register	40081610h
GPIO	0	GPIO105 Pin Control 2 Register	40081614h
GPIO	0	GPIO107 Pin Control 2 Register	4008161Ch
GPIO	0	GPIO112 Pin Control 2 Register	40081628h
GPIO	0	GPIO113 Pin Control 2 Register	4008162Ch
GPIO	0	GPIO120 Pin Control 2 Register	40081640h
GPIO	0	GPIO121 Pin Control 2 Register	40081644h
GPIO	0	GPIO122 Pin Control 2 Register	40081648h
GPIO	0	GPIO124 Pin Control 2 Register	40081650h
GPIO	0	GPIO125 Pin Control 2 Register	40081654h
GPIO	0	GPIO127 Pin Control 2 Register	4008165Ch
GPIO	0	GPIO134 Pin Control 2 Register	40081670h
GPIO	0	GPIO135 Pin Control 2 Register	40081674h
GPIO	0	GPIO140 Pin Control 2 Register	40081680h
GPIO	0	GPIO145 Pin Control 2 Register	40081694h
GPIO	0	GPIO146 Pin Control 2 Register	40081698h
GPIO	0	GPIO147 Pin Control 2 Register	4008169Ch
GPIO	0	GPIO150 Pin Control 2 Register	400816A0h
GPIO	0	GPIO154 Pin Control 2 Register	400816B0h
GPIO	0	GPIO155 Pin Control 2 Register	400816B4h
GPIO	0	GPIO156 Pin Control 2 Register	400816B8h

Block	Instance	Register	Register Address
GPIO	0	GPIO157 Pin Control 2 Register	400816BCh
GPIO	0	GPIO162 Pin Control 2 Register	400816C8h
GPIO	0	GPIO163 Pin Control 2 Register	400816CCh
GPIO	0	GPIO165 Pin Control 2 Register	400816D4h
GPIO	0	GPIO166 Pin Control 2 Register	400816D8h
GPIO	0	GPIO170 Pin Control 2 Register	400816E0h
GPIO	0	GPIO171 Pin Control 2 Register	400816E4h
GPIO	0	GPIO200 Pin Control 2 Register	40081700h
GPIO	0	GPIO201 Pin Control 2 Register	40081704h
GPIO	0	GPIO202 Pin Control 2 Register	40081708h
GPIO	0	GPIO203 Pin Control 2 Register	4008170Ch
GPIO	0	GPIO204 Pin Control 2 Register	40081710h
GPIO	0	GPIO223 Pin Control 2 Register	4008174Ch
GPIO	0	GPIO224 Pin Control 2 Register	40081750h
GPIO	0	GPIO225 Pin Control 2 Register	40081754h
GPIO	0	GPIO227 Pin Control 2 Register	4008175Ch
eFuse	0	Control Register	40082000h
eFuse	0	Manual Control Register	40082004h
eFuse	0	Manual Mode Address Register	40082006h
eFuse	0	Manual Mode Data Register	4008200Ch
eFuse	0	eFUSE Memory	40082010h
UART	0	Receive Buffer Register	400F2400h
UART	0	Transmit Buffer Register	400F2400h
UART	0	Programmable Baud Rate Generator LSB Register	400F2400h
UART	0	Programmable Baud Rate Generator MSB Register	400F2401h
UART	0	Interrupt Enable Register	400F2401h
UART	0	FIFO Control Register	400F2402h
UART	0	Interrupt Identification Register	400F2402h
UART	0	Line Control Register	400F2403h
UART	0	Modem Control Register	400F2404h
UART	0	Line Status Register	400F2405h
UART	0	Modem Status Register	400F2406h
UART	0	Scratchpad Register	400F2407h
UART	0	Activate Register	400F2730h
UART	0	Configuration Select Register	400F27F0h
UART	1	Receive Buffer Register	400F2800h
UART	1	Transmit Buffer Register	400F2800h
UART	1	Programmable Baud Rate Generator LSB Register	400F2800h
UART	1	Programmable Baud Rate Generator MSB Register	400F2801h
UART	1	Interrupt Enable Register	400F2801h

Block	Instance	Register	Register Address
UART	1	FIFO Control Register	400F2802h
UART	1	Interrupt Identification Register	400F2802h
UART	1	Line Control Register	400F2803h
UART	1	Modem Control Register	400F2804h
UART	1	Line Status Register	400F2805h
UART	1	Modem Status Register	400F2806h
UART	1	Scratchpad Register	400F2807h
UART	1	Activate Register	400F2B30h
UART	1	Configuration Select Register	400F2BF0h
Real Time Clock	0	Seconds Register	400F5000h
Real Time Clock	0	Seconds Alarm Register	400F5001h
Real Time Clock	0	Minutes Register	400F5002h
Real Time Clock	0	Minutes Alarm Register	400F5003h
Real Time Clock	0	Hours Register	400F5004h
Real Time Clock	0	Hours Alarm Register	400F5005h
Real Time Clock	0	Day of Week Register	400F5006h
Real Time Clock	0	Day of Month Register	400F5007h
Real Time Clock	0	Month Register	400F5008h
Real Time Clock	0	Year Register	400F5009h
Real Time Clock	0	Register A	400F500Ah
Real Time Clock	0	Register B	400F500Bh
Real Time Clock	0	Register C	400F500Ch
Real Time Clock	0	Register D	400F500Dh
Real Time Clock	0	Reserved	400F500Eh
Real Time Clock	0	Reserved	400F500Fh
Real Time Clock	0	RTC Control Register	400F5010h
Real Time Clock	0	Week Alarm Register	400F5014h
Real Time Clock	0	Daylight Savings Forward Register	400F5018h
Real Time Clock	0	Daylight Savings Backward Register	400F501Ch
Real Time Clock	0	TEST	400F5020h
Global Configuration	0	Global Configuration Reserved	400FFF00h
Global Configuration	0	TEST	400FFF07h
Global Configuration	0	Device ID	400FFF20h
Global Configuration	0	Revision	400FFF21h
Global Configuration	0	TEST	400FFF24h
Global Configuration	0	TEST	400FFF28h
Global Configuration	0	TEST	400FFF29h
Global Configuration	0	TEST	400FFF2Ch

4.0 CHIP CONFIGURATION

4.1 Introduction

The Global Configuration Registers are use for chip-level configuration. The chip's Device ID and Revision are located in the Global Configuration space and may be used to uniquely identify this chip.

4.2 Configuration Registers

The EC address for each register is formed by adding the Base Address for Global Configuration block shown in the Block Overview and Base Address Table in Section 3.0, "Device Inventory" to the offset shown in the "Offset" column.

TABLE 4-1: CHIP-LEVEL (GLOBAL) CONTROL/CONFIGURATION REGISTERS

Register	Host Offset	Description
Chip (Global) Control Registers		
TEST	02h	TEST. This register location is reserved for Microchip use. Modifying this location may cause unwanted results.
Reserved	03h - 06h	Reserved - Writes are ignored, reads return 0.
TEST	07h	TEST. This register location is reserved for Microchip use. Modifying this location may cause unwanted results.
Reserved	08h - 1Fh	Reserved - Writes are ignored, reads return 0.
Device ID	20h	A read-only register which provides device identification.
Device Revision	21h	A read-only register which provides device revision information. Bits[7:0] = current revision when read
TEST	22h - 23h	TEST. This register locations are reserved for Microchip use. Modifying these locations may cause unwanted results.
Reserved	24h	Reserved – writes are ignored, reads return "0".
TEST	25h - 2Fh	TEST. This register locations are reserved for Microchip use. Modifying these locations may cause unwanted results.

Note: Device ID for Device Revision values are in Table 4-1. Refer to errata sheet for the Device Revision value.

5.0 POWER, CLOCKS, AND RESETS

5.1 Introduction

The Power, Clocks, and Resets (PCR) chapter identifies all the power supplies, clock sources, and reset inputs to the chip and defines all the derived power, clock, and reset signals. In addition, this section identifies Power, Clock, and Reset events that may be used to generate an interrupt event, as well as, the Chip Power Management Features.

5.2 References

No references have been cited for this chapter.

5.3 Interrupts

The Power, Clocks, and Resets logic generates no events

5.4 Power

TABLE 5-1: POWER SOURCE DEFINITIONS

Power Well	Nominal Voltage	Description	Source
VTR_REG	1.8V - 3.3V	This supply is used to derive the chip's core power.	Pin Interface
VTR_ANALOG	3.3V	3.3V Analog Power Supply.	Pin Interface
VTR_PLL	3.3V	3.3V Power Supply for the 48MHz PLL. This must be connected to the same supply as VTR_ANALOG.	Pin Interface
VTR1	3.3V or 1.8V	Variable voltage I/O Power Supply. Power supply for a bank of I/O pins. See Note 1. This supply must be 3.3V if an EEPROM is included.	Pin Interface
VTR2	3.3V or 1.8V	Variable voltage I/O Power Supply. Power supply for a bank of I/O pins. See Note 1.	Pin Interface
VTR	1.2V	The main power well for internal logic	Internal regulator
VBAT	3.0V - 3.3V	System Battery Back-up Power Well. This is the "coin-cell" battery. GPIOs that share pins with VBAT signals are powered by this supply.	Pin Interface VBAT
VSS	0V	Digital Ground	Pin Interface
Note 1: See Section 5.4.1, "I/O Rail Requirements" for connection requirements for VTRx			

5.4.1 I/O RAIL REQUIREMENTS

All pins are powered by three power supply pins: VBAT, VTR1, VTR2. The VBAT supply must be 3V to 3.6V maximum, as shown in the following section. The VTRx pins, however, may be connected to either a 3.3V or a 1.8V power supply. The device must be able to determine the voltage when the internal RESET_SYS is de-asserted in order to configure the pins properly. The device remains in reset until VTR_ANALOG has ramped to 3.3V and VTR_REG has ramped to at least 1.6V.

Software can determine whether a VTRx region is 1.8V or 3.3V by examining the GPIO Bank Power Register.

If a power rail is not powered and stable when RESET_SYS is de-asserted and is not required for booting, software can configure the pins on that bank appropriately by setting the corresponding bit in the GPIO Bank Power Register, once software can determine that the power supply is up and stable. All GPIOs in the bank must be left in their default state and not modified until the Bank Power is configured properly.

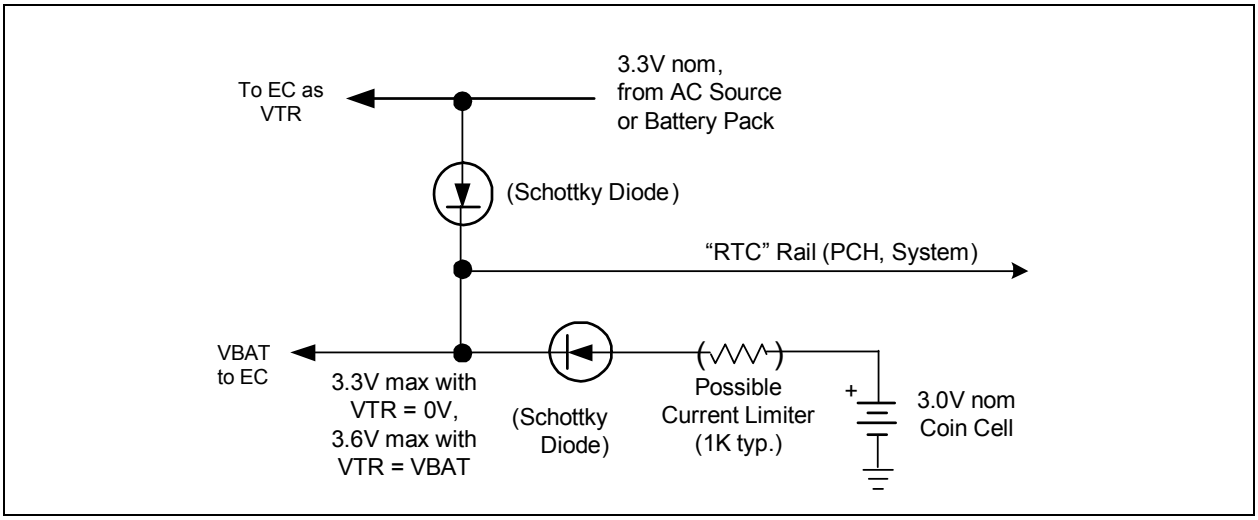
5.4.2 BATTERY CIRCUIT REQUIREMENTS

VBAT must always be present if VTR_ANALOG is present.

Microchip recommends removing all power sources to the device defined in Table 5-1, "Power Source Definitions" and all external voltage references defined in Table 5-2, "Voltage Reference Definitions" before removing and replacing the battery. In addition, upon removing the battery, discharge the battery pin before replacing the battery.

The following external circuit is recommended to fulfill this requirement:

FIGURE 5-1: RECOMMENDED BATTERY CIRCUIT



5.4.3 VOLTAGE REFERENCES

Table 5-2 lists the External Voltage References to which the CEC1702 provides high impedance interfaces.

TABLE 5-2: VOLTAGE REFERENCE DEFINITIONS

Power Well	Nominal Input Voltage	Scaling Ratio	Nominal Monitored Voltage	Description	Source
VREF_ADC	Variable	n/a	Variable	ADC Reference Voltage	Pin Interface

5.5 Clocks

The following section defines the clocks that are generated and derived.

5.5.1 RAW CLOCK SOURCES

The table defines raw clocks that are either generated externally or via an internal oscillator.

TABLE 5-3: SOURCE CLOCK DEFINITIONS

Clock Name	Frequency	Description	Source
32KHZ_IN	32.768 kHz (nominal)	Single-ended external clock input pin	32KHZ_IN pin
32.768 kHz Crystal Oscillator	32.768 kHz	A 32.768 kHz parallel resonant crystal connected between the XTAL1 and XTAL2 pins. The accuracy of the clock depends on the accuracy of the crystal and the characteristics of the analog components used as part of the oscillator. The crystal oscillator source can bypass the crystal with a single-ended clock input. This option is configured with the Clock Enable Register.	Pin Interface (XTAL1 and XTAL2) When used singled-ended, pin XTAL2
32.768 kHz Silicon Oscillator	32.768 kHz	32.768 kHz low power Internal Oscillator. The frequency is 32.768KHz ±2%.	Internal Oscillator powered by VBAT
32 MHz Ring Oscillator	32MHz	The 32MHz Ring Oscillator is used to supply a clock for the 48MHz main clock domain while the 48MHz PLL is not locked. Its frequency can range from 16Mhz to 48MHz.	Powered by VTR.
48 MHz PLL	48MHz	The 48 MHz Phase Locked Loop generates a 48MHz clock locked to the Always-on Internal 32KHz Clock Source.	Powered by VTR. May be stopped by Chip Power Management Features.

5.5.2 CLOCK DOMAINS

TABLE 5-4: CLOCK DOMAIN DEFINITIONS

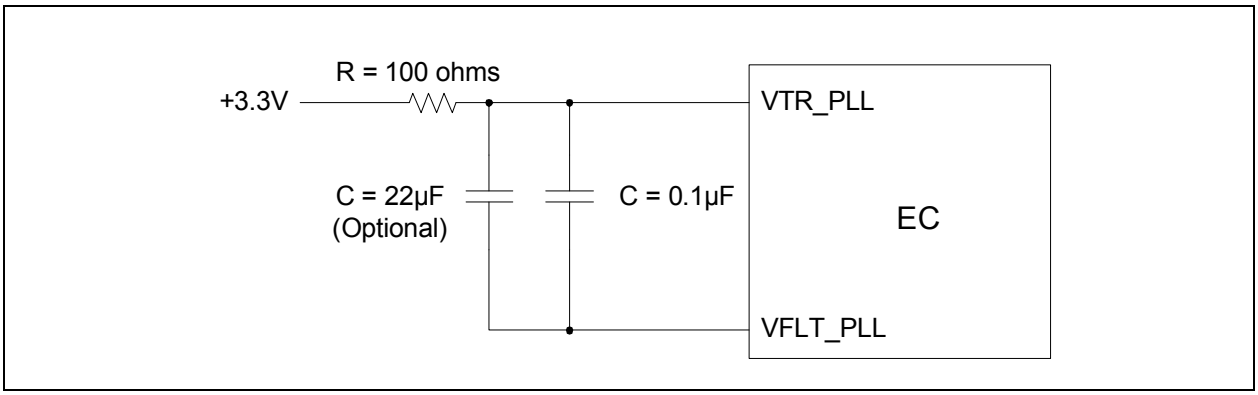
Clock Domain	Description
32KHz	The clock source used by internal blocks that require an always-on low speed clock.
48MHz	The main clock source used by most internal blocks.
96MHz	The clock source used by the Public Key Cryptographic Engine. It is derived from the same PLL as the 48MHz clock source.
100KHz	A low-speed clock derived from the 48MHz clock domain. Used as a time base for PWMs and Tachs.
EC_CLK	The clock used by the EC processor. The frequency is determined by the Processor Clock Control Register.

5.5.3 48MHZ PLL

The 48MHz clock domain is primarily driven by a 48MHz PLL, which derives 48MHz from the 32KHz always-on clock domain. In Heavy Sleep mode, the 48MHz PLL is shut off. When the PLL is started, either from waking from the Heavy Sleep mode, or after a Power On Reset, the 32MHz ring oscillator becomes the clock source for the 48MHz clock domain until the PLL is stable. The PLL becomes stable after about 3ms; until that time, the 48MHz clock domain may range from 16MHz to 48MHz, as this is the accuracy range of the 32MHz ring.

The PLL requires its own power 3.3V power supply, VTR_PLL. This power rail must be active and stable no later than the latest of VTR_REG and VTR_ANALOG. There is no hardware detection of VTR_PLL power good in the reset generator. The VTR_PLL supply must be filtered, as shown in Figure 5-2, "Power Supply Filtering for PLL". There is no special ground connection required for the PLL.

FIGURE 5-2: POWER SUPPLY FILTERING FOR PLL



5.5.4 32KHZ CLOCK SWITCHING

The 32KHz Clock Domain may be sourced by a crystal oscillator, using an external crystal, by an internal 32KHz oscillator, or from a single-ended clock input. The external single-ended clock source can itself be sourced either from the 32KHZ_IN signal that is a GPIO alternate function or from the XTAL2 crystal pin. The Clock Enable Register is used to configure the source for the 32 kHz clock domain.

When VTR is off, the 32 kHz clock domain can be disabled, for lowest standby power, or it can be kept running in order to provide a clock for the Real Time Clock or the Week Timer.

An external single-ended clock input for 32KHZ_IN may be supplied by any accurate 32KHz clock source in the system. The SUSCLK output from the chipset may be used as the 32KHz source whenever RSMRST# is de-asserted. See chipset documentation for details on the use of SUSCLK.

If firmware switches the 32KHz clock source, the 48MHz PLL will be shut off and then restarted. The 48MHz clock domain will become unlocked and be sourced from the 32 MHz Ring Oscillator until the 48MHz PLL is on and locked.

5.5.4.1 Always-on Internal 32KHz Clock Source

The 32Khz clock domain can be driven from an internal 32KHz clock source that is always on. This clock source is used to drive the 48MHz PLL and remains on at all times, even when an external input is selected as the source. The internal source provides a reference for the Activity Detect that monitors the external clock input, as well as providing a low latency backup clock source when the Activity Detector cannot detect a clock on the external input.

The Always-on 32KHz Internal Clock Source can be driven either by the 32.768 kHz Silicon Oscillator or the 32.768 kHz Crystal Oscillator.

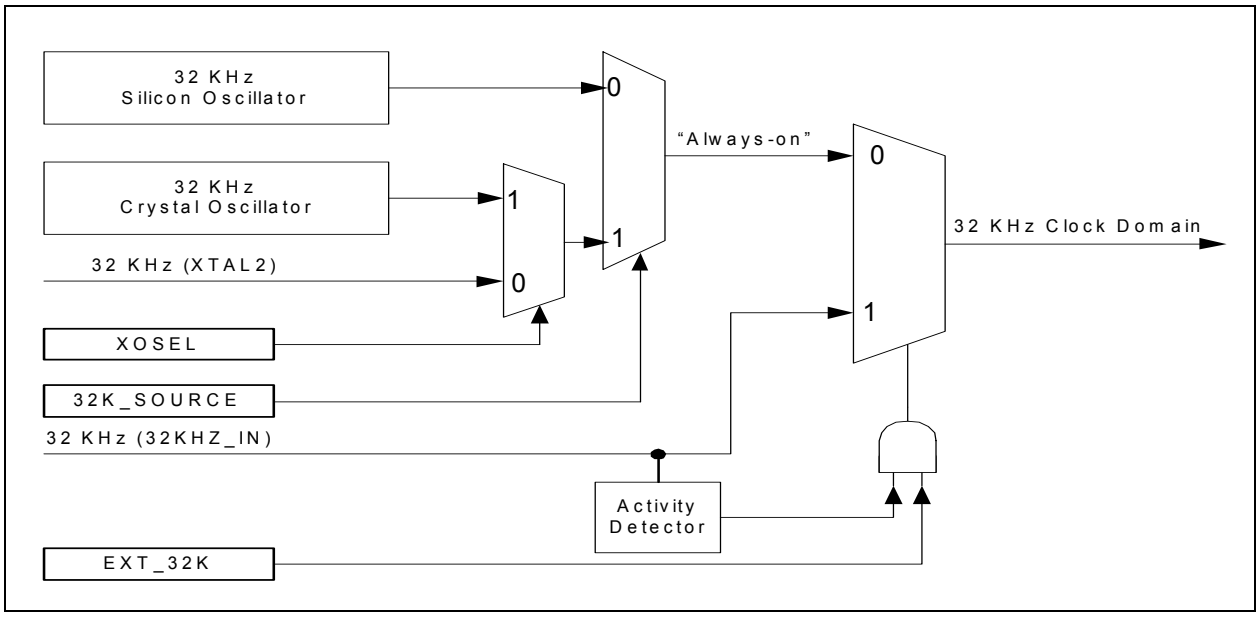
Note:	If the 32KHZ_SOURCE field in the Clock Enable Register selects the crystal oscillator as the source for the always-on clock source, and the XOSEL field selects a single-ended input for the crystal oscillator, the system must ensure that the single-ended input remains on at all times. The Activity Detector will not monitor the single-ended input to the crystal oscillator.

5.5.4.2 External 32KHz Clock Activity Detector

When the EXT_32K field in the Clock Enable Register is set for an external clock source an Activity Detector monitors the external 32KHz signal at all times. If there is no clock detected on the pin, the 32KHZ clock domain is switched to the internal 32KHz silicon oscillator. If a clock is again detected on the pin, the 32KHz clock domain is switched to the pin

The following figure illustrates the 32KHz clock domain sourcing.

FIGURE 5-3: 32KHZ ACTIVITY DETECTOR



Note:	Once the internal 32KHz clock domain switches to an external single-ended clock source, the external source **must** remain active until VTR power is removed, or internal clocking may not function correctly.

5.5.4.3 32KHz Crystal Oscillator

If the 32KHz source will never be the crystal oscillator, then the XTAL2 pin should be grounded. The XTAL1 pin should be left unconnected.

5.6 Resets

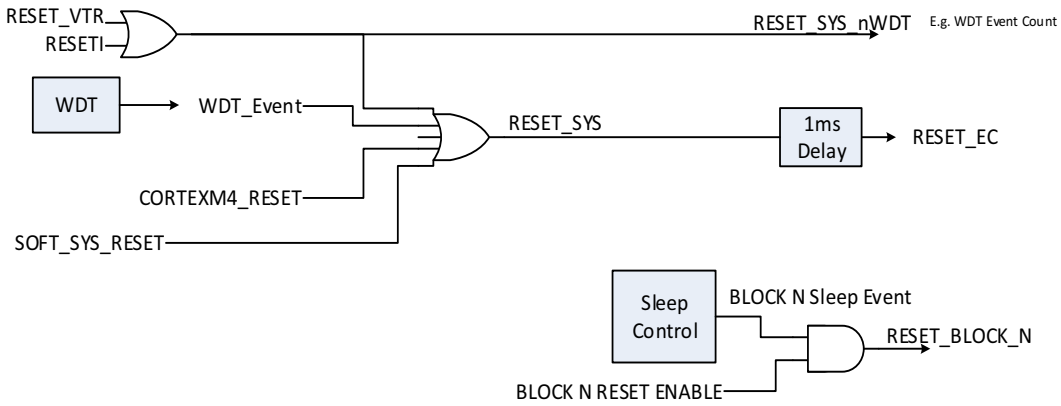
TABLE 5-5: DEFINITION OF RESET SIGNALS

Reset	Description	Source
RESET_VBAT	Internal VBAT Reset signal. This signal is used to reset VBAT powered registers.	RESET_VBAT is a pulse that is asserted at the rising edge of VTR power if the VBAT voltage is below a nominal 1.25V. RESET_VBAT is also asserted as a level if, while VTR power is not present, the coin cell is replaced with a new cell that delivers at least a nominal 1.25V. In this latter case RESET_VBAT is de-asserted when VTR power is applied. No action is taken if the coin cell is replaced, or if the VBAT voltage falls below 1.25 V nominal, while VTR power is present.
RESET_VTR	Internal VTR Reset signal.	This internal reset signal is asserted as long as the reset generator determines that the output of the internal regulator is stable at its target voltage and that the voltage rail supplying the main clock PLL is at 3.3V. Although most VTR-powered registers are reset on RESET_SYS, some registers are only reset on this reset.

TABLE 5-5: DEFINITION OF RESET SIGNALS (CONTINUED)

Reset	Description	Source
RESET_SYS	Internal Reset signal. This signal is used to reset VTR powered registers.	RESET_SYS is the main global reset signal. This reset signal will be asserted if: • RESET_VTR is asserted • The RESETI# pin asserted • A WDT Event event is asserted • A soft reset is asserted by the SOFT_SYS-_RESET bit in the System Reset Register • ARM M4 SYSRESETREQ
WDT Event	A WDT Event generates the RESET_SYS event. This signal resets VTR powered registers with the exception of the WDT Event Count Register register. Note that the glitch protect circuits do not activate on a WDT reset. WDT Event does not reset VBAT registers or logic.	This reset signal will be asserted if: • A WDT Event event is asserted This event is indicated by the WDT bit in the Power-Fail and Reset Status Register
RESET_SYS_nWDT	Internal Reset signal. This signal is used to reset VTR powered registers not effected by a WDT Event A RESET_SYS_nWDT is used to reset registers that need to be preserved through a WDT Event like a WDT Event Count Register.	This reset signal will be asserted if: • RESET_VTR is asserted • The RESETI# pin asserted
RESET_EC	Internal reset signal to reset the processor in the EC Subsystem.	This reset is a stretched version of RESET_SYS. This reset asserts at the same time that RESET_SYS asserts and is held asserted for 1ms after RESET_SYS deasserts.
RESET_BLOCK_N	Each IP block in the device may be configured to be reset when it enters the Low Power Mode referred to as SLEEP.	This reset signal will be asserted if Block N SLEEP_ENABLE and Block N RESET_ENABLE are all set to 1, Block N CLOCK_REQUIRED signal is low, and Block N Enters Sleep.

FIGURE 5-4: RESETS BLOCK DIAGRAM (CEC1702)



Note 1: SOFT_SYS_RESET is implemented in bit[8] of the System Reset Register

5.7 Chip Power Management Features

This device is designed to always operate in its lowest power state during normal operation. In addition, this device offers additional programmable options to put individual logical blocks to sleep as defined in the following section, Section 5.7.1.

5.7.1 BLOCK LOW POWER MODES

All power related control signals are generated and monitored centrally in the chip's Power, Clocks, and Resets (PCR) block. The power manager of the PCR block uses a sleep interface to communicate with all the blocks. The sleep interface consists of three signals:

- **SLEEP_ENABLE (request to sleep the block)** is generated by the PCR block. A group of SLEEP_ENABLE signals are generated for every clock segment. Each group consists of a SLEEP_ENABLE signal for every block in that clock segment.
- **CLOCK_REQUIRED (request clock on)** is generated by every block. They are grouped by blocks on the same clock segment. The PCR monitors these signals to see when it can gate off clocks.
- **RESET_ENABLE (reset on sleep)** bits determine if the block (including registers) will be reset when it enters sleep mode.

A block can always drive CLOCK_REQUIRED low synchronously, but it *must* drive it high asynchronously since its internal clocks are gated and it has to assume that the clock input itself is gated. Therefore the block can only drive CLOCK_REQUIRED high as a result of a register access or some other input signal.

The following table defines a block's power management protocol:

TABLE 5-6: POWER MANAGEMENT PROTOCOL

Power State	SLEEP_ENABLE	CLOCK_REQUIRED	Description
Normal operation	Low	Low	Block is idle and NOT requesting clocks. The block gates its own internal clock.
Normal operation	Low	High	Block is NOT idle and requests clocks.
Request sleep	Rising Edge	Low	Block is IDLE and enters sleep mode immediately. The block gates its own internal clock. The block cannot request clocks again until SLEEP_ENABLE goes low.

TABLE 5-6: POWER MANAGEMENT PROTOCOL

Power State	SLEEP_ENABLE	CLOCK_REQUIRED	Description
Request sleep	Rising Edge	High then Low	Block is not IDLE and will stop requesting clocks and enter sleep when it finishes what it is doing. This delay is block specific, but should be less than 1 ms. The block gates its own internal clock. After driving CLOCK_REQUIRED low, the block cannot request clocks again until SLEEP_ENABLE goes low.
Register Access	X	High	Register access to a block is always available regardless of SLEEP_ENABLE. Therefore the block ungates its internal clock and drives CLOCK_REQUIRED high during the access. The block will regate its internal clock and drive CLOCK_REQUIRED low when the access is done.

A wake event clears all SLEEP_ENABLE bits momentarily, and then returns the SLEEP_ENABLE bits back to their original state. The block that needs to respond to the wake event will do so.

The Sleep Enable, Clock Required and Reset Enable Registers are defined in Section 5.8.

5.7.2 CONFIGURING THE CHIP'S SLEEP STATES

The chip supports two sleep states: LIGHT SLEEP and HEAVY SLEEP. The chip will enter one of these two sleep states only when all the blocks have been commanded to sleep and none of them require a 48MHz clock source (i.e., all CLOCK_REQUIRED status bits are 0), and the processor has executed its sleep instruction. These sleep states must be selected by firmware via the System Sleep Control bits implemented in the System Sleep Control Register prior to issuing the sleep instruction. Table 5-8, "System Sleep Modes" defines each of these sleep states.

There are two ways to command the chip blocks to enter sleep.

1. Assert the SLEEP_ALL bit located in the System Sleep Control Register
2. Assert all the individual block sleep enable bits

Blocks will only enter sleep after their sleep signal is asserted and they no longer require the 48MHz source. Each block has a corresponding clock required status bit indicating when the block has entered sleep. The general operation is that a block will keep the 48MHz clock source on until it completes its current transaction. Once the block has completed its work, it deasserts its clock required signal. Blocks like timers, PWMs, etc. will de-assert their clock required signals immediately. See the individual block Low Power Mode sections to determine how each individual block enters sleep.

5.7.3 WAKING THE CHIP FROM SLEEPING STATE

The chip will remain in the configured sleep state until it detects either a wake event or a full VTR POR. A wake event occurs when a wake-capable interrupt is enabled and triggered. Interrupts that are not wake-capable cannot occur while the system is in LIGHT SLEEP or HEAVY SLEEP.

In LIGHT SLEEP, the 48MHz clock domain is gated off, but the 48 MHz PLL remains operational and locked to the 32KHz clock domain. On wake, the PLL output is ungated and the 48MHz clock domain starts immediately, with the PLL_LOCK bit in the Oscillator ID Register set to '1'. Any device that requires an accurate clock, such as a UART, me be used immediately on wake.

In HEAVY SLEEP, the 48 MHz PLL is shut down. On wake, the 32 MHz Ring Oscillator is used to provide a clock source for the 48MHz clock domain until the PLL locks to the 32KHz clock domain. The ring oscillator starts immediately on wake, so there is no latency for the EC to start after a wake, However, the ring oscillator is only accurate to ±50%, so any device that requires an accurate 48MHz clock will not operate correctly until the PLL locks.The time to lock latency for the PLL is shown in Table 5-8, "System Sleep Modes".

The SLEEP_ALL bit is automatically cleared when the processor responds to an interrupt. This applies to non-wake interrupts as well as wake interrupts, in the event an interrupt occurs between the time the processor issued a WAIT FOR INTERRUPT instruction and the time the system completely enters the sleep state.

5.7.3.1 Wake-Only Events

Some devices which respond to an external master require the 48MHz clock domain to operate but do not necessarily require and immediate processing by the EC. Wake-only events provide the means to start the 48MHz clock domain without triggering an EC interrupt service routine. This events are grouped into a single GIRQ, GIRQ22. Events that are

enabled in that GIRQ will start the clock domain when the event occurs, but will not invoke an EC interrupt. The SLEEP_ENABLE flags all remain asserted. If the activity for the event does not in turn trigger another EC interrupt, the CLOCK_REQUIRED for the block will re-assert and the configured sleep state will be re-entered.

The Wake-Only Events and interrupts are responsible for waking up the respective blocks from where the Event or interrupt becomes active, but will not enable the clock to the processor.

For example, when RSMRST is high and there is a desire to wake from an ESPI cycle, GIRQ22[9] is the correct wake source to use. When Chip is asleep and there is a ESPI cycle, the falling edge of the CS will cause the chips clock to turn on the ESPI block, but not the processor itself. Upon conclusion of the ESPI cycle, if no ESPI interrupt was generated (i.e. most cycles), then the clock to the ESPI block will go off, and the chip will go back to sleep. If the ESPI cycle creates an interrupt to the processor (i.e. downstream wire or downstream OOB packet for example), then an processor interrupt will be generated if enabled and the clock will remain on and the processor can service the interrupt and the processor can put the chip back to sleep when it has completed its work.

Note:	The ESPI Reset itself is NOT a wake event. If wake from ESPI reset is required, then the GPIO interrupt for the ESPI reset pin can be used as a wake event.

5.8 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for the Power, Clocks, and Resets Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 5-7: REGISTER SUMMARY

Offset	Name
0h	System Sleep Control Register
4h	Processor Clock Control Register
8h	Slow Clock Control Register
Ch	Oscillator ID Register
10h	PCR Power Reset Status Register
18h	System Reset Register
1Ch	TEST
20h	TEST
30h	Sleep Enable 0 Register
34h	Sleep Enable 1 Register
38h	Sleep Enable 2 Register
3Ch	Sleep Enable 3 Register
40h	Sleep Enable 4 Register
50h	Clock Required 0 Register
54h	Clock Required 1 Register
58h	Clock Required 2 Register
5Ch	Clock Required 3 Register
60h	Clock Required 4 Register
70h	Reset Enable 0 Register
74h	Reset Enable 1 Register
78h	Reset Enable 2 Register
7Ch	Reset Enable 3 Register
80h	Reset Enable 4 Register

Note:	All register addresses are naturally aligned on 32-bit boundaries. Offsets for registers that are smaller than 32 bits are reserved and must not be used for any other purpose.

The bit definitions for the Sleep Enable, Clock Required and Reset Enable Registers are defined in the Sleep Enable Register Assignments Table in Section 3.0, "Device Inventory".

5.9 Sleep Enable *n* Registers

5.9.1 SLEEP ENABLE *N* REGISTER FORMAT

Offset	See Sleep Enable Register Assignments Table in Section 3.0, "Device Inventory"			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	SLEEP_ENABLE 1=Block is commanded to sleep at next available moment 0=Block is free to use clocks as necessary Unassigned bits are reserved. They must be set to '1b' when written. When read, unassigned bits return the last value written.	R/W	0h	RESET_SYS

5.9.2 CLOCK REQUIRED *N* REGISTER FORMAT

Offset	See Sleep Enable Register Assignments Table in Section 3.0, "Device Inventory"			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	CLOCK_REQUIRED 1=Bock requires clocks 0=Block does not require clocks Unassigned bits are reserved and always return 0 when read.	R	0h	RESET_SYS

5.9.3 RESET ENABLE *N* REGISTER FORMAT

Note:	If a block is configured such that it is to be reset when it goes to sleep, then registers within the block may not be writable when the block is asleep.

Offset	See Sleep Enable Register Assignments Table in Section 3.0, "Device Inventory"			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	RESET_ENABLE 1=Bock will reset on sleep 0=Block will not reset on sleep Unassigned bits are reserved and always return 0 when read.	R/W	0h	RESET_SYS

5.9.4 SYSTEM SLEEP CONTROL REGISTER

Offset	0h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:4	Reserved	R	-	-
3	SLEEP_ALL By setting this bit to '1b' and then issuing a WAIT FOR INTERRUPT instruction, the EC can initiate the System Sleep mode. When no device requires the main system clock, the system enters the sleep mode defined by the field SLEEP_MODE. This bit is automatically cleared when the processor vectors to an interrupt. 1=Assert all sleep enables 0=Do not sleep all	R/W	0h	RESET_SYS
2	TEST Test bit. Should always be written with a '0b'.	R/W	0h	RESET_SYS
1	Reserved	R	-	-
0	SLEEP_MODE Sleep modes differ only in the time it takes for the 48MHz clock domain to lock to 48MHz. The wake latency in all sleep modes is 0ms. Table 5-8 shows the time to lock latency for the different sleep modes. 1=Heavy Sleep 0=Light Sleep	R/W	0h	RESET_SYS

TABLE 5-8: SYSTEM SLEEP MODES

SLEEP_MODE	Sleep State	Latency to Lock	Description
0	LIGHT SLEEP	0	Output of the PLL is gated in sleep. The PLL remains on.
1	HEAVY SLEEP	3ms	The PLL is shut down while in sleep.

5.9.5 PROCESSOR CLOCK CONTROL REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:8	Reserved	R	-	-
7:0	PROCESSOR_CLOCK_DIVIDE The following list shows examples of settings for this field and the resulting EC clock rate. 48=divide the 48MHz clock by 48 (1MHz processor clock) 16=divide the 48MHz clock by 16 (4MHz processor clock) 4=divide the 48MHz clock by 4 (12MHz processor clock) 3=divide the 48MHz clock by 3 (16MHz processor clock) 1=divide the 48MHz clock by 1 (48MHz processor clock) No other values are supported.	R/W	4h	RESET_SYS

5.9.6 SLOW CLOCK CONTROL REGISTER

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:10	Reserved	R	-	-
9:0	SLOW_CLOCK_DIVIDE Configures the 100KHz clock domain. n=Divide by n 0=Clock off The default setting is for 100KHz.	R/W	1E0h	RESET_SYS

5.9.7 OSCILLATOR ID REGISTER

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:9	Reserved	R	-	-
8	PLL_LOCK Phase Lock Loop Lock Status	R	0h	RESET_SYS
7:0	TEST	R	N/A	RESET_SYS

5.9.8 PCR POWER RESET STATUS REGISTER

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:12	Reserved	R	-	-
10	32K_ACTIVE This bit monitors the state of the 32K clock input. This status bit detects edges on the clock input but does not validate the frequency. 1=The 32K clock input is present. The internal 32K clock is derived from the pin and the ring oscillator is synchronized to the external 32K clock 0=The 32K clock input is not present. The internal 32K clock is derived from the ring oscillator	R	-	RESET_SYS
9:8	Reserved	R	-	-
7	JTAG_RST# Indicates the status of JTAG_RST# pin. 0 = JTAG_RST# pin is low 1 = JTAG_RST# pin is high	R	1h	RESET_SYS
6	RESET_SYS_STATUS Indicates the current value of RESET_SYS. The bit will not clear if a write 1 is attempted at the same time that a RESET_SYS occurs; this way a reset event is never missed. 1=A reset occurred 0=No reset occurred since the last time this bit was cleared	R/WC	1h	RESET_SYS
5	VBAT_RESET_STATUS Indicates the status of RESET_VBAT. The bit will not clear if a write of ‘1’b is attempted at the same time that a VBAT_RST_N occurs, this way a reset event is never missed. 1=A reset occurred 0=No reset occurred while VTR was off or since the last time this bit was cleared	R/WC	-	RESET_SYS
4	Reserved	R	-	-
1:0	Reserved	R	-	-
Note 1: This read-only status bit always reflects the current status of the event and is not affected by any Reset events.				

5.9.9 SYSTEM RESET REGISTER

Offset	18h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:9	Reserved	R	-	-
8	SOFT_SYS_RESET A write of a '1' to this bit will force an assertion of the RESET_SYS reset signal, resetting the device. A write of a '0' has no effect. Reads always return '0'.	W	-	-
7:0	Reserved	R	-	-

6.0 RAM AND ROM

6.1 References

None.

6.2 SRAM

The CEC1702 contains two blocks of SRAM. The two SRAM blocks in the CEC1702 total 480KB. Both SRAM blocks can be used for either program or data accesses. Performance is enhanced when program fetches and data accesses are to different SRAM blocks, but a program will operate correctly even if both program and data accesses are targeting the same block simultaneously.

- The first SRAM, which is optimized for code access, is 416KB
- The second SRAM, which is optimized for data access, is 64KB

6.3 ROM

The CEC1702 contains a 64KB block of ROM, located at address 00000000h in the ARM address space. The ROM contains boot code that is executed after the de-assertion of RESET_SYS. The boot code loads an executable code image into SRAM. The ROM also includes a set of API functions that can be used for cryptographic functions, as well as loading SRAM with programs or data.

6.4 Additional Memory Regions

6.4.1 ALIAS RAM

The Alias RAM region, starting at address 20000000h, is an alias of the SRAM located at 118000h, and is the same size as that SRAM block. EC software can access memory in either the primary address or in the alias region; however, access is considerably slower to the alias region. The alias region exists in order to enable the ARM bit-band region located at address 20000000h.

6.4.2 RAM BIT-BAND REGION

The RAM bit-band region is an alias of the SRAM located at 118000h, except that each bit is aliased to bit 0 of a 32-bit doubleword in the bit-band region. The upper 31 bits in each doubleword of the bit-band region are always 0. The bit-band region is therefore 32 times the size of the SRAM region. It can be used for atomic updates of individual bits of the SRAM, and is a feature of the ARM architecture.

The bit-band region can only be accessed by the ARM processor. Accesses by any other bus master will cause a memory fault.

6.4.3 CRYPTOGRAPHIC RAM

The cryptographic RAM is used by the cryptographic API functions in the ROM

6.4.4 REGISTER BIT-BAND REGION

The Register bit-band region is an 32-to-1 alias of the device register space starting at address 40000000h and ending with the Host register space at 400FFFFF. Every bit in the register space is aliased to a byte in the Register bit-band region, and like the RAM bit-band region, can be used by EC software to read and write individual register bits. Only the EC Device Registers and the GPIO Registers can be accessed via the bit-band region.

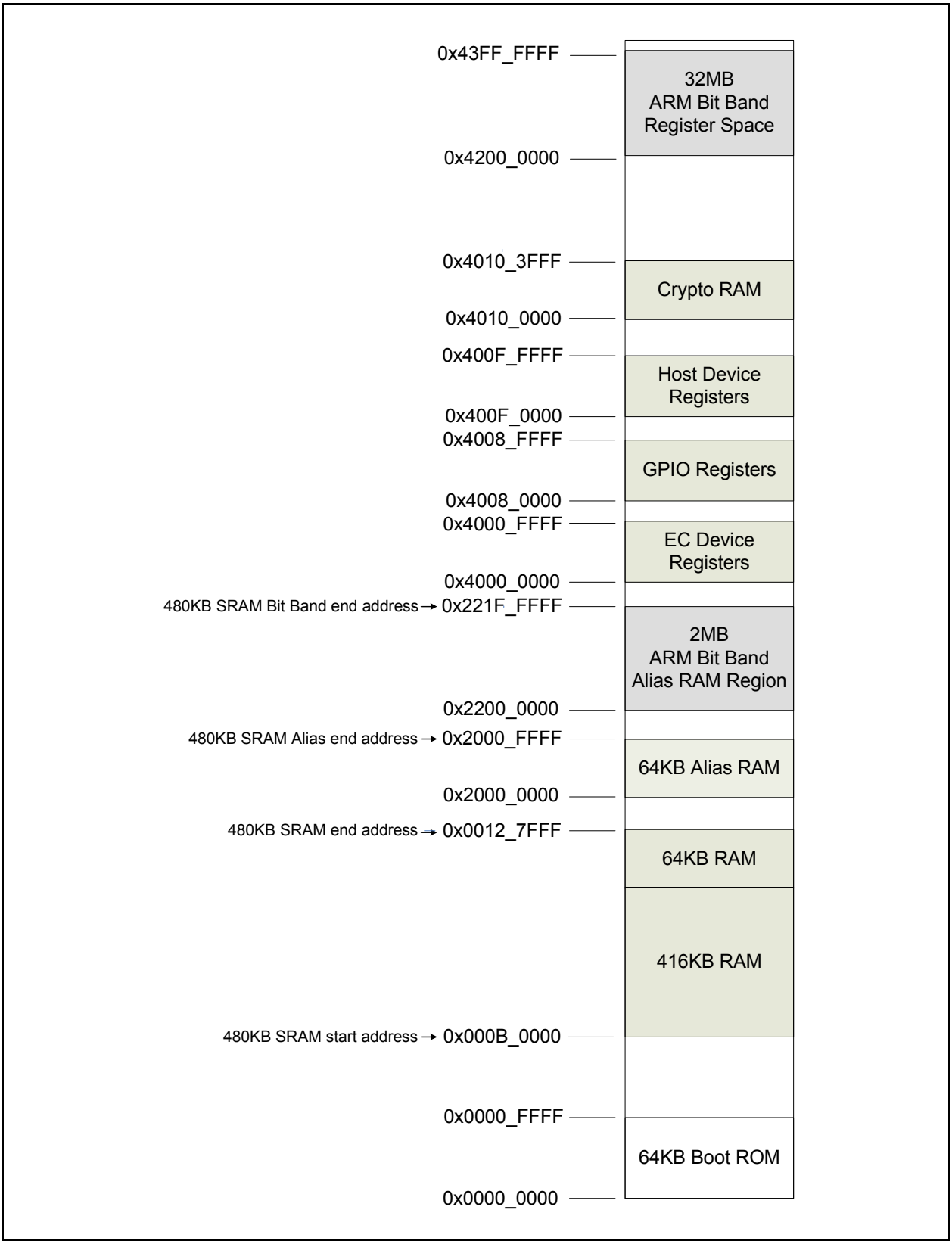
A one bit write operation to a register bit in the bit-band region is implemented by the ARM processor by performing a read, a bit modification, followed by a write back to the same register. Software must be careful when using bit-banding if a register contains bits have side effects triggered by a read.

The bit-band region can only be accessed by the ARM processor. Accesses by any other bus master will cause a memory fault.

6.5 Memory Map

The memory map of the RAM and ROM is represented as follows:

FIGURE 6-1: MEMORY LAYOUT



7.0 INTERNAL DMA CONTROLLER

7.1 Introduction

The Internal DMA Controller transfers data to/from the source from/to the destination. The firmware is responsible for setting up each channel. Afterwards either the firmware or the hardware may perform the flow control. The hardware flow control exists entirely inside the source device. Each transfer may be 1, 2, or 4 bytes in size, so long as the device supports a transfer of that size. Every device must be on the internal 32-bit address space.

7.2 References

No references have been cited for this chapter.

7.3 Terminology

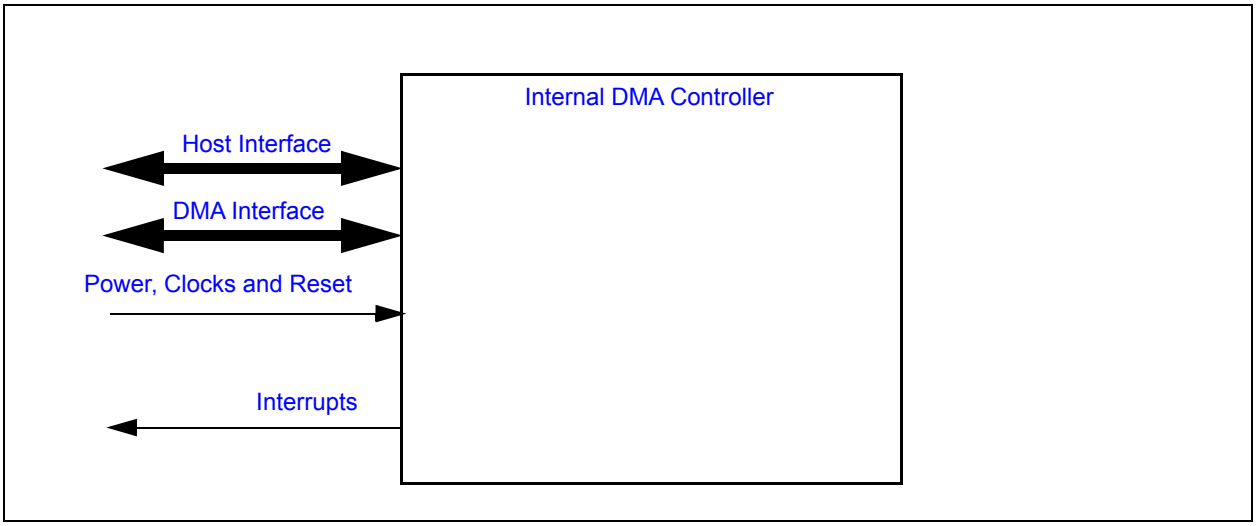
TABLE 7-1: TERMINOLOGY

Term	Definition
DMA Transfer	This is a complete **DMA Transfer** which is done after the **Master Device** terminates the transfer, the Firmware Aborts the transfer or the DMA reaches its transfer limit. A DMA Transfer may consist of one or more data packets.
Data Packet	Each data packet may be composed of 1, 2, or 4 bytes. The size of the data packet is limited by the max size supported by both the source and the destination. Both source and destination will transfer the same number of bytes per packet.
Channel	The Channel is responsible for end-to-end (source-to-destination) Data Packet delivery.
Device	A Device may refer to a Master or Slave connected to the DMA Channel. Each DMA Channel may be assigned one or more devices.
Master Device	This is the master of the DMA, which determines when it is active. The Firmware is the master while operating in Firmware Flow Control. The Hardware is the master while operating in Hardware Flow Control. The Master Device in Hardware Mode is selected by **DMA Channel Control:Hardware Flow Control Device**. It is the index of the **Flow Control** Port.
Slave Device	The Slave Device is defined as the device associated with the targeted Memory Address.
Source	The DMA Controller moves data from the Source to the Destination. The Source provides the data. The Source may be either the Master or Slave Controller.
Destination	The DMA Controller moves data from the Source to the Destination. The Destination receives the data. The Destination may be either the Master or Slave Controller.

7.4 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

FIGURE 7-1: INTERNAL DMA CONTROLLER I/O DIAGRAM



7.4.1 SIGNAL DESCRIPTION

This block doesn't have any external signals that may be routed to the pin interface. This DMA Controller is intended to be used internally to transfer large amounts of data without the embedded controller being actively involved in the transfer.

7.4.2 HOST INTERFACE

The registers defined for the Internal DMA Controller are accessible by the various hosts as indicated in Section 7.9, "EC Registers".

7.4.3 DMA INTERFACE

Each DMA Master Device that may engage in a DMA transfer must have a compliant DMA interface. The following table lists the DMA Devices in the CEC1702.

TABLE 7-2: DMA CONTROLLER DEVICE SELECTION

Device Name	Device Number (Note 1)	Controller Source
SMB-I2C 0 Controller	0	Slave
	1	Master
SMB-I2C 1 Controller	2	Slave
	3	Master
SMB-I2C 2 Controller	4	Slave
	5	Master
SMB-I2C 3 Controller	6	Slave
	7	Master
SPI 0 Controller	8	Transmit
	9	Receive
SPI 1 Controller	10	Transmit
	11	Receive

Note 1: The Device Number is programmed into field HARDWARE_FLOW_CONTROL_DEVICE of the DMA Channel N Control Register register.

TABLE 7-3: DMA CONTROLLER MASTER DEVICES SIGNAL LIST

Device Name	Dev Num	Device Signal Name	Direction	Description
SMB-I2C 0 Controller	0	SMB-I2C_SDMA_Req	INPUT	DMA request control from SMB-I2C Slave channel.
		SMB-I2C_SDMA_Term	INPUT	DMA termination control from SMB-I2C Slave channel.
		SMB-I2C_SDMA_Done	OUTPUT	DMA termination control from DMA Controller to Slave channel.
	1	SMB-I2C_MDMA_Req	INPUT	DMA request control from SMB-I2C Master channel.
		SMB-I2C_MDMA_Term	INPUT	DMA termination control from SMB-I2C Master channel.
		SMB-I2C_MDMA_Done	OUTPUT	DMA termination control from DMA Controller to Master channel.
SMB-I2C 1 Controller	2	SMB-I2C_SDMA_Req	INPUT	DMA request control from SMB-I2C Slave channel.
		SMB-I2C_SDMA_Term	INPUT	DMA termination control from SMB-I2C Slave channel.
		SMB-I2C_SDMA_Done	OUTPUT	DMA termination control from DMA Controller to Slave channel.
	3	SMB-I2C_MDMA_Req	INPUT	DMA request control from SMB-I2C Master channel.
		SMB-I2C_MDMA_Term	INPUT	DMA termination control from SMB-I2C Master channel.
		SMB-I2C_MDMA_Done	OUTPUT	DMA termination control from DMA Controller to Master channel.
SMB-I2C 2 Controller	4	SMB-I2C_SDMA_Req	INPUT	DMA request control from SMB-I2C Slave channel.
		SMB-I2C_SDMA_Term	INPUT	DMA termination control from SMB-I2C Slave channel.
		SMB-I2C_SDMA_Done	OUTPUT	DMA termination control from DMA Controller to Slave channel.
	5	SMB-I2C_MDMA_Req	INPUT	DMA request control from SMB-I2C Master channel.
		SMB-I2C_MDMA_Term	INPUT	DMA termination control from SMB-I2C Master channel.
		SMB-I2C_MDMA_Done	OUTPUT	DMA termination control from DMA Controller to Master channel.

TABLE 7-3: DMA CONTROLLER MASTER DEVICES SIGNAL LIST (CONTINUED)

Device Name	Dev Num	Device Signal Name	Direction	Description
SMB-I2C 3 Controller	6	SMB-I2C_SDMA_Req	INPUT	DMA request control from SMB-I2C Slave channel.
		SMB-I2C_SDMA_Term	INPUT	DMA termination control from SMB-I2C Slave channel.
		SMB-I2C_SDMA_Done	OUTPUT	DMA termination control from DMA Controller to Slave channel.
	7	SMB-I2C_MDMA_Req	INPUT	DMA request control from SMB-I2C Master channel.
		SMB-I2C_MDMA_Term	INPUT	DMA termination control from SMB-I2C Master channel.
		SMB-I2C_MDMA_Done	OUTPUT	DMA termination control from DMA Controller to Master channel.
SPI 0 Controller	8	SPI_TDMA_Req	INPUT	DMA request control from GP-SPI TX channel.
	9	SPI_RDMA_Req	INPUT	DMA request control from GP-SPI RX channel.
SPI 1 Controller	10	SPI_TDMA_Req	INPUT	DMA request control from GP-SPI TX channel.
	11	SPI_RDMA_Req	INPUT	DMA request control from GP-SPI RX channel.
Quad SPI Controller	12	QSPI_TDMA_Req	INPUT	DMA request control from Quad SPI TX channel.
	13	QSPI_RDMA_Req	INPUT	DMA request control from Quad SPI RX channel.

7.5 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

7.5.1 POWER DOMAINS

TABLE 7-4: POWER SOURCES

Name	Description
VTR	This power well sources the registers and logic in this block.

7.5.2 CLOCK INPUTS

TABLE 7-5: CLOCK INPUTS

Name	Description
48MHz	This clock signal drives selected logic (e.g., counters).

7.5.3 RESETS

TABLE 7-6: RESET SIGNALS

Name	Description
RESET_SYS	This reset signal resets all of the registers and logic in this block.
RESET	This reset is generated if either the RESET_SYS is asserted or the SOFT_RESET bit is asserted.

7.6 Interrupts

This section defines the Interrupt Sources generated from this block.

TABLE 7-7: INTERRUPTS

Source	Description
DMA0	Direct Memory Access Channel 0 This signal is generated by the STATUS_DONE bit.
DMA1	Direct Memory Access Channel 1 This signal is generated by the STATUS_DONE bit.
DMA2	Direct Memory Access Channel 2 This signal is generated by the STATUS_DONE bit.
DMA3	Direct Memory Access Channel 3 This signal is generated by the STATUS_DONE bit.
DMA4	Direct Memory Access Channel 4 This signal is generated by the STATUS_DONE bit.
DMA5	Direct Memory Access Channel 5 This signal is generated by the STATUS_DONE bit.
DMA6	Direct Memory Access Channel 6 This signal is generated by the STATUS_DONE bit.
DMA7	Direct Memory Access Channel 7 This signal is generated by the STATUS_DONE bit.
DMA8	Direct Memory Access Channel 8 This signal is generated by the STATUS_DONE bit.
DMA9	Direct Memory Access Channel 9 This signal is generated by the STATUS_DONE bit.
DMA10	Direct Memory Access Channel 10 This signal is generated by the STATUS_DONE bit.
DMA11	Direct Memory Access Channel 11 This signal is generated by the STATUS_DONE bit.
DMA12	Direct Memory Access Channel 12 This signal is generated by the STATUS_DONE bit.
DMA13	Direct Memory Access Channel 13 This signal is generated by the STATUS_DONE bit.

7.7 Low Power Modes

The Internal DMA Controller may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry.

When the block is commanded to go to sleep it will place the DMA block into sleep mode only after all transactions on the DMA have been completed. For Firmware Flow Controlled transactions, the DMA will wait until it hits its terminal count and clears the Go control bit. For Hardware Flow Control, the DMA will go to sleep after either the terminal count is hit, or the Master device flags the terminate signal.

7.8 Description

The CEC1702 features a 14 channel DMA controller. The DMA controller can autonomously move data from/to any DMA capable master device to/from any populated memory location. This mechanism allows hardware IP blocks to transfer large amounts of data into or out of memory without EC intervention.

The DMA has the following characteristics:

- Data is only moved 1 Data Packet at a time
- Data only moves between devices on the accessible via the internal 32-bit address space
- The DMA Controller has 14 DMA Channels
- Each DMA Channel may be configured to communicate with any DMA capable device on the 32-bit internal address space. Each device has been assigned a device number. See Section 7.4.3, "DMA Interface".

The controller will accesses SRAM buffers only with incrementing addresses (that is, it cannot start at the top of a buffer, nor does it handle circular buffers automatically). The controller does not handle chaining (that is, automatically starting a new DMA transfer when one finishes).

7.8.1 CONFIGURATION

The DMA Controller is enabled via the ACTIVATE bit in DMA Main Control Register register.

Each DMA Channel must also be individually enabled via the CHANNEL_ACTIVATE bit in the DMA Channel N Activate Register to be operational.

Before starting a DMA transaction on a DMA Channel the host must assign a DMA Master to the channel via HARDWARE_FLOW_CONTROL_DEVICE. The host must not configure two different channels to the same DMA Master at the same time.

Data will be transfered between the DMA Master, starting at the programmed DEVICE_ADDRESS, and the targeted memory location, starting at the MEMORY_START_ADDRESS. The address for either the DMA Master or the targeted memory location may remain static or it may increment. To enable the DMA Master to increment its address set the INCREMENT_DEVICE_ADDRESS bit. To enable the targeted memory location to increment its addresses set the INCREMENT_MEMORY_ADDRESS. The DMA transfer will continue as long as the target memory address being accessed is less than the MEMORY_END_ADDRESS. If the DMA Controller detects that the memory location it is attempting to access on the Target is equal to the MEMORY_END_ADDRESS it will notify the DMA Master that the transaction is done. Otherwise the Data will be transferred in packets. The size of the packet is determined by the TRANSFER_SIZE.

7.8.2 OPERATION

The DMA Controller is designed to move data from one memory location to another.

7.8.2.1 Establishing a Connection

A DMA Master will initiate a DMA Transaction by requesting access to a channel. The DMA arbiter, which evaluates each channel request using a basic round robin algorithm, will grant access to the DMA master. Once granted, the channel will hold the grant until it decides to release it, by notifying the DMA Controller that it is done.

If Firmware wants to prevent any other channels from being granted while it is active it can set the LOCK_CHANNEL bit.

7.8.2.2 Initiating a Transfer

Once a connection is established the DMA Master will issue a DMA request to start a DMA transfer. If Firmware wants to have a transfer request serviced it must set the RUN bit to have its transfer requests serviced.

Firmware can initiate a transaction by setting the TRANSFER_GO bit. The DMA transfer will remain active until either the Master issues a Terminate or the DMA Controller signals that the transfer is DONE. Firmware may terminate a transaction by setting the TRANSFER_ABORT bit.

Note:	Before initiating a DMA transaction via firmware the hardware flow control must be disabled via the DISABLE_HARDWARE_FLOW_CONTROL bit.

Data may be moved from the DMA Master to the targeted Memory address or from the targeted Memory Address to the DMA Master. The direction of the transfer is determined by the TRANSFER_DIRECTION bit.

Once a transaction has been initiated firmware can use the STATUS_DONE bit to determine when the transaction is completed. This status bit is routed to the interrupt interface. In the same register there are additional status bits that indicate if the transaction completed successfully or with errors. These bits are OR'd together with the STATUS_DONE bit to generate the interrupt event. Each status be may be individually enabled/disabled from generating this event.

7.8.2.3 Reusing a DMA Channel

After a DMA Channel controller has completed, firmware **must** clear both the DMA Channel N Control Register and the DMA Channel N Interrupt Status Register. After both have been cleared to 0, the Channel Control Register can then be configured for the next transaction.

7.8.2.4 CRC Generation

A CRC generator can be attached to a DMA channel in order to generate a CRC on the data as it is transfered from the source to the destination. The CRC used is the CRC-32 algorithm used in IEEE 802.3 and many other protocols, using the polynomial $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$. The CRC generation takes

place in parallel with the data transfer; enabling CRC will not increase the time to complete a DMA transaction. The CRC generator has the optional ability to automatically transfer the generated CRC to the destination after the data transfer has completed.

CRC generation is subject to a number of restrictions:

- The CRC is only generated on channels that have the CRC hardware. See Table 7-10, "Channel Register Summary" for a definition of which channels have the ability to generate a CRC
- The DMA transfer must be 32-bits
- If CRC is enabled, DMA interrupts are inhibited until the CRC is completed, including the optional post-transfer copy of it is enabled
- The CRC must be initialized by firmware. The value FFFFFFFFh must be written to the Data Register in order to initialize the generator for the standard CRC-32-IEEE algorithm
- The CRC will bit-order reverse In and Out, and Invert Out, as required by the CRC algorithm

7.8.2.5 Block Fill Option

A Fill engine can be attached to a DMA channel in order to provide a fast mechanism to set a block of memory to a fixed value (for example, clearing a block of memory to zero). The block fill operation runs approximately twice as fast as a memory-to-memory copy.

In order to fill memory with a constant value, firmware **must** configure the channel in the following order:

1. Set the DMA Channel N Fill Data Register to the desired fill value
2. Set the DMA Channel N Fill Enable Register to '1b', enabling the Fill engine
3. Set the DMA Channel N Control Register to the following values:
   - RUN = 0
   - TRANSFER_DIRECTION = 0 (memory destination)
   - INCREMENT_MEMORY_ADDRESS = 1 (increment memory address after each transfer)
   - INCREMENT_DEVICE_ADDRESS = 1
   - DISABLE_HARDWARE_FLOW_CONTROL = 1 (no hardware flow control)
   - TRANSFER_SIZE = 1, 2 or 4 (as required)
   - TRANSFER_ABORT = 0
   - TRANSFER_GO = 1 (this starts the transfer)

7.9 EC Registers

The DMA Controller consists of a Main Block and a number of Channels. Table 7-9, "Main Register Summary" lists the registers in the Main Block and Table 7-10, "Channel Register Summary" lists the registers in each channel. Addresses for each register are determined by adding the offset to the Base Address for the DMA Controller Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

Registers are listed separately for the Main Block of the DMA Controller and for a DMA Channel. Each Channel has the same set of registers. The absolute register address for registers in each channel are defined by adding the Base Address for the DMA Controller Block, the Offset for the Channel shown in Table 7-8, "DMA Channel Offsets" to the offsets listed in Table 7-9, "Main Register Summary" or Table 7-10, "Channel Register Summary".

TABLE 7-8: DMA CHANNEL OFFSETS

Instance Name	Channel Number	Offset
DMA Controller	Main Block	000h
DMA Controller	0	040h
DMA Controller	1	080h
DMA Controller	2	0C0h
DMA Controller	3	100h
DMA Controller	4	140h
DMA Controller	5	180h
DMA Controller	6	1C0h
DMA Controller	7	200h

TABLE 7-8: DMA CHANNEL OFFSETS (CONTINUED)

Instance Name	Channel Number	Offset
DMA Controller	8	240h
DMA Controller	9	28h
DMA Controller	10	2C0h
DMA Controller	11	300h
DMA Controller	12	340h
DMA Controller	13	380h

TABLE 7-9: MAIN REGISTER SUMMARY

Offset	Register Name
00h	DMA Main Control Register
04h	DMA Data Packet Register

7.9.1 DMA MAIN CONTROL REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:2	Reserved	R	-	-
1	SOFT_RESET Soft reset the entire module. This bit is self-clearing.	W	0b	-
0	ACTIVATE Enable the blocks operation. 1=Enable block. Each individual channel must be enabled separately. 0=Disable all channels.	R/WS	0b	RESET

7.9.2 DMA DATA PACKET REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	DATA_PACKET Debug register that has the data that is stored in the Data Packet. This data is read data from the currently active transfer source.	R	0000h	-

TABLE 7-10: CHANNEL REGISTER SUMMARY

Offset	Register Name (Note 1)
00h	DMA Channel N Activate Register

Note 1: The letter 'N' following DMA Channel indicates the Channel Number. Each Channel implemented will have these registers to determine that channel's operation.

2: These registers are only present on DMA Channel 0. They are reserved on all other channels.

3: These registers are only present on DMA Channel 1. They are reserved on all other channels.

TABLE 7-10: CHANNEL REGISTER SUMMARY (CONTINUED)

Offset	Register Name (Note 1)
04h	DMA Channel N Memory Start Address Register
08h	DMA Channel N Memory End Address Register
0Ch	DMA Channel N Device Address
10h	DMA Channel N Control Register
14h	DMA Channel N Interrupt Status Register
18h	DMA Channel N Interrupt Enable Register
1Ch	TEST
20h (Note 2)	DMA Channel N CRC Enable Register
24h (Note 2)	DMA Channel N CRC Data Register
28h (Note 2)	DMA Channel N CRC Post Status Register
2Ch (Note 2)	TEST
20h (Note 3)	DMA Channel N Fill Enable Register
24h (Note 3)	DMA Channel N Fill Data Register
28h (Note 3)	DMA Channel N Fill Status Register
2Ch (Note 3)	TEST

Note 1: The letter 'N' following DMA Channel indicates the Channel Number. Each Channel implemented will have these registers to determine that channel's operation.

2: These registers are only present on DMA Channel 0. They are reserved on all other channels.

3: These registers are only present on DMA Channel 1. They are reserved on all other channels.

7.9.3 DMA CHANNEL N ACTIVATE REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:1	Reserved	R	-	-
0	CHANNEL_ACTIVATE Enable this channel for operation. The DMA Main Control:Activate must also be enabled for this channel to be operational.	R/W	0h	RESET

7.9.4 DMA CHANNEL N MEMORY START ADDRESS REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	MEMORY_START_ADDRESS This is the starting address for the Memory device. This field is updated by Hardware after every packet transfer by the size of the transfer, as defined by DMA Channel Control:Channel Transfer Size while the DMA Channel Control:Increment Memory Address is Enabled. The Memory device is defined as the device that is the slave device in the transfer. With Hardware Flow Control, the Memory device is the device that is not connected to the Hardware Flow Controlling device.	R/W	0000h	RESET

7.9.5 DMA CHANNEL N MEMORY END ADDRESS REGISTER

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	MEMORY_END_ADDRESS This is the ending address for the Memory device. This will define the limit of the transfer, so long as DMA Channel Control:Increment Memory Address is Enabled. When the Memory Start Address is equal to this value, the DMA will terminate the transfer and flag the status DMA Channel Interrupt:Status Done. **Note:** If the TRANSFER_SIZE field in the DMA Channel N Control Register is set to 2 (for 2-byte transfers, this address **must** be evenly divisible by 2 or the transfer will not terminate properly. If the TRANSFER_SIZE field is set to 4 (for 4-byte transfers, this address **must** be evenly divisible by 4 or the transfer will not terminate properly.	R/W	0000h	RESET

7.9.6 DMA CHANNEL N DEVICE ADDRESS

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	DEVICE_ADDRESS This is the Master Device address. This is used as the address that will access the Device on the DMA. The Device is defined as the Master of the DMA transfer; as in the device that is controlling the Hardware Flow Control. This field is updated by Hardware after every Data Packet transfer by the size of the transfer, as defined by DMA Channel Control:Transfer Size while the DMA Channel Control:Increment Device Address is Enabled.	R/W	0000h	RESET

7.9.7 DMA CHANNEL N CONTROL REGISTER

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:26	Reserved	R	-	-
25	TRANSFER_ABORT This is used to abort the current transfer on this DMA Channel. The aborted transfer will be forced to terminate immediately.	R/W	0h	RESET
24	TRANSFER_GO This is used for the **Firmware Flow Control** DMA transfer. This is used to start a transfer under the **Firmware Flow Control**. Do not use this in conjunction with the **Hardware Flow Control**; DISABLE_HARDWARE_FLOW_CONTROL must be set in order for this field to function correctly.	R/W	0h	RESET
23	Reserved	R	-	-
22:20	TRANSFER_SIZE This is the transfer size in Bytes of each Data Packet transfer. The transfer size must be a legal transfer size. Valid sizes are 1, 2 and 4 Bytes.	R/W	0h	RESET
19	DISABLE_HARDWARE_FLOW_CONTROL Setting this bit to '1'b will Disable **Hardware Flow Control**. When disabled, any DMA Master device attempting to communicate to the DMA over the DMA Flow Control Interface will be ignored. This should be set before using the DMA channel in **Firmware Flow Control** mode.	R/W	0h	RESET
18	LOCK_CHANNEL This is used to lock the arbitration of the Channel Arbiter on this channel once this channel is granted. Once this is locked, it will remain on the arbiter until it has completed it transfer (either the Transfer Aborted, Transfer Done or Transfer Terminated conditions). **Note:** This setting may starve other channels if the locked channel takes an excessive period of time to complete.	R/W	0h	RESET
17	INCREMENT_DEVICE_ADDRESS If this bit is '1'b, the DEVICE_ADDRESS will be incremented by TRANSFER_SIZE after every Data Packet transfer	R/W	0h	RESET
16	INCREMENT_MEMORY_ADDRESS If this bit is '1'b, the MEMORY_START_ADDRESS will be incremented by TRANSFER_SIZE after every Data Packet transfer **Note:** If this is not set, the DMA will never terminate the transfer on its own. It will have to be terminated through the Hardware Flow Control or through a DMA Channel Control:Transfer Abort.	R/W	0h	RESET

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
15:9	HARDWARE_FLOW_CONTROL_DEVICE This is the device that is connected to this channel as its Hardware Flow Control master. The Flow Control Interface is a bus with each master concatenated onto it. This selects which bus index of the concatenated Flow Control Interface bus is targeted towards this channel.	R/W	0h	RESET
8	TRANSFER_DIRECTION This determines the direction of the DMA Transfer. 1=Data Packet Read from MEMORY_START_ADDRESS followed by Data Packet Write to DEVICE_ADDRESS 0=Data Packet Read from DEVICE_ADDRESS followed by Data Packet Write to MEMORY_START_ADDRESS	R/W	0h	RESET
7:6	Reserved	R	-	-
5	BUSY This is a status signal. 1=The DMA Channel is busy (FSM is not IDLE) 0=The DMA Channel is not busy (FSM is IDLE)	R	0h	RESET
4:3	TEST	R	0h	RESET
2	DONE This is a status signal. It is only valid while RUN is Enabled. This is the inverse of the **DMA Channel Control:Busy** field, except this is qualified with the **DMA Channel Control:Run** field. 1=Channel is done 0=Channel is not done or it is OFF	R	0h	RESET
1	REQUEST This is a status field. 1=There is a transfer request from the Master Device 0=There is no transfer request from the Master Device	R	0h	RESET
0	RUN This is a control field. It only applies to **Hardware Flow Control** mode. 1=This channel is enabled and will service transfer requests 0=This channel is disabled. All transfer requests are ignored	R/W	0h	RESET

7.9.8 DMA CHANNEL N INTERRUPT STATUS REGISTER

Offset	14h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:3	Reserved	R	-	-
2	STATUS_DONE This is an interrupt source register. This flags when the DMA Channel has completed a transfer successfully on its side. A completed transfer is defined as when the DMA Channel reaches its limit; **Memory Start Address** equals **Memory End Address**. A completion due to a **Hardware Flow Control Terminate** will not flag this interrupt. 1=MEMORY_START_ADDRESS equals MEMORY_END_ADDRESS 0=MEMORY_START_ADDRESS does not equal MEMORY_END_ADDRESS	R/WC	0h	RESET
1	STATUS_FLOW_CONTROL This is an interrupt source register. This flags when the DMA Channel has encountered a **Hardware Flow Control Request** after the DMA Channel has completed the transfer. This means the Master Device is attempting to overflow the DMA. 1=Hardware Flow Control is requesting after the transfer has completed 0=No Hardware Flow Control event	R/WC	0h	RESET
0	STATUS_BUS_ERROR This is an interrupt source register. This flags when there is an Error detected over the internal 32-bit Bus. 1=Error detected.	R/WC	0h	RESET

7.9.9 DMA CHANNEL N INTERRUPT ENABLE REGISTER

Offset	18h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:3	Reserved	R	-	-
2	STATUS_ENABLE_DONE This is an interrupt enable for STATUS_DONE. 1=Enable Interrupt 0=Disable Interrupt	R/W	0h	RESET
1	STATUS_ENABLE_FLOW_CONTROL_ERROR This is an interrupt enable for STATUS_FLOW_CONTROL. 1=Enable Interrupt 0=Disable Interrupt	R/W	0h	RESET

Offset	18h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
0	STATUS_ENABLE_BUS_ERROR This is an interrupt enable for STATUS_BUS_ERROR. 1=Enable Interrupt 0=Disable Interrupt	R/W	0h	RESET

7.9.10 DMA CHANNEL N CRC ENABLE REGISTER

Offset	20h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:2	Reserved	R	-	-
1	CRC_POST_TRANSFER_ENABLE The bit enables the transfer of the calculated CRC-32 after the completion of the DMA transaction. If the DMA transaction is aborted by either firmware or an internal bus error, the transfer will not occur. If the target of the DMA transfer is a device and the device signaled the termination of the DMA transaction, the CRC post transfer will not occur. 1=Enable the transfer of CRC-32 for DMA Channel N after the DMA transaction completes 0=Disable the automatic transfer of the CRC	R/W	0h	RESET
0	CRC_MODE_ENABLE 1=Enable the calculation of CRC-32 for DMA Channel N 0=Disable the calculation of CRC-32 for DMA Channel N	R/W	0h	RESET

7.9.11 DMA CHANNEL N CRC DATA REGISTER

Offset	24h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	CRC Writes to this register initialize the CRC generator. Reads from this register return the output of the CRC that is calculated from the data transfered by DMA Channel N. The output of the CRC generator is bit-reversed and inverted on reads, as required by the CRC-32-IEEE definition. A CRC can be accumulated across multiple DMA transactions on Channel N. If it is necessary to save the intermediate CRC value, the result of the read of this register must be bit-reversed and inverted before being written back to this register.	R/W	0h	RESET

7.9.12 DMA CHANNEL N CRC POST STATUS REGISTER

Offset	28h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:4	Reserved	R	-	-
3	CRC_DATA_READY This bit is set to ‘1b’ when the DMA controller is processing the post-transfer of the CRC data. This bit is cleared to ‘0b’ when the post-transfer completes.	R	0h	RESET
2	CRC_DATA_DONE This bit is set to ‘1b’ when the DMA controller has completed the post-transfer of the CRC data. This bit is cleared to ‘0b’ when the a new DMA transfer starts.	R	0h	RESET
1	CRC_RUNNING This bit is set to ‘1b’ when the DMA controller starts the post-transfer transmission of the CRC. It is only set when the post-transfer is enabled by the CRC_POST_TRANSFER_ENABLE field. This bit is cleared to ‘0b’ when the post-transfer completes.	R	0h	RESET
0	CRC_DONE This bit is set to ‘1b’ when the CRC calculation has completed from either normal or forced termination. It is cleared to ‘0b’ when the DMA controller starts a new transfer on the channel.	R	0h	RESET

7.9.13 DMA CHANNEL N FILL ENABLE REGISTER

Offset	20h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:1	Reserved	R	-	-
0	FILL_MODE_ENABLE 1=Enable the calculation of CRC-32 for DMA Channel N 0=Disable the calculation of CRC-32 for DMA Channel N	R/W	0h	RESET

7.9.14 DMA CHANNEL N FILL DATA REGISTER

Offset	24h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	DATA This is the data pattern used to fill memory.	R/W	0h	RESET

7.9.15 DMA CHANNEL N FILL STATUS REGISTER

Offset	28h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:2	Reserved	R	-	-
1	FILL_RUNNING This bit is ‘1b’ when the Fill operation starts and is cleared to ‘0b’ when the Fill operation completes.	R	0h	RESET
0	FILL_DONE This bit is set to ‘1b’ when the Fill operation has completed from either normal or forced termination. It is cleared to ‘0b’ when the DMA controller starts a new transfer on the channel.	R	0h	RESET

8.0 EC INTERRUPT AGGREGATOR

8.1 Introduction

The EC Interrupt Aggregator works in conjunction with the processor's interrupt interface to handle hardware interrupts and exceptions.

Exceptions are synchronous to instructions, are not maskable, and have higher priority than interrupts. All three exceptions - reset, memory error, and instruction error - are hardwired directly to the processor. Interrupts are typically asynchronous and are maskable.

Interrupts classified as wake events can be recognized without a running clock, e.g., while the CEC1702 is in sleep state.

This chapter focuses on the EC Interrupt Aggregator. Please refer to embedded controller's documentation for more information on interrupt and exception handling.

8.2 References

None

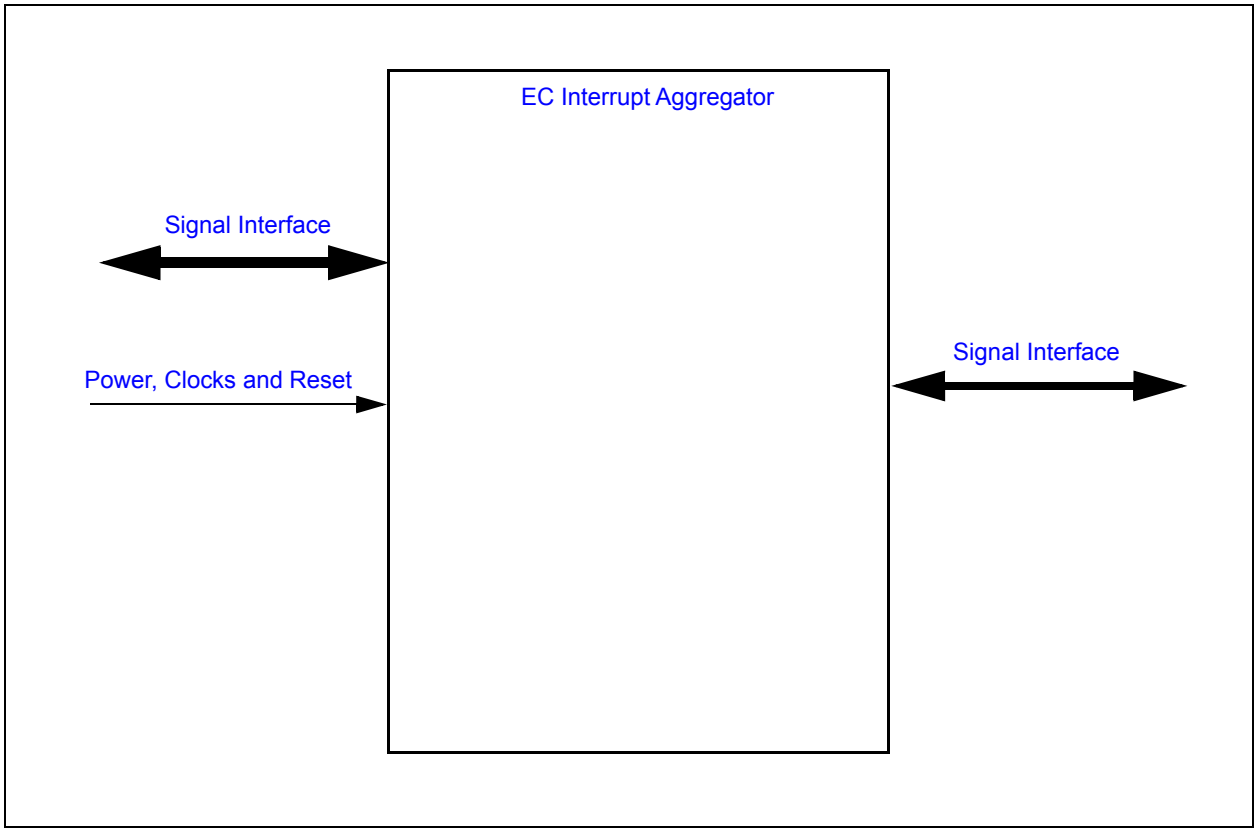
8.3 Terminology

None

8.4 Interface

This block is an IP block designed to be incorporated into a chip. It is designed to be accessed externally via the pin interface and internally via a registered host interface. The following diagram illustrates the various interfaces to the block.

FIGURE 8-1: EC INTERRUPT AGGREGATOR INTERFACE DIAGRAM



8.5 Signal Description

8.5.1 SIGNAL INTERFACE

There are no external signals for this block.

8.6 Host Interface

The registers defined for the EC Interrupt Aggregator are only accessible by the embedded controller via the EC Registers.

8.7 Power, Clocks and Reset

8.7.1 BLOCK POWER DOMAIN

TABLE 8-1: BLOCK POWER

Power Well Source	Effect on Block
VTR	The EC Interrupt Aggregator block and registers operate on this single power well.

8.7.2 BLOCK CLOCKS

None

8.7.3 BLOCK RESET

TABLE 8-2: BLOCK RESETS

Reset Name	Reset Description
RESET_SYS	This signal is used to indicate when the VTR logic and registers in this block are reset.

8.8 Interrupts

This block aggregates all the interrupts targeted for the embedded controller into the Source Registers defined in Section 8.11, "EC Registers". The unmasked bits of each source register are then OR'd together and routed to the embedded controller's interrupt interface. The name of each Source Register identifies the IRQ number of the interrupt port on the embedded controller.

8.9 Low Power Modes

This block always automatically adjusts to operate in the lowest power mode.

8.10 Description

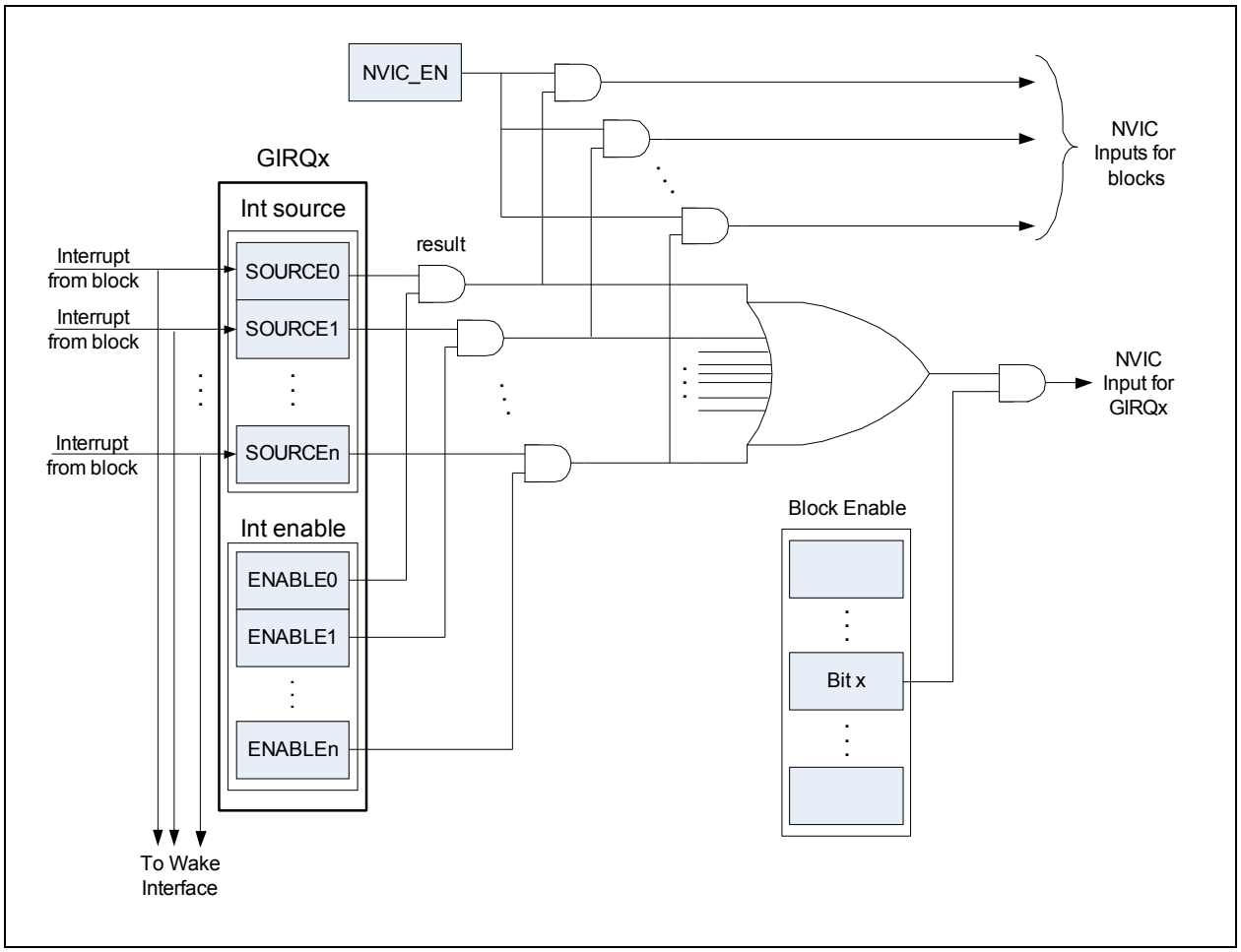
The interrupt generation logic is made of groups of signals, each of which consist of a Status register, a Enable Set register, and Enable Clear register and a Result register.

The Status and Enable are latched registers. There is one set of Enable register bits; both the Enable Set and Enable Clear registers return the same result when read. The Enable Set interface is used to set individual bits in the Enable register, and the Enable Clear is used to clear individual bits. The Result register is a bit by bit AND function of the Source and Enable registers. All the bits of the Result register are OR'ed together and AND'ed with the corresponding bit in the Block Select register to form the interrupt signal that is routed to the ARM interrupt controller.

The Result register bits may also be enabled to the NVIC block via the NVIC_EN bit in the Interrupt Control Register register. See Chapter 33.0, "EC Subsystem Registers"

Section 8.10.1 shows a representation of the interrupt structure.

FIGURE 8-2: INTERRUPT STRUCTURE



8.10.1 AGGREGATED INTERRUPTS

All interrupts are routed to the ARM processor through the ARM Nested Vectored Interrupt Controller (NVIC). As shown in Figure 8-2, "Interrupt Structure", all interrupt sources are aggregated into the GIRQx Source registers. In many cases, the Result bit for an individual interrupt source is tied directly to the NVIC. These interrupts are shown in the “Direct NVIC” column in the Interrupt Bit Assignments table. In addition, all GIRQx can also generate an interrupt to the NVIC when any of the enabled interrupts in its group is asserted. The NVIC vectors for the aggregated GIRQ interrupts are shown tin the “Agg NVIC” column.

Firmware should not enable the group GIRQ NVIC interrupt at the same time individual direct interrupts for members of the group are enabled. If both are enabled, the processor will receive two interrupts for an event, one from the GIRQ and one from the direct interrupt.

Note:	The four Soft Interrupts that are defined by the RTOS Timer do not have individual NVIC vectors. If the use of the SWI interrupts is required, then all interrupts in the GIRQ must disable the individual NVIC vectors.

8.10.2 WAKE GENERATION

The EC Interrupt Aggregator notifies the Chip Power Management Features to wake the system when it detects a wake capable event has occurred. This logic requires no clocks.

The interrupt sources AND’ed with the corresponding Enable bit will be OR’ed to produce a wake event

The wake up sources are identified with a “Y” in the “WAKE” column of the Bit definitions table for each IRQ’s Source Register.

8.10.2.1 Configuring Wake Interrupts

All GPIO inputs are wake-capable. In order for a GPIO input to wake the CEC1702 from a sleep state, the Interrupt Detection field of the GPIO Pin Control Register must be set to Rising Edge Triggered, Falling Edge Triggered, or Either Edge Triggered. If the Interrupt Detection field is set to any other value, a GPIO input will not trigger a wake interrupt.

Some of the Wake Capable Interrupts are triggered by activity on pins that are shared with a GPIO. These interrupts will only trigger a wake if the Interrupt Detection field of the corresponding GPIO Pin Control Register is set to Rising Edge Triggered, Falling Edge Triggered, or Either Edge Triggered.

8.10.3 INTERRUPT SUMMARY

Interrupt bit assignments, including wake capabilities and NVIC vector locations, are shown in the Interrupt Aggregator Bit Assignments Table in Section 3.0, "Device Inventory". The table lists all possible interrupt sources; the register bits for any interrupt source, such as a GPIO, that is not implemented in a particular part are reserved.

8.10.4 DISABLING INTERRUPTS

The Block Enable Clear Register and Block Enable Set Register should not be used for disabling and enabling interrupts for software operations i.e., critical sections. The ARM enable disable mechanisms should be used.

8.11 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for of the EC Interrupt Aggregator Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 8-3: REGISTER SUMMARY

Offset	Register Name
00h	GIRQ8 Source Register
04h	GIRQ8 Enable Set Register
08h	GIRQ8 Result Register
0Ch	GIRQ8 Enable Clear Register
10h	Reserved
14h	GIRQ9 Source Register
18h	GIRQ9 Enable Set Register
1Ch	GIRQ9 Result Register
20h	GIRQ9 Enable Clear Register
24h	Reserved
28h	GIRQ10 Source Register
2Ch	GIRQ10 Enable Set Register
30h	GIRQ10 Result Register
34h	GIRQ10 Enable Clear Register
38h	Reserved
3Ch	GIRQ11 Source Register
40h	GIRQ11 Enable Set Register
44h	GIRQ11 Result Register
48h	GIRQ11 Enable Clear Register
4Ch	Reserved
50h	GIRQ12 Source Register
54h	GIRQ12 Enable Set Register
58h	GIRQ12 Result Register

TABLE 8-3: REGISTER SUMMARY (CONTINUED)

Offset	Register Name
5Ch	GIRQ12 Enable Clear Register
60h	Reserved
64h	GIRQ13 Source Register
68h	GIRQ13 Enable Set Register
6Ch	GIRQ13 Result Register
70h	GIRQ13 Enable Clear Register
74h	Reserved
78h	GIRQ14 Source Register
7Ch	GIRQ14 Enable Set Register
80h	GIRQ14 Result Register
84h	GIRQ14 Enable Clear Register
88h	Reserved
8Ch	GIRQ15 Source Register
90h	GIRQ15 Enable Set Register
94h	GIRQ15 Result Register
98h	GIRQ15 Enable Clear Register
9Ch	Reserved
A0h	GIRQ16 Source Register
A4h	GIRQ16 Enable Set Register
A8h	GIRQ16 Result Register
ACh	GIRQ16 Enable Clear Register
B0h	Reserved
B4h	GIRQ17 Source Register
B8h	GIRQ17 Enable Set Register
BCh	GIRQ17 Result Register
C0h	GIRQ17 Enable Clear Register
C4h	Reserved
C8h	GIRQ18 Source Register
CCh	GIRQ18 Enable Set Register
D0h	GIRQ18 Result Register
D4h	GIRQ18 Enable Clear Register
D8h	Reserved
DCh	GIRQ19 Source Register
E0h	GIRQ19 Enable Set Register
E4h	GIRQ19 Result Register
E8h	GIRQ19 Enable Clear Register
ECh	Reserved
F0h	GIRQ20 Source Register
F4h	GIRQ20 Enable Set Register
F8h	GIRQ20 Result Register
FCh	GIRQ20 Enable Clear Register
100h	Reserved
104h	GIRQ21 Source Register
108h	GIRQ21 Enable Set Register

TABLE 8-3: REGISTER SUMMARY (CONTINUED)

Offset	Register Name
10Ch	GIRQ21 Result Register
110h	GIRQ21 Enable Clear Register
114h	Reserved
118h	GIRQ22 Source Register
11Ch	GIRQ22 Enable Set Register
120h	GIRQ22 Result Register
124h	GIRQ22 Enable Clear Register
128h	Reserved
12Ch	GIRQ23 Source Register
130h	GIRQ23 Enable Set Register
134h	GIRQ23 Result Register
138h	GIRQ23 Enable Clear Register
13Ch	Reserved
140h	GIRQ24 Source Register
144h	GIRQ24 Enable Set Register
148h	GIRQ24 Result Register
14Ch	GIRQ24 Enable Clear Register
150h	Reserved
154h	GIRQ25 Source Register
158h	GIRQ25 Enable Set Register
15Ch	GIRQ25 Result Register
160h	GIRQ25 Enable Clear Register
164h	Reserved
168h	GIRQ26 Source Register
16Ch	GIRQ26 Enable Set Register
170h	GIRQ26 Result Register
174h	GIRQ26 Enable Clear Register
200h	Block Enable Set Register
204h	Block Enable Clear Register
208h	Block IRQ Vector Register

All of the GIRQx Source, Enable Set, Enable Clear and Result registers have the same format. The following tables define the generic format for each of these registers. The bit definitions are defined in the sections that follow.

The behavior of the enable bit controlled by the GIRQx Enable Set and GIRQx Enable Clear Registers, the GIRQx Source bit, and the GIRQx Result bit is illustrated in Section 8.10.1, "Aggregated Interrupts".

8.11.1 GIRQ SOURCE REGISTERS

All of the GIRQx Source registers have the same format. The following table defines the generic format for each of these registers. The bit definitions are defined in the Interrupt Aggregator Bit Assignments Table in Section 3.0, "Device Inventory". Unassigned bits are Reserved and return 0.

Note:	If a GPIO listed in the tables does not appear in the pin list of a particular device, then the bits for that GPIO in the GIRQx Source, GIRQx Enable Clear, GIRQx Enable Set and GIRQx Result are reserved.

Offset	See Section 3.0, "Device Inventory"			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31	Reserved	R	-	-
30:0	GIRQX_SOURCE The GIRQx Source bits are R/WC sticky status bits indicating the state of interrupt before the interrupt enable bit.	R/WC	0h	RESET_SYS

8.11.2 GIRQ ENABLE SET REGISTERS

All of the GIRQx Enable Set registers have the same format. The following table defines the generic format for each of these registers. Unassigned bits are Reserved and return 0.

Offset	See Section 3.0, "Device Inventory"			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31	Reserved	R	-	-
30:0	GIRQX_ENABLE_SET Each GIRQx bit can be individually enabled to assert an interrupt event. Reads always return the current value of the internal GIRQX_ENABLE bit. The state of the GIRQX_ENABLE bit is determined by the corresponding GIRQX_ENABLE_SET bit and the GIRQX_ENABLE_CLEAR bit. (0=disabled, 1-enabled) 1=The corresponding interrupt in the GIRQx Source Register is enabled 0=No effect	R/WS	0h	RESET_SYS

8.11.3 GIRQ ENABLE CLEAR REGISTERS

All of the GIRQx Enable Clear registers have the same format. The following table defines the generic format for each of these registers. Unassigned bits are Reserved and return 0.

Offset	See Section 3.0, "Device Inventory"			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31	Reserved	R	-	-
30:0	GIRQX_ENABLE_CLEAR Each GIRQx bit can be individually enabled to assert an interrupt event. Reads always return the current value of the internal GIRQX_ENABLE bit. The state of the GIRQX_ENABLE bit is determined by the corresponding GIRQX_ENABLE_SET bit and the GIRQX_ENABLE_CLEAR bit. (0=disabled, 1-enabled) 1=The corresponding interrupt in the GIRQx Source Register is disabled 0=No effect	R/WC	0h	RESET_SYS

8.11.4 GIRQ RESULT REGISTERS

Offset	See Section 3.0, "Device Inventory"			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31	Reserved	R	1h	-
30:0	GIRQX_RESULT The GIRQX_RESULT bits are Read-Only status bits indicating the state of an interrupt. The RESULT is asserted ‘1’b when both the GIRQX_SOURCE bit and the corresponding GIRQX_ENABLE bit are ‘1’b.	R	0h	RESET_SYS

8.11.5 BLOCK ENABLE SET REGISTER

Offset	200h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:27	Reserved	R	-	-
26:8	IRQ_VECTOR_ENABLE_SET Each GIRQx register can be individually enabled to assert an interrupt event. Reads always return the current value of the enable bits for each of the GIRQs. The state of the GIRQX_ENABLE bit is determined by the corresponding GIRQX_ENABLE_SET bit and the GIRQX_ENABLE_CLEAR bit. (0=disabled, 1-enabled) 1=Interrupts in the GIRQx Source Register may be enabled 0=No effect	R/WS	0h	RESET_SYS
7:0	Reserved	R	-	-

8.11.6 BLOCK ENABLE CLEAR REGISTER

Offset	204h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:27	Reserved	R	-	-
26:8	IRQ_VECTOR_ENABLE_CLEAR Each GIRQx register can be individually disabled to inhibit interrupt events. Reads always return the current value of the internal GIRQX_ENABLE bit. The state of the GIRQX_ENABLE bit is determined by the corresponding GIRQX_ENABLE_SET bit and the GIRQX_ENABLE_CLEAR bit. (0=disabled, 1-enabled) 1=All interrupts in the GIRQx Source Register are disabled 0=No effect	R/WC	0h	RESET_SYS
7:0	Reserved	R	-	-

8.11.7 BLOCK IRQ VECTOR REGISTER

Offset	208h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:27	Reserved	R	0h	-
26:8	IRQ_VECTOR Each bit in this field reports the status of the group GIRQ interrupt assertion to the NVIC. If the GIRQx interrupt is disabled as a group, by the Block Enable Clear Register, then the corresponding bit will be ‘0’b and no interrupt will be asserted.	R	0h	RESET_SYS
7:0	Reserved	R	0h	-

9.0 UART

9.1 Introduction

The 16550 UART (Universal Asynchronous Receiver/Transmitter) is a full-function Serial Port that supports the standard RS-232 Interface.

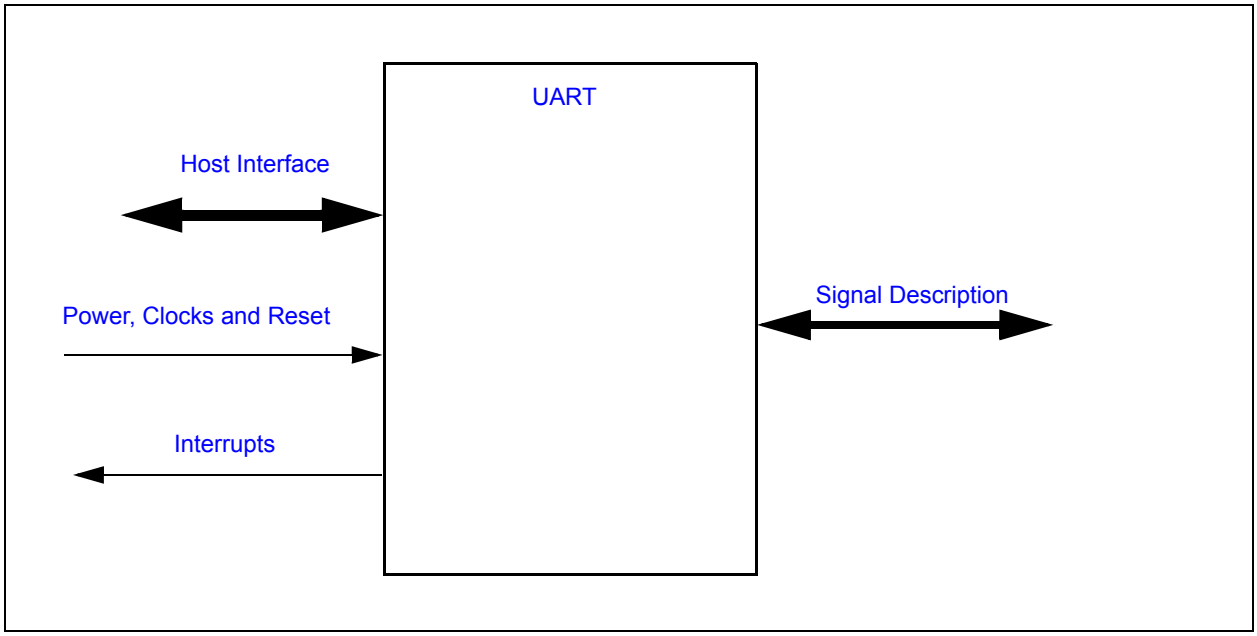
9.2 References

- EIA Standard RS-232-C specification

9.3 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

FIGURE 9-1: I/O DIAGRAM OF BLOCK



9.4 Signal Description

TABLE 9-1: SIGNAL DESCRIPTION TABLE

Name	Direction	Description
DTR#	Output	Active low Data Terminal ready output for the Serial Port. Handshake output signal notifies modem that the UART is ready to transmit data. This signal can be programmed by writing to bit 1 of the Modem Control Register (MCR). **Note:** Defaults to tri-state on V3_DUAL power on.
DCD#	Output	Active low Data Carrier Detect input for the serial port. Handshake signal which notifies the UART that carrier signal is detected by the modem. The CPU can monitor the status of DCD# signal by reading bit 7 of Modem Status Register (MSR). A DCD# signal state change from low to high after the last MSR read will set MSR bit 3 to a 1. If bit 3 of Interrupt Enable Register is set, the interrupt is generated when DCD # changes state.

TABLE 9-1: SIGNAL DESCRIPTION TABLE (CONTINUED)

Name	Direction	Description
DSR#	Input	Active low Data Set Ready input for the serial port. Handshake signal which notifies the UART that the modem is ready to establish the communication link. The CPU can monitor the status of DSR# signal by reading bit 5 of Modem Status Register (MSR). A DSR# signal state change from low to high after the last MSR read will set MSR bit 1 to a 1. If bit 3 of Interrupt Enable Register is set, the interrupt is generated when DSR# changes state.
RI#	Input	Active low Ring Indicator input for the serial port. Handshake signal which notifies the UART that the telephone ring signal is detected by the modem. The CPU can monitor the status of RI# signal by reading bit 6 of Modem Status Register (MSR). A RI# signal state change from low to high after the last MSR read will set MSR bit 2 to a 1. If bit 3 of Interrupt Enable Register is set, the interrupt is generated when RI# changes state.
RTS#	Output	Active low Request to Send output for the Serial Port. Handshake output signal notifies modem that the UART is ready to transmit data. This signal can be programmed by writing to bit 1 of the Modem Control Register (MCR). The hardware reset will reset the RTS# signal to inactive mode (high). RTS# is forced inactive during loop mode operation. Defaults to tri-state on V3_DUAL power on.
CTS#	Input	Active low Clear to Send input for the serial port. Handshake signal which notifies the UART that the modem is ready to receive data. The CPU can monitor the status of CTS# signal by reading bit 4 of Modem Status Register (MSR). A CTS# signal state change from low to high after the last MSR read will set MSR bit 0 to a 1. If bit 3 of the Interrupt Enable Register is set, the interrupt is generated when CTS# changes state. The CTS# signal has no effect on the transmitter.
TXD	Output	Transmit serial data output.
RXD	Input	Receiver serial data input.

9.5 Host Interface

The UART is accessed by host software via a registered interface, as defined in Section 9.11, "Configuration Registers" and Section 9.10, "Runtime Registers".

9.6 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

9.6.1 POWER DOMAINS

TABLE 9-2: POWER SOURCES

Name	Description
VTR	This Power Well is used to power the registers and logic in this block.

9.6.2 CLOCKS

TABLE 9-3: CLOCK INPUTS

Name	Description
48MHz	This is the main clock domain. Because the clock input must be within ± 2% in order to generate standard baud rates, the 48MHz clock must be generated by a reference clock with better than 1% accuracy (i.e., external crystal) and locked to its frequency before the UART will work with the standard rates.

TABLE 9-4: DERIVED CLOCKS

Name	Description
1.8432MHz	The UART requires a 1.8432 MHz ± 2% clock input for baud rate generation of standard baud rates up to 115,200 baud. It is derived from the system 48MHz clock domain.
24MHz	A 24MHz ± 2% clock input for baud rate generation, derived from the system 48MHz clock domain. It may be used as an alternative to the 1.8432MHz clock, generating non-standard baud rates up to 1,500,000 baud.

9.6.3 RESETS

TABLE 9-5: RESET SIGNALS

Name	Description
RESET_SYS	This reset is asserted when VTR is applied.

9.7 Interrupts

This section defines the Interrupt Sources generated from this block.

TABLE 9-6: SYSTEM INTERRUPTS

Source	Description
UART	The UART interrupt event output indicates if an interrupt is pending. See Table 9-12, "Interrupt Control Table".

TABLE 9-7: EC INTERRUPTS

Source	Description
UART	The UART interrupt event output indicates if an interrupt is pending. See Table 9-12, "Interrupt Control Table".

9.8 Low Power Modes

The UART may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry.

9.9 Description

The UART is compatible with the 16450, the 16450 ACE registers and the 16C550A. The UART performs serial-to-parallel conversions on received characters and parallel-to-serial conversions on transmit characters. Two sets of baud rates are provided. When the 1.8432 MHz source clock is selected, standard baud rates from 50 to 115.2K are available. When the source clock is 24 MHz, baud rates up to 1,500K are available. The character options are programmable for 1 start; 1, 1.5 or 2 stop bits; even, odd, sticky or no parity; and prioritized interrupts. The UART contains a programmable baud rate generator that is capable of dividing the input clock signal by 1 to 65535. The UART is also capable of sup-

porting the MIDI data rate. Refer to the Configuration Registers for information on disabling, powering down and changing the base address of the UART. The UART interrupt is enabled by programming OUT2 of the UART to logic “1.” Because OUT2 is logic “0,” it disables the UART's interrupt. The UART is accessible by both the Host and the EC.

9.9.1 PROGRAMMABLE BAUD RATE

The Serial Port contains a programmable Baud Rate Generator that is capable of dividing the internal clock source by any divisor from 1 to 65535. Unless an external clock source is configured, the clock source is either the 1.8432MHz clock source or the 48MHz clock source. The output frequency of the Baud Rate Generator is 16x the Baud rate. Two eight bit latches store the divisor in 16 bit binary format. These Divisor Latches must be loaded during initialization in order to ensure desired operation of the Baud Rate Generator. Upon loading either of the Divisor Latches, a 16 bit Baud counter is immediately loaded. This prevents long counts on initial load. If a 0 is loaded into the BRG registers, the output divides the clock by the number 3. If a 1 is loaded, the output is the inverse of the input oscillator. If a two is loaded, the output is a divide by 2 signal with a 50% duty cycle. If a 3 or greater is loaded, the output is low for 2 bits and high for the remainder of the count.

The following tables show possible baud rates.

TABLE 9-8: UART BAUD RATES USING CLOCK SOURCE 1.8432MHz

Desired Baud Rate	BAUD_CLOCK_SEL	Divisor Used to Generate 16X Clock
50	0	2304
75	0	1536
110	0	1047
134.5	0	857
150	0	768
300	0	384
600	0	192
1200	0	96
1800	0	64
2000	0	58
2400	0	48
3600	0	32
4800	0	24
7200	0	16
9600	0	12
19200	0	6
38400	0	3
57600	0	2
115200	0	1

TABLE 9-9: UART BAUD RATES USING CLOCK SOURCE 48MHz

Desired Baud Rate	BAUD_CLOCK_SEL	Divisor Used to Generate 16X Clock
125000	1	24
136400	1	22
150000	1	20
166700	1	18
187500	1	16
214300	1	14
250000	1	12

TABLE 9-9: UART BAUD RATES USING CLOCK SOURCE 48MHz (CONTINUED)

Desired Baud Rate	BAUD_CLOCK_SEL	Divisor Used to Generate 16X Clock
300000	1	10
375000	1	8
500000	1	6
750000	1	4
1500000	1	2
3000000	1	1

9.10 Runtime Registers

The registers listed in the Runtime Register Summary table are for a single instance of the UART. Host access for each register listed in this table is defined as an offset in the Host address space to the Block's Base Address, as defined by the instance's Base Address Register.

The EC address for each register is formed by adding the Base Address for each instance of the UART shown in the Block Overview and Base Address Table in Section 3.0, "Device Inventory" to the offset shown in the "Offset" column.

TABLE 9-10: RUNTIME REGISTER SUMMARY

DLAB Note 1	Offset	Register Name
0	0h	Receive Buffer Register
0	0h	Transmit Buffer Register
1	0h	Programmable Baud Rate Generator LSB Register
1	1h	Programmable Baud Rate Generator MSB Register
0	1h	Interrupt Enable Register
x	02h	FIFO Control Register
x	02h	Interrupt Identification Register
x	03h	Line Control Register
x	04h	Modem Control Register
x	05h	Line Status Register
x	06h	Modem Status Register
x	07h	Scratchpad Register
Note 1: DLAB is bit 7 of the Line Control Register.		

9.10.1 RECEIVE BUFFER REGISTER

Offset	0h (DLAB=0)			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	RECEIVED_DATA This register holds the received incoming data byte. Bit 0 is the least significant bit, which is transmitted and received first. Received data is double buffered; this uses an additional shift register to receive the serial data stream and convert it to a parallel 8 bit word which is transferred to the Receive Buffer register. The shift register is not accessible.	R	0h	RESET_SYS

9.10.2 TRANSMIT BUFFER REGISTER

Offset	0h (DLAB=0)			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	TRANSMIT_DATA This register contains the data byte to be transmitted. The transmit buffer is double buffered, utilizing an additional shift register (not accessible) to convert the 8 bit data word to a serial format. This shift register is loaded from the Transmit Buffer when the transmission of the previous byte is complete.	W	0h	RESET_SYS

9.10.3 PROGRAMMABLE BAUD RATE GENERATOR LSB REGISTER

Offset	00h (DLAB=1)			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	BAUD_RATE_DIVISOR_LSB See Section 9.9.1, "Programmable Baud Rate".	R/W	0h	RESET_SYS

9.10.4 PROGRAMMABLE BAUD RATE GENERATOR MSB REGISTER

Offset	01h (DLAB=1)			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7	BAUD_CLK_SEL If CLK_SRC is '0': • 0=The baud clock is derived from the 1.8432MHz. • 1=IThe baud clock is derived from the 24MHz. If CLK_SRC is '1': • This bit has no effect	R/W	0h	RESET_SYS
6:0	BAUD_RATE_DIVISOR_MSB See Section 9.9.1, "Programmable Baud Rate".	R/W	0h	RESET_SYS

9.10.5 INTERRUPT ENABLE REGISTER

The lower four bits of this register control the enables of the five interrupt sources of the Serial Port interrupt. It is possible to totally disable the interrupt system by resetting bits 0 through 3 of this register. Similarly, setting the appropriate bits of this register to a high, selected interrupts can be enabled. Disabling the interrupt system inhibits the Interrupt Identification Register and disables any Serial Port interrupt out of the CEC1702. All other system functions operate in their normal manner, including the Line Status and MODEM Status Registers. The contents of the Interrupt Enable Register are described below.

Offset	01h (DLAB=0)			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:4	Reserved	R	-	-
3	EMSI This bit enables the MODEM Status Interrupt when set to logic “1”. This is caused when one of the Modem Status Register bits changes state.	R/W	0h	RESET_SYS
2	ELSI This bit enables the Received Line Status Interrupt when set to logic “1”. The error sources causing the interrupt are Overrun, Parity, Framing and Break. The Line Status Register must be read to determine the source.	R/W	0h	RESET_SYS
1	ETHREI This bit enables the Transmitter Holding Register Empty Interrupt when set to logic “1”.	R/W	0h	RESET_SYS
0	ERDAI This bit enables the Received Data Available Interrupt (and timeout interrupts in the FIFO mode) when set to logic “1”.	R/W	0h	RESET_SYS

9.10.6 FIFO CONTROL REGISTER

This is a write only register at the same location as the Interrupt Identification Register.

Note:	DMA is not supported.

Offset	02h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:6	RECV_FIFO_TRIGGER_LEVEL These bits are used to set the trigger level for the RCVR FIFO interrupt.	W	0h	RESET_SYS
5:4	Reserved	R	-	-
3	DMA_MODE_SELECT Writing to this bit has no effect on the operation of the UART. The RXRDY and TXRDY pins are not available on this chip.	W	0h	RESET_SYS
2	CLEAR_XMIT_FIFO Setting this bit to a logic “1” clears all bytes in the XMIT FIFO and resets its counter logic to “0”. The shift register is not cleared. This bit is self-clearing.	W	0h	RESET_SYS
1	CLEAR_RECv_FIFO Setting this bit to a logic “1” clears all bytes in the RCVR FIFO and resets its counter logic to “0”. The shift register is not cleared. This bit is self-clearing.	W	0h	RESET_SYS
0	EXRF Enable XMIT and RECV FIFO. Setting this bit to a logic “1” enables both the XMIT and RCVR FIFOs. Clearing this bit to a logic “0” disables both the XMIT and RCVR FIFOs and clears all bytes from both FIFOs. When changing from FIFO Mode to non-FIFO (16450) mode, data is automatically cleared from the FIFOs. This bit must be a 1 when other bits in this register are written to or they will not be properly programmed.	W	0h	RESET_SYS

TABLE 9-11: RECV FIFO TRIGGER LEVELS

Bit 7	Bit 6	RECV FIFO Trigger Level (BYTES)
0	0	1
	1	4
1	0	8
	1	14

9.10.7 INTERRUPT IDENTIFICATION REGISTER

By accessing this register, the host CPU can determine the highest priority interrupt and its source. Four levels of priority interrupt exist. They are in descending order of priority:

1. Receiver Line Status (highest priority)
2. Received Data Ready
3. Transmitter Holding Register Empty
4. MODEM Status (lowest priority)

Information indicating that a prioritized interrupt is pending and the source of that interrupt is stored in the Interrupt Identification Register (refer to Table 9-12). When the CPU accesses the IIR, the Serial Port freezes all interrupts and indicates the highest priority pending interrupt to the CPU. During this CPU access, even if the Serial Port records new interrupts, the current indication does not change until access is completed. The contents of the IIR are described below.

Offset	02h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:6	FIFO_EN These two bits are set when the FIFO CONTROL Register bit 0 equals 1.	R	0h	RESET_SYS
5:4	Reserved	R	-	-
3:1	INTID These bits identify the highest priority interrupt pending as indicated by Table 9-12, "Interrupt Control Table". In non-FIFO mode, Bit[3] is a logic “0”. In FIFO mode Bit[3] is set along with Bit[2] when a timeout interrupt is pending.	R	0h	RESET_SYS
0	IPEND This bit can be used in either a hardwired prioritized or polled environment to indicate whether an interrupt is pending. When bit 0 is a logic ‘0’ an interrupt is pending and the contents of the IIR may be used as a pointer to the appropriate internal service routine. When bit 0 is a logic ‘1’ no interrupt is pending.	R	1h	RESET_SYS

TABLE 9-12: INTERRUPT CONTROL TABLE

FIFO Mode Only	Interrupt Identification Register			Interrupt SET and RESET Functions			
Bit 3	**Bit 2**	**Bit 1**	**Bit 0**	**Priority Level**	**Interrupt Type**	**Interrupt Source**	**Interrupt Reset Control**
0	0	0	1	-	None	None	-
	1	1	0	Highest	Receiver Line Status	Overrun Error, Parity Error, Framing Error or Break Interrupt	Reading the Line Status Register
		0		Second	Received Data Available	Receiver Data Available	Read Receiver Buffer or the FIFO drops below the trigger level.
1					Character Timeout Indication	No Characters Have Been Removed From or Input to the RCVR FIFO during the last 4 Char times and there is at least 1 char in it during this time	Reading the Receiver Buffer Register
0	0	1		Third	Transmitter Holding Register Empty	Transmitter Holding Register Empty	Reading the IIR Register (if Source of Interrupt) or Writing the Transmitter Holding Register
	0	0		Fourth	MODEM Status	Clear to Send or Data Set Ready or Ring Indicator or Data Carrier Detect	Reading the MODEM Status Register

9.10.8 LINE CONTROL REGISTER

Offset	03h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7	DLAB Divisor Latch Access Bit (DLAB). It must be set high (logic “1”) to access the Divisor Latches of the Baud Rate Generator during read or write operations. It must be set low (logic “0”) to access the Receiver Buffer Register, the Transmitter Holding Register, or the Interrupt Enable Register.	R/W	0h	RESET_SYS
6	BREAK_CONTROL Set Break Control bit. When bit 6 is a logic “1”, the transmit data output (TXD) is forced to the Spacing or logic “0” state and remains there (until reset by a low level bit 6) regardless of other transmitter activity. This feature enables the Serial Port to alert a terminal in a communications system.	R/W	0h	RESET_SYS
5	STICK_PARITY Stick Parity bit. When parity is enabled it is used in conjunction with bit 4 to select Mark or Space Parity. When LCR bits 3, 4 and 5 are 1 the Parity bit is transmitted and checked as a 0 (Space Parity). If bits 3 and 5 are 1 and bit 4 is a 0, then the Parity bit is transmitted and checked as 1 (Mark Parity). If bit 5 is 0 Stick Parity is disabled. Bit 3 is a logic “1” and bit 5 is a logic “1”, the parity bit is transmitted and then detected by the receiver in the opposite state indicated by bit 4.	R/W	0h	RESET_SYS
4	PARITY_SELECT Even Parity Select bit. When bit 3 is a logic “1” and bit 4 is a logic “0”, an odd number of logic “1”'s is transmitted or checked in the data word bits and the parity bit. When bit 3 is a logic “1” and bit 4 is a logic “1” an even number of bits is transmitted and checked.	R/W	0h	RESET_SYS
3	ENABLE_PARITY Parity Enable bit. When bit 3 is a logic “1”, a parity bit is generated (transmit data) or checked (receive data) between the last data word bit and the first stop bit of the serial data. (The parity bit is used to generate an even or odd number of 1s when the data word bits and the parity bit are summed).	R/W	0h	RESET_SYS
2	STOP_BITS This bit specifies the number of stop bits in each transmitted or received serial character. Table 9-13 summarizes the information. The receiver will ignore all stop bits beyond the first, regardless of the number used in transmitting.	R/W	0h	RESET_SYS
1:0	WORD_LENGTH These two bits specify the number of bits in each transmitted or received serial character. The encoding of bits 0 and 1 is as follows:	R/W	0h	RESET_SYS

TABLE 9-13: STOP BITS

Bit 2	Word Length	Number of Stop Bits
0	--	1
1	5 bits	1.5
	6 bits	2
	7 bits	
	8 bits	

TABLE 9-14: SERIAL CHARACTER

Bit 1	Bit 0	Word Length
0	0	5 Bits
0	1	6 Bits
1	0	7 Bits
1	1	8 Bits

The Start, Stop and Parity bits are not included in the word length.

9.10.9 MODEM CONTROL REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:5	Reserved	R	-	-
4	LOOPBACK This bit provides the loopback feature for diagnostic testing of the Serial Port. When bit 4 is set to logic “1”, the following occur: 1. The TXD is set to the Marking State (logic “1”). 2. The receiver Serial Input (RXD) is disconnected. 3. The output of the Transmitter Shift Register is “looped back” into the Receiver Shift Register input. 4. All MODEM Control inputs (CTS#, DSR#, RI# and DCD#) are disconnected. 5. The four MODEM Control outputs (DTR#, RTS#, OUT1 and OUT2) are internally connected to the four MODEM Control inputs (DSR#, CTS#, RI#, DCD#). 6. The Modem Control output pins are forced inactive high. 7. Data that is transmitted is immediately received. This feature allows the processor to verify the transmit and receive data paths of the Serial Port. In the diagnostic mode, the receiver and the transmitter interrupts are fully operational. The MODEM Control Interrupts are also operational but the interrupts' sources are now the lower four bits of the MODEM Control Register instead of the MODEM Control inputs. The interrupts are still controlled by the Interrupt Enable Register.	R/W	0h	RESET_SYS
3	OUT2 Output 2 (OUT2). This bit is used to enable an UART interrupt. When OUT2 is a logic “0”, the serial port interrupt output is forced to a high impedance state - disabled. When OUT2 is a logic “1”, the serial port interrupt outputs are enabled.	R/W	0h	RESET_SYS
2	OUT1 This bit controls the Output 1 (OUT1) bit. This bit does not have an output pin and can only be read or written by the CPU.	R/W	0h	RESET_SYS
1	RTS This bit controls the Request To Send (RTS#) output. When bit 1 is set to a logic “1”, the RTS# output is forced to a logic “0”. When bit 1 is set to a logic “0”, the RTS# output is forced to a logic “1”.	R/W	0h	RESET_SYS
0	DTR This bit controls the Data Terminal Ready (DTR#) output. When bit 0 is set to a logic “1”, the DTR# output is forced to a logic “0”. When bit 0 is a logic “0”, the DTR# output is forced to a logic “1”.	R/W	0h	RESET_SYS

9.10.10 LINE STATUS REGISTER

Offset	05h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7	FIFO_ERROR This bit is permanently set to logic “0” in the 450 mode. In the FIFO mode, this bit is set to a logic “1” when there is at least one parity error, framing error or break indication in the FIFO. This bit is cleared when the LSR is read if there are no subsequent errors in the FIFO.	R	0h	RESET_SYS
6	TRANSMIT_ERROR Transmitter Empty. Bit 6 is set to a logic “1” whenever the Transmitter Holding Register (THR) and Transmitter Shift Register (TSR) are both empty. It is reset to logic “0” whenever either the THR or TSR contains a data character. Bit 6 is a read only bit. In the FIFO mode this bit is set whenever the THR and TSR are both empty,	R	0h	RESET_SYS
5	TRANSMIT_EMPTY Transmitter Holding Register Empty Bit 5 indicates that the Serial Port is ready to accept a new character for transmission. In addition, this bit causes the Serial Port to issue an interrupt when the Transmitter Holding Register interrupt enable is set high. The THRE bit is set to a logic “1” when a character is transferred from the Transmitter Holding Register into the Transmitter Shift Register. The bit is reset to logic “0” whenever the CPU loads the Transmitter Holding Register. In the FIFO mode this bit is set when the XMIT FIFO is empty, it is cleared when at least 1 byte is written to the XMIT FIFO. Bit 5 is a read only bit.	R	0h	RESET_SYS
4	BREAK_INTERRUPT Break Interrupt. Bit 4 is set to a logic “1” whenever the received data input is held in the Spacing state (logic “0”) for longer than a full word transmission time (that is, the total time of the start bit + data bits + parity bits + stop bits). The BI is reset after the CPU reads the contents of the Line Status Register. In the FIFO mode this error is associated with the particular character in the FIFO it applies to. This error is indicated when the associated character is at the top of the FIFO. When break occurs only one zero character is loaded into the FIFO. Restarting after a break is received, requires the serial data (RXD) to be logic “1” for at least 1/2 bit time. Bits 1 through 4 are the error conditions that produce a Receiver Line Status Interrupt BIT 3 whenever any of the corresponding conditions are detected and the interrupt is enabled	R	0h	RESET_SYS

Offset	05h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
3	FRAME_ERROR Framing Error. Bit 3 indicates that the received character did not have a valid stop bit. Bit 3 is set to a logic “1” whenever the stop bit following the last data bit or parity bit is detected as a zero bit (Spacing level). This bit is reset to a logic “0” whenever the Line Status Register is read. In the FIFO mode this error is associated with the particular character in the FIFO it applies to. This error is indicated when the associated character is at the top of the FIFO. The Serial Port will try to resynchronize after a framing error. To do this, it assumes that the framing error was due to the next start bit, so it samples this 'start' bit twice and then takes in the 'data'.	R	0h	RESET_SYS
2	PARITY ERROR Parity Error. Bit 2 indicates that the received data character does not have the correct even or odd parity, as selected by the even parity select bit. This bit is set to a logic “1” upon detection of a parity error and is reset to a logic “0” whenever the Line Status Register is read. In the FIFO mode this error is associated with the particular character in the FIFO it applies to. This error is indicated when the associated character is at the top of the FIFO.	R	0h	RESET_SYS
1	OVERRUN_ERROR Overrun Error. Bit 1 indicates that data in the Receiver Buffer Register was not read before the next character was transferred into the register, thereby destroying the previous character. In FIFO mode, an overrun error will occur only when the FIFO is full and the next character has been completely received in the shift register, the character in the shift register is overwritten but not transferred to the FIFO. This bit is set to a logic “1” immediately upon detection of an overrun condition, and reset whenever the Line Status Register is read.	R	0h	RESET_SYS
0	DATA_READY Data Ready. It is set to a logic ‘1’ whenever a complete incoming character has been received and transferred into the Receiver Buffer Register or the FIFO. Bit 0 is reset to a logic ‘0’ by reading all of the data in the Receive Buffer Register or the FIFO.	R	0h	RESET_SYS

9.10.11 MODEM STATUS REGISTER

Offset	06h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7	DCD This bit is the complement of the Data Carrier Detect (DCD#) input. If bit 4 of the MCR is set to logic ‘1’, this bit is equivalent to OUT2 in the MCR.	R	0h	RESET_SYS
6	RI This bit is the complement of the Ring Indicator (RI#) input. If bit 4 of the MCR is set to logic ‘1’, this bit is equivalent to OUT1 in the MCR.	R	0h	RESET_SYS
5	DSR This bit is the complement of the Data Set Ready (DSR#) input. If bit 4 of the MCR is set to logic ‘1’, this bit is equivalent to DTR# in the MCR.	R	0h	RESET_SYS
4	CTS This bit is the complement of the Clear To Send (CTS#) input. If bit 4 of the MCR is set to logic ‘1’, this bit is equivalent to RTS# in the MCR.	R	0h	RESET_SYS
3	DCD Delta Data Carrier Detect (DDCD). Bit 3 indicates that the DCD# input to the chip has changed state. NOTE: Whenever bit 0, 1, 2, or 3 is set to a logic ‘1’, a MODEM Status Interrupt is generated.	R	0h	RESET_SYS
2	RI Trailing Edge of Ring Indicator (TERI). Bit 2 indicates that the RI# input has changed from logic ‘0’ to logic ‘1’.	R	0h	RESET_SYS
1	DSR Delta Data Set Ready (DDSR). Bit 1 indicates that the DSR# input has changed state since the last time the MSR was read.	R	0h	RESET_SYS
0	CTS Delta Clear To Send (DCTS). Bit 0 indicates that the CTS# input to the chip has changed state since the last time the MSR was read.	R	0h	RESET_SYS

9.10.12 SCRATCHPAD REGISTER

Offset	07h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	SCRATCH This 8 bit read/write register has no effect on the operation of the Serial Port. It is intended as a scratchpad register to be used by the programmer to hold data temporarily.	R/W	0h	RESET_SYS

9.11 Configuration Registers

Configuration Registers for an instance of the UART are listed in the following table. Host access to Configuration Registers is through the Configuration Port using the Logical Device Number of each instance of the UART and the Index shown in the “Host Index” column of the table. The EC can access Configuration Registers directly. The EC address for each register is formed by adding the Base Address for each instance of the UART shown in the Block Overview and Base Address Table in Section 3.0, "Device Inventory" to the offset shown in the “EC Offset” column.

TABLE 9-15: CONFIGURATION REGISTER SUMMARY

EC Offset	Host Index	Register Name
330h	30h	Activate Register
3F0h	F0h	Configuration Select Register

9.11.1 ACTIVATE REGISTER

Offset	30h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:1	Reserved	R	-	-
0	ACTIVATE When this bit is 1, the UART logical device is powered and functional. When this bit is 0, the UART logical device is powered down and inactive.	R/W	0b	RESET_SYS

9.11.2 CONFIGURATION SELECT REGISTER

Offset	F0h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:3	Reserved	R	-	-
2	POLARITY 1=The UART_TX and UART_RX pins functions are inverted 0=The UART_TX and UART_RX pins functions are not inverted	R/W	0b	RESET_SYS
1	Reserved [a]	R/W	1b	RESET_SYS
0	CLK_SRC 1=The UART Baud Clock is derived from an external clock source 0=The UART Baud Clock is derived from one of the two internal clock sources	R/W	0b	RESET_SYS

a. This bit (Bit 1 of Configuration Select Register) must be programmed by the application to 0h for proper working of the device.

10.0 GPIO INTERFACE

10.1 Overview

The CEC1702 GPIO interface provides general purpose input monitoring and output control, as well as managing many aspects of pin functionality; including, multi-function Pin Multiplexing Control, GPIO Direction control, Pull-up and Pull-down resistors, asynchronous wakeup and synchronous interrupt detection and Polarity control, as well as control of pin drive strength and slew rate.

Features of the GPIO interface include:

- Inputs:
  - Asynchronous rising and falling edge wakeup detection
  - Interrupt High or Low Level
- On Output:
  - Push Pull or Open Drain output
- Pull up or pull down resistor control
- Interrupt and wake capability available for all GPIOs
- Programmable pin drive strength and slew rate limiting
- Group- or individual control of GPIO data.
- Multiplexing of all multi-function pins are controlled by the GPIO interface

10.2 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

10.2.1 POWER DOMAINS

TABLE 10-1: POWER SOURCES

Name	Description
VTR	The I/O power for a subset of GPIOs is powered by this supply voltage.It may be 3.3V or 1.8V.

10.2.2 CLOCK INPUTS

TABLE 10-2: CLOCK INPUTS

Name	Description
48MHz	This clock domain is used for synchronizing GPIO inputs.

10.2.3 RESETS

TABLE 10-3: RESET SIGNALS

Name	Description
RESET_SYS	This reset is asserted when VTR is applied.

10.3 Interrupts

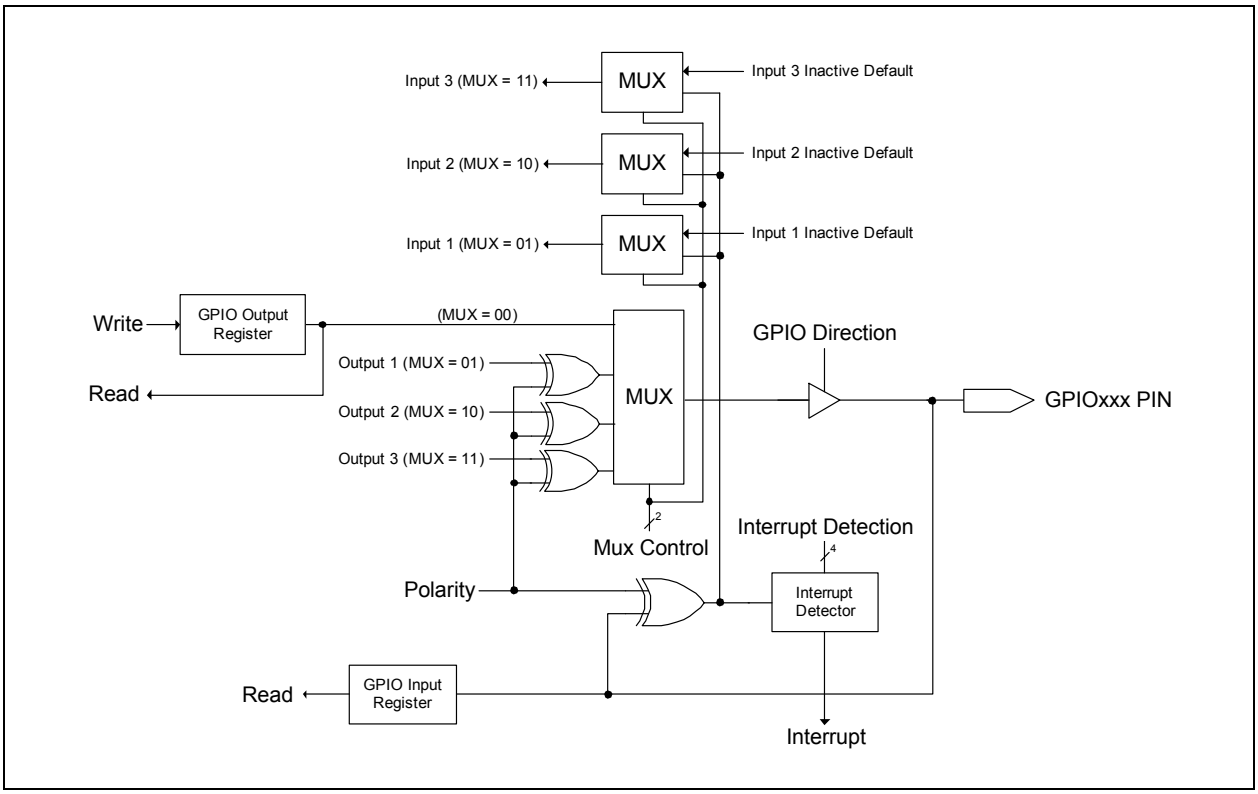
This section defines the Interrupt Sources generated from this block.

TABLE 10-4: INTERRUPTS

Source	Description
GPIO_Event	Each GPIO pin has the ability to generate an interrupt event. This event may be used as a wake event. The event can be generated on a high level, low level, rising edge, falling edge or either edge input, as configured by the INTERRUPT_DETECTION bits in the Pin Control Registers associated with the GPIO signal function. The minimum pulse width to generate in interrupt / wakeup event must be at least 5ns.

10.4 Description

FIGURE 10-1: GPIO BLOCK DIAGRAM

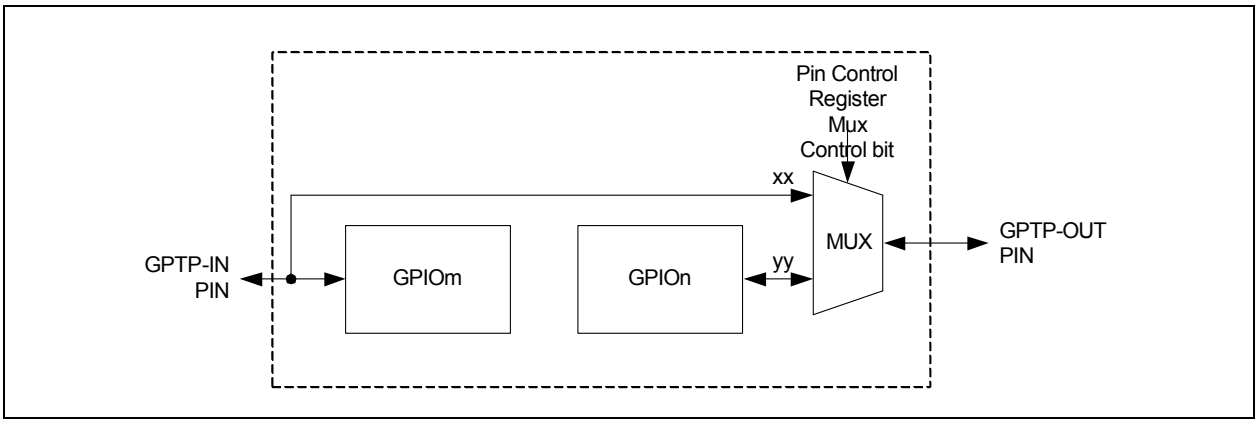


10.5 GPIO Pass-Through Ports

GPIO Pass-Through Ports (GPTP) can multiplex two general purpose I/O pins as shown in Figure 10-2. GPIO Pass-Through Ports connect the GPTP-IN pin to the GPTP-OUT pin. The GPTP are sequentially assigned values 0:7. The GPTP port assignment have no relation to the GPIO Indexing assignments. The GPTP ports are controlled by the Mux Control bits in the Pin Control Register associated with the GPTP-OUT signal function.

In order to enable the GPTP Pass-Through Mode, the GPTP-IN (GPIOm in Figure 10-2) Pin Control Register must assign the Mux Control to the GPTP_IN signal function and the GPIO Direction bit to 0 (input); the GPTP-OUT (GPIOn in Figure 10-2) Pin Control Register must assign the Mux Control to the GPTP_OUT signal function and the GPIO Direction bit to 1 (output). The GPTP-OUT signal function can differ from pin to pin.

FIGURE 10-2: GPIO PASS-THROUGH PORT EXAMPLE



The Pin Control Register Mux Control fields shown in Figure 10-2 are illustrated as ‘xx’ and ‘yy’ because this figure is an example, it does not represent the actual GPIO multiplexing configuration. The GPIO Multiplexing tables in this chapter must be used to determine the correct values to use to select between a GPIO and the pass-through.

When Pass-Through Mode is enabled, the GPIOn output is disconnected from the GPIOn pin and the GPIOm pin signal appears on GPIOn pin. Note that in this case the GPIOm input register still reflects the state of the GPIOm pin.

10.6 Accessing GPIOs

There are two ways to access GPIO output data. GPIO_OUTPUT_SELECT is used to determine which GPIO output data bit affects the GPIO output pin.

- Group Output GPIO Data
  - Outputs to individual GPIO ports are grouped into 32-bit GPIO Output Registers.
- Individual Output GPIO Data
  - Each GPIO output port may be individually accessible in the ALTERNATE_GPIO_DATA field of the port’s Pin Control Register. On reads, ALTERNATE_GPIO_DATA returns the programmed value, not the value on the pin.

There are two ways to access GPIO input data.

- Group Input GPIO Data
  - Inputs from individual GPIO ports are grouped into 32-bit GPIO Input Registers and always reflect the current state of the GPIO input from the pads, independent of the setting of the MUX_CONTROL field in the Pin Control Register.
- Group Input GPIO Data
  - Each GPIO input port is individually accessible in the GPIO_INPUT field of the port’s Pin Control Register. The GPIO_INPUT field always reflects the current state of GPIO input from the pad, independent of the setting of the MUX_CONTROL field in the Pin Control Register.

10.6.1 HOST ACCESS OF GPIOS

The GPIO Output Registers and GPIO Input Registers can be configured to be Host accessible via one of the SRAM Base Address Register, if the base of the internal address of the BAR is set to an offset of 200h from the GPIO base address, with a SIZE of 9, setting of block of 512 bytes. All of the Output and Input registers can then be accessed as offsets from the Host base address.

The GPIO Output and Input registers can also be accessed as one of the regions in an EMI block. This access is defined in the EMI Protocols chapter of the firmware specification.

10.7 GPIO Indexing

Each GPIO signal function name consists of a 4-character prefix (“GPIO”) followed by a 3-digit octal-encoded index number. In the CEC1702 GPIO indexing is done sequentially starting from GPIO000.

10.8 GPIO Multiplexing and Control Register Defaults

The default values for all GPIO Control Register and Control 2 Registers are shown in the GPIO Pin Control Registers Defaults Table in the GPIO Register Assignments Subsection of Section 3.0, "Device Inventory". The function multiplexing for each GPIO is shown in the GPIO Multiplexing Table in the same Subsection.

Note:	If a GPIO listed in the tables does not appear in the pin list of a particular device, then the Control Registers are reserved and should not be written. Similarly, if GPIO does not appear in the pin list then the bit position for the Input GPIO Register and Output GPIO Register that contains that GPIO is reserved.

10.9 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for the GPIO Interface Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 10-5: REGISTER SUMMARY

Offset	Register Name
000h - 2B3h	Pin Control Register for all GPIOs
300h	Input GPIO[000:036] Register
304h	Input GPIO[040:076] Register
308h	Input GPIO[100:136] Register
30Ch	Input GPIO[140:176] Register
310h	Input GPIO[200:236] Register
314h	Input GPIO[240:276] Register
380h	Output GPIO[000:036] Register
384h	Output GPIO[040:076] Register
388h	Output GPIO[100:136] Register
38Ch	Output GPIO[140:176] Register
390h	Output GPIO[200:236] Register
394h	Output GPIO[240:276] Register
500h - 7B3h	Pin Control 2 Register for all GPIOs

10.10 Pin Control Registers

Two Control Registers are implemented for each GPIO, the Pin Control Register and the Pin Control 2 Register.

10.10.1 PIN CONTROL REGISTER

Offset	See Section 10.8			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:25	Reserved	R	-	-
24	GPIO_INPUT Reads of this bit always return the state of GPIO input from the pad, independent of the Mux selection for the pin or the Direction, except as follows: 1. POWER_GATING = 11b - Input Disabled This bit is forced low when the input is disabled 2. POWER_GATING = 10b - Unpowered This bit is forced high when the pad is unpowered.	R	x	RESET_SYS
23:17	Reserved	R	-	-

Offset	See Section 10.8			
Bits	**Description**	**Type**	**Default**	**Reset Event**
16	ALTERNATE_GPIO_DATA Reads of this bit always return the last data written to the GPIO output data register bit; reads do not return the current output value of the GPIO pin if it is configured as an output. If the GPIO_OUTPUT_SELECT bit in this register is '1', then this bit is Read Only and the GPIO output data register bit is only written by the GPIO Output Register. If the GPIO_OUTPUT_SELECT bit in this register is '0', then this bit is R/W, and the bit corresponding to this GPIO in the GPIO Output Register is Read Only.	R or R/W	See Section 10.8	RESET_SYS
15:14	Reserved	R	-	-
13:12	MUX_CONTROL The Mux Control field determines the active signal function for a pin. 11b=Signal Function 3 Selected 10b=Signal Function 2 Selected 01b=Signal Function 1 Selected 00b=GPIO Function Selected	R/W	See Section 10.8	RESET_SYS
11	POLARITY When the Polarity bit is set to '1' and the MUX_CONTROL bits are greater than '00,' the selected signal function outputs are inverted and Interrupt Detection sense defined in Table 10-6, "Edge Enable and Interrupt Detection Bits Definition" is inverted. When the MUX_CONTROL field selects the GPIO signal function (Mux='00'), the Polarity bit does not effect the output. Regardless of the state of the MUX_CONTROL field and the Polarity bit, the state of the pin is always reported without inversion in the GPIO input register. 1=Inverted 0=Non-inverted	R/W	See Section 10.8	RESET_SYS
10	GPIO_OUTPUT_SELECT This control bit determines which register is used to update the data register for GPIO outputs. See Section 10.4, "Description" 1=GPIO output data for this GPIO come from the bit representing this GPIO in the GPIO Output Register; writes to the ALTERNATE_GPIO_DATA field of this register do not affect the GPIO 0=GPIO output data for this GPIO come from the ALTERNATE_GPIO_DATA field of this register; writes to the bit representing this GPIO in the GPIO Output Register do not affect the GPIO	R/W	See Section 10.8	RESET_SYS
9	GPIO_DIRECTION This bit controls the buffer direction only when the MUX_CONTROL field is '00' selecting the pin signal function to be GPIO. When the MUX_CONTROL field is greater than '00' (i.e., a non-GPIO signal function is selected) this bit has no affect and the selected signal function logic directly controls the pin direction. 1=Output 0=Input	R/W	See Section 10.8	RESET_SYS

Offset	See Section 10.8			
Bits	**Description**	**Type**	**Default**	**Reset Event**
8	OUTPUT_BUFFER_TYPE Unless explicitly stated otherwise, pins with (I/O/OD) or (O/OD) in their buffer type column in the tables in are compliant with the following Programmable OD/PP Multiplexing Design Rule: Each compliant pin has a programmable open drain/push-pull buffer controlled by the Output Buffer Type bit in the associated Pin Control Register. The state of this bit controls the mode of the interface buffer for all selected functions, including the GPIO function. 1=Open Drain 0=Push-Pull	R/W	See Section 10.8	RESET_SYS
7	EDGE_ENABLE When combined with the field INTERRUPT_DETECTION in this register, determines the interrupt capability of the GPIO input. See Table 10-6, "Edge Enable and Interrupt Detection Bits Definition" for details. 1=Edge detection enabled 0=Edge detection disabled	R/W	See Section 10.8	RESET_SYS
6:4	INTERRUPT_DETECTION When combined with the field INTERRUPT_DETECTION in this register, determines the interrupt capability of the GPIO input. See Table 10-6, "Edge Enable and Interrupt Detection Bits Definition" for details.	R/W	See Section 10.8	RESET_SYS
3:2	POWER_GATING The GPIO pin will be tristated when the selected power well is off. 11b=VTR Powered Output Only. Input pad is disabled and output will be tristated when VTR Power Rail is off. 10b=Unpowered. The GPIO pad is turned off completely. Both the input buffer and output buffer on the pad are disabled. Pull-up and pull-down resisters are disabled independent of the setting of the PU/PD field 01b=Reserved 00b=VTR Power Rail **Note:** The Under Voltage Support feature requires that this bit field be set to the 11b option, VTR Powered Output Only, when pad VTR=3.3V and pin is driving out to 1.8V using Open Drain Mode.	R/W	See Section 10.8	RESET_SYS
1:0	PU/PD These bits are used to enable an internal pull-up or pull-down resistor. 11b=”Keeper Mode”. In this mode a weak latch circuit holds the last value on a pad when it becomes tri-stated and undriven 10b=Pull Down Enabled 01b=Pull Up Enabled 00b=None	R/W	See Section 10.8	RESET_SYS

TABLE 10-6: EDGE ENABLE AND INTERRUPT DETECTION BITS DEFINITION

Edge Enable	Interrupt Detection Bits			Selected Function
D7	D6	D5	D4	
0	0	0	0	Low Level Sensitive
0	0	0	1	High Level Sensitive
0	0	1	0	Reserved
0	0	1	1	Reserved
0	1	0	0	Interrupt events are disabled
0	1	0	1	Reserved
0	1	1	0	Reserved
0	1	1	1	Reserved
1	1	0	1	Rising Edge Triggered
1	1	1	0	Falling Edge Triggered
1	1	1	1	Either edge triggered

Note: Only edge triggered interrupts can wake up the main clock domain. The GPIO must be enabled for edge-triggered interrupts and the GPIO interrupt must be enabled in the interrupt aggregator in order to wake from the Heavy Sleep state.

10.10.2 PIN CONTROL 2 REGISTER

Offset	See Section 10.8			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:6	Reserved	R	-	-
5:4	DRIVE_STRENGTH These bits are used to select the drive strength on the pin. The drive strength is the same whether the pin is powered by 3.3V or 1.8V. 11b=12mA 10b=8mA 01b=4mA 00b=2mA	R/W	See Section 10.8	RESET_SYS
3:1	Reserved	R	-	-
0	SLEW_RATE This bit is used to select the slew rate on the pin. 1=fast 0=slow (half frequency)	R/W	0h	RESET_SYS

10.10.3 GPIO INPUT REGISTERS

The GPIO Input Registers can always be used to read the state of a pin, even when the pin is configured as an output, /or when the pin is configured for a signal function other than the GPIO (i.e., the MUX_CONTROL field is not equal to '00.').

Note: Bits associated with GPIOs not present in the pinout for a particular device are Reserved.

10.10.3.1 Input GPIO[000:036] Register

Offset	300h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31	Reserved	R	-	-
30:24	GPIO[036:030] Input	R	00h	RESET_SYS
23:16	GPIO[027:020] Input	R	00h	RESET_SYS
15:8	GPIO[017:010] Input	R	00h	RESET_SYS
7:0	GPIO[007:000] Input	R	00h	RESET_SYS

10.10.3.2 Input GPIO[040:076] Register

Offset	304h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31	Reserved	R	-	-
30:24	GPIO[076:070] Input	R	00h	RESET_SYS
23:16	GPIO[067:060] Input	R	00h	RESET_SYS
15:8	GPIO[057:050] Input	R	00h	RESET_SYS
7:0	GPIO[047:040] Input	R	00h	RESET_SYS

10.10.3.3 Input GPIO[100:136] Register

Offset	308h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31	Reserved	R	-	-
30:24	GPIO[136:130] Input	R	00h	RESET_SYS

Offset	308h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
23:16	GPIO[127:120] Input	R	00h	RESET_SYS
15:8	GPIO[117:110] Input	R	00h	RESET_SYS
7:0	GPIO[107:100] Input	R	00h	RESET_SYS

10.10.3.4 Input GPIO[140:176] Register

Offset	30Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31	Reserved	R	-	-
30:24	GPIO[176:170] Input	R	00h	RESET_SYS
23:16	GPIO[167:160] Input	R	00h	RESET_SYS
15:8	GPIO[157:150] Input	R	00h	RESET_SYS
7:0	GPIO[147:140] Input	R	00h	RESET_SYS

10.10.3.5 Input GPIO[200:236] Register

Offset	310h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31	Reserved	R	-	-
30:24	GPIO[236:230] Input	R/W	00h	RESET_SYS
23:16	GPIO[227:220] Input	R/W	00h	RESET_SYS
15:8	GPIO[217:210] Input	R/W	00h	RESET_SYS
7:0	GPIO[207:200] Input	R/W	00h	RESET_SYS

10.10.3.6 Input GPIO[240:276] Register

Offset	314h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31	Reserved	R	-	-
30:24	GPIO[276:270] Input	R/W	00h	RESET_SYS
23:16	GPIO[267:260] Input	R/W	00h	RESET_SYS
15:8	GPIO[257:250] Input	R/W	00h	RESET_SYS
7:0	GPIO[247:240] Input	R/W	00h	RESET_SYS

10.10.4 GPIO OUTPUT REGISTERS

If enabled by the GPIO_OUTPUT_SELECT bit, the GPIO Output bits determine the level on the GPIO pin when the pin is configured for the GPIO output function. If the GPIO Output Register bit is not enabled, its value does not affect the GPIO pin. In all cases, reads return the last programmed value in the GPIO Output Register.

Note: Bits associated with GPIOs not present in the pinout for a particular device are Reserved.

10.10.4.1 Output GPIO[000:036] Register

Offset	380h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31	Reserved	R	-	-
30:24	GPIO[036:030] Output	R/W	00h	RESET_SYS
23:16	GPIO[027:020] Output	R/W	00h	RESET_SYS
15:8	GPIO[017:010] Output	R/W	00h	RESET_SYS
7:0	GPIO[007:000] Output	R/W	00h	RESET_SYS

10.10.4.2 Output GPIO[040:076] Register

Offset	384h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:24	Reserved	R	-	-
30:24	GPIO[076:070] Output	R/W	00h	RESET_SYS
23:16	GPIO[067:060] Output	R/W	00h	RESET_SYS
15:8	GPIO[057:050] Output	R/W	00h	RESET_SYS
7:0	GPIO[047:040] Output	R/W	00h	RESET_SYS

10.10.4.3 Output GPIO[100:136] Register

Offset	388h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31	Reserved	R	-	-
30:24	GPIO[136:130] Output	R/W	00h	RESET_SYS
23:16	GPIO[127:120] Output	R/W	00h	RESET_SYS
15:8	GPIO[117:110] Output	R/W	00h	RESET_SYS
7:0	GPIO[107:100] Output	R/W	00h	RESET_SYS

10.10.4.4 Output GPIO[140:176] Register

Offset	38Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:22	Reserved	R	-	-
30:24	GPIO[176:170] Output	R/W	00h	RESET_SYS

Offset	38Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
23:16	GPIO[175:160] Output	R/W	00h	RESET_SYS
15:8	GPIO[157:150] Output	R/W	00h	RESET_SYS
7:0	GPIO[147:140] Output	R/W	00h	RESET_SYS

10.10.4.5 Output GPIO[200:236] Register

Offset	390h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31	Reserved	R	-	-
30:24	GPIO[236:230] Output	R/W	00h	RESET_SYS
23:16	GPIO[227:220] Output	R/W	00h	RESET_SYS
15:8	GPIO[217:210] Output	R/W	00h	RESET_SYS
7:0	GPIO[207:200] Output	R/W	00h	RESET_SYS

10.10.4.6 Output GPIO[240:276] Register

Offset	390h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31	Reserved	R	-	-
30:24	GPIO[276:270] Output	R/W	00h	RESET_SYS
23:16	GPIO[267:260] Output	R/W	00h	RESET_SYS

Offset	390h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
15:8	GPIO[257:250] Output	R/W	00h	RESET_SYS
7:0	GPIO[247:240] Output	R/W	00h	RESET_SYS

11.0 WATCHDOG TIMER (WDT)

11.1 Introduction

The function of the Watchdog Timer is to provide a mechanism to detect if the internal embedded controller has failed.

When enabled, the Watchdog Timer (WDT) circuit will generate a WDT Event if the user program fails to reload the WDT within a specified length of time known as the WDT Interval.

11.2 References

No references have been cited for this chapter.

11.3 Terminology

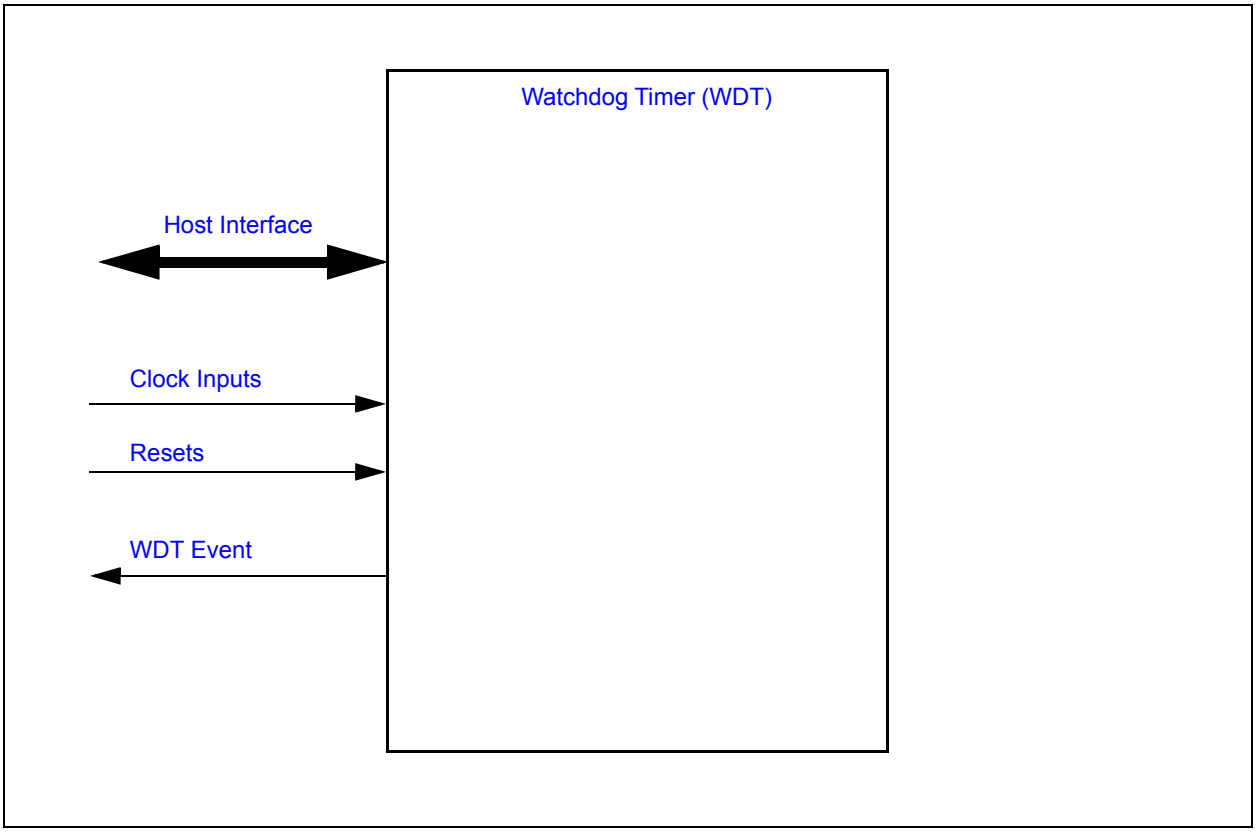
There is no terminology defined for this chapter.

11.4 Interface

This block is designed to be accessed internally via a registered host interface or externally via the signal interface.

11.5 Host Interface

FIGURE 11-1: I/O DIAGRAM OF BLOCK



The registers defined for the Watchdog Timer (WDT) are accessible by the embedded controller as indicated in Section 11.8, "EC Registers". All registers accesses are synchronized to the host clock and complete immediately. Register reads/writes are not delayed by the 32KHz.

11.6 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

11.6.1 POWER DOMAINS

Name	Description
VTR	The logic and registers implemented in this block reside on this single power well.

11.6.2 CLOCK INPUTS

Name	Description
32KHz	The 32KHz clock input is the clock source to the Watchdog Timer functional logic, including the counter.

11.6.3 RESETS

TABLE 11-1: RESET INPUTS

Name	Description
RESET_SYS	Power on Reset to the block. This signal resets all the register and logic in this block to its default state following a POR or a WDT Event event.
RESET_SYS_nWDT	This reset signal is used on WDT registers/bits that need to be preserved through a WDT Event.

TABLE 11-2: RESET OUTPUTS

Source	Description
WDT Event	Pulse generated when WDT expires. This signal is used to reset the embedded controller and its subsystem. The event is cleared after a RESET_SYS.

11.7 Description

11.7.1 WDT OPERATION

11.7.1.1 WDT Activation Mechanism

The WDT is activated by the following sequence of operations during normal operation:

1. Load the WDT Load Register with the count value.
2. Set the WDT_ENABLE bit in the WDT Control Register.

The WDT Activation Mechanism starts the WDT decrementing counter.

11.7.1.2 WDT Deactivation Mechanism

The WDT is deactivated by the clearing the WDT_ENABLE bit in the WDT Control Register. The WDT Deactivation Mechanism places the WDT in a low power state in which clock are gated and the counter stops decrementing.

11.7.1.3 WDT Reload Mechanism

The WDT must be reloaded within periods that are shorter than the programmed watchdog interval; otherwise, the WDT will underflow and a WDT Event will be generated and the WDT bit in Power-Fail and Reset Status Register on page 339 will be set. It is the responsibility of the user program to continually execute code which reloads the watchdog timer, causing the counter to be reloaded.

There are three methods of reloading the WDT: a write to the WDT Load Register, a write to the WDT Kick Register, or WDT event.

11.7.1.4 WDT Interval

The WDT Interval is the time it takes for the WDT to decrements from the WDT Load Register value to 0000h. The WDT Count Register value takes 33/32KHz seconds (ex. 33/32.768 KHz = 1.007ms) to decrement by 1 count.

11.7.1.5 WDT STALL Operation

There are three STALL_ENABLE inputs to the WDT, each of which is connected to an internal signal. If enabled, and the STALL event is asserted, the WDT stops decrementing, and the WDT enters a low power state. When a WDT STALL event is de-asserted, the counter continues decrementing from the value it had when the STALL was asserted.

11.8 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the Watchdog Timer (WDT) Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 11-3: REGISTER SUMMARY

Offset	Register Name
00h	WDT Load Register
04h	WDT Control Register
08h	WDT Kick Register
0Ch	WDT Count Register

11.8.1 WDT LOAD REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
15:0	WDT_LOAD Writing this field reloads the Watch Dog Timer counter.	R/W	FFFFh	RESET_SYS

11.8.2 WDT CONTROL REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:5	Reserved	R	-	-
4	JTAG_STALL This bit enables the WDT Stall function if JTAG or SWD debug functions are active 1=The WDT is stalled while either JTAG or SWD is active 0=The WDT is not affected by the JTAG debug interface	R/W	0b	RESET_SYS
3	WEEK_TIMER_STALL This bit enables the WDT Stall function if the Week Timer is active. 1=The WDT is stalled while the Week Timer is active 0=The WDT is not affected by the Week Timer	R/W	0b	RESET_SYS
2	HIBERNATION_TIMER0_STALL This bit enables the WDT Stall function if the Hibernation Timer 0 is active. 1=The WDT is stalled while the Hibernation Timer 0 is active 0=The WDT is not affected by Hibernation Timer 0	R/W	0b	RESET_SYS
1	TEST	R	0b	RESET_SYS
0	WDT_ENABLE In WDT Operation, the WDT is activated by the sequence of operations defined in Section 11.7.1.1, "WDT Activation Mechanism" and deactivated by the sequence of operations defined in Section 11.7.1.2, "WDT Deactivation Mechanism". 1=block enabled 0=block disabled	R/W	0b	RESET_SYS

11.8.3 WDT KICK REGISTER

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	KICK The WDT Kick Register is a strobe. Reads of this register return 0. Writes to this register cause the WDT to reload the WDT Load Register value and start decrementing when the WDT_ENABLE bit in the WDT Control Register is set to '1'. When the WDT_ENABLE bit in the WDT Control Register is cleared to '0', writes to the WDT Kick Register have no effect.	W	n/a	RESET_SYS

11.8.4 WDT COUNT REGISTER

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
15:0	WDT_COUNT This read-only register provide the current WDT count.	R	FFFFh	RESET_SYS

12.0 BASIC TIMER

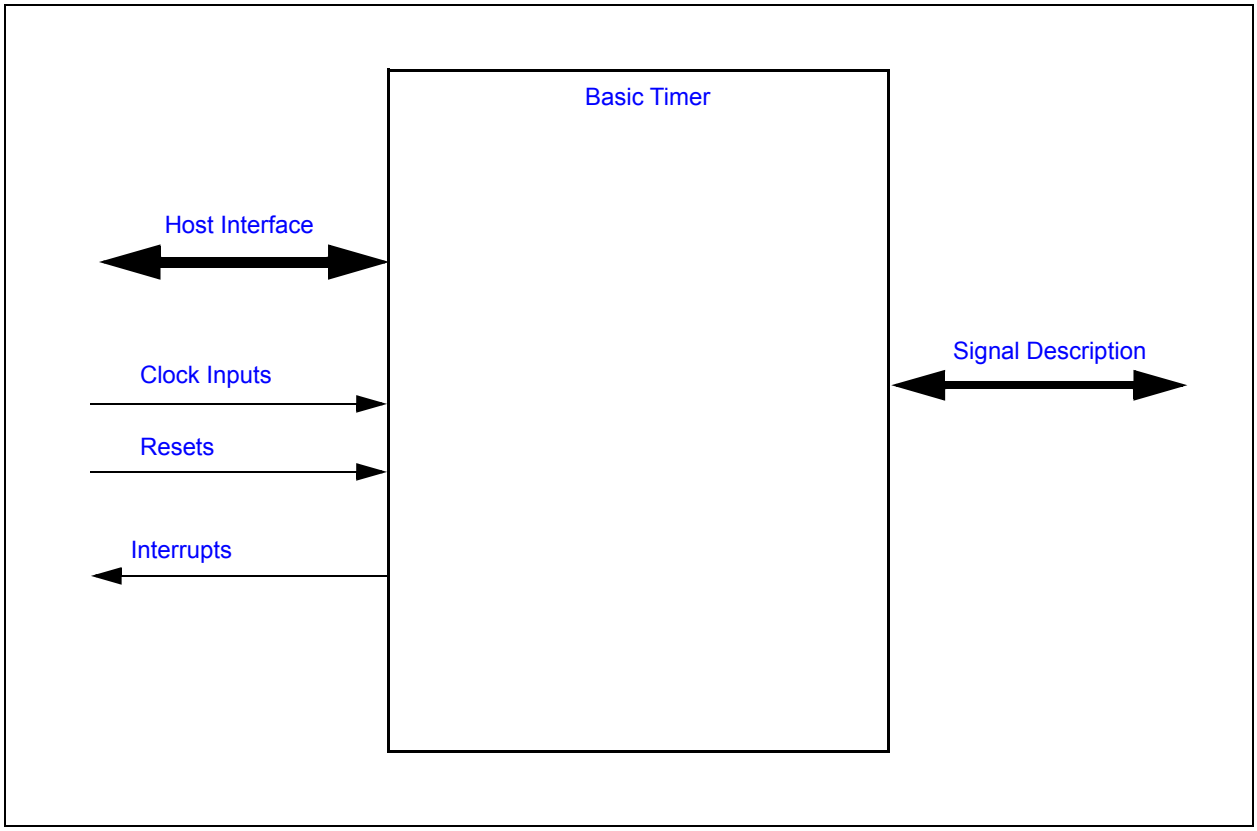
12.1 Introduction

This timer block offers a simple mechanism for firmware to maintain a time base. This timer may be instantiated as 16 bits or 32 bits. The name of the timer instance indicates the size of the timer.

12.2 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

FIGURE 12-1: I/O DIAGRAM OF BLOCK



12.3 Signal Description

There are no external signals for this block.

12.4 Host Interface

The embedded controller may access this block via the registers defined in Section 12.9, "EC Registers".

12.5 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

12.5.1 POWER DOMAINS

TABLE 12-1: POWER SOURCES

Name	Description
VTR	The timer control logic and registers are all implemented on this single power domain.

12.5.2 CLOCK INPUTS

TABLE 12-2: CLOCK INPUTS

Name	Description
48MHz	This is the clock source to the timer logic. The Pre-scaler may be used to adjust the minimum resolution per bit of the counter.

12.5.3 RESETS

TABLE 12-3: RESET SIGNALS

Name	Description
RESET_SYS	This reset signal, which is an input to this block, resets all the logic and registers to their initial default state.
Soft Reset	This reset signal, which is created by this block, resets all the logic and registers to their initial default state. This reset is generated by the block when the **SOFT_RESET** bit is set in the Timer Control Register register.
RESET_Timer	This reset signal, which is created by this block, is asserted when either the RESET_SYS or the Soft Reset signal is asserted. The RESET_SYS and Soft Reset signals are OR'd together to create this signal.

12.6 Interrupts

TABLE 12-4: EC INTERRUPTS

Source	Description
TIMER_16_*x*	This interrupt event fires when a 16-bit timer *x* reaches its limit. • If configured to count up, the limit is triggered when the counter wraps from FFFFh to 0h • If configured to count down, the limit is triggered when the counter wraps from 0h to FFFFh. This event is sourced by the EVENT_INTERRUPT status bit if enabled.
TIMER_32_*x*	This interrupt event fires when a 32-bit timer *x* reaches its limit. • If configured to count up, the limit is triggered when the counter wraps from FFFF_FFFFh to 0h • If configured to count down, the limit is triggered when the counter wraps from 0h to FFFF_FFFFh. This event is sourced by the EVENT_INTERRUPT status bit if enabled.

12.7 Low Power Modes

The Basic Timer may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry. This block is only be permitted to enter low power modes when the block is not active.

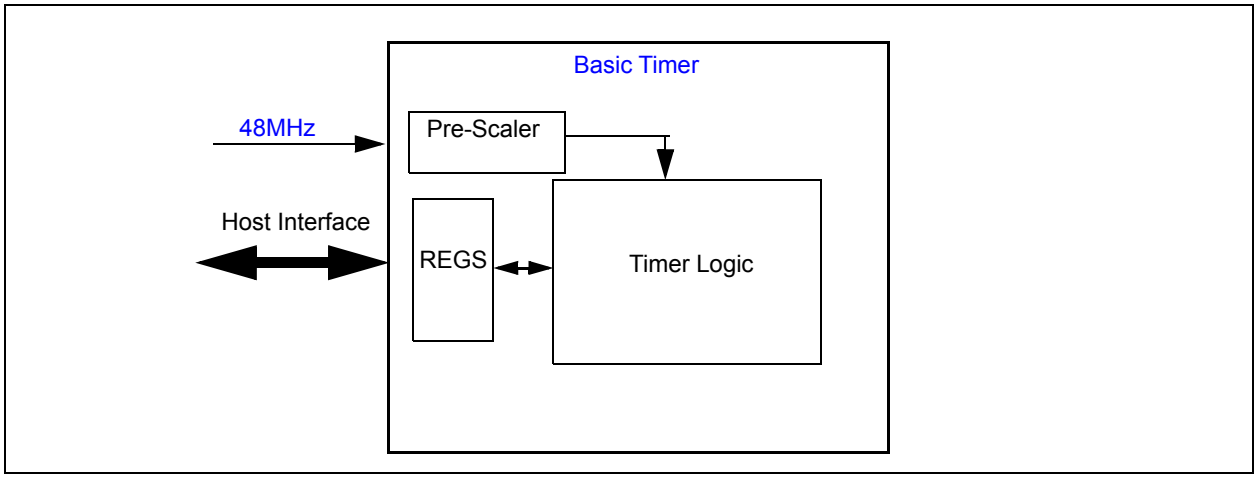
The sleep state of this timer is as follows:

- Asleep while the block is not Enabled
- Asleep while the block is not running (start inactive).
- Asleep while the block is halted (even if running).

The block is active while start is active.

12.8 Description

FIGURE 12-2: BLOCK DIAGRAM



This timer block offers a simple mechanism for firmware to maintain a time base in the design. The timer may be enabled to execute the following features:

- Programmable resolution per LSB of the counter via the Pre-scale bits in the Timer Control Register
- Programmable as either an up or down counter
- One-shot or Continuous Modes
- In one-shot mode the Auto Restart feature stops the counter when it reaches its limit and generates a level event.
- In Continuous Mode the Auto Restart feature restarts that counter from the programmed preload value and generates a pulse event.
- Counter may be reloaded, halted, or started via the Timer Control register
- Block may be reset by either a Power On Reset (POR) or via a Soft Reset.

12.9 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the Basic Timer Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 12-5: REGISTER SUMMARY

Offset	Register Name
00h	Timer Count Register
04h	Timer Preload Register
08h	Timer Status Register
0Ch	Timer Int Enable Register
10h	Timer Control Register

12.9.1 TIMER COUNT REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	COUNTER This is the value of the Timer counter. This is updated by Hardware but may be set by Firmware. If it is set by firmware while the Hardware Timer is operating, functionality cannot be assured. When read, it is buffered so single byte reads will be able to catch the full 4 byte register without it changing. The size of the Counter is indicated by the instance name (e.g., 16-bit Basic Timer -> SIZE=16). Bits 0 to (SIZE-1) are r/w counter bits. Bits 31 down to SIZE are unused and should be set to zero when writing this register.	R/W	0h	RESET_Timer

12.9.2 TIMER PRELOAD REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	PRE_LOAD This is the value of the Timer pre-load for the counter. This is used by H/W when the counter is to be restarted automatically; this will become the new value of the counter upon restart. The size of the Pre-Load value is the same as the size of the counter. The size of the Counter is indicated by the instance name (e.g., 16-bit Basic Timer -> SIZE=16). Bits 0 to (SIZE-1) are r/w pre-load bits. Bits 31 down to SIZE are unused and should be set to zero when writing this register.	R/W	0h	RESET_Timer

12.9.3 TIMER STATUS REGISTER

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	Reserved	R	-	-
0	EVENT_INTERRUPT This is the interrupt status that fires when the timer reaches its limit. This may be level or a self clearing signal cycle pulse, based on the AUTO_RESTART bit in the Timer Control Register. If the timer is set to automatically restart, it will provide a pulse, otherwise a level is provided.	R/WC	0h	RESET_Timer

12.9.4 TIMER INT ENABLE REGISTER

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	Reserved	R	-	-
0	EVENT_INTERRUPT_ENABLE This is the interrupt enable for the status EVENT_INTERRUPT bit in the Timer Status Register	R/W	0h	RESET_Timer

12.9.5 TIMER CONTROL REGISTER

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	PRE_SCALE This is used to divide down the system clock through clock enables to lower the power consumption of the block and allow slow timers. Updating this value during operation may result in erroneous clock enable pulses until the clock divider restarts. The number of clocks per clock enable pulse is (Value + 1); a setting of 0 runs at the full clock speed, while a setting of 1 runs at half speed.	R/W	0h	RESET_Timer
15:8	Reserved	R	-	-
7	HALT This is a halt bit. This will halt the timer as long as it is active. Once the halt is inactive, the timer will start from where it left off. 1=Timer is halted. It stops counting. The clock divider will also be reset. 0=Timer runs normally	R/W	0h	RESET_Timer
6	RELOAD This bit reloads the counter without interrupting it operation. This will not function if the timer has already completed (when the START bit in this register is ‘0’). This is used to periodically prevent the timer from firing when an event occurs. Usage while the timer is off may result in erroneous behavior.	R/W	0h	RESET_Timer

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
5	START This bit triggers the timer counter. The counter will operate until it hits its terminating condition. This will clear this bit. It should be noted that when operating in restart mode, there is no terminating condition for the counter, so this bit will never clear. Clearing this bit will halt the timer counter. Setting this bit will: • Reset the clock divider counter. • Enable the clock divider counter. • Start the timer counter. • Clear all interrupts. Clearing this bit will: • Disable the clock divider counter. • Stop the timer counter.	R/W	0h	RESET_Timer
4	SOFT_RESET This is a soft reset. This is self clearing 1 cycle after it is written. Firmware does not need to wait before reconfiguring the Basic Timer following soft reset.	WO	0h	RESET_Timer
3	AUTO_RESTART This will select the action taken upon completing a count. 1=The counter will automatically restart the count, using the contents of the Timer Preload Register to load the Timer Count Register The interrupt will be set in edge mode 0=The counter will simply enter a done state and wait for further control inputs. The interrupt will be set in level mode.	R/W	0h	RESET_Timer
2	COUNT_UP This selects the counter direction. When the counter in incrementing the counter will saturate and trigger the event when it reaches all F's. When the counter is decrementing the counter will saturate when it reaches 0h. 1=The counter will increment 0=The counter will decrement **Note:** Counter will saturate when it reaches the max count, which is dependent on the size of the counter. Examples: 16-bit timer 0=saturate and transition from 00h to FFh 1=saturate and transition from FFh to 00h 32-bit timer 0=saturate and transition from 0000h to FFFFh 1=saturate and transition from FFFFh to 0000h	R/W	0h	RESET_Timer

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
1	Reserved	R	-	-
0	ENABLE This enables the block for operation. 1=This block will function normally 0=This block will gate its clock and go into its lowest power state	R/W	0h	RESET_Timer

13.0 16-BIT COUNTER-TIMER INTERFACE

13.1 Introduction

The 16-Bit Counter-Timer Interface implements four 16-bit auto-reloading timer/counters. The clock for each timer/counter is derived from the system clock and can be divided down by a prescaler. Input-Only and Input/Output timers can also use an external input pin to clock or gate the counter. To aid operation in noisy environments the external input pin also has a selectable noise filter. If large counts are required, the output of each timer/counter can be internally connected to the next timer/counter.

13.2 References

No references have been cited for this feature.

13.3 Terminology

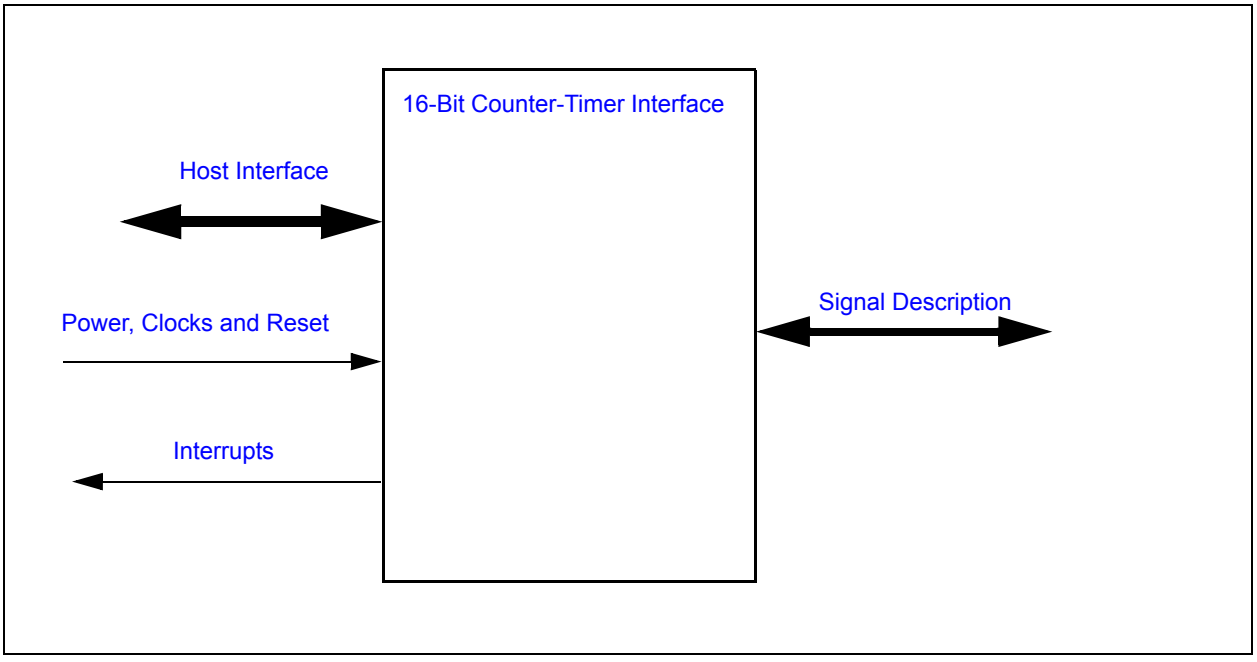
TABLE 13-1: TERMINOLOGY

Term	Definition
Overflow	When the timer counter transitions from FFFFh to 0000h
Underflow	When the timer counter transitions from 0000h to FFFFh.
Timer Tick Rate	This is the rate at which the timer is incremented or decremented.

13.4 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

FIGURE 13-1: I/O DIAGRAM OF BLOCK



13.5 Signal Description

TABLE 13-2: SIGNAL DESCRIPTION TABLE

Name	Direction	Description
TINx	INPUT	Timer x Input signal
TOUTx	OUTPUT	Timer x Output signal

13.6 Host Interface

The registers defined for 16-bit Timers are accessible by the various hosts as indicated in Section 13.11, "EC Registers".

13.7 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

13.7.1 POWER DOMAINS

Name	Description
VTR	The logic and registers implemented in this block are powered by this power well.

13.7.2 CLOCK INPUTS

Name	Description
48MHz	This is the clock source for this block.

13.7.3 RESETS

Name	Description
RESET_SYS	This signal resets all the registers and logic in this block to their default state.
Soft Reset	This reset signal, which is created by this block, resets all the logic and registers to their initial default state. This reset is generated by the block when the RESET bit is set in the Timer x Control Register.
Reset_Timer	This reset signal, which is created by this block, is asserted when either the RESET_SYS or the Soft Reset signal is asserted. The RESET_SYS and Soft Reset signals are OR'd together to create this signal.

13.8 Interrupts

This section defines the Interrupt Sources generated from this block.

Source	Description
TIMER*x*	This interrupt event fires when a 16-bit timer *x* overflows or underflows.

13.9 Low Power Modes

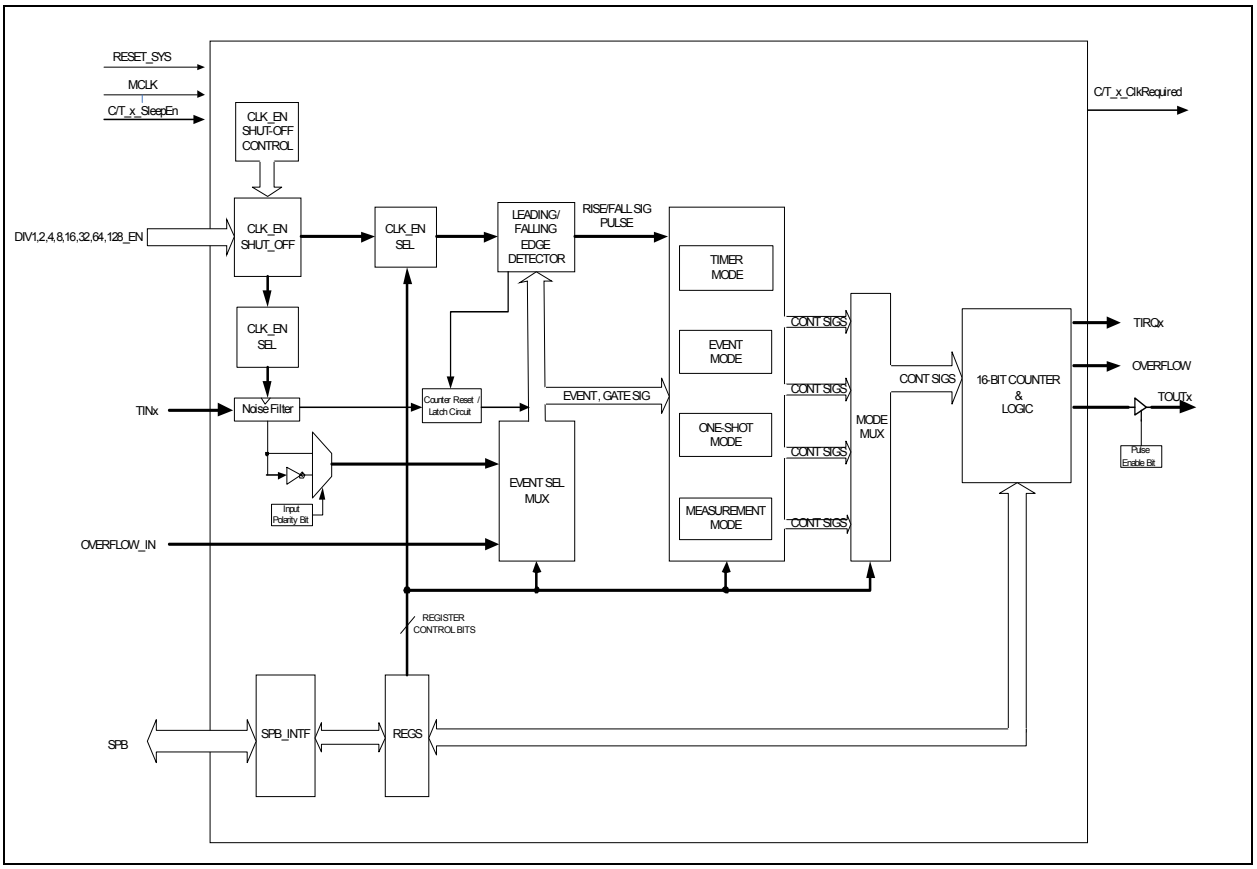
The 16-bit Timer may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry. This block is only be permitted to enter low power modes when the block is not active. The block is inactive in the following conditions:

- The block is not running (ENABLE de-asserted)
- The block is powered down (PD asserted).

The timer requires one Timer Clock period to halt after receiving a Sleep_En signal. When the block returns from sleep, if enabled, it will be restarted from the preload value.

13.10 Description

FIGURE 13-2: BLOCK DIAGRAM FOR TIMER X



The 16-bit Timer consists of a 16-bit counter, clocked by a by a configurable Timer Clock. The Timer can operate in any of 4 Modes: Timer Mode, Event Mode, One-Shot Mode, and Measurement Mode. The Timer can be used to generate an interrupt to the EC. Depending on the mode, the Timer can also generate an output signal.

13.10.1 TIMER CLOCK

Any of the frequencies listed in Table 13-3 may be used as the time base for the 16-bit counter.

TABLE 13-3: TIMER CLOCK FREQUENCIES

Timer Clock Select	Frequency Divide Select	Frequency Selected
0000b	Divide by 1	48MHz
0001b	Divide by 2	24MHz
0010b	Divide by 4	12MHz

TABLE 13-3: TIMER CLOCK FREQUENCIES (CONTINUED)

Timer Clock Select	Frequency Divide Select	Frequency Selected
0011b	Divide by 8	6MHz
0100b	Divide by 16	3MHz
0101b	Divide by 32	1.5MHz
0110b	Divide by 64	750KHz
0111b	Divide by 128	375KHz
1xxxb	Reserved	Reserved

For the Timer Clock, the **Timer Clock Select** value is defined by the TCLK field in the Timer x Clock and Event Control Register

13.10.2 FILTER CLOCK AND NOISE FILTER

The noise filter uses the Filter Clock (FCLK) to filter the signal on the TINx pins. for Event Mode and One-Shot Mode.

In Event Mode, the Event input is synchronized to FCLK and (if enabled) filtered by a three stage filter. The resulting recreated clock is used to clock the timer in Event mode. In Bypass Mode, configured by the FILTER_BYPASS bit in the Timer x Control Register, the pulse width of the external signal must be at least 2x the pulse width of the FCLK source. In Filter Mode, the pulse width of the external signal must be at least 4x the pulse width of the sync and filter clock

In One-Shot mode, the TIN duration could be smaller than a TCLK period. The filtered signal is latched until the signal is seen in the TCLK domain. This also applies in the filter bypass mode

Frequencies for the Filter Clock are the as those available for the Timer Clock, and are listed in Table 13-3. For the Filter Clock, the **Timer Clock Select** value is defined by the FCLK field in the Timer x Clock and Event Control Register. The choice of frequency is independent of the value chosen for the Timer Clock.

13.10.3 TIMER CONNECTIONS

For external inputs/outputs (TINx/TOUTx) to/from timers, please see Pin Configuration chapter for a description of the 16-bit Counter/Timer Interface.

TABLE 13-4: TIMER CASCADING DESCRIPTION

Timer Name	Timer Type	Over-Flow/ Under-flow Input's Connection
Timer 0	General Purpose	from Timer 3
Timer 1	General Purpose	from Timer 0
Timer 2	General Purpose	from Timer 1
Timer 3	General Purpose	from Timer 2

Note: The cascading connections are independent of the TINx/TOUTx connections.

13.10.4 STARTING AND STOPPING

The 16-bit timers can be started and stopped by setting and clearing the ENABLE bit in the Timer x Control Register in all modes, except one-shot.

13.10.5 TIMER MODE

Timer mode is used to generate periodic interrupts to the EC. When operating in this mode the timer always counts down based on one of the internally generated clock sources. The Timer mode is selected by setting the Timer Mode Select bits in the Timer Control Register. See Section 13.11.1, "Timer x Control Register".

The period between timer interrupts and the width of the output pulse is determined by the speed of the clock source, the clock divide ratio and the value programmed into the Timer Reload Register. The timer clock source and clock rate are selected using the Clock Source Select bits (TCLK) in the Timer x Clock and Event Control Register. See Section 13.11.2, "Timer x Clock and Event Control Register".

TABLE 13-5: TIMER MODE OPERATIONAL SUMMARY

Item	Description
Timer Clock Frequencies	This mode supports all the programmable frequencies listed in Table 13-3, "Timer Clock Frequencies"
Filter Clock Frequencies	This mode supports all the programmable frequencies listed in Table 13-3, "Timer Clock Frequencies"
Count Operation	Down Counter
Reload Operation	When the timer underflows: RLOAD = 1, timer reloads from Timer Reload Reg RLOAD = 0, timer rolls over to FFFFh.
Count Start Condition	UPDN = 0 (timer only mode): ENABLE = 1 UPDN = 1 (timer gate mode): ENABLE = 1 & TIN = 1;
Count Stop Condition	UPDN = 0=ENABLE = 0; UPDN = 1: (ENABLE= 0 | TIN = 0)
Interrupt Request Generation Timing	When timer underflows from 0000h to reload value (as determined by RLOAD) an interrupt is generated.
TINx Pin Function	Provides timer gate function
TOUTx Pin Function	TOUT toggles each time the timer underflows (if enabled).
Read From Timer	Current count value can be read by reading the Timer Count Register
Write to Preload Register	After the firmware writes to the Timer Reload Register asserting the RESET loads the timer with the new value programmed in the Timer Reload Register. Note: If the firmware does not assert RESET, the timer will automatically load the Timer Reload Register value when the timer underflows. When the timer is running, values written to the Timer Reload Register are written to the timer counter when the timer underflows. The assertion of Reset also copies the Timer Reload Register into the timer counter.
Selectable Functions	• Reload timer on underflow with programmed Preload value (Basic Timer) • Reload timer with FFFFh in Free Running Mode (Free-running Timer) • Timer can be started and stopped by the TINx input pin (Gate Function) • The TOUTx pin changes polarity each time the timer underflows (Pulse Output Function)

13.10.5.1 Timer Mode Underflow

The timer operating in Timer mode can underflow in two different ways. One method, the Reload mode shown in Figure 13-3, is to reload the value programmed into the Reload register and continue counting from this value. The second method, Free Running mode Figure 13-4, is to set the timer to FFFFh and continue counting from this value. The underflow behavior is controlled by the RLOAD bit in the Timer Control Register.

FIGURE 13-3: RELOAD MODE BEHAVIOR

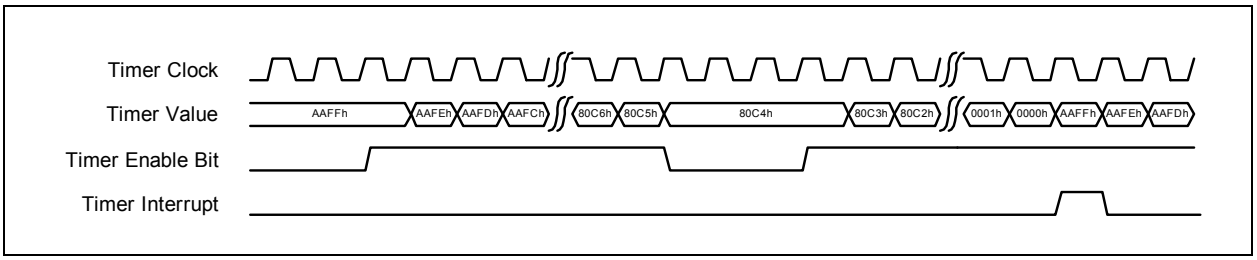
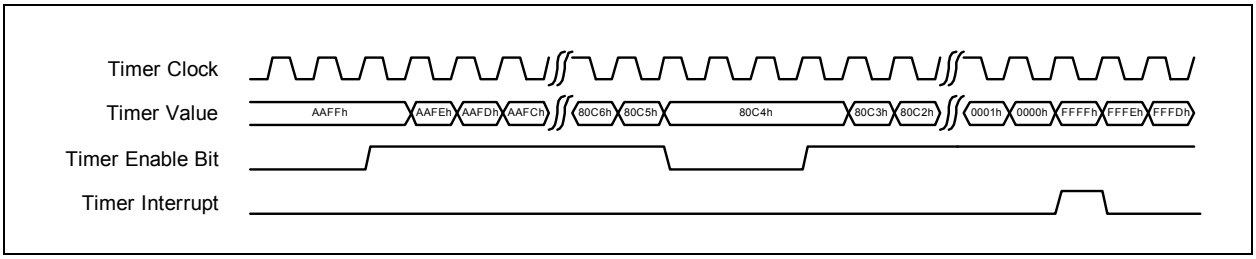


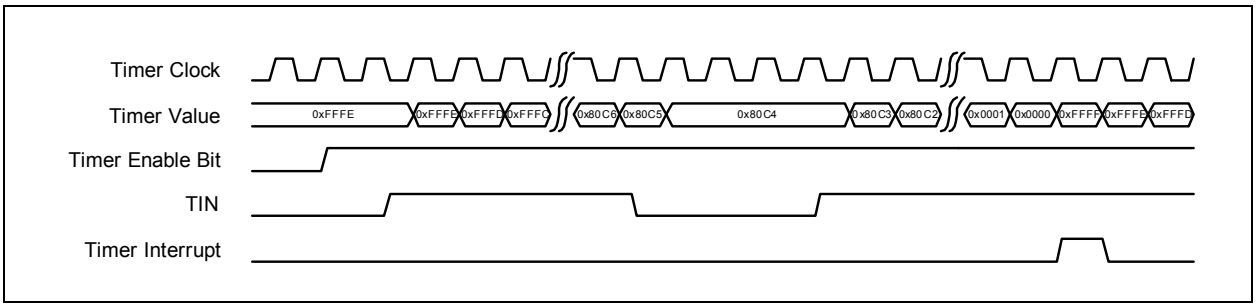
FIGURE 13-4: FREE RUNNING MODE BEHAVIOR



13.10.5.2 Timer Gate Function

The TIN pin on each timer can be used to pause the timer's operation when the timer is running. The timer will stop counting when the TIN pin is deasserted and count when the TIN pin is asserted. Figure 13-5 shows the timer behavior when the TIN pin is used to gate the timer function. The UPDN bit is used to enable and disable the Timer Gate function when in the Timer mode.

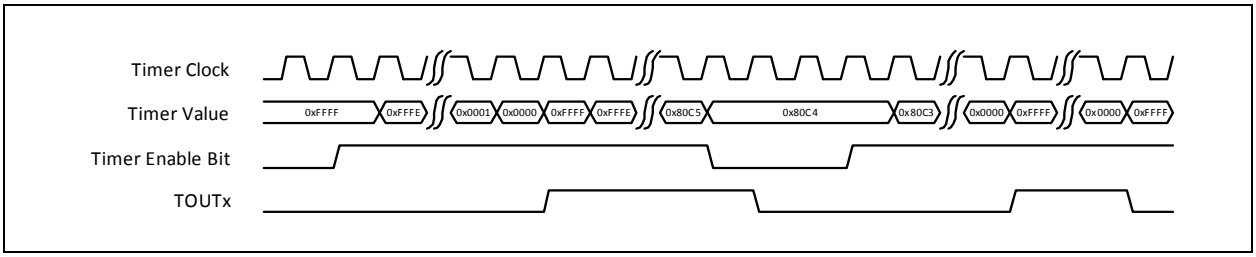
FIGURE 13-5: TIMER GATE OPERATION



13.10.5.3 Timer Mode Pulse Output

The four Timers can be used to generate a periodic output pulse. The output pulse changes state each time the timer underflows. The output is also cleared when the EN bit is cleared. Figure 13-6 shows the behavior of the TOUTx pin when it is used as a pulse output pin.

FIGURE 13-6: TIMER PULSE OUTPUT



13.10.6 EVENT MODE

Event mode is used to count events that occur external to the timer. The timer can be programmed to count the overflow output from the previous timer or an edge on the TIN pin. The direction the timer counts in Event mode is controlled by the UPDN bit in the Timer Control Register. When the timer is in Event mode, the TOUTx signal can be used to generate a periodic output pulse when the timer overflows or underflows. Figure 13-6 illustrates the pulse output behavior of the TOUTx pin in event mode when the timer underflows.

The timer can be programmed using the Clock and Event Control register to respond to the following events using the EVENT bits and the EDGE bits: rising edge of TINx, falling edge of TINx, rising and falling edge of TINx, rising edge of overflow input, falling edge of the overflow input, and the rising and falling edges of the overflow input.

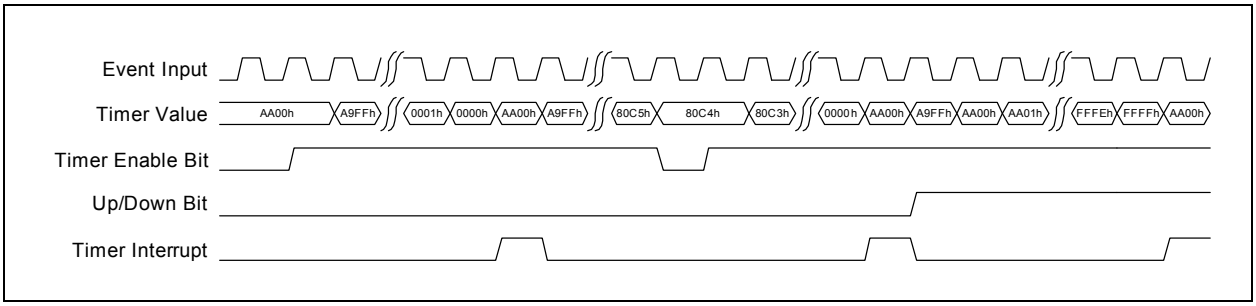
TABLE 13-6: EVENT MODE OPERATIONAL SUMMARY

Item	Description
Count Source	• External signal input to TINx pin (effective edge can be selected by software) • Timer x-1 overflow
Timer Clock Frequencies	This mode supports all the programmable frequencies listed in Table 13-3, "Timer Clock Frequencies"
Filter Clock Frequencies	This mode supports all the programmable frequencies listed in Table 13-3, "Timer Clock Frequencies"
Count Operation	Up/Down Counter
Reload Operation	• When the timer underflows: RLOAD = 1, timer reloads from Timer Reload Reg RLOAD = 0, timer rolls over to FFFFh. • When the timer overflows: RLOAD = 1, timer reloads from Timer Reload Reg RLOAD = 0, timer rolls over to 0000h.
Count Start Condition	Timer Enable is set (ENABLE = 1)
Count Stop Condition	Timer Enable is cleared (ENABLE = 0)
Interrupt Request Generation Timing	When timer overflows or underflows
TINx Pin Function	Event Generation
TOUTx Pin Function	TOUT toggles each time the timer underflows/overflows (if enabled).
Read From Timer	Current count value can be read by reading the Timer Count Register
Write to Preload Register	After the firmware writes to the Timer Reload Register, asserting the RESET loads the timer with the new value programmed in the Timer Reload Register. Note: If the firmware does not assert RESET, the timer will automatically load the Timer Reload Register value when the timer underflows.
Selectable Functions	• The direction of the counter is selectable via the UPDN bit. • Reload timer on underflow/overflow with programmed Preload value (Basic Timer) • Reload timer with FFFFh in Free Running Mode (Free-running Timer) • Pulse Output Function The TOUTx pin changes polarity each time the timer underflows or overflows.

13.10.6.1 Event Mode Operation

The timer starts counting events when the ENABLE bit in the Timer Control Register is set and continues to count until the ENABLE bit is cleared. When the ENABLE bit is set, the timer continues counting from the current value in the timer except after a reset event. After a reset event, the timer always starts counting from the value programmed in the Reload Register if counting down or from 0000h if counting up. Figure 13-7 shows an example of timer operation in Event mode. The RLOAD bit controls the behavior of the timer when it underflows or overflows.

FIGURE 13-7: EVENT MODE OPERATION



13.10.7 ONE-SHOT MODE

The One-Shot mode of the timer is used to generate a single interrupt to the EC after a specified amount of time. The timer can be configured to start using the ENABLE bit (Figure 13-8) or on a timer overflow event from the previous timer. See Section 13.11.2, "Timer x Clock and Event Control Register" for configuration details. The ENABLE bit must be set for an event to start the timer. The ENABLE bit is cleared one clock after the timer starts. The timer always starts from the value in the Reload Register and counts down in One-Shot mode.

TABLE 13-7: ONE SHOT MODE OPERATIONAL SUMMARY

Item	Description
Timer Clock Frequencies	This mode supports all the programmable frequencies listed in Table 13-3, "Timer Clock Frequencies"
Filter Clock Frequencies	This mode supports all the programmable frequencies listed in Table 13-3, "Timer Clock Frequencies"
Count Operation	Down Counter
Reload Operation	When the timer underflows the timer will stop. When the timer is enabled timer starts counting from value programmed in Timer Reload Register. (RLOAD has no effect in this mode)
Count Start Condition	Setting the ENABLE bit to 1 starts One-Shot mode. The timer clock automatically clears the enable bit one timer tick later. One-Shot mode may be enabled in Event Mode. In Event mode an overflow from the previous timer is used for timer tick rate.
Count Stop Condition	• Timer is reset (RESET = 1) • Timer underflows
Interrupt Request Generation Timing	When an underflow occurs.
TINx Pin Function	One Shot External input
TOUTx Pin Function	The TOUTx pin is asserted when the timer starts and de-asserted when the timer stops
Read From Timer	Current count value can be read by reading the Timer Count Register
Write to Preload Register	After the firmware writes to the Timer Reload Register, asserting the RESET loads the timer with the new value programmed in the Timer Reload Register. Note: If the firmware does not assert RESET, the timer will automatically load the Timer Reload Register value when the timer underflows.
Selectable Functions	• Pulse Output Function The TOUTx pin is asserted when the timer starts and de-asserted when the timer stops.

FIGURE 13-8: TIMER START BASED ON ENABLE BIT

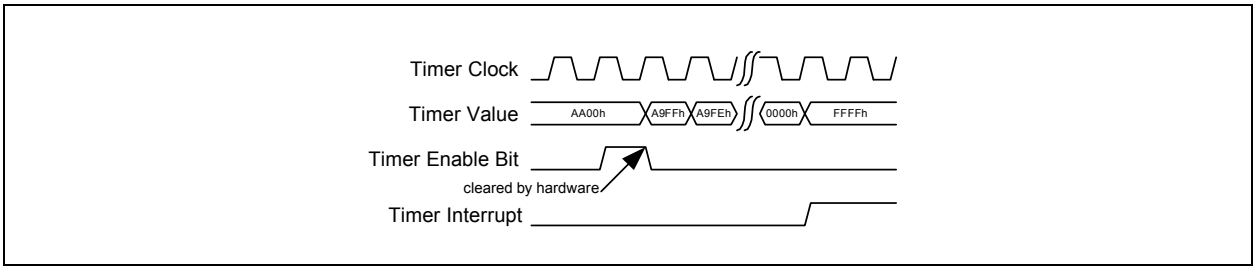


FIGURE 13-9: TIMER START BASED ON EXTERNAL EVENT

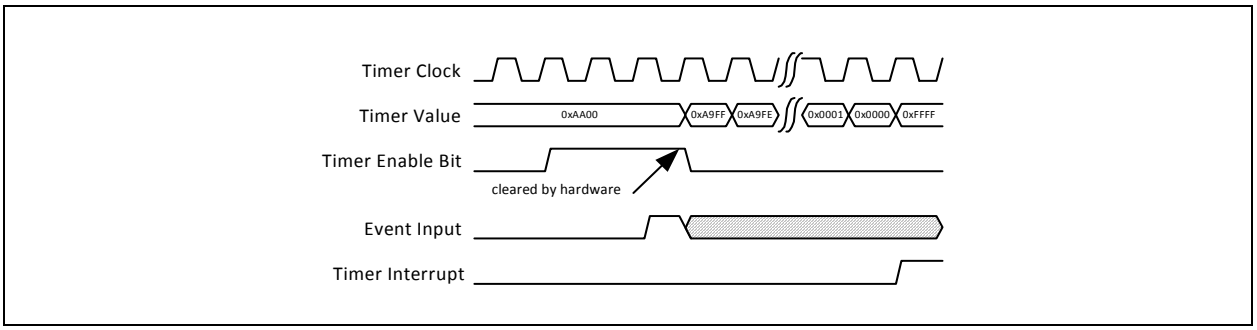
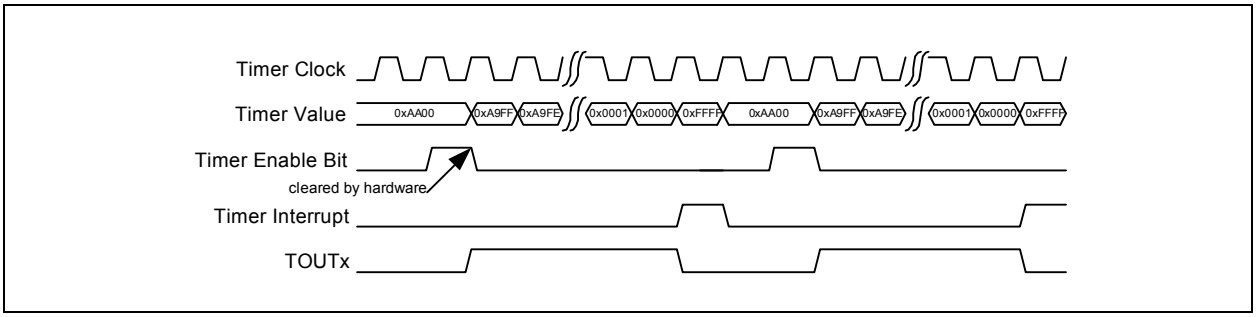


FIGURE 13-10: ONE SHOT TIMER WITH PULSE OUTPUT



13.10.8 MEASUREMENT MODE

The Measurement mode is used to measure the pulse width or period of an external signal. An interrupt to the EC is generated after each measurement or if the timer overflows and no measurement occurred. The timer measures the pulse width or period by counting the number of clock between edges on the TINx pin. The timer always stars counting at zero and counts up to 0xFFFF. The accuracy of the measurement depends on the speed of the clock being used. The speed of the clock also determines the maximum pulse width or period that can be detected.

TABLE 13-8: MEASUREMENT MODE OPERATIONAL SUMMARY

Item	Description
Timer Clock Frequencies	This mode supports all the programmable frequencies listed in Table 13-3, "Timer Clock Frequencies"
Filter Clock Frequencies	This mode supports all the programmable frequencies listed in Table 13-3, "Timer Clock Frequencies"

TABLE 13-8: MEASUREMENT MODE OPERATIONAL SUMMARY (CONTINUED)

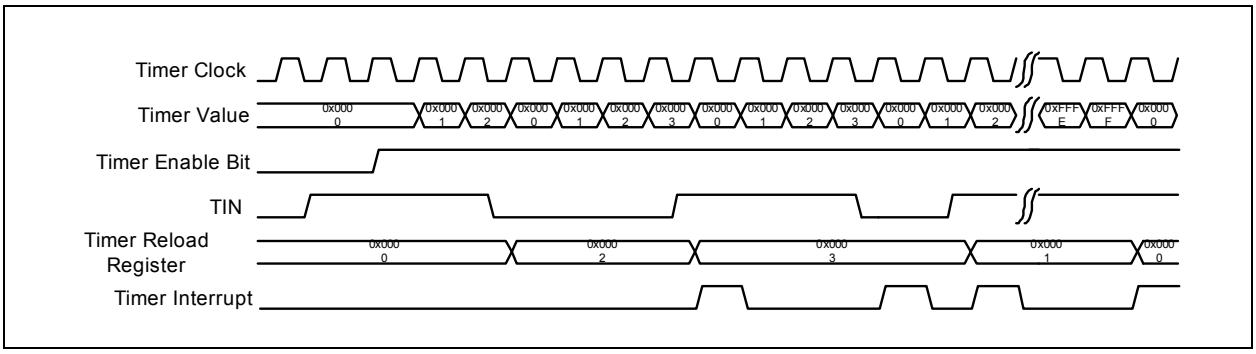
Item	Description
Count Operation	• Up Count • At measurement pulse's effective edge, the count value is transferred to the Timer Reload Register and the timer is loaded with 0000h and continues counting.
Count Start Condition	• Timer enable is set (ENABLE = 1)
Count Stop Condition	• Timer is reset (RESET = 1) • Timer overflows • Timer enable is cleared (ENABLE = 0)
Interrupt Request Generation Timing	• When timer overflows • When a measurement pulse's effective edge is input. (An interrupt is not generated on the first effective edge after the timer is started.)
TINx Pin Function	Programmable Input port or Measurement input
Read From Timer	When the Timer x Reload Register is read it indicates the measurement result from the last measurement made. The Timer x Reload Register reads 0000h if the timer overflows before a measurement is made.
Write to Timer	Timer x Reload Register is Read-Only in Measurement mode

13.10.8.1 Pulse Width Measurements

The timers measure pulse width by counting the number of timer clocks since the last rising or falling edge of the TINx input. To measure the pulse width of a signal on the TINx pin, the EDGE bits in the Clock and Event Control Register, must be set to start counting on rising and falling edges. The timer starts measuring on the next edge (rising or falling) on the TINx pin after the ENABLE bit is set. The Reload register stores the result of the last measurement taken. If the timer overflows, 0x0000 is written to the Reload register and the ENABLE bit is cleared stopping the timer. Figure 13-11 shows the timer behavior when measuring pulse widths.

The timer will not assert an interrupt in Pulse Measurement mode until the timer detects both a rising and a falling edge.

FIGURE 13-11: PULSE WIDTH MEASUREMENT

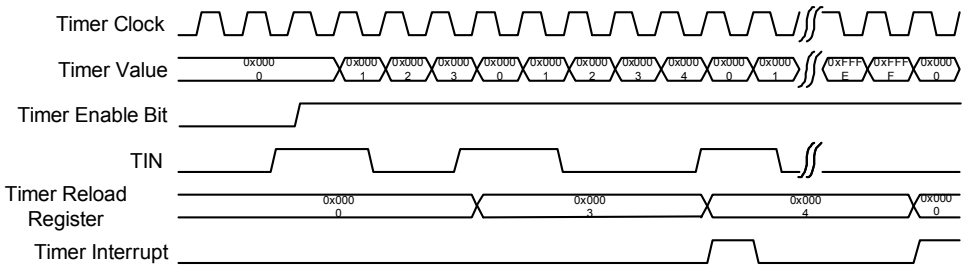


13.10.8.2 Period Measurements

The 16-bit timer measures the period of a signal by counting the number of timer clocks between either rising or falling edges of the TINx input. The measurement edge is determined by the EDGE bits in the Clock and Event Control Register. The timer starts measuring on the next edge (rising or falling) on the TINx pin after the ENABLE bit is set. The reload register stores the result of the last measurement taken. If the timer overflows, 0x0000 is written to the reload register. Figure 13-12 shows the timer behavior when measuring the period of a signal.

The timer will not signal an interrupt in period measurement mode until the timer detects either two rising edges or two falling edges.

FIGURE 13-12: PULSE PERIOD MEASUREMENT



13.11 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the 16-Bit Counter-Timer Interface Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 13-9: REGISTER SUMMARY

Offset	Register Name
00h	Timer x Control Register
04h	Timer x Clock and Event Control Register
08h	Timer x Reload Register
0Ch	Timer x Count Register

13.11.1 TIMER X CONTROL REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:13	Reserved	R	-	-
12	TIMERX_CLK_REQ This bit reflects the current state of the timer's Clock_Required output signal. 1=The main clock is required by this block 0=The main clock is not required by this block	R	0h	Reset_Timer
11	SLEEP_ENABLE This bit reflects the current state of the timer's Sleep_Enable input signal. 1=Normal operation 0=Sleep Mode is requested	R	0h	Reset_Timer

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
10	TOUT_POLARITY This bit determines the polarity of the TOUTx output signal. In timer modes that toggle the TOUTx signal, this polarity bit will not have a perceivable difference, except to determine the inactive state. In One-Shot mode this determines if the pulsed output is active high or active low. 1=Active low 0=Active high	R/W	0h	Reset_Timer
9	PD Power Down. 1=The timer is powered down and all clocks are gated 0=The timer is in a running state	R/W	1h	Reset_Timer
8	FILTER_BYPASS This bit is used to enable or disable the noise filter on the TINx input signal. 1=IBypass Mode: input filter disabled. The TINx input directly affects the timer 0=Filter Mode: input filter enabled. The TINx input is filtered by the input filter	R/W	0h	Reset_Timer
7	RLOAD Reload Control. This bit controls how the timer is reloaded on over-flow or underflow in Event and Timer modes. It has no effect in One Shot mode. 1=Reload timer from Timer Reload Register and continue counting 0=Roll timer over to FFFFh and continue counting when counting down and rolls over to 0000h and continues counting when counting up	R/W	0h	Reset_Timer
6	TOUT_EN This bit enables the TOUTx pin 1=TOUTx pin function is enabled 0=TOUTx pin is inactive	R/W	0h	Reset_Timer
4	UPDN In Event Mode, this bit selects the timer count direction. In Timer Mode enables timer control by the TINx input pin. Event Mode: 1=The timer counts up 0=The timer counts down Timer Mode: 1=TINx pin pauses the timer when de-asserted 0=TINx pin has no effect on the timer	R/W	0h	Reset_Timer

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
4	INPOL This bit selects the polarity of the TINx input 1=TINx is active low 0=TINx is active high	R/W	0h	Reset_ Timer
3:2	MODE Timer Mode. 3=Measurement Mode 2=One Shot Mode 1=Event Mode 0=Timer Mode	R/W	0h	Reset_ Timer
1	RESET This bit stops the timer and resets the internal counter to the value in the Timer Reload Register. This bit also clears the ENABLE bit if it is set. This bit is self-clearing after the timer is reset. Firmware must poll the RESET bit in order to determine when the timer is active after reset. The polling time may be any value from 0 ms to 2^(TCLK+1))/48MHz. If it the TCLK value was set to 0111b then the polling time will be a 5.33us (typ). Worst case polling time is dependent on accuracy of 48MHz clock source. Interrupts are blocked only when RESET takes effect and the ENABLE bit is cleared. If interrupts are not desired, firmware must mask the interrupt in the interrupt block. 1=Timer reset 0=Normal timer operation	R/W	0h	Reset_ Timer
0	ENABLE This bit is used to start and stop the timer. This bit does not reset the timer count but does reset the timer pulse output. This bit will be cleared when the timer stops counting in One-Shot mode. The ENABLE bit is cleared after a RESET cycle has completed. Firmware must poll the RESET bit in order to determine when the timer is active after reset. 1=Timer is enabled 0=Timer is disabled	R/W	0h	Reset_ Timer

13.11.2 TIMER X CLOCK AND EVENT CONTROL REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:12	Reserved	R	-	-
11:8	FCLK Timer Clock Select. This field determines the clock source for the TINx noise filter. See Section 13.10.2, "Filter Clock and Noise Filter" for a description of the available frequencies. The available frequencies are the same as for TCLK.	R/W	0h	Reset_Timer
7	EVENT Event Select. This bit is used to select the count source when the timer is operating in Event Mode. 1=TINx is count source 0=Timer x-1 overflow is count source	R/W	0h	Reset_Timer
6:5	EDGE This field selects which edge of the TINx input signal affects the timer in Event Mode, One-Shot Mode and Measurement Mode. Event Mode: 11b=No event selected 10b=Counts rising and falling edges 01b=Counts rising edges 00b=Counts falling edges One-Shot Mode: 11b=Start counting when the Enable bit is set 10b=Starts counting on a rising or falling edge 01b=Starts counting on a rising edge 00b=Starts counting on a falling edge Measurement Mode: 11b=No event selected 10b=Measures the time between rising edges and falling edges and the time between falling edges and rising edges 01b=Measures the time between rising edges 00b=Measures the time between falling edges	R/W	0h	Reset_Timer
4	Reserved	R	-	-
3:0	TCLK Timer Clock Select. This field determines the clock source for the 16-bit counter in the timer. See Section 13.10.1, "Timer Clock" for a description of the available frequencies.	R/W	0h	Reset_Timer

13.11.3 TIMER X RELOAD REGISTER

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	Reserved	R	-	-
15:0	TIMER_RELOAD The Timer Reload register is used in Timer and One-Shot modes to set the lower limit of the timer. In Event mode the Timer Reload register sets either the upper or lower limit of the timer depending on if the timer is counting up or down. Valid Timer Reload values are 0001h - FFFFh. If the timer is running, the reload value will not be updated until the timer overflows or underflows. Programming a 0000h as a preload value is not a valid count value. Using a value of 0000h will cause unpredictable behavior.	R/W	FFFFh	Reset_ Timer

13.11.4 TIMER X COUNT REGISTER

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	Reserved	R	-	-
15:0	TIMER_COUNT The Timer Count register returns the current value of the timer in all modes.	R	FFFFh	Reset_ Timer

14.0 INPUT CAPTURE AND COMPARE TIMER

14.1 Introduction

The Input Capture and Compare Timers block contains a 32-bit timer running at the main system clock frequency. The timer is free-running and is associated with six 32-bit capture registers and two compare registers. Each capture register can record the value of the free-running timer based on a programmable edge of its associated input pin. An interrupt can be generated for each capture register each time it acquires a new timer value. The timer can also generate an interrupt when it automatically resets and can additionally generate two more interrupts when the timer matches the value in either of two 32-bit compare registers.

14.2 References

No references have been cited for this feature.

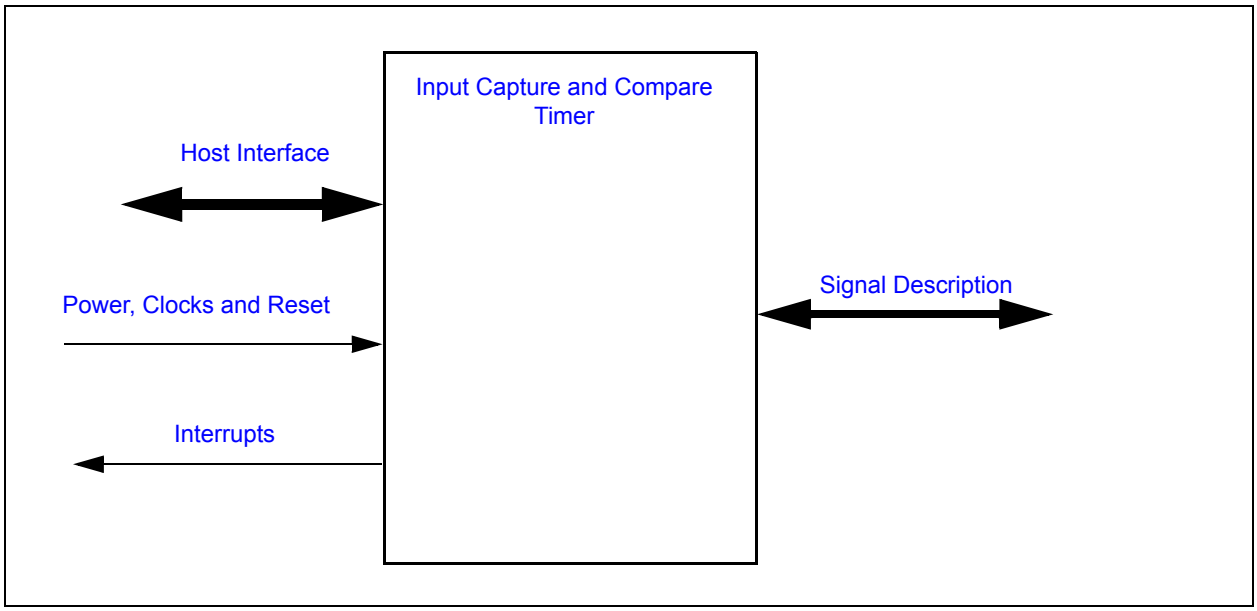
14.3 Terminology

There is no terminology for this block.

14.4 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

FIGURE 14-1: I/O DIAGRAM OF BLOCK



14.5 Signal Description

TABLE 14-1: SIGNAL DESCRIPTION

Name	Direction	Description
ICT0	INPUT	External capture trigger signal for Capture Register 0. Identical to signal FAN_TACH0.
ICT1	INPUT	External capture trigger signal for Capture Register 1. Identical to signal FAN_TACH1.

TABLE 14-1: SIGNAL DESCRIPTION (CONTINUED)

Name	Direction	Description
ICT2	INPUT	External capture trigger signal for Capture Register 2. Identical to signal FAN_TACH2.
ICT3	INPUT	External capture trigger signal for Capture Register 3
ICT4	INPUT	External capture trigger signal for Capture Register 4
ICT5	INPUT	External capture trigger signal for Capture Register 5
CTOUT0	OUTPUT	External compare match signal for Compare Register 0
CTOUT1	OUTPUT	External compare match signal for Compare Register 1

14.6 Host Interface

The registers defined for 16-bit Timers are accessible by the various hosts as indicated in Section 14.12, "EC Registers".

14.7 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

14.7.1 POWER DOMAINS

Name	Description
VTR	The logic and registers implemented in this block are powered by this power well.

14.7.2 CLOCK INPUTS

Name	Description
48MHz	This is the clock source for this block.

14.7.3 RESETS

Name	Description
RESET_SYS	This signal resets all the registers and logic in this block to their default state.

14.8 Interrupts

This section defines the Interrupt Sources generated from this block.

Source	Description
CAPTURE TIMER	This interrupt event fires when the 32-bit free running counter overflows from FFFF_FFFFh to 0000_0000h.
CAPTURE 0	This interrupt event fires when Capture Register 0 acquires a new value.
CAPTURE 1	This interrupt event fires when Capture Register 1 acquires a new value.
CAPTURE 2	This interrupt event fires when Capture Register 2 acquires a new value.
CAPTURE 3	This interrupt event fires when Capture Register 3 acquires a new value.
CAPTURE 4	This interrupt event fires when Capture Register 4 acquires a new value.
CAPTURE 5	This interrupt event fires when Capture Register 5 acquires a new value.
COMPARE 0	This interrupt event fires when the contents of Compare 0 Register match the contents of the Free Running Counter.
COMPARE 1	This interrupt event fires when the contents of Compare 1 Register match the contents of the Free Running Counter.

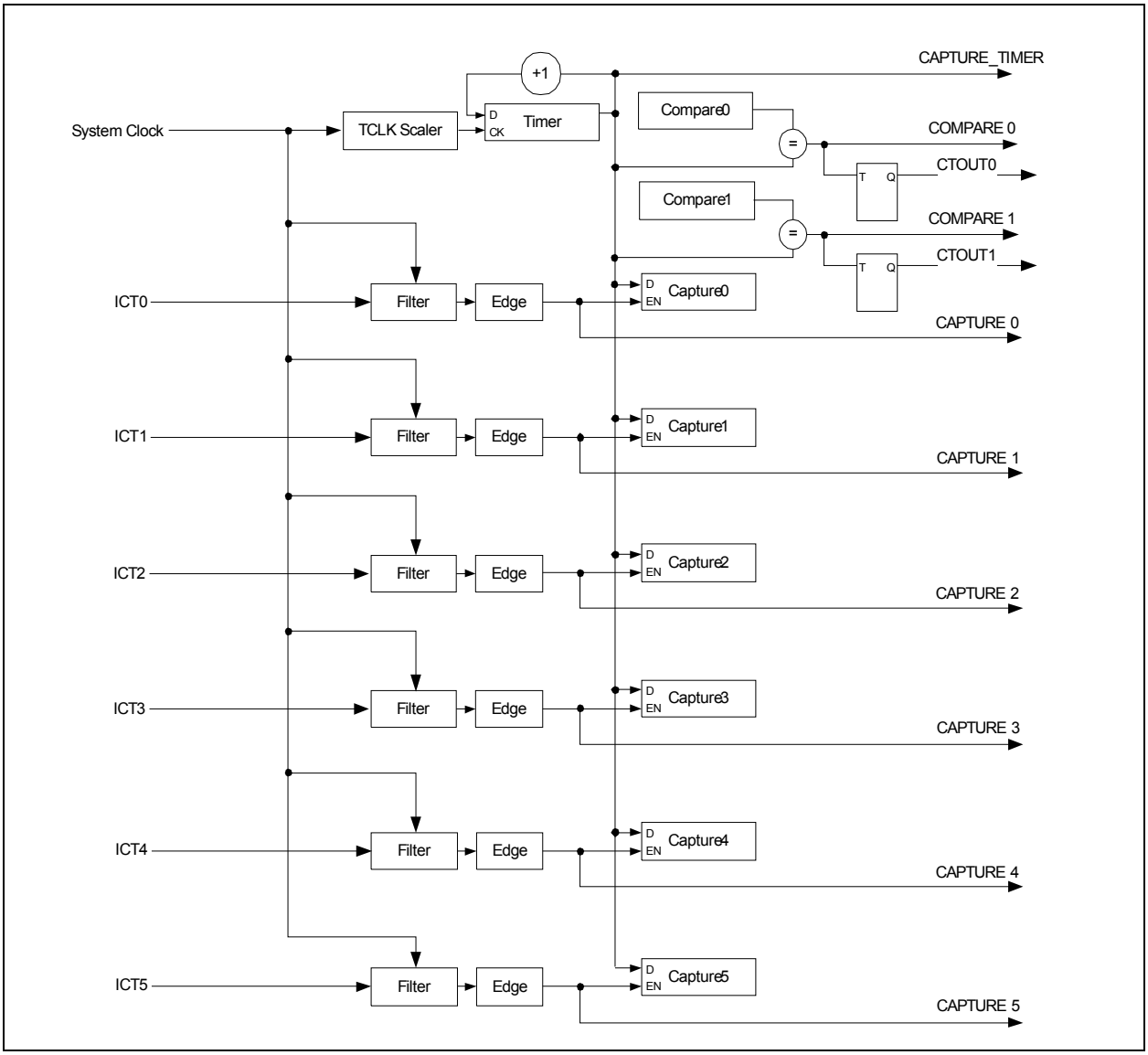
14.9 Low Power Modes

The Capture and Compare Timer may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry. This block is only be permitted to enter low power modes when the block is not active. The block is inactive if the ACTIVATE bit is de-asserted, and will also become inactive when the block's SLEEP_EN signal is asserted.

When the block returns from sleep, if enabled, the Free Running Timer Register value will continue counting from where it was when the block entered the Sleep state.

14.10 Description

FIGURE 14-2: CAPTURE AND COMPARE TIMER BLOCK DIAGRAM



14.10.1 TIMER CLOCK

Any of the frequencies listed in Table 14-2 may be used as the time base for the Free Running Counter.

TABLE 14-2: TIMER CLOCK FREQUENCIES

Timer Clock Select	Frequency Divide Select	Frequency Selected
0000b	Divide by 1	48MHz
0001b	Divide by 2	24MHz
0010b	Divide by 4	12MHz
0011b	Divide by 8	6MHz
0100b	Divide by 16	3MHz
0101b	Divide by 32	1.5MHz

TABLE 14-2: TIMER CLOCK FREQUENCIES (CONTINUED)

Timer Clock Select	Frequency Divide Select	Frequency Selected
0110b	Divide by 64	750KHz
0111b	Divide by 128	375KHz
1xxxb	Reserved	Reserved

For the Timer Clock, the **Timer Clock Select** value is defined by the TCLK field in the Capture and Compare Timer Control Register

14.10.2 FILTER CLOCK AND NOISE FILTER

The noise filter uses the Filter Clock (FCLK) to filter the signal on the Input Capture pins. An Input Capture pin must remain in the same state for three FCLK ticks before the internal state changes. The FILTER_BYPASS bit for the Input Capture pin may be used to bypass the input filter. Each Capture Register can individually bypass the filter.

When the input filter is bypassed, the minimum period of FCLK must be at least 2X the duration of an input signal pulse in order for an edge event to be captured reliably. When the input filter is enabled, the minimum period of FCLK must be at least 4X the duration of an input signal pulse in order for an edge event to be captured reliably.

14.11 Operation

14.11.1 INPUT CAPTURE

The Input Capture block consists of a free-running 32-bit timer and 2 capture registers. Each of the capture registers is associated with an input pin as well as an interrupt source bit in the Interrupt Aggregator: The Capture registers store the current value of the Free Running timer whenever the associated input signal changes, according to the programmed edge detection. An interrupt is also generated to the EC. The Capture registers are read-only. The registers are updated every time an edge is detected. If software does not read the register before the next edge, the value is lost.

14.11.2 COMPARE TIMER

There are two 32-bit Compare registers. Each of these registers can independently generate an interrupt to the EC when the 32-bit Free Running Timer matches the contents of the Compare register. The compare operation for each is enabled or disabled by a bit in the Capture and Compare Timer Control Register.

14.11.2.1 Interrupt Generation

Whenever a Compare Timer is enabled and the Compare register matches the Free Running Timer, a COMPARE event is sent to the Interrupt Aggregator. The event will trigger an EC interrupt if enabled by the appropriate Interrupt Enable register in the Aggregator.

14.11.2.2 Compare Output Generation

Each Compare Timer is associated with a toggle flip-flop. When the 32-bit Free Running Timer matches the contents of the Compare register the output off the flip-flop is complemented. Each of the toggle flip-flops can be independently set or cleared by using the COMPARE_SET or COMPARE_CLEAR fields, respectively, in the Capture and Compare Timer Control Register.

A Compare Timer should be disabled before setting or clearing the output, when updating the Compare register, or when updating the Free Running Timer, so spurious events are not generated by the matcher.

14.12 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the Input Capture and Compare Timer Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

Note:	All registers in this block must be accessed as DWORDs.

TABLE 14-3: REGISTER SUMMARY

Offset	Register Name
00h	Capture and Compare Timer Control Register
04h	Capture Control 0 Register
08h	Capture Control 1 Register
0Ch	Free Running Timer Register
10h	Capture 0 Register
14h	Capture 1 Register
18h	Capture 2 Register
1Ch	Capture 3 Register
20h	Capture 4 Register
24h	Capture 5 Register
28h	Compare 0 Register
2Ch	Compare 1 Register

14.12.1 CAPTURE AND COMPARE TIMER CONTROL REGISTER

Note:	It is not recommended to use Read-Modify-Write operations on this register. May inadvertently cause the COMPARE_SET and COMPARE_CLEAR bits to be written to ‘1’ in error.

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:26	Reserved	R	-	-
25	COMPARE_CLEAR0 When read, returns the current value off the Compare Timer Output 0 state. If written with a ‘1b’, the output state is cleared to ‘0’. Writes have no effect if COMPARE_SET1 in this register is written with a ‘1b’ at the same time. Writes of ‘0b’ have no effect.	R/WC	0	RESET_SYS
24	COMPARE_CLEAR1 When read, returns the current value off the Compare Timer Output 1 state. If written with a ‘1b’, the output state is cleared to ‘0’. Writes have no effect if COMPARE_SET0 in this register is written with a ‘1b’ at the same time. Writes of ‘0b’ have no effect.	R/WC	0	RESET_SYS
23:18	Reserved	R	-	-

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
17	COMPARE_SET0 When read, returns the current value off the Compare Timer Output 0 state. • If written with a '1b', the output state is set to '1'. • Writes of '0b' have no effect	R/WS	0	RESET_SYS
16	COMPARE_SET1 When read, returns the current value off the Compare Timer Output 1 state. If written with a '1b', the output state is set to '1'. Writes of '0b' have no effect	R/WS	0	RESET_SYS
15:10	Reserved	R	-	-
9	COMPARE_ENABLE1 Compare Enable for Compare 1 Register. When enabled, a match between the Compare 1 Register and the Free Running Timer Register will cause the TOUT1 output to toggle and will send a COMPARE event to the Interrupt Aggregator. 1=Enabled 0=Disabled	R/W	0b	RESET_SYS
8	COMPARE_ENABLE0 Compare Enable for Compare 0 Register. When enabled, a match between the Compare 0 Register and the Free Running Timer Register will cause the TOUT0 output to toggle and will send a COMPARE event to the Interrupt Aggregator. 1=Enabled 0=Disabled	R/W	0b	RESET_SYS
7	Reserved	R	-	-
6:4	TCLK This 3-bit field sets the clock source for the Free-Running Counter. See Table 14-2, "Timer Clock Frequencies" for a list of available frequencies.	R/W	0b	RESET_SYS
3	Reserved	R	-	-
2	FREE_RESET Free Running Timer Reset. This bit stops the timer and resets the internal counter to 0000_0000h. This bit does not affect the FREE_ENABLE bit. This bit is self clearing after the timer is reset. 1=Timer reset 0=Normal timer operation	R/W	0h	RESET_SYS

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
1	FREE_ENABLE Free-Running Timer Enable. This bit is used to start and stop the free running timer. This bit does not reset the timer count. The timer starts counting at 0000_0000h on reset and wraps around back to 0000_0000h after it reaches FFFF_FFFFh. The FREE_ENABLE bit is cleared after the RESET cycle is done. Firmware must poll the FREE_RESET bit to determine when it is safe to re-enable the timer. 1=Timer is enabled. The Free Running Timer Register is read-only. 0=Timer is disabled. The Free Running Timer Register is writable.	R/W	0h	RESET_SYS
0	ACTIVATE 1=The timer block is in a running state 0=The timer block is powered down and all clocks are gated	R/W	0h	RESET_SYS

14.12.2 CAPTURE CONTROL 0 REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:29	FCLK_SEL3 This 3-bit field sets the clock source for the input filter for Capture Register 3. See Table 14-2, "Timer Clock Frequencies" for a list of available frequencies.	R/W	0h	RESET_SYS
28:27	Reserved	R	-	-
26	FILTER_BYP3 This bit enables bypassing the input noise filter for Capture Register 3, so that the input signal goes directly into the timer. 1=Input filter bypassed 0=Input filter enabled	R/W	0h	RESET_SYS
25:24	CAPTURE_EDGE3 This field selects the edge type that triggers the capture of the Free Running Counter into Capture Register 3. 3=Capture event disabled 2=Both rising and falling edges 1=Rising edges 0=Falling edges	R/W	0h	RESET_SYS
23:21	FCLK_SEL2 This 3-bit field sets the clock source for the input filter for Capture Register 2. See Table 14-2, "Timer Clock Frequencies" for a list of available frequencies.	R/W	0h	RESET_SYS

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
20:19	Reserved	R	-	-
18	FILTER_BYP2 This bit enables bypassing the input noise filter for Capture Register 2, so that the input signal goes directly into the timer. 1=Input filter bypassed 0=Input filter enabled	R/W	0h	RESET_SYS
17:16	CAPTURE_EDGE2 This field selects the edge type that triggers the capture of the Free Running Counter into Capture Register 2. 3=Capture event disabled 2=Both rising and falling edges 1=Rising edges 0=Falling edges	R/W	0h	RESET_SYS
15:13	FCLK_SEL1 This 3-bit field sets the clock source for the input filter for Capture Register 1. See Table 14-2, "Timer Clock Frequencies" for a list of available frequencies.	R/W	0b	RESET_SYS
12:11	Reserved	R	-	-
10	FILTER_BYP1 This bit enables bypassing the input noise filter for Capture Register 1, so that the input signal goes directly into the timer. 1=Input filter bypassed 0=Input filter enabled	R/W	0h	RESET_SYS
9:8	CAPTURE_EDGE1 This field selects the edge type that triggers the capture of the Free Running Counter into Capture Register 1. 3=Capture event disabled 2=Both rising and falling edges 1=Rising edges 0=Falling edges	R/W	0h	RESET_SYS
7:5	FCLK_SEL0 This 3-bit field sets the clock source for the input filter for Capture Register 0. See Table 14-2, "Timer Clock Frequencies" for a list of available frequencies.	R/W	0h	RESET_SYS
4:3	Reserved	R	-	-

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
2	FILTER_BYP0 This bit enables bypassing the input noise filter for Capture Register 0, so that the input signal goes directly into the timer. 1=Input filter bypassed 0=Input filter enabled	R/W	0h	RESET_SYS
1:0	CAPTURE_EDGE0 This field selects the edge type that triggers the capture of the Free Running Counter into Capture Register 0. 3=Capture event disabled 2=Both rising and falling edges 1=Rising edges 0=Falling edges	R/W	0h	RESET_SYS

14.12.3 CAPTURE CONTROL 1 REGISTER

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	Reserved	R	-	-
15:13	FCLK_SEL5 This 3-bit field sets the clock source for the input filter for Capture Register 5. See Table 14-2, "Timer Clock Frequencies" for a list of available frequencies.	R/W	0b	RESET_SYS
12:11	Reserved	R	-	-
10	FILTER_BYP5 This bit enables bypassing the input noise filter for Capture Register 5, so that the input signal goes directly into the timer. 1=Input filter bypassed 0=Input filter enabled	R/W	0h	RESET_SYS
9:8	CAPTURE_EDGE5 This field selects the edge type that triggers the capture of the Free Running Counter into Capture Register 5. 3=Capture event disabled 2=Both rising and falling edges 1=Rising edges 0=Falling edges	R/W	0h	RESET_SYS
7:5	FCLK_SEL4 This 3-bit field sets the clock source for the input filter for Capture Register 4. See Table 14-2, "Timer Clock Frequencies" for a list of available frequencies.	R/W	0h	RESET_SYS

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
4:3	Reserved	R	-	-
2	FILTER_BYP4 This bit enables bypassing the input noise filter for Capture Register 4, so that the input signal goes directly into the timer. 1=Input filter bypassed 0=Input filter enabled	R/W	0h	RESET_SYS
1:0	CAPTURE_EDGE4 This field selects the edge type that triggers the capture of the Free Running Counter into Capture Register 4. 3=Capture event disabled 2=Both rising and falling edges 1=Rising edges 0=Falling edges	R/W	0h	RESET_SYS

14.12.4 FREE RUNNING TIMER REGISTER

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	FREE_RUNNING_TIMER This register contains the current value of the Free Running Timer. A Capture Timer interrupt is signaled to the Interrupt Aggregator when this register transitions from FFFF_FFFFh to 0000_0000h. When FREE_ENABLE in the Capture and Compare Timer Control Register is '1', this register is read-only. When FREE_ENABLE is '0', this register may be written.	R/W	0h	RESET_SYS

14.12.5 CAPTURE 0 REGISTER

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	CAPTURE_0 This register saves the value copied from the Free Running timer on a programmed edge of ICT0.	R	0h	RESET_SYS

14.12.6 CAPTURE 1 REGISTER

Offset	14h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	CAPTURE_1 This register saves the value copied from the Free Running timer on a programmed edge of ICT1.	R	0h	RESET_SYS

14.12.7 CAPTURE 2 REGISTER

Offset	18h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	CAPTURE_2 This register saves the value copied from the Free Running timer on a programmed edge of ICT2.	R	0h	RESET_SYS

14.12.8 CAPTURE 3 REGISTER

Offset	1Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	CAPTURE_3 This register saves the value copied from the Free Running timer on a programmed edge of ICT3.	R	0h	RESET_SYS

14.12.9 CAPTURE 4 REGISTER

Offset	20h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	CAPTURE_4 This register saves the value copied from the Free Running timer on a programmed edge of ICT4.	R	0h	RESET_SYS

14.12.10 CAPTURE 5 REGISTER

Offset	24h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	CAPTURE_5 This register saves the value copied from the Free Running timer on a programmed edge of ICT5.	R	0h	RESET_SYS

14.12.11 COMPARE 0 REGISTER

Offset	28h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	COMPARE_0 A COMPARE 0 interrupt is generated when this register matches the value in the Free Running Timer.	R/W	0h	RESET_SYS

14.12.12 COMPARE 1 REGISTER

Offset	2Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	COMPARE_1 A COMPARE 1 interrupt is generated when this register matches the value in the Free Running Timer.	R/W	0h	RESET_SYS

15.0 HIBERNATION TIMER

15.1 Introduction

The Hibernation Timer can generate a wake event to the Embedded Controller (EC) when it is in a hibernation mode. This block supports wake events up to 2 hours in duration. The timer is a 16-bit binary count-down timer that can be programmed in 30.5µs and 0.125 second increments for period ranges of 30.5µs to 2s or 0.125s to 136.5 minutes, respectively. Writing a non-zero value to this register starts the counter from that value. A wake-up interrupt is generated when the count reaches zero.

15.2 References

No references have been cited for this chapter

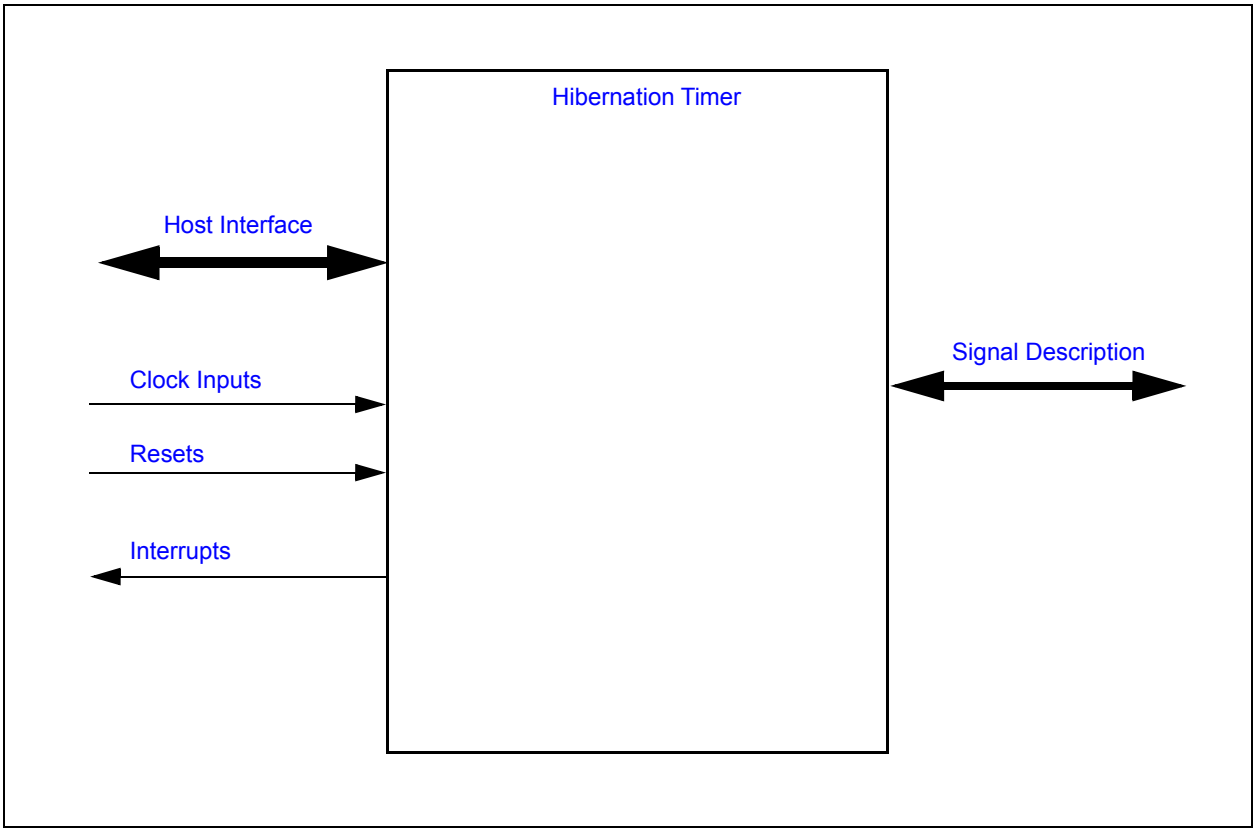
15.3 Terminology

No terms have been cited for this chapter.

15.4 Interface

This block is an IP block designed to be incorporated into a chip. It is designed to be accessed externally via the pin interface and internally via a registered host interface. The following diagram illustrates the various interfaces to the block.

FIGURE 15-1: HIBERNATION TIMER INTERFACE DIAGRAM



15.5 Signal Description

There are no external signals for this block.

15.6 Host Interface

The registers defined for the Hibernation Timer are accessible by the various hosts as indicated in Section 15.10, "EC Registers".

15.7 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

15.7.1 POWER DOMAINS

TABLE 15-1: POWER SOURCES

Name	Description
VTR	The timer control logic and registers are all implemented on this single power domain.

15.7.2 CLOCK INPUTS

TABLE 15-2: CLOCK INPUTS

Name	Description
32KHz	This is the clock source to the timer logic. The Pre-scaler may be used to adjust the minimum resolution per bit of the counter. if the main oscillator is stopped then an external 32.768kHz clock source must be active for the Hibernation Timer to continue to operate.

15.7.3 RESETS

TABLE 15-3: RESET SIGNALS

Name	Description
RESET_SYS	This reset signal, which is an input to this block, resets all the logic and registers to their initial default state.

15.8 Interrupts

This section defines the interrupt Interface signals routed to the chip interrupt aggregator.

Each instance of the Hibernation Timer in the CEC1702 can be used to generate interrupts and wake-up events when the timer decrements to zero.

TABLE 15-4: INTERRUPT INTERFACE SIGNAL DESCRIPTION TABLE

Name	Direction	Description
HTIMER	Output	Signal indicating that the timer is enabled and decrements to 0. This signal is used to generate an Hibernation Timer interrupt event.

15.9 Low Power Modes

The timer operates off of the 32KHz clock, and therefore will operate normally when the main oscillator is stopped.

The sleep enable inputs have no effect on the Hibernation Timer and the clock required outputs are only asserted during register read/write cycles for as long as necessary to propagate updates to the block core.

15.10 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the Hibernation Timer Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 15-5: REGISTER SUMMARY

Offset	Register Name
00h	HTimer Preload Register
04h	HTimer Control Register
08h	HTimer Count Register

15.10.1 HTIMER PRELOAD REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
15:0	HT_PRELOAD This register is used to set the Hibernation Timer Preload value. Writing this register to a non-zero value resets the down counter to start counting down from this programmed value. Writing this register to 0000h disables the hibernation counter. The resolution of this timer is determined by the CTRL bit in the HTimer Control Register. Writes to the HTimer Control Register are completed with an EC bus cycle.	R/W	000h	RESET_SYS

15.10.2 HTIMER CONTROL REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
15:1	Reserved	R	-	-
0	CTRL 1=The Hibernation Timer has a resolution of 0.125s per LSB, which yields a maximum time in excess of 2 hours. 0=The Hibernation Timer has a resolution of 30.5µs per LSB, which yields a maximum time of ~2seconds.	R	0000h	RESET_SYS

15.10.3 HTIMER COUNT REGISTER

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
15:0	COUNT The current state of the Hibernation Timer.	R	0000h	RESET_SYS

16.0 RTOS TIMER

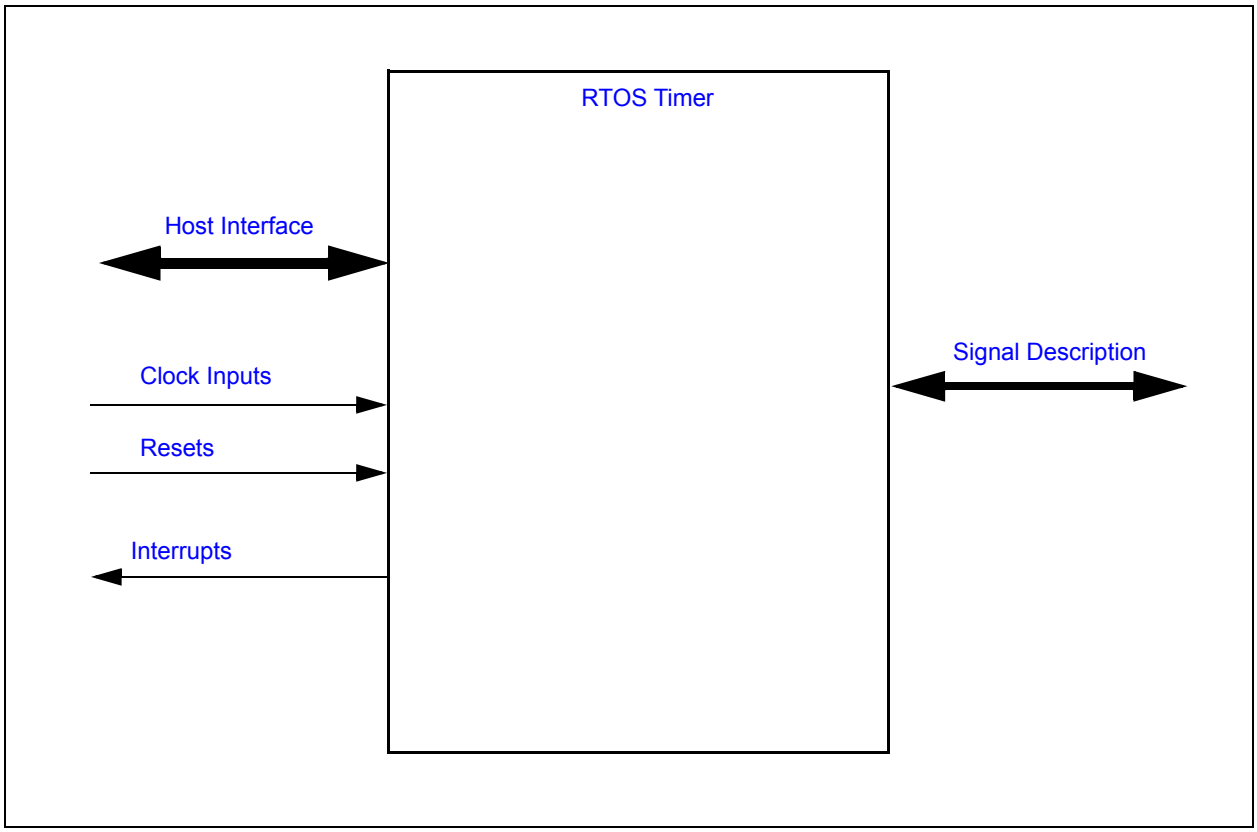
16.1 Introduction

The RTOS Timer is a low-power, 32-bit timer designed to operate on the 32kHz oscillator which is available during all chip sleep states. This allows firmware the option to sleep the processor and wake after a programmed amount of time. The timer may be used as a one-shot timer or a continuous timer. When the timer transitions to 0 it is capable of generating a wake-capable interrupt to the embedded controller. This timer may be halted during debug by hardware or via a software control bit.

16.2 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

FIGURE 16-1: I/O DIAGRAM OF BLOCK



16.3 Signal Description

There are no external signals for this block.

Name	Description
HALT	RTOS Timer Halt signal. This signal is connected to the same signal that halts the embedded controller during debug (e.g., JTAG Debugger is active, break points, etc.).

16.4 Host Interface

The embedded controller may access this block via the registers defined in Section 16.9, "EC Registers".

16.5 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

16.5.1 POWER DOMAINS

Name	Description
VTR	The timer control logic and registers are all implemented on this single power domain.

16.5.2 CLOCK INPUTS

Name	Description
32KHz	This is the clock source to the timer logic.

16.5.3 RESETS

Name	Description
RESET_SYS	This reset signal, which is an input to this block, resets all the logic and registers to their initial default state.

16.6 Interrupts

Source	Description
RTOS_TIMER	RTOS Timer interrupt event. The interrupt is signaled when the timer counter transitions from 1 to 0 while counting.

16.7 Low Power Modes

The Basic Timer may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry. This block is only be permitted to enter low power modes when the block is not active.

16.8 Description

The RTOS Timer is a basic down counter that can operate either as a continuous timer or a one-shot timer. When it is started, the counter is loaded with a pre-load value and counts towards 0. When the counter counts down from 1 to 0, it will generate an interrupt. In one-shot mode (the AUTO_RELOAD bit is '0'), the timer will then halt; in continuous mode (the AUTO_RELOAD bit is '1'), the counter will automatically be restarted with the pre-load value.

The timer counter can be halted by firmware by setting the FIRMWARE_TIMER_HALT bit to '1'. In addition, if enabled, the timer counter can be halted by the external HALT signal.

16.9 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the RTOS Timer Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 16-1: REGISTER SUMMARY

Offset	Register Name
00h	RTOS Timer Count Register
04h	RTOS Timer Preload Register
08h	RTOS Timer Control Register
0Ch	Soft Interrupt Register

16.9.1 RTOS TIMER COUNT REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	COUNTER This register contains the current value of the RTOS Timer counter. This register should be read as a DWORD. There is no latching mechanism of the upper bytes implemented if the register is accessed as a byte or word. Reading the register with byte or word operations may give incorrect results.	R/W	0h	RESET_SYS

16.9.2 RTOS TIMER PRELOAD REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	PRE_LOAD The this register is loaded into the RTOS Timer counter either when the TIMER_START bit is written with a ‘1’, or when the timer counter counts down to ‘0’ and the AUTO_RELOAD bit is ‘1’. This register must be programmed with a new count value before the TIMER_START bit is set to ‘1’. If this register is updated while the counter is operating, the new count value will only take effect if the counter transitions form 1 to 0 while the AUTO_RELOAD bit is set.	R/W	0h	RESET_SYS

16.9.3 RTOS TIMER CONTROL REGISTER

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:5	Reserved	R	-	-
4	FIRMWARE_TIMER_HALT 1=The timer counter is halted. If the counter was running, clearing this bit will restart the counter from the value at which it halted 0=The timer counter, if enabled, will continue to run	R/W	0h	RESET_SYS
3	EXT_HARDWARE_HALT_EN 1=The timer counter is halted when the external HALT signal is asserted. Counting is always enabled if HALT is de-asserted. 0=The HALT signal does not affect the RTOS Timer	R/W	0h	RESET_SYS
2	TIMER_START Writing a '1' to this bit will load the timer counter with the RTOS Timer Preload Register and start counting. If the Preload Register is 0, counting will not start and this bit will be cleared to '0'. Writing a '0' to this bit will halt the counter and clear its contents to 0. The RTOS timer interrupt will not be generated. This bit is automatically cleared if the AUTO_RELOAD bit is '0' and the timer counter transitions from 1 to 0.	R/W	0h	RESET_SYS
1	AUTO_RELOAD 1=The the RTOS Timer Preload Register is loaded into the timer counter and the counter is restarted when the counter transitions from 1 to 0 0=The timer counter halts when it transitions from 1 to 0 and will not restart	R/W	0h	RESET_SYS
0	BLOCK_ENABLE 1=RTOS timer counter is enabled 0=RTOS timer disabled. All register bits are reset to their default state	R/W	0h	RESET_SYS

16.9.4 SOFT INTERRUPT REGISTER

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:4	Reserved	R	-	-
3	SWI_3 Software Interrupt. A write of a ‘1’ to this bit will generate an SWI interrupt to the EC. Writes of a ‘0’ have no effect. Reads return ‘0’.	W	0h	RESET_SYS
2	SWI_2 Software Interrupt. A write of a ‘1’ to this bit will generate an SWI interrupt to the EC. Writes of a ‘0’ have no effect. Reads return ‘0’.	W	0h	RESET_SYS
1	SWI_1 Software Interrupt. A write of a ‘1’ to this bit will generate an SWI interrupt to the EC. Writes of a ‘0’ have no effect. Reads return ‘0’.	W	0h	RESET_SYS
0	SWI_0 Software Interrupt. A write of a ‘1’ to this bit will generate an SWI interrupt to the EC. Writes of a ‘0’ have no effect. Reads return ‘0’.	W	0h	RESET_SYS

17.0 REAL TIME CLOCK

17.1 Introduction

This block provides the capabilities of an industry-standard 146818B Real-Time Clock module, without CMOS RAM. Enhancements to this architecture include:

- Industry standard Day of Month Alarm field, allowing for monthly alarms
- Configurable, automatic Daylight Savings adjustment
- Week Alarm for periodic interrupts and wakes based on Day of Week
- System Wake capability on interrupts.

17.2 References

1. Motorola 146818B Data Sheet, available on-line
2. Intel Lynx Point PCH EDS specification

17.3 Terminology

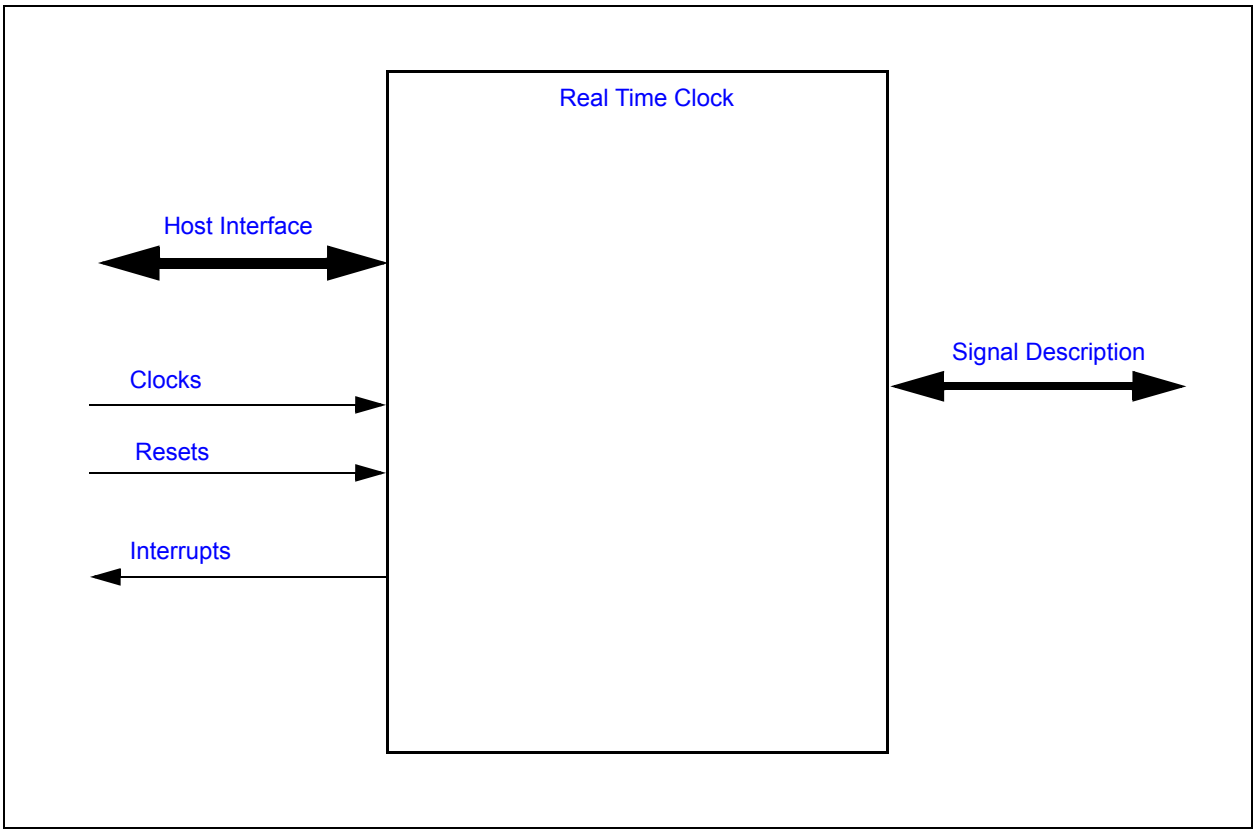
Time and Date Registers:

This is the set of registers that are automatically counted by hardware every 1 second while the block is enabled to run and to update. These registers are: **Seconds**, **Minutes**, **Hours**, **Day of Week**, **Day of Month**, **Month**, and **Year**.

17.4 Interface

This block's connections are entirely internal to the chip.

FIGURE 17-1: I/O DIAGRAM OF BLOCK



17.5 Signal Description

There are no external signals.

17.6 Host Interface

The registers defined for the Real Time Clock are accessible by the host and EC.

17.7 Power, Clocks and Resets

This section defines the Power, Clock, and Reset parameters of the block.

17.7.1 POWER DOMAINS

TABLE 17-1: POWER SOURCES

Name	Description
VBAT	This power well sources all of the internal registers and logic in this block.
VTR	This power well sources only bus communication. The block continues to operate internally while this rail is down.

17.7.2 CLOCKS

TABLE 17-2: CLOCKS

Name	Description
32KHz	This clock input drives all internal logic, and will be present at all times that the VBAT well is powered.

17.7.3 RESETS

TABLE 17-3: RESET SIGNALS

Name	Description
RESET_VBAT	This reset signal is used in the RESET_RTC signal to reset all of the registers and logic in this block. It directly resets the Soft Reset bit in the RTC Control Register.
RESET_RTC	This reset signal resets all of the registers and logic in this block, except for the Soft Reset bit in the RTC Control Register. It is triggered by RESET_VBAT, but can also be triggered by a Soft Reset from the RTC Control Register.
RESET_SYS	This reset signal is used to inhibit the bus communication logic, and isolates this block from VTR powered circuitry on-chip. Otherwise it has no effect on the internal state.

17.8 Interrupts

TABLE 17-4: SYSTEM INTERRUPTS

Source	Description
RTC	This interrupt source for the SIRQ logic is generated when any of the following events occur: • Update complete. This is triggered, at 1-second intervals, when the Time register updates have completed • Alarm. This is triggered when the alarm value matches the current time (and date, if used) • Periodic. This is triggered at the chosen programmable rate

TABLE 17-5: EC INTERRUPTS

Source	Description
RTC	This interrupt is signaled to the Interrupt Aggregator when any of the following events occur: • Update complete. This is triggered, at 1-second intervals, when the Time register updates have completed • Alarm. This is triggered when the alarm value matches the current time (and date, if used) • Periodic. This is triggered at the chosen programmable rate
RTC ALARM	This wake interrupt is signaled to the Interrupt Aggregator when an Alarm event occurs.

17.9 Low Power Modes

The RTC has no low-power modes. It runs continuously while the VBAT well is powered.

17.10 Description

This block provides the capabilities of an industry-standard 146818B Real-Time Clock module, excluding the CMOS RAM and the SQW output. See the following registers, which represent enhancements to this architecture. These enhancements are listed below.

See the Date Alarm field of Register D for a Day of Month qualifier for alarms.

See the Week Alarm Register for a Day of Week qualifier for alarms.

See the registers Daylight Savings Forward Register and Daylight Savings Backward Register for setting up hands-off Daylight Savings adjustments.

See the RTC Control Register for enhanced control over the block's operations.

17.11 Runtime Registers

The registers listed in the Runtime Register Summary table are for a single instance of the Real Time Clock. Host access for each register listed in this table is defined as an offset in the Host address space to the Block's Base Address, as defined by the instance's Base Address Register.

The EC address for each register is formed by adding the Base Address for each instance of the Real Time Clock shown in the Block Overview and Base Address Table in Section 3.0, "Device Inventory" to the offset shown in the "Offset" column.

TABLE 17-6: RUNTIME REGISTER SUMMARY

Offset	Register Name
00h	Seconds Register
01h	Seconds Alarm Register
02h	Minutes Register
03h	Minutes Alarm Register
04h	Hours Register
05h	Hours Alarm Register
06h	Day of Week Register
07h	Day of Month Register
08h	Month Register
09h	Year Register
0Ah	Register A
0Bh	Register B
0Ch	Register C

TABLE 17-6: RUNTIME REGISTER SUMMARY (CONTINUED)

Offset	Register Name
0Dh	Register D
0Eh	Reserved
0Fh	Reserved
10h	RTC Control Register
14h	Week Alarm Register
18h	Daylight Savings Forward Register
1Ch	Daylight Savings Backward Register
20h	TEST

Note: This extended register set occupies offsets that have historically been used as CMOS RAM. Code ported to use this block should be examined to ensure that it does not assume that RAM exists in this block.

17.11.1 SECONDS REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	SECONDS Displays the number of seconds past the current minute, in the range 0--59. Presentation may be selected as binary or BCD, depending on the DM bit in Register B. Values written must also use the format defined by the current setting of the DM bit.	R/W	00h	RESET_RTC

17.11.2 SECONDS ALARM REGISTER

Offset	01h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	SECONDS_ALARM Holds a match value, compared against the Seconds Register to trigger the Alarm event. Values written to this register must use the format defined by the current setting of the DM bit in Register B. A value of 11xxxxxxb written to this register makes it don't-care (always matching).	R/W	00h	RESET_RTC

17.11.3 MINUTES REGISTER

Offset	02h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	MINUTES Displays the number of minutes past the current hour, in the range 0--59. Presentation may be selected as binary or BCD, depending on the DM bit in Register B. Values written must also use the format defined by the current setting of the DM bit.	R/W	00h	RESET_RTC

17.11.4 MINUTES ALARM REGISTER

Offset	03h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	MINUTES_ALARM Holds a match value, compared against the Minutes Register to trigger the Alarm event. Values written to this register must use the format defined by the current setting of the DM bit in Register B. A value of 11xxxxxxb written to this register makes it don't-care (always matching).	R/W	00h	RESET_RTC

17.11.5 HOURS REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7	HOURS_AM_PM In 12-hour mode (see bit "24/12" in register B), this bit indicates AM or PM. 1=PM 0=AM	R/W	0b	RESET_RTC
6:0	HOURS Displays the number of the hour, in the range 1--12 for 12-hour mode (see bit "24/12" in register B), or in the range 0--23 for 24-hour mode. Presentation may be selected as binary or BCD, depending on the DM bit in Register B. Values written must also use the format defined by the current setting of the DM bit.	R/W	00h	RESET_RTC

17.11.6 HOURS ALARM REGISTER

Offset	05h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	HOURS_ALARM Holds a match value, compared against the Hours Register to trigger the Alarm event. Values written to this register must use the format defined by the current settings of the DM bit and the 24/12 bit in Register B. A value of 11xxxxxxb written to this register makes it don't-care (always matching).	R/W	00h	RESET_RTC

17.11.7 DAY OF WEEK REGISTER

Offset	06h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	DAY_OF_WEEK Displays the day of the week, in the range 1 (Sunday) through 7 (Saturday). Numbers in this range are identical in both binary and BCD notation, so this register's format is unaffected by the DM bit.	R/W	00h	RESET_RTC

17.11.8 DAY OF MONTH REGISTER

Offset	07h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	DAY_OF_MONTH Displays the day of the current month, in the range 1--31. Presentation may be selected as binary or BCD, depending on the DM bit in Register B. Values written must also use the format defined by the current setting of the DM bit.	R/W	00h	RESET_RTC

17.11.9 MONTH REGISTER

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	MONTH Displays the month, in the range 1--12. Presentation may be selected as binary or BCD, depending on the DM bit in Register B. Values written must also use the format defined by the current setting of the DM bit.	R/W	00h	RESET_RTC

17.11.10 YEAR REGISTER

Offset	09h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	YEAR Displays the number of the year in the current century, in the range 0 (year 2000) through 99 (year 2099). Presentation may be selected as binary or BCD, depending on the DM bit in Register B. Values written must also use the format defined by the current setting of the DM bit.	R/W	00h	RESET_RTC

17.11.11 REGISTER A

Offset	0Ah			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7	UPDATE_IN_PROGRESS '0' indicates that the Time and Date registers are stable and will not be altered by hardware soon. '1' indicates that a hardware update of the Time and Date registers may be in progress, and those registers should not be accessed by the host program. This bit is set to '1' at a point 488us (16 cycles of the 32K clock) before the update occurs, and is cleared immediately after the update. See also the Update-Ended Interrupt, which provides more useful status.	R	0b	RESET_RTC
6:4	DIVISION_CHAIN_SELECT This field provides general control for the Time and Date register updating logic. 11xb=Halt counting. The next time that 010b is written, updates will begin 500ms later. 010b=Required setting for normal operation. It is also necessary to set the Block Enable bit in the RTC Control Register to '1' for counting to begin 000b=Reserved. This field should be initialized to another value before Enabling the block in the RTC Control Register Other values Reserved	R/W	000b	RESET_RTC
3:0	RATE_SELECT This field selects the rate of the Periodic Interrupt source. See Table 17-7	R/W	0h	RESET_RTC

TABLE 17-7: REGISTER A FIELD RS: PERIODIC INTERRUPT SETTINGS

RS (hex)	Interrupt Period
0	Never Triggered
1	3.90625 ms
2	7.8125 ms
3	122.070 us
4	244.141 us
5	488.281 us
6	976.5625 us
7	1.953125 ms
8	3.90625 ms
9	7.8125 ms
A	15.625 ms
B	31.25 ms
C	62.5 ms
D	125 ms
E	250 ms
F	500 ms

17.11.12 REGISTER B

Offset	0Bh			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7	UPDATE_CYCLE_INHIBIT In its default state '0', this bit allows hardware updates to the Time and Date registers, which occur at 1-second intervals. A '1' written to this field inhibits updates, allowing these registers to be cleanly written to different values. Writing '0' to this bit allows updates to continue.	R/W	0b	RESET_RTC
6	PERIODIC_INTERRUPT_ENABLE 1=Alows the Periodic Interrupt events to be propagated as interrupts 0=Periodic events are not propagates as interrupts	R/W	0b	RESET_RTC
5	ALARM_INTERRUPT_ENABLE 1=Alows the Alarm Interrupt events to be propagated as interrupts 0=Alarm events are not propagates as interrupts	R/W	0b	RESET_RTC
4	UPDATE_ENDED_INTERRUPT_ENABLE 1=Alows the Update Ended Interrupt events to be propagated as interrupts 0=Update Ended events are not propagates as interrupts	R/W	0b	RESET_RTC
3	Reserved	R	-	-
2	DATA_MODE 1=Binary Mode for Dates and Times 0=BCD Mode for Dates and Times	R/W	0b	RESET_RTC
1	HOUR_FORMAT_24_12 1=24-Hour Format for Hours and Hours Alarm registers. 24-Hour format keeps the AM/PM bit off, with value range 0--23 0=12-Hour Format for Hours and Hours Alarm registers. 12-Hour format has an AM/PM bit, and value range 1--12	R/W	0b	RESET_RTC
0	DAYLIGHT_SAVINGS_ENABLE 1=Enables automatic hardware updating of the hour, using the registers Daylight Savings Forward and Daylight Savings Backward to select the yearly date and hour for each update 0=Automatic Daylight Savings updates disabled	R/W	0b	RESET_RTC

Note: The DATA_MODE and HOUR_FORMAT_24_12 bits affect only how values are presented as they are being read and how they are interpreted as they are being written. They do not affect the internal contents or interpretations of registers that have already been written, nor do they affect how those registers are represented or counted internally. This mode bits may be set and cleared dynamically, for whatever I/O data representation is desired by the host program.

17.11.13 REGISTER C

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7	INTERRUPT_REQUEST_FLAG 1=Any of bits[6:4] below is active after masking by their respective Enable bits in Register B. 0=No bits in this register are active This bit is automatically cleared by every Read access to this register.	RC	0b	RESET_RTC
6	PERIODIC_INTERRUPT_FLAG 1=A Periodic Interrupt event has occurred since the last time this register was read. This bit displays status regardless of the Periodic Interrupt Enable bit in Register B 0=A Periodic Interrupt event has not occurred This bit is automatically cleared by every Read access to this register.	RC	0b	RESET_RTC
5	ALARM_FLAG 1=An Alarm event has occurred since the last time this register was read. This bit displays status regardless of the Alarm Interrupt Enable bit in Register B. 0=An Alarm event has not occurred This bit is automatically cleared by every Read access to this register.	RC	0b	RESET_RTC
4	UPDATE_ENDED_INTERRUPT_FLAG 1=A Time and Date update has completed since the last time this register was read. This bit displays status regardless of the Update-Ended Interrupt Enable bit in Register B. Presentation of this status indicates that the Time and Date registers will be valid and stable for over 999ms 0=A Time and Data update has not completed since the last time this register was read This bit is automatically cleared by every Read access to this register.	RC	0b	RESET_RTC
3:0	Reserved	R	-	-

17.11.14 REGISTER D

Offset	0Dh			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:6	Reserved	R	-	-
5:0	DATE_ALARM This field, if set to a non-zero value, will inhibit the Alarm interrupt unless this field matches the contents of the Month register also. If this field contains 00h (default), it represents a don't-care, allowing more frequent alarms.	R/W	00h	RESET_RTC

17.11.15 RTC CONTROL REGISTER

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:4	Reserved	R	-	-
3	ALARM_ENABLE 1=Enables the Alarm features 0=Disables the Alarm features	R/W	0b	RESET_RTC
2	VCI_ENABLE 1=Allows Alarm events to activate the RTC ALARM signal to chip level VCI circuitry and to the RTC ALARM Interrupt in Interrupt aggregator. 0=Inihibits Alarm events from activating the RTC_ALARM signal to the Chip level VCI circuitry and to the RTC alarm interrupt in Interrupt aggregator.	R/W	0b	RESET_RTC
1	SOFT_RESET A '1' written to this bit position will trigger the RESET_RTC reset, resetting the block and all registers except this one and the Test Register. This bit is self-clearing at the end of the reset and so requires no waiting.	R/W	0b	RESET_VBAT
0	BLOCK_ENABLE This bit must be '1' in order for the block to function internally. Registers may be initialized first, before setting this bit to '1' to start operation.	R/W	0b	RESET_RTC

17.11.16 WEEK ALARM REGISTER

Offset	14h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	ALARM_DAY_OF_WEEK This register, if written to a value in the range 1--7, will inhibit the Alarm interrupt unless this field matches the contents of the Day of Week Register also. If this field is written to any value 11xxxxxxb (like the default FFh), it represents a don't-care, allowing more frequent alarms, and will read back as FFh until another value is written.	R/W	FFh	RESET_RTC

17.11.17 DAYLIGHT SAVINGS FORWARD REGISTER

Offset	18h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31	DST_FORWARD_AM_PM This bit selects AM vs. PM, to match bit[7] of the Hours Register if 12-Hour mode is selected in Register B at the time of writing.	R/W	0b	RESET_RTC
30:24	DST_FORWARD_HOUR This field holds the matching value for bits[6:0] of the Hours register. The written value will be interpreted according to the 24/12 Hour mode and DM mode settings at the time of writing.	R/W	00h	RESET_RTC

Offset	18h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
23:19	Reserved	R	-	-
18:16	DST_FORWARD_WEEK This value matches an internally-maintained week number within the current month. Valid values for this field are: 5=Last week of month 4 =Fourth week of month 3=Third week of month 2=Second week of month 1=First week of month	R/W	0h	RESET_RTC
15:11	Reserved	R	-	-
10:8	DST_FORWARD_DAY_OF_WEEK This field matches the Day of Week Register bits[2:0].	R/W	0h	RESET_RTC
7:0	DST_FORWARD_MONTH This field matches the Month Register.	R/W	00h	RESET_RTC

This is a 32-bit register, accessible also as individual bytes. When writing as individual bytes, ensure that the DSE bit (in Register B) is off first, or that the block is disabled or stopped (SET bit), to prevent a time update while this register may have incompletely-updated contents.

When enabled by the DSE bit in Register B, this register defines an hour and day of the year at which the Hours register will be automatically incremented by 1 additional hour.

There are no don’t-care fields recognized. All fields must be already initialized to valid settings whenever the DSE bit is ‘1’.

Fields other than Week and Day of Week use the current setting of the DM bit (binary vs. BCD) to interpret the information as it is written to them. Their values, as held internally, are not changed by later changes to the DM bit, without subsequently writing to this register as well.

Note:	An Alarm that is set inside the hour after the time specified in this register will not be triggered, because that one-hour period is skipped. This period includes the exact time (0 minutes: 0 seconds) given by this register, through the 59 minutes: 59 seconds point afterward.

17.11.18 DAYLIGHT SAVINGS BACKWARD REGISTER

Offset	1Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31	DST_BACKWARD_AM_PM This bit selects AM vs. PM, to match bit[7] of the Hours register if 12-Hour mode is selected in Register B at the time of writing.	R/W	0b	RESET_RTC
30:24	DST_BACKWARD_HOUR This field holds the matching value for bits[6:0] of the Hours register. The written value will be interpreted according to the 24/12 Hour mode and DM mode settings at the time of writing.	R/W	00h	RESET_RTC
23:19	Reserved	R	-	-

Offset	1Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
18:16	DST_BACKWARD_WEEK This value matches an internally-maintained week number within the current month. Valid values for this field are: 5=Last week of month 4 =Fourth week of month 3=Third week of month 2=Second week of month 1=First week of month	R/W	0h	RESET_RTC
15:11	Reserved	R	-	-
10:8	DST_BACKWARD_DAY_OF_WEEK This field matches the Day of Week Register bits[2:0].	R/W	0h	RESET_RTC
7:0	DST_BACKWARD_MONTH This field matches the Month Register.	R/W	00h	RESET_RTC

This is a 32-bit register, accessible also as individual bytes. When writing as individual bytes, ensure that the DSE bit (in Register B) is off first, or that the block is disabled or stopped (SET bit), to prevent a time update while this register may have incompletely-updated contents.

When enabled by the DSE bit in Register B, this register defines an hour and day of the year at which the Hours register increment will be inhibited from occurring. After triggering, this feature is automatically disabled for long enough to ensure that it will not retrigger the second time this Hours value appears, and then this feature is re-enabled automatically.

There are no don’t-care fields recognized. All fields must be already initialized to valid settings whenever the DSE bit is ‘1’.

Fields other than Week and Day of Week use the current setting of the DM bit (binary vs. BCD) to interpret the information as it is written to them. Their values, as held internally, are not changed by later changes to the DM bit, without subsequently writing to this register as well.

Note: An Alarm that is set inside the hour before the time specified in this register will be triggered twice, because that one-hour period is repeated. This period will include the exact time (0 minutes: 0 seconds) given by this register, through the 59 minutes: 59 seconds point afterward.

18.0 WEEK TIMER

18.1 Introduction

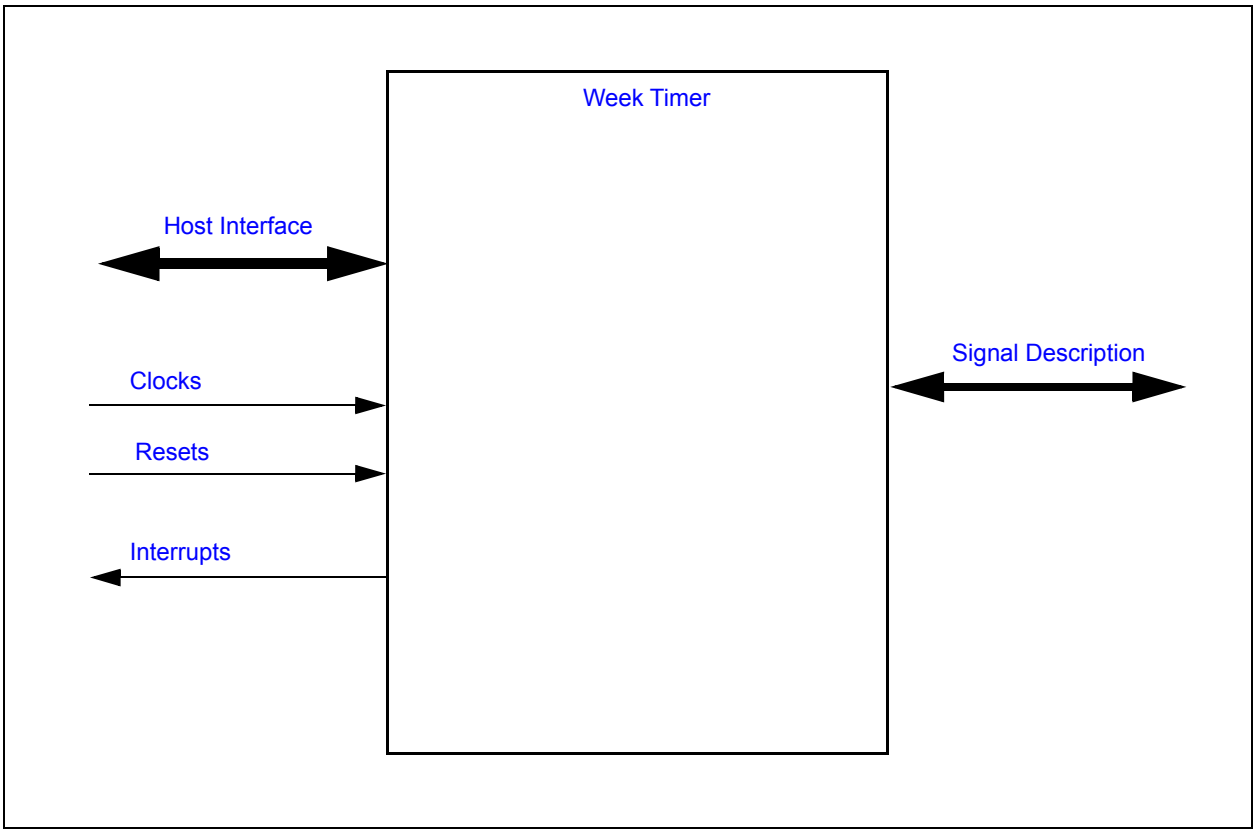
The Week Alarm Interface provides two timekeeping functions: a Week Timer and a Sub-Week Timer. Both the Week Timer and the Sub-Week Timer assert the Power-Up Event Output which automatically powers-up the system from the G3 state. Features include:

- EC interrupts based on matching a counter value
- Repeating interrupts at 1 second and sub-1 second intervals
- System Wake capability on interrupts, including Wake from Heavy Sleep

18.2 Interface

This block's connections are entirely internal to the chip.

FIGURE 18-1: I/O DIAGRAM OF BLOCK



18.3 Signal Description

TABLE 18-1: SIGNAL DESCRIPTION TABLE

Name	Direction	Description
BGPO[9:0]	OUTPUT	Battery-powered general purpose outputs

TABLE 18-2: INTERNAL SIGNAL DESCRIPTION TABLE

Name	Direction	Description
POWER_UP_EVENT	OUTPUT	Signal to the VBAT-Powered Control Interface. When this signal is asserted, the VCI output signal asserts. See Section 18.8, "Power-Up Events".

18.4 Host Interface

The registers defined for the Week Timer are accessible only by the EC.

18.5 Power, Clocks and Resets

This section defines the Power, Clock, and Reset parameters of the block.

18.5.1 POWER DOMAINS

TABLE 18-3: POWER SOURCES

Name	Description
VBAT	This power well sources all of the internal registers and logic in this block.
VTR	This power well sources only bus communication. The block continues to operate internally while this rail is down.

18.5.2 CLOCKS

TABLE 18-4: CLOCKS

Name	Description
48MHz	Clock used for host register access
32KHz	This 32KHz clock input drives all internal logic, and will be present at all times that the VBAT well is powered.

18.5.3 RESETS

TABLE 18-5: RESET SIGNALS

Name	Description
RESET_VBAT	This reset signal is used reset all of the registers and logic in this block.
RESET_SYS	This reset signal is used to inhibit the bus communication logic, and isolates this block from VTR powered circuitry on-chip. Otherwise it has no effect on the internal state.

18.6 Interrupts

TABLE 18-6: EC INTERRUPTS

Source	Description
WEEK_ALARM_INT	This interrupt is signaled to the Interrupt Aggregator when the Week Alarm Counter Register is greater than or equal to the Week Timer Compare Register. The interrupt signal is always generated by the Week Timer if the block is enabled; the interrupt is enabled or disabled in the Interrupt Aggregator.
SUB_WEEK_ALARM_INT	This interrupt is signaled to the Interrupt Aggregator when the Sub-Week Alarm Counter Register decrements from '1' to '0'. The interrupt signal is always generated by the Week Timer if the block is enabled; the interrupt is enabled or disabled in the Interrupt Aggregator.
ONE_SECOND	This interrupt is signaled to the Interrupt Aggregator at an isochronous rate of once per second. The interrupt signal is always generated by the Week Timer if the block is enabled; the interrupt is enabled or disabled in the Interrupt Aggregator.
SUB_SECOND	This interrupt is signaled to the Interrupt Aggregator at an isochronous rate programmable between 0.5Hz and 32.768KHz. The rate interrupts are signaled is determined by the SPISR field in the Sub-Second Programmable Interrupt Select Register. See Table 18-9, "SPISR Encoding". The interrupt signal is always generated by the Week Timer if the block is enabled; the interrupt is enabled or disabled in the Interrupt Aggregator.

18.7 Low Power Modes

The Week Alarm has no low-power modes. It runs continuously while the VBAT well is powered.

18.8 Power-Up Events

The Week Timer POWER_UP_EVENT can be used to power up the system after a timed interval. The POWER_UP_EVENT is routed to the VBAT-Powered Control Interface (VCI). The VCI_OUT pin that is part of the VCI is asserted if the POWER_UP_EVENT is asserted.

The POWER_UP_EVENT can be asserted under the following two conditions:

1. The Week Alarm Counter Register is greater than or equal to the Week Timer Compare Register
2. The Sub-Week Alarm Counter Register decrements from '1' to '0'

The assertion of the POWER_UP_EVENT is inhibited if the POWERUP_EN field in the Control Register is '0'

Once a POWER_UP_EVENT is asserted the POWERUP_EN bit must be cleared to reset the output. Clearing POWERUP_EN is necessary to avoid unintended power-up cycles.

18.9 Description

The Week Alarm block provides battery-powered timekeeping functions, derived from a low-power 32KHz clock, that operate even when the device's main power is off. The block contains a set of counters that can be used to generate one-shot and periodic interrupts to the EC for periods ranging from about 30 microseconds to over 8 years. The Week Alarm can be used in conjunction with the VBAT-Powered Control Interface to power up a sleeping system after a configurable period.

In addition to basic timekeeping, the Week Alarm block can be used to control the battery-powered general purpose BGPO outputs.

18.9.1 INTERNAL COUNTERS

The Week Timer includes 3 counters:

18.9.1.1 28-bit Week Alarm Counter

This counter is 28 bits wide. The clock for this counter is the overflow of the Clock Divider, and as long as the Week Timer is enabled, it is incremented at a 1 Hz rate.

Both an interrupt and a power-up event can be generated when the contents of this counter matches the contents of the Week Timer Compare Register.

18.9.1.2 9-bit Sub-Week Alarm Counter

This counter is 9 bits wide. It is decremented by 1 at each tick of its selected clock. It can be configured either as a one-shot or repeating event generator.

Both an interrupt and a power-up event can be generated when this counter decrements from 1 to 0.

The Sub-Week Alarm Counter can be configured with a number of different clock sources for its time base, derived from either the Week Alarm Counter or the Clock Divider, by setting the SUBWEEK_TICK field of the Sub-Week Control Register.

TABLE 18-7: SUB-WEEK ALARM COUNTER CLOCK

SUBWEEK_ TICK	Source	SPISR	Frequency	Minimum Duration	Maximum Duration
0	Counter Disabled				
1	Sub-Second	0	Counter Disabled		
		1	2 Hz	500 ms	255.5 sec
		2	4 Hz	250 ms	127.8 sec
		3	8 Hz	125 ms	63.9 sec
		4	16 Hz	62.5	31.9 sec
		5	32 Hz	31.25 ms	16.0 sec
		6	64 Hz	15.6 ms	8 sec
		7	128 Hz	7.8 ms	4 sec
		8	256 Hz	3.9 ms	2 sec
		9	512 Hz	1.95 ms	1 sec
		10	1024 Hz	977 µS	499 ms
		11	2048 Hz	488 µS	249.5 ms
		12	4096 Hz	244 µS	124.8 ms
		13	8192 Hz	122 µS	62.4 ms
		14	16.384 KHz	61.1 µS	31.2 ms
		15	32.768 KHz	30.5 µS	15.6 ms
2	Second	n/a	1 Hz	1 sec	511 sec
3	Reserved				
4	Week Counter bit 3	n/a	125 Hz	8 sec	68.1 min
5	Week Counter bit 5	n/a	31.25 Hz	32 sec	272.5 min
6	Week Counter bit 7	n/a	7.8125 Hz	128 sec	18.17 hour
7	Week Counter bit 9	n/a	1.95 Hz	512 sec	72.68 hour

Note 1: The Week Alarm Counter **must not** be modified by firmware if Sub-Week Alarm Counter is using the Week Alarm Counter as its clock source (i.e., the SUBWEEK_TICK field is set to any of the values 4, 5, 6 or 7). The Sub-Week Alarm Counter must be disabled before changing the Week Alarm Counter. For example, the following sequence may be used:

1.Write 0h to the Sub-Week Alarm Counter Register (disabling the Sub-Week Counter)

2.Write the Week Alarm Counter Register

3.Write a new value to the Sub-Week Alarm Counter Register, restarting the Sub-Week Counter

18.9.1.3 15-bit Clock Divider

This counter is 15 bits wide. The clock for this counter is 32KHz, and as long as the Week Timer is enabled, it is incremented at 32.768KHz rate. The Clock Divider automatically The Clock Divider generates a clock out of 1 Hz when the counter wraps from 7FFFh to 0h.

By selecting one of the 15 bits of the counter, using the Sub-Second Programmable Interrupt Select Register, the Clock Divider can be used either to generate a time base for the Sub-Week Alarm Counter or as an isochronous interrupt to the EC, the SUB_SECOND interrupt. See Table 18-9, "SPISR Encoding" for a list of available frequencies.

18.9.2 TIMER VALID STATUS

If power on reset occurs on the VBAT power rail while the main device power is off, the counters in the Week Alarm are invalid. If firmware detects a POR on the VBAT power rail after a system boot, by checking the status bits in the Power, Clocks and Resets registers, the Week Alarm block must be reinitialized.

18.9.3 APPLICATION NOTE: REGISTER TIMING

Register writes in the Week Alarm complete within two cycles of the 32KHz clock.The write completes even if the main system clock is stopped before the two cycles of the 32K clock complete. Register reads complete in one cycle of the internal bus clock.

All Week Alarm interrupts that are asserted within the same cycle of the 32KHz clock are synchronously asserted to the EC.

18.9.4 APPLICATION NOTE: USE OF THE WEEK TIMER AS A 43-BIT COUNTER

The Week Timer cannot be directly used as a 42-bit counter that is incremented directly by the 32.768KHz clock domain. The upper 28 bits (28-bit Week Alarm Counter) are incremented at a 1Hz rate and the lower 16 bits (15-bit Clock Divider) are incremented at a 32.768KHz rate, but the increments are not performed in parallel. In particular, the upper 28 bits are incremented when the lower 15 bits increment from 0 to 1, so as long as the Clock Divider Register is 0 the two registers together, treated as a single value, have a smaller value then before the lower register rolled over from 7FFFh to 0h.

The following code can be used to treat the two registers as a single large counter. This example extracts a 32-bit value from the middle of the 43-bit counter:

```
dword TIME_STAMP(void)
{
    AHB_dword wct_value;
    AHB_dword cd_value1;
    AHB_dword cd_value2;
    dword irqEnableSave;

    //Disable interrupts
    irqEnableSave = IRQ_ENABLE;
    IRQ_ENABLE = 0;

    //Read 15-bit clk divider reading register, save result in A
    cd_value1 = WTIMER->CLOCK_DIVIDER;
    //Read 28 bit up-counter timer register, save result in B
    wct_value = WTIMER->WEEK_COUNTER_TIMER;
    //Read 15-bit clk divider reading register, save result in C
    cd_value2 = WTIMER->CLOCK_DIVIDER;

    if (0 == cd_value2)
```

```
    {
        wct_value = wct_value + 1;
    }
    else if ( (cd_value2 < cd_value1) || (0 == cd_value1))
    {
        wct_value = WTIMER->WEEK_COUNTER_TIMER;
    }

    //Enable interrupts
    IRQ_ENABLE = irqEnableSave;

    return (WTIMER_BASE + ((wct_value << 10) | (cd_value2>>5)));
}
```

18.10 Battery-Powered General Purpose Outputs

The Week Timer contains the control logic for Battery-Powered General Purposes Outputs (BGPOs). These are output-only pins whose state can be controlled by firmware and preserved when the device is operating on VBAT power alone.

When a BGPO function is selected on a pin that can also serve as a GPIO, using the BGPO Power Register, the GPIO input register is readable but always returns a '1b'.

18.11 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the Week Timer Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 18-8: REGISTER SUMMARY

Offset	Register Name
00h	Control Register
04h	Week Alarm Counter Register
08h	Week Timer Compare Register
0Ch	Clock Divider Register
10h	Sub-Second Programmable Interrupt Select Register
14h	Sub-Week Control Register
18h	Sub-Week Alarm Counter Register
1Ch	BGPO Data Register
20h	BGPO Power Register
24h	BGPO Reset Register

18.11.1 CONTROL REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:7	Reserved	R	-	-
6	POWERUP_EN This bit controls the state of the Power-Up Event Output and enables Week POWER-UP Event decoding in the VBAT-Powered Control Interface. See Section 18.8, "Power-Up Events" for a functional description of the POWER-UP_EN bit. 1=Power-Up Event Output Enabled 0=Power-Up Event Output Disabled and Reset	R/W	00h	RESET_VBAT
5:1	Reserved	R	-	-
0	WT_ENABLE The WT_ENABLE bit is used to start and stop the Week Alarm Counter Register and the Clock Divider Register. The value in the Counter Register is held when the WT_ENABLE bit is not asserted (‘0’) and the count is resumed from the last value when the bit is asserted (‘1’). The 15-Bit Clock Divider is reset to 00h and the Week Alarm Interface is in its lowest power consumption state when the WT_ENABLE bit is not asserted.	R/W	1h	RESET_VBAT

18.11.2 WEEK ALARM COUNTER REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:28	Reserved	R	-	-
27:0	WEEK_COUNTER While the WT_ENABLE bit is ‘1’, this register is incremented at a 1 Hz rate. Writes of this register may require one second to take effect. Reads return the current state of the register. Reads and writes complete independently of the state of WT_ENABLE.	R/W	00h	RESET_VBAT

18.11.3 WEEK TIMER COMPARE REGISTER

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:28	Reserved	R	-	-
27:0	WEEK_COMPARE A Week Alarm Interrupt and a Week Alarm Power-Up Event are asserted when the Week Alarm Counter Register is greater than or equal to the contents of this register. Reads and writes complete independently of the state of WT_ENABLE.	R/W	FFFFFFFh	RESET_VBAT

18.11.4 CLOCK DIVIDER REGISTER

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:15	Reserved	R	-	-
14:0	CLOCK_DIVIDER Reads of this register return the current state of the Week Timer 15-bit clock divider.	R	-	RESET_VBAT

18.11.5 SUB-SECOND PROGRAMMABLE INTERRUPT SELECT REGISTER

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:4	Reserved	R	-	-
3:0	SPISR This field determines the rate at which Sub-Second interrupt events are generated. Table 18-9, "SPISR Encoding" shows the relation between the SPISR encoding and Sub-Second interrupt rate.	R/W	00h	RESET_VBAT

TABLE 18-9: SPISR ENCODING

SPISR Value	Sub-Second Interrupt Rate, Hz	Interrupt Period
0	Interrupts disabled	
1	2	500 ms
2	4	250 ms
3	8	125 ms

TABLE 18-9: SPISR ENCODING (CONTINUED)

SPISR Value	Sub-Second Interrupt Rate, Hz	Interrupt Period
4	16	62.5 ms
5	32	31.25 ms
6	64	15.63 ms
7	128	7.813 ms
8	256	3.906 ms
9	512	1.953 ms
10	1024	977 µS
11	2048	488 µS
12	4096	244 µS
13	8192	122 µS
14	16384	61 µS
15	32768	30.5 µS

18.11.6 SUB-WEEK CONTROL REGISTER

Offset	14h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:10	Reserved	R	-	-
9:7	SUBWEEK_TICK This field selects the clock source for the Sub-Week Counter. See Table 18-7, "Sub-Week Alarm Counter Clock" for the description of the options for this field. See also Note 1.	R/W	0	RESET_VBAT
6	AUTO_RELOAD 1= No reload occurs when the Sub-Week Counter expires 0= Reloads the SUBWEEK_COUNTER_LOAD field into the Sub-Week Counter when the counter expires.	R/W	0	RESET_VBAT
5	TEST Must always be written with 0.	R/W	0	-
4:2	Reserved	R	-	-

Offset	14h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
1	WEEK_TIMER_POWERUP_EVENT_STATUS This bit is set to '1' when the Week Alarm Counter Register is greater than or equal the contents of the Week Timer Compare Register and the POWERUP_EN is '1'. Writes of '1' clear this bit. Writes of '0' have no effect. **Note:** This bit does not have to be cleared to remove a Week Timer Power-Up Event.	R/WC	0	RESET_VBAT
0	SUBWEEK_TIMER_POWERUP_EVENT_STATUS This bit is set to '1' when the Sub-Week Alarm Counter Register decrements from '1' to '0' and the POWERUP_EN is '1'. Writes of '1' clear this bit. Writes of '0' have no effect. **Note:** This bit MUST be cleared to remove a Sub-Week Timer Power-Up Event.	R/WC	0	RESET_VBAT

18.11.7 SUB-WEEK ALARM COUNTER REGISTER

Offset	18h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:25	Reserved	R	-	-
24:16	SUBWEEK_COUNTER_STATUS Reads of this register return the current state of the 9-bit Sub-Week Alarm counter.	R	00h	RESET_VBAT
15:9	Reserved	R	-	-
8:0	SUBWEEK_COUNTER_LOAD Writes with a non-zero value to this field reload the 9-bit Sub-Week Alarm counter. Writes of 0 disable the counter. If the Sub-Week Alarm counter decrements to 0 and the AUTO_RELOAD bit is set, the value in this field is automatically loaded into the Sub-Week Alarm counter.	R/W	00h	RESET_VBAT

18.11.8 BGPO DATA REGISTER

Offset	1Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:10	Reserved	R	-	-
9:0	BGPO Battery powered General Purpose Output. Each output pin may be individually configured to be either a VBAT-power BGPO or a VTR-powered GPIO, based on the corresponding settings in the BGPO Power Register. Additionally, each output pin may be individually configured to reset to 0 on either RESET_VBAT or RESET_SYS, based on the corresponding settings in the BGPO Reset Register. For each bit [*i*] in the field: 1=BGPO[*i*] output is high 0=BGPO[*i*] output is low If a BGPO[*i*] does not appear in a package, the corresponding bit **must** be written with a 0 or undesirable results will occur.	R/W	0h	RESET_VBAT or RESET_SYS

18.11.9 BGPO POWER REGISTER

Offset	20h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:6	Reserved	R	-	-
5:1	BGPO_POWER Battery powered General Purpose Output power source. For each bit [*i*] in the field: 1=BGPO[*i*] is powered by VBAT. The BGPO[*i*] pin is always determined by the corresponding bit in the BGPO Data Register. The GPIO Input register for the GPIO that is multiplexed with the BGPO always returns the value of the pin associated with BGPO[*i*]. 0=The pin for BGPO[*i*] functions as a GPIO. When VTR is powered, the pin associated with BGPO[*i*] is determined by the GPIO associated with the pin. When VTR is unpowered, the pin is tri-stated	R/W	1Fh	RESET_VBAT
0	Reserved	R	-	-

Note: Because BGPO[9:6] and BGPO0 are not multiplexed with GPIOs, bits 9:6 and 0 are reserved.

18.11.10 BGPO RESET REGISTER

Offset	24h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:10	Reserved	R	-	-
9:0	BGPO_RESET Battery powered General Purpose Output reset event. For each bit [*i*] in the field: 1=BGPO[*i*] is reset to 0 on RESET_VBAT 0=BGPO[*i*] is reset to 0 on RESET_SYS	R/W	0h	RESET_VBAT

19.0 TACH

19.1 Introduction

This block monitors TACH output signals (or locked rotor signals) from various types of fans, and determines their speed.

19.2 References

No references have been cited for this feature.

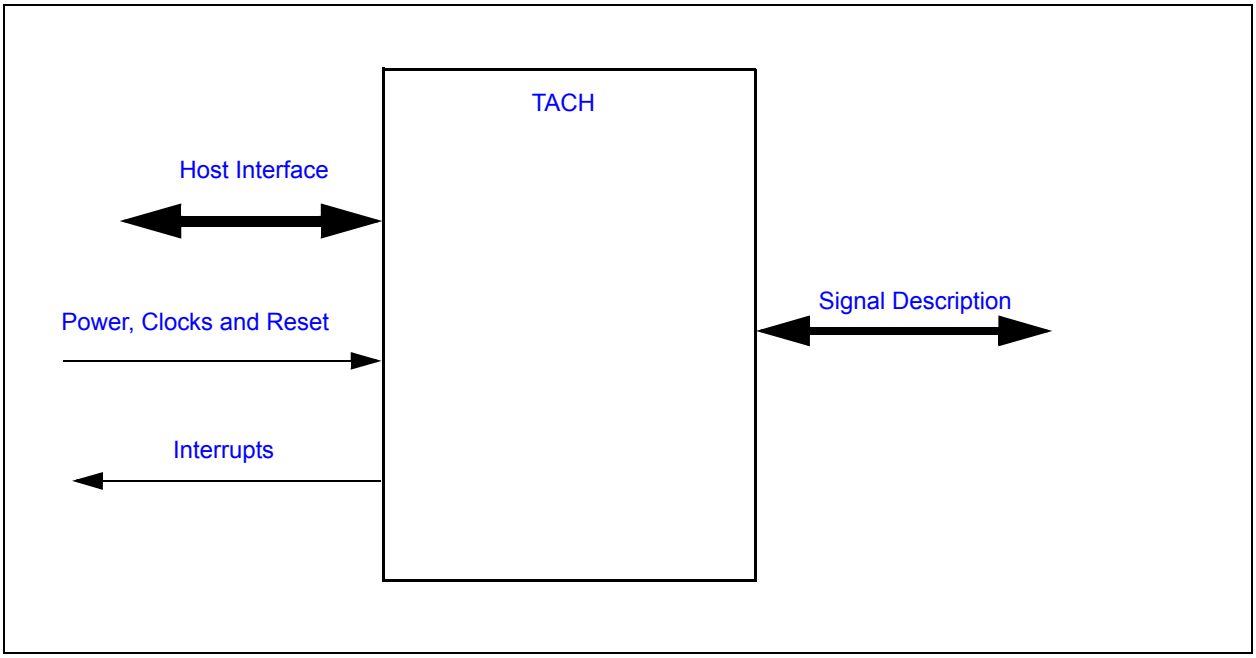
19.3 Terminology

There is no terminology defined for this section.

19.4 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

FIGURE 19-1: I/O DIAGRAM OF BLOCK



19.5 Signal Description

TABLE 19-1: SIGNAL DESCRIPTION

Name	Direction	Description
TACH INPUT	Input	Tachometer signal from TACHx Pin.

19.6 Host Interface

The registers defined for the TACH are accessible by the various hosts as indicated in Section 19.11, "EC Registers".

19.7 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

19.7.1 POWER DOMAINS

Name	Description
VTR	The logic and registers implemented in this block are powered by this power well.

19.7.2 CLOCK INPUTS

Name	Description
100KHz	This is the clock input to the tachometer monitor logic. In Mode 1, the TACH is measured in the number of these clocks. This clock is derived from the main clock domain.

19.7.3 RESETS

Name	Description
RESET_SYS	This signal resets all the registers and logic in this block to their default state.

19.8 Interrupts

This section defines the Interrupt Sources generated from this block.

TABLE 19-2: EC INTERRUPTS

Source	Description
TACH	This internal signal is generated from the OR'd result of the status events, as defined in the TACHx Status Register.

19.9 Low Power Modes

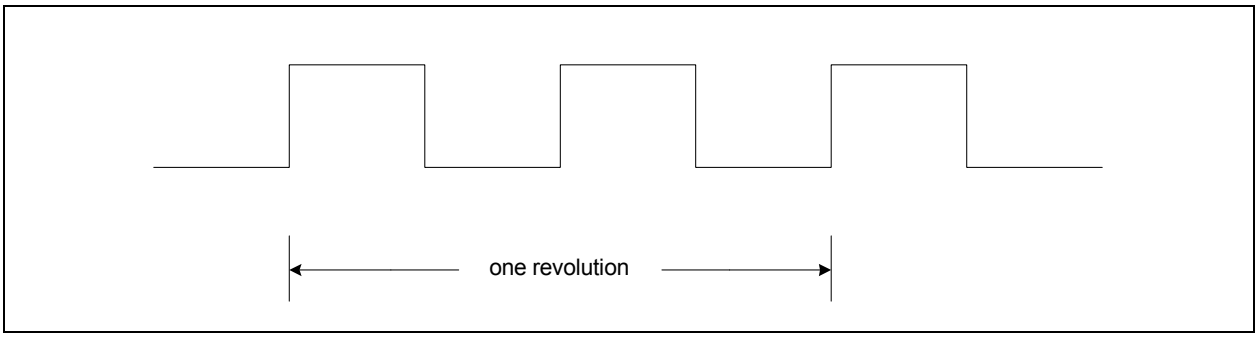
The TACH may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry.

19.10 Description

The TACH block monitors Tach output signals or locked rotor signals generated by various types of fans. These signals can be used to determine the speed of the attached fan. This block is designed to monitor fans at fan speeds from 100 RPMs to 30,000 RPMs.

Typically, these are DC brushless fans that generate (with each revolution) a 50% duty cycle, two-period square wave, as shown in Figure 19-2 below.

FIGURE 19-2: FAN GENERATED 50%DUTY CYCLE WAVEFORM



In typical systems, the fans are powered by the main power supply. Firmware may disable this block when it detects that the main power rail has been turned off by either clearing the <enable> TACH_ENABLE bit or putting the block to sleep via the supported Low Power Mode interface (see Low Power Modes).

19.10.1 MODES OF OPERATION

The Tachometer block supports two modes of operation. The mode of operation is selected via the TACH_READING_-MODE_SELECT bit.

19.10.1.1 Free Running Counter

In Mode 0, the Tachometer block uses the TACH input as the clock source for the internal TACH pulse counter (see TACHX_COUNTER). The counter is incremented when it detects a rising edge on the TACH input. In this mode, the firmware may periodically poll the TACHX_COUNTER field to determine the average speed over a period of time. The firmware must store the previous reading and the current reading to compute the number of pulses detected over a period of time. In this mode, the counter continuously increments until it reaches FFFFh. It then wraps back to 0000h and continues counting. The firmware must ensure that the sample rate is greater than the time it takes for the counter to wrap back to the starting point.

Note:	Tach interrupts should be disabled in Mode 0.

19.10.1.2 Mode 1 -- Number of Clock Pulses per Revolution

In Mode 1, the Tachometer block uses its 100KHz clock input to measure the programmable number of TACH pulses. In this mode, the internal TACH pulse counter (TACHX_COUNTER) returns the value in number of 100KHz pulses per programmed number of TACH_EDGES. For fans that generate two square waves per revolution, these bits should be configured to five edges.

When the number of edges is detected, the counter is latched and the COUNT_READY_STATUS bit is asserted. If the COUNT_READY_INT_EN bit is set a TACH interrupt event will be generated.

19.10.2 OUT-OF-LIMIT EVENTS

The TACH Block has a pair of limit registers that may be configured to generate an event if the Tach indicates that the fan is operating too slow or too fast. If the <TACH reading> exceeds one of the programmed limits, the TACHx High Limit Register and the TACHx Low Limit Register, the bit TACH_OUT_OF_LIMIT_STATUS will be set. If the TACH_OUT_OF_LIMIT_STATUS bit is set, the Tachometer block will generate an interrupt event.

19.11 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the TACH Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 19-3: REGISTER SUMMARY

Offset	Register Name
00h	TACHx Control Register
04h	TACHx Status Register
08h	TACHx High Limit Register
0Ch	TACHx Low Limit Register

19.11.1 TACHX CONTROL REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	TACHX_COUNTER This 16-bit field contains the latched value of the internal Tach pulse counter, which may be configured by the Tach Reading Mode Select field to operate as a free-running counter or to be gated by the Tach input signal. If the counter is free-running (Mode 0), the internal Tach counter increments (if enabled) on transitions of the raw Tach input signal and is latched into this field every time it is incremented. The act of reading this field will not reset the counter, which rolls over to 0000h after FFFFh. The firmware will compute the delta between the current count reading and the previous count reading, to determine the number of pulses detected over a programmed period. If the counter is gated by the Tach input and clocked by 100KHz (Mode 1), the internal counter will be latched into the reading register when the programmed number of edges is detected or when the counter reaches FFFFh. The internal counter is reset to zero after it is copied into this register. **Note:** In Mode 1, a counter value of FFFFh means that the Tach did not detect the programmed number of edges in 655ms. A stuck fan can be detected by setting the TACHx High Limit Register to a number less than FFFFh. If the internal counter then reaches FFFFh, the reading register will be set to FFFFh and an out-of-limit interrupt can be sent to the EC.	R	00h	RESET_SYS
15	TACH_INPUT_INT_EN 1=Enable Tach Input toggle interrupt from Tach block 0=Disable Tach Input toggle interrupt from Tach block	R/W	0b	RESET_SYS
14	COUNT_READY_INT_EN 1=Enable Count Ready interrupt from Tach block 0=Disable Count Ready interrupt from Tach block	R/W	0b	RESET_SYS
13	Reserved	R	-	-
12:11	TACH_EDGES A Tach signal is a square wave with a 50% duty cycle. Typically, two Tach periods represents one revolution of the fan. A Tach period consists of three Tach edges. This programmed value represents the number of Tach edges that will be used to determine the interval for which the number of 100KHz pulses will be counted 11b=9 Tach edges (4 Tach periods) 10b=5 Tach edges (2 Tach periods) 01b=3 Tach edges (1 Tach period) 00b=2 Tach edges (1/2 Tach period)	R/W	00b	RESET_SYS

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
10	TACH_READING_MODE_SELECT 1=Counter is incremented on the rising edge of the 100KHz input. The counter is latched into the TACHX_COUNTER field and reset when the programmed number of edges is detected. 0=Counter is incremented when Tach Input transitions from low-to-high state (default)	R/W	0b	RESET_SYS
9	Reserved	R	-	-
8	FILTER_ENABLE This filter is used to remove high frequency glitches from Tach Input. When this filter is enabled, Tach input pulses less than two 100KHz periods wide get filtered. 1=Filter enabled 0=Filter disabled (default) It is recommended that the Tach input filter always be enabled.	R/W	0b	RESET_SYS
7:2	Reserved	R	-	-
1	TACH_ENABLE This bit gates the clocks into the block. When clocks are gated, the TACHx pin is tristated. When re-enabled, the internal counters will continue from the last known state and stale status events may still be pending. Firmware should discard any status or reading values until the reading value has been updated at least one time after the enable bit is set. 1=TACH Monitoring enabled, clocks enabled. 0=TACH Idle, clocks gated	R/W	0b	RESET_SYS
0	TACH_OUT_OF_LIMIT_ENABLE This bit is used to enable the TACH_OUT_OF_LIMIT_STATUS bit in the TACHx Status Register to generate an interrupt event. 1=Enable interrupt output from Tach block 0=Disable interrupt output from Tach block (default)	R/W	0b	RESET_SYS

19.11.2 TACHX STATUS REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:4	Reserved	R	-	-
3	COUNT_READY_STATUS This status bit is asserted when the Tach input changes state and when the counter value is latched. This bit remains cleared to '0' when the TACH_READING_MODE_SELECT bit in the TACHx Control Register is '0'. When the TACH_READING_MODE_SELECT bit in the TACHx Control Register is set to '1', this bit is set to '1' when the counter value is latched by the hardware. It is cleared when written with a '1'. If COUNT_READY_INT_EN in the TACHx Control Register is set to 1, this status bit will assert the Tach Interrupt signal. 1=Reading ready 0=Reading not ready	R/WC	0b	RESET_SYS
2	TOGGLE_STATUS This bit is set when Tach Input changes state. It is cleared when written with a '1b'. If TACH_INPUT_INT_EN in the TACHx Control Register is set to '1b', this status bit will assert the Tach Interrupt signal. 1=Tach Input changed state (this bit is set on a low-to-high or high-to-low transition) 0=Tach stable	R/WC	0b	RESET_SYS
1	TACH_PIN_STATUS This bit reflects the state of Tach Input. This bit is a read only bit that may be polled by the embedded controller. 1=Tach Input is high 0=Tach Input is low	R	0b	RESET_SYS
0	TACH_OUT_OF_LIMIT_STATUS This bit is set when the Tach Count value is greater than the high limit or less than the low limit. It is cleared when written with a '1b'. To disable this status event set the limits to their extreme values. If TACH_OUT_OF_LIMIT_ENABLE in the TACHx Control Register is set to 1', this status bit will assert the Tach Interrupt signal. 1=Tach is outside of limits 0=Tach is within limits	R/WC	0b	RESET_SYS

Note:

- Some fans offer a Locked Rotor output pin that generates a level event if a locked rotor is detected. This bit may be used in combination with the Tach pin status bit to detect a locked rotor signal event from a fan.
- Tach Input may come up as active for Locked Rotor events. This would not cause an interrupt event because the pin would not toggle. Firmware must read the status events as part of the initialization process, if polling is not implemented.

19.11.3 TACHX HIGH LIMIT REGISTER

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	Reserved	-	-	-
15:0	TACH_HIGH_LIMIT This value is compared with the value in the TACHX_COUNTER field. If the value in the counter is greater than the value programmed in this register, the TACH_OUT_OF_LIMIT_STATUS bit will be set. The TACH_OUT_OF_LIMIT_STATUS status event may be enabled to generate an interrupt to the embedded controller via the TACH_OUT_OF_LIMIT_ENABLE bit in the TACHx Control Register.	R/W	FFFFh	RESET_SYS

19.11.4 TACHX LOW LIMIT REGISTER

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	Reserved	R	-	-
15:0	TACHX_LOW_LIMIT This value is compared with the value in the TACHX_COUNTER field of the TACHx Control Register. If the value in the counter is less than the value programmed in this register, the TACH_OUT_OF_LIMIT_STATUS bit will be set. The TACH_OUT_OF_LIMIT_STATUS status event may be enabled to generate an interrupt to the embedded controller via the TACH_OUT_OF_LIMIT_ENABLE bit in the TACHx Control Register To disable the TACH_OUT_OF_LIMIT_STATUS low event, program 0000h into this register.	R/W	0000h	RESET_SYS

20.0 PWM

20.1 Introduction

This block generates a PWM output that can be used to control 4-wire fans, blinking LEDs, and other similar devices. Each PWM can generate an arbitrary duty cycle output at frequencies from less than 0.1 Hz to 24 MHz.

The PWMx Counter ON Time registers and PWMx Counter OFF Time registers determine the operation of the PWM_OUTPUT signals. See Section 20.11.1, "PWMx Counter ON Time Register" and Section 20.11.2, "PWMx Counter OFF Time Register" for a description of the PWM_OUTPUT signals.

20.2 References

There are no standards referenced in this chapter.

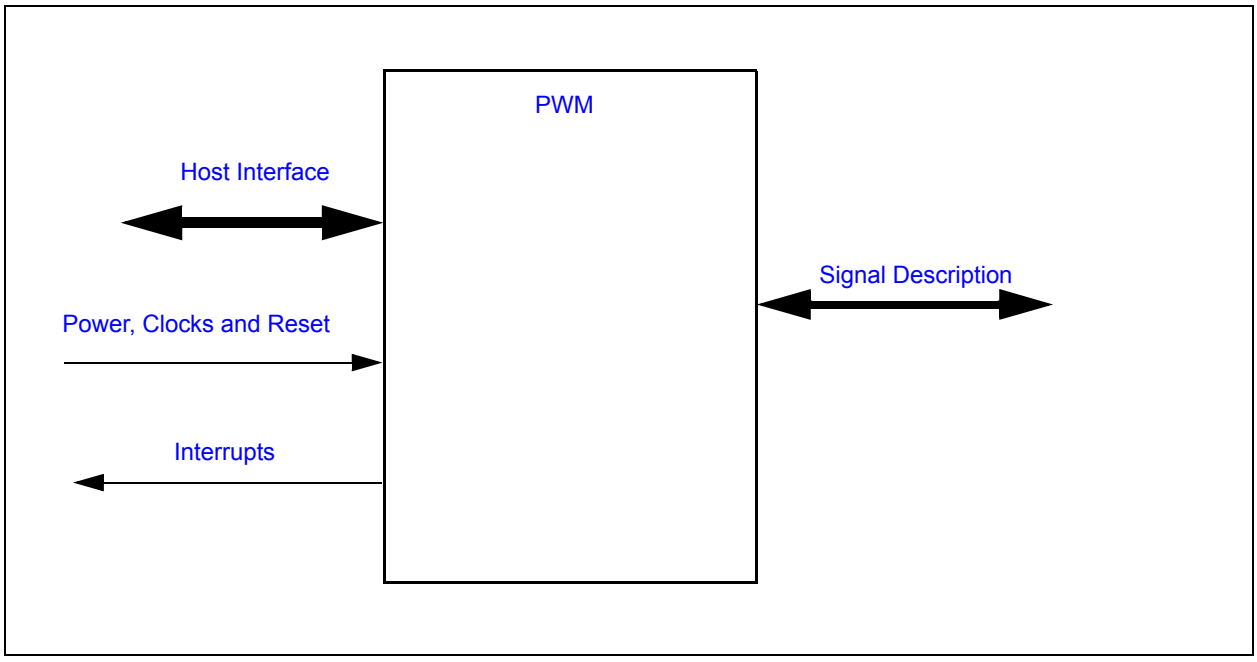
20.3 Terminology

There is no terminology defined for this section.

20.4 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

FIGURE 20-1: I/O DIAGRAM OF BLOCK



There are no external signals for this block.

20.5 Signal Description

TABLE 20-1: SIGNAL DESCRIPTION

Name	Direction	Description
PWM_OUTPUT	OUTPUT	Pulse Width Modulated signal to PWMx pin.

20.6 Host Interface

The registers defined for the PWM Interface are accessible by the various hosts as indicated in Section 20.11, "EC Registers".

20.7 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

20.7.1 POWER DOMAINS

Name	Description
VTR	The logic and registers implemented in this block are powered by this power well.

20.7.2 CLOCK INPUTS

Name	Description
48MHz	Clock input for generating high PWM frequencies, such as 15 kHz to 30 kHz.
100KHz	This is the clock input for generating low PWM frequencies, such as 10 Hz to 100 Hz.

20.7.3 RESETS

Name	Description
RESET_SYS	This signal resets all the registers and logic in this block to their default state.

20.8 Interrupts

The PWM block does not generate any interrupt events.

20.9 Low Power Modes

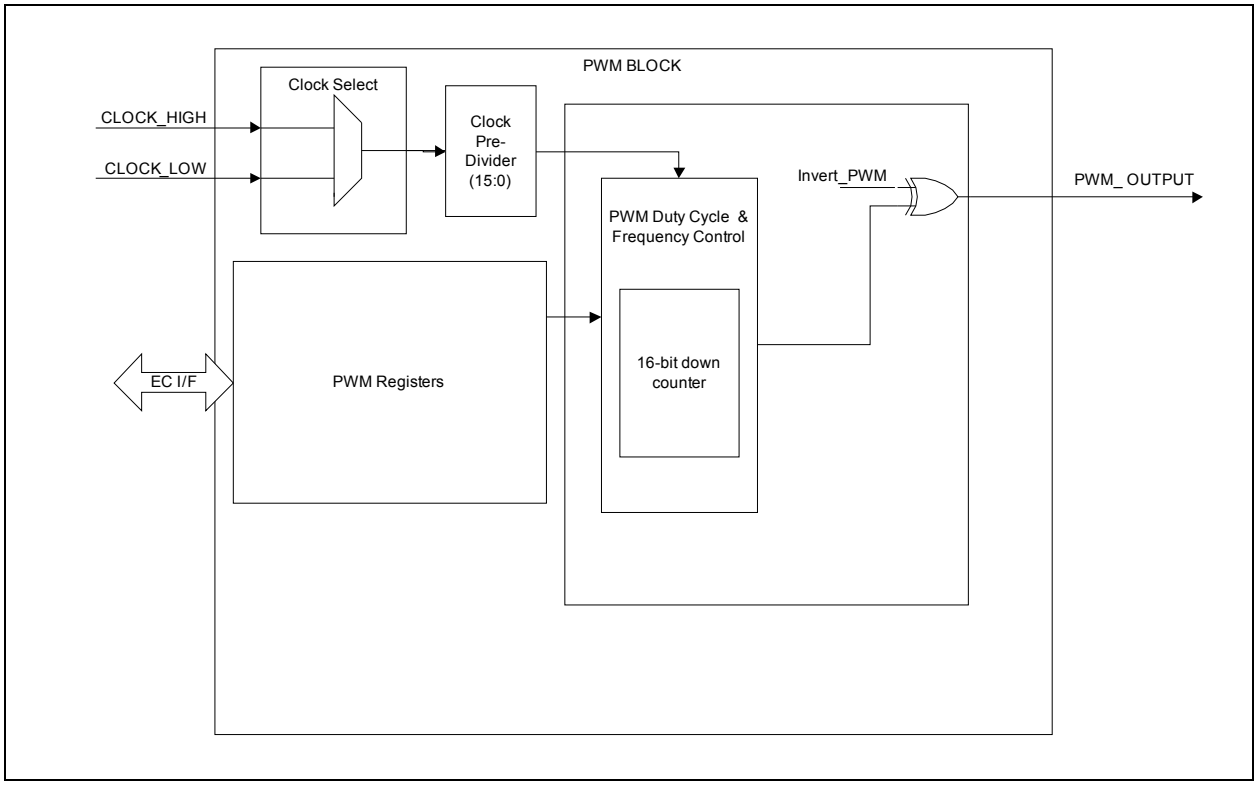
The PWM may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry. When the PWM is in the sleep state, the internal counters reset to 0 and the internal state of the PWM and the PWM_OUTPUT signal set to the OFF state.

20.10 Description

The PWM_OUTPUT signal is used to generate a duty cycle of specified frequency. This block can be programmed so that the PWM signal toggles the PWM_OUTPUT, holds it high, or holds it low. When the PWM is configured to toggle, the PWM_OUTPUT alternates from high to low at the rate specified in the PWMx Counter ON Time Register and PWMx Counter OFF Time Register.

The following diagram illustrates how the clock inputs and registers are routed to the PWM Duty Cycle & Frequency Control logic to generate the PWM output.

FIGURE 20-2: BLOCK DIAGRAM OF PWM CONTROLLER



Note: In Figure 20-2, the 48MHz clock is represented as CLOCK_HIGH and the 100KHz clock is represented as CLOCK_LOW.

The PWM clock source to the PWM Down Counter, used to generate a duty cycle and frequency on the PWM, is determined through the Clock select[1] and Clock Pre-Divider[6:3] bits in the PWMx Configuration Register register.

The PWMx Counter ON/OFF Time registers determine both the frequency and duty cycle of the signal generated on PWM_OUTPUT as described below.

The PWM frequency is determined by the selected clock source and the total on and off time programmed in the PWMx Counter ON Time Register and PWMx Counter OFF Time Register registers. The frequency is the time it takes (at that clock rate) to count down to 0 from the total on and off time.

The PWM duty cycle is determined by the relative values programmed in the PWMx Counter ON Time Register and PWMx Counter OFF Time Register registers.

The PWM Frequency Equation and PWM Duty Cycle Equation are shown below.

EQUATION 20-1: PWM FREQUENCY EQUATION

$$\text{PWM Frequency} = \frac{1}{(PreDivisor + 1)} \times \frac{(ClockSourceFrequency)}{((PWMCounterOnTime + 1) + (PWMCounterOffTime + 1))}$$

In this equation, the ClockSourceFrequency variable is the frequency of the clock source selected by the Clock Select bit in the PWMx Configuration Register, and PreDivisor is a field in the PWMx Configuration Register. The PWMCounterOnTime, PWMCounterOffTime are registers that are defined in Section 20.11, "EC Registers".

EQUATION 20-2: PWM DUTY CYCLE EQUATION

$$\text{PWM Duty Cycle} = \frac{(PWMCounterOnTime + 1)}{((PWMCounterOnTime + 1) + (PWMCounterOffTime + 1))}$$

The PWMx Counter ON Time Register and PWMx Counter OFF Time Register registers should be accessed as 16-bit values.

20.10.1 PWM REGISTER UPDATES

The PWMx Counter ON Time Register and PWMx Counter OFF Time Register may be updated at any time. Values written into the two registers are kept in holding registers. The holding registers are transferred into the two user-visible registers when all four bytes have been written with new values and the internal counter completes the OFF time count. If the PWM is in the Full On state then the two user-visible registers are updated from the holding registers as soon as all four bytes have been written. Once the two registers have been updated the holding registers are marked empty. and all four bytes must again be written before the holding registers will be reloaded into the On Time Register and the Off Time Register. Reads of both registers return the current contents of the registers that are used to load the counter and not the holding registers.

20.11 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the PWM Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 20-2: REGISTER SUMMARY

Offset	Register Name
00h	PWMx Counter ON Time Register
04h	PWMx Counter OFF Time Register
08h	PWMx Configuration Register

20.11.1 PWMX COUNTER ON TIME REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	Reserved	R	-	-
15:0	PWMX_COUNTER_ON_TIME This field determine both the frequency and duty cycle of the PWM signal. Setting this field to a value of *n* will cause the On time of the PWM to be *n*+1 cycles of the PWM Clock Source. When this field is set to zero and the PWMX_COUNTER_OFF_-TIME is not set to zero, the PWM_OUTPUT is held low (Full Off).	R/W	0000h	RESET_SYS

20.11.2 PWMX COUNTER OFF TIME REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	Reserved	R	-	-
15:0	PWMX_COUNTER_OFF_TIME This field determine both the frequency and duty cycle of the PWM signal. Setting this field to a value of *n* will cause the Off time of the PWM to be *n*+1 cycles of the PWM Clock Source. When this field is set to zero, the PWM_OUTPUT is held high (Full On).	R/W	FFFFh	RESET_SYS

20.11.3 PWMX CONFIGURATION REGISTER

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:7	Reserved	R	-	-
6:3	CLOCK_PRE_DIVIDER The Clock source for the 16-bit down counter (see PWMx Counter ON Time Register and PWMx Counter OFF Time Register) is determined by bit D1 of this register. The Clock source is then divided by the value of Pre-Divider+1 and the resulting signal determines the rate at which the down counter will be decremented. For example, a Pre-Divider value of 1 divides the input clock by 2 and a value of 2 divides the input clock by 3. A Pre-Divider of 0 will disable the Pre-Divider option.	R/W	0000b	RESET_SYS
2	INVERT 1=PWM_OUTPUT ON State is active low 0=PWM_OUTPUT ON State is active high	R/W	0b	RESET_SYS
1	CLOCK_SELECT This bit determines the clock source used by the PWM duty cycle and frequency control logic. 1=CLOCK_LOW 0=CLOCK_HIGH	R/W	0b	RESET_SYS
0	PWM_ENABLE When the PWM_ENABLE is set to 0 the internal counters are reset and the internal state machine is set to the OFF state. In addition, the PWM_OUTPUT signal is set to the inactive state as determined by the Invert bit. The PWMx Counter ON Time Register and PWMx Counter OFF Time Register are not affected by the PWM_ENABLE bit and may be read and written while the PWM enable bit is 0. 1=Enabled (default) 0=Disabled (gates clocks to save power)	R/W	0b	RESET_SYS

21.0 ANALOG TO DIGITAL CONVERTER

21.1 Introduction

This block is designed to convert external analog voltage readings into digital values. It consists of a single successive-approximation Analog-Digital Converter that can be shared among multiple inputs.

21.2 References

No references have been cited for this chapter

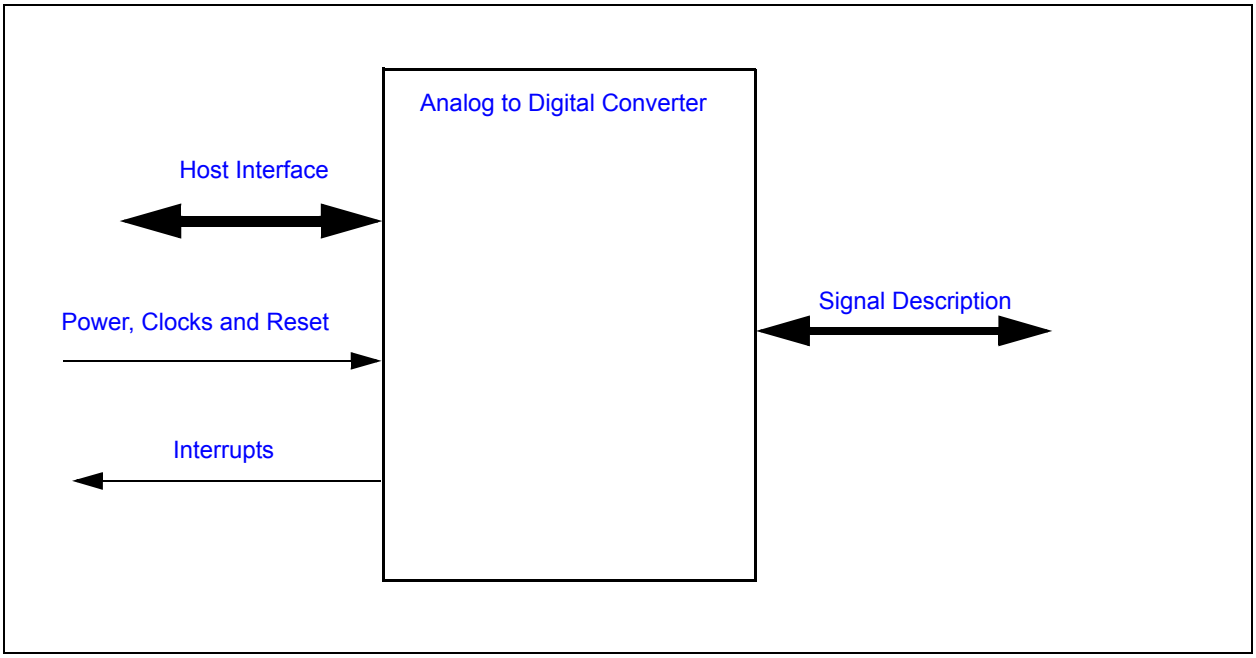
21.3 Terminology

No terminology is defined for this chapter

21.4 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

FIGURE 21-1: I/O DIAGRAM OF BLOCK



21.5 Signal Description

The Signal Description Table lists the signals that are typically routed to the pin interface.

TABLE 21-1: SIGNAL DESCRIPTION

Name	Direction	Description
ADC [15:0]	Input	ADC Analog Voltage Input from pins. **Note:** The ADC IP supports up to 16 channels. The number of channels implemented is package dependent. Refer to the Pin Chapter for the number of channels implemented in a package.

21.6 Host Interface

The registers defined for the ADC are accessible by the various hosts as indicated in Section 21.11, "EC Registers".

21.7 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

21.7.1 POWER DOMAINS

TABLE 21-2: POWER SOURCES

Name	Description
VTR	This power well supplies power for the registers tn this block.
VTR_ANALOG	This power well supplies power for the analog circuitry in this block.

21.7.2 CLOCK INPUTS

TABLE 21-3: CLOCK INPUTS

Name	Description
16MHz	This derived clock signal drives controls the conversion rate of the ADC. At 16MHz, the ADC does one channel conversion in 1.125µS.

21.7.3 RESETS

TABLE 21-4: RESET SIGNALS

Name	Description
RESET_SYS	This reset signal resets all of the registers and logic in this block.

21.8 Interrupts

TABLE 21-5: EC INTERRUPTS

Source	Description
ADC_Single_Int	Interrupt signal from ADC controller to EC for Single-Sample ADC conversion.
ADC_Repeat_Int	Interrupt signal from ADC controller to EC for Repeated ADC conversion.

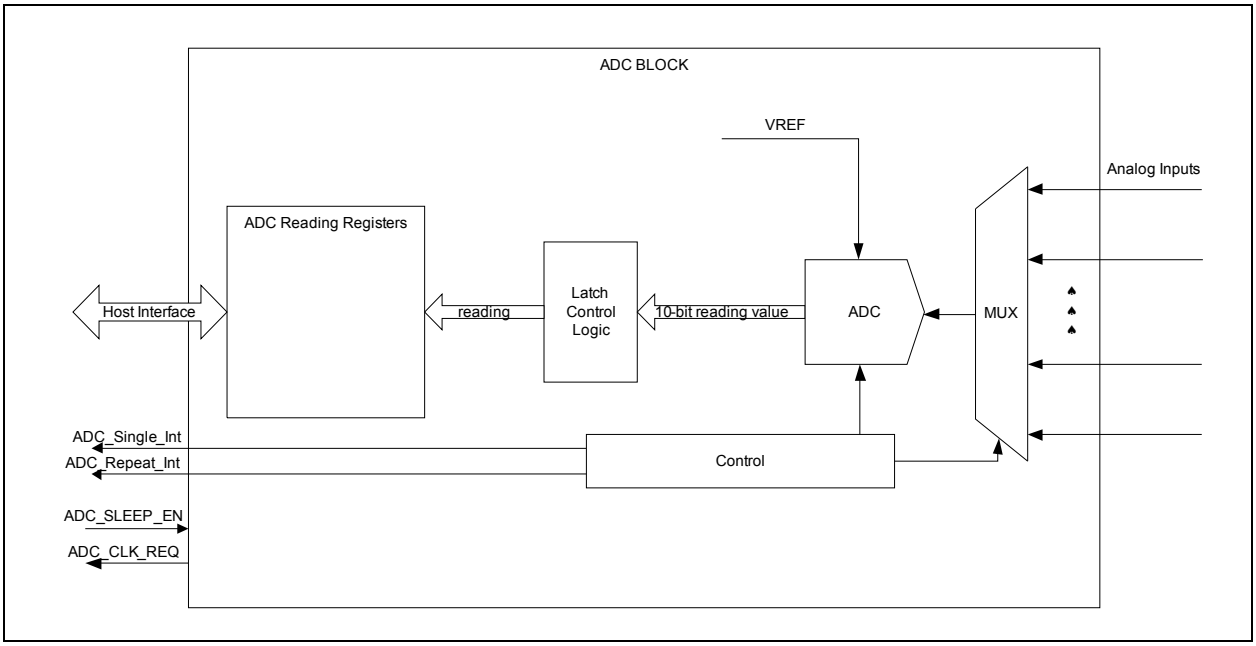
21.9 Low Power Modes

The ADC may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry.

The ADC is designed to conserve power when it is either sleeping or disabled. It is disabled via the ACTIVATE Bit and sleeps when the ADC_SLEEP_EN signal is asserted. The sleeping state only controls clocking in the ADC and does not power down the analog circuitry. For lowest power consumption, the ADC ACTIVATE bit must be set to '0.'

21.10 Description

FIGURE 21-2: ADC BLOCK DIAGRAM



The CEC1702 features a sixteen channel successive approximation Analog to Digital Converter. The ADC architecture features excellent linearity and converts analog signals to 10 bit words. Conversion takes less than 1.125 microseconds per 10-bit word. The sixteen channels are implemented with a single high speed ADC fed by a sixteen input analog multiplexer. The multiplexer cycles through the sixteen voltage channels, starting with the lowest-numbered channel and proceeding to the highest-number channel, selecting only those channels that are programmed to be active.

The input range on the voltage channels spans from 0V to the voltage reference. With an voltage reference of 3.3V, this provides resolutions of 3.2mV. The range can easily be extended with the aid of resistor dividers. The accuracy of any voltage reading depends on the accuracy and stability of the voltage reference input.

Note:	The ADC pins are 3.3V tolerant.

The ADC conversion cycle starts either when the START_SINGLE bit in the ADC to set to 1 or when the ADC Repeat Timer counts down to 0. When the START_SINGLE is set to 1 the conversion cycle converts channels enabled by configuration bits in the ADC Single Register. When the Repeat Timer counts down to 0 the conversion cycle converts channels enabled by configuration bits in the ADC Repeat Register. When both the START_SINGLE bit and the Repeat Timer request conversions the START_SINGLE conversion is completed first.

Conversions always start with the lowest-numbered enabled channel and proceed to the highest-numbered enabled channel.

Note:	If software repeatedly sets Start_Single to 1 at a rate faster than the Repeat Timer count down interval, the conversion cycle defined by the ADC Repeat Register will not be executed.

21.10.1 REPEAT MODE

- Repeat Mode will start a conversion cycle of all ADC channels enabled by bits RPT_EN in the ADC Repeat Register. The conversion cycle will begin after a delay determined by START_DELAY in the ADC Delay Register.
- After all channels enabled by RPT_EN are complete, REPEAT_DONE_STATUS will be set to 1. This status bit is cleared when the next repeating conversion cycle begins to give a reflection of when the conversion is in progress.

- As long as START_REPEAT is 1 the ADC will repeatedly begin conversion cycles with a period defined by REPEAT_DELAY.
- If the delay period expires and a conversion cycle is already in progress because START_SINGLE was written with a 1, the cycle in progress will complete, followed immediately by a conversion cycle using RPT_EN to control the channel conversions.

21.10.2 SINGLE MODE

- The Single Mode conversion cycle will begin without a delay. After all channels enabled by SINGLE_EN are complete, SINGLE_DONE_STATUS will be set to 1. When the next conversion cycle begins the bit is cleared.
- If START_SINGLE is written with a 1 while a conversion cycle is in progress because START_REPEAT is set, the conversion cycle will complete, followed immediately by a conversion cycle using SINGLE_EN to control the channel conversions.

21.10.3 APPLICATION NOTES

Transitions on ADC GPIOs are not permitted when Analog to Digital Converter readings are being taken.

Note:	ADC inputs require at least a 0.1 uF capacitor to filter glitches.

21.11 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the Analog to Digital Converter Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 21-6: REGISTER SUMMARY

Offset	Register Name
00h	ADC Control Register
04h	ADC Delay Register
08h	ADC Status Register
0Ch	ADC Single Register
10h	ADC Repeat Register
14h	ADC Channel 0 Reading Register
18h	ADC Channel 1 Reading Register
1Ch	ADC Channel 2 Reading Register
20h	ADC Channel 3 Reading Register
24h	ADC Channel 4 Reading Register
28h	ADC Channel 5 Reading Register
2Ch	ADC Channel 6 Reading Register
30h	ADC Channel 7 Reading Register
34h	ADC Channel 8 Reading Register
38h	ADC Channel 9 Reading Register
3Ch	ADC Channel 10 Reading Register
40h	ADC Channel 11 Reading Register
44h	ADC Channel 12 Reading Register
48h	ADC Channel 13 Reading Register
4Ch	ADC Channel 14 Reading Register
50h	ADC Channel 15 Reading Register

21.11.1 ADC CONTROL REGISTER

The ADC Control Register is used to control the behavior of the Analog to Digital Converter.

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:8	Reserved	R	-	-
7	SINGLE_DONE_STATUS This bit is cleared when it is written with a 1. Writing a 0 to this bit has no effect. This bit can be used to generate an EC interrupt. 1=ADC single-sample conversion is completed. This bit is set to 1 when all enabled channels in the single conversion cycle 0=ADC single-sample conversion is not complete. This bit is cleared whenever an ADC conversion cycle begins for a single conversion cycle	R/WC	0h	RESET_SYS
6	REPEAT_DONE_STATUS This bit is cleared when it is written with a 1. Writing a 0 to this bit has no effect. This bit can be used to generate an EC interrupt. 1=ADC repeat-sample conversion is completed. This bit is set to 1 when all enabled channels in a repeating conversion cycle complete 0=ADC repeat-sample conversion is not complete. This bit is cleared whenever an ADC conversion cycle begins for a repeating conversion cycle	R/WC	0h	RESET_SYS
5	Reserved	R	-	-
4	SOFT_RESET 1=writing one causes a reset of the ADC block hardware (not the registers) 0=writing zero takes the ADC block out of reset	R/W	0h	RESET_SYS
3	POWER_SAVER_DIS 1=Power saving feature is disabled 0=Power saving feature is enabled. The Analog to Digital Converter controller powers down the ADC between conversion sequences.	R/W	0h	RESET_SYS
2	START_REPEAT 1=The ADC Repeat Mode is enabled. This setting will start a conversion cycle of all ADC channels enabled by bits RPT_EN in the ADC Repeat Register. 0=The ADC Repeat Mode is disabled. Note: This setting will not terminate any conversion cycle in process, but will clear the Repeat Timer and inhibit any further periodic conversions.	R/W	0h	RESET_SYS

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
1	START_SINGLE 1=The ADC Single Mode is enabled. This setting starts a single conversion cycle of all ADC channels enabled by bits SINGLE_EN in the ADC Single Register. 0=The ADC Single Mode is disabled. This bit is self-clearing	R/W	0h	RESET_SYS
0	ACTIVATE 1=ADC block is enabled for operation. START_SINGLE or START_REPEAT can begin data conversions by the ADC. Note: A reset pulse is sent to the ADC core when this bit changes from 0 to 1. 0=The ADC is disabled and placed in its lowest power state. Note: Any conversion cycle in process will complete before the block is shut down, so that the reading registers will contain valid data but no new conversion cycles will begin.	R/W	0h	RESET_SYS

21.11.2 ADC DELAY REGISTER

The ADC Delay register determines the delay from setting START_REPEAT in the ADC Control Register and the start of a conversion cycle. This register also controls the interval between conversion cycles in repeat mode.

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	REPEAT_DELAY This field determines the interval between conversion cycles when START_REPEAT is 1. The delay is in units of 40µs. A value of 0 means no delay between conversion cycles, and a value of 0xFFFF means a delay of 2.6 seconds. This field has no effect when START_SINGLE is written with a 1.	R/W	0000h	RESET_SYS
15:0	START_DELAY This field determines the starting delay before a conversion cycle is begun when START_REPEAT is written with a 1. The delay is in units of 40µs. A value of 0 means no delay before the start of a conversion cycle, and a value of 0xFFFF means a delay of 2.6 seconds. This field has no effect when START_SINGLE is written with a 1.	R/W	0000h	RESET_SYS

21.11.3 ADC STATUS REGISTER

The ADC Status Register indicates whether the ADC has completed a conversion cycle.

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	Reserved	R	-	-
15:0	ADC_CH_STATUS All bits are cleared by being written with a '1'. 1=conversion of the corresponding ADC channel is complete 0=conversion of the corresponding ADC channel is not complete For enabled single cycles, the SINGLE_DONE_STATUS bit in the ADC Control Register is also set after all enabled channel conversion are done; for enabled repeat cycles, the REPEAT_DONE_STATUS in the ADC Control Register is also set after all enabled channel conversion are done.	R/WC	00h	RESET_SYS

21.11.4 ADC SINGLE REGISTER

The ADC Single Register is used to control which ADC channel is captured during a Single-Sample conversion cycle initiated by the START_SINGLE bit in the ADC Control Register.

Note: Do not change the bits in this register in the middle of a conversion cycle to insure proper operation.

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	Reserved	R	-	-
15:0	SINGLE_EN Each bit in this field enables the corresponding ADC channel when a single cycle of conversions is started when the START_SINGLE bit in the ADC Control Register is written with a 1. 1=single cycle conversions for this channel are enabled 0=single cycle conversions for this channel are disabled	R/W	0h	RESET_SYS

21.11.5 ADC REPEAT REGISTER

The ADC Repeat Register is used to control which ADC channels are captured during a repeat conversion cycle initiated by the START_REPEAT bit in the ADC Control Register.

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	Reserved	R	-	-
15:0	RPT_EN Each bit in this field enables the corresponding ADC channel for each pass of the Repeated ADC Conversion that is controlled by bit START_REPEAT in the ADC Control Register. 1=repeat conversions for this channel are enabled 0=repeat conversions for this channel are disabled	R/W	00h	RESET_SYS

21.11.6 ADC CHANNEL READING REGISTERS

All 16 ADC channels return their results into a 32-bit reading register. In each case the low 10 bits of the reading register return the result of the Analog to Digital conversion and the upper 22 bits return 0. Table 21-6, "Register Summary" shows the addresses of all the reading registers.

Note:	The ADC Channel Reading Registers access require single 16, or 32 bit reads; i.e., two 8 bit reads will not provide data coherency.

22.0 RPM-PWM INTERFACE

22.1 Introduction

The RPM-PWM Interface is a closed-loop RPM based Fan Control Algorithm that monitors a fan's speed and automatically adjusts the drive to the fan in order to maintain the desired fan speed.

The RPM-PWM Interface functionality consists of a closed-loop "set-and-forget" RPM-based fan controller.

22.2 References

No references have been cited for this chapter

22.3 Terminology

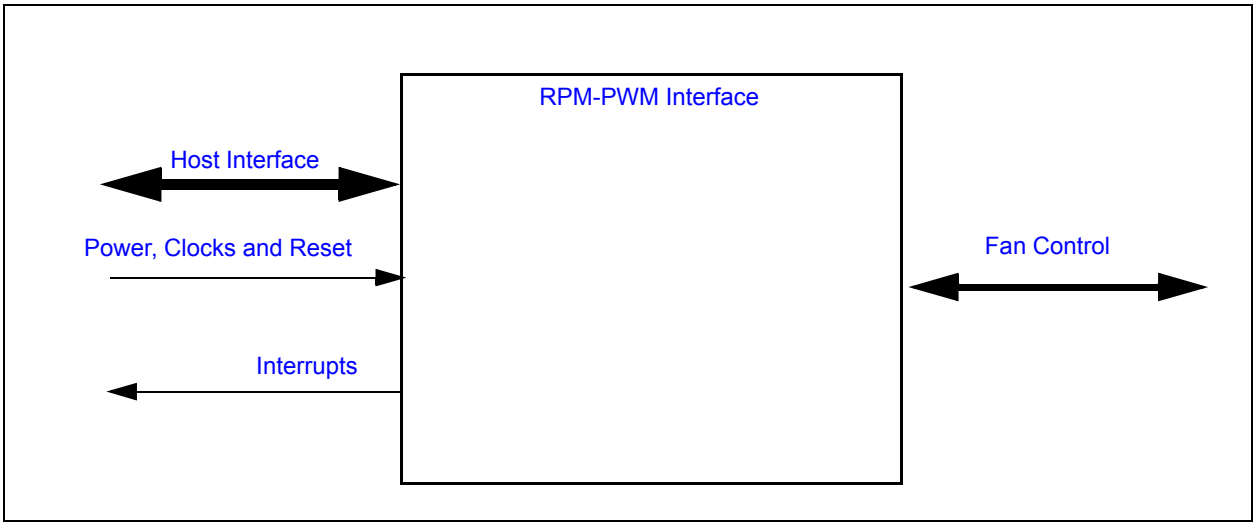
There is no terminology defined for this chapter.

22.4 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

The registers in the block are accessed by embedded controller code at the addresses shown in Section 22.9, "EC Registers".

FIGURE 22-1: RPM-PWM INTERFACE I/O DIAGRAM



22.4.1 FAN CONTROL

The Fan Control Signal Description Table lists the signals that are routed to/from the block.

Name	Direction	Description
GTACH	Input	Tachometer input from fan
GPWM	Output	PWM fan drive output

22.4.2 HOST INTERFACE

The registers defined for the RPM-PWM Interface are accessible by the various hosts as indicated in Section 22.9, "EC Registers".

22.5 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

22.5.1 POWER DOMAINS

Name	Description
VTR	This power well sources the registers and logic in this block.

22.5.2 CLOCK INPUTS

Name	Description
48MHz	This clock signal drives selected logic (e.g., counters).
32KHz	This clock signal drives selected logic (e.g., counters).

22.5.3 RESETS

Name	Description
RESET_SYS	This reset signal resets all of the registers and logic in this block.

22.6 Interrupts

This section defines the Interrupt Sources generated from this block.

Source	Description
FAN_FAIL	The DRIVE_FAIL & FAN_SPIN bits in the Fan Status Register are logically ORed and routed to the FAIL_SPIN Interrupt
FAN_STALL	The FAN_STALL bit in the Fan Status Register is routed to the FAN_STALL Interrupt

22.7 Low Power Modes

The RPM-PWM Interface may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry.

22.8 Description

This section defines the functionality of the block.

22.8.1 GENERAL OPERATION

The RPM-PWM Interface is an RPM based Fan Control Algorithm that monitors the fan's speed and automatically adjusts the drive to maintain the desired fan speed. This RPM based Fan Control Algorithm controls a PWM output based on a tachometer input.

22.8.2 FAN CONTROL MODES OF OPERATION

The RPM-PWM Interface has two modes of operation for the PWM Fan Driver. They are:

1. Manual Mode - in this mode of operation, the user directly controls the fan drive setting. Updating the Fan Driver Setting Register (see Section 22.9.1, "Fan Setting Register") will update the fan drive based on the programmed ramp rate (default disabled).
   - The Manual Mode is enabled by clearing the EN_ALGO bit in the Fan Configuration Register (see Section 22.9.2, "Fan Configuration Register").
   - Whenever the Manual Mode is enabled the current drive settings will be changed to what was last used by the RPM control algorithm.
   - Setting the drive value to 00h will disable the PWM Fan Driver.
   - Changing the drive value from 00h will invoke the Spin Up Routine.
2. Using RPM based Fan Control Algorithm - in this mode of operation, the user determines a target tachometer reading and the drive setting is automatically updated to achieve this target speed.

Manual Mode	Algorithm
Fan Driver Setting (read / write)	Fan Driver Setting (read only)
EDGES[1:0] (Fan Configuration)	EDGES[1:0] (Fan Configuration)
UPDATE[2:0] (Fan configuration)	UPDATE[2:0] (Fan configuration)
LEVEL (Spin Up Configuration)	LEVEL (Spin Up Configuration)
SPINUP_TIME[1:0] (Spin Up Configuration)	SPINUP_TIME[1:0] (Spin Up Configuration)
Fan Step	Fan Step
-	Fan Minimum Drive
Valid TACH Count	Valid TACH Count
-	TACH Target
TACH Reading	TACH Reading
RANGE[2:0] (Fan Configuration 2)	RANGE[2:0] (Fan Configuration 2)
-	DRIVE_FAIL_CNT[2:0] (Spin Up Config) and Drive Fail Band

22.8.3 RPM BASED FAN CONTROL ALGORITHM

The RPM-PWM Interface includes an RPM based Fan Control Algorithm.

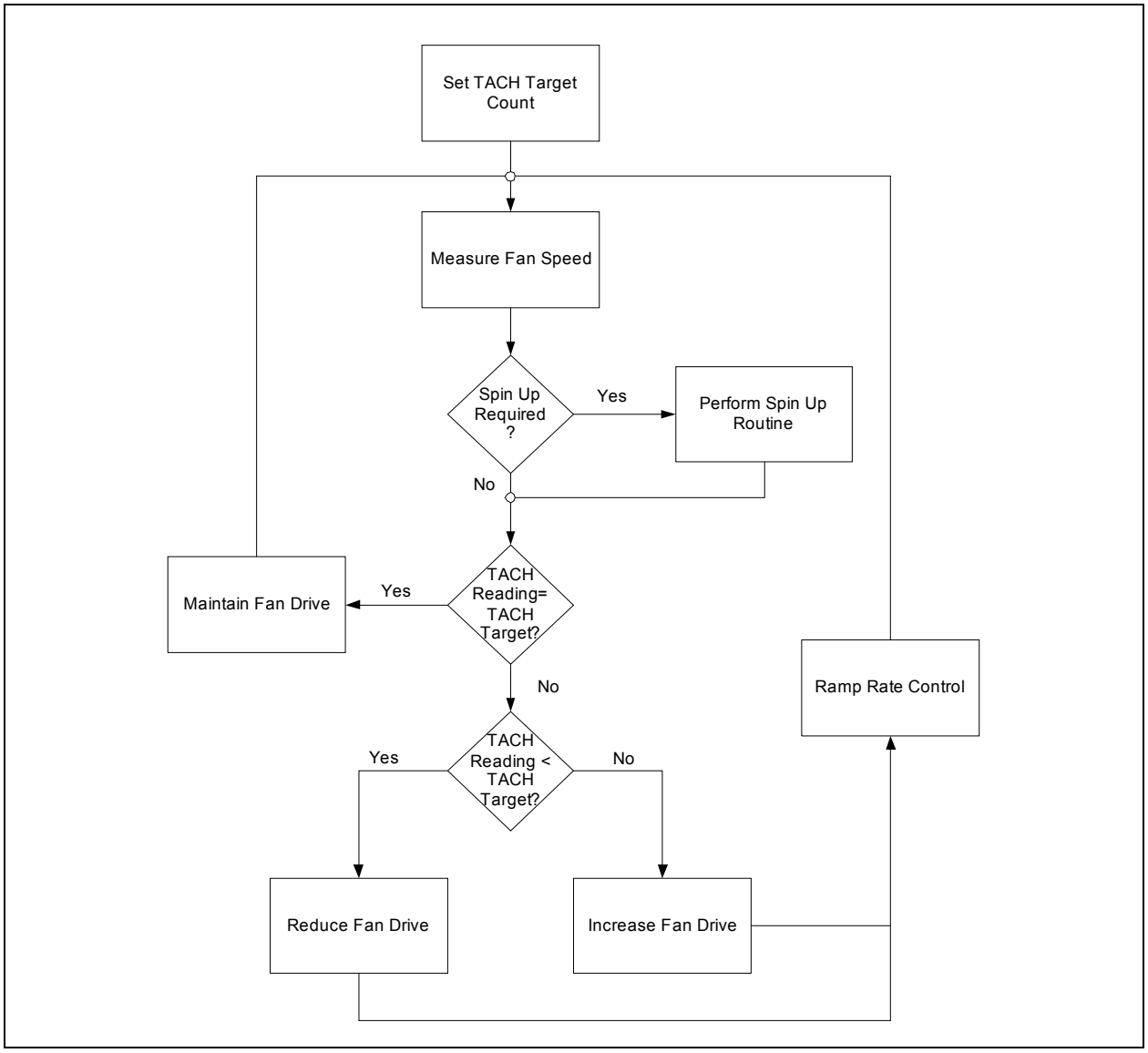
The fan control algorithm uses Proportional, Integral, and Derivative terms to automatically approach and maintain the system's desired fan speed to an accuracy directly proportional to the accuracy of the clock source. Figure 22-2, "RPM based Fan Control Algorithm" shows a simple flow diagram of the RPM based Fan Control Algorithm operation.

The desired tachometer count is set by the user inputting the desired number of 32.768KHz cycles that occur per fan revolution. The user may change the target count at any time. The user may also set the target count to FFh in order to disable the fan driver.

For example, if a desired RPM rate for a 2-pole fan is 3000 RPMs, the user would input the hexadecimal equivalent of 1312d (52_00h in the TACH Target Registers). This number represents the number of 32.768KHz cycles that would occur during the time it takes the fan to complete a single revolution when it is spinning at 3000RPMs (see Section 22.9.10, "TACH Target Register" and Section 22.9.11, "TACH Reading Register").

The RPM-PWM Interface's RPM based Fan Control Algorithm has programmable configuration settings for parameters such as ramp-rate control and spin up conditions. The fan driver automatically detects and attempts to alleviate a stalled/stuck fan condition while also asserting the interrupt signal. The RPM-PWM Interface works with fans that operate up to 16,000 RPMs and provide a valid tachometer signal.

FIGURE 22-2: RPM BASED FAN CONTROL ALGORITHM



22.8.3.1 Programming the RPM Based Fan Control Algorithm

The RPM based Fan Control Algorithm powers-up disabled. The following registers control the algorithm. The RPM-PWM Interface fan control registers are pre-loaded with defaults that will work for a wide variety of fans so only the TACH Target Register is required to set a fan speed. The other fan control registers can be used to fine-tune the algorithm behavior based on application requirements.

1. Set the Valid TACH Count Register to the minimum tachometer count that indicates the fan is spinning.
2. Set the Spin Up Configuration Register to the spin up level and Spin Time desired.
3. Set the Fan Step Register to the desired step size.
4. Set the Fan Minimum Drive Register to the minimum drive value that will maintain fan operation.
5. Set the Update Time, and Edges options in the Fan Configuration Register.
6. Set the TACH Target Register to the desired tachometer count.
7. Enable the RPM based Fan Control Algorithm by setting the EN_ALGO bit.

22.8.3.2 Tachometer Measurement

In both modes of operation, the tachometer measurement operates independently of the mode of operation of the fan driver and RPM based Fan Speed Control algorithm. Any tachometer reading that is higher than the Valid TACH Count (see Section 22.9.8, "Valid TACH Count Register") will flag a stalled fan and trigger an interrupt.

When measuring the tachometer, the fan must provide a valid tachometer signal at all times to ensure proper operation. The tachometer measurement circuitry is programmable to detect the fan speed of a variety of fan configurations and architectures including 1-pole, 2-pole (default), 3-pole, and 4-pole fans.

Note:	The tachometer measurement works independently of the drive settings. If the device is put into manual mode and the fan drive is set at a level that is lower than the fan can operate (including zero drive), the tachometer measurement may signal a Stalled Fan condition and assert an interrupt.

STALLED FAN

If the TACH Reading Register exceeds the user-programmable Valid TACH Count setting, it will flag the fan as stalled and trigger an interrupt. If the RPM based Fan Control Algorithm is enabled, the algorithm will automatically attempt to restart the fan until it detects a valid tachometer level or is disabled.

The FAN_STALL Status bit indicates that a stalled fan was detected. This bit is checked conditionally depending on the mode of operation.

- Whenever the Manual Mode is enabled or whenever the drive value is changed from 00h, the FAN_STALL interrupt will be masked for the duration of the programmed Spin Up Time (see Section 22.9.5, "Fan Spin Up Configuration Register") to allow the fan an opportunity to reach a valid speed without generating unnecessary interrupts.
- In Manual Mode, whenever the TACH Reading Register exceeds the Valid TACH Count Register setting, the FAN_STALL status bit will be set.
- When the RPM based Fan Control Algorithm, the stalled fan condition is checked whenever the Update Time is met and the fan drive setting is updated. It is not a continuous check.

22.8.3.3 Spin Up Routine

The RPM-PWM Interface also contains programmable circuitry to control the spin up behavior of the fan driver to ensure proper fan operation. The Spin Up Routine is initiated under the following conditions:

- The TACH Target High Byte Register value changes from a value of FFh to a value that is less than the Valid TACH Count (see Section 22.9.8, "Valid TACH Count Register").
- The RPM based Fan Control Algorithm's measured tachometer reading is greater than the Valid TACH Count.
- When in Manual Mode, the Drive Setting changes from a value of 00h.

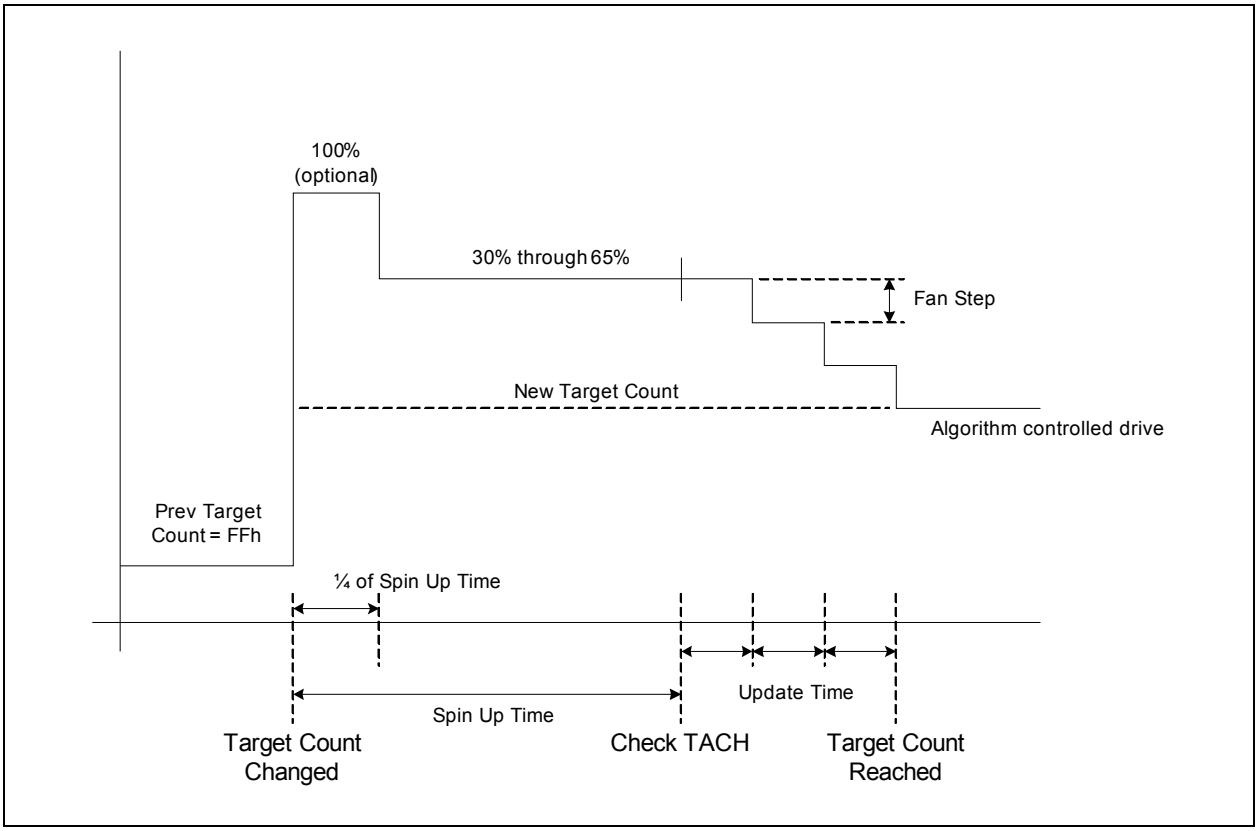
When the Spin Up Routine is operating, the fan driver is set to full scale for one quarter of the total user defined spin up time. For the remaining spin up time, the fan driver output is set a a user defined level (30% to 65% drive).

After the Spin Up Routine has finished, the RPM-PWM Interface measures the tachometer. If the measured tachometer reading is higher than the Valid TACH Count Register setting, the FAN_SPIN status bit is set and the Spin Up Routine will automatically attempt to restart the fan.

Note:	When the device is operating in manual mode, the FAN_SPIN status bit may be set if the fan drive is set at a level that is lower than the fan can operate (excluding zero drive which disables the fan driver). If the FAN_SPIN interrupt is unmasked, this condition will trigger an errant interrupt.

Figure 22-3, "Spin Up Routine" shows an example of the Spin Up Routine in response to a programmed fan speed change based on the first condition above.

FIGURE 22-3: SPIN UP ROUTINE



22.8.4 PWM DRIVER

The RPM-PWM Interface contains an optional, programmable 10-bit PWM driver which can serve as part of the RPM based Fan Speed Control Algorithm or in Manual Mode.

When enabled, the PWM driver can operate in four programmable frequency bands. The lower frequency bands offer frequencies in the range of 9.5Hz to 4.8kHz while the higher frequency options offer frequencies of 21Hz or 25.2kHz.

The highest frequency available, 25.2KHz, operates in 8-bit resolution. All other PWM frequencies operate in 10-bit resolution.

22.8.5 FAN SETTING

The Fan Setting Registers are used to control the output of the Fan Driver. The driver setting operates independently of the Polarity bit for the PWM output. That is, a setting of 0000h will mean that the fan drive is at minimum drive while a value of FFC0h will mean that the fan drive is at maximum drive.

If the Spin Up Routine is invoked, reading from the registers will return the current fan drive setting that is being used by the Spin Up Routine instead of what was previously written into these registers.

The Fan Driver Setting Registers, when the RPM based Fan Control Algorithm is enabled, are read only. Writing to the register will have no effect and the data will not be stored. Reading from the register will always return the current fan drive setting.

If the INT_PWRGD pin is de-asserted, the Fan Driver Setting Register will be made read only. Writing to the register will have no effect and reading from the register will return 0000h.

When the RPM based Fan Control Algorithm is disabled, the current fan drive setting that was last used by the algorithm is retained and will be used.

If the Fan Driver Setting Register is set to a value of 0000h, all tachometer related status bits will be masked until the setting is changed. Likewise, the FAN_SHORT bit will be cleared and masked until the setting is changed.

The contents of the register represent the weighting of each bit in determining the final duty cycle. The output drive for a PWM output is given by the following equation:

- Drive = (FAN_SETTING VALUE/1023) x 100%.

The PWM Divide Register determines the final PWM frequency. The base frequency set by the PWM_BASE[1:0] bits is divided by the decimal equivalent of the register settings.

The final PWM frequency is derived as the base frequency divided by the value of this register as shown in the equation below:

- PWM_Frequency = base_clk / PWM_D

Where:

- base_clk = The base frequency set by the PWMx_CFG[1:0] bits
- PWM_D = the divide setting set by the PWM Divide Register.

22.8.6 ALERTS AND LIMITS

Figure 22-4, "Interrupt Flow" shows the interactions of the interrupts for fan events.

If the Fan Driver detects a drive fail, spin-up or stall event, the interrupt signal will be asserted (if enabled).

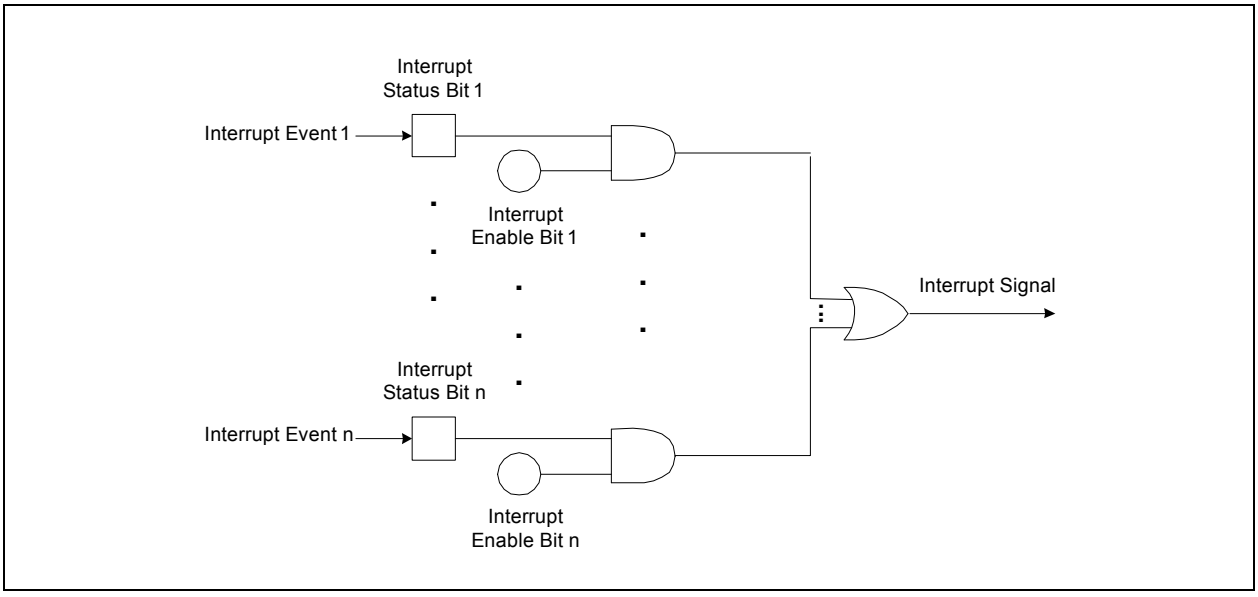
All of these interrupts can be masked from asserting the interrupt signal individually. If any bit of either Status register is set, the interrupt signal will be asserted provided that the corresponding interrupt enable bit is set accordingly.

The Status register will be updated due to an active event, regardless of the setting of the individual enable bits. Once a status bit has been set, it will remain set until the Status register bit is written to 1 (and the error condition has been removed).

If the interrupt signal is asserted, it will be cleared immediately if either the status or enable bit is cleared.

See Section 22.6, "Interrupts".

FIGURE 22-4: INTERRUPT FLOW



22.9 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the RPM-PWM Interface Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 22-1: REGISTER SUMMARY

Offset	Register Name
00h	Fan Setting
02h	Fan Configuration Register
04h	PWM Divide Register
05h	Gain Register
06h	Fan Spin Up Configuration Register
07h	Fan Step Register
08h	Fan Minimum Drive Register
09h	Valid TACH Count Register
0Ah	Fan Drive Fail Band Register
0Ch	TACH Target Register
0Eh	TACH Reading Register
10h	PWM Driver Base Frequency Register
11h	Fan Status Register

22.9.1 FAN SETTING REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
15:6	FAN_SETTING The Fan Driver Setting used to control the output of the Fan Driver.	R/W	00h	RESET_SYS
5:0	Reserved	R	-	-

22.9.2 FAN CONFIGURATION REGISTER

Offset	02h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
15	EN_RRC Enables the ramp rate control circuitry during the Manual Mode of operation. 1=The ramp rate control circuitry for the Manual Mode of operation is enabled. The PWM setting will follow the ramp rate controls as determined by the Fan Step and Update Time settings. The maximum PWM step is capped at the Fan Step setting and is updated based on the Update Time as given by the field UPDATE. 0=The ramp rate control circuitry for the Manual Mode of operation is disabled. When the Fan Drive Setting values are changed and the RPM based Fan Control Algorithm is disabled, the fan driver will be set to the new setting immediately.	R/W	0b	RESET_SYS
14	DIS_GLITCH Disables the low pass glitch filter that removes high frequency noise injected on the TACH pin. 1=The glitch filter is disabled 0=The glitch filter is enabled	R/W	0b	RESET_SYS
13:12	DER_OPT Control some of the advanced options that affect the derivative portion of the RPM based fan control algorithm as shown in Table 22-3, "Derivative Options". These bits only apply if the Fan Speed Control Algorithm is used.	R/W	3h	RESET_SYS
11:10	ERR_RNG Control some of the advanced options that affect the error window. When the measured fan speed is within the programmed error window around the target speed, the fan drive setting is not updated. These bits only apply if the Fan Speed Control Algorithm is used. 3=200 RPM 2=100 RPM 1=50 RPM 0=0 RPM	R/W	1h	RESET_SYS
9	POLARITY Determines the polarity of the PWM driver. This does NOT affect the drive setting registers. A setting of 0% drive will still correspond to 0% drive independent of the polarity. 1=The Polarity of the PWM driver is inverted. A drive setting of 00h will cause the output to be set at 100% duty cycle and a drive setting of FFh will cause the output to be set at 0% duty cycle. 0=The Polarity of the PWM driver is normal. A drive setting of 00h will cause the output to be set at 0% duty cycle and a drive setting of FFh will cause the output to be set at 100% duty cycle.	R/W	0h	RESET_SYS

Offset	02h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
8	Reserved	R	-	-
7	EN_ALGO Enables the RPM based Fan Control Algorithm. 1=The control circuitry is enabled and the Fan Driver output will be automatically updated to maintain the programmed fan speed as indicated by the TACH Target Register. 0=The control circuitry is disabled and the fan driver output is determined by the Fan Driver Setting Register.	R/W	0b	RESET_SYS
6:5	RANGE Adjusts the range of reported and programmed tachometer reading values. The RANGE bits determine the weighting of all TACH values (including the Valid TACH Count, TACH Target, and TACH reading). 3=Reported Minimum RPM: 4000. Tach Count Multiplier: 8 2=Reported Minimum RPM: 2000. Tach Count Multiplier: 4 1=Reported Minimum RPM: 1000. Tach Count Multiplier: 2 0=Reported Minimum RPM: 500. Tach Count Multiplier: 1	R/W	1h	RESET_SYS

Offset	02h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
4:3	EDGES Determines the minimum number of edges that must be detected on the TACH signal to determine a single rotation. A typical fan measured 5 edges (for a 2-pole fan). Increasing the number of edges measured with respect to the number of poles of the fan will cause the TACH Reading registers to indicate a fan speed that is higher or lower than the actual speed. In order for the FSC Algorithm to operate correctly, the TACH Target must be updated by the user to accommodate this shift. The Effective Tach Multiplier shown in Table 22-2, "Minimum Edges for Fan Rotation" is used as a direct multiplier term that is applied to the Actual RPM to achieve the Reported RPM. It should only be applied if the number of edges measured does not match the number of edges expected based on the number of poles of the fan (which is fixed for any given fan). Contact Microchip for recommended settings when using fans with more or less than 2 poles.	R/W	1h	RESET_SYS
2:0	UPDATE Determines the base time between fan driver updates. The Update Time, along with the Fan Step Register, is used to control the ramp rate of the drive response to provide a cleaner transition of the actual fan operation as the desired fan speed changes. 7=1600ms 6=1200ms 5=800ms 4=500ms 3=400ms 2=300ms 1=200ms 0=100ms **Note:** This ramp rate control applies for all changes to the active PWM output including when the RPM based Fan Speed Control Algorithm is disabled.	R/W	3h	RESET_SYS

TABLE 22-2: MINIMUM EDGES FOR FAN ROTATION

Edges	Minimum TACH Edges	Number of Fan Poles	Effective TACH Multiplier (Based on 2 Pole Fans) If Edges Changed
0h	3	1	0.5
1h	5	2 (default)	1
2h	7	3	1.5
3h	9	4	2

TABLE 22-3: DERIVATIVE OPTIONS

DER_OPT	Operation	Note (see Section 22.9.6, "Fan Step Register")
0	No derivative options used	PWM steps are limited to the maximum PWM drive step value in Fan Step Register
1	Basic derivative. The derivative of the error from the current drive setting and the target is added to the iterative PWM drive setting (in addition to proportional and integral terms)	PWM steps are limited to the maximum PWM drive step value in Fan Step Register
2	Step derivative. The derivative of the error from the current drive setting and the target is added to the iterative PWM drive setting and is not capped by the maximum PWM drive step. This allows for very fast response times	PWM steps are not limited to the maximum PWM drive step value in Fan Step Register (i.e., maximum fan step setting is ignored)
3	Both the basic derivative and the step derivative are used effectively causing the derivative term to have double the effect of the derivative term (default).	PWM steps are not limited to the maximum PWM drive step value in Fan Step Register (i.e., maximum fan step setting is ignored)

22.9.3 PWM DIVIDE REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	PWM_DIVIDE The PWM Divide value determines the final frequency of the PWM driver. The driver base frequency is divided by the PWM Divide value to determine the final frequency.	R/W	01h	RESET_SYS

22.9.4 GAIN REGISTER

The Gain Register stores the gain terms used by the proportional and integral portions of the RPM based Fan Control Algorithm. These terms will affect the FSC closed loop acquisition, overshoot, and settling as would be expected in a classic PID system.

This register only applies if the Fan Speed Control Algorithm is used.

Offset	05h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:6	Reserved	R	-	-
5:4	GAIND The derivative gain term. Gain Factor: 3=8x 2=4x 1=2x 0=1x	R/W	2h	RESET_SYS

Offset	05h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
3:2	GAINI The integral gain term. Gain Factor: 3=8x 2=4x 1=2x 0=1x	R/W	2h	RESET_SYS
1:0	GAINP The proportional gain term. Gain Factor: 3=8x 2=4x 1=2x 0=1x	R/W	2h	RESET_SYS

22.9.5 FAN SPIN UP CONFIGURATION REGISTER

Offset	06h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:6	DRIVE_FAIL_CNT Determines how many update cycles are used for the Drive Fail detection function. This circuitry determines whether the fan can be driven to the desired Tach target. These settings only apply if the Fan Speed Control Algorithm is enabled. 3=Drive Fail detection circuitry will count for 64 update periods 2=Drive Fail detection circuitry will count for 32 update periods 1=Drive Fail detection circuitry will count for 16 update periods 0=Drive Fail detection circuitry is disabled	R/W	00b	RESET_SYS
5	NOKICK Determines if the Spin Up Routine will drive the fan to 100% duty cycle for 1/4 of the programmed spin up time before driving it at the programmed level. 1=The Spin Up Routine will not drive the PWM to 100%. It will set the drive at the programmed spin level for the entire duration of the programmed spin up time 0=The Spin Up Routine will drive the PWM to 100% for 1/4 of the programmed spin up time before reverting to the programmed spin level	R/W	0b	RESET_SYS

Offset	06h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
4:2	SPIN_LVL Determines the final drive level that is used by the Spin Up Routine. 7=65% 6=60% 5=55% 4=50% 3=45% 2=40% 1=35% 0=30%	R/W	6h	RESET_SYS
1:0	SPINUP_TIME Determines the maximum Spin Time that the Spin Up Routine will run for. If a valid tachometer measurement is not detected before the Spin Time has elapsed, an interrupt will be generated. When the RPM based Fan Control Algorithm is active, the fan driver will attempt to re-start the fan immediately after the end of the last spin up attempt. 3=2 seconds 2=1 second 1=500 ms 0=250 ms	R/W	1h	RESET_SYS

22.9.6 FAN STEP REGISTER

The Fan Step Register, along with the Update Time, controls the ramp rate of the fan driver response calculated by the RPM based Fan Control Algorithm for the Derivative Options field values of “00” and “01” in the Fan Configuration Register.

The value of the register represents the maximum step size the fan driver will take for each update.

When the maximum step size limitation is applied, if the necessary fan driver delta is larger than the Fan Step, it will be capped at the Fan Step setting and updated every Update Time ms.

The maximum step size is ignored for the Derivative Options field values of “10” and “11”.

Offset	07h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	FAN_STEP The Fan Step value represents the maximum step size the fan driver will take between update times. When the PWM_BASE frequency range field in the PWM Driver Base Frequency Register is set to the value 1, 2 or 3, this 8-bit field is added to the 10-bit PWM duty cycle, for a maximum step size of 25%. When the PWM_BASE field is set to 0, the PWM operates in an 8-bit mode. In 8-bit mode, this 8-bit field is added to the 8-bit duty cycle, for a maximum step size of 100%.	R/W	10h	RESET_SYS

22.9.7 FAN MINIMUM DRIVE REGISTER

the Fan Minimum Drive Register stores the minimum drive setting for the RPM based Fan Control Algorithm. The RPM based Fan Control Algorithm will not drive the fan at a level lower than the minimum drive unless the target Fan Speed is set at FFh (see "TACH Target Registers").

During normal operation, if the fan stops for any reason (including low drive), the RPM based Fan Control Algorithm will attempt to restart the fan. Setting the Fan Minimum Drive Registers to a setting that will maintain fan operation is a useful way to avoid potential fan oscillations as the control circuitry attempts to drive it at a level that cannot support fan operation.

These registers only apply if the Fan Speed Control Algorithm is used.

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	MIN_DRIVE The minimum drive setting.	R/W	66h	RESET_SYS

Note: To ensure proper operation, the Fan Minimum Drive register must be set prior to setting the Tach Target High and Low Byte registers, and then the Tach Target registers can be subsequently updated. At a later time, if the Fan Minimum Drive register is changed to a value higher than current Fan value, the Tach Target registers must also be updated.

22.9.8 VALID TACH COUNT REGISTER

The Valid TACH Count Register stores the maximum TACH Reading Register value to indicate that the fan is spinning properly. The value is referenced at the end of the Spin Up Routine to determine if the fan has started operating and decide if the device needs to retry. See the equation in the TACH Reading Registers section for translating the RPM to a count.

If the TACH Reading Register value exceeds the Valid TACH Count Register (indicating that the Fan RPM is below the threshold set by this count), a stalled fan is detected. In this condition, the algorithm will automatically begin its Spin Up Routine.

Note: The automatic invoking of the Spin Up Routine only applies if the Fan Speed Control Algorithm is used. If the FSC is disabled, then the device will only invoke the Spin Up Routine when the PWM setting changes from 00h.

If a TACH Target setting is set above the Valid TACH Count setting, that setting will be ignored and the algorithm will use the current fan drive setting.

These registers only apply if the Fan Speed Control Algorithm is used.

Offset	09h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	VALID_TACH_CNT The maximum TACH Reading Register value to indicate that the fan is spinning properly.	R/W	F5h	RESET_SYS

22.9.9 FAN DRIVE FAIL BAND REGISTER

The Fan Drive Fail Band Registers store the number of Tach counts used by the Fan Drive Fail detection circuitry. This circuitry is activated when the fan drive setting high byte is at FFh. When it is enabled, the actual measured fan speed is compared against the target fan speed.

This circuitry is used to indicate that the target fan speed at full drive is higher than the fan is actually capable of reaching. If the measured fan speed does not exceed the target fan speed minus the Fan Drive Fail Band Register settings for a period of time longer than set by the DRIVE_FAIL_CNTx[1:0] bits in the Fan Spin Up Configuration Register, the DRIVE_FAIL status bit will be set and an interrupt generated.

These registers only apply if the Fan Speed Control Algorithm is used.

Offset	0Ah			
Bits	**Description**	**Type**	**Default**	**Reset Event**
15:3	FAN_DRIVE_FAIL_BAND The number of Tach counts used by the Fan Drive Fail detection circuitry	R	0h	RESET_SYS
2:0	Reserved	R	-	-

22.9.10 TACH TARGET REGISTER

The TACH Target Registers hold the target tachometer value that is maintained for the RPM based Fan Control Algorithm.

If the algorithm is enabled, setting the TACH Target Register High Byte to FFh will disable the fan driver (or set the PWM duty cycle to 0%). Setting the TACH Target to any other value (from a setting of FFh) will cause the algorithm to invoke the Spin Up Routine after which it will function normally.

These registers only apply if the Fan Speed Control Algorithm is used.

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
15:3	TACH_TARGET The target tachometer value.	R	-	RESET_SYS
2:0	Reserved	R	-	-

22.9.11 TACH READING REGISTER

The TACH Reading Registers' contents describe the current tachometer reading for the fan. By default, the data represents the fan speed as the number of 32.768kHz clock periods that occur for a single revolution of the fan.

The Equation below shows the detailed conversion from tachometer measurement (COUNT) to RPM.

$$RPM = \frac{1}{Poles} \times \frac{(n-1)}{COUNT \times \frac{1}{m}} \times f_{TACH} \times 60$$

where:

- *Poles* = number of poles of the fan (typically 2)
- f_{TACH} = the frequency of the tachometer measurement clock
- *n* = number of edges measured (typically 5 for a 2 pole fan)
- *m* = the multiplier defined by the RANGE bits
- *COUNT* = TACH Reading Register value (in decimal)

The following equation shows the simplified translation of the TACH Reading Register count to RPM assuming a 2-pole fan, measuring 5 edges, with a frequency of 32.768kHz.

$$RPM = \frac{3932160 \times m}{COUNT}$$

Offset	0Eh			
Bits	**Description**	**Type**	**Default**	**Reset Event**
15:3	TACH_READING The current tachometer reading value.	R	-	RESET_SYS
2:0	Reserved	R	-	-

22.9.12 PWM DRIVER BASE FREQUENCY REGISTER

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:2	Reserved	R	-	-
1:0	PWM_BASE Determines the frequency range of the PWM fan driver (when enabled). PWM resolution is 10-bit, except when this field is set to '0b', when it is 8-bit. 3=2.34KHz 2=4.67KHz 1=23.4KHz 0=26.8KHz	R/W	00b	RESET_SYS

22.9.13 FAN STATUS REGISTER

Offset	11h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:6	Reserved	R	-	-
5	DRIVE_FAIL The bit Indicates that the RPM-based Fan Speed Control Algorithm cannot drive the Fan to the desired target setting at maximum drive. 1=The RPM-based Fan Speed Control Algorithm cannot drive Fan to the desired target setting at maximum drive. 0=The RPM-based Fan Speed Control Algorithm can drive Fan to the desired target setting.	R/WC	0b	RESET_SYS
4:2	Reserved	R	-	-
1	FAN_SPIN The bit Indicates that the Spin up Routine for the Fan could not detect a valid tachometer reading within its maximum time window. 1=The Spin up Routine for the Fan could not detect a valid tachometer reading within its maximum time window. 0=The Spin up Routine for the Fan detected a valid tachometer reading within its maximum time window.	R/WC	0b	RESET_SYS
0	FAN_STALL The bit Indicates that the tachometer measurement on the Fan detects a stalled fan. 1=Stalled fan not detected 0=Stalled fan not detected	R/WC	0b	RESET_SYS

23.0 BLINKING/BREATHING PWM

23.1 Introduction

LEDs are used in computer applications to communicate internal state information to a user through a minimal interface. Typical applications will cause an LED to blink at different rates to convey different state information. For example, an LED could be full on, full off, blinking at a rate of once a second, or blinking at a rate of once every four seconds, in order to communicate four different states.

As an alternative to blinking, an LED can “breathe”, that is, oscillate between a bright state and a dim state in a continuous, or apparently continuous manner. The rate of breathing, or the level of brightness at the extremes of the oscillation period, can be used to convey state information to the user that may be more informative, or at least more novel, than traditional blinking.

The blinking/breathing hardware is implemented using a PWM. The PWM can be driven either by the Main system clock or by a 32.768 KHz clock input. When driven by the Main system clock, the PWM can be used as a standard 8-bit PWM in order to control a fan. When used to drive blinking or breathing LEDs, the 32.768 KHz clock source is used.

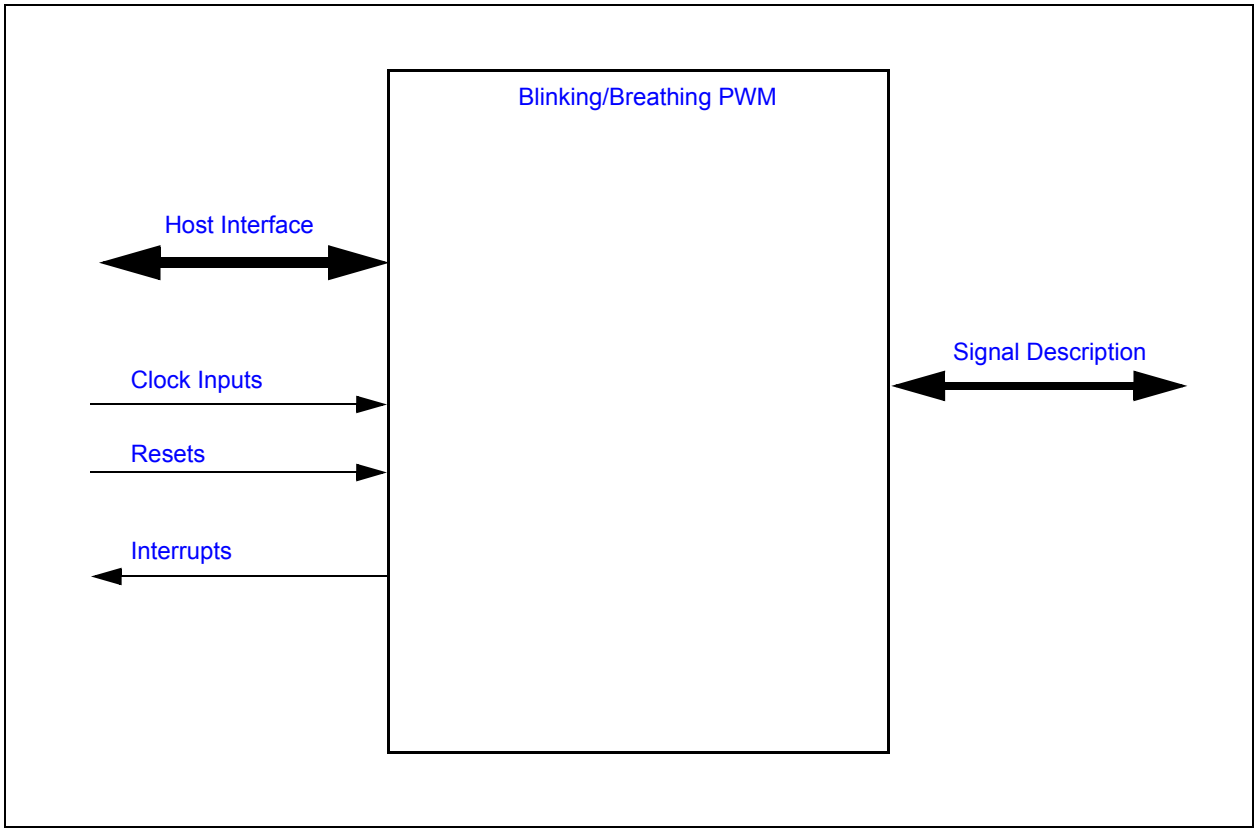
Features:

- Each PWM independently configurable
- Each PWM configurable for LED blinking and breathing output
- Highly configurable breathing rate from 60ms to 1min
- Non-linear brightness curves approximated with 8 piece wise-linear segments
- All LED PWMs can be synchronized
- Each PWM configurable for 8-bit PWM support
- Multiple clock rates
- Configurable Watchdog Timer

23.2 Interface

This block is designed to drive a pin on the pin interface and to be accessed internally via a registered host interface.

FIGURE 23-1: I/O DIAGRAM OF BLOCK



23.3 Signal Description

Name	Direction	Description
PWM Output	Output	Output of PWM By default, the PWM pin is configured to be active high: when the PWM is configured to be fully on, the pin is driving high. When the PWM is configured to be fully off, the pin is low. If firmware requires the Blinking/Breathing PWM to be active low, the Polarity bit in the GPIO Pin Control Register associated with the PWM can be set to 1, which inverts the output polarity.

23.4 Host Interface

The blinking/breathing PWM block is accessed by a controller over the standard register interface.

23.5 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

23.5.1 POWER DOMAINS

Name	Description
VTR	Main power. The source of main power for the device is system dependent.

23.5.2 CLOCK INPUTS

Name	Description
32KHz	32.768 KHz clock
48MHz	Main system clock

23.5.3 RESETS

Name	Description
RESET_SYS	Block reset

23.6 Interrupts

Each PWM can generate an interrupt. The interrupt is asserted for one Main system clock period whenever the PWM WDT times out. The PWM WDT is described in Section 23.8.3.1, "PWM WDT".

Source	Description
PWM_WDT	PWM watchdog time out

23.7 Low Power Mode

The Blinking/Breathing PWM may be put into a low power mode by the chip-level power, clocks, and reset (PCR) circuitry. The low power mode is only applicable when the Blinking/Breathing PWM is operating in the General Purpose PWM mode. When the low speed clock mode is selected, the blinking/breathing function continues to operate, even when the 48MHz is stopped. Low power mode behavior is summarized in the following table:

TABLE 23-1: LOW POWER MODE BEHAVIOR

CLOCK_SOURCE	CONTROL	Mode	Low Power Mode	Description
X	‘00’b	PWM ‘OFF’	Yes	32.768 KHz clock is required.
X	‘01’b	Breathing	Yes	
1	‘10’b	General Purpose PWM	No	Main system clock is required, even when a sleep command to the block is asserted.
0	‘10’b	Blinking	Yes	32.768 KHz clock is required.
X	‘11’b	PWM ‘ON’	Yes	

Note: In order for the CEC1702 to enter its.Heavy Sleep state, the SLEEP_ENABLE input for all Blinking/Breathing PWM instances must be asserted, even if the PWMs are configured to use the low speed clock.

23.8 Description

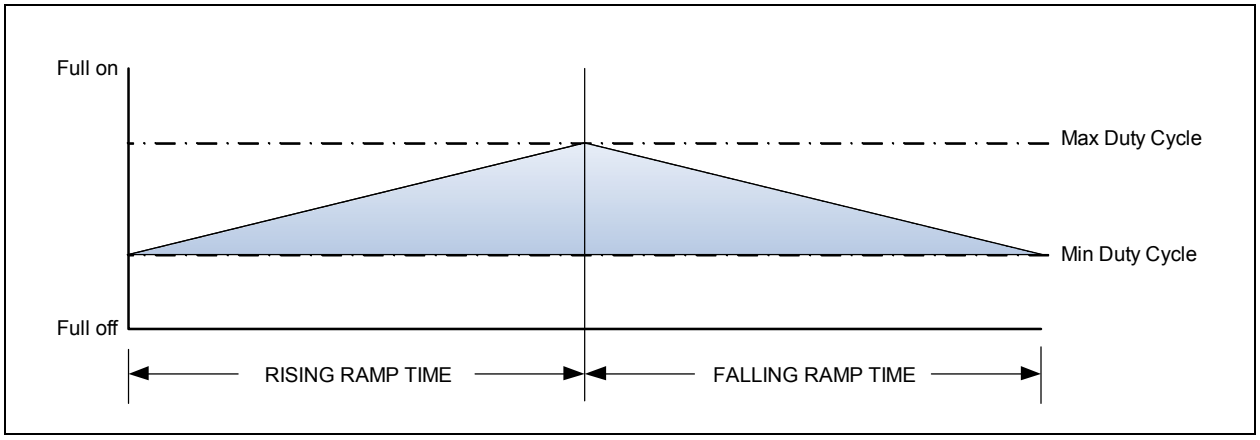
23.8.1 BREATHING

If an LED blinks rapidly enough, the eye will interpret the light as reduced brightness, rather than a blinking pattern. Therefore, if the blinking period is short enough, modifying the duty cycle will set the apparent brightness, rather than a blinking rate. At a blinking rate of 128Hz or greater, almost all people will perceive a continuous light source rather than an intermittent pattern.

Because making an LED appear to breathe is an aesthetic effect, the breathing mechanism must be adjustable or customers may find the breathing effect unattractive. There are several variables that can affect breathing appearance, as described below.

The following figure illustrates some of the variables in breathing:

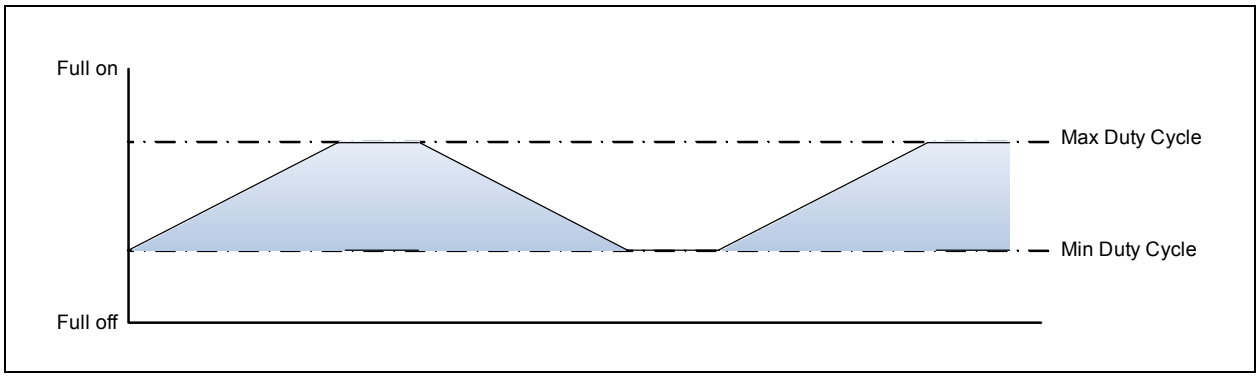
FIGURE 23-2: BREATHING LED EXAMPLE



The breathing range of and LED can range between full on and full off, or in a range that falls within the full-on/full-off range, as shown in this figure. The ramp time can be different in different applications. For example, if the ramp time was 1 second, the LED would appear to breathe quickly. A time of 2 seconds would make the LED appear to breathe more leisurely.

The breathing pattern can be clipped, as shown in the following figure, so that the breathing effect appears to pause at its maximum and minimum brightnesses:

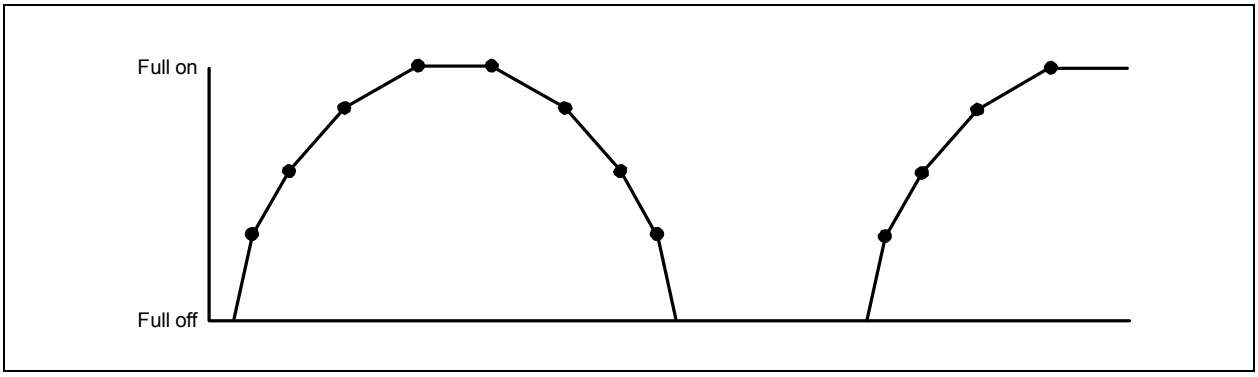
FIGURE 23-3: CLIPPING EXAMPLE



The clipping periods at the two extremes can be adjusted independently, so that for example an LED can appear to breathe (with a short delay at maximum brightness) followed by a longer “resting” period (with a long delay at minimum brightness).

The brightness can also be changed in a non-linear fashion, as shown in the following figure:

FIGURE 23-4: EXAMPLE OF A SEGMENTED CURVE



In this figure, the rise and fall curves are implemented in 4 linear segments and are the rise and fall periods are symmetric.

The breathing mode uses the 32.768 KHz clock for its time base.

23.8.2 BLINKING

When configured for blinking, a subset of the hardware used in breathing is used to implement the blinking function. The PWM (an 8-bit accumulator plus an 8-bit duty cycle register) drives the LED directly. The Duty Cycle register is programmed directly by the user, and not modified further. The PWM accumulator is configured as a simple 8-bit up counter. The counter uses the 32.768 KHz clock, and is pre-scaled by the Delay counter, to slow the PWM down from the 128Hz provided by directly running the PWM on the 32.768 KHz clock.

With the pre-scaler, the blink rate of the LED could be as fast as 128Hz (which, because it is blinking faster than the eye can distinguish, would appear as a continuous level) to 0.03125Hz (that is, with a period of 7.8ms to 32 seconds). Any duty cycle from 0% (0h) to 100% (FFh) can be configured, with an 8-bit precision. An LED with a duty cycle value of 0h will be fully off, while an LED with a duty cycle value of FFh will be fully on.

In Blinking mode the PWM counter is always in 8-bit mode.

Table 23-2, "LED Blink Configuration Examples" shows some example blinking configurations:

TABLE 23-2: LED BLINK CONFIGURATION EXAMPLES

Prescale	Duty Cycle	Blink Frequency	Blink
000h	00h	128Hz	full off
000h	FFh	128Hz	full on
001h	40h	64Hz	3.9ms on, 11.5ms off
003h	80h	32Hz	15.5ms on, 15.5ms off
07Fh	20h	1Hz	125ms on, 0.875s off
0BFh	16h	0.66Hz	125ms on, 1.375s off
0FFh	10h	0.5Hz	125ms on, 1.875s off
180h	0Bh	0.33Hz	129ms on, 2.875s off
1FFh	40h	0.25Hz	1s on, 3s off

The Blinking and General Purpose PWM modes share the hardware used in the breathing mode. The Prescale value is derived from the LD field of the LED_DELAY register and the Duty Cycle is derived from the MIN field of the LED_LIMITS register.

TABLE 23-3: BLINKING MODE CALCULATIONS

Parameter	Unit	Equation
Frequency	Hz	(32KHz frequency) /(PRESCALE + 1)/256
'H' Width	Seconds	(1/Frequency) x (DutyCycle/256)
'L' Width	Seconds	(1/Frequency) x ((1-DutyCycle)/256)

23.8.3 GENERAL PURPOSE PWM

When used in the Blinking configuration with the 48MHz, the LED module can be used as a general-purpose programmable Pulse-Width Modulator with an 8-bit programmable pulse width. It can be used for fan speed control, sound volume, etc. With the 48MHz source, the PWM frequency can be configured in the range shown in Table 23-4.

TABLE 23-4: PWM CONFIGURATION EXAMPLES

Prescale	PWM Frequency
000h	187.5 KHz
001h	94 KHz
003h	47 KHz
006h	26.8 KHz
00Bh	15.625 KHz
07Fh	1.46 KHz
1FFh	366 Hz
FFFh	46 Hz

TABLE 23-5: GENERAL PURPOSE PWM MODE CALCULATIONS

Parameter	Unit	Equation
Frequency	Hz	(48MHz frequency) / (PRESCALE + 1) / 256
'H' Width	Seconds	(1/Frequency) x (DutyCycle/256)
'L' Width	Seconds	(1/Frequency) x (256 - DutyCycle)

23.8.3.1 PWM WDT

When the PWM is configured as a general-purpose PWM (in the Blinking configuration with the Main system clock), the PWM includes a Watch Dog Timer (WDT). The WDT consists of an internal 8-bit counter and an 8-bit reload value (the field WDTLD in LED Configuration Register register). The internal counter is loaded with the reset value of WDTLD (14h, or 4 seconds) on system RESET_SYS and loaded with the contents of WDTLD whenever either the LED Configuration Register register is written or the MIN byte in the LED Limits Register register is written (the MIN byte controls the duty cycle of the PWM).

Whenever the internal counter is non-zero, it is decremented by 1 for every tick of the 5 Hz clock. If the counter decrements from 1 to 0, a WDT Terminal Count causes an interrupt to be generated and reset sets the CONTROL bit in the LED Configuration Register to 3h, which forces the PWM to be full on. No other PWM registers or fields are affected.

If the 5 Hz clock halts, the watchdog timer stops decrementing but retains its value, provided the device continues to be powered. When the 5 Hz clock restarts, the watchdog counter will continue decrementing where it left off.

Setting the WDTLD bits to 0 disables the PWM WDT. Other sample values for WDTLD are:

01h = 200 ms

02h = 400 ms

03h = 600 ms

04h = 800 ms

...

14h = 4seconds

FFh = 51 seconds

23.9 Implementation

In addition to the registers described in Section 23.10, "EC Registers", the PWM is implemented using a number of components that are interconnected differently when configured for breathing operation and when configured for blinking/PWM operation.

23.9.1 BREATHING CONFIGURATION

The **PSIZE** parameter can configure the PWM to one of three modes: 8-bit, 7-bit and 6-bit. The **PERIOD CTR** counts ticks of its input clock. In 8-bit mode, it counts from 0 to 255 (that is, 256 steps), then repeats continuously. In this mode, a full cycle takes 7.8ms (128Hz). In 7-bit mode it counts from 0 to 127 (128 steps), and a full cycle takes 3.9ms (256Hz). In 6-bit mode it counts from 0 to 63 (64 steps) and a full cycle takes 1.95ms (512Hz).

The output of the LED circuit is asserted whenever the **PERIOD CTR** is less than the contents of the **DUTY CYCLE** register. The appearance of breathing is created by modifying the contents of the **DUTY CYCLE** register in a continuous manner. When the LED control is off the internal counters and registers are all reset to 0 (i.e. after a write setting the RESET bit in the LED Configuration Register Register.) Once enabled, the **DUTY CYCLE** register is increased by an amount determined by the LED_STEP register and at a rate determined by the **DELAY** counter. Once the duty cycle reaches its maximum value (determined by the field MAX), the duty cycle is held constant for a period determined by the field HD. Once the hold time is complete, the **DUTY CYCLE** register is decreased, again by an amount determined by the LED_STEP register and at a rate determined by the **DELAY** counter. When the duty cycle then falls at or below the minimum value (determined by the field MIN), the duty cycle is held constant for a period determined by the field HD. Once the hold time is complete, the cycle repeats, with the duty cycle oscillating between MIN and MAX.

The rising and falling ramp times as shown in Figure 23-2, "Breathing LED Example" can be either symmetric or asymmetric depending on the setting of the SYMMETRY bit in the LED Configuration Register Register. In Symmetric mode the rising and falling ramp rates have mirror symmetry; both rising and falling ramp rates use the same (all) 8 segments fields in each of the following registers (see Table 23-6): the LED Update Stepsize Register register and the LED Update Interval Register register. In Asymmetric mode the rising ramp rate uses 4 of the 8 segments fields and the falling ramp rate uses the remaining 4 of the 8 segments fields (see Table 23-6).

The parameters MIN, MAX, HD, LD and the 8 fields in LED_STEP and LED_INT determine the brightness range of the LED and the rate at which its brightness changes. See the descriptions of the fields in Section 23.10, "EC Registers", as well as the examples in Section 23.9.3, "Breathing Examples" for information on how to set these fields.

TABLE 23-6: SYMMETRIC BREATHING MODE REGISTER USAGE

Rising/ Falling Ramp Times in Figure 23-3, "Clipping Example"	Duty Cycle	Segment Index	Symmetric Mode Register Fields Utilized	
X	000xxxxxb	000b	STEP[0]/INT[0]	Bits[3:0]
X	001xxxxxb	001b	STEP[1]/INT[1]	Bits[7:4]
X	010xxxxxb	010b	STEP[2]/INT[2]	Bits[11:8]
X	011xxxxxb	011b	STEP[3]/INT[3]	Bits[15:12]
X	100xxxxxb	100b	STEP[4]/INT[4]	Bits[19:16]
X	101xxxxxb	101b	STEP[5]/INT[5]	Bits[23:20]
X	110xxxxxb	110b	STEP[6]/INT[6]	Bits[27:24]
X	111xxxxxb	111b	STEP[7]/INT[7]	Bits[31:28]
Note: In Symmetric Mode the Segment_Index[2:0] = Duty Cycle Bits[7:5]				

TABLE 23-7: ASYMMETRIC BREATHING MODE REGISTER USAGE

Rising/ Falling Ramp Times in Figure 23-3, "Clipping Example"	Duty Cycle	Segment Index	Asymmetric Mode Register Fields Utilized	
Rising	00xxxxxxb	000b	STEP[0]/INT[0]	Bits[3:0]
Rising	01xxxxxxb	001b	STEP[1]/INT[1]	Bits[7:4]
Rising	10xxxxxxb	010b	STEP[2]/INT[2]	Bits[11:8]

TABLE 23-7: ASYMMETRIC BREATHING MODE REGISTER USAGE (CONTINUED)

Rising/ Falling Ramp Times in Figure 23-3, "Clipping Example"	Duty Cycle	Segment Index	Asymmetric Mode Register Fields Utilized	
Rising	11xxxxxxb	011b	STEP[3]/INT[3]	Bits[15:12]
falling	00xxxxxxb	100b	STEP[4]/INT[4]	Bits[19:16]
falling	01xxxxxxb	101b	STEP[5]/INT[5]	Bits[23:20]
falling	10xxxxxxb	110b	STEP[6]/INT[6]	Bits[27:24]
falling	11xxxxxxb	111b	STEP[7]/INT[7]	Bits[31:28]
Note: In Asymmetric Mode the Segment_Index[2:0] is the bit concatenation of following: Segment_Index[2] = (FALLING RAMP TIME in Figure 23-3, "Clipping Example") and Segment_Index[1:0] = Duty Cycle Bits[7:6].				

23.9.2 BLINKING CONFIGURATION

The Delay counter and the PWM counter are the same as in the breathing configuration, except in this configuration they are connected differently. The Delay counter is clocked on either the 32.768 KHz clock or the Main system clock, rather than the output of the PWM. The PWM counter is clocked by the zero output of the Delay counter, which functions as a prescalar for the input clocks to the PWM. The Delay counter is reloaded from the LD field of the LED_DELAY register. When the LD field is 0 the input clock is passed directly to the PWM counter without prescaling. In Blinking/PWM mode the PWM counter is always 8-bit, and the PSIZE parameter has no effect.

The frequency of the PWM pulse waveform is determined by the formula:

$$f_{PWM} = \frac{f_{clock}}{(256 \times (LD + 1))}$$

where f_{PWM} is the frequency of the PWM, f_{clock} is the frequency of the input clock (32.768 KHz clock or Main system clock) and LD is the contents of the LD field.

Note:	At a duty cycle value of 00h (in the MIN register), the LED output is fully off. At a duty cycle value of 255h, the LED output is fully on. Alternatively, In order to force the LED to be fully on, firmware can set the CONTROL field of the Configuration register to 3 (always on).

The other registers in the block do not affect the PWM or the LED output in Blinking/PWM mode.

23.9.3 BREATHING EXAMPLES

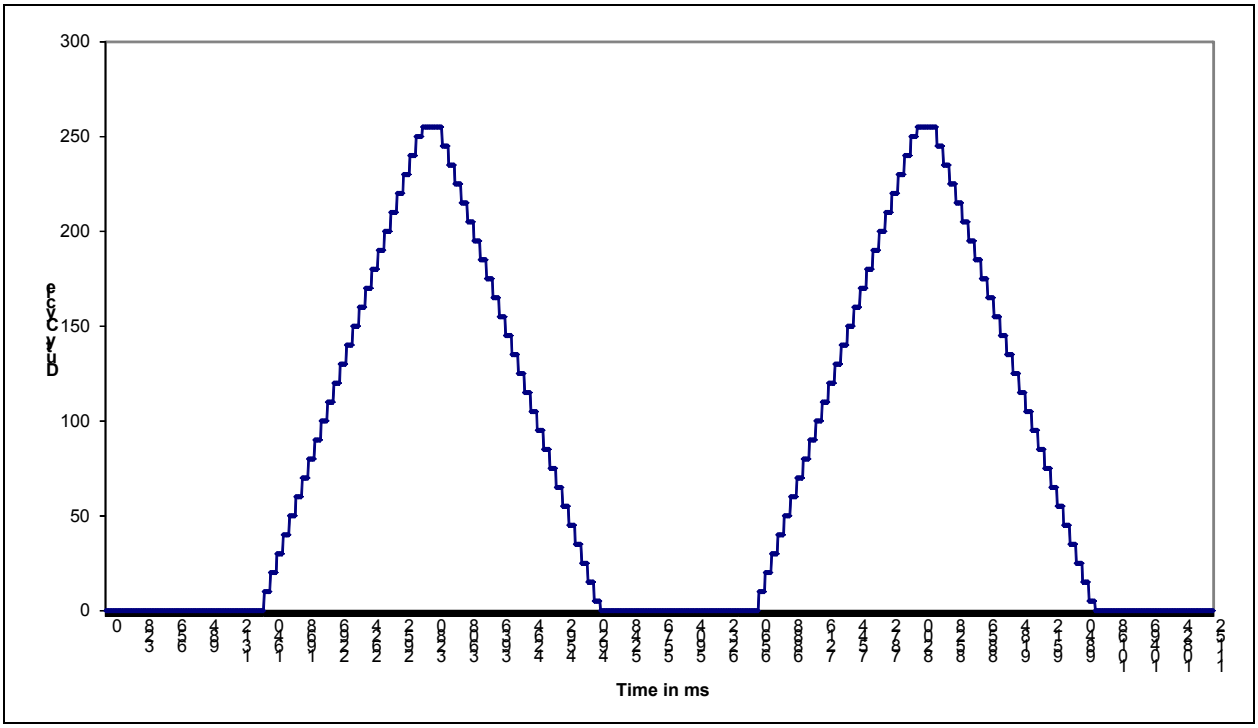
23.9.3.1 Linear LED brightness change

In this example, the brightness of the LED increases and diminishes in a linear fashion. The entire cycle takes 5 seconds. The rise time and fall time are 1.6 seconds, with a hold time at maximum brightness of 200ms and a hold time at minimum brightness of 1.6 seconds. The LED brightness varies between full off and full on. The PWM size is set to 8-bit, so the time unit for adjusting the PWM is approximately 8ms. The registers are configured as follows:

TABLE 23-8: LINEAR EXAMPLE CONFIGURATION

Field	Value							
PSIZE	8-bit							
MAX	255							
MIN	0							
HD	25 ticks (200ms)							
LD	200 ticks (1.6s)							
Duty cycle most significant bits	000b	001b	010b	011b	100b	101b	110b	1110
LED_INT	8	8	8	8	8	8	8	8
LED_STEP	10	10	10	10	10	10	10	10

FIGURE 23-5: LINEAR BRIGHTNESS CURVE EXAMPLE



23.9.3.2 Non-linear LED brightness change

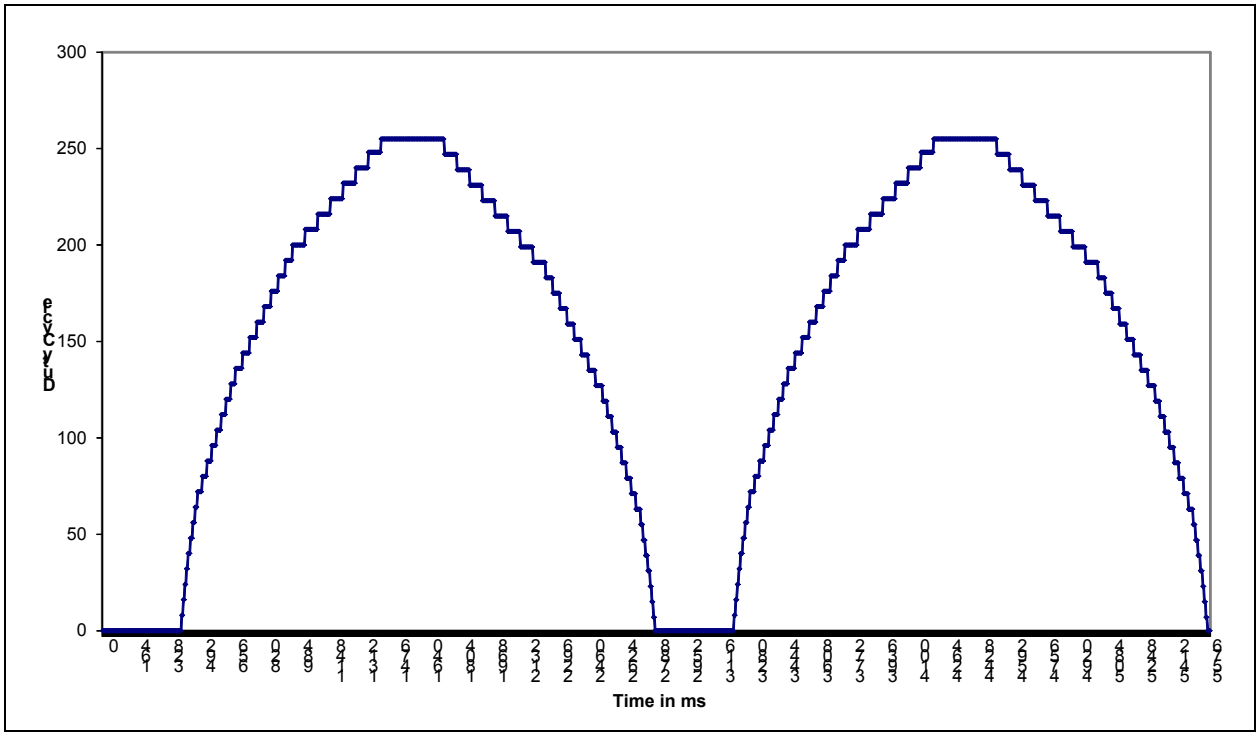
In this example, the brightness of the LED increases and diminishes in a non-linear fashion. The brightness forms a curve that is approximated by four piece wise-linear line segments. The entire cycle takes about 2.8 seconds. The rise time and fall time are about 1 second, with a hold time at maximum brightness of 320ms and a hold time at minimum brightness of 400ms. The LED brightness varies between full off and full on. The PWM size is set to 7-bit, so the time unit for adjusting the PWM is approximately 4ms. The registers are configured as follows:

TABLE 23-9: NON-LINEAR EXAMPLE CONFIGURATION

Field	Value							
PSIZE	7-bit							
MAX	255 (effectively 127)							
MIN	0							
HD	80 ticks (320ms)							
LD	100 ticks (400ms)							
Duty cycle most significant bits	000b	001b	010b	011b	100b	101b	110b	1110
LED_INT	2	3	6	6	9	9	16	16
LED_STEP	4	4	4	4	4	4	4	4

The resulting curve is shown in the following figure:

FIGURE 23-6: NON-LINEAR BRIGHTNESS CURVE EXAMPLE



23.10 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the Blinking/Breathing PWM Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 23-10: REGISTER SUMMARY

Offset	Register Name
00h	LED Configuration Register
04h	LED Limits Register
08h	LED Delay Register
0Ch	LED Update Stepsize Register
10h	LED Update Interval Register
14h	LED Output Delay

In the following register definitions, a “PWM period” is defined by time the PWM counter goes from 000h to its maximum value (FFh in 8-bit mode, FEh in 7-bit mode and FCh in 6-bit mode, as defined by the PSCALE field in register LED_CFG). The end of a PWM period occurs when the PWM counter wraps from its maximum value to 0.

The registers in this block can be written 32-bits, 16-bits or 8-bits at a time. Writes to LED Configuration Register take effect immediately. Writes to LED Limits Register are held in a holding register and only take effect only at the end of a PWM period. The update takes place at the end of every period, even if only one byte of the register was updated. This means that in blink/PWM mode, software can change the duty cycle with a single 8-bit write to the MIN field in the LED_LIMIT register. Writes to LED Delay Register, LED Update Stepsize Register and LED Update Interval Register also go initially into a holding register. The holding registers are copied to the operating registers at the end of a PWM period only if the Enable Update bit in the LED Configuration Register is set to 1. If LED_CFG is 0, data in the holding registers is retained but not copied to the operating registers when the PWM period expires. To change an LED breath-

ing configuration, software should write these three registers with the desired values and then set LED_CFG to 1. This mechanism ensures that all parameters affecting LED breathing will be updated consistently, even if the registers are only written 8 bits at a time.

23.10.1 LED CONFIGURATION REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	Reserved	R	-	-
16	SYMMETRY 1=The rising and falling ramp times are in Asymmetric mode. Table 23-7, "Asymmetric Breathing Mode Register Usage" shows the application of the Stepsize and Interval registers to the four segments of rising duty cycles and the four segments of falling duty cycles. 0=The rising and falling ramp times (as shown in Figure 23-2, "Breathing LED Example") are in Symmetric mode. Table 23-6, "Symmetric Breathing Mode Register Usage" shows the application of the Stepsize and Interval registers to the 8 segments of both rising and falling duty cycles.	R/W	0b	RESET_SYS
15:8	WDT_RELOAD The PWM Watchdog Timer counter reload value. On system reset, it defaults to 14h, which corresponds to a 4 second Watchdog timeout value.	R/W	14h	RESET_SYS
7	RESET Writes of'1' to this bit resets the PWM registers to their default values. This bit is self clearing. Writes of '0' to this bit have no effect.	W	0b	RESET_SYS
6	ENABLE_UPDATE This bit is set to 1 when written with a '1'. Writes of '0' have no effect. Hardware clears this bit to 0 when the breathing configuration registers are updated at the end of a PWM period. The current state of the bit is readable any time. This bit is used to enable consistent configuration of LED_DELAY, LED_STEP and LED_INT. As long as this bit is 0, data written to those three registers is retained in a holding register. When this bit is 1, data in the holding register are copied to the operating registers at the end of a PWM period. When the copy completes, hardware clears this bit to 0.	R/WS	0b	RESET_SYS
5:4	PWM_SIZE This bit controls the behavior of PWM: 3=Reserved 2=PWM is configured as a 6-bit PWM 1=PWM is configured as a 7-bit PWM 0=PWM is configured as an 8-bit PWM	R/W	0b	RESET_SYS

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
3	SYNCHRONIZE When this bit is '1', all counters for all LEDs are reset to their initial values. When this bit is '0' in the LED Configuration Register for all LEDs, then all counters for LEDs that are configured to blink or breathe will increment or decrement, as required. To synchronize blinking or breathing, the SYNCHRONIZE bit should be set for at least one LED, the control registers for each LED should be set to their required values, then the SYNCHRONIZE bits should all be cleared. If the all LEDs are set for the same blink period, they will all be synchronized.	R/W	0b	RESET_SYS
2	CLOCK_SOURCE This bit controls the base clock for the PWM. It is only valid when CNTRL is set to blink (2). 1=Clock source is the Main system clock 0=Clock source is the 32.768 KHz clock	R/W	0b	RESET_SYS
1:0	CONTROL This bit controls the behavior of PWM: 3=PWM is always on 2=LED blinking (standard PWM) 1=LED breathing configuration 0=PWM is always off. All internal registers and counters are reset to 0. Clocks are gated	R/W	00b	RESET_SYS
			11b	WDT TC

23.10.2 LED LIMITS REGISTER

This register may be written at any time. Values written into the register are held in an holding register, which is transferred into the actual register at the end of a PWM period. The two byte fields may be written independently. Reads of this register return the current contents and not the value of the holding register.

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	Reserved	R	-	-
15:8	MAXIMUM In breathing mode, when the current duty cycle is greater than or equal to this value the breathing apparatus holds the current duty cycle for the period specified by the field HD in register LED_DELAY, then starts decrementing the current duty cycle	R/W	0h	RESET_SYS
7:0	MINIMUM In breathing mode, when the current duty cycle is less than or equal to this value the breathing apparatus holds the current duty cycle for the period specified by the field LD in register LED_DELAY, then starts incrementing the current duty cycle In blinking mode, this field defines the duty cycle of the blink function.	R/W	0h	RESET_SYS

23.10.3 LED DELAY REGISTER

This register may be written at any time. Values written into the register are held in an holding register, which is transferred into the actual register at the end of a PWM period if the Enable Update bit in the LED Configuration register is set to 1. Reads of this register return the current contents and not the value of the holding register.

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:24	Reserved	R	-	-
23:12	HIGH_DELAY In breathing mode, the number of PWM periods to wait before updating the current duty cycle when the current duty cycle is greater than or equal to the value MAX in register LED_LIMIT. 4095=The current duty cycle is decremented after 4096 PWM periods … 1=The delay counter is bypassed and the current duty cycle is decremented after two PWM period 0=The delay counter is bypassed and the current duty cycle is decremented after one PWM period	R/W	000h	RESET_SYS
11:0	LOW_DELAY The number of PWM periods to wait before updating the current duty cycle when the current duty cycle is greater than or equal to the value MIN in register LED_LIMIT. 4095=The current duty cycle is incremented after 4096 PWM periods … 0=The delay counter is bypassed and the current duty cycle is incremented after one PWM period In blinking mode, this field defines the prescalar for the PWM clock	R/W	000h	RESET_SYS

23.10.4 LED UPDATE STEPSIZE REGISTER

This register has eight segment fields which provide the amount the current duty cycle is adjusted at the end of every PWM period. Segment field selection is decoded based on the segment index. The segment index equation utilized depends on the SYMMETRY bit in the LED Configuration Register Register)

- In Symmetric Mode the Segment_Index[2:0] = Duty Cycle Bits[7:5]
- In Asymmetric Mode the Segment_Index[2:0] is the bit concatenation of following: Segment_Index[2] = (FALLING RAMP TIME in Figure 36-3, "Clipping Example") and Segment_Index[1:0] = Duty Cycle Bits[7:6].

This register may be written at any time. Values written into the register are held in an holding register, which is transferred into the actual register at the end of a PWM period if the Enable Update bit in the LED Configuration register is set to 1. Reads of this register return the current contents and not the value of the holding register.

In 8-bit mode, each 4-bit STEPSIZE field represents 16 possible duty cycle modifications, from 1 to 16 as the duty cycle is modified between 0 and 255:

15: Modify the duty cycle by 16

...

1: Modify the duty cycle by 2

0=Modify the duty cycle by 1

In 7-bit mode, the least significant bit of the 4-bit field is ignored, so each field represents 8 possible duty cycle modifications, from 1 to 8, as the duty cycle is modified between 0 and 127:

14, 15: Modify the duty cycle by 8

...

2, 3: Modify the duty cycle by 2

0, 1: Modify the duty cycle by 1

In 6-bit mode, the two least significant bits of the 4-bit field is ignored, so each field represents 4 possible duty cycle modifications, from 1 to 4 as the duty cycle is modified between 0 and 63:

12, 13, 14, 15: Modify the duty cycle by 4

8, 9, 10, 11: Modify the duty cycle by 3

4, 5, 6, 7: Modify the duty cycle by 2

0, 1, 2, 3: Modify the duty cycle by 1

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:28	UPDATE_STEP7 Amount the current duty cycle is adjusted at the end of every PWM period when the segment index is equal to 111.	R/W	0h	RESET_SYS
27:24	UPDATE_STEP6 Amount the current duty cycle is adjusted at the end of every PWM period when the segment index is equal to 110.	R/W	0h	RESET_SYS
23:20	UPDATE_STEP5 Amount the current duty cycle is adjusted at the end of every PWM period when the segment index is equal to 101	R/W	0h	RESET_SYS
19:16	UPDATE_STEP4 Amount the current duty cycle is adjusted at the end of every PWM period when the segment index is equal to 100.	R/W	0h	RESET_SYS
15:12	UPDATE_STEP3 Amount the current duty cycle is adjusted at the end of every PWM period when the segment index is equal to 011.	R/W	0h	RESET_SYS
11:8	UPDATE_STEP2 Amount the current duty cycle is adjusted at the end of every PWM period when the segment index is equal to 010.	R/W	0h	RESET_SYS
7:4	UPDATE_STEP1 Amount the current duty cycle is adjusted at the end of every PWM period when the segment index is equal to 001.	R/W	0h	RESET_SYS
3:0	UPDATE_STEP0 Amount the current duty cycle is adjusted at the end of every PWM period when the segment index is equal to 000.	R/W	0h	RESET_SYS

23.10.5 LED UPDATE INTERVAL REGISTER

This register has eight segment fields which provide the number of PWM periods between updates to current duty cycle. Segment field selection is decoded based on the segment index. The segment index equation utilized depends on the SYMMETRY bit in the LED Configuration Register Register)

- In Symmetric Mode the Segment_Index[2:0] = Duty Cycle Bits[7:5]
- In Asymmetric Mode the Segment_Index[2:0] is the bit concatenation of following: Segment_Index[2] = (FALLING RAMP TIME in Figure 36-3, "Clipping Example") and Segment_Index[1:0] = Duty Cycle Bits[7:6].

This register may be written at any time. Values written into the register are held in an holding register, which is transferred into the actual register at the end of a PWM period if the Enable Update bit in the LED Configuration register is set to 1. Reads of this register return the current contents and not the value of the holding register.

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:28	UPDATE_INTERVAL7 The number of PWM periods between updates to current duty cycle when the segment index is equal to 111b. 15=Wait 16 PWM periods … 0=Wait 1 PWM period	R/W	0h	RESET_SYS
27:24	UPDATE_INTERVAL6 The number of PWM periods between updates to current duty cycle when the segment index is equal to 110b. 15=Wait 16 PWM periods … 0=Wait 1 PWM period	R/W	0h	RESET_SYS
23:20	UPDATE_INTERVAL5 The number of PWM periods between updates to current duty cycle when the segment index is equal to 101b. 15=Wait 16 PWM periods … 0=Wait 1 PWM period	R/W	0h	RESET_SYS
19:16	UPDATE_INTERVAL4 The number of PWM periods between updates to current duty cycle when the segment index is equal to 100b. 15=Wait 16 PWM periods … 0=Wait 1 PWM period	R/W	0h	RESET_SYS
15:12	UPDATE_INTERVAL3 The number of PWM periods between updates to current duty cycle when the segment index is equal to 011b. 15=Wait 16 PWM periods … 0=Wait 1 PWM period	R/W	0h	RESET_SYS
11:8	UPDATE_INTERVAL2 The number of PWM periods between updates to current duty cycle when the segment index is equal to 010b. 15=Wait 16 PWM periods … 0=Wait 1 PWM period	R/W	0h	RESET_SYS

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:4	UPDATE_INTERVAL1 The number of PWM periods between updates to current duty cycle when the segment index is equal to 001b. 15=Wait 16 PWM periods … 0=Wait 1 PWM period	R/W	0h	RESET_SYS
3:0	UPDATE_INTERVAL0 The number of PWM periods between updates to current duty cycle when the segment index is equal to 000b. 15=Wait 16 PWM periods … 0=Wait 1 PWM period	R/W	0h	RESET_SYS

23.10.6 LED OUTPUT DELAY

This register permits the transitions for multiple blinking/breathing LED outputs to be skewed, so as not to present too great a current load. The register defines a count for the number of clocks the circuitry waits before turning on the output, either on initial enable, after a resume from Sleep, or when multiple outputs are synchronized through the Sync control in the LED CONFIGURATION (LED_CFG) register.

When more than one LED outputs are used simultaneously, the LED OUTPUT DELAY fields of each should be configured with different values so that the outputs are skewed. When used with the 32KHz clock domain as a clock source, the differences can be as small as 1.

Offset	14h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:8	Reserved	R	-	-
7:0	OUTPUT_DELAY The delay, in counts of the clock defined in Clock Source (CLKSRC), in which output transitions are delayed. When this field is 0, there is no added transition delay. When the LED is programmed to be Always On or Always Off, the Output Delay field has no effect.	R/W	000h	RESET_SYS

24.0 RC IDENTIFICATION DETECTION (RC_ID)

24.1 Introduction

The Resistor/Capacitor Identification Detection (RC_ID) interface provides a single pin interface which can discriminate a number of quantized RC constants.

24.2 References

No references have been cited for this feature.

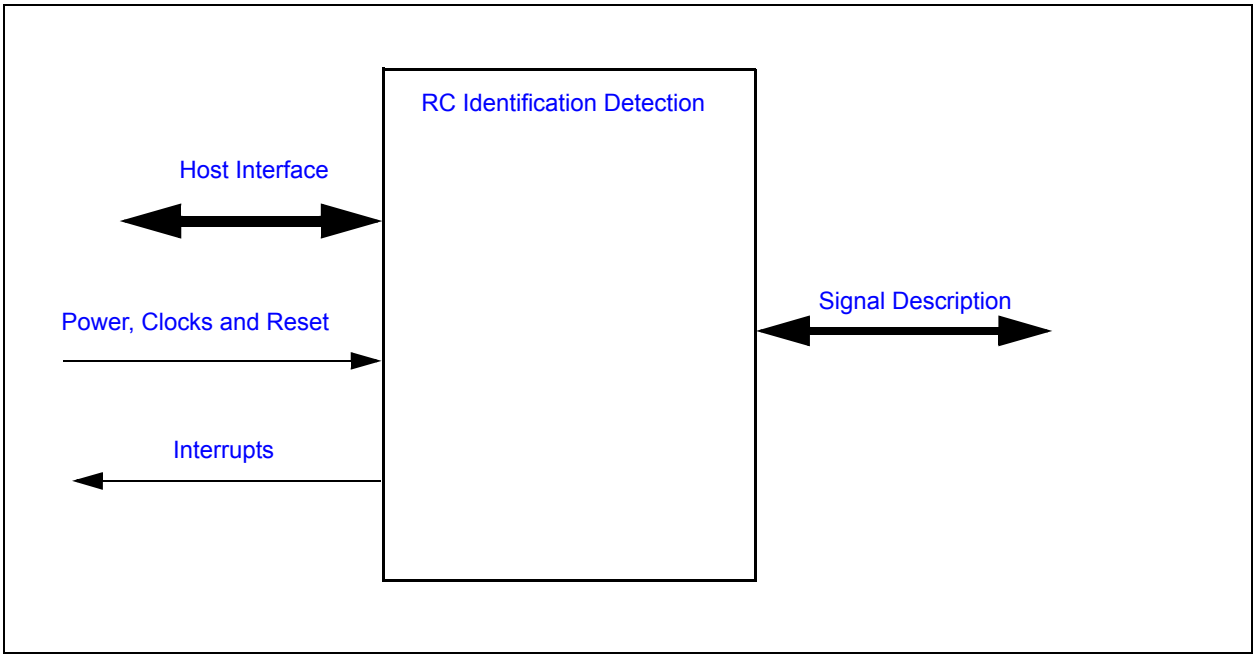
24.3 Terminology

There is no terminology defined for this section.

24.4 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

FIGURE 24-1: I/O DIAGRAM OF BLOCK



24.5 Signal Description

Name	Direction	Description
RC_ID	Input	Analog input used for measuring an external Resistor-Capacitor delay.

24.6 Host Interface

The registers defined for this block are accessible by the various hosts as indicated in Section 24.12, "EC Registers".

24.7 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

24.7.1 POWER DOMAINS

Name	Description
VTR	The logic and registers implemented in this block are powered by this power well.

24.7.2 CLOCK INPUTS

Name	Description
48MHz	The main clock domain, used to generate the time base that measures the RC delay.

24.7.3 RESETS

Name	Description
RESET_SYS	This signal resets all the registers and logic in this block to their default state.

24.8 Interrupts

This section defines the Interrupt Sources generated from this block.

Source	Description
RCID	This internal signal is generated when the DONE bit in the RC_ID Control Register is set to '1'.

24.9 Low Power Modes

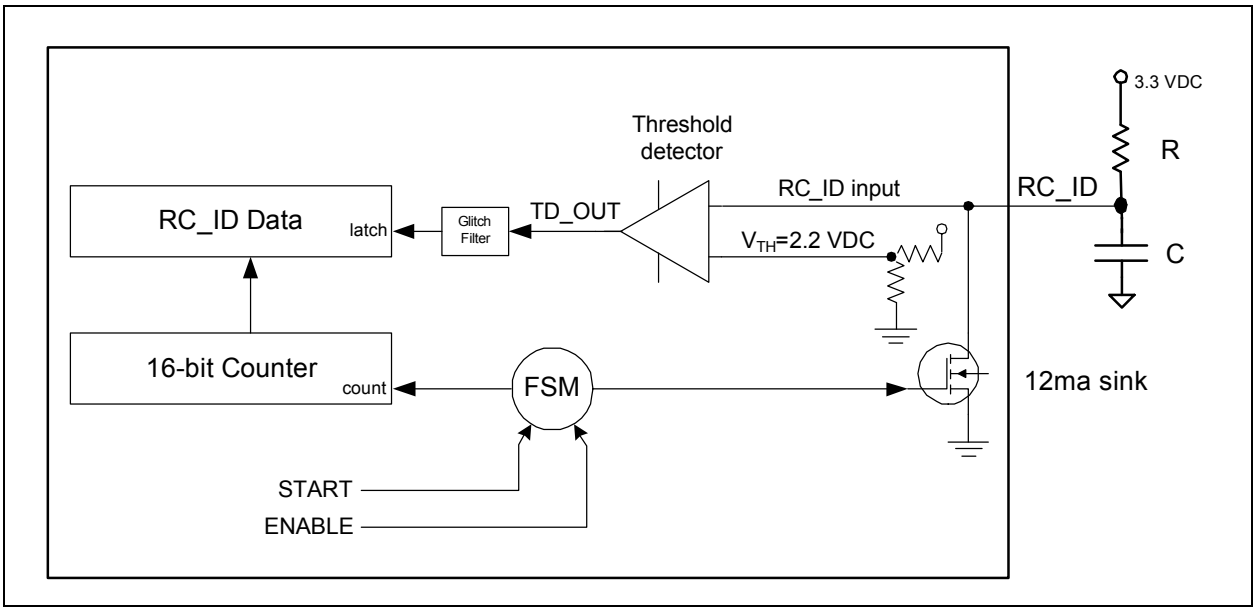
This block may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry. If a measurement has been started, the block will continue to assert its clock_req output until the measurement completes.

24.10 Description

Note: The RC_ID block only operates on 3.3V. The VTR pin associated with RC_ID signals must be connected to a 3.3V supply. If the VTR pin is supplied with 1.8V, the RC_ID logic will not function correctly.

The Resistor/Capacitor Identification Detection (RC_ID) interface provides a single pin interface which can discriminate a number of quantized RC constants. The judicious selection of RC values can provide a low cost means for system element configuration identification. The RC_ID I/O pin measures the charge/discharge time for an RC circuit connected to the pin as shown in Figure 24-2.

FIGURE 24-2: BLOCK DIAGRAM OF RC Identification Detection (RC_ID)



The RC_ID interface determines the selected RC delay by measuring the rise time on the RC_ID pin that is attached to the RC circuit, as shown in the above figure. The measurement is performed by first discharging the external capacitor for a fixed period of time, set by an internal 16-bit counter running at a configurable time base, and then letting the capacitor charge again, using the same counter and time base to count how many clock ticks are required until the voltage on the capacitor exceeds 2.2V. A glitch filter, consisting of three ticks of the 48MHz main oscillator, smooths the threshold detection.

By fixing the capacitor value and varying the resistor value, up to eight discrete values can be determined based on the final count. Section 24.11, "Time Constants" shows a range of possible R and C values that can be used to create eight ID values.

Measurement requires five phases:

1. Reset. The two control bits (ENABLE and START) and the three status bits (TC, DONE and CY_ER) in the RC_ID Control Register are all '0'. The RD_IC pin is tri-stated and the block is in its lowest power state. In order to enter the Reset state, firmware must write the ENABLE, START and CLOCK_SET fields to '0' simultaneously or unpredictable results may occur.
2. Armed. Firmware enables the transition to this state by setting the ENABLE bit to '1' and the CLOCK_SET field to the desired time base. The START must remain at '0'. All three fields must be set with one write to the RC_ID Control Register. In this state the RC_ID clock is enabled and the 16-bit counter is armed. Firmware must wait a minimum of 300µS in the Armed phase before starting the Discharged phase.
3. Discharged. Firmware initiates the transition to the Discharged state by setting the ENABLE bit to '1', the START bit to '1' and the CLOCK_SET field to the desired clock rate, in a single write to the RC_ID Control Register. The RC_ID pin is discharged while the 16-bit counter counts from 0000h to FFFFh at the configured time base. When the counter reaches FFFFh the TC status bit is set to '1'. If at the end of the Discharged state the RC_ID pin remains above the 2.2V threshold, the CY_ER bit is set to '1', since the measurement will not be valid.
4. Charged.The RC_ID state machine automatically transitions to this state after the 16-bit counter reaches FFFFh while in the Discharged state. The 16-bit counter starts counting up from 0000h. The counter stops counting and its value is copied into the RC_ID Data Register when the voltage on the pin exceeds 2.2V. If the counter reaches FFFFh and the pin voltage remains below 2.2V, the CY_ER bit is set to '1'.
5. Done. After the counter stops counting, either because the pin voltage exceed the 2.2V threshold or the 16-bit counter reached FFFFh, the state machine transitions to this state. The DONE bit is set to '1' and the RC_ID interface re-enters its lowest power state. The interface will remain in the Done state until firmware explicitly initiates the Reset state.

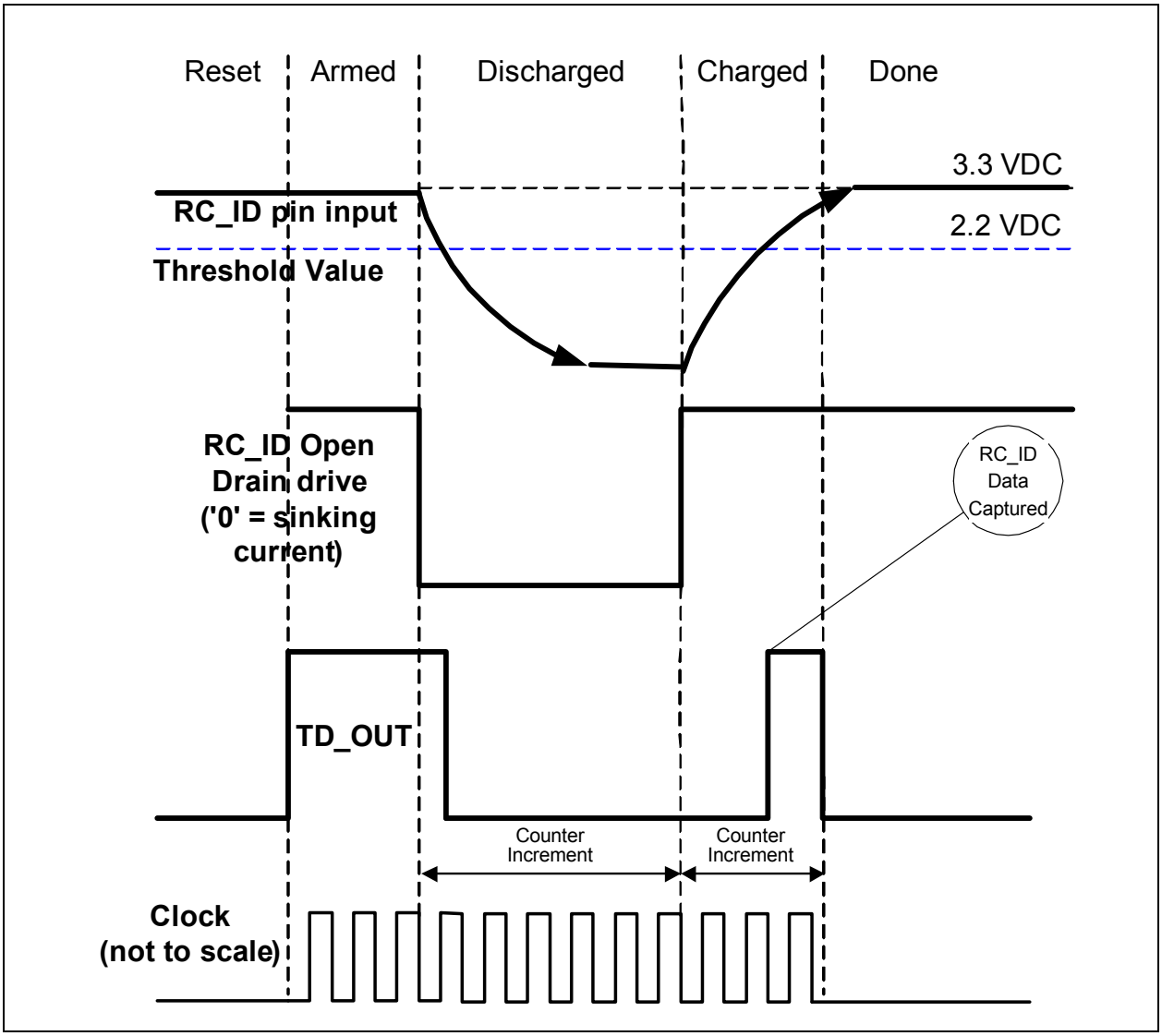
A new measurement must be started by putting the RC_ID Interface into the "Reset" state.

The five phases, along with the values of the control and status bits in the Control Register at the end of each phase, are summarized in the following table and figure:

TABLE 24-1: RC ID STATE TRANSITIONS

	State	ENABLE	START	TC	DONE
1.	Reset	0	0	0	0
2.	Armed	1	0	0	0
3.	Discharged	1	1	0	0
4.	Charged	1	1	1	0
5.	Done	1	1	1	1

FIGURE 24-3: RCID STATE TRANSITIONS



24.11 Time Constants

This section lists a set of R and C values which can be connected to the RC_ID pin. Note that risetime generally follow RC time Tau. Firmware should use the Max and Min Counts in the tables to create quantized states.

In the following tables, the CLOCK_SET field in the RC_ID Control Register is set to '1', so the time base for measuring the rise time is 24MHz, the speed of the system clock. All capacitor values are ±10% and all resistor values are ±5%. Minimum and maximum count values are suggested ranges, calculated to provide reasonable margins around the nominal rise times. Rise times have been confirmed by laboratory measurements.

TABLE 24-2: SAMPLE RC VALUES, C=2200PF @24MHZ

R (KΩ)	Nominal Tau (µS)	Minimum Count	Maximum Count
1	2.2	60.00	72.00
2	4.4	115.00	140.00
4.3	9.5	241.00	294.00
8.2	18.04	456.00	557.00
33	72.6	1819.00	2224.00
62	136.4	3456.00	4224.00
130	286	7470.00	9130.00
240	528	14400.00	17600.00

TABLE 24-3: SAMPLE RC VALUES, C=3000PF @24MHZ

R (KΩ)	Nominal Tau (µS)	Minimum Count	Maximum Count
1	3	77.00	95.00
2	6	151.00	184.00
4.3	12.9	320.00	391.00
8.2	24.6	604.00	739.00
33	99	2439.00	2981.00
62	186	4647.00	5680.00
130	390	9990.00	12210.00
240	720	193508.00	23650.00

TABLE 24-4: SAMPLE RC VALUES, C=4700PF @24MHZ

R (KΩ)	Nominal Tau (µS)	Minimum Count	Maximum Count
1	4.7	116.00	142.00
2	9.4	229.00	280.00
4.3	20.2	495.00	605.00
8.2	38.5	945.00	1160.00
33	155.1	3780.00	4650.00
62	291.4	7249.00	8859.00
130	611	15480.00	18920.00
240	1128	29880.00	36520.00

24.12 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the RC Identification Detection (RC_ID) Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 24-5: REGISTER SUMMARY

Offset	Register Name
00h	RC_ID Control Register
04h	RC_ID Data Register

24.12.1 RC_ID CONTROL REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:10	Reserved	R	-	-
9:8	CLOCK_SET This field selects the frequency of the Counter circuit clock. This field must retain the same value as long as the ENABLE bit in this register is '1'. 3=6MHz 2=12MHz 1=24MHz 0=48MHz	R/W	0h	RESET_SYS
7	ENABLE Clearing the bit to '0' causes the RC_ID interface to enter the Reset state, gating its clocks, clearing the status bits in this register and entering into its lowest power state. Setting this bit to '1' causes the RC_ID interface to enter the Armed phase of an RC_ID measurement. When this bit is cleared to '0', the CLOCK_SET and START fields in this register must also be cleared to '0' in the same register write.	R/W	0h	RESET_SYS
6	START Setting this bit to '1' initiates the Discharged phase of an RC_ID measurement. Writes that change this bit from '0' to '1' must also write the ENABLE bit to '1', and must not change the CLOCK_SET field. A period of at least 300µS must elapse between setting the ENABLE bit to '1' and setting this bit to '1'.	R/W	0h	RESET_SYS
5:3	Reserved	R	-	-

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
2	CY_ER This bit is ‘1’ if an RC_ID measurement encountered an error and the reading in the RC_ID Data Register is invalid. This bit is cleared to ‘0’ when the RC_ID interface is in the Reset phase. It is set either if during the Discharged phase the RC_ID pin did not fall below the 2.2V threshold, or if in the Charged phase the RC_ID pin did not rise above the 2.2V threshold and the 16-bit counter ended its count at FFFFh.	R	0h	RESET_SYS
1	TC This bit is cleared to ‘0’ when the RC_ID interface is in the Reset phase, and set to ‘1’ when the interface completes the Discharged phase of an RC_ID measurement.	R	0h	RESET_SYS
0	DONE This bit is cleared to ‘0’ when the RC_ID interface is in the Reset phase, and set to ‘1’ when the interface completes an RC_ID measurement.	R	0h	RESET_SYS

24.12.2 RC_ID DATA REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	Reserved	R	-	-
15:0	DATA Reads of this register provide the result of an RC_ID measurement.	R	0h	RESET_SYS

25.0 KEYBOARD SCAN INTERFACE

25.1 Overview

The Keyboard Scan Interface block provides a register interface to the EC to directly scan an external keyboard matrix of size up to 18x8.

The maximum configuration of the Keyboard Scan Interface is 18 outputs by 8 inputs. For a smaller matrix size, firmware should configure unused KSO pins as GPIOs or another alternate function, and it should mask out unused KSIs and associated interrupts.

25.2 References

No references have been cited for this feature.

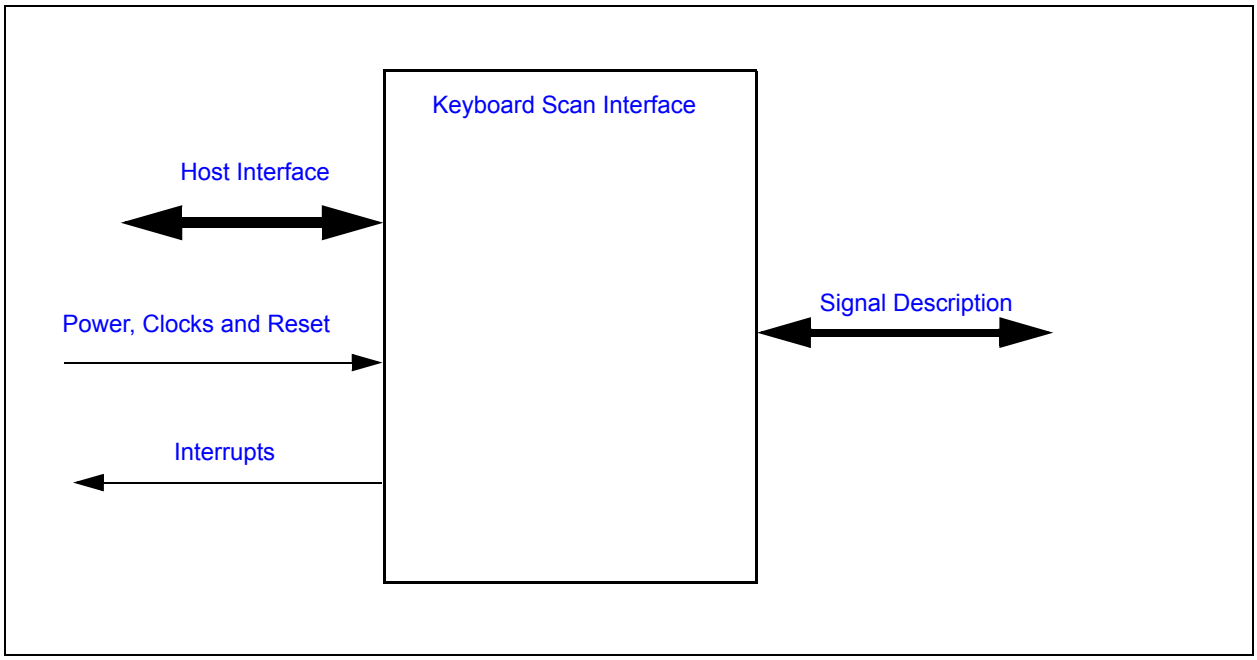
25.3 Terminology

There is no terminology defined for this section.

25.4 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

FIGURE 25-1: I/O DIAGRAM OF BLOCK



25.5 Signal Description

Name	Direction	Description
KSI[7:0]	Input	Column inputs from external keyboard matrix.
KSO[17:0]	Output	Row outputs to external keyboard matrix.

25.6 Host Interface

The registers defined for the Keyboard Scan Interface are accessible by the various hosts as indicated in Section 25.11, "EC Registers".

25.7 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

25.7.1 POWER DOMAINS

Name	Description
VTR	The logic and registers implemented in this block are powered by this power well.

25.7.2 CLOCK INPUTS

Name	Description
48MHz	This is the clock source for Keyboard Scan Interface logic.

25.7.3 RESETS

Name	Description
RESET_SYS	This signal resets all the registers and logic in this block to their default state.

25.8 Interrupts

This section defines the Interrupt Sources generated from this block.

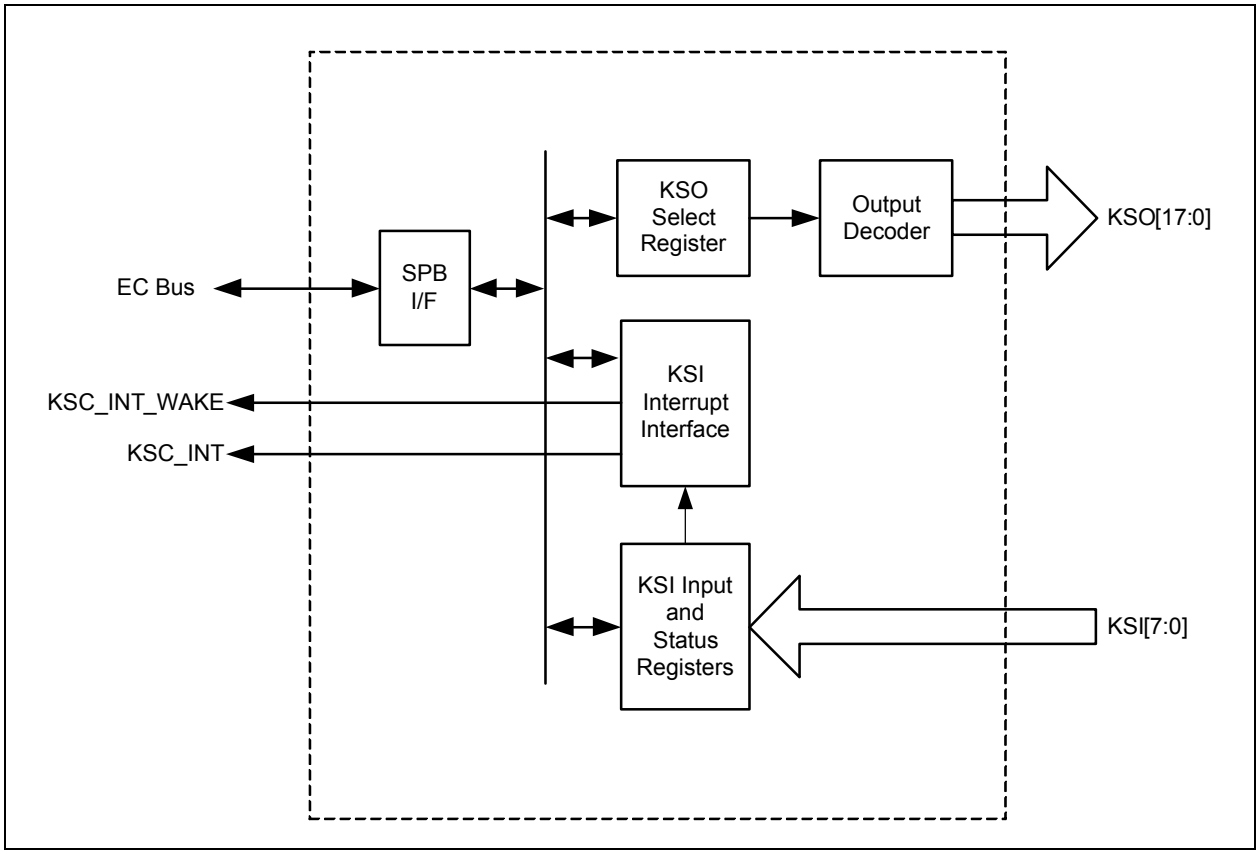
Source	Description
KSC_INT	Interrupt request to the Interrupt Aggregator.
KSC_INT_WAKE	Wake-up request to the Interrupt Aggregator’s wake-up interface.

25.9 Low Power Modes

The Keyboard Scan Interface automatically enters a low power mode whenever it is not actively scanning the keyboard matrix. The block is also placed in a low-power state when it is disabled by the KSEN bit. When the interface is in a low-power mode it will not prevent the chip from entering a sleep state. When the interface is active it will inhibit the chip sleep state until the interface has re-entered its low power mode.

25.10 Description

FIGURE 25-2: KEYBOARD SCAN INTERFACE BLOCK DIAGRAM



During scanning the firmware sequentially drives low one of the rows (KSO[17:0]) and then reads the column data line (KSI[7:0]). A key press is detected as a zero in the corresponding position in the matrix. Keys that are pressed are debounced by firmware. Once confirmed, the corresponding keycode is loaded into host data read buffer in the 8042 Host Interface module. Firmware may need to buffer keycodes in memory in case this interface is stalled or the host requests a Resend.

25.10.1 INITIALIZATION OF KSO PINS

If the Keyboard Scan Interface is not configured for PREDRIVE Mode, KSO pins should be configured as open-drain outputs. Internal or external pull-ups should be used so that the GPIO functions that share the pins do not have a floating input when the KSO pins are tri-stated.

If the Keyboard Scan Interface is configured for PREDRIVE Mode, KSO pins must be configured as push-pull outputs. Internal or external pull-ups should be used to protect the GPIO inputs associated with the KSO pins from floating inputs.

25.10.2 PREDRIVE MODE

There is an optional Predrive Mode that can be enabled to actively drive the KSO pins high before switching to open-drain operation. The PREDRIVE ENABLE bit in the Keyscan Extended Control Register is used to enable the PRE-DRIVE option. Timing for the Predive mode is shown in Section 38.8, Keyboard Scan Matrix Timing.

25.10.2.1 Predrive Mode Programming

The following precautions should be taken to prevent output pad damage during Predrive Mode Programming.

25.10.2.2 Asserting PREDRIVE_ENABLE

1. Disable Key Scan Interface (KSEN = '1')
2. Enable Predrive function (PREDRIVE_ENABLE = '1')
3. Program buffer type for all KSO pins to "push-pull”
4. Enable Keyscan Interface (KSEN ='0')

25.10.2.3 De-asserting PREDRIVE_ENABLE

1. Disable Key Scan Interface (KSEN = '1')
2. Program buffer type for all KSO pins to "open-drain”
3. Disable Predrive function (PREDRIVE_ENABLE = '0')
4. Enable Keyscan Interface (KSEN ='0')

25.10.3 INTERRUPT GENERATION

To support interrupt-based processing, an interrupt can optionally be generated on the high-to-low transition on any of the KSI inputs. A running clock is not required to generate interrupts.

25.10.3.1 Runtime interrupt

KSC_INT is the block’s runtime active-high level interrupt. It is connected to the interrupt interface of the Interrupt Aggregator, which then relays interrupts to the EC.

Associated with each KSI input is a status register bit and an interrupt enable register bit. A status bit is set when the associated KSI input goes from high to low. If the interrupt enable bit for that input is set, an interrupt is generated. An Interrupt is de-asserted when the status bit and/or interrupt enable bit is clear. A status bit cleared when written to a ‘1’.

Interrupts from individual KSIs are logically ORed together to drive the KSC_INT output port. Once asserted, an interrupt is not asserted again until either all KSI[7:0] inputs have returned high or the has changed.

25.10.3.2 Wake-up interrupt

KSC_INT_WAKE is the block’s wakeup interrupt. It is routed to the Interrupt Aggregator.

During sleep mode, i.e., when the bus clock is stopped, a high-to-low transition on any KSI whose interrupt enable bit is set causes the KSC_INT_WAKE to be asserted. The block also indicates that it requires a clock to the system Power Management Interface. KSC_WAKEUP_INT remains active until the bus clock is started.

The aforementioned transition on KSI also sets the corresponding status bit in the KSI STATUS Register. If enabled, a runtime interrupt is also asserted on KSC_INT when the bus clock resumes running.

25.10.4 WAKE PROGRAMMING

Using the Keyboard Scan Interface to ‘wake’ the CEC1702 can be accomplished using either the Keyboard Scan Interface wake interrupt, or using the wake capabilities of the GPIO Interface pins that are multiplexed with the Keyboard Scan Interface pins. Enabling the Keyboard Scan Interface wake interrupt requires only a single interrupt enable access and is recommended over using the GPIO Interface for this purpose.

25.11 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the Keyboard Scan Interface Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 25-1: EC-ONLY REGISTER SUMMARY

Offset	Register Name
0h	Reserved
4h	KSO Select Register
8h	KSI INPUT Register
Ch	KSI STATUS Register
10h	KSI INTERRUPT ENABLE Register
14h	Keyscan Extended Control Register

25.11.1 KSO SELECT REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:4	Reserved	R	-	-
7	KSO_INVERT This bit controls the output level of KSO pins when selected. 0=KSO[x] driven low when selected 1=KSO[x] driven high when selected.	R/W	0h	RESET_SYS
6	KSEN This field enables and disables keyboard scan 0=Keyboard scan enabled 1=Keyboard scan disabled. All KSO output buffers disabled.	R/W	1h	RESET_SYS
5	KSO_ALL 0=When key scan is enabled, KSO output controlled by the KSO_SELECT field. 1=KSO[x] driven high when selected.	R/W	0h	RESET_SYS
4:0	KSO_SELECT This field selects a KSO line (00000b = KSO[0] etc.) for output according to the value off KSO_INVERT in this register. See Table 25-2, "KSO Select Decode"	R/W	0h	RESET_SYS

TABLE 25-2: KSO SELECT DECODE

KSO Select [4:0]	KSO Selected
00h	KSO00
01h	KSO01
02h	KSO02
03h	KSO03
04h	KSO04
05h	KSO05
06h	KSO06
07h	KSO07
08h	KSO08
09h	KSO09
0Ah	KSO10
0Bh	KSO11
0Ch	KSO12
0Dh	KSO13
0Eh	KSO14
0Fh	KSO15

TABLE 25-2: KSO SELECT DECODE (CONTINUED)

KSO Select [4:0]	KSO Selected
10h	KSO16
11h	KSO17

TABLE 25-3: KEYBOARD SCAN OUT CONTROL SUMMARY

KSO_INVERTt	KSEN	KSO_ALL	KSO_SELECT	Description
X	1	x	x	Keyboard Scan disabled. KSO[17:0] output buffers disabled.
0	0	0	10001b-00000b	KSO[Drive Selected] driven low. All others driven high
1	0	0	10001b-00000b	KSO[Drive Selected] driven high. All others driven low
0	0	0	11111b-10010b	All KSO's driven high
1	0	0	11111b-10010b	All KSO's driven low
0	0	1	x	All KSO's driven high
1	0	1	x	All KSO's driven low

25.11.2 KSI INPUT REGISTER

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:8	Reserved	R	-	-
7:0	KSI This field returns the current state of the KSI pins.	R	0h	RESET_SYS

25.11.3 KSI STATUS REGISTER

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:8	Reserved	R	-	-
7:0	KSI_STATUS Each bit in this field is set on the falling edge of the corresponding KSI input pin. A KSI interrupt is generated when its corresponding status bit and interrupt enable bit are both set. KSI interrupts are logically ORed together to produce KSC_INT and KSC_INT_WAKE. Writing a '1' to a bit will clear it. Writing a '0' to a bit has no effect.	R/WC	0h	RESET_SYS

25.11.4 KSI INTERRUPT ENABLE REGISTER

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:8	Reserved	R	-	-
7:0	KSI_INT_EN Each bit in KSI_INT_EN enables interrupt generation due to high-to-low transition on a KSI input. An interrupt is generated when the corresponding bits in KSI_STATUS and KSI_INT_EN are both set.	R/W	0h	RESET_SYS

25.11.5 KEYSCAN EXTENDED CONTROL REGISTER

Offset	14h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
32:1	Reserved	R	-	-
0	PREDRIVE_ENABLE PREDRIVE_ENABLE enables the PREDRIVE mode to actively drive the KSO pins high for two 48 MHz PLL clocks before switching to open-drain operation. 0=Disable predrive on KSO pins 1=Enable predrive on KSO pins.	R/W	0h	RESET_SYS

26.0 I2C/SMBUS INTERFACE

26.1 Introduction

This section describes the Power Domain, Resets, Clocks, Interrupts, Registers and the Physical Interface of the I2C/SMBus interface. For a General Description, Features, Block Diagram, Functional Description, Registers Interface and other core-specific details, *see* Ref [1] (note: in this chapter, *italicized text* typically refers to SMB-I2C Controller core interface elements as described in Ref [1]).

26.2 References

1. I2C_SMB Controller Core with Network Layer Support (SMB2) - 16MHz I2C Baud Clock“, Revision 3.6, Core-Level Architecture Specification, Microchip, date TBD

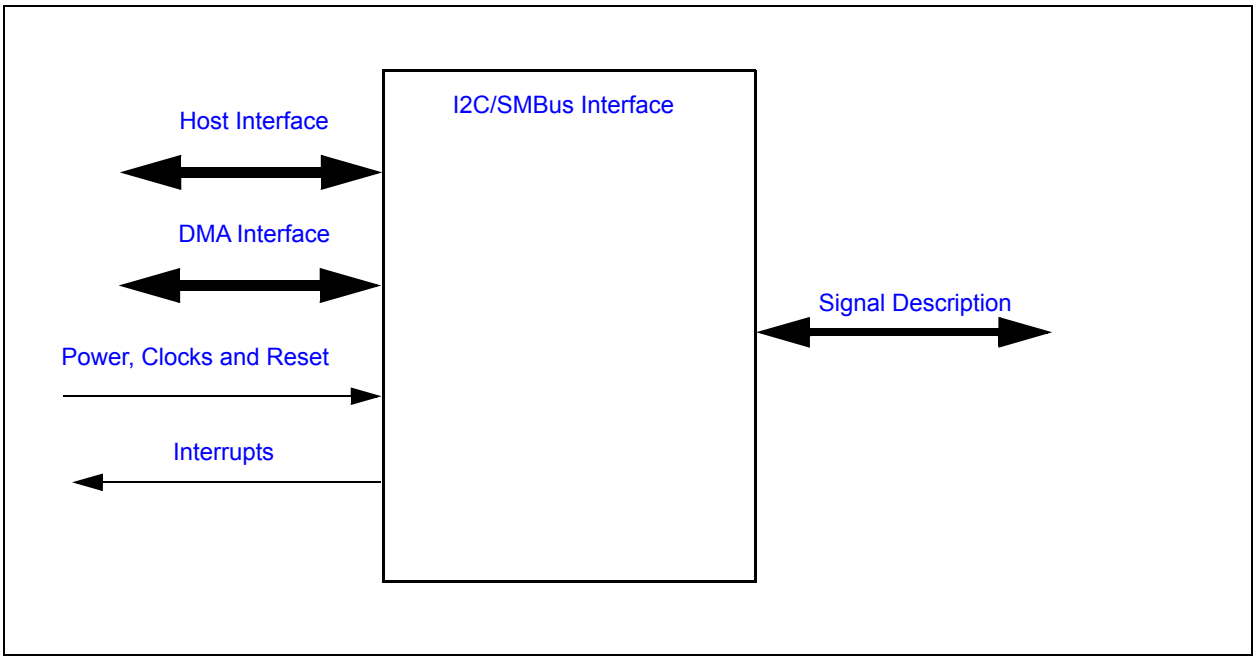
26.3 Terminology

There is no terminology defined for this chapter.

26.4 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface. In addition, this block is equipped with:

FIGURE 26-1: I/O DIAGRAM OF BLOCK



26.5 Signal Description

see the Pin Configuration section for a description of the SMB-I2C pin configuration.

26.6 Host Interface

The registers defined for the I2C/SMBus Interface are accessible as indicated in Section 26.12, "EC Registers".

26.7 DMA Interface

This block is designed to communicate with the Internal DMA Controller. This feature is defined in the SMB-I2C Controller Core Interface specification (*See* Ref [1]).

Note: For a description of the Internal DMA Controller implemented in this design see Section 7.0, "Internal DMA Controller".

26.8 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

26.8.1 POWER DOMAINS

Name	Description
VTR	This power well sources all of the registers and logic in this block, except where noted.

26.8.2 CLOCK INPUTS

Name	Description
16MHz	This is the clock signal drives the SMB-I2C Controller core. The core also uses this clock to generate the SMB-I2C_CLK on the pin interface. It is derived from the main system clock

26.8.3 RESETS

Name	Description
RESET_SYS	This reset signal resets all of the registers and logic in the SMB-I2C Controller core.

26.9 Interrupts

Source	Description
SMB-I2C	I^2C Activity Interrupt Event
SMB-I2C_WAKE	This interrupt event is triggered when an SMB/I2C Master initiates a transaction by issuing a START bit (a high-to-low transition on the SDA line while the SCL line is high) on the bus currently connected to the SMB-I2C Controller. The EC interrupt handler for this event only needs to clear the interrupt SOURCE bit and return; if the transaction results in an action that requires EC processing, that action will trigger the SMB-I2C interrupt event.

26.10 Low Power Modes

The SMB-I2C Controller may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry.

26.11 Description

26.11.1 SMB-I2C CONTROLLER CORE

The SMB-I2C Controller behavior is defined in the SMB-I2C Controller Core Interface specification (*See* Ref [1]).

26.11.2 PHYSICAL INTERFACE

The Physical Interface for the SMB-I2C Controller core is configurable for up to 15 ports. Each I2C_WAKE Controller can be connected to any of the ports defined in Table 26-1, "SMB-I2C Port Selection". The *PORT SEL [3:0]* bit field in each controller independently sets the port for the controller. The default for each field is Fh, Reserved, which means that the SMB-I2C Controller is not connected to a port.

An I^2C port should be connected to a single controller. An attempt to configure the *PORT SEL [3:0]* bits in one controller to a value already assigned to another controller may result in unexpected results.

The port signal-function names and pin numbers are defined in Pin Configuration section. The I^2C port selection is made using the *PORT SEL [3:0]* bits in the *Configuration Register* as described in Ref [1].. In the Pin section, the SDL (Data) pins are listed as SMB*i*_DATA and the SCL (Clock) pins are listed as I2C*i*_CLK, where *i* represents the port number 00 through 10. The CPU-voltage-level port SB_TSI is also listed in the pin section with the pins _DATA and _CLK.

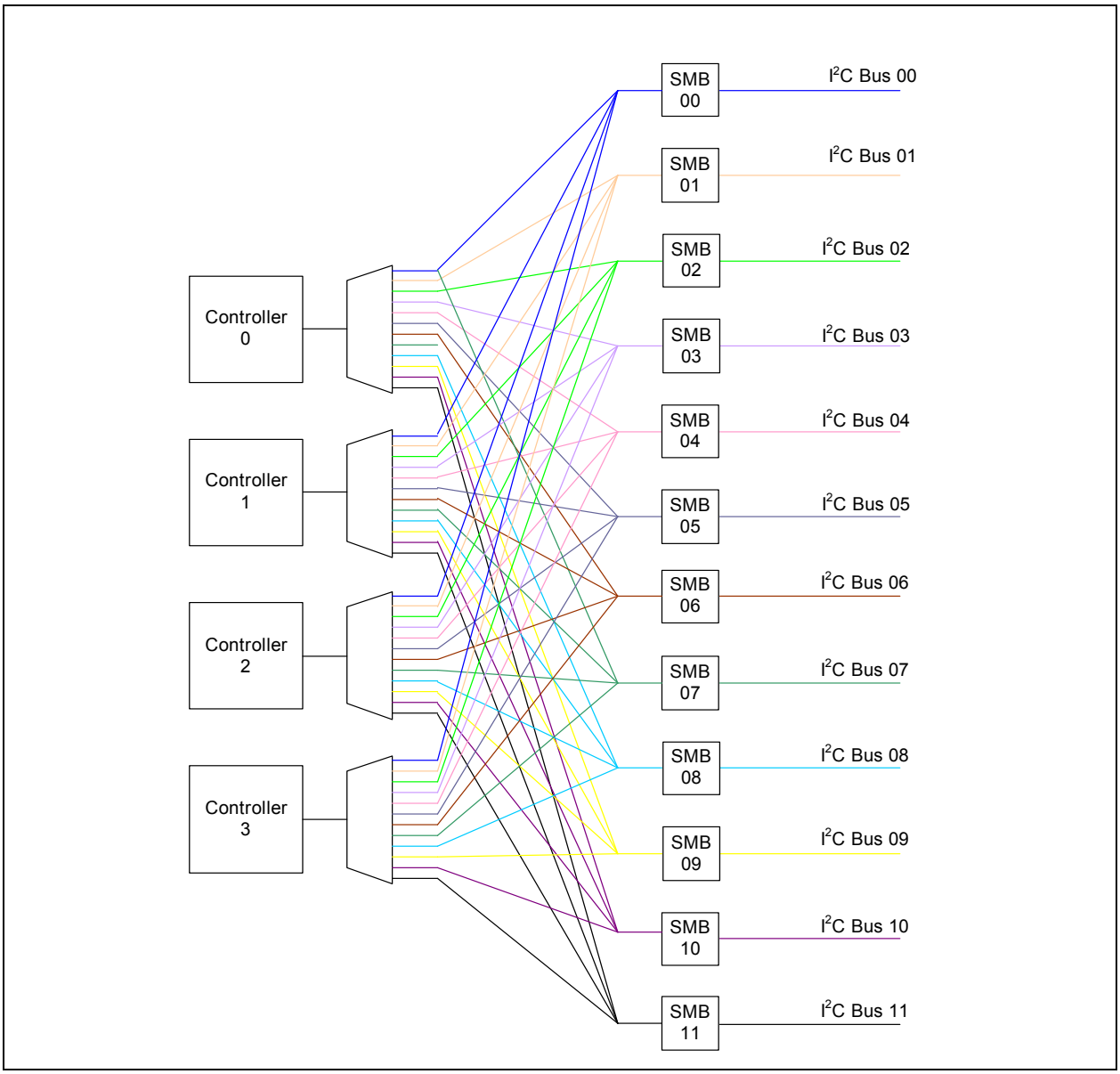
For I^2C port signal functions that are alternate functions of GPIO pins, the buffer type for these pins must be configured as open-drain outputs when the port is selected as an I^2C port.

For more information regarding the SMB-I2C Controller core see *Section 2.2,* "*Physical Interface*" in Ref[1].

TABLE 26-1: SMB-I2C PORT SELECTION

PORT_SEL[3:0]				Port
3	**2**	**1**	**0**	
0	0	0	0	SMB00 or I2C00
0	0	0	1	SMB01 or I2C01
0	0	1	0	SMB02 or I2C02
0	0	1	1	SMB03 or I2C03
0	1	0	0	SMB04 or I2C04
0	1	0	1	SMB05 or I2C05
0	1	1	0	SMB06 or I2C06
0	1	1	1	SMB07 or I2C07
1	0	0	0	SMB08 or I2C08
1	0	0	1	SMB09 or I2C09
1	0	1	0	SMB10 or I2C10
1	0	1	1	SB-TSI
1100b - 1111b				Reserved

FIGURE 26-2: SMB-I2C PORT CONNECTIVITY



26.12 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the SMB-I2C Controller Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

Registers for the SMB-I2C Controllers are listed in Reference[1].

27.0 GENERAL PURPOSE SERIAL PERIPHERAL INTERFACE

27.1 Overview

The General Purpose Serial Peripheral Interface (GP-SPI) may be used to communicate with various peripheral devices, e.g., EEPROMS, DACs, ADCs, that use a standard Serial Peripheral Interface.

.Characteristics of the GP-SPI Controller include:

- 8-bit serial data transmitted and received simultaneously over two data pins in Full Duplex mode with options to transmit and receive data serially on one data pin in Half Duplex (Bidirectional) mode.
- An internal programmable clock generator and clock polarity and phase controls allowing communication with various SPI peripherals with specific clocking requirements.
- SPI cycle completion that can be determined by status polling or interrupts.
- The ability to read data in on both SPDIN and SPDOUT in parallel. This allows this SPI Interface to support dual data rate read accesses for emerging double rate SPI flashes
- Support of back-to-back reads and writes without clock stretching, provided the host can read and write the data registers within one byte transaction time.

27.2 References

No references have been cited for this feature.

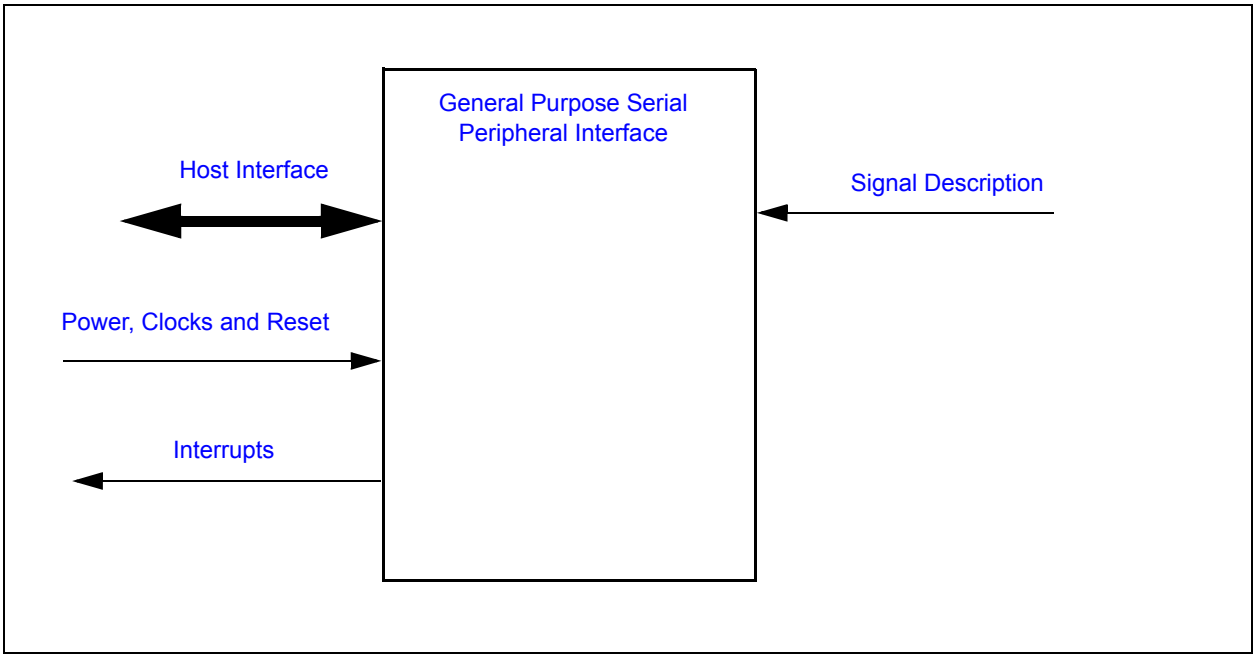
27.3 Terminology

No terminology for this block.

27.4 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

FIGURE 27-1: I/O DIAGRAM OF BLOCK



27.5 Signal Description

See the Pin Description chapter for the pins and the signal names associated with the following signals.

TABLE 27-1: EXTERNAL SIGNAL DESCRIPTION

Name	Direction	Description
SP_DIN	Input	Serial Data In pin
SP_DOUT	Input/Output	Serial Data Output pin. Switches to input when used in double-data-rate mode
SP_CLK	Output	SPI Clock output used to drive the SPCLK pin.
SP_CS#	Output	SPI chip select

TABLE 27-2: INTERNAL SIGNAL DESCRIPTION

Name	Direction	Description
SPI_TDMA_REQ	Output	DMA Request control for GP-SPI Controller Transmit Channel
SPI_RDMA_REQ	Output	DMA Request control for GP-SPI Controller Receive Channel

27.6 Host Interface

The registers defined for the General Purpose Serial Peripheral Interface are accessible by the various hosts as indicated in Section 27.12, "EC-Only/Runtime Registers".

27.7 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

27.7.1 POWER DOMAINS

Name	Description
VTR	The logic and registers implemented in this block are powered by this power well.

27.7.2 CLOCK INPUTS

Name	Description
48MHz	This is a clock source for the SPI clock generator.
2MHz	This is a clock source for the SPI clock generator. It is derived from the 48MHz clock domain.

27.7.3 RESETS

Name	Description
RESET_SYS	This signal resets all the registers and logic in this block to their default state.

27.8 Interrupts

This section defines the Interrupt Sources generated from this block.

TABLE 27-3: EC INTERRUPTS

Source	Description
TXBE_STS	Transmit buffer empty status (TXBE), in the SPI Status Register, sent as an interrupt request to the Interrupt Aggregator.
RXBF_STS	Receive buffer full status (RXBF), in the SPI Status Register, sent as an interrupt request to the Interrupt Aggregator.

These status bits are also connected respectively to the DMA Controller's SPI Controller TX and RX requests signals.

27.9 Low Power Modes

The GP-SPI Interface may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry.

27.10 Description

The Serial Peripheral Interface (SPI) block is a master SPI block used to communicate with external SPI devices. The SPI master is responsible for generating the SPI clock and is designed to operate in Full Duplex, Half Duplex, and Dual modes of operation. The clock source may be programmed to operated at various clock speeds. The data is transmitted serially via 8-bit transmit and receive shift registers. Communication with SPI peripherals that require transactions of varying lengths can be achieved with multiple 8-bit cycles.

This block has many configuration options: The data may be transmitted and received either MSbit or LSbit first; The SPI Clock Polarity may be either active high or active low; Data may be sampled or presented on either the rising of falling edge of the clock (referred to as the transmit clock phase); and the SPI_CLK SPDOUT frequency may be programmed to a range of values as illustrated in Table 27-4, "SPI_CLK Frequencies". In addition to these many programmable options, this feature has several status bits that may be enabled to notify the host that data is being transmitted or received.

27.10.1 INITIATING AN SPI TRANSACTION

All SPI transactions are initiated by a write to the TX_DATA register. No read or write operations can be initiated until the Transmit Buffer is Empty, which is indicated by a one in the TXBE status bit.

If the transaction is a write operation, the host writes the TX_DATA register with the value to be transmitted. Writing the TX_DATA register causes the TXBE status bit to be cleared, indicating that the value has been registered. If empty, the SPI Core loads this TX_DATA value into an 8-bit transmit shift register and begins shifting the data out. Loading the value into the shift register causes the TXBE status bit to be asserted, indicating to software that the next byte can be written to the TX_DATA register.

If the transaction is a read operation, the host initiates a write to the TX_DATA register in the same manner as the write operation. Unlike the transmit command, the host must clear the RXBF status bit by reading the RX_DATA register before writing the TX_DATA register. This time, the host will be required to poll the RXBF status bit to determine when the value in the RX_DATA register is valid.

Note 1: If the SPI interface is configured for Half Duplex mode, the host must still write a dummy byte to receive data.

2: Since RX and TX transactions are executed by the same sequence of transactions, data is always shifted into the RX_DATA register. Therefore, every write operation causes data to be latched into the RX_DATA register and the RXBF bit is set. This status bit should be cleared before initiating subsequent transactions. The host utilizing this SPI core to transmit SPI Data must discard the unwanted receive bytes.

3: The length and order of data sent to and received from a SPI peripheral varies between peripheral devices. The SPI must be properly configured and software-controlled to communicate with each device and determine whether SPIRD data is valid slave data.

The following diagrams show sample single byte and multi-byte SPI Transactions.

FIGURE 27-2: SINGLE BYTE SPI TX/RX TRANSACTIONS (FULL DUPLEX MODE)

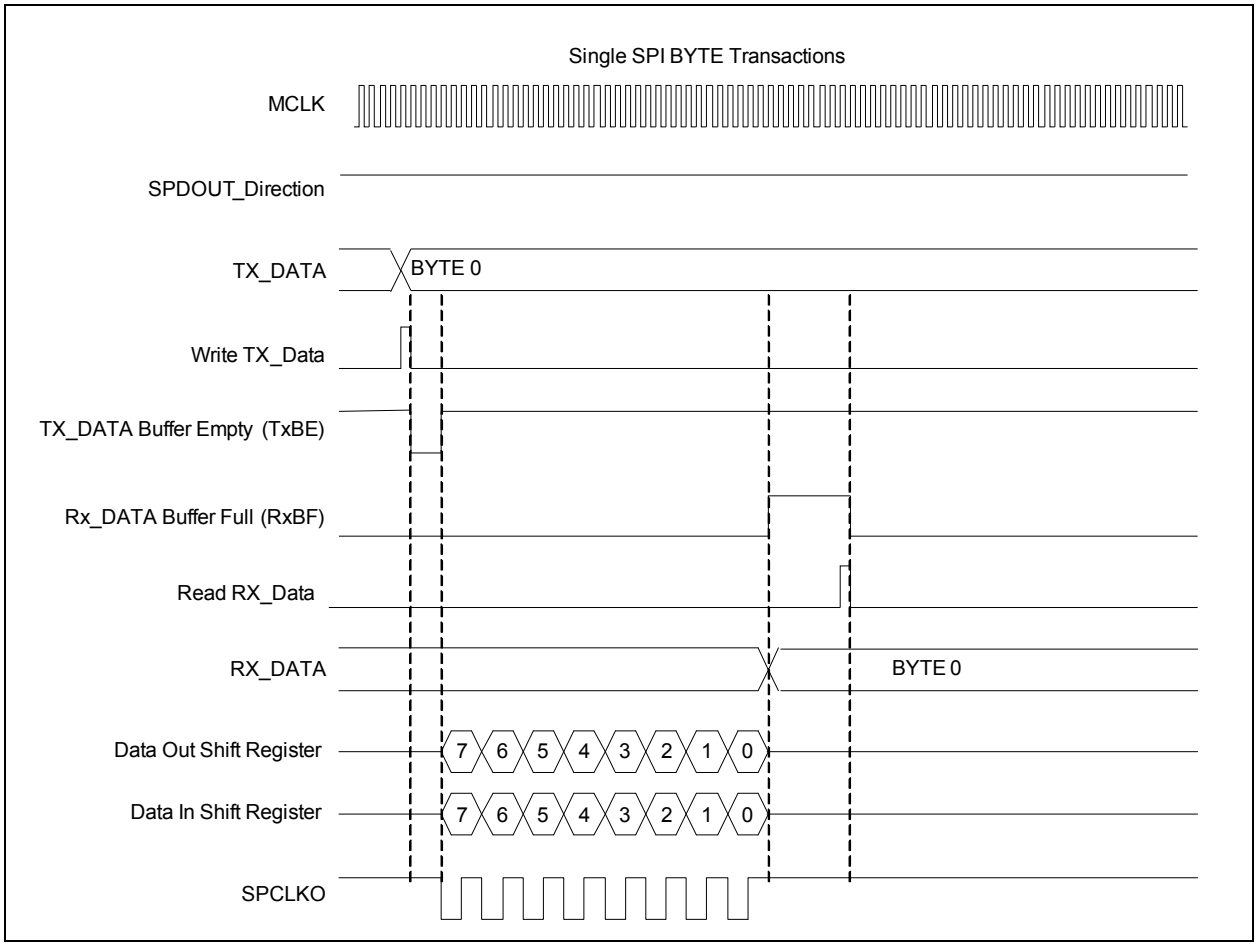
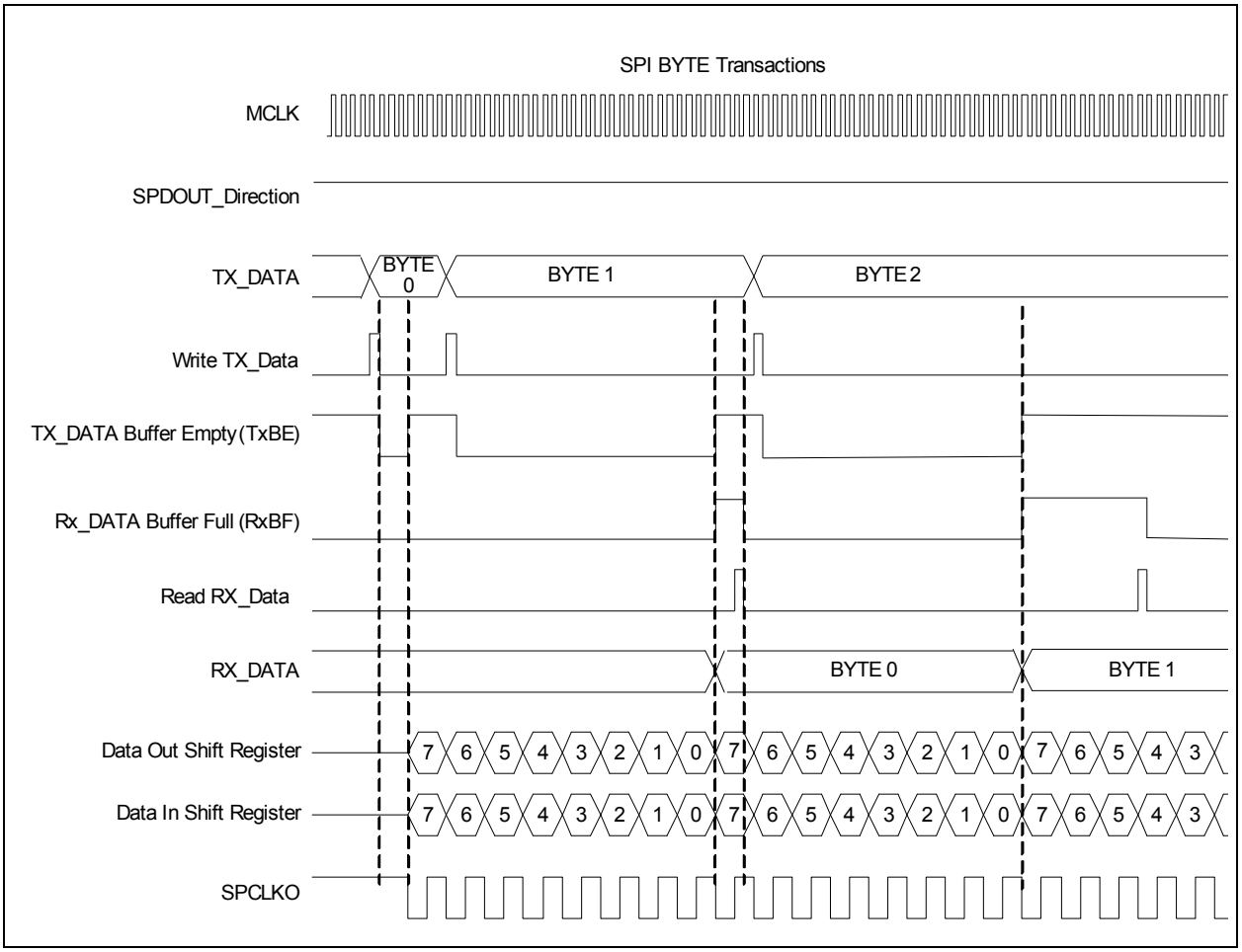


FIGURE 27-3: MULTI-BYTE SPI TX/RX TRANSACTIONS (FULL DUPLEX MODE)



The data may be configured to be transmitted MSB or LSB first. This is configured by the LSBF bit in the SPI Control Register. The transmit data is shifted out on the edge as selected by the TCLKPH bit in the SPI Clock Control Register. All received data can be sampled on a rising or falling SPI_CLK edge using the RCLKPH bit in the SPI Clock Control Register This clock setting must be identical to the clocking requirements of the current SPI slave.

Note:	Common peripheral devices require a chip select signal to be asserted during a transaction. Chip selects for SPI devices may be controlled by CEC1702 GPIO pins.

There are three types of transactions that can be implemented for transmitting and receiving the SPI data. They are Full Duplex, Half Duplex, and Dual Mode. These modes are define in Section 27.10.3, "Types of SPI Transactions".

27.10.2 DMA MODE

Transmit and receive operations can use a DMA channel. Note that only one DMA channel may be enabled at a time. Setting up the DMA Controller involves specifying the device (Flash GP-SPI), direction (transmit/receive), and the start and end addresses of the DMA buffers in the closely couple memory. Please refer to the DMA Controller chapter for register programming information.

SPI transmit / DMA write: the GP-SPI block's transmit empty (TxBE) status signal is used as a write request to the DMA controller, which then fetches a byte from the DMA transmit buffer and writes it to the GP-SPI's SPI TX Data Register (SPITD). As content of the latter is transferred to the internal Tx shift register from which data is shifted out onto the SPI

bus bit by bit, the Tx Empty signal is again asserted, triggering the DMA fetch-and-write cycle. The process continues until the end of the DMA buffer is reached - the DMA controller stops responding to an active Tx Empty until the buffer's address registers are reprogrammed.

SPI receive / DMA read: the AUTO_READ bit in the SPI Control Register must be set. The driver first writes (dummy data) to the SPI TX Data Register (SPITD) to initiate the toggling of the SPI clock, enabling data to be shifted in. After one byte is received, the Rx Full (RxBF) status signal, used as a read request to the DMA controller, is asserted. The DMA controller then reads the received byte from the GP-SPI's SPI RX Data Register (SPIRD) and stores it in the DMA receive buffer. With AUTO_READ set, this read clears both the RxBF and TxBE. Clearing TxBE causes (dummy) data from the SPI TX Data Register (SPITD) to be transferred to the internal shift register, mimicking the effect of the aforementioned write to the SPI TX Data Register (SPITD) by the driver. SPI clock is toggled again to shift in the second read byte. This process continues until the end of the DMA buffer is reached - the DMA controller stops responding to an active Tx Empty until the buffer's address registers are reprogrammed.

27.10.3 TYPES OF SPI TRANSACTIONS

The GP-SPI controller can be configured to operate in three modes: Full Duplex, Half Duplex, and Dual Mode.

27.10.3.1 Full Duplex

In Full Duplex Mode, serial data is transmitted and received simultaneously by the SPI master over the SPDOUT and SPDIN pins. To enable Full Duplex Mode clear SPDIN Select.

When a transaction is completed in the full-duplex mode, the RX_DATA shift register always contains received data (valid or not) from the last transaction.

27.10.3.2 Half Duplex

In Half Duplex Mode, serial data is transmitted and received sequentially over a single data line (referred to as the SPDOUT pin). To enable Half Duplex Mode set SPDIN Select to 01b. The direction of the SPDOUT signal is determined by the BIOEN bit.

- To transmit data in half duplex mode set the BIOEN bit before writing the TX_DATA register.
- To receive data in half duplex mode clear the BIOEN bit before writing the TX_DATA register with a dummy byte.

Note:	The Software driver must properly drive the BIOEN bit and store received data depending on the transaction format of the specific slave device.

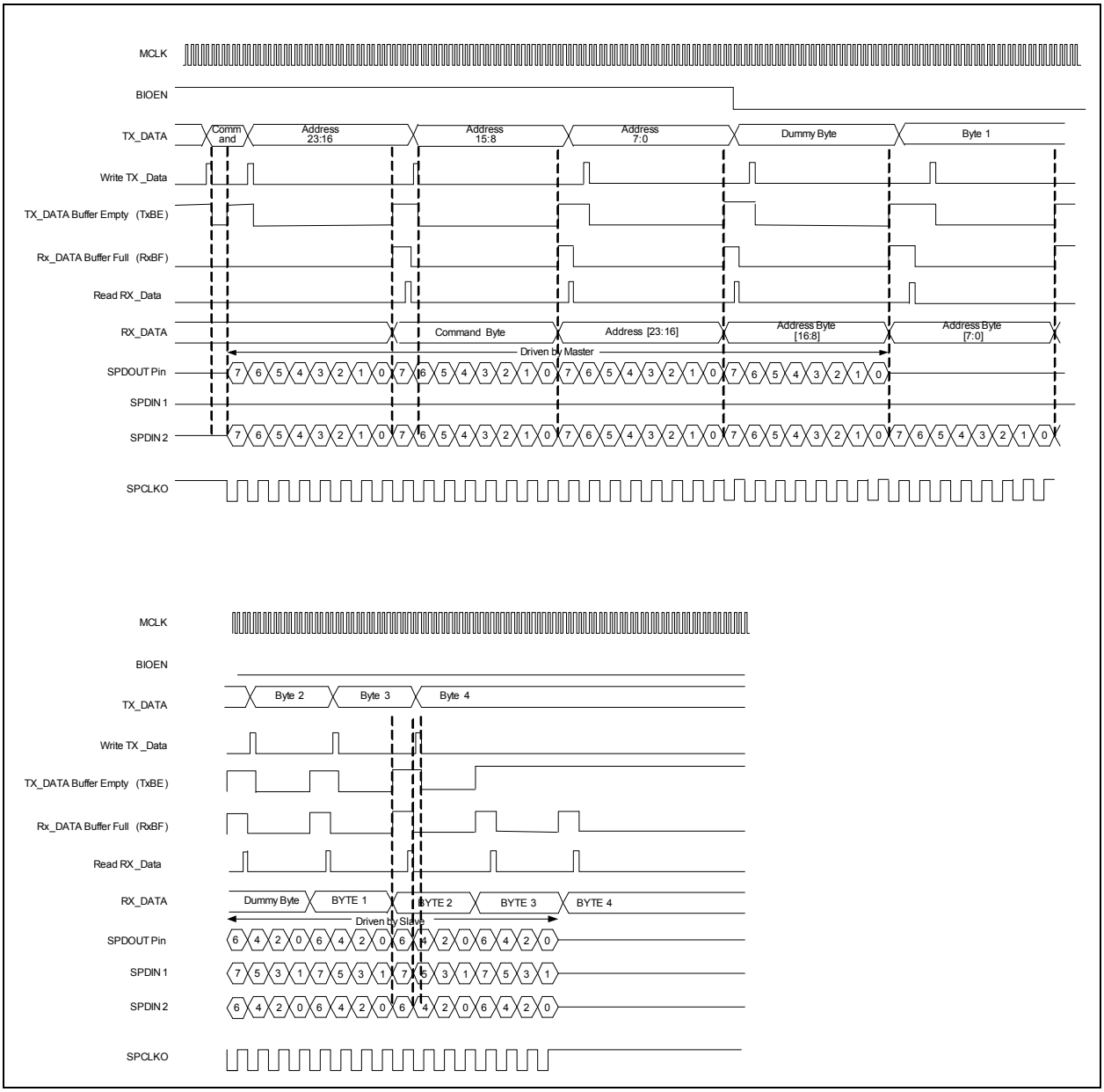
27.10.3.3 Dual Mode of Operation

In Dual Mode, serial data is transmitted sequentially from the SPDOUT pin and received in by the SPI master from the SPDOUT and SPDIN pins. This essentially doubles the received data rate and is often available in SPI Flash devices. To enable Dual Mode of operation the SPI core must be configured to receive data in path on the SPDIN1 and SPDIN2 inputs via SPDIN Select. The BIOEN bit determines if the SPI core is transmitting or receiving. The setting of this bit determines the direction of the SPDOUT signal. The SPDIN Select bits are configuration bits that remain static for the duration of a dual read command. The BIOEN bit must be toggled to indicate when the SPI core is transmitting and receiving.

- To transmit data in dual mode set the BIOEN bit before writing the TX_DATA register.
- To receive data in dual mode clear the BIOEN bit before writing the TX_DATA register with a dummy byte. The even bits (0,2,4,and 6) are received on the SPDOUT pin and the odd bits (1,3,5,and 7) are received on the SPDIN pin. The hardware assembles these received bits into a single byte and loads them into the RX_DATA register accordingly.

The following diagram illustrates a Dual Fast Read Command that is supported by some SPI Flash devices.

FIGURE 27-4: DUAL FAST READ FLASH COMMAND



Note:	When the SPI core is used for flash commands, like the Dual Read command, the host discards the bytes received during the command, address, and dummy byte portions of the transaction.

27.10.4 HOW BIOEN BIT CONTROLS DIRECTION OF SPDOUT BUFFER

When the SPI is configured for Half Duplex mode or Dual Mode the SPDOUT pin operates as a bi-directional signal. The BIOEN bit is used to determine the direction of the SPDOUT buffer when a byte is transmitted. Internally, the BIOEN bit is sampled to control the direction of the SPDOUT buffer when the TX_DATA value is loaded into the transmit shift register. The direction of the buffer is never changed while a byte is being transmitted.

Since the TX_DATA register may be written while a byte is being shifted out on the SPDOUT pin, the BIOEN bit does not directly control the direction of the SPDOUT buffer. An internal DIRECTION bit, which is a latched version of the BIOEN bit determines the direction of the SPDOUT buffer. The following list summarizes when the BIOEN bit is sampled.

- The DIRECTION bit is equal to the BIOEN bit when data is not being shifted out (i.e., SPI interface is idle).
- The hardware samples the BIOEN bit when it is shifting out the last bit of a byte to determine if the buffer needs to be turned around for the next byte.
- The BIOEN bit is also sampled any time the value in the TX_DATA register is loaded into the shift register to be transmitted.

If a TAR (Turn-around time) is required between transmitting and receiving bytes on the SPDOUT signal, software should allow all the bytes to be transmitted before changing the buffer to an input and then load the TX_DATA register to begin receiving bytes. If TAR greater than zero is required, software must wait for the transmission in one direction to complete before writing the TX_DATA register to start sending/receiving in the opposite direction. This allows the SPI block to operate the same as legacy Microchip SPI devices.

27.10.5 CONFIGURING THE SPI CLOCK GENERATOR

The SPI controller generates the SPI_CLK signal to the external SPI device. The frequency of the SPI_CLK signal is determined by one of two clock sources and the Preload value of the clock generator down counter. The clock generator toggles the SPI_CLK output every time the counter underflows, while data is being transmitted.

Note:	When the SPI interface is in the idle state and data is not being transmitted, the SPI_CLK signal stops in the inactive state as determined by the configuration bits.

The clock source to the down counter is determined by Bit CLKSRC. Either the main system clock or the 2MHz clock can be used to decrement the down counter in the clock generator logic.

The SPI_CLK frequency is determined by the following formula:

$$\text{SPI\_CLK\_FREQ} = \left(\left(\frac{1}{2} \times \text{REFERENCE\_CLOCK} \right) / \text{PRELOAD} \right)$$

The REFERENCE_CLOCK frequency is selected by CLKSRC in the SPI Clock Control Register and PRELOAD is the PRELOAD field of the SPI Clock Generator Register. The frequency can be either the 48MHz clock or a 2MHz clock. When the PRELOAD value is 0, the REFERENCE_CLOCK is always the 48MHz clock and the CLKSRC bit is ignored.

Sample SPI Clock frequencies are shown in the following table:

TABLE 27-4: SPI_CLK FREQUENCIES

Clock Source	Preload	SPI_CLK Frequency
Don't Care	0	48MHz
48MHz	1	24MHz
48MHz	2	12MHz (default)
48MHz	3	6MHz
48MHz	63	381KHz
2MHz	1	1MHz
2MHz	2	500KHz
2MHz	3	333KHz
2MHz	63	15.9KHz

27.10.6 CONFIGURING SPI MODE

In practice, there are four modes of operation that define when data should be latched. These four modes are the combinations of the SPI_CLK polarity and phase.

The output of the clock generator may be inverted to create an active high or active low clock pulse. This is used to determine the inactive state of the SPI_CLK signal and is used for determining the first edge for shifting the data. The polarity is selected by CLKPOL in the SPI Clock Control Register.

The phase of the clock is selected independently for receiving data and transmitting data. The receive phase is determine by RCLKPH and the transmit phase is determine by TCLKPH in the SPI Clock Control Register.

The following table summarizes the effect of CLKPOL, RCLKPH and TCLKPH.

TABLE 27-5: SPI DATA AND CLOCK BEHAVIOR

CLKPOL	RCLKPH	TCLKPH	Behavior
0	0	0	Inactive state is low. First edge is rising edge. Data is sampled on the rising edge. Data is transmitted on the falling edge. Data is valid before the first rising edge.
0	0	1	Inactive state is low. First edge is rising edge. Data is sampled on the rising edge. Data is transmitted on the rising edge.
0	1	0	Inactive state is low. First edge is rising edge. Data is sampled on the falling edge. Data is transmitted on the falling edge. Data is valid before the first rising edge.
0	1	1	Inactive state is low. First edge is rising edge. Data is sampled on the falling edge. Data is transmitted on the rising edge.
1	0	0	Inactive state is high. First edge is falling edge. Data is sampled on the falling edge. Data is transmitted on the rising edge. Data is valid before the first falling edge.
1	0	1	Inactive state is high. First edge is falling edge. Data is sampled on the falling edge. Data is transmitted on the falling edge.
1	1	0	Inactive state is high. First edge is falling edge. Data is sampled on the rising edge. Data is transmitted on the rising edge. Data is valid before the first falling edge.
1	1	1	Inactive state is high. First edge is falling edge. Data is sampled on the rising edge. Data is transmitted on the falling edge.

27.11 SPI Examples

27.11.1 FULL DUPLEX MODE TRANSFER EXAMPLES

27.11.1.1 Read Only

The slave device used in this example is a MAXIM MAX1080 10 bit, 8 channel ADC:

- The SPI block is activated by setting the enable bit in SPIAR - SPI Enable Register
- The SPIMODE bit is de-asserted '0' to enable the SPI interface in Full Duplex mode.
- The CLKPOL and TCLKPH bits are de-asserted '0', and RCLKPH is asserted '1' to match the clocking requirements of the slave device.
- The LSBF bit is de-asserted '0' to indicate that the slave expects data in MSB-first order.
- Assert CS# using a GPIO pin.
- Write a valid command word (as specified by the slave device) to the SPITD - SPI TX_Data Register with TXFE asserted '1'. The SPI master automatically clears the TXFE bit indicating the byte has been put in the TX buffer. If the shift register is empty the TX_DATA byte is loaded into the shift register and the SPI master reasserts the

TXFE bit. Once the data is in the shift register the SPI master begins shifting the data value onto the SPDOUT pin and drives the SPCLK pin. Data on the SPDIN pin is also sampled on each clock.

- Once the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX_DATA value, software must clear the RXBF status bit by reading the SPIRD - SPI RX_Data Register.
- A dummy 8 bit data value (any value) is written to the TX_DATA register. The SPI master automatically clears the TXFE bit, but does not begin shifting the dummy data value onto the SPDOUT pin. This byte will remain in the TX_DATA register until the TX shift register is empty.
- After 8 SPI_CLK pulses from the first transmit bytes:
  - The first SPI cycle is complete, RXBF bit is asserted '1', and the SPINT interrupt is asserted, if enabled. The data now contained in SPIRD - SPI RX_Data Register is invalid since the last cycle was initiated solely to transmit command data to the slave. This particular slave device drives '0' on the SPDIN pin to the master while it is accepting command data. This SPIRD data is ignored.
  - Once the first SPI cycle is completed, the SPI master takes the pending data in the TX_DATA register and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the SPCLK pin. Data on the SPDIN pin is also sampled on each clock.
- Once the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX_DATA value, software must clear the RXBF status bit by reading the SPIRD - SPI RX_Data Register.
- The final SPI cycle is initiated when another dummy 8 bit data value (any value) is written to the TX_DATA register. Note that this value may be another dummy value or it can be a new 8 bit command to be sent to the ADC. The new command will be transmitted while the final data from the last command is received simultaneously. This overlap allows ADC data to be read every 16 SPCLK cycles after the initial 24 clock cycle.The SPI master automatically clears the TXFE bit, but does not begin shifting the dummy data value onto the SPDOUT pin. This byte will remain in the TX_DATA register until the TX shift register is empty.
- After 8 SPI_CLK pulses, the second SPI cycle is complete:
  - The first SPI cycle is complete, RXBF bit is asserted '1', and the SPINT interrupt is asserted, if enabled. The data now contained in SPIRD - SPI RX_Data Register is the first half of a valid 16 bit ADC value. SPIRD is read and stored.
  - Once the second SPI cycle is completed, the SPI master takes the pending data in the TX_DATA register and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the data value onto the SPDOUT pin, and drives the SPCLK pin. Data on the SPDIN pin is also sampled on each clock.
- After 8 SPI_CLK pulses, the final SPI cycle is complete, TXBF is asserted '1', and the SPINT interrupt is asserted (if enabled). The data now contained in SPIRD - SPI RX_Data Register is the second half of a valid 16 bit ADC value. SPIRD is read and stored.
- If a command was overlapped with the received data in the final cycle, #CS should remain asserted and the SPI master will initiate another SPI cycle. If no new command was sent, #CS is released and the SPI is idle.

27.11.1.2 Read/Write

The slave device used in this example is a Fairchild NS25C640 FM25C640 64K Bit Serial EEPROM. The following subsections describe the read and write sequences.

Read

- The SPI block is activated by setting the enable bit in SPIAR - SPI Enable Register
- The SPIMODE bit is de-asserted '0' to enable the SPI interface in Full Duplex mode.
- The CLKPOL, TCLKPH and RCLKPH bits are de-asserted '0' to match the clocking requirements of the slave device.
- The LSBF bit is de-asserted '0' to indicate that the slave expects data in MSB-first order.
- Assert CS# low using a GPIO pin.
- Write a valid command word (as specified by the slave device) to the SPITD - SPI TX_Data Register with TXFE asserted '1'. The SPI master automatically clears the TXFE bit indicating the byte has been put in the TX buffer. If the shift register is empty the TX_DATA byte is loaded into the shift register and the SPI master reasserts the TXFE bit. Once the data is in the shift register the SPI master begins shifting the data value onto the SPDOUT pin and drives the SPI_CLK pin. Data on the SPDIN pin is also sampled on each clock.
- Once the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX_DATA value, software must clear the RXBF status bit by reading the SPIRD - SPI RX_Data Register.

- Next, EEPROM address A15-A8 is written to the TX_DATA register. The SPI master automatically clears the TXFE bit, but does not begin shifting the dummy data value onto the SPDOUT pin. This byte will remain in the TX_DATA register until the TX shift register is empty.
- After 8 SPI_CLK pulses from the first transmit byte (Command Byte transmitted):
  - The first SPI cycle is complete, RXBF bit is asserted '1', and the SPINT interrupt is asserted, if enabled. The data now contained in SPIRD - SPI RX_Data Register is invalid since the last cycle was initiated solely to transmit command data to the slave. This particular slave device tri-states the SPDIN pin to the master while it is accepting command data. This SPIRD data is ignored.

USER'S NOTE: External pull-up or pull-down is required on the SPDIN pin if it is tri-stated by the slave device.

  - Once the first SPI cycle is completed, the SPI master takes the pending data in the TX_DATA register (EEPROM address A15-A8) and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the SPI_CLK pin. Data on the SPDIN pin is also sampled on each clock. Note: The particular slave device ignores address A15-A13.
- Once the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX_DATA value, software must clear the RXBF status bit by reading the SPIRD - SPI RX_Data Register.
- Next, EEPROM address A7-A0 is written to the TX_DATA register. The SPI master automatically clears the TXFE bit, but does not begin shifting this data value onto the SPDOUT pin. This byte will remain in the TX_DATA register until the TX shift register is empty.
- After 8 SPI_CLK pulses from the second transmit byte (Address Byte (MSB) transmitted):
  - EEPROM address A15-A8 has been transmitted to the slave completing the second SPI cycle. Once again, the RXBF bit is asserted '1' and the SPINT interrupt is asserted, if enabled. The data now contained in SPIRD - SPI RX_Data Register is invalid since the last cycle was initiated solely to transmit address data to the slave.
  - Once the second SPI cycle is completed, the SPI master takes the pending data in the TX_DATA register (EEPROM address A7-A0) and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the SPCLK pin. Data on the SPDIN pin is also sampled on each clock.
- Once the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX_DATA value, software must clear the RXBF status bit by reading the SPIRD - SPI RX_Data Register.
- Next, a dummy byte is written to the TX_DATA register. The SPI master automatically clears the TXFE bit, but does not begin shifting this data value onto the SPDOUT pin. This byte will remain in the TX_DATA register until the TX shift register is empty.
- After 8 SPI_CLK pulses, the third SPI cycle is complete (Address Byte (LSB) transmitted):
  - EEPROM address A7-A0 has been transmitted to the slave completing the third SPI cycle. Once again, the RXBF bit is asserted '1' and the SPINT interrupt is asserted, if enabled. The data now contained in SPIRD - SPI RX_Data Register is invalid since the last cycle was initiated solely to transmit address data to the slave.
  - Once the third SPI cycle is completed, the SPI master takes the pending data in the TX_DATA register (dummy byte) and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the SPCLK pin. Data on the SPDIN pin is also sampled on each clock.
- Once the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX_DATA value, software must clear the RXBF status bit by reading the SPIRD - SPI RX_Data Register.
- If only one receive byte is required, the host would not write any more value to the TX_DATA register until this transaction completes. If more than one byte of data is to be received, another dummy byte would be written to the TX_DATA register (one dummy byte per receive byte is required). The SPI master automatically clears the TXFE bit when the TX_DATA register is written, but does not begin shifting this data value onto the SPDOUT pin. This byte will remain in the TX_DATA register until the TX shift register is empty.
- After 8 SPI_CLK pulses, the fourth SPI cycle is complete (First Data Byte received):
  - The dummy byte has been transmitted to the slave completing the fourth SPI cycle. Once again, the RXBF bit is asserted '1' and the SPINT interrupt is asserted, if enabled. Unlike the command and address phases, the data now contained in SPIRD - SPI RX_Data Register is the 8-bit EEPROM data since the last cycle was initiated to receive data from the slave.

- Once the fourth SPI cycle is completed, the SPI master takes the pending data in the TX_DATA register (if any) and loads it into the TX shift register. This process will be repeated until all the desired data is received.

- The host software will read and store the EEPROM data value in SPIRD - SPI RX_Data Register.
- If no more data needs to be received by the master, CS# is released and the SPI is idle. Otherwise, master continues reading the data by writing a dummy value to the TX_DATA register after every 8 SPI_CLK cycles.

Write

- The SPI block is activated by setting the enable bit in SPIAR - SPI Enable Register
- The SPIMODE bit is de-asserted '0' to enable the SPI interface in Full Duplex mode.
- The CLKPOL, TCLKPH and RCLKPH bits are de-asserted '0' to match the clocking requirements of the slave device.
- The LSBF bit is de-asserted '0' to indicate that the slave expects data in MSB-first order.
- Assert WR# high using a GPIO pin.
- Assert CS# low using a GPIO pin.
- Write a valid command word (as specified by the slave device) to the SPITD - SPI TX_Data Register with TXFE asserted '1'. The SPI master automatically clears the TXFE bit indicating the byte has been put in the TX buffer. If the shift register is empty the TX_DATA byte is loaded into the shift register and the SPI master reasserts the TXFE bit. Once the data is in the shift register the SPI master begins shifting the data value onto the SPDOUT pin and drives the SPI_CLK pin. Data on the SPDIN pin is also sampled on each clock.
- Once the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX_DATA value, software must clear the RXBF status bit by reading the SPIRD - SPI RX_Data Register.
- Next, EEPROM address A15-A8 is written to the TX_DATA register. The SPI master automatically clears the TXFE bit, but does not begin shifting the dummy data value onto the SPDOUT pin. This byte will remain in the TX_DATA register until the TX shift register is empty.
- After 8 SPI_CLK pulses from the first transmit byte (Command Byte transmitted):
  - The first SPI cycle is complete, RXBF bit is asserted '1', and the SPINT interrupt is asserted, if enabled. The data now contained in SPIRD - SPI RX_Data Register is invalid since the last cycle was initiated solely to transmit command data to the slave. This particular slave device tri-states the SPDIN pin to the master while it is accepting command data. This SPIRD data is ignored.

USER'S NOTE: External pull-up or pull-down is required on the SPDIN pin if it is tri-stated by the slave device.

  - Once the first SPI cycle is completed, the SPI master takes the pending data in the TX_DATA register (EEPROM address A15-A8) and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the SPI_CLK pin. Data on the SPDIN pin is also sampled on each clock. Note: The particular slave device ignores address A15-A13.
- Once the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX_DATA value, software must clear the RXBF status bit by reading the SPIRD - SPI RX_Data Register.
- Next, EEPROM address A7-A0 is written to the TX_DATA register. The SPI master automatically clears the TXFE bit, but does not begin shifting this data value onto the SPDOUT pin. This byte will remain in the TX_DATA register until the TX shift register is empty.
- After 8 SPI_CLK pulses from the second transmit byte (Address Byte (MSB) transmitted):
  - EEPROM address A15-A8 has been transmitted to the slave completing the second SPI cycle. Once again, the RXBF bit is asserted '1' and the SPINT interrupt is asserted, if enabled. The data now contained in SPIRD - SPI RX_Data Register is invalid since the last cycle was initiated solely to transmit address data to the slave.
  - Once the second SPI cycle is completed, the SPI master takes the pending data in the TX_DATA register (EEPROM address A7-A0) and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the SPCLK pin. Data on the SPDIN pin is also sampled on each clock.
- Once the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX_DATA value, software must clear the RXBF status bit by reading the SPIRD - SPI RX_Data Register.
- Next, a data byte (D7:D0) is written to the TX_DATA register. The SPI master automatically clears the TXFE bit, but does not begin shifting this data value onto the SPDOUT pin. This byte will remain in the TX_DATA register until the TX shift register is empty.

- After 8 SPI_CLK pulses, the third SPI cycle is complete (Address Byte (LSB) transmitted):
  - EEPROM address A7-A0 has been transmitted to the slave completing the third SPI cycle. Once again, the RXBF bit is asserted '1' and the SPINT interrupt is asserted, if enabled. The data now contained in SPIRD - SPI RX_Data Register is invalid since the last cycle was initiated solely to transmit address data to the slave.
  - Once the third SPI cycle is completed, the SPI master takes the pending data in the TX_DATA register (data byte D7:D0) and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the SPCLK pin. Data on the SPDIN pin is also sampled on each clock.
- Once the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX_DATA value, software must clear the RXBF status bit by reading the SPIRD - SPI RX_Data Register.
- If only one data byte is to be written, the host would not write any more values to the TX_DATA register until this transaction completes. If more than one byte of data is to be written, another data byte would be written to the TX_DATA register. The SPI master automatically clears the TXFE bit when the TX_DATA register is written, but does not begin shifting this data value onto the SPDOUT pin. This byte will remain in the TX_DATA register until the TX shift register is empty.
- After 8 SPI_CLK pulses, the fourth SPI cycle is complete (First Data Byte transmitted):
  - The data byte has been transmitted to the slave completing the fourth SPI cycle. Once again, the RXBF bit is asserted '1' and the SPINT interrupt is asserted, if enabled. Like the command and address phases, the data now contained in SPIRD - SPI RX_Data Register is invalid since the last cycle was initiated to transmit data to the slave.
  - Once the fourth SPI cycle is completed, the SPI master takes the pending data in the TX_DATA register (if any) and loads it into the TX shift register. This process will be repeated until all the desired data is transmitted.
- If no more data needs to be transmitted by the master, CS# and WR# are released and the SPI is idle.

27.11.2 HALF DUPLEX (BIDIRECTIONAL MODE) TRANSFER EXAMPLE

The slave device used in this example is a National LM74 12 bit (plus sign) temperature sensor.

- The SPI block is activated by setting the enable bit in SPIAR - SPI Enable Register
- The SPIMODE bit is asserted '1' to enable the SPI interface in Half Duplex mode.
- The CLKPOL, TCLKPH and RCLKPH bits are de-asserted '0' to match the clocking requirements of the slave device.
- The LSBF bit is de-asserted '0' to indicate that the slave expects data in MSB-first order.
- BIOEN is asserted '0' to indicate that the first data in the transaction is to be received from the slave.
- Assert CS# using a GPIO pin.

//Receive 16-bit Temperature Reading

- Write a dummy command byte (as specified by the slave device) to the SPITD - SPI TX_Data Register with TXFE asserted '1'. The SPI master automatically clears the TXFE bit indicating the byte has been put in the TX buffer. If the shift register is empty the TX_DATA byte is loaded into the shift register and the SPI master reasserts the TXFE bit. Once the data is in the shift register the SPI master begins shifting the data value onto the SPDOUT pin and drives the SPI_CLK pin. This data is lost because the output buffer is disabled. Data on the SPDIN pin is sampled on each clock.
- Once the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX_DATA value, software must clear the RXBF status bit by reading the SPIRD - SPI RX_Data Register.
- Next, another dummy byte is written to the TX_DATA register. The SPI master automatically clears the TXFE bit, but does not begin shifting the dummy data value onto the SPDOUT pin. This byte will remain in the TX_DATA register until the TX shift register is empty.
- After 8 SPI_CLK pulses from the first receive byte
  - The first SPI cycle is complete, RXBF bit is asserted '1', and the SPINT interrupt is asserted, if enabled. The data now contained in SPIRD - SPI RX_Data Register is the first half of the 16 bit word containing the temperature data.
  - Once the first SPI cycle is completed, the SPI master takes the pending data in the TX_DATA register (dummy byte 2) and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the SPI_CLK pin. Data on the SPDIN pin is also sampled on each clock.

- Once the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX_DATA value, software must clear the RXBF status bit by reading the SPIRD - SPI RX_Data Register.

//Transmit next reading command

- BIOEN is asserted '1' to indicate that data will now be driven by the master.
- Next, a command byte is written to the TX_DATA register. This value is the first half of a 16 bit command to be sent to temperature sensor peripheral. The SPI master automatically clears the TXFE bit, but does not begin shifting the command data value onto the SPDOUT pin. This byte will remain in the TX_DATA register until the TX shift register is empty. This data will be transmitted because the output buffer is enabled. Data on the SPDIN pin is sampled on each clock.
- After 8 SPI_CLK pulses from the second receive byte:
  - The second SPI cycle is complete, RXBF bit is asserted '1', and the SPINT interrupt is asserted, if enabled. The data now contained in SPIRD - SPI RX_Data Register is the second half of the 16 bit word containing the temperature data.
  - Once the first SPI cycle is completed, the SPI master takes the pending data in the TX_DATA register (command byte 1) and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the SPI_CLK pin. Data on the SPDIN pin is also sampled on each clock.
- Once the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX_DATA value, software must clear the RXBF status bit by reading the SPIRD - SPI RX_Data Register.
- Next, the second command byte is written to the TX_DATA register. The SPI master automatically clears the TXFE bit, but does not begin shifting the command data value onto the SPDOUT pin. This byte will remain in the TX_DATA register until the TX shift register is empty.
- After 8 SPI_CLK pulses from the first transmit byte:
  - The third SPI cycle is complete, RXBF bit is asserted '1', and the SPINT interrupt is asserted, if enabled. The data now contained in SPIRD - SPI RX_Data Register is invalid, since this command was used to transmit the first command byte to the SPI slave.
  - Once the first SPI cycle is completed, the SPI master takes the pending data in the TX_DATA register (command byte 2) and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the SPI_CLK pin. Data on the SPDIN pin is also sampled on each clock.
- Once the TXFE bit is asserted the SPI Master is ready to transmit or receive its next byte. Before writing the next TX_DATA value, software must clear the RXBF status bit by reading the SPIRD - SPI RX_Data Register.
- Since no more data needs to be transmitted, the host software will wait for the RXBF status bit to be asserted indicating the second command byte was transmitted successfully.
- CS# is de-asserted.

27.12 EC-Only/Runtime Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the General Purpose Serial Peripheral Interface Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 27-6: REGISTER SUMMARY

Offset	Register Name
0h	SPI Enable Register
4h	SPI Control Register
8h	SPI Status Register
Ch	SPI TX_Data Register
10h	SPI RX_Data Register
14h	SPI Clock Control Register
18h	SPI Clock Generator Register

27.12.1 SPI ENABLE REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:1	Reserved	R	-	-
0	ENABLE 1=Enabled. The device is fully operational 0=Disabled. Clocks are gated to conserve power and the SPDOUT and SPI_CLK signals are set to their inactive state	R/W	0h	RESET_SYS

27.12.2 SPI CONTROL REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:7	Reserved	R	-	-
6	CE SPI Chip Select Enable. 1=SPI_CS# output signal is asserted, i.e., driven to logic ‘0’ 0=SPI_CS# output signal is deasserted, i.e., driven to logic ‘1’	R/W	0h	RESET_SYS
5	AUTO_READ Auto Read Enable. 1=A read of the SPI RX_DATA Register will clear both the RXBF status bit and the TXBE status bit 0=A read of the SPI RX_DATA Register will clear the RXBF status bit. The TXBE status bit will not be modified	R/W	0h	RESET_SYS
4	SOFT_RESET Soft Reset is a self-clearing bit. Writing zero to this bit has no effect. Writing a one to this bit resets the entire SPI Interface, including all counters and registers back to their initial state.	R/W	0h	RESET_SYS
3:2	SPDIN_SELECT The SPDIN Select which SPI input signals are enabled when the BIOEN bit is configured as an input. 1xb=SPDIN1 and SPDIN2. Select this option for Dual Mode 01b=SPDIN2 only. Select this option for Half Duplex 00b=SPDIN1 only. Select this option for Full Duplex	R/W	0h	RESET_SYS

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
1	BIOEN Bidirectional Output Enable control. When the SPI is configured for Half Duplex mode or Dual Mode the SPDOUT pin operates as a bi-directional signal. The BIOEN bit is used by the internal DIRECTION bit to control the direction of the SPDOUT buffers. The direction of the buffer is never changed while a byte is being transmitted. 1=The SPDOUT_Direction signal configures the SPDOUT signal as an output. 0=The SPDOUT_Direction signal configures the SPDOUT signal as an input. See Section 27.10.4, "How BIOEN Bit Controls Direction of SPD-OUT Buffer" for details on the use of BIOEN.	R/W	1h	RESET_SYS
0	LSBF Least Significant Bit First 1=The data is transferred in LSB-first order. 0=The data is transferred in MSB-first order. (default)	R/W	0h	RESET_SYS

27.12.3 SPI STATUS REGISTER

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:3	Reserved	R	-	-
2	ACTIVE The ACTIVE bit goes high (Active) in the following conditions: While transmitting data While receiving data While a received byte is in the RX_Data buffer	R	0h	RESET_SYS
1	RXBF Receive Data Buffer Full status. When this bit is '1' the Rx_Data buffer is full. Reading the SPI RX_Data Register clears this bit. This signal may be used to generate a SPI_RX interrupt to the EC. 1=RX_Data buffer is full 0=RX_Data buffer is not full	R	0h	RESET_SYS
0	TXBE Transmit Data Buffer Empty status. When this bit is '1' the Tx_Data buffer is empty. Writing the SPI TX_Data Register clears this bit. This signal may be used to generate a SPI_TX interrupt to the EC. 1=TX_Data buffer is empty 0=TX_Data buffer is not empty	R	1h	RESET_SYS

27.12.4 SPI TX_DATA REGISTER

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:8	Reserved	R	-	-
7:0	TX_DATA A write to this register when the Tx_Data buffer is empty (TXBE in the SPI Status Register is '1') initiates a SPI transaction. The byte written to this register will be loaded into the shift register and the TXBE flag will be asserted. This indicates that the next byte can be written into the TX_DATA register. This byte will remain in the TX_-DATA register until the SPI core has finished shifting out the previ-ous byte. Once the shift register is empty, the hardware will load the pending byte into the shift register and once again assert the TxBE bit. The TX_DATA register must not be written when the TXBE bit is zero. Writing this register may overwrite the transmit data before it is loaded into the shift register.	R/W	0h	RESET_SYS

27.12.5 SPI RX_DATA REGISTER

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:8	Reserved	R	-	-
7:0	RX_DATA This register is used to read the value returned by the external SPI device. At the end of a byte transfer the RX_DATA register contains serial input data (valid or not) from the last transaction and the RXBF bit is set to one. This status bit indicates that the RX_DATA register has been loaded with a the serial input data. The RX_DATA register should not be read before the RXBF bit is set. The RX_DATA register must be read, clearing the RXBF status bit before writing the TX_DATA register. The data in the receive shift register is only loaded into the RX_DATA register when this bit is cleared. If a data byte is pending in the receive shift register the value will be loaded immediately into the RX_DATA register and the RXBF status flag will be asserted. Software should read the RX_-DATA register twice before starting a new transaction to make sure the RX_DATA buffer and shift register are both empty.	R/W	0h	RESET_SYS

27.12.6 SPI CLOCK CONTROL REGISTER

This register should not be changed during an active SPI transaction.

Offset	14h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:5	Reserved	R	-	-
4	CLKSRC Clock Source for the SPI Clock Generator. This bit should not be changed during a SPI transaction. When the field PRELOAD in the SPI Clock Generator Register is 0, this bit is ignored and the Clock Source is always the main system clock (the equivalent of setting this bit to '0'). 1=2MHz 0=48MHz	R/W	0h	RESET_SYS
3	Reserved	R	-	-
2	CLKPOL SPI Clock Polarity. 1=The SPI_CLK signal is high when the interface is idle and the first clock edge is a falling edge 0=The SPI_CLK is low when the interface is idle and the first clock edge is a rising edge	R/W	0h	RESET_SYS
1	RCLKPH Receive Clock Phase, the SPI_CLK edge on which the master will sample data. The receive clock phase is not affected by the SPI Clock Polarity. 1=Valid data on SPDIN signal is expected after the first SPI_CLK edge. This data is sampled on the second and following even SPI_CLK edges (i.e., sample data on falling edge) 0=Valid data is expected on the SPDIN signal on the first SPI_CLK edge. This data is sampled on the first and following odd SPI_-CLK edges (i.e., sample data on rising edge)	R/W	1h	RESET_SYS
0	TCLKPH Transmit Clock Phase, the SPCLK edge on which the master will clock data out. The transmit clock phase is not affected by the SPI Clock Polarity. 1=Valid data is clocked out on the first SPI_CLK edge on SPDOUT signal. The slave device should sample this data on the second and following even SPI_CLK edges (i.e., sample data on falling edge) 0=Valid data is clocked out on the SPDOUT signal prior to the first SPI_CLK edge. The slave device should sample this data on the first and following odd SPI_CLK edges (i.e., sample data on rising edge)	R/W	0h	RESET_SYS

27.12.7 SPI CLOCK GENERATOR REGISTER

Offset	18h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	Reserved	R	-	-
5:0	PRELOAD SPI Clock Generator Preload value.	R/W	2h	RESET_SYS

28.0 QUAD SPI MASTER CONTROLLER

28.1 Overview

The Quad SPI Master Controller may be used to communicate with various peripheral devices that use a Serial Peripheral Interface, such as EEPROMS, DACs and ADCs. The controller can be configured to support advanced SPI Flash devices with multi-phase access protocols. Data can be transfered in Half Duplex, Single Data Rate, Dual Data Rate and Quad Data Rate modes. In all modes and all SPI clock speeds, the controller supports back-to-back reads and writes without clock stretching if internal bandwidth permits.

28.2 References

No references have been cited for this feature.

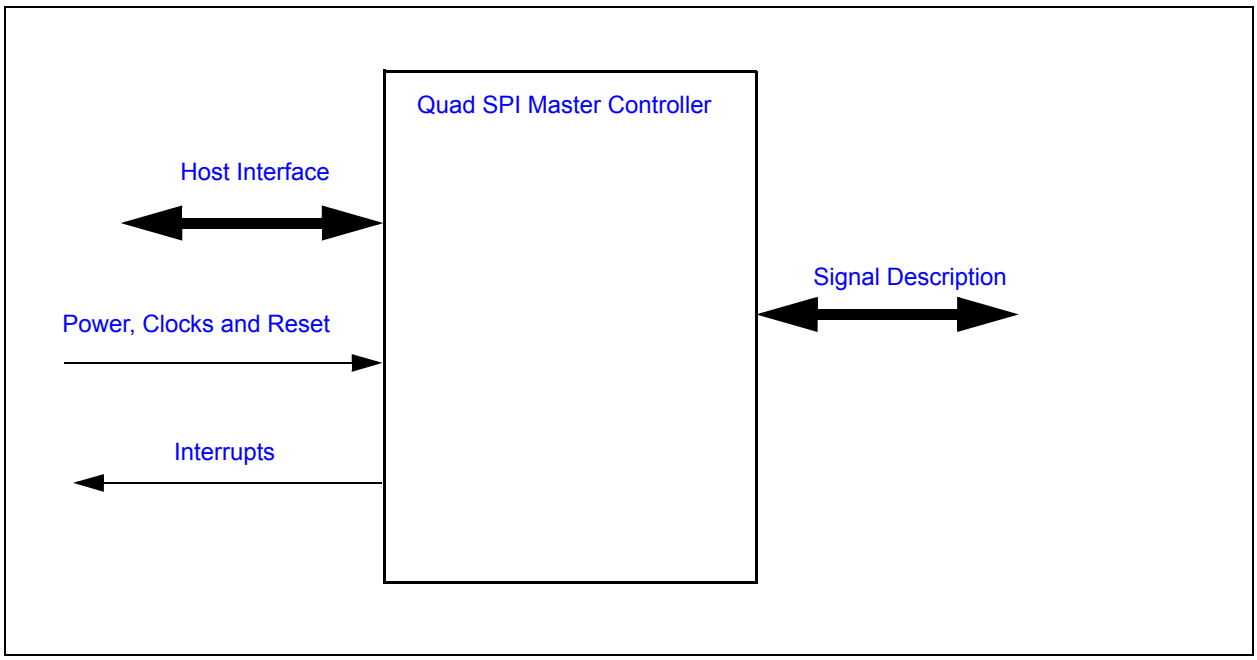
28.3 Terminology

No terminology for this block.

28.4 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

FIGURE 28-1: I/O DIAGRAM OF BLOCK



28.5 Signal Description

TABLE 28-1: EXTERNAL SIGNAL DESCRIPTION

Name	Direction	Description
SPI_CLK	Output	SPI Clock output used to drive the SPCLK pin.
SPI_CS#	Output	SPI chip select
SPI_IO0	Input/Output	SPI Data pin 0. Also used as SPI_MOSI, Master-Out/Slave-In when the interface is used in Single wire mode
SPI_IO1	Input/Output	SPI Data pin 1. Also used as SPI_MISO, Master-In/Slave-Out when the interface is used in Single wire mode

TABLE 28-1: EXTERNAL SIGNAL DESCRIPTION (CONTINUED)

Name	Direction	Description
SPI_IO2	Input/Output	SPI Data pin 2 when the SPI interface is used in Quad Mode. Also can be used by firmware as WP.
SPI_IO3	Input/Output	SPI Data pin 3 when the SPI interface is used in Quad Mode. Also can be used by firmware as HOLD.

28.6 Host Interface

The registers defined for the General Purpose Serial Peripheral Interface are accessible by the various hosts as indicated in Section 28.11, "EC Registers".

28.7 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

28.7.1 POWER

Name	Description
VTR	The logic and registers implemented in this block are powered by this power well.

28.7.2 CLOCKS

Name	Description
48MHz	This is a clock source for the SPI clock generator.

28.7.3 RESETS

Name	Description
RESET_SYS	This signal resets all the registers and logic in this block to their default state.QMSPI Status Register
RESET	This reset is generated if either the RESET_SYS is asserted or the SOFT_RESET is asserted.

28.8 Interrupts

This section defines the Interrupt Sources generated from this block.

Source	Description
QMSPI_INT	Interrupt generated by the Quad SPI Master Controller. Events that may cause the interrupt to be asserted are stored in the QMSPI Status Register.

28.9 Low Power Modes

The Quad SPI Master Controller is always in its lowest power state unless a transaction is in process. A transaction is in process between the time the START bit is written with a ‘1’ and the TRANSFER_DONE bit is set by hardware to ‘1’.

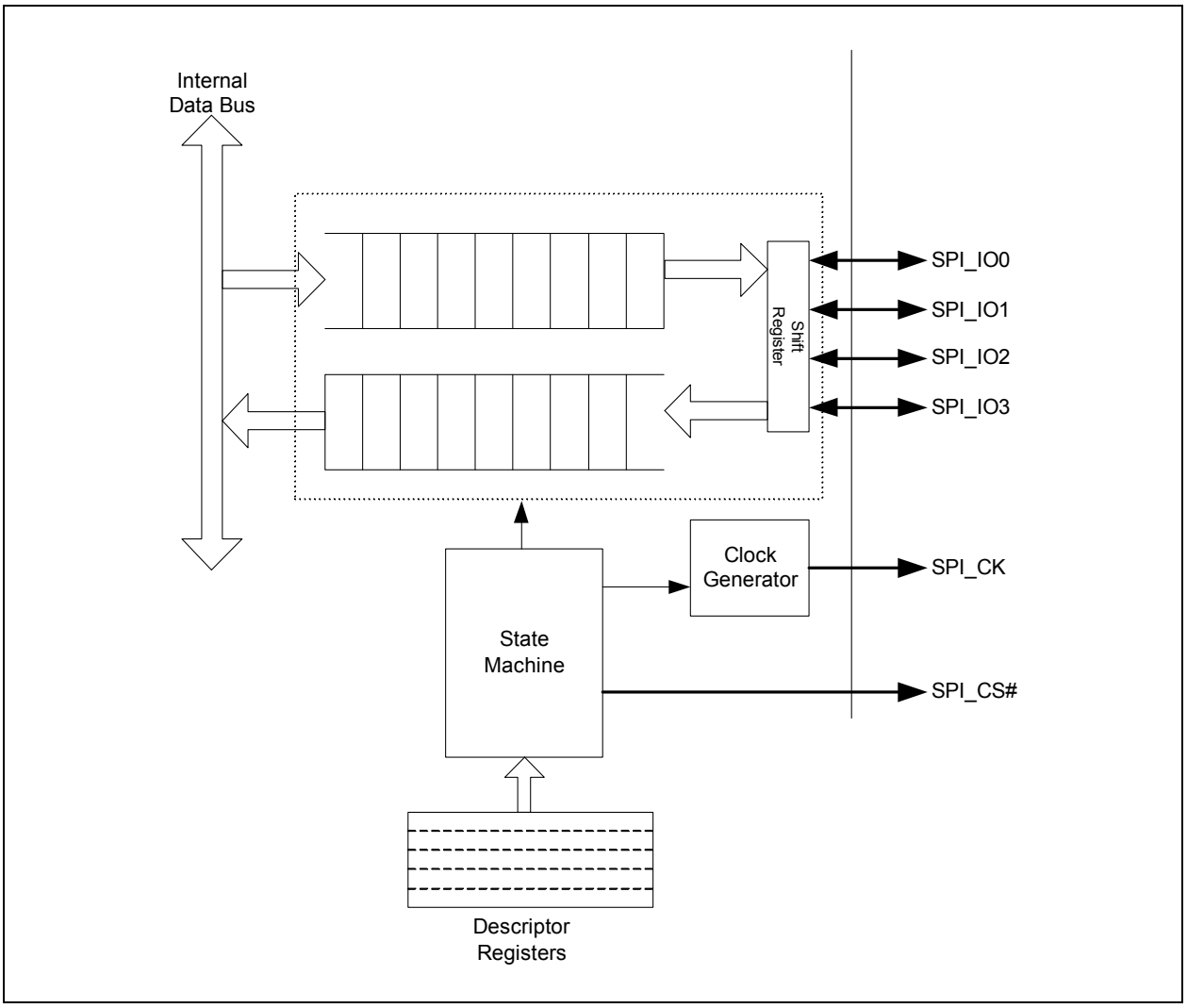
If the QMSPI SLEEP_ENABLE input is asserted, writes to the START bit are ignored and the Quad SPI Master Controller will remain in its lowest power state.

28.10 Description

- Support for multiple SPI pin configurations
  - Single wire half duplex
  - Two wire full duplex

  - Two wire double data rate
  - Four wire quad data rate
- Separate FIFO buffers for Receive and Transmit
  - 8 byte FIFO depth in each FIFO
  - Each FIFO can be 1 byte, 2 bytes or 4 bytes wide
- Support for all four SPI clock formats
- Programmable SPI Clock generator, with clock polarity and phase controls
- Separate DMA support for Receive and Transmit data transfers
- Configurable interrupts, for errors, individual bytes, or entire transactions
- Descriptor Mode, in which a set of five descriptor registers can configure the controller to autonomously perform multi-phase SPI data transfers
- Capable of wire speed transfers in all SPI modes and all configurable SPI clock rates (internal bus contention may cause clock stretching)

FIGURE 28-2: QUAD MASTER SPI BLOCK DIAGRAM



28.10.1 SPI CONFIGURATIONS MODES

- Half Duplex. All SPI data transfers take place on a single wire, SPI_IO0
- Full Duplex. This is the legacy SPI configuration, where all SPI data is transfered one bit at a time and data from the SPI Master to the SPI Slave takes place on SPI_MOSI (SPI_IO0) and at the same time data from the SPI Slave to the SPI Master takes place on SPI_MISO (SPI_IO1)
- Dual Data Rate. Data transfers between the SPI Master and the SPI Slave take place two bits at a time, using SPI_IO0 and SPI_IO1
- Quad Data Rate. Data transfers between the SPI Master and the SPI Slave take place four bits at a time, using all four SPI data wires, SPI_IO0, SPI_IO1, SPI_IO2 and SPI_IO3

28.10.2 SPI CONTROLLER MODES

- Manual. In this mode, firmware control all SPI data transfers byte at a time
- DMA. Firmware configures the SPI Master controller for characteristics like data width but the transfer of data between the FIFO buffers in the SPI controller and memory is controlled by the DMA controller. DMA transfers can take place from the Slave to the Master, from the Master to the Slave, or in both directions simultaneously
- Descriptor. Descriptor Mode extends the SPI Controller so that firmware can configure a multi-phase SPI transfer, in which each phase may have a different SPI bus width, a different direction, and a different length. For example, firmware can configure the controller so that a read from an advanced SPI Flash, which consists of a command phase, an address phase, a dummy cycle phase and the read phase, can take place as a single operation, with a single interrupt to firmware when the entire transfer is completed

28.10.3 SPI CLOCK

The SPI output clock is derived from the 48MHz, divided by a value programmed in the CLOCK_DIVIDE field of the QMSPI Mode Register. Sample frequencies are shown in the following table:

TABLE 28-2: EXAMPLE SPI FREQUENCIES

CLOCK_DIVIDE	SPI Clock Frequency
0	187.5 KHz
1	48 MHz
2	24 MHz
3	16 MHz
6	8 MHz
48	1 MHz
128	375 KHz
255	188.25 KHz

28.10.4 ERROR CONDITIONS

The Quad SPI Master Controller can detect some illegal configurations. When these errors are detected, an error is signaled via the PROGRAMMING_ERROR status bit. This bit is asserted when any of the following errors are detected:

- Both Receive and the Transmit transfers are enabled when the SPI Master Controller is configured for Dual Data Rate or Quad Data Rate
- Both Pull-up and Pull-down resistors are enabled on either the Receive data pins or the Transmit data pins
- The transfer length is programmed in bit mode, but the total number of bits is not a multiple of 2 (when the controller is configured for Dual Data Rate) or 4 (when the controller is configured for Quad Data Rate)
- Both the STOP bit and the START bits in the QMSPI Execute Register are set to ‘1’ simultaneously

28.11 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for each instance of the Quad SPI Master Controller Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 28-3: REGISTER SUMMARY

Offset	Register Name
0h	QMSPI Mode Register
4h	QMSPI Control Register
8h	QMSPI Execute Register
Ch	QMSPI Interface Control Register
10h	QMSPI Status Register
14h	QMSPI Buffer Count Status Register
18h	QMSPI Interrupt Enable Register
1Ch	QMSPI Buffer Count Trigger Register
20h	QMSPI Transmit Buffer Register
24h	QMSPI Receive Buffer Register
30h	QMSPI Description Buffer 0 Register
34h	QMSPI Description Buffer 1 Register
38h	QMSPI Description Buffer 2 Register
3Ch	QMSPI Description Buffer 3 Register
40h	QMSPI Description Buffer 4 Register

28.11.1 QMSPI MODE REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:24	Reserved	R	-	-
24:16	CLOCK_DIVIDE The SPI clock divide in number of system clocks. A value of 1 divides the master clock by 1, a value of 255 divides the master clock by 255. A value of 0 divides the master clock by 256. See Table 28-2, "Example SPI Frequencies" for examples.	R/W	0h	RESET
15:11	Reserved	R	-	-

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
10	CHPA_MISO If CPOL=1: 1=Data are captured on the rising edge of the SPI clock 0=Data are captured on the falling edge of the SPI clock If CPOL=0: 1=Data are captured on the falling edge of the SPI clock 0=Data are captured on the rising edge of the SPI clock Application Notes: Common SPI Mode configurations: Common SPI Modes require the CHPA_MISO and CHPA_MOSI programmed to the same value. E.g., - Mode 0: CPOL=0; CHPA_MISO=0; CHPA_MOSI=0 - Mode 3: CPOL=1; CHPA_MISO=1; CHPA_MOSI=1 Alternative SPI Mode configurations When configured for quad mode, applications operating at 48MHz may find it difficult to meet the minimum setup timing using the default Mode 0. It is recommended to configure the Master to sample and change data on the same edge when operating at 48MHz as shown in these examples. E.g, - Mode 0, Mode 3: CPOL=0; CHPA_MISO=1; CHPA_-MOSI=0 For applications operating at less than 48MHz, it is recommended to program as shown in these examples. E.g - Mode 0, Mode 3: CPOL=0, CHPA_MISO=0 and CHPA_-MOSI=0.	R/W	0h	RESET
9	CHPA_MOSI If CPOL=1: 1=Data changes on the falling edge of the SPI clock 0=Data changes on the rising edge of the SPI clock If CPOL=0: 1=Data changes on the rising edge of the SPI clock 0=Data changes on the falling edge of the SPI clock	R/W	0h	RESET
8	CPOL Polarity of the SPI clock line when there are no transactions in process. 1=SPI Clock starts High 0=SPI Clock starts Low	R/W	0h	RESET
7:2	Reserved	R	-	-
1	SOFT_RESET Writing this bit with a '1' will reset the Quad SPI block. It is self-clearing.	W	0h	RESET_SYS

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
0	ACTIVATE 1=Enabled. The block is fully operational 0=Disabled. Clocks are gated to conserve power and the output signals are set to their inactive state	R/W	0h	RESET

28.11.2 QMSPI CONTROL REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:17	TRANSFER_LENGTH The length of the SPI transfer. The count is in bytes or bits, depending on the value of TRANSFER_LENGTH_BITS. A value of '0' means an infinite length transfer.	R/W	0h	RESET
16	DESCRIPTION_BUFFER_ENABLE This enables the Description Buffers to be used. 1=Description Buffers in use. The first buffer is defined in DESCRIPTION_BUFFER_POINTER 0=Description Buffers disabled	R/W	0h	RESET
15:12	DESCRIPTION_BUFFER_POINTER This field selects the first buffer used if Description Buffers are enabled.	R/W	0h	RESET
11:10	TRANSFER_UNITS 3=TRANSFER_LENGTH defined in units of 16-byte segments 2=TRANSFER_LENGTH defined in units of 4-byte segments 1=TRANSFER_LENGTH defined in units of bytes 0=TRANSFER_LENGTH defined in units of bits	R/W	0h	RESET
9	CLOSE_TRANSFER_ENABLE This selects what action is taken at the end of a transfer. When the transaction closes, the Chip Select de-asserts, the SPI interface returns to IDLE and the DMA interface terminates When Description Buffers are in use this bit must be set only on the Last Buffer. 1=The transaction is terminated 0=The transaction is not terminated	R/W	1h	RESET
8:7	RX_DMA_ENABLE This bit enables DMA support for Receive Transfer. If enabled, DMA will be requested to empty the FIFO until either the interface reaches TRANSFER_LENGTH or the DMA sends a termination request. The size defined here must match DMA programmed access size. 1=DMA is enabled.and set to 1 Byte 2=DMA is enabled and set to 2 Bytes 3=DMA is enabled and set to 4 Bytes 0=DMA is disabled. All data in the Receive Buffer must be emptied by firmware	R/W	0h	RESET
6	RX_TRANSFER_ENABLE This bit enables the receive function of the SPI interface. 1=Receive is enabled. Data received from the SPI Slave is stored in the Receive Buffer 0=Receive is disabled	R/W	0h	RESET

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
5:4	TX_DMA_ENABLE This bit enables DMA support for Transmit Transfer. If enabled, DMA will be requested to fill the FIFO until either the interface reaches TRANSFER_LENGTH or the DMA sends a termination request. The size defined here must match DMA programmed access size. 1=DMA is enabled.and set to 1 Byte 2=DMA is enabled and set to 2 Bytes 3=DMA is enabled and set to 4 Bytes 0=DMA is disabled. All data in the Transmit Buffer must be emptied by firmware	R/W	0h	RESET
3:2	TX_TRANSFER_ENABLE This field bit selects the transmit function of the SPI interface. 3=Transmit Enabled in 1 Mode. The MOSI or IO Bus will send out only 1's. The Transmit Buffer will not be used 2=Transmit Enabled in 0 Mode. The MOSI or IO Bus will send out only 0's. The Transmit Buffer will not be used. 1=Transmit Enabled. Data will be fetched from the Transmit Buffer and sent out on the MOSI or IO Bus. 0=Transmit is Disabled. Not data is sent. This will cause the MOSI be to be undriven, or the IO bus to be undriven if Receive is also disabled.	R/W	0h	RESET
1:0	INTERFACE_MODE This field sets the transmission mode. If this field is set for Dual Mode or Quad Mode then either TX_TRANSFER_ENABLE or RX_TRANSFER_ENABLE **must** be 0. 3=Reserved 2=Quad Mode 1=Dual Mode 0=Single/Duplex Mode	R/W	0h	RESET

28.11.3 QMSPI EXECUTE REGISTER

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:3	Reserved	R	-	-
2	CLEAR_DATA_BUFFER Writing a ‘1’ to this bit will clear out the Transmit and Receive FIFOs. Any data stored in the FIFOs is discarded and all count fields are reset. Writing a ‘0’ to this bit has no effect. This bit is self-clearing.	W	0h	RESET
1	STOP Writing a ‘1’ to this bit will stop any transfer in progress at the next byte boundary. Writing a ‘0’ to this bit has no effect. This bit is self-clearing. This bit must not be set to ‘1’ if the field START in this register is set to ‘1’.	W	0h	RESET

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
0	START Writing a '1' to this bit will start the SPI transfer. Writing a '0' to this bit has no effect. This bit is self-clearing. This bit must not be set to '1' if the field STOP in this register is set to '1'.	W	1h	RESET

28.11.4 QMSPI INTERFACE CONTROL REGISTER

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:8	Reserved	R	-	-
7	PULLUP_ON_NOT_DRIVEN 1=Enable pull-up resistors on Transmit pins while the pins are not driven 0=No pull-up resistors enabled ion Transmit pins	R/W	0h	RESET
6	PULLDOWN_ON_NOT_DRIVEN 1=Enable pull-down resistors on Transmit pins while the pins are not driven 0=No pull-down resistors enabled ion Transmit pins	R/W	0h	RESET
5	PULLUP_ON_NOT_SELECTED 1=Enable pull-up resistors on Receive pins while the SPI Chip Select signal is not asserted 0=No pull-up resistors enabled on Receive pins	R/W	1h	RESET
4	PULLDOWN_ON_NOT_SELECTED 1=Enable pull-down resistors on Receive pins while the SPI Chip Select signal is not asserted 0=No pull-down resistors enabled on Receive pins	R/W	0h	RESET
3	HOLD_OUT_ENABLE 1=HOLD SPI Output Port is driven 0=HOLD SPI Output Port is not driven	R/W	0h	RESET
2	HOLD_OUT_VALUE This bit sets the value on the HOLD SPI Output Port if it is driven. 1=HOLD is driven to 1 0=HOLD is driven to 0	R/W	1h	RESET
1	WRITE_PROTECT_OUT_ENABLE 1=WRITE PROTECT SPI Output Port is driven 0=WRITE PROTECT SPI Output Port is not driven	R/W	0h	RESET

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
0	WRITE_PROTECT_OUT_VALUE This bit sets the value on the WRITE PROTECT SPI Output Port if it is driven. 1=WRITE PROTECT is driven to 1 0=WRITE PROTECT is driven to 0	R/W	1h	RESET

28.11.5 QMSPI STATUS REGISTER

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:28	Reserved	R	-	-
27:24	CURRENT_DESCRIPTION_BUFFER This field shows the Description Buffer currently active. This field has no meaning if Description Buffers are not enabled.	R	0h	RESET
23:17	Reserved	R	-	-
16	TRANSFER_ACTIVE 1=A transfer is currently executing 0=No transfer currently in progress	R	0h	RESET
15	RECEIVE_BUFFER_STALL 1=The SPI interface had been stalled due to a flow issue (an attempt by the interface to write to a full Receive Buffer) 0=No stalls occurred	R/WC	0h	RESET
14	RECEIVE_BUFFER_REQUEST This status is asserted if the Receive Buffer reaches a high water mark established by the RECEIVE_BUFFER_TRIGGER field. 1=RECEIVE_BUFFER_COUNT is greater than or equal to RECEIVE_BUFFER_TRIGGER 0=RECEIVE_BUFFER_COUNT is less than RECEIVE_BUFFER_TRIGGER	R/WC	0h	RESET
13	RECEIVE_BUFFER_EMPTY 1=The Receive Buffer is empty 0=The Receive Buffer is not empty	R	1h	RESET
12	RECEIVE_BUFFER_FULL 1=The Receive Buffer is full 0=The Receive Buffer is not full	R	0h	RESET
11	TRANSMIT_BUFFER_STALL 1=The SPI interface had been stalled due to a flow issue (an attempt by the interface to read from an empty Transmit Buffer) 0=No stalls occurred	R/WC	0h	RESET

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
10	TRANSMIT_BUFFER_REQUEST This status is asserted if the Transmit Buffer reaches a high water mark established by the TRANSMIT_BUFFER_TRIGGER field. 1=TRANSMIT_BUFFER_COUNT is less than or equal to TRANSMIT_BUFFER_TRIGGER 0=TRANSMIT_BUFFER_COUNT is greater than TRANSMIT_BUFFER_TRIGGER	R/WC	0h	RESET
9	TRANSMIT_BUFFER_EMPTY 1=The Transmit Buffer is empty 0=The Transmit Buffer is not empty	R	0h	RESET
8	TRANSMIT_BUFFER_FULL 1=The Transmit Buffer is full 0=The Transmit Buffer is not full	R	0h	RESET
7:5	Reserved	R	-	-
4	PROGRAMMING_ERROR This bit if a programming error is detected. Programming errors are listed in Section 28.10.4, "Error Conditions". 1=Programming Error detected 0=No programming error detected	R/WC	0h	RESET
3	RECEIVE_BUFFER_ERROR 1=Underflow error occurred (attempt to read from an empty Receive Buffer) 0=No underflow occurred	R/WC	0h	RESET
2	TRANSMIT_BUFFER_ERROR 1=Overflow error occurred (attempt to write to a full Transmit Buffer) 0=No overflow occurred	R/WC	0h	RESET
1	DMA_COMPLETE This field has no meaning if DMA is not enabled. This bit will be set to ‘1’ when the DMA controller asserts the DONE signal to the SPI controller. This occurs either when the SPI controller has closed the DMA transfer, or the DMA channel has completed its count. If both Transmit and Receive DMA transfers are active, then this bit will only assert after both have completed. If CLOSE_TRANSFER_ENABLE is enabled, DMA_COMPLETE and TRANSFER_COMPLETE will be asserted simultaneously. This status is not inhibited by the description buffers, so it can fire on all valid description buffers while operating in that mode. 1=DMA completed 0=DMA not completed	R/WC	0h	RESET

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
0	TRANSFER_COMPLETE In Manual Mode (neither DMA nor Description Buffers are enabled), this bit will be set to '1' when the transfer matches TRANSFER_LENGTH. If DMA Mode is enabled, this bit will be set to '1' when DMA_COMPLETE is set to '1'. In Description Buffer Mode, this bit will be set to '1' only when the Last Buffer completes its transfer. In all cases, this bit will be set to '1' if the STOP bit is set to '1' and the controller has completed the current 8 bits being copied. 1=Transfer completed 0=Transfer not complete	R/WC	0h	RESET

28.11.6 QMSPI BUFFER COUNT STATUS REGISTER

Offset	14h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	RECEIVE_BUFFER_COUNT This is a count of the number of bytes currently valid in the Receive Buffer.	R	0h	RESET
15:0	TRANSMIT_BUFFER_COUNT This is a count of the number of bytes currently valid in the Transmit Buffer.	R	0h	RESET

28.11.7 QMSPI INTERRUPT ENABLE REGISTER

Offset	18h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:15	Reserved	R	-	-
14	RECEIVE_BUFFER_REQUEST_ENABLE 1=Enable an interrupt if RECEIVE_BUFFER_REQUEST is asserted 0=Disable the interrupt	R/W	0h	RESET
13	RECEIVE_BUFFER_EMPTY_ENABLE 1=Enable an interrupt if RECEIVE_BUFFER_EMPTY is asserted 0=Disable the interrupt	R/W	1h	RESET
12	RECEIVE_BUFFER_FULL_ENABLE 1=Enable an interrupt if RECEIVE_BUFFER_FULL is asserted 0=Disable the interrupt	R/W	0h	RESET
11	Reserved	R	-	-

Offset	18h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
10	TRANSMIT_BUFFER_REQUEST_ENABLE 1=Enable an interrupt if TRANSMIT_BUFFER_REQUEST is asserted 0=Disable the interrupt	R/W	0h	RESET
9	TRANSMIT_BUFFER_EMPTY_ENABLE 1=Enable an interrupt if TRANSMIT_BUFFER_EMPTY is asserted 0=Disable the interrupt	R/W	0h	RESET
8	TRANSMIT_BUFFER_FULL_ENABLE 1=Enable an interrupt if TRANSMIT_BUFFER_FULL is asserted 0=Disable the interrupt	R/W	0h	RESET
7:5	Reserved	R	-	-
4	PROGRAMMING_ERROR_ENABLE 1=Enable an interrupt if PROGRAMMING_ERROR is asserted 0=Disable the interrupt	R/W	0h	RESET
3	RECEIVE_BUFFER_ERROR_ENABLE 1=Enable an interrupt if RECEIVE_BUFFER_ERROR is asserted 0=Disable the interrupt	R/W	0h	RESET
2	TRANSMIT_BUFFER_ERROR_ENABLE 1=Enable an interrupt if TRANSMIT_BUFFER_ERROR is asserted 0=Disable the interrupt	R/W	0h	RESET
1	DMA_COMPLETE_ENABLE 1=Enable an interrupt if DMA_COMPLETE is asserted 0=Disable the interrupt	R/W	0h	RESET
0	TRANSFER_COMPLETE_ENABLE 1=Enable an interrupt if TRANSFER_COMPLETE is asserted 0=Disable the interrupt	R/W	0h	RESET

28.11.8 QMSPI BUFFER COUNT TRIGGER REGISTER

Offset	1Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	RECEIVE_BUFFER_TRIGGER An interrupt is triggered if the RECEIVE_BUFFER_COUNT field is greater than or equal to this value. A value of ‘0’ disables the interrupt.	R/W	0h	RESET
15:0	TRANSMIT_BUFFER_TRIGGER An interrupt is triggered if the TRANSMIT_BUFFER_COUNT field is less than or equal to this value. A value of ‘0’ disables the interrupt.	R/W	0h	RESET

28.11.9 QMSPI TRANSMIT BUFFER REGISTER

Offset	20h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	TRANSMIT_BUFFER Writes to this register store data to be transmitted from the SPI Master to the external SPI Slave. Writes to this block will be written to the Transmit FIFO. A 1 Byte write fills 1 byte of the FIFO. A Word write fills 2 Bytes and a Doubleword write fills 4 bytes. The data must always be aligned to the bottom most byte (so 1 byte write is on bits [7:0] and Word write is on [15:0]). An overflow condition,TRANSMIT_BUFFER_ERROR, if a write to a full FIFO occurs. Write accesses to this register increment the TRANSMIT_BUFFER_COUNT field.	W	0h	RESET

28.11.10 QMSPI RECEIVE BUFFER REGISTER

Offset	24h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	RECEIVE_BUFFER Buffer that stores data from the external SPI Slave device to the SPI Master (this block), which is received over MISO or IO. Reads from this register will empty the Rx FIFO. A 1 Byte read will have valid data on bits [7:0] and a Word read will have data on bits [15:0]. It is possible to request more data than the FIFO has (underflow condition), but this will cause an error (Rx Buffer Error). Read accesses to this register decrement the RECEIVE_BUFFER_COUNT field.	R	0h	RESET

28.11.11 QMSPI DESCRIPTION BUFFER 0 REGISTER

Offset	30h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:17	TRANSFER_LENGTH The length of the SPI transfer. The count is in bytes or bits, depending on the value of TRANSFER_LENGTH_BITS. A value of ‘0’ means an infinite length transfer.	R/W	0h	RESET
16	DESCRIPTION_BUFFER_LAST If this bit is ‘1’ then this is the last Description Buffer in the chain. When the transfer described by this buffer completes the TRANSFER_COMPLETE status will be set to ‘1’. If this bit is ‘0’, then this is not the last buffer in use. When the transfer completes the next buffer will be activated, and no additional status will be asserted.	R/W	0h	RESET

Offset	30h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
15:12	DESCRIPTION_BUFFER_NEXT_POINTER This defines the next buffer to be used if Description Buffers are enabled and this is not the last buffer. This can point to the current buffer, creating an infinite loop.	R/W	0h	RESET
11:10	TRANSFER_UNITS 3=TRANSFER_LENGTH defined in units of 16-byte segments 2=TRANSFER_LENGTH defined in units of 4-byte segments 1=TRANSFER_LENGTH defined in units of bytes 0=TRANSFER_LENGTH defined in units of bits	R/W	0h	RESET
9	CLOSE_TRANFSER_ENABLE This selects what action is taken at the end of a transfer. This bit must be set only on the Last Buffer. 1=The transfer is terminated. The Chip Select de-asserts, the SPI interface returns to IDLE and the DMA interface completes the transfer. 0=The transfer is not closed. Chip Select remains asserted and the DMA interface and the SPI interface remain active	R/W	1h	RESET
8:7	RX_DMA_ENABLE This bit enables DMA support for Receive Transfer. If enabled, DMA will be requested to empty the FIFO until either the interface reaches TRANSFER_LENGTH or the DMA sends a termination request. The size defined here must match DMA programmed access size. 1=DMA is enabled.and set to 1 Byte 2=DMA is enabled and set to 2 Bytes 3=DMA is enabled and set to 4 Bytes 0=DMA is disabled. All data in the Receive Buffer must be emptied by firmware	R/W	0h	RESET
6	RX_TRANSFER_ENABLE This bit enables the receive function of the SPI interface. 1=Receive is enabled. Data received from the SPI Slave is stored in the Receive Buffer 0=Receive is disabled	R/W	0h	RESET
5:4	TX_DMA_ENABLE This bit enables DMA support for Transmit Transfer. If enabled, DMA will be requested to fill the FIFO until either the interface reaches TRANSFER_LENGTH or the DMA sends a termination request. The size defined here must match DMA programmed access size. 1=DMA is enabled.and set to 1 Byte 2=DMA is enabled and set to 2 Bytes 3=DMA is enabled and set to 4 Bytes 0=DMA is disabled. All data in the Transmit Buffer must be emptied by firmware	R/W	0h	RESET

Offset	30h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
3:2	TX_TRANSFER_ENABLE This field bit selects the transmit function of the SPI interface. 3=Transmit Enabled in 1 Mode. The MOSI or IO Bus will send out only 1's. The Transmit Buffer will not be used 2=Transmit Enabled in 0 Mode. The MOSI or IO Bus will send out only 0's. The Transmit Buffer will not be used. 1=Transmit Enabled. Data will be fetched from the Transmit Buffer and sent out on the MOSI or IO Bus. 0=Transmit is Disabled. No data is sent. This will cause the MOSI be to be undriven, or the IO bus to be undriven if Receive is also disabled.	R/W	0h	RESET
1:0	INTERFACE_MODE This field sets the transmission mode. If this field is set for Dual Mode or Quad Mode then either TX_TRANSFER_ENABLE or RX_TRANSFER_ENABLE **must** be 0. 3=Reserved 2=Quad Mode 1=Dual Mode 0=Single/Duplex Mode	R/W	0h	RESET

28.11.12 QMSPI DESCRIPTION BUFFER 1 REGISTER

The format for this register is the same as the format o the QMSPI Description Buffer 0 Register.

28.11.13 QMSPI DESCRIPTION BUFFER 2 REGISTER

The format for this register is the same as the format o the QMSPI Description Buffer 0 Register.

28.11.14 QMSPI DESCRIPTION BUFFER 3 REGISTER

The format for this register is the same as the format o the QMSPI Description Buffer 0 Register.

28.11.15 QMSPI DESCRIPTION BUFFER 4 REGISTER

The format for this register is the same as the format o the QMSPI Description Buffer 0 Register.

29.0 TRACE FIFO DEBUG PORT (TFDP)

29.1 Introduction

The TFDP serially transmits Embedded Controller (EC)-originated diagnostic vectors to an external debug trace system.

29.2 References

No references have been cited for this chapter.

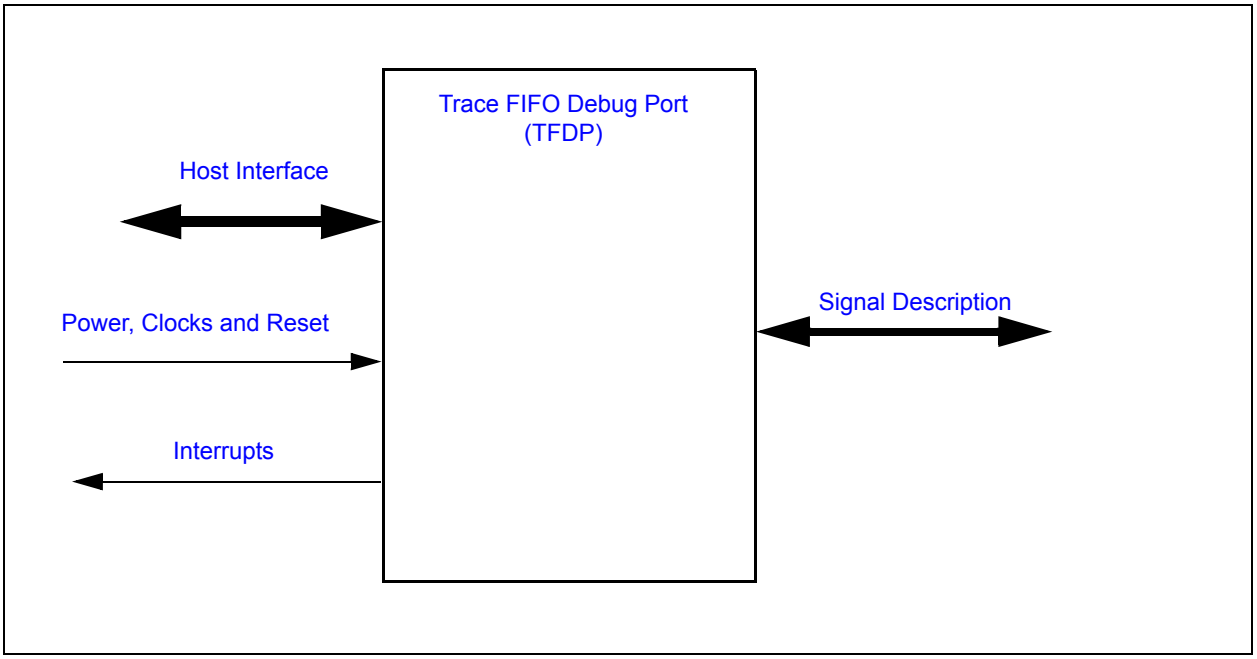
29.3 Terminology

There is no terminology defined for this chapter.

29.4 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

FIGURE 29-1: I/O DIAGRAM OF BLOCK



29.5 Signal Description

The Signal Description Table lists the signals that are typically routed to the pin interface.

TABLE 29-1: SIGNAL DESCRIPTION

Name	Direction	Description
TFDP Clk	Output	Derived from EC Bus Clock.
TFDP Data	Output	Serialized data shifted out by TFDP Clk.

29.6 Host Interface

The registers defined for the Trace FIFO Debug Port (TFDP) are accessible by the various hosts as indicated in Section 29.11, "EC-Only Registers".

29.7 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

29.7.1 POWER DOMAINS

Name	Description
VTR	The logic and registers implemented in this block are powered by this power well.

29.7.2 CLOCK INPUTS

Name	Description
48MHz	This is the main system clock.

29.7.3 RESETS

Name	Description
RESET_SYS	This signal resets all the registers and logic in this block to their default state.

29.8 Interrupts

There are no interrupts generated from this block.

29.9 Low Power Modes

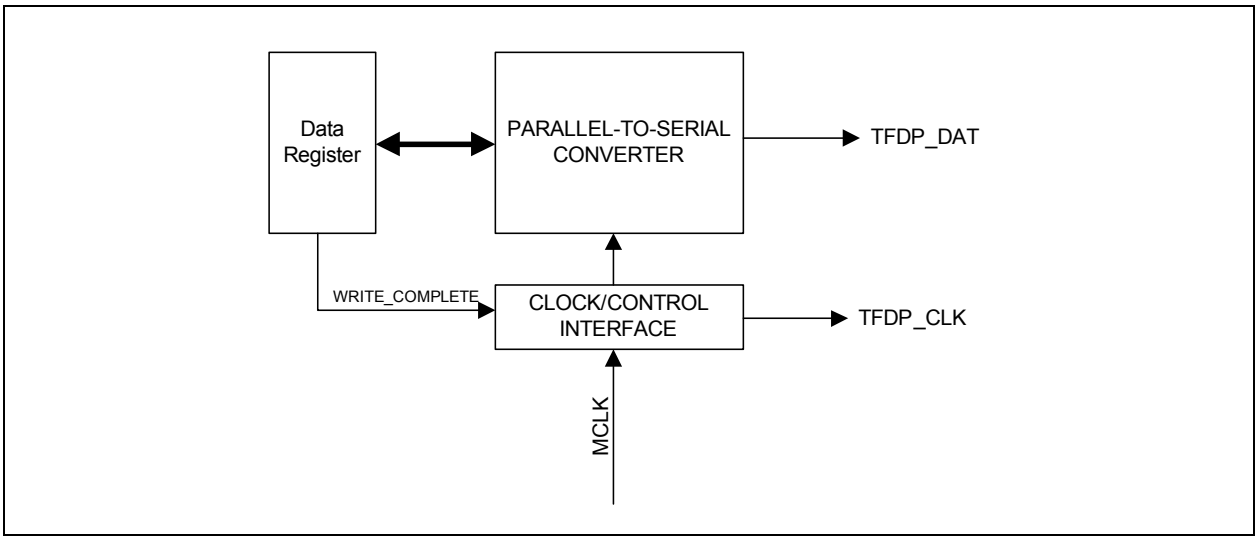
The Trace FIFO Debug Port (TFDP) may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry.

29.10 Description

The TFDP is a unidirectional (from processor to external world) two-wire serial, byte-oriented debug interface for use by processor firmware to transmit diagnostic information.

The TFDP consists of the Debug Data Register, Debug Control Register, a Parallel-to-Serial Converter, a Clock/Control Interface and a two-pin external interface (TFDP Clk, TFDP Data). See Figure 29-2.

FIGURE 29-2: BLOCK DIAGRAM OF TFDP DEBUG PORT

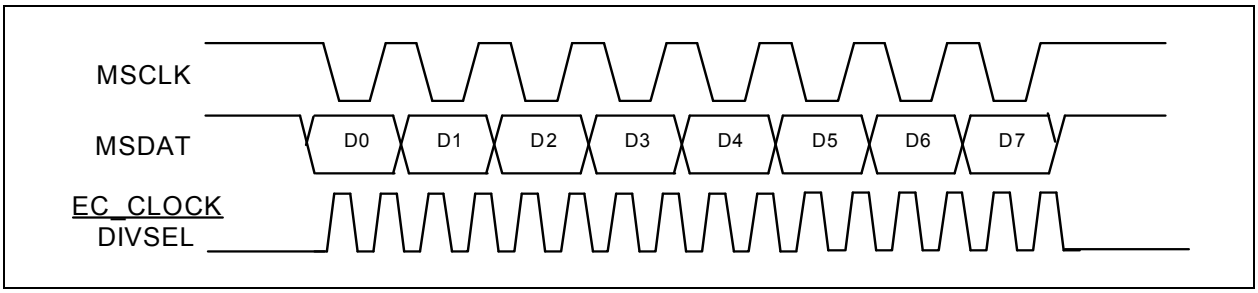


The firmware executing on the embedded controller writes to the Debug Data Register to initiate a transfer cycle (Figure 29.11). At first, data from the Debug Data Register is shifted into the LSB. Afterwards, it is transmitted at the rate of one byte per transfer cycle.

Data is transferred in one direction only from the Debug Data Register to the external interface. The data is shifted out at the clock edge. The clock edge is selected by the EDGE_SEL bit in the Debug Control Register. After being shifted out, valid data will be presented at the opposite edge of the TFDP_CLK. For example, when the EDGE_SEL bit is '0' (default), valid data will be presented on the falling edge of the TFDP_CLK. The Setup Time (to the falling edge of TFDP_CLK) is 10 ns, minimum. The Hold Time is 1 ns, minimum.

When the Serial Debug Port is inactive, the TFDP_CLK and TFDP_DAT outputs are '1.' The EC Bus Clock clock input is the transfer clock.

FIGURE 29-3: DATA TRANSFER



29.11 EC-Only Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for the Trace FIFO Debug Port (TFDP) Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 29-2: REGISTER SUMMARY

Offset	Register Name
00h	Debug Data Register
04h	Debug Control Register

29.11.1 DEBUG DATA REGISTER

The Debut Data Register is Read/Write. It always returns the last data written by the TFDP or the power-on default ‘00h’.

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:0	DATA Debug data to be shifted out on the TFDP Debug port. While data is being shifted out, the Host Interface will ‘hold-off’ additional writes to the data register until the transfer is complete.	R/W	00h	RESET_SYS

29.11.2 DEBUG CONTROL REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7	Reserved	R	-	-
6:4	IP_DELAY Inter-packet Delay. The delay is in terms of TFDP Debug output clocks. A value of 0 provides a 1 clock inter-packet period, while a value of 7 provides 8 clocks between packets:	R/W	000b	RESET_SYS
3:2	DIVSEL Clock Divider Select. The TFDP Debug output clock is determined by this field, according to Table 29-3, "TFDP Debug Clocking":	R/W	00b	RESET_SYS
1	EDGE_SEL 1=Data is shifted out on the falling edge of the debug clock 0=Data is shifted out on the rising edge of the debug clock (Default)	R/W	0b	RESET_SYS
0	EN Enable. 1=Clock enabled 0=Clock is disabled (Default)	R/W	0b	RESET_SYS

TABLE 29-3: TFDP DEBUG CLOCKING

divsel	TFDP Debug Clock
00	24 MHz
01	12 MHz
10	6 MHz
11	Reserved

30.0 VBAT-POWERED CONTROL INTERFACE

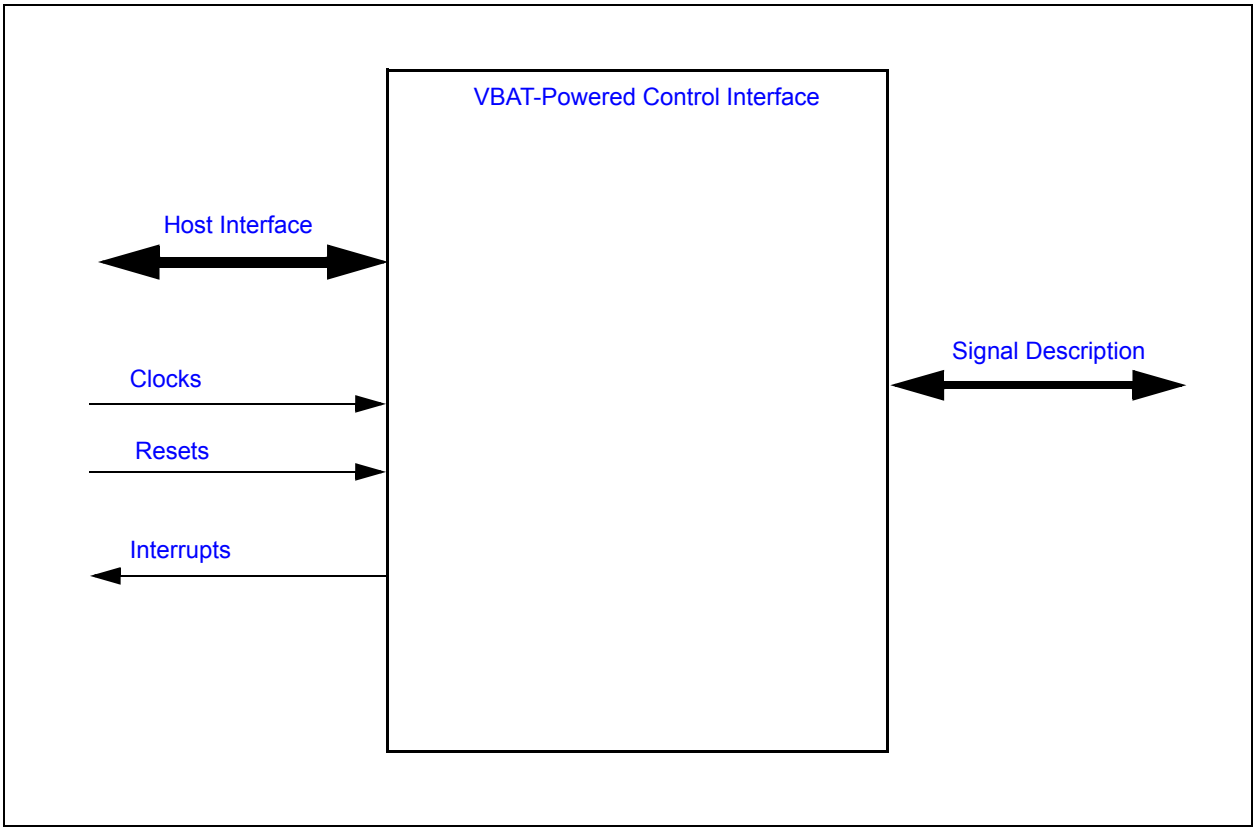
30.1 General Description

The VBAT-Powered Control Interface (VCI) has VBAT-powered combinational logic and input and output signal pins. The block interfaces with the Real Time Clock as well as the Week Alarm.

30.2 Interface

This block's connections are entirely internal to the chip.

FIGURE 30-1: I/O DIAGRAM OF BLOCK



30.3 Signal Description

TABLE 30-1: EXTERNAL SIGNAL DESCRIPTION

Name	Direction	Description
VCI_IN[6:0]	INPUT	Active-low inputs that can cause wakeup or interrupt events. **Note:** The VCI IP supports up to seven VCI_IN inputs. These inputs are generically referred to as VCI_INx. Input signals not routed to pins or balls on the package are connected to VBAT.
VCI_OUT	OUTPUT	Output status driven by this block.

TABLE 30-2: INTERNAL SIGNAL DESCRIPTION

Name	Direction	Description
Week_Alarm	INPUT	Signal from the Week Timer block. The alarm is asserted by the timer when the Week_Alarm Power-Up Output is asserted
RTC_Alarm	INPUT	Signal from the Real Time Clock block. The alarm is asserted by the RTC when the RTC_ALRM signal is asserted.
VTR_PWRGD	INPUT	Status signal for the state of the VTR power rail. This signal is high if the power rail is on, and low if the power rail is off.

30.4 Host Interface

The registers defined for the VBAT-Powered Control Interface are accessible only by the EC.

30.5 Power, Clocks and Resets

This section defines the Power, Clock, and Reset parameters of the block.

30.5.1 POWER DOMAINS

Name	Description
VBAT	This power well sources all of the internal registers and logic in this block.
VTR	This power well sources only bus communication. The block continues to operate internally while this rail is down.

30.5.2 CLOCKS

This block does not require clocks.

30.5.3 RESETS

Name	Description
RESET_VBAT	This reset signal is used reset all of the registers and logic in this block.
RESET_SYS	This reset signal is used to inhibit the bus communication logic, and isolates this block from VTR powered circuitry on-chip. Otherwise it has no effect on the internal state.

30.6 Interrupts

Source	Description
VCI_IN[6:0]	These interrupts are routed to the Interrupt Controller They are only asserted when both VBAT and VTR are powered. Edge detection and assertion level for the interrupt are configured in the GPIO Pin Control Registers for the GPIOs that shares pins with VCI_INx# inputs. The interrupts are equivalent to the GPIO interrupts for the GPIOs that share the pins, but appear on different registers in the Interrupt Aggregator.

30.7 Low Power Modes

The VBAT-powered Control Interface has no low-power modes. It runs continuously while the VBAT well is powered.

30.8 General Description

The VBAT-Powered Control Interface (VCI) is used to drive the VCI_OUT pin. The output pin can be controlled either by VBAT-powered inputs, or by firmware when the VTR is active and the EC is powered and running. When the VCI_OUT pin is controlled by hardware, either because VTR is inactive or because the VCI block is configured for hardware control, the VCI_OUT pin can be asserted by a number of inputs:

- When one or more of the VCI_INx# pins are asserted. By default, the VCI_INx# pins are active low, but firmware can switch each input individually to an active-high input. See Section 30.8.1, "Input Polarity".
- When the Week Alarm from the Week Alarm Interface is asserted
- When the RTC Alarm from the Real Time Clock is asserted

Firmware can configure which of the hardware pin inputs contribute to the VCI_OUT output by setting the enable bits in the VCI Input Enable Register. Even if the input pins are not configured to affect VCI_OUT, firmware can monitor their current state through the status bits in the VCI Register. Firmware can also enable EC interrupts from the state of the input pins.

Each of the VCI_INx# pins can be configured for additional properties.

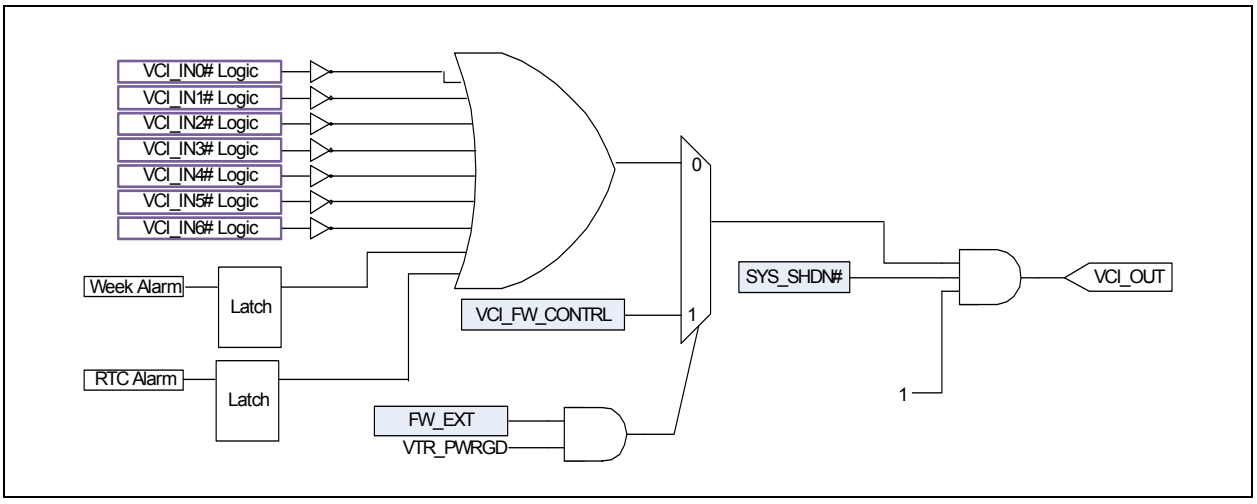
- By default, each of the VCI_INx# pins have an input glitch filter. All glitch filters can be disabled by the FILTERS_BYPASS bit in the VCI Register
- Assertions of each of the VCI_INx# pins can optionally be latched, so hardware can maintain the assertion of a VCI_INx# even after the physical pin is de-asserted, or so that firmware can determine which of the VCI_INx# inputs contributed to VCI_OUT assertion. See the Latch Enable Register and the Latch Resets Register.
- Rising edges and falling edges on the VCI_INx# pins are latched, so firmware can detect transitions on the VCI_INx# pins even if the transitions occurred while EC power was not available. See Section 30.8.2, "Edge Event Status".

If none of the additional properties are required, firmware can disable a VCI_INx# pin completely, by clearing both the corresponding bit in the VCI Input Enable Register and the corresponding bit in the VCI Buffer Enable Register. When both bits are '0', the input is disabled and will not be a drain on the VBAT power rail.

When VTR power is present and the EC is operating, firmware can configure the VCI_OUT pin to operate as a general-purpose output pin. The VCI_OUT pin is firmware-controlled when the FW_EXT bit in the VCI Register is '1'. When firmware is controlling the output, the state of VCI_OUT is defined by the VCI_FW_CNTRL bit in the same register. When VTR is not present (the VTR_PWRGD input is low), the VCI_OUT pin is also determined by the hardware circuit.

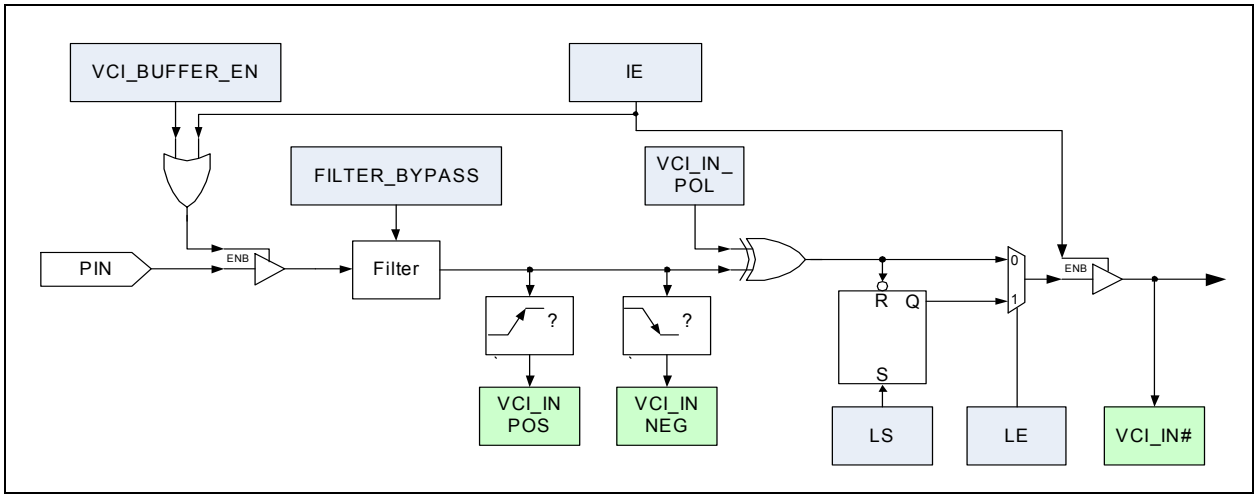
The following figures illustrate the VBAT-Power Control Interface logic:

FIGURE 30-2: VBAT-POWERED CONTROL INTERFACE BLOCK DIAGRAM



The VCI_INx# Logic in the block diagram is illustrated in the following figure:

FIGURE 30-3: VBAT-POWERED CONTROL INTERFACE BLOCK DIAGRAM



30.8.1 INPUT POLARITY

The VCI_INx# pins have an optional polarity inversion. The inversion takes place after any input filtering and before the VCI_INx# signals are latched in the VCI_INx# status bits in the VCI Register. Edge detection occurs before the polarity inversion. The inversion is controlled by battery-backed configuration bits in the VCI Polarity Register.

30.8.2 EDGE EVENT STATUS

Each VCI_INx# input pin is associated with two register bits used to record edge transitions on the pins. The edge detection takes place after any input filtering, before polarity control and occurs even if the VCI_INx# input is not enabled as part of the VCI_OUT logic (the corresponding control bit in the VCI Input Enable Register is '0') or if the state of the VCI_INx# input is not latched (the corresponding control bit in the Latch Enable Register is '0'). One bit is set whenever there is a high-to-low transition on the VCI_INx# pin (the VCI Negedge Detect Register) and the other bit is set whenever there is a low-to-high transition on the VCI_INx# pin (the VCI Posedge Detect Register).

In order to minimize power drain on the VBAT circuit, the edge detection logic operates only when the input buffer for a VCI_INx# pin is enabled. The input buffer is enabled either when the VCI_INx# pin is configured to determine the VCI_OUT pin, as controlled by the VCI_IN[1:0]# field of the VCI Register, or when the input buffer is explicitly enabled in the VCI Input Enable Register. When the pins are not enabled transitions on the pins are ignored.

30.8.3 VCI PIN MULTIPLEXING

Each of the VCI inputs, as well as VCI_OUT, are multiplexed with standard VTR-powered GPIOs. When VTR power is off, the mux control is disabled and the pin always reverts to the VCI function. The VCI_INx# function should be disabled in the VCI Input Enable Register VCI Buffer Enable Register and for any pin that is intended to be used as a GPIO rather than a VCI_INx#, so that VCI_OUT is not affected by the state of the pin.

30.8.4 POWER ON INHIBIT TIMER

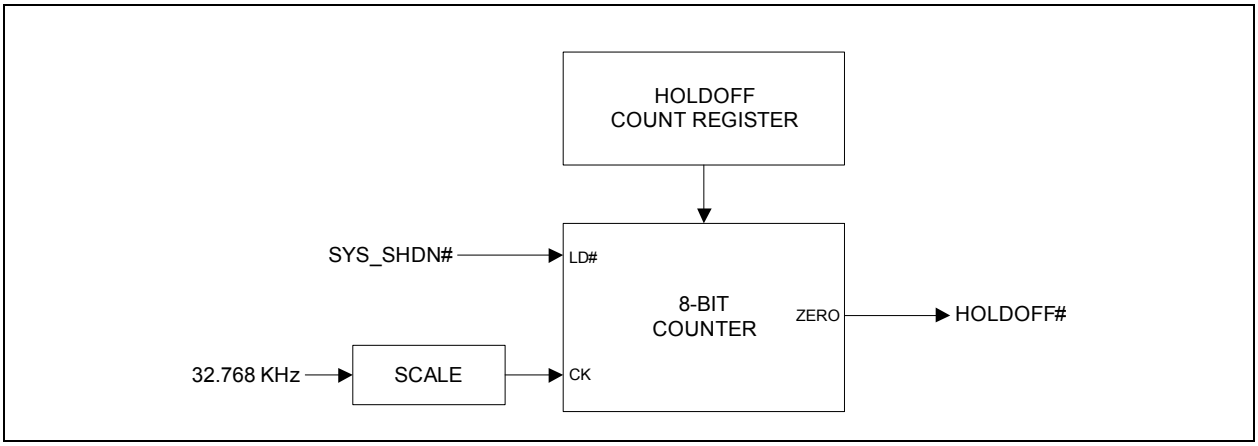
The Power On Inhibit Timer prevents the VBAT-Powered Control Interface VCI_OUT pin from being asserted for a programmable time period after the SYS_QSPI0N# pin asserted. This holdoff time can be used to give a system the opportunity to cool down after a thermal shutdown before allowing a user to attempt to turn the system on. While the Inhibit Timer is asserted, the VCI_OUT pin remains de-asserted and is unaffected by the VCI, Week Alarm and RTC interfaces.

The holdoff time is configured using the Holdoff Count Register. By setting the Holdoff Count Register to 0 the Inhibit Timer is disabled. When disabled, the HOLDOFF# signal is de-asserted and no counting takes place.

The HOLDOFF# output is asserted within one 32.768KHz clock cycle from the time SYS_QSPI0N# is asserted.

The following figure illustrates the operation of the Inhibit Timer:

FIGURE 30-4: POWER ON INHIBIT TIMER



The SCALE function reduces the 32.768KHz clock to 8Hz, so that the 8-bit counter counts intervals of 125ms. The following table shows some of examples of the effect of several settings of the Holdoff Count Register:

TABLE 30-3: HOLDOFF TIMING EXAMPLES

Holdoff Count Register	**Holdoff Time (SEC)**
0	Disabled (default)
1	0.125
5	0.625
10	1.25
15	1.875
100	12.5
150	18.75
200	25
255	31.875

30.8.5 APPLICATION EXAMPLE

For this example, a mobile platform configures the VBAT-Powered Control Interface (VCI) as follows:

- VCI_IN0# is wired to a power button on the mobile platform
- VCI_IN1# is wired to a power button on a dock
- The VCI_OUT pin is connected to the regulator that sources the VTR power rail, the rail which powers the EC

The VCI can be used in a system as follows:

1. In the initial condition, there is no power on either the VTR or VBAT power rails. All registers in the VCI are in an indeterminate state
2. A coin cell battery is installed, causing a RESET_VBAT. All registers in the interface are forced to their default conditions. The VCI_OUT pin is driven by hardware, input filters on the VCI_INx# pins are enabled, the VCI_INx# pins are all active low, all VCI inputs are enabled and all edge and status latches are in their non-asserted state
3. The power button on VCI_IN0# is pushed. This causes VCI_OUT to be asserted, powering the VTR rail. This causes the EC to boot and start executing EC firmware
4. The EC changes the VCI configuration so that firmware controls the VCI_OUT pin, and sets the output control so that VCI_OUT is driven high. With this change, the power button can be released without removing the EC power rail.

5. EC firmware re-configures the VCI logic so that the VCI_INx# input latches are enabled. This means that subsequent presses of the power button do not have to be held until EC firmware switches the VCI logic to firmware control
6. During this phase the VCI_OUT pin is driven by the firmware-controlled state bit and the VCI input pins are ignored. However, the EC can monitor the state of the pins, or generate inputs when their state changes
7. At some later point, EC firmware must enter a long-term power-down state.
   - Firmware configures the Week Timer for a Week Alarm once every 8 hours. This will turn on the EC power rail three times a day and enable the EC to perform low frequency housekeeping tasks even in its lowest-power state
   - Firmware de-asserts VCI_OUT. This action kills power to the EC and automatically returns control of the VCI_OUT pin to hardware.
   - The EC will remain in its lowest-power state until a power pin is pushed, AC power is connected, or the Sub-Week Alarm is active

30.9 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for the VBAT-Powered Control Interface Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 30-4: REGISTER SUMMARY

EC Offset	Register Name
00h	VCI Register
04h	Latch Enable Register
08h	Latch Resets Register
0Ch	VCI Input Enable Register
10h	Holdoff Count Register
14h	VCI Polarity Register
18h	VCI Posedge Detect Register
1Ch	VCI Negedge Detect Register
20h	VCI Buffer Enable Register

30.9.1 VCI REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:18	Reserved	R	-	-
17	RTC_ALRM If enabled by RTC_ALRM_LE, this bit is set to '1' if the RTC Alarm signal is asserted. It is reset by writes to RTC_ALRM_LS.	R	0	RESET_VBAT
16	WEEK_ALRM If enabled by WEEK_ALRM_LE, this bit is set to '1' if the Week Alarm signal is asserted. It is reset by writes to WEEK_ALRM_LS.	R	0	RESET_VBAT
15:13	Reserved	R	-	-
Note 1: The VCI_IN[6:0]# bits default to the state of their respective input pins. The VCI_OUT bit is determined by the VCI hardware circuit				

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
12	FILTERS_BYPASS The Filters Bypass bit is used to enable and disable the input filters on the VCI_INx# pins. See Section 47.17, "VBAT-Powered Control Interface Timing". 1=Filters disabled 0=Filters enabled (default)	R/W	0	RESET_VBAT
11	FW_EXT This bit controls selecting between the external VBAT-Powered Control Interface inputs, or the VCI_FW_CNTRL bit output to control the VCI_OUT pin. 1=VCI_OUT is determined by the VCI_FW_CNTRL field, when VTR is active **Note:** 0=VCI_OUT is determined by the external inputs.	R/W	0	RESET_SYS and RESET_VBAT
10	VCI_FW_CNTRL This bit can allow EC firmware to control the state of the VCI_OUT pin. For example, when VTR_PWRGD is asserted and the FW_EXT bit is '1', clearing the VCI_FW_CNTRL bit de-asserts the active high VCI_OUT pin. BIOS must set this bit to '1' prior to setting the FW_EXT bit to '1' on power up, in order to avoid glitches on the VCI_OUT pin.	R/W	0	RESET_SYS and RESET_VBAT
9	VCI_OUT This bit provides the current status of the VCI_OUT pin.	R	See Note 1	–
8:7	Reserved	R	-	-
6:0	VCI_IN[6:0]# These bits provide the latched state of the associated VCI_INx# pin, if latching is enabled or the current state of the pin if latching is not enabled. In both cases, the value is determined after the action of the VCI Polarity Register.	R	See Note 1	
Note 1:	The VCI_IN[6:0]# bits default to the state of their respective input pins. The VCI_OUT bit is determined by the VCI hardware circuit			

30.9.2 LATCH ENABLE REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:18	Reserved	R	-	-
17	RTC_ALRM_LE Latch enable for the RTC Power-Up signal. 1=Enabled. Assertions of the RTC Alarm are held until the latch is reset by writing the correspondingLS[6:0] bit 0=Not Enabled. The RTC Alarm signal is not latched but passed directly to the VCI_OUT logic	R/W	0h	RESET_VBAT
16	WEEK_ALRM_LE Latch enable for the Week Alarm Power-Up signal. 1=Enabled. Assertions of the Week Alarm are held until the latch is reset by writing the correspondingLS[6:0] bit 0=Not Enabled. The Week Alarm signal is not latched but passed directly to the VCI_OUT logic	R/W	0h	RESET_VBAT
15:7	Reserved	R	-	-
6:0	LE[6:0] Latching Enables. Latching occurs after the Polarity configuration, so a VCI_INx# pin is asserted when it is ‘0’ if VCI_IN_POL[6:0] is ‘0’, and asserted when it is ‘1 ‘if VCI_IN_POL[6:0] is ‘1’. For each LE[x] bit in the field: 1=Enabled. Assertions of the VCI_INx# pin are held until the latch is reset by writing the corresponding LS[6:0] bit 0=Not Enabled. The VCI_INx# signal is not latched but passed directly to the VCI_OUT logic	R/W	30h	RESET_VBAT

30.9.3 LATCH RESETS REGISTER

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:18	Reserved	R	-	-
17	RTC_ALRM_LS RTC Alarm Latch Reset. When this bit is written with a ‘1’, the RTC Alarm Event latch is reset The RTC Alarm input to the latch has priority over the Reset input Reads of this register are undefined.	W	-	–

Offset	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
16	WEEK_ALRM_LS Week Alarm Latch Reset. When this bit is written with a '1', the Week Alarm Event latch is reset The Week Alarm input to the latch has priority over the Reset input Reads of this register are undefined.	W	-	–
15:7	Reserved	R	-	-
6:0	LS[6:0] Latch Resets. When a Latch Resets bit (LS[x]) is written with a '1', the corresponding VCI_IN*x*# latch is de-asserted ('1'). The VCI_IN*x*# input to the latch has priority over the Latch Reset input, so firmware cannot reset the latch while the VCI_INx# pin is asserted. Firmware should sample the state of the pin in the VCI Register before attempting to reset the latch. As noted in the Latch Enable Register, the assertion level is determined by the VCI_IN_POL[6:0] bit. Reads of this register are undefined.	W	–	–

30.9.4 VCI INPUT ENABLE REGISTER

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:7	Reserved	R	-	-
6:0	IE[6:0] Input Enables for VCI_INx# signals. After changing the input enable for a VCI input, firmware should reset the input latch and clear any potential interrupt that may have been triggered by the input, as changing the enable may cause the internal status to change. For each IE[x] bit in the field: 1=Enabled. The corresponding VCI_INx# input is not gated and toggling the pin will affect the VCI_OUT pin 0=Not Enabled. The corresponding VCI_INx# input does not affect the VCI_OUT pin, even if the input is '0.' Unless the corresponding bit in the VCI Buffer Enable Register is 1, latches are not asserted, even if the VCI_INx# pin is low, during a VBAT power transition	R/W	Fh	RESET_VBAT

30.9.5 HOLDOFF COUNT REGISTER

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:8	Reserved	R	-	-
7:0	HOLDOFF_TIME These bits determine the period of time the VCI_OUT logic is inhibited from re-asserting VCI_OUT after a SYS_SHDN# event. FFh-01h=The Power On Inhibit Holdoff Time is set to a period between 125ms and 31.875 seconds. See Table 30-3 for examples 0=The Power On Inhibit function is disabled	RW	0	RESET_VBAT

30.9.6 VCI POLARITY REGISTER

Offset	14h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:7	Reserved	R	-	-
6:0	VCI_IN_POL[6:0] These bits determine the polarity of the VCI_INx input signals: For each VCI_IN_POL[x] bit in the field: 1=Active High. The value on the pins is inverted before use 0=Active Low (default)	RW	0	RESET_VBAT

30.9.7 VCI POSEDGE DETECT REGISTER

Offset	18h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:7	Reserved	R	-	-
6:0	VCI_IN_POS[6:0] These bits record a low to high transition on the VCI_INx# pins. A “1” indicates a transition occurred. For each VCI_IN_POS[x] bit in the field: 1=Positive Edge Detected 0=No edge detected	R/WC	0	RESET_VBAT

30.9.8 VCI NEGEDGE DETECT REGISTER

Offset	1Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:7	Reserved	R	-	-
6:0	VCI_IN_NEG[6:0] These bits record a high to low transition on the VCI_INx# pins. A “1” indicates a transition occurred. For each VCI_IN_NEG[x] bit in the field: 1=Negative Edge Detected 0=No edge detected	R/WC	0	RESET_VBAT

30.9.9 VCI BUFFER ENABLE REGISTER

Offset	20h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:7	Reserved	R	-	-
6:0	VCI_BUFFER_EN[6:0] Input Buffer enable. After changing the buffer enable for a VCI input, firmware should reset the input latch and clear any potential interrupt that may have been triggered by the input, as changing the buffer may cause the internal status to change. This register has no effect when VTR is powered. When VTR is on, the input buffers are enabled only by the IE[6:0] bit. For each VCI_BUFFER_EN[x] bit in the field: 1=VCI_INx# input buffer enabled independent of the IE[6:0] bit. The edge detection latches for this input are always enabled 0=VCI_INx# input buffer enabled by the IE[6:0] bit. The edge detection latches are only enabled when the IE[6:0] bit is ‘1’ (default)	RW	0	RESET_VBAT

31.0 VBAT-POWERED RAM

31.1 Overview

The VBAT Powered RAM provides a 128 Byte Random Accessed Memory that is operational while the main power rail is operational, and will retain its values powered by battery power while the main rail is unpowered.

31.2 References

No references have been cited for this feature.

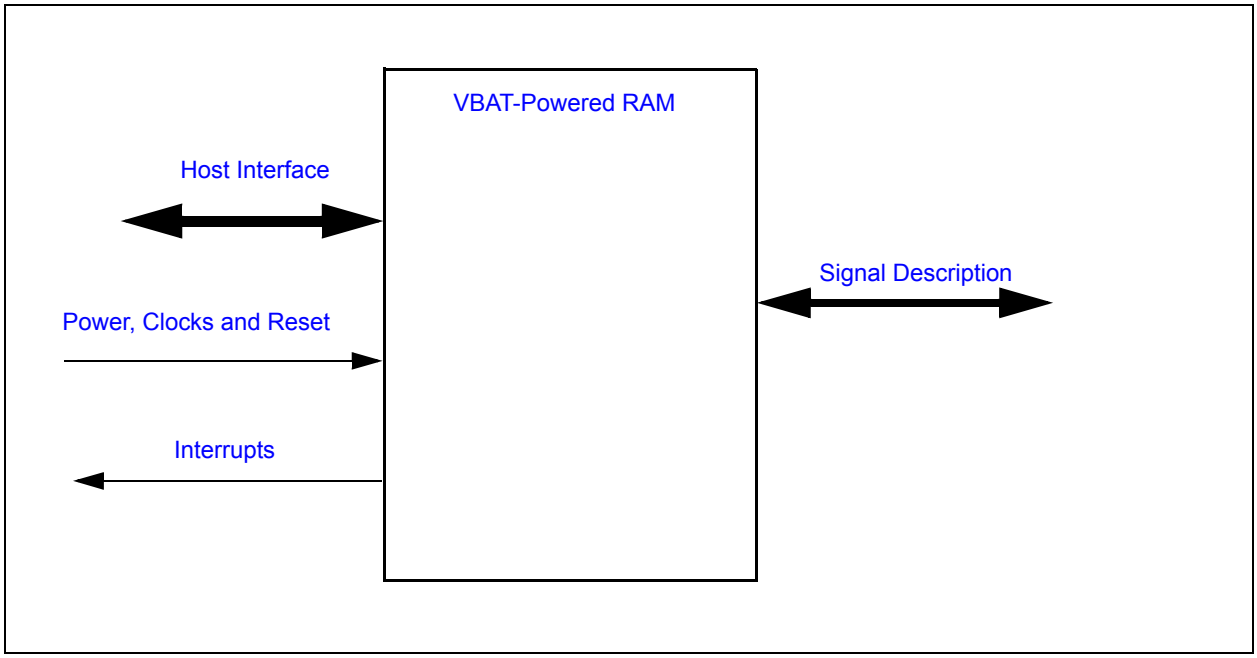
31.3 Terminology

There is no terminology defined for this section.

31.4 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

FIGURE 31-1: I/O DIAGRAM OF BLOCK



31.5 Signal Description

There are no external signals for this block.

31.6 Host Interface

The contents of the VBAT RAM are accessible to the EC over the Host Interface.

31.7 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

31.7.1 POWER DOMAINS

Name	Description
VTR	The main power well used when the VBAT RAM is accessed by the EC.
VBAT	The power well used to retain memory state while the main power rail is unpowered.

31.7.2 CLOCK INPUTS

No special clocks are required for this block.

31.7.3 RESETS

Name	Description
RESET_VBAT	This signal resets all the registers and logic in this block to their default state.

31.8 Interrupts

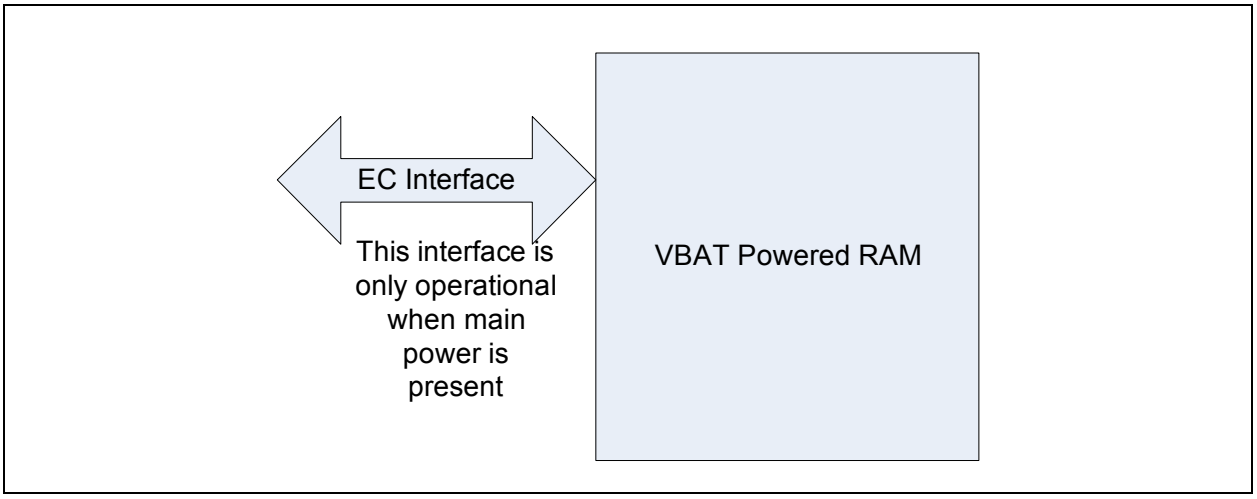
This block does not generate any interrupts.

31.9 Low Power Modes

The VBAT-Powered RAM automatically enters a low power mode whenever it is not being accessed by the EC. There is no chip-level Sleep Enable input.

31.10 Description

FIGURE 31-2: VBAT RAM BLOCK DIAGRAM



The VBAT Powered RAM provides a 128 Byte Random Accessed Memory that is operational while VTR is powered, and will retain its values powered by VBAT while VTR is unpowered. The RAM is organized as a 32 words x 32-bit wide for a total of 128 bytes.

The contents of the VBAT RAM is indeterminate after a RESET_VBAT.

32.0 VBAT REGISTER BANK

32.1 Introduction

This chapter defines a bank of registers powered by VBAT.

32.2 Interface

This block is designed to be accessed internally by the EC via the register interface.

32.3 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

32.3.1 POWER DOMAINS

Name	Description
VBAT	The VBAT Register Bank are all implemented on this single power domain.

32.3.2 CLOCK INPUTS

This block does not require any special clock inputs. All register accesses are synchronized to the host clock.

32.3.3 RESETS

Name	Description
RESET_VBAT	This reset signal, which is an input to this block, resets all the logic and registers to their initial default state.

32.4 Interrupts

This block does not generate any interrupt events.

32.5 Low Power Modes

The VBAT Register Bank is designed to always operate in the lowest power consumption state.

32.6 Description

The VBAT Register Bank block is a block implemented for aggregating miscellaneous battery-backed registers required the host and by the Embedded Controller (EC) Subsystem that are not unique to a block implemented in the EC subsystem.

32.7 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for the VBAT Register Bank Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 32-1: REGISTER SUMMARY

Offset	Register Name
00h	Power-Fail and Reset Status Register
04h	TEST
08h	Clock Enable Register
0Ch	TEST
10h	TEST
14h	TEST

TABLE 32-1: REGISTER SUMMARY (CONTINUED)

Offset	Register Name
1Ch	TEST
20h	Monotonic Counter Register
24h	Counter HiWord Register
28h	TEST
2Ch	TEST

32.7.1 POWER-FAIL AND RESET STATUS REGISTER

The Power-Fail and Reset Status Register collects and retains the VBAT RST and WDT event status when VTR is unpowered.

Address	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7	VBAT_RST The VBAT_RST bit is set to '1' by hardware when a RESET_VBAT is detected. This is the register default value. To clear VBAT RST EC firmware must write a '1' to this bit; writing a '0' to VBAT RST has no affect.	R/WC	1	RESET_VBAT
6	SYSRESETREQ This bit is set to '1b' if a RESET_SYS was triggered by an ARM SYSRESETREQ event. This bit is cleared to '0b' when written with a '1b'; writes of a '0b' have no effect.	R/WC	-	-
5	WDT This bit is set to '1b' if a RESET_SYS was triggered by a Watchdog Timer event. This bit is cleared to '0b' when written with a '1b'; writes of a '0b' have no effect.	R/WC	0	RESET_VBAT
4	RESETI This bit is set to '1b' if a RESET_SYS was triggered by a low signal on the RESETI# input pin. This bit is cleared to '0b' when written with a '1b'; writes of a '0b' have no effect.	R/WC	0	RESET_VBAT
3	TEST	R/WC	0	RESET_VBAT
2	SOFT_SYS_RESET Status This bit is set to '1b' if a was triggered by an assertion of the SOFT-_SYS_RESET bit in the System Reset Register. This bit is cleared to '0b' when written with a '1b'; writes of a '0b' have no effect.	R/WC	0	RESET_VBAT
1	Reserved	R	0	RESET_VBAT
0	Reserved	R	-	-

32.7.2 CLOCK ENABLE REGISTER

Address	08h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:3	Reserved	R	-	-
3	XOSEL This bit selects between a single-ended clock source for the crystal oscillator or an external parallel crystal. 1=The crystal oscillator is driven by a single-ended 32KHz clock source connected to the XTAL2 pin 0=The crystal oscillator requires a 32KHz parallel resonant crystal connected between the XTAL1 and XTAL2 pins	R/W	0b	RESET_VBAT
2	32KHZ_SOURCE This field determines the source for the always-on 32KHz internal clock source. If set to '1b', this bit will only take effect if an active clock has been detected on the crystal pins. Once the 32KHz source has been switched, activity detection on the crystal no longer functions. Therefore, if the crystal oscillator uses a single-ended input, once started that input must not stop while this bit is '1b'. 1=Crystal Oscillator. The selection between a singled-ended input or a resonant crystal is determined by XOSEL in this register 0=Silicon Oscillator	R/W	0b	RESET_VBAT
1	EXT_32K This bit selects the source for the 32KHz clock domain. 1=The 32KHZ_IN VTR-powered pin is used as a source for the 32KHz clock domain. If an activity detector does not detect a clock on the selected source, the always-on 32KHz internal clock source is automatically selected 0=The always-on32Khz clock source is used as the source for the 32KHz clock domain	R/W	0b	RESET_VBAT
0	32K_SUPPRESS 1=32KHz clock domain is off while VTR is off (i.e., while on VBAT only). The 32KHz domain is always on while VTR is on, so the PLL always has a reference 0=32KHz clock domain is enabled while VTR is off (i.e., while on VBAT only). The clock source for the 32KHz domain is determined by the other bits in this register	R/W	0b	RESET_VBAT

32.7.3 MONOTONIC COUNTER REGISTER

Address	20h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	MONOTTONIC_COUNTER Read-only register that increments by 1 every time it is read. It is reset to 0 on a VBAT Power On Reset.	R	0b	RESET_VBAT

32.7.4 COUNTER HIWORD REGISTER

Address	24h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	COUNTER_HIWORD Thirty-two bit read/write register. If software sets this register to an incrementing value, based on an external non-volatile store, this register may be combined with the Monotonic Counter Register to form a 64-bit monotonic counter.	R/W	0b	RESET_VBAT

33.0 EC SUBSYSTEM REGISTERS

33.1 Introduction

This chapter defines a bank of registers associated with the EC Subsystem.

33.2 References

None

33.3 Interface

This block is designed to be accessed internally by the EC via the register interface.

33.4 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

33.4.1 POWER DOMAINS

Name	Description
VTR	The logic and registers implemented in this block are powered by this power well.

33.4.2 CLOCK INPUTS

This block does not require any special clock inputs. All register accesses are synchronized to the host clock.

33.4.3 RESETS

Name	Description
RESET_SYS	This signal resets all the registers and logic in this block to their default state, except WDT Event Count Register.
RESET_SYS_nWDT	This signal resets the WDT Event Count Register register. This reset is not asserted on a WDT Event.
RESET_VTR	This reset signal is asserted only on VTR power on.

33.5 Interrupts

This block does not generate any interrupt events.

33.6 Low Power Modes

The EC Subsystem Registers may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry. When this block is commanded to sleep it will still allow read/write access to the registers.

33.7 Description

The EC Subsystem Registers block is a block implemented for aggregating miscellaneous registers required by the Embedded Controller (EC) Subsystem that are not unique to a block implemented in the EC subsystem.

33.8 EC-Only Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for the EC Subsystem Registers Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 33-1: REGISTER SUMMARY

Offset	Register Name
04h	AHB Error Address Register
08h	TEST
0Ch	TEST
10h	TEST
14h	AHB Error Control Register
18h	Interrupt Control Register
20h	Debug Enable Register
24h	OTP Lock Register
28h	WDT Event Count Register
2Ch	TEST
30h	TEST
34h	TEST
38h	TEST
3Ch	TEST
44h	TEST
48h	TEST
5Ch	TEST
60h	TEST
64h	GPIO Bank Power Register
68h	TEST
6Ch	TEST
70h	JTAG Master Configuration Register
74h	JTAG Master Status Register
78h	JTAG Master TDO Register
7Ch	JTAG Master TDI Register
80h	JTAG Master TMS Register
84h	JTAG Master Command Register
90h	TEST
100h	TEST

33.8.1 AHB ERROR ADDRESS REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	AHB_ERR_ADDR In priority order: 1. AHB address is registered when an AHB error occurs on the processors AHB master port and the register value was already 0. This way only the first address to generate an exception is captured. 2. The processor can clear this register by writing any 32-bit value to this register.	R/WZC	0h	RESET_SYS

33.8.2 AHB ERROR CONTROL REGISTER

Offset	14h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
7:2	Reserved	R	-	-
1	TEST	R/W	0h	RESET_SYS
0	AHB_ERROR_DISABLE 1=EC memory exceptions are disabled 0=EC memory exceptions are enabled	R/W	0h	RESET_SYS

33.8.3 INTERRUPT CONTROL REGISTER

Offset	18h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:1	Reserved	R	-	-
0	NVIC_EN This bit enables Alternate NVIC IRQ's Vectors. The Alternate NVIC Vectors provides each interrupt event with a dedicated (direct) NVIC vector. 1=Alternate NVIC vectors enabled 0=Alternate NVIC vectors disabled	R/W	1b	RESET_SYS

33.8.4 DEBUG ENABLE REGISTER

Offset	20h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:4	Reserved	R	-	-
3	DEBUG_PU_EN If this bit is set to '1b' internal pull-up resistors are automatically enabled on the appropriate debugging port wires whenever the debug port is enabled (the DEBUG_EN bit in this register is '1b' and the JTAG_RST# pin is high). The setting of DEBUG_PIN_CFG determines which pins have pull-ups enabled when the debug port is enabled.	R/W	0h	RESET_SYS

Offset	20h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
2:1	DEBUG_PIN_CFG This field determines which pins are affected by the TRST# debug enable pin. 3=Reserved 2=The pins associated with the JTAG TCK and TMS switch to the debug interface when TRST# is de-asserted high. The pins associated with TDI and TDO remain controlled by the associated GPIO. This setting should be used when the ARM Serial Wire Debug (SWD) is required for debugging and the Serial Wire Viewer is not required 1=The pins associated with the JTAG TCK, TMS and TDO switch to the debug interface when TRST# is de-asserted high. The pin associated with TDI remains controlled by the associated GPIO. This setting should be used when the ARM Serial Wire Debug (SWD) and Serial Wire Viewer (SWV) are both required for debugging 0=All four pins associated with JTAG (TCK, TMS, TDI and TDO) switch to the debug interface when TRST# is de-asserted high. This setting should be used when the JTAG TAP controller is required for debugging	R/W	0h	RESET_SYS
0	DEBUG_EN This bit enables the JTAG/SWD debug port. 1=JTAG/SWD port enabled. A high on TRST# enables JTAG or SWD, as determined by SWD_EN 0=JTAG/SWD port disabled. JTAG/SWD cannot be enabled (the TRST# pin is ignored and the JTAG signals remain in their non-JTAG state)	R/W	0b	RESET_SYS

33.8.5 OTP LOCK REGISTER

Offset	24h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:5	Reserved	R	-	-
4	PUBLIC_KEY_LOCK This bit controls access to the Public Key region of the eFuse memory, bytes 128 to 191. Once written, this bit becomes Read Only. If the JTAG_EN bit is 1 (enabled), the Public Key is inaccessible, independent of the state of this bit. 1=The Public Key is inaccessible (i.e, always returns 0 or 1 for every bit) 0=The Public Key is accessible	R/W / R	0b	RESET_SYS

Offset	24h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
3	USER_OTP_LOCK This bit controls access to the User region of the eFuse memory, bytes 192 to 511. Once written, this bit becomes Read Only. If the JTAG_EN bit is 1 (enabled), the User region is inaccessible, independent of the state of this bit. 1=The User region is inaccessible (i.e, always returns 0 or 1 for every bit) 0=The User region is accessible	R/W / R	0b	RESET_SYS
2	PRIVATE_KEY_LOCK This bit controls access to Private Key region of the eFuse memory, bytes 0 to 31. Once written, this bit becomes Read Only. If the JTAG_EN bit is 1 (enabled), the Private Key is inaccessible, independent of the state of this bit. 1=The Private Key is inaccessible (i.e, always returns 0 or 1 for every bit) 0=The Private Key is accessible	R/W / R	0b	RESET_SYS
1	MCHIP_LOCK This bit controls access to Microchip region of the eFuse memory, bytes 32 to 127. Once written, this bit becomes Read Only. If the JTAG_EN bit is 1 (enabled), the Private Key is inaccessible, independent of the state of this bit. 1=The Microchip region is inaccessible (i.e, always returns 0 or 1 for every bit) 0=The Microchip region is accessible	R/W / R	0b	RESET_SYS
0	TEST	R	0b	RESET_SYS

33.8.6 WDT EVENT COUNT REGISTER

Offset	28h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:4	Reserved	R	-	-
3:0	WDT_EVENT_COUNT This field is cleared to 0 on a reset triggered by the main power on reset, but not on a reset triggered by the Watchdog Timer. This field needs to be written by application to indicate the number of times a WDT fired before loading a good EC code image. Note 1	R/W	0b	RESET_SYS_n-WDT

Offset	28h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
Note 1: The recommended procedure is to first clear the VTR WDT STATUS bit, increment the WDT_EVENT_COUNT, clear the VBAT WDT STATUS bit and then rearm the WDT.				

33.8.7 GPIO BANK POWER REGISTER

Offset	64h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:8	Reserved	R	-	–
7	GPIO Bank Power Lock 0 = VTR_LEVEL bits[2:0] and GPIO Bank Power Lock bit are R/W 1 = VTR_LEVEL bits[2:0] and GPIO Bank Power Lock bit are Read Only This bit cannot be cleared once it is set to '1'. Writing zero has no effect.	Bit[7]=0 R/W Bit[7]=1 RO	0h	RESET_SYS
6:3	Reserved	R	-	–
1	VTR_LEVEL2 Voltage value on VTR2. This bit is set by hardware after a VTR Power On Reset, but may be overridden by software. It must be set by software if the VTR power rail is not active when RESET_SYS is de-asserted. 1=VTR2 is powered by 1.8V 0=VTR2 is powered by 3.3V	see Bit[7]	0h	RESET_SYS
0	VTR_LEVEL1 Voltage value on VTR1. This bit is set by hardware after a VTR Power On Reset, but may be overridden by software. It must be set by software if the VTR power rail is not active when RESET_SYS is de-asserted. 1=VTR1 is powered by 1.8V 0=VTR1 is powered by 3.3V	see Bit[7]	0h	RESET_SYS

Note:	The Boot ROM reads the VTR_LEVEL1, VTR_LEVEL2 values from the SPI Flash Header and writes the VTR_LEVEL1, VTR_LEVEL2 bits. If the SPI Flash load fails, the Boot ROM clears all VTR_LEVEL1, VTR_LEVEL2 bits.

33.8.8 JTAG MASTER CONFIGURATION REGISTER

Offset	70h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:4	Reserved	R	-	–
3	MASTER_SLAVE This bit controls the direction of the JTAG port. 1=The JTAG Port is configured as a Master 0=The JTAG Port is configures as a Slave	R/W	0h	RESET_SYS
2:0	JTM_CLK This field determines the JTAG Master clock rate, derived from the 48MHz master clock. 7=375KHz 6=750KHz 5=1.5Mhz 4=3Mhz 3=6Mhz 2=12Mhz 1=24MHz 0=Reserved.	R/W	3h	RESET_SYS

33.8.9 JTAG MASTER STATUS REGISTER

Offset	74h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:1	Reserved	R	-	–
0	JTM_DONE This bit is set to ‘1b’ when the JTAG Master Command Register is written. It becomes ‘0b’ when shifting has completed. Software can poll this bit to determine when a command has completed and it is therefore safe to remove the data in the JTAG Master TDO Register and load new data into the JTAG Master TMS Register and the JTAG Master TDI Register.	R	-	RESET_SYS

33.8.10 JTAG MASTER TDO REGISTER

Offset	78h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	JTM_TDO When the JTAG Master Command Register is written, from 1 to 32 bits are shifted into this register, starting with bit 0, from the JTAG_TDO pin. Shifting is at the rate determined by the JTM_CLK field in the JTAG Master Configuration Register	R/W	0h	RESET_SYS

33.8.11 JTAG MASTER TDI REGISTER

Offset	7Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	JTM_TDI When the JTAG Master Command Register is written, from 1 to 32 bits are shifted out of this register, starting with bit 0, onto the JTAG_TDI pin. Shifting is at the rate determined by the JTM_CLK field in the JTAG Master Configuration Register	R/W	0h	RESET_SYS

33.8.12 JTAG MASTER TMS REGISTER

Offset	80h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:0	JTM_TMS When the JTAG Master Command Register is written, from 1 to 32 bits are shifted out of this register, starting with bit 0, onto the JTAG_TMS pin. Shifting is at the rate determined by the JTM_CLK field in the JTAG Master Configuration Register	R/W	0h	RESET_SYS

33.8.13 JTAG MASTER COMMAND REGISTER

Offset	84h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:5	Reserved	R	-	–
4:0	JTM_COUNT If the JTAG Port is configured as a Master, writing this register starts clocking and shifting on the JTAG port. The JTAG Master port will shift JTM_COUNT+1 times, so writing a '0h' will shift 1 bit, and writing '31h' will shift 32 bits. The signal JTAG_CLK will cycle JTM_COUNT+1 times. The contents of the JTAG Master TMS Register and the JTAG Master TDI Register will be shifted out on the falling edge of JTAG_CLK and the.JTAG Master TDO Register will get shifted in on the rising edge of JTAG_CLK. If the JTAG Port is configured as a Slave, writing this register has no effect.	W	–	RESET_SYS

34.0 SECURITY FEATURES

34.1 Overview

This device includes a set of components that can support a high level of system security. Hardware support is provided for:

- Authentication, using public key algorithms
- Integrity, using Secure Hash Algorithms (SHA)
- Privacy, using symmetric encryption (Advanced Encryption Standard, AES)
- Entropy, using a true Random Number Generator

34.2 References

- American National Standards Institute, “Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography”, X9.63-2011, December 2011
- American National Standards Institute, “Public Key Cryptography for the Financial Servic3es Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)”, X9.62-2005, November 2005
- International Standards Organization, “Information Technology - Security techniques - Cryptographic techniques based on elliptic curves -- Part 2: Digital Signatures”, ISO/IEC 15946-2, December 2002
- National Institute of Standards and Technology, “Secure Hash Standard (SHS)”, FIPS Pub 180-4, March 2012
- National Institute of Standards and Technology, “Digital Signature Standard (DSS)”, FIPS Pub 186-3, June 2009
- National Institute of Standards and Technology, “Advanced Encryption Standard (AES)”, FIPS Pub 197, November 2001
- National Institute of Standards and Technology, “Recommendation for Block Cipher Modes of Operation”, FIPS SP 800-38A, 2001
- RSA Laboratories, “PKCS#1 v2.2: RSA Cryptography Standard”, October 2012

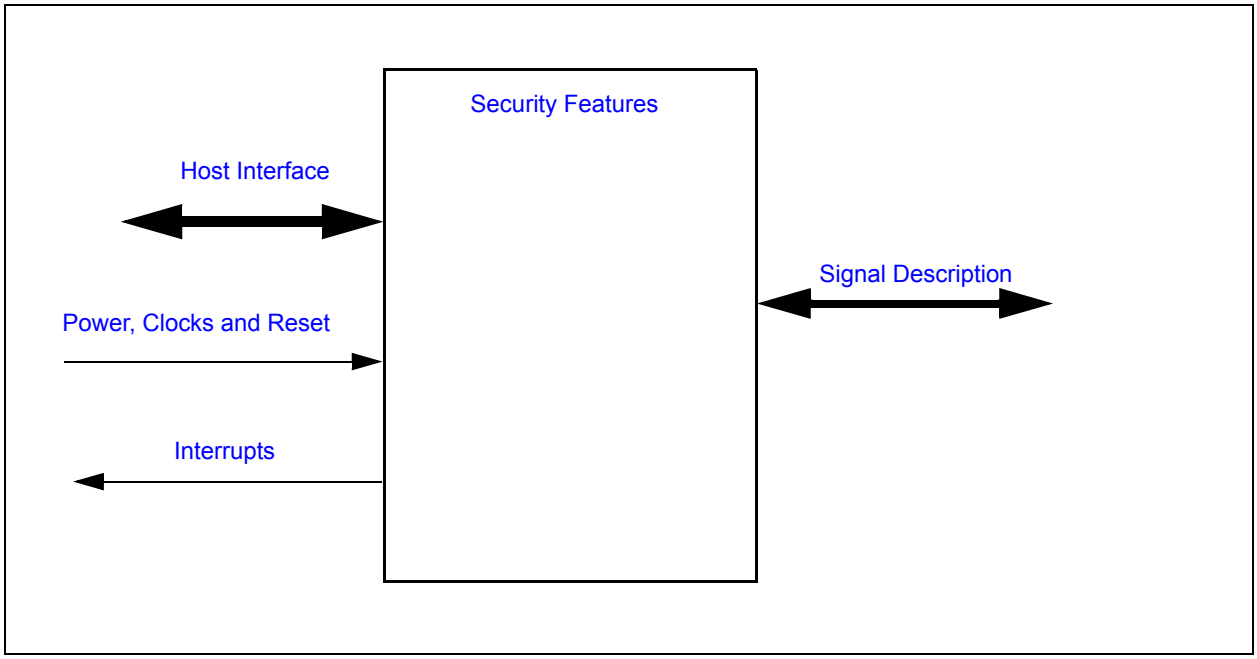
34.3 Terminology

There is no terminology defined for this section.

34.4 Interface

This block is designed to be accessed externally via the pin interface and internally via a registered host interface.

FIGURE 34-1: I/O DIAGRAM OF BLOCK



34.5 Signal Description

There are no external signals for this block.

34.6 Host Interface

Registers for the cryptographic hardware are accessible by the EC.

34.7 Power, Clocks and Reset

34.7.1 POWER DOMAINS

Name	Description
VTR	The main power well used when the VBAT RAM is accessed by the EC.

34.7.2 CLOCK INPUTS

No special clocks are required for this block.

34.7.3 RESETS

Name	Description
RESET_SYS	This signal resets all the registers and logic in this block to their default state.

34.8 Interrupts

This section defines the Interrupt Sources generated from this block.

Source	Description
Public Key Engine	
PKE ERROR	Public Key Engine core error detected
PKE END	Public Key Engine completed processing
Symmetric Encryption	
AES	Symmetric Encryption block completed processing
Cryptographic Hashing	
HASH	HASH
Random Number Generator	
RNG	Random Number Generator filled its FIFO

34.9 Low Power Modes

The Security Features may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry.

34.10 Description

The security hardware incorporates the following functions:

34.10.1 SYMMETRIC ENCRYPTION/DECRYPTION

Standard AES encryption and decryption, with key sizes of 128 bits, 192 bits and 256 bits, are supported with a hardware accelerator. AES modes that can be configured include Electronic Code Block (ECB), Cipher Block Chaining (CBC), Counter Mode (CTR), Output Feedback (OFB) and Cipher Feedback (CFB).

34.10.2 CRYPTOGRAPHIC HASHING

Standard SHA hash algorithms, including SHA-1, SHA-256, SHA-384 and SHA-512, are supported by hardware.

34.10.3 PUBLIC KEY CRYPTOGRAPHIC ENGINE

A large variety of public key algorithms are supported directly in hardware. These include:

- RSA encryption and decryption, with key sizes of 1024 bits, 2048 bits, 3072 bits and 4096 bits
- Elliptic Curve point multiply, with all standard NIST curves, using either binary fields or prime fields
- Elliptic Curve point multiply with Curve25519
- The Elliptic Curve Digital Signature Algorithm (ECDSA), using all supported NIST curves
- The Elliptic Curve Korean Certificate-based Digital Signature Algorithm (EC-KCDSA), using all supported NIST curves
- The Edwards-curve Digital Signature Algorithm (EdDSA), using Curve25519
- Miller-Rabin primality testing

The Public Key Engine includes a 24KB cryptographic SRAM, which can be accessed by the EC when the engine is not in operation. With its private SRAM memory, the Public Key Engine can process public key operations independently of the EC.

34.10.4 TRUE RANDOM NUMBER GENERATOR

A true Random Number Generator, which includes a 1K bit FIFO for pre-calculation of random bits.

34.10.5 MONOTONIC COUNTER

The Monotonic Counter is defined in Section 32.7.3, "Monotonic Counter Register". The counter automatically increments every time it is accessed, as long as VBAT power is maintained. If it is necessary to maintain a monotonic counter across VBAT power cycles, the Counter HiWord Register can be combined with the Monotonic Counter Register to form a 64-bit monotonic counter. Firmware would be responsible for updating the Counter HiWord on a VBAT POR. The HiWord could be maintained in a non-volatile source, such as the EEPROM or an external SPI Flash.

34.10.6 CRYPTOGRAPHIC API

The Boot ROM includes an API for direct software access to cryptographic functions. API functions for Hashing and AES include a DMA interface, so the operations can function on large blocks of SRAM with a single call.

34.11 Registers

TABLE 34-1: CRYPTOGRAPHIC SRAM

Block Instance	Start Address	End Address	Size
Cryptographic SRAM	4010_0000h	4010_5FFF	24KB

34.11.1 REGISTERS SUMMARY

The Public Key Engine, The Random Number Generator, the Hash Engine and the Symmetric Encryption Engine are all listed in the Block Overview and Base Addresses in Section 3.0, "Device Inventory".

35.0 TEST MECHANISMS

35.1 ARM Test Functions

Test mechanisms for the ARM are described in Section 39.0, "ARM M4F Based Embedded Controller".

35.2 JTAG Boundary Scan

Note:	Boundary Scan operates in 4-wire JTAG mode only. This is not supported by 2-wire SWD.

JTAG Boundary Scan includes registers and functionality as defined in IEEE 1149.1 and the CEC1702 BSDL file. Functionality implemented beyond the standard definition is summarized in Table 35-1. The CEC1702 Boundary Scan JTAG ID is shown in Table 1-1.

Note:	Must wait a minimum of 35ms after a POR to accurately read the Boundary Scan JTAG ID. Reading the JTAG ID too soon may return a Boundary Scan JTAG ID of 00000000h. This is not a valid ID value.

35.2.1 TAP CONTROLLER SELECT STRAP OPTION

The TAP Controller Select Strap Option determines the JTAG slave that is selected when JTAG_RST# is not asserted. The state of the TAP Controller Select Strap Option pin, defined in the Pin Configuration chapter, is sampled by hardware at POR according to the Slave Select Timing as defined in Section 38.16, "JTAG Interface Timing" and is registered internally to select between the debug and boundary scan TAP controllers.

If the strap is sampled low, the debug TAP controller is selected; if the strap is sampled high, the boundary scan slave is selected. An internal pull-up resistor is enabled by default on the TAP Controller Select Strap Option pin and can be disabled by firmware, if necessary.

TABLE 35-1: EXTENDED BOUNDARY SCAN FUNCTIONALITY

Bits	Function	Description
12, 14	TAP Controller Select Strap Option Override	When the Strap Option Override is '1,' the strap option is overridden to select the debug TAP Controller until the next time the strap is sampled. To set Strap Override Function, write 0X1FFFFD to the TAP controller instruction register, then write 0x5000 to the TAP controller data register. Note that the instruction register is 18 bits long; the data register is 16 bits long.

35.3 JTAG Master

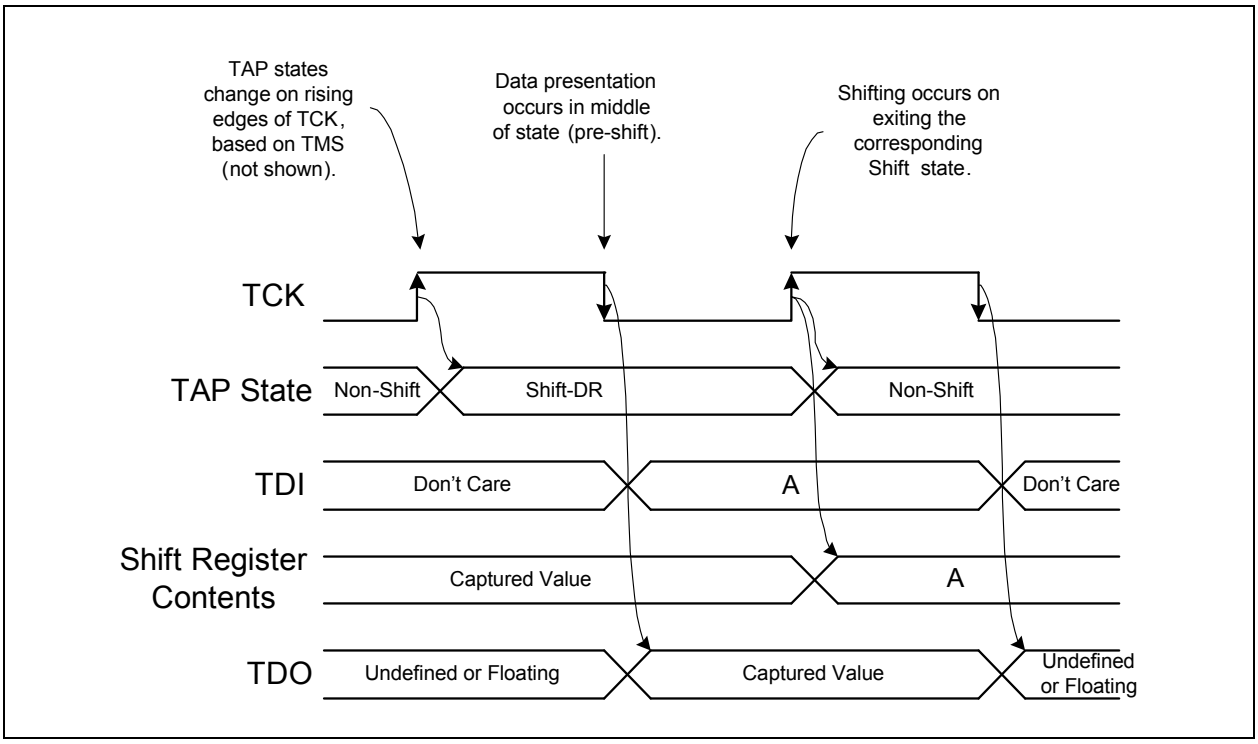
The JTAG Master controller in the CEC1702 enables the embedded controller to perform full IEEE 1149.1 test functions as the master controller for test operations at assembly time or in the field.

The JTAG Master interface shares the JTAG pin interface with the JTAG Boundary Scan and Debug TAP controllers; including, JTAG_CLK, JTAG_TDI, JTAG_TDO and JTAG_TMS. When the CEC1702 JTAG interface is configured as master, it is the responsibility of the master firmware to satisfy all requirements regarding JTAG port multiplexing. It is also it is the responsibility of the JTAG Master firmware to satisfy all requirements for external JTAG slave devices that require an external asynchronous reset (TRST#) input.

When JTAG slave functions are not required and the JTAG Master is enabled, the JTAG Interface pins are turned around so that the pins JTAG_CLK, JTAG_TMS and JTAG_TDI become outputs and the JTAG_TDO becomes an input.

Figure 35-1, "JTAG Signal Clocking" shows the clocking behavior of JTAG in the TAP controller in a JTAG Slave device. The rows "TAP State" and "Shift Reg. Contents" refer to the state of the JTAG Slave device and are provided for reference. When configured as a Master, the JTAG interface drives JTAG_CLK and will shift out data onto JTAG_TMS and JTAG_TDI in parallel, updating the pins on the falling edge of JTAG_CLK. The Master will sample data on JTAG_TDO on the rising edge of JTAG_CLK.

FIGURE 35-1: JTAG SIGNAL CLOCKING



35.3.1 JTAG MASTER REGISTER INTERFACE

Registers that control the JTAG Master Port are located in the EC Subsystem Registers block. These registers are listed in the following table:

TABLE 35-2: JTAG MASTER REGISTERS

Offset	Register Name
20h	Debug Enable Register
70h	JTAG Master Configuration Register
74h	JTAG Master Status Register
78h	JTAG Master TDO Register
7Ch	JTAG Master TDI Register
80h	JTAG Master TMS Register
84h	JTAG Master Command Register

35.3.1.1 Procedure to Enable the XNOR Chain

```
//BEGIN PROCEDURE TO ENTER XNOR CHAIN

/* Go into Reset */
    force `PIN_JTAG_TRST_N     = 1'b0;
    force `PIN_TEST_JTAG_TCLK  = 1'b0;
    force `PIN_TEST_JTAG_TMS   = 1'b0;
    force `PIN_TEST_JTAG_TDI   = 1'b0;
    #100;
```

```
force `PIN_JTAG_TRST_N     = 1'b1;
#100;


/* Initialize FSM TAP to RESET */
force `PIN_TEST_JTAG_TMS   = 1'b1;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1; // Clock Pulse 1
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b1;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1; // Clock Pulse 2
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b1;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1; // Clock Pulse 3
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b1;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1; // Clock Pulse 4
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b1;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1; // Clock Pulse 5
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b1;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1; // Clock Pulse 6
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

/* Send IR */
force `PIN_TEST_JTAG_TMS   = 1'b1;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
```

```
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b1;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b1;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

// Shift IR 0xD
force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
```

```
force `PIN_TEST_JTAG_TDI   = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b1;
force `PIN_TEST_JTAG_TDI   = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

// Latch IR / Send DR
force `PIN_TEST_JTAG_TMS   = 1'b1;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b1;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;
```

```
force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

// Shift DR
// Command Test Register Write
force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

// DATA 0x01
force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;
```

```
force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

// ADDR 0x88
force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
```

```
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b1;
force `PIN_TEST_JTAG_TDI   = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

// Complete
force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b1;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b1;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
```

```
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;

force `PIN_TEST_JTAG_TMS   = 1'b0;
force `PIN_TEST_JTAG_TDI   = 1'b0;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b1;
#20 force `PIN_TEST_JTAG_TCLK  = 1'b0;
```

36.0 EFUSE BLOCK

36.1 Introduction

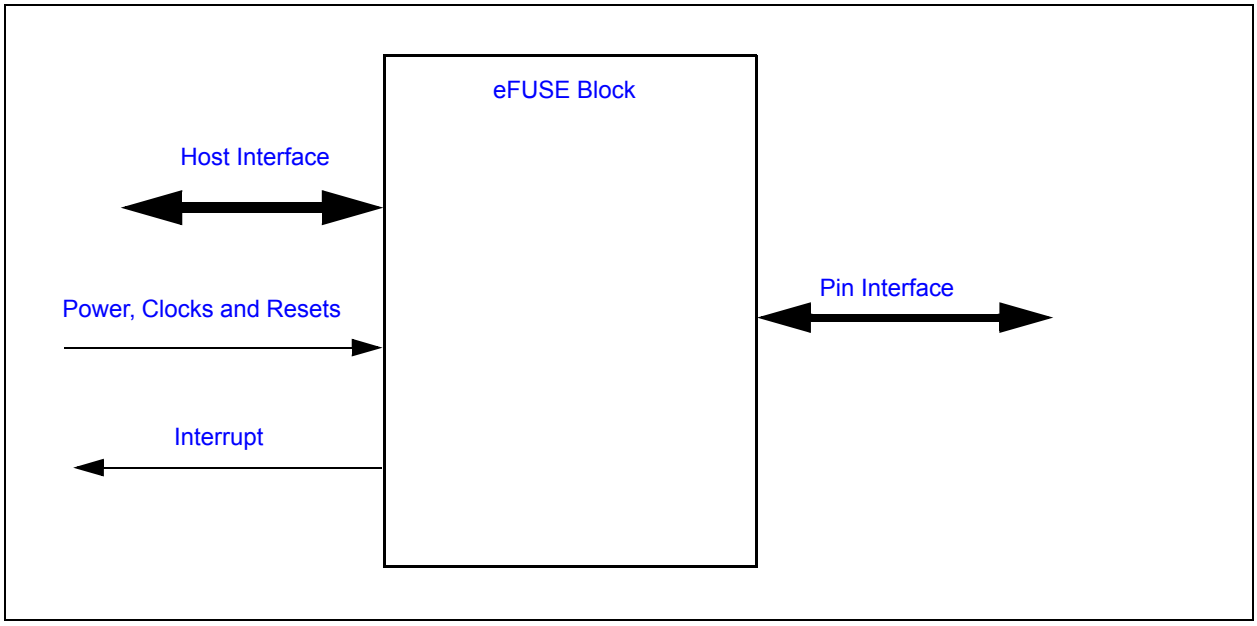
The eFUSE block provides a means of programming and accessing a block of One Time Programmable memory.

36.2 Terminology

None.

36.3 Interface

FIGURE 36-1: EFUSE BLOCK INTERFACE DIAGRAM



36.3.1 PIN INTERFACE

Table 36-1, "Signal Description" lists the signals that are routed to the pin interface.

TABLE 36-1: SIGNAL DESCRIPTION

Name	Direction	Description
VREF_ADC	Input	VPP Programming Pin

36.3.2 HOST INTERFACE

The registers defined for the eFUSE Block are accessible by the EC.

36.3.3 CLOCKING AND RESETS

This IP block has the following clocks and reset ports. For a complete list of all the clocks and resets associated with this block see Section 36.4, "Power, Clocks and Resets".

36.3.4 INTERRUPT INTERFACE

There are no interrupts from this block.

36.4 Power, Clocks and Resets

This section defines the Power, Clock, and Reset parameters of the block.

36.4.1 POWER DOMAINS

TABLE 36-2: POWER SOURCES

Name	Description
VTR	This power well sources all of the registers and logic in this block, except where noted.

36.4.2 CLOCKS

This section describes all the clocks in the block, including those that are derived from the I/O Interface as well as the ones that are derived or generated internally.

TABLE 36-3: CLOCKS

Name	Description
48MHz	This clock signal drives selected logic (e.g., counters).

36.4.3 RESETS

TABLE 36-4: RESET SIGNALS

Name	Description
RESET_SYS	This reset signal resets all of the registers and logic in this block.

36.5 Interrupt Generation

There are no interrupts from this block.

36.6 Low Power Modes

The eFUSE Block may be put into a low power state by the chip's Power, Clocks, and Reset (PCR) circuitry.

36.7 Description

The eFuse memory consists of four blocks of 1K bits each, for a total of 4K bits. The assignment of the eFuse data is shown in Section 36.9, "eFuse Memory Map".The eFUSE bits are programmed one bit at a time through a register interface. Addressing bits for writing bits is by a 2-bit block address field and a 10-bit offset within block field. The eFUSE memory can be read through 8-bit or 16-bit reads of a block of register addresses.

Note:	Only 8-bit and 16-bit access to the eFUSE memory is supported. A 32-bit access or larger will just have the 16-bit read-back value replicated in the other byte lanes.

36.7.1 EFUSE PROGRAMMING SEQUENCE

Programming of the eFuse array is one bit at a time. Programming changes a bit of '0b' to '1b'. There is no way to change the value of a bit that is already '1b'.

Sequence for programming one bit of eFuse memory:

1. Set the VREF_ADC pin to ground before powering up.
2. Power up the rest of the supplies
3. Set the MAN_ENABLE bit in the Manual Control Register to '1b'
4. Set FSOURCE_EN_READ to '0b' - DO NOT combine with step 5; writing FSOURCE_EN_PRGM and FSOURCE_EN_READ MUST be performed with separate writes.
5. Set the FSOURCE_EN_PRGM to '1b'
6. Set the VREF_ADC pin to the VPP programming voltage in between1.52V to 1.60V.

7. Set the IP_CS bit in the Manual Control Register to ‘0b’. The disables the eFuse memory array
8. Select the IP_ADDR_HI field in the Manual Mode Address Register to the block number of the block to be programmed
9. Set the IP_CS bit in the Manual Control Register to ‘1b’. The enables and powers up the selected block in the eFuse memory array
10. Select the IP_ADDR_LO field in the Manual Mode Address Register to the address of the bit within the current block to be programmed
11. Wait 100 ns (min)
12. Set PROG_EN to HIGH for 10µs (typical.
13. Set PROG_EN to LOW for 100 ns (min)
14. Repeat steps 10 to 13 for all bits within a block of 1K bits that are to be programmed to ‘1b’
15. In order to program bits in another 1K bit block, go back to Step 7 and follow the subsequent steps to programs the bits in that block

Programming one 1K bit block at a time eliminates glitches when switching between physical eFUSE blocks to prevent memory corruption.

Power down sequence after programming operation

1. Set the IP_CS bit in the Manual Control Register to ‘0b’. The disables and powers down the eFuse memory array
2. Set the VREF_ADC pin to ground
3. Set FSOURCE_EN_PRGM to ‘0b’ - DO NOT combine with step 4; writing FSOURCE_EN_PRGM and FSOURCE_EN_READ MUST be performed with separate writes
4. Set FSOURCE_EN_READ to ‘1b’

After programming is completed, the eFuse memory array may be read.

36.8 eFuse Reading Sequence

After power-up, the eFUSE Block is enabled for reading, but the block is in low-power/disabled mode.

1. Control Register should contain 0x10 (FSOURCE_EN_READ = 1); set this bit if it is not already set.
2. Set the bit 0 to 1b (ENABLE block) in Control Register.
3. Read the desired eFuse content (by reading eFUSE Memory).
4. If power savings is desired, turn the eFuse block off (clear ENABLE bit in Control Register)

36.9 eFuse Memory Map

The eFuse memory array is organized into four regions. Each of the four regions may be locked by a control bit in the EC Register Bank. When locked, a region in eFuse memory cannot be written, and always returns 0 on reads. The lock bits are located in the OTP Lock Register. In the following table, the four regions are identified by the lock bit names in the Lock Register.

Note:	Any secret customer information stored on chip in either EFUSE or VBAT memory must be encrypted for best security practices.

The below table is not accurate. Contact Microchip for details.

TABLE 36-5: EFUSE MEMORY MAP

Byte Number	Lock Bit	Location Name	Description
0-31	PRIVATE_KEY_LOCK	Encryption ECDH private key (aka, ECC private key)	256-bit P-256 Elliptic Curve private key, for key exchange as part of the optional decryption step in the Boot ROM Load process. Stored big-endian. • If this region is programmed by Microchip, it is encrypted with AES-256 and is always locked when the Boot ROM exits. • If this region is not programmed by Microchip, it is not encrypted, and is left unlocked when the Boot ROM exits and can be programmed by customers.
32-127	MCHIP_LOCK	Microchip	OTP data required by Microchip. This region is always locked when the Boot ROM exits.
128-159	PUBLIC_KEY_LOCK	Qx	Authentication Public Key X coordinate, NIST P-256 Elliptic Curve. 256 bits. Stored big-endian. This region is never locked after the Boot ROM exits.
160-191		Qy	Authentication Public Key Y coordinate, NIST P-256 Elliptic Curve. 256 bits. Stored big-endian. This region is never locked after the Boot ROM exits.
192-415	USER_OTP_LOCK	Customer use	One-time programmable memory available for customer use. This region is never locked after the Boot ROM exits.
416-447	USER_OTP_LOCK	R_X	Authentication of second ECDH Public Key X coordinate, NIST P-256 Elliptic Curve. 256 bits. Stored big-endian. This region is never locked after the Boot ROM exits.
448-479	USER_OTP_LOCK	R_Y	Authentication of second ECDH Public Key Y coordinate, NIST P-256 Elliptic Curve. 256 bits. Stored big-endian. This region is never locked after the Boot ROM exits.
480-481	USER_OTP_LOCK	TEST	Microchip test functions. These bits should not be modified.
482	USER_OTP_LOCK	Microchip test functions	Bits[5:0] Microchip test functions. These bits should not be modified. Bit[6] Debug Select 1=Configure debug port to use SWD for debugging 0=Configure debug port to use JTAG for debugging Bit[7] Debug Disable =0 (T/Eng)/ =1 (prod) 1=JTAG and SWD disabled on ROM code exit. See bit 6 0=JTAG and SWD enabled on ROM code exit. See bit 6

TABLE 36-5: EFUSE MEMORY MAP (CONTINUED)

Byte Number	Lock Bit	Location Name	Description
483	USER_ OTP_ LOCK	Customer Flags	Bit[0]: Authenticate 1=Header and firmware authenticated with ECC public key stored in Qx and Qy 0=Header and firmware checked with SHA-256 Bit[1]: Private Key Encryption - Enable 1=Private ECC key (bytes 0-31) is AES-encrypted with ROM AES key 0=Private ECC key does not need decryption Bit[2]: Private Key Encryption - Lock 1= ECC key (bytes 0-31) is locked 0= ECC key (bytes 0-31) is not locked Bit[3]: ECC508 Support - Enable 1=ECC508 support enabled 0=ECC508 support disabled Bits[5:4]: Undefined Bit[6]: Undefined Bit[7]: eFuse Bytes 0-31 Secure AES Encryption Key Select 0=Derived Secure AES Encryption Key 1=ROM Secure AES Encryption Key
484-507	USER_ OTP_ LOCK	TEST	Microchip test functions. These bits should not be modified.
508-509	USER_ OTP_ LOCK (Bytes 192-511)	SPI Flash Tag base	Tag block address in the SPI Flash, containing pointers to EC load image. Byte 508: Bits 15:8 of the Tag Block address Byte 509: Bits 23:16 of the Tag Block address Bits 7:0 of the Tag Block address are always 0
510-511	USER_ OTP_ LOCK	OTP Version	Version Identifier

36.10 EC Registers

Registers for this block are shown in the following summary table. Addresses for each register are determined by adding the offset to the Base Address for the eFUSE Block Block in the Block Overview and Base Address Table in Section 3.0, "Device Inventory".

TABLE 36-6: REGISTER SUMMARY

Offset	Register Name
00h	Control Register
04h	Manual Control Register
06h	Manual Mode Address Register
0Ch	Manual Mode Data Register
10h	eFUSE Memory

36.10.1 CONTROL REGISTER

Offset	00h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
15:5	Reserved	R	-	-
4	FSOURCE_EN_READ FSOURCE pin enable for reading: 1=FSOURCE switch logic connects eFUSE FSOURCE pin to a power pad for read mode. Only set this bit when FSOURCE_EN_PRGM bit is already 0 to avoid shorting the power pad to ground. 0=FSOURCE switch logic isolates eFUSE FSOURCE pin from ground. **Note:** Modifying FSOURCE_EN_PRGM and FSOURCE_EN_READ MUST be performed with separate writes. Refer Section 36.7.1, "eFUSE Programming Sequence" for programming sequence details.	R/W	1b	RESET_SYS
3	FSOURCE_EN_PRGM FSOURCE pin enable for programming: 1=FSOURCE switch logic connects eFUSE FSOURCE pin to a power pad for PROGRAM mode. 0=FSOURCE switch logic isolates eFUSE FSOURCE pin from power pad. **Note:** Modifying FSOURCE_EN_PRGM and FSOURCE_EN_READ MUST be performed with separate writes. Refer Section 36.7.1, "eFUSE Programming Sequence" for programming sequence details.	R/W	0b	RESET_SYS
2	EXT_PGM External programming enable: 1=eFUSE programming is done via external pin interface 0=Manual/Normal mode. eFUSE programming is done via this block's register set	R/W	0b	RESET_SYS
1	RESET Block reset: 1=Block is reset 0=Normal operation This bit self-clears and always reads back 0.	R/W	0b	RESET_SYS
0	ENABLE Block enable: 1=block is enabled for operation 0=block is disabled and in lowest power state	W	0b	RESET_SYS

36.10.2 MANUAL CONTROL REGISTER

Offset	04h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
15:6	Reserved	R	-	-
5	IP_OE eFUSE output enable. The IP might tri-state at various times, so this bit isolates the outputs to avoid potential crowbar. 1=eFUSE outputs enabled for read 0=eFUSE outputs isolated	R/W	0b	RESET_SYS
4	IP_SENSE_PULSE eFUSE sense, outputs are valid on falling edge of this bit.	R/W	0b	RESET_SYS
3	IP_PRCHG eFUSE precharge: 1=Outputs are being precharged 0=Outputs are not precharged	R/W	0b	RESET_SYS
2	IP_PRGM_EN) eFUSE program enable. Can also be considered the write signal: 1=eFUSE is programming 0=eFUSE is in read mode	R/W	0b	RESET_SYS
1	IP_CS eFUSE chip select (CS) pin: 1=eFUSE is enabled for PROGRAM/READ modes 0=eFUSE is disabled and in low power state	R/W	0b	RESET_SYS
0	MAN_ENABLE Manual mode enable bit: 1=Manual mode is enabled and this register interfaces to the eFUSE 0=Normal mode, internal controller interfaces to eFUSE IP This bit only takes affect when the REG_CTRL.EXT_PRGM bit is 0.	W	0b	RESET_SYS

36.10.3 MANUAL MODE ADDRESS REGISTER

Offset	06h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:12	Reserved	R	-	-
11:10	IP_ADDR_HI Manual mode address, selecting a 1K bit block of eFuse data.	R/W	0b	RESET_SYS
9:0	IP_ADDR_LO Manual mode address, selecting the bit address within a 1K bit block.	R/W	0b	RESET_SYS

36.10.4 MANUAL MODE DATA REGISTER

Offset	0Ch			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:16	Reserved	R	-	-
15:0	IP_DATA Manual mode data: This field connects to the eFUSE data output pins.	R/W	0b	RESET_SYS

36.10.5 EFUSE MEMORY

Offset	10h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
4095:0	IP_MEM eFUSE memory read-back data, used to read eFuse data. Although these registers can be written, writes only change the read-back data and do not update the eFuse memory itself.	R/W	0	RESET_SYS

37.0 ELECTRICAL SPECIFICATIONS

37.1 Maximum Ratings*

*Stresses exceeding those listed could cause permanent damage to the device. This is a stress rating only and functional operation of the device at any other condition above those indicated in the operation sections of this specification is not implied.

Note:	When powering this device from laboratory or system power supplies, it is important that the Absolute Maximum Ratings not be exceeded or device failure can result. Some power supplies exhibit voltage spikes on their outputs when the AC power is switched on or off. In addition, voltage transients on the AC power line may appear on the DC output. If this possibility exists, it is suggested that a clamp circuit be used.

37.1.1 ABSOLUTE MAXIMUM THERMAL RATINGS

Parameter	Maximum Limits
Operating Temperature Range	0^oC to $+70^oC$ Commercial -40^oC to $+85^oC$ Industrial
Storage Temperature Range	-55^o to $+150^oC$
Lead Temperature Range	Refer to JEDEC Spec J-STD-020B

37.1.2 ABSOLUTE MAXIMUM SUPPLY VOLTAGE RATINGS

Symbol	Parameter	Maximum Limits
VBAT	3.0V Battery Backup Power Supply with respect to ground	-0.3V to +3.63V
VTR_REG	Main Regulator Power Supply with respect to ground	-0.3V to +3.63V
VTR_ANALOG	3.3V Analog Power Supply with respect to ground	-0.3V to +3.63V
VTR1	3.3V or 1.8V Power Supply with respect to ground	-0.3V to +3.63V
VTR2	3.3V or 1.8V Power Supply with respect to ground	-0.3V to +3.63V

37.1.3 ABSOLUTE MAXIMUM I/O VOLTAGE RATINGS

Parameter	Maximum Limits
Voltage on any Digital Pin with respect to ground	Determined by Power Supply of I/O Buffer and Pad Type

37.2 Operational Specifications

37.2.1 POWER SUPPLY OPERATIONAL CHARACTERISTICS

TABLE 37-1: POWER SUPPLY OPERATING CONDITIONS

Symbol	Parameter	MIN	TYP	MAX	Units
VBAT	Battery Backup Power Supply	2.0	3.0	3.465	V
VTR_REG	Main Regulator Power Supply	1.71	3.0	3.465	V
VTR_ANALOG	Analog Power Supply	3.135	3.3	3.465	V
VTRx	3.3V Power Supply	3.135	3.3	3.465	V
	1.8V Power Supply	1.71	1.80	1.89	V

Note: The specification for the VTRx supplies are +/- 5%.

37.2.2 AC ELECTRICAL SPECIFICATIONS

The AC Electrical Specifications for the clock input time are defined in Section 38.4, "Clocking AC Timing Characteristics". The clock rise and fall times use the standard input thresholds of 0.8V and 2.0V unless otherwise specified and the capacitive values listed in this section.

37.2.3 CAPACITIVE LOADING SPECIFICATIONS

The following table defines the maximum capacitive load validated for the buffer characteristics listed in Table 37-3, "DC Electrical Characteristics".

CAPACITANCE T_A = 25°C; fc = 1MHz; V_{TR} = 3.3 VDC

Note: All output pins, except pin under test, tied to AC ground.

TABLE 37-2: MAXIMUM CAPACITIVE LOADING

Parameter	Symbol	Limits			Unit	Notes
		MIN	TYP	MAX		
Input Capacitance (all input pins)	C_{IN}			10	pF	Note 1
Output Capacitance (all output pins)	C_{OUT}			20	pF	Note 2

Note 1: All input buffers can be characterized by this capacitance unless otherwise specified.

2: All output buffers can be characterized by this capacitance unless otherwise specified.

37.2.4 DC ELECTRICAL CHARACTERISTICS FOR I/O BUFFERS

TABLE 37-3: DC ELECTRICAL CHARACTERISTICS

Parameter	Symbol	MIN	TYP	MAX	Units	Comments
PIO Type Buffer						
All PIO Buffers						Internal PU/PD selected via the GPIO Pin Control Register.
Pull-up current	R_{PU}	39	84	162	KΩ	
Pull-down current	R_{PD}	39	65	105	KΩ	
PIO						The drive strength is determined by programming bits[5:4] of the Pin Control 2 Register
DRIVE_STRENGTH = 00b	_	_	_	_	_	Same characteristics as an IO-2 mA.
DRIVE_STRENGTH = 01b	_	_	_	_	_	Same characteristics as an IO-4 mA.
DRIVE_STRENGTH = 10b	_	_	_	_	_	Same characteristics as an IO-8 mA.
DRIVE_STRENGTH = 11b	_	_	_	_	_	Same characteristics as an IO-12 mA.
I Type Input Buffer						TTL Compatible Schmitt Trigger Input
Low Input Level	V_{ILI}			0.3x VTR	V	
High Input Level	V_{IHI}	0.7x VTR			V	
Schmitt Trigger Hysteresis	V_{HYS}		400		mV	
O-2 mA Type Buffer						
Low Output Level	V_{OL}			0.4	V	I_{OL} = 2 mA (min)
High Output Level	V_{OH}	VTR-0.4			V	I_{OH} = -2 mA (min)
IO-2 mA Type Buffer	_	_	_	_	_	Same characteristics as an I and an O-2mA.
OD-2 mA Type Buffer						
Low Output Level	V_{OL}			0.4	V	I_{OL} = 2 mA (min)
IOD-2 mA Type Buffer	_	_	_	_	_	Same characteristics as an I and an OD-2mA.

TABLE 37-3: DC ELECTRICAL CHARACTERISTICS (CONTINUED)

Parameter	Symbol	MIN	TYP	MAX	Units	Comments
O-4 mA Type Buffer						
Low Output Level	V_{OL}			0.4	V	I_{OL} = 4 mA (min)
High Output Level	V_{OH}	VTR-0.4			V	I_{OH} = -4 mA (min)
IO-4 mA Type Buffer	_	_	_	_	_	Same characteristics as an I and an O-4mA.
OD-4 mA Type Buffer						
Low Output Level	V_{OL}			0.4	V	I_{OL} = 4 mA (min)
IOD-4 mA Type Buffer	_	_	_	_	_	Same characteristics as an I and an OD-4mA.
O-8 mA Type Buffer						
Low Output Level	V_{OL}			0.4	V	I_{OL} = 8 mA (min)
High Output Level	V_{OH}	VTR-0.4			V	I_{OH} = -8 mA (min)
						Unless the pin chapter explicitly indicates specific pin has “Over-voltage protection” feature.
IO-8 mA Type Buffer	_	_	_	_	_	Same characteristics as an I and an O-8mA.
OD-8 mA Type Buffer						
Low Output Level	V_{OL}			0.4	V	I_{OL} = 8 mA (min)
IOD-8 mA Type Buffer	_	_	_	_	_	Same characteristics as an I and an OD-8mA.
O-12 mA Type Buffer						
Low Output Level	V_{OL}			0.4	V	I_{OL} = 12mA (min)
High Output Level	V_{OH}	VTR-0.4			V	I_{OH} = -12mA (min)
IO-12 mA Type Buffer	_	_	_	_	_	Same characteristics as an I and an O-12mA.
OD-12 mA Type Buffer						
Low Output Level	V_{OL}			0.4	V	I_{OL} = 12mA (min)
IOD-12 mA Type Buffer	_	_	_	_	_	Same characteristics as an I and an OD-12mA.

TABLE 37-3: DC ELECTRICAL CHARACTERISTICS (CONTINUED)

Parameter	Symbol	MIN	TYP	MAX	Units	Comments
I_AN Type Buffer						
I_AN Type Buffer (Analog Input Buffer)	I_AN					Voltage range on pins: -0.3V to +3.63V These buffers are not 5V tolerant buffers and they are not back-drive protected
Crystal Oscillator						
XTAL1 (OCLK)	The CEC1702 crystal oscillator design requires a 32.768 KHz parallel resonant crystal with load caps in the range 4-18pF. Refer to “Application Note PCB Layout Guide for CEC1702” for more information.					
XTAL2 (ICLK)						
Low Input Level	V_{ILI}			0.4	V	
High Input Level	V_{ILH}	2.0			V	VIN = 0 to VTR
ADC Reference Pins						
ADC_VREF						
Voltage (Option A)	V		VTR		V	Connect to same power supply as VTR
Voltage (Option B)	V	2.97	3.0	3.03	V	
Input Impedance	R_{REF}	66	67	68	KΩ	
Input Low Current	ILEAK	-0.05		+0.05	µA	This buffer is not 5V tolerant This buffer is not backdrive protected.

37.2.4.1 Pin Leakage

Leakage characteristics for all digital I/O pins is shown in the following Pin Leakage table, unless otherwise specified. Two exceptions are pins with Over-voltage protection and Backdrive protection. Leakage characteristics for Over-Voltage protected pins and Backdrive protected pins are shown in the two sub-sections following the Pin Leakage table.

TABLE 37-4: PIN LEAKAGE (VTR=3.3V + 5%; VTR = 1.8V +5%)

(T_A = 0°C to +70°C)

Parameter	Symbol	MIN	TYP	MAX	Units	Comments
Leakage Current	I_{IL}			+/- 2	µA	VIN=0V to VTR
(T_A = -40°C to +85°C) Note 4						
Leakage Current	I_{IL}			+/- 3	µA	VIN=0V to VTR

OVER-VOLTAGE PROTECTION TOLERANCE

All the I/O buffers that do not have “Over-voltage Protection” are can only tolerate up to +/-10% I/O operation (or +1.98V when powered by 1.8V, or 3.63V when powered by 3.3V).

Functional pins that have “Over-voltage Protection” can tolerate up to 3.63V when powered by 1.8V, or 5.5V when powered by 3.3V. These pins are also backdrive protected. Backdrive Protection characteristics are shown in the following table:

TABLE 37-5: 5V TOLERANT LEAKAGE CURRENTS (VTR = 3.3V-5%)

(T_A = 0°C to +70°C)

Parameter	Symbol	MIN	TYP	MAX	Units	Comments
Three-State Input Leakage Current for 5V Tolerant Pins	I_{IL}	-	-	+/- 4	µA	VIN: 4.0V < Vin ≤ 5.5V
		-	-	+/- 66	µA	VIN: 3.6V < Vin ≤ 4.0V
		-	-	+/- 2	µA	VIN: ≤3.6V
(T_A = -40°C to +85°C) Note 4						
Three-State Input Leakage Current for 5V Tolerant Pins	I_{IL}	-	-	+/- 4	µA	VIN: 4.0V < Vin ≤ 5.5V
		-	-	+/- 70	µA	VIN: 3.6V < Vin ≤ 4.0V
		-	-	+/- 3	µA	VIN: ≤3.6V

Note: These measurements are done without an external pull-up.

TABLE 37-6: 3.6V TOLERANT LEAKAGE CURRENTS (VTR = 1.8V-5%)

(T_A = 0°C to +70°C)

Parameter	Symbol	MIN	TYP	MAX	Units	Comments
Three-State Input Leakage Current for Under-Voltage Tolerant Pins	I_{IL}	-	-	+/- 2	µA	VIN: 2.47V < Vin <3.6V
				+/- 42	µA	VIN: 1.92V < Vin <2.47V
		-	-	+/- 2	µA	VIN: ≤1.92V
(T_A = -40°C to +85°C) Note 4						
Three-State Input Leakage Current for Under-Voltage Tolerant Pins	I_{IL}	-	-	+/- 3	µA	VIN: 2.47V < Vin <3.6V
				+/- 50	µA	VIN: 1.92V < Vin <2.47V
		-	-	+/- 3	µA	VIN: ≤1.92V

Note: This measurements are done without an external pull-up.

BACKDRIVE PROTECTION

TABLE 37-7: BACKDRIVE PROTECTION LEAKAGE CURRENTS (VTR=0V)

(T_A = 0°C to +70°C)

Parameter	Symbol	MIN	TYP	MAX	Units	Comments
Input Leakage	I_{IL}			4	µA	3.6V < VIN ≤ 5.5V
Input Leakage	I_{IL}			2	µA	0V < VIN ≤ 3.6V
(T_A = -40°C to +85°C)Note 4						
Input Leakage	I_{IL}			5	µA	3.6V < VIN ≤ 5.5V
Input Leakage	I_{IL}			3	µA	0V < VIN ≤ 3.6V

37.2.5 ADC ELECTRICAL CHARACTERISTICS

TABLE 37-8: ADC CHARACTERISTICS

Symbol	Parameter	MIN	TYP	MAX	Units	Comments
VTR_ ANALOG	Analog Supply Voltage (powered by VTR)	3.135	3.3	3.465	V	
V_{RNG}	Input Voltage Range	0		ADC_ VREF	V	Range of ADC_VREF input to ADC ground
RES	Resolution	–	–	10	Bits	Guaranteed Mono-tonic
ACC	Absolute Accuracy	–	2	4	LSB	
DNL	Differential Non Linearity, DNL	-1	–	+1	LSB	Guaranteed Mono-tonic
INL	Integral Non Linearity, INL	-1.5	–	+1.5	LSB	Guaranteed Mono-tonic
E_{GAIN}	Gain Error, E_{GAIN}	-2	–	2	LSB	
E_{OFFSET}	Offset Error, E_{OFFSET}	-2	–	2	LSB	
CONV	Conversion Time		1.125		µS/channel	
II	Input Impedance	3.5	–	–	MΩ	

37.2.6 THERMAL CHARACTERISTICS

TABLE 37-9: THERMAL OPERATING CONDITIONS

Rating	Symbol	MIN	TYP	MAX	Unit
Consumer Temperature Devices					
Operating Junction Temperature Range	T_J	0	—	125	°C
Operating Ambient Temperature Range - Commercial	T_A	0	—	+70	°C
Operating Ambient Temperature Range - Industrial	T_A	-40	—	+85	°C
Power Dissipation: Internal Chip Power Dissipation: $P_{INT} = V_{DD} \times (I_{DD} - S\ I_{OH})$ I/O Pin Power Dissipation: $I/O = S\ (\{V_{DD} - V_{OH}\} \times I_{OH}) + S\ (V_{OL} \times I_{OL})$	P_D	69.3 (P_{INT} + $P_{I/O}$)			mW
Maximum Allowed Power Dissipation	P_{DMAX}	$(T_J - T_A)/\theta_{JA}$			W

TABLE 37-10: THERMAL PACKAGING CHARACTERISTICS

Characteristics	Symbol	TYP	MAX	Unit	Part #
Package Thermal Resistance, 84-pin WFBGA	θ_{JA}	62.7	—	°C/W	**CEC1702**
	θjC	17.6	—	°C/W	

Note: Junction to ambient thermal resistance, Theta-$_{JA}$ (θ_{JA}), and Junction to case thermal resistance, Theta-$_{JC}$ (θ_{JC}), numbers are achieved by package simulations

37.3 Power Consumption

TABLE 37-11: VTR SUPPLY CURRENT, I_VTR

VTR	48 MHz PLL	EC_CLK Freq	Typical (3.3V, 25^0 C)	Max (3.45V, 70^0 C)	Max (3.45V, 85^0 C) (Note 3)	Units	Comments (Note 1)
On	On	48MHz	12.5	14.5	16.0	mA	FULL ON (48MHz)and No eSPI Traffic
On	On	12MHz	8.0	9.5	11.5	mA	FULL ON (12MHz)and No eSPI Traffic
On	On	1MHz	5.5	6.5	8.0	mA	FULL ON (1MHz)and No eSPI Traffic
On	On	Off	1.5	2.5	4.0	mA	Light Sleep (Note 2) and No eSPI Traffic
On	Off	Off	0.5	1.9	3.0	mA	Heavy Sleep(Note 2) and No eSPI Traffic

Note 1: FULL ON is defined as follows: The processor is not sleeping, the Core regulator and the PLL remain powered, and at least one block is not sleeping.

2: The sleep states are defined in the System Sleep Control Register in the Power, Clocks and Resets Chapter. See Table 5.9.4, "System Sleep Control Register".

3: Applicable to CEC1702 only

TABLE 37-12: VBAT SUPPLY CURRENT, I_VBAT (VBAT=3.0V)

VTR	48 MHz PLL	Typical (3.0V, 25^0 C)	Max (3.0V, 25^0 C)	Units	Comments
Off	Off	11.0	20.0	uA	Internal 32kHz oscillator
Off	Off	5.0	9.0	uA	32kHz crystal oscillator
Off	Off	5.0	9.0	uA	External 32kHz clock on XTAL2 pin

TABLE 37-13: VBAT SUPPLY CURRENT, I_VBAT (VBAT=3.3V)

VTR	48 MHz PLL	Typical (3.3V, 25^0 C)	Max (3.3V, 25^0 C)	Units	Comments
Off	Off	12.0	22.0	uA	Internal 32kHz oscillator
Off	Off	6.0	10.0	uA	32kHz crystal oscillator
Off	Off	6.0	10.0	uA	External 32kHz clock on XTAL2 pin

38.0 TIMING DIAGRAMS

38.1 Power-up and Power-down Timing

FIGURE 38-1: VTR/VBAT POWER-UP TIMING

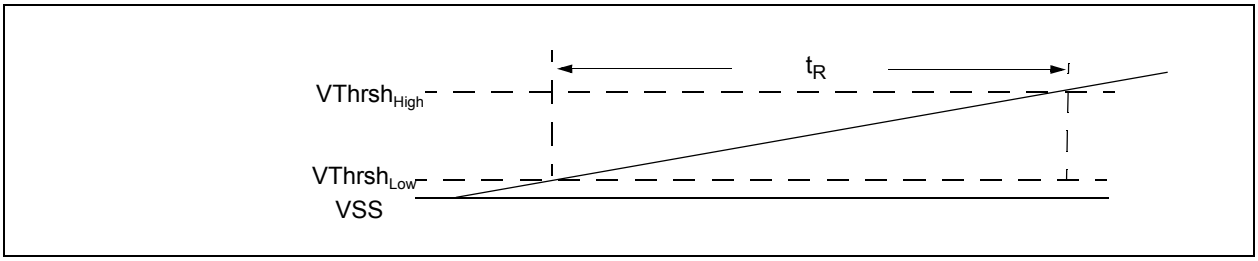


FIGURE 38-2: VTR RESET AND POWER-DOWN

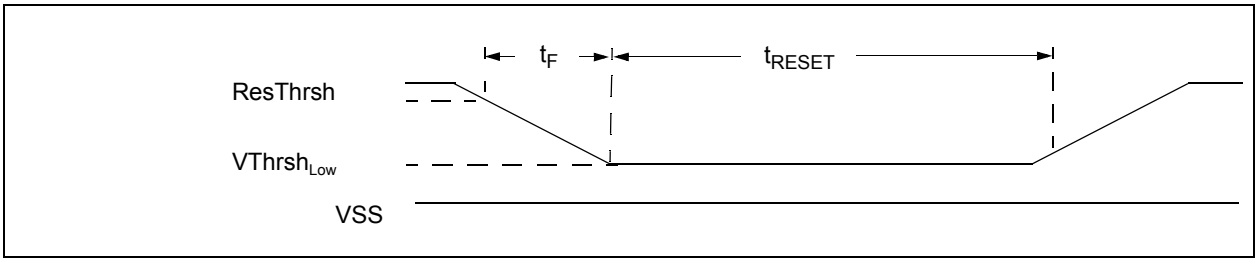


TABLE 38-1: VTR/VBAT TIMING PARAMETERS

Symbol	Parameter	MIN	TYP	MAX	Units	Notes
t_F	VTR Fall time	30			µs	1
	VBAT Fall time	30			µs	
t_R	VTR Rise time	0.050		20	ms	1
	VBAT Rise time	0.100		20	ms	
t_{RESET}	Minimum Reset Time	1			µs	
$VThrsh_{Low}$	VTR Low Voltage Threshold	0.1 × VTR			V	1
	VBAT Low Voltage Threshold	0.1 × VBAT			V	
$VThrsh_{High}$	VTR High Voltage Threshold			0.9 × VTR	V	1
	VBAT High Voltage Threshold			0.9 × VBAT	V	
ResThrsh	VTR Reset Threshold	0.5	1.8	2.7	V	1
	VBAT Reset Threshold	0.5	1.25	1.9	V	

Note 1: VTR applies to both VTR_REG and VTR_ANALOG

38.2 Power Sequencing

FIGURE 38-3: POWER RAIL SEQUENCING

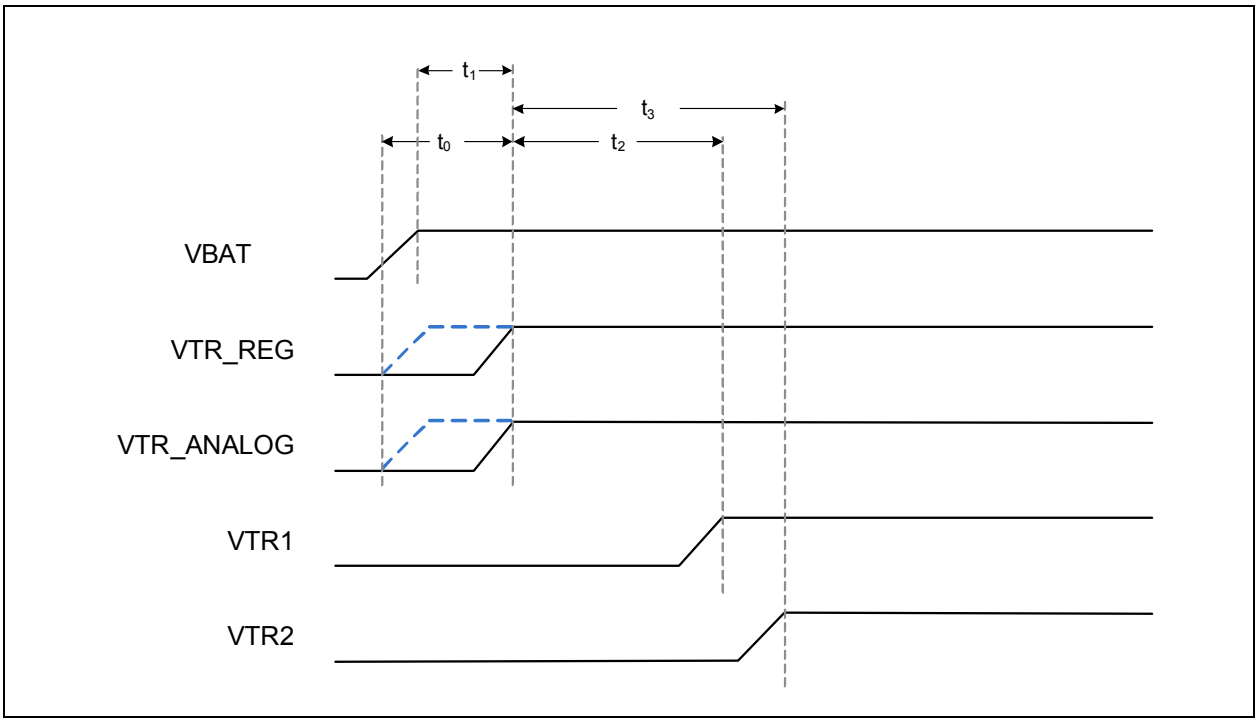


TABLE 38-2: POWER SEQUENCING PARAMETERS

Symbol	Parameter	Min	Typ	Max	Units	Notes
t_0	VTR_ANALOG stable relative to VTR_REG stable	0			ms	1
	VTR_REG stable relative to VTR_ANALOG stable	0			ms	
t_1	VBAT stable to VTR_ANALOG and VTR_REG stable	0			ms	2
t_2	VTR_ANALOG and VTR_REG stable to VTR1 stable, VTR1 at 3.3V	0		500	µs	
	VTR_ANALOG and VTR_REG stable to VTR1 stable, VTR1 at 1.8V	0		3	ms	

Note 1: VTR_ANALOG and VTR_REG may ramp in either order. There is no limit on the time between the ramp of one rail and the ramp of the other.

2: VBAT must rise no later than VTR_ANALOG and VTR_REG. This relationship is ensured by the recommended battery circuit.

TABLE 38-2: POWER SEQUENCING PARAMETERS (CONTINUED)

Symbol	Parameter	Min	Typ	Max	Units	Notes
t_3	VTR_ANALOG and VTR_REG stable to VTR2 stable, VTR2 at 3.3V	0		500	µs	
	VTR_ANALOG and VTR_REG stable to VTR2 stable, VTR2 at 1.8V	0		3	ms	

Note 1: VTR_ANALOG and VTR_REG may ramp in either order. There is no limit on the time between the ramp of one rail and the ramp of the other.

2: VBAT must rise no later than VTR_ANALOG and VTR_REG. This relationship is ensured by the recommended battery circuit.

38.3 RESETI# Timing

FIGURE 38-4: RESETI# TIMING

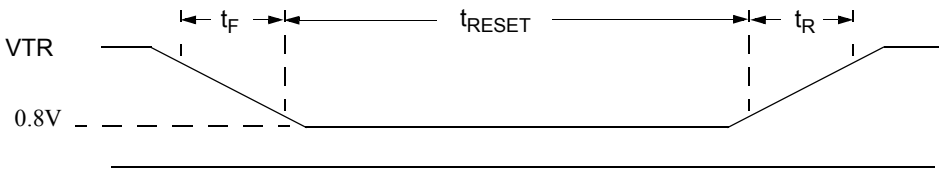


TABLE 38-3: RESETI# TIMING PARAMETERS

Symbol	Parameter	Limits		Units	Comments
		MIN	MAX		
t_F	RESETI# Fall time	0	1	ms	
t_R	RESETI# Rise time	0	1	ms	
t_{RESET}	Minimum Reset Time	1		µs	Note 1

Note 1: The RESETI# input pin can tolerate glitches of no more than 50ns.

38.4 Clocking AC Timing Characteristics

FIGURE 38-5: CLOCK TIMING DIAGRAM

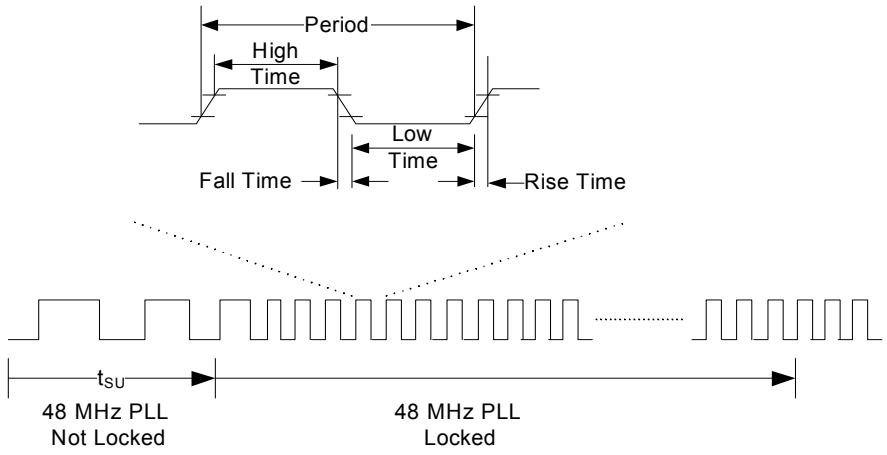


TABLE 38-4: CLOCK TIMING PARAMETERS

Clock	Symbol	Parameters	MIN	TYP	MAX	Units
48 MHz PLL	t_{SU}	Start-up accuracy from power-on-reset and waking from Heavy Sleep	-	-	3	ms
	-	Operating Frequency (locked to 32KHz single-ended input) (Note 1)	47.5	48	48.5	MHz
	-	Operating Frequency (locked to 32KHz Silicon Oscillator) (Note 1)	46.56	48	49.44	MHz
	CCJ	Cycle to Cycle Jitter(Note 2)	-200		200	ps
	t_{DO}	Output Duty Cycle	45	-	55	%
32MHz Ring Oscillator	-	Operating Frequency	16	-	48	MHz

Note 1: The 48MHz PLL is frequency accuracy is computed by adding +/-1% to the accuracy of the 32kHz reference clock.

2: The Cycle to Cycle Jitter of the 48MHz PLL is +/-200ps based on an ideal 32kHz clock source. The actual jitter on the 48MHz clock generated is computed by adding the clock jitter of the 32kHz reference clock to the 48MHz PLL jitter (e.g., 32kHz jitter +/- 200ps).

3: See the PCB Layout guide for design requirements and recommended 32.768 kHz Crystal Oscillators.

4: An external single-ended 32KHz clock is required to have an accuracy of +/- 100 ppm.

5: The external single-ended 32KHz clock source may be connected to either the XTAL2 pin or 32KHZ_IN pin.

6: PLL is started, either from waking from the Heavy Sleep mode, or after a Power On Reset

TABLE 38-4: CLOCK TIMING PARAMETERS (CONTINUED)

Clock	Symbol	Parameters	MIN	TYP	MAX	Units
32.768 kHz Crystal Oscillator (Note 3)	-	Operating Frequency	-	32.768	-	kHz
32KHz single-ended input (Note 5)	-	Operating Frequency	-	32.768	-	kHz
	-	Period	(Note 4)	30.52	(Note 4)	µs
	-	High Time	10			us
	-	Low Time	10			us
	-	Fall Time	-	-	1	us
	-	Rise Time	-	-	1	us
32KHz Silicon Oscillator	-	Operating Frequency	32.112	32.768	33.424	kHz
	-	Start-up delay from 0k Hz to Operating Frequency			150	us

Note 1: The 48MHz PLL is frequency accuracy is computed by adding +/-1% to the accuracy of the 32kHz reference clock.

2: The Cycle to Cycle Jitter of the 48MHz PLL is +/-200ps based on an ideal 32kHz clock source. The actual jitter on the 48MHz clock generated is computed by adding the clock jitter of the 32kHz reference clock to the 48MHz PLL jitter (e.g., 32kHz jitter +/- 200ps).

3: See the PCB Layout guide for design requirements and recommended 32.768 kHz Crystal Oscillators.

4: An external single-ended 32KHz clock is required to have an accuracy of +/- 100 ppm.

5: The external single-ended 32KHz clock source may be connected to either the XTAL2 pin or 32KHZ_IN pin.

6: PLL is started, either from waking from the Heavy Sleep mode, or after a Power On Reset

38.5 GPIO Timings

FIGURE 38-6: GPIO TIMING

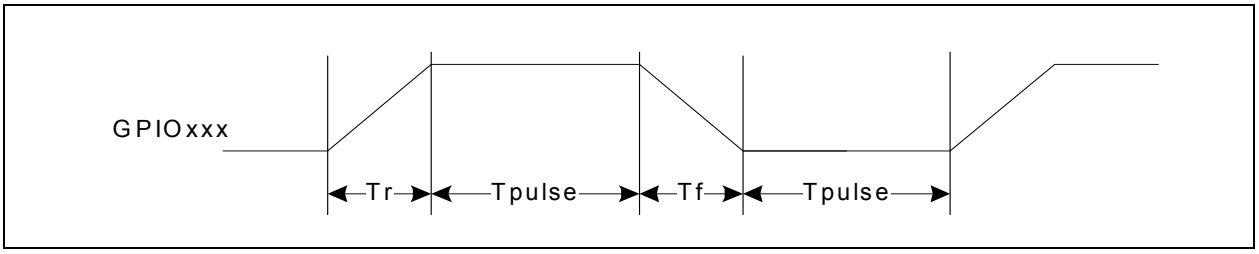


TABLE 38-5: GPIO TIMING PARAMETERS

Symbol	Parameter	MIN	TYP	MAX	Unit	Notes
t_R	GPIO Rise Time (push-pull)	0.54		1.31	ns	1
t_F	GPIO Fall Time	0.52		1.27	ns	
t_R	GPIO Rise Time (push-pull)	0.58		1.46	ns	2
t_F	GPIO Fall Time	0.62		1.48	ns	
t_R	GPIO Rise Time (push-pull)	0.80		2.00	ns	3
t_F	GPIO Fall Time	0.80		1.96	ns	
t_R	GPIO Rise Time (push-pull)	1.02		2.46	ns	4
t_F	GPIO Fall Time	1.07		2.51	ns	
t_{pulse}	GPIO Pulse Width	60			ns	

Note 1: Pad configured for 2ma, CL=2pF

2: Pad configured for 4ma, CL=5pF

3: Pad configured for 8ma, CL=10pF

4: Pad configured for 12ma, CL=20pF

38.6 Boot from SPI Flash Timing

FIGURE 38-7: SPI BOOT TIMING

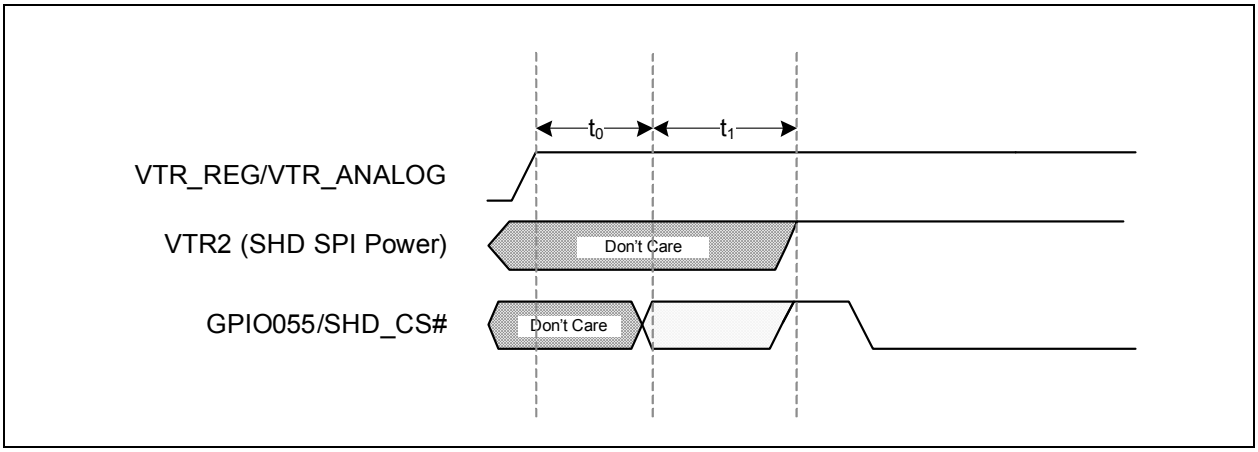


TABLE 38-6: SPI FLASH BOOT TIMING PARAMETERS

Symbol	Parameter	Min	Typ	Max	Unit	Notes
t_0	Time from VTR_REG Power On to Boot ROM samples SHD_CS#	3			ms	
t_1	Time from Boot ROM sample SHD_CS# to Boot ROM requires VTR2 (SHD SPI Power) On			5	ms	1

Note 1: The SPI Shared Flash interface is powered by VTR2. The max time for VTR2 Power on is determined by the CEC1702 Power Sequencing Requirements. See Section 38.2, "Power Sequencing," on page 382

2: GPIO171(JTAG_STRAP), which is powered by the VTR1, must be low pulled low at power-on to Boot from SPI Flash. The max time for VTR1 Power on is determined by the CEC1702 Power Sequencing Requirements. See Section 38.2, "Power Sequencing," on page 382

38.7 Serial Port (UART) Data Timing

FIGURE 38-8: SERIAL PORT DATA

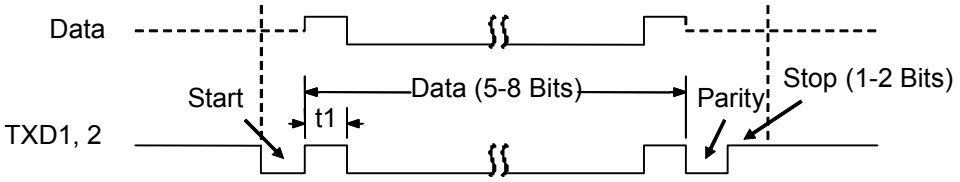


TABLE 38-7: SERIAL PORT DATA PARAMETERS

Name	Description	MIN	TYP	MAX	Units
t1	Serial Port Data Bit Time		t_{BR} (Note 1)		nsec

Note 1: tBR is 1/Baud Rate. The Baud Rate is programmed through the Baud_Rate_Divisor bits located in the Programmable Baud Rate Generator registers. Some of the baud rates have some percentage of error because the clock does not divide evenly. This error can be determined from the values in these baud rate tables.

38.8 Keyboard Scan Matrix Timing

TABLE 38-8: ACTIVE PRE DRIVE MODE TIMING

Parameter	Symbol	Value			Units	Notes
		MIN	TYP	MAX		
Active Predrive Mode	$t_{PREDRIVE}$		41.7		ns	

Note: The TYP value is based on two 48 MHz PLL clocks. The MIN and MAX values are dependent on the accuracy of the 48 MHz PLL.

38.9 PWM Timing

FIGURE 38-9: PWM OUTPUT TIMING

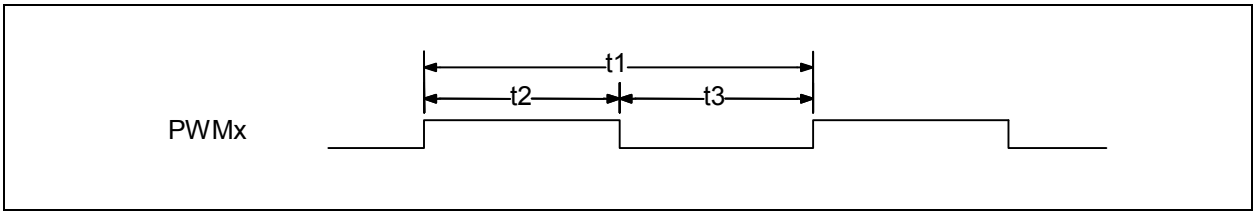


TABLE 38-9: PWM TIMING PARAMETERS

Name	Description	MIN	TYP	MAX	Units
t1	Period	42ns		23.3sec	
t_f	Frequency	0.04Hz		24MHz	
t2	High Time	0		11.65	sec
t3	Low Time	0		11.65	sec
t_d	Duty cycle	0		100	%

38.10 Fan Tachometer Timing

FIGURE 38-10: FAN TACHOMETER INPUT TIMING

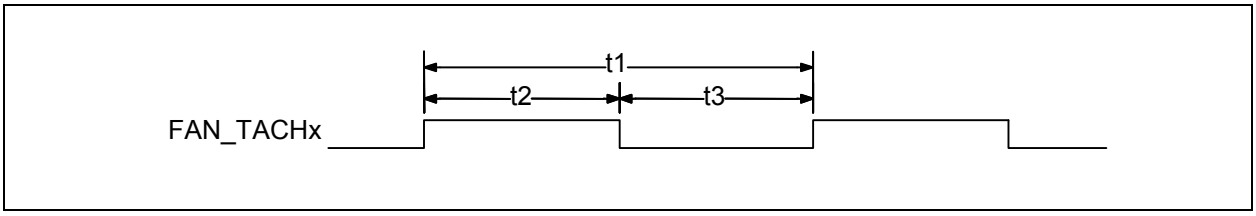


TABLE 38-10: FAN TACHOMETER INPUT TIMING PARAMETERS

Name	Description	MIN	TYP	MAX	Units
t1	Pulse Time	100			µsec
t2	Pulse High Time	20			
t3	Pulse Low Time	20			
Note:	t_{TACH} is the clock used for the tachometer counter. It is 30.52 * prescaler, where the prescaler is programmed in the Fan Tachometer Timebase Prescaler register.				

38.11 Blinking/Breathing PWM Timing

FIGURE 38-11: BLINKING/BREATHING PWM OUTPUT TIMING

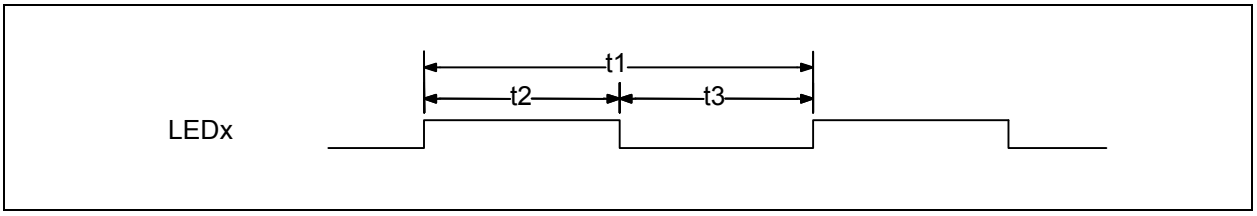


TABLE 38-11: BLINKING/BREATHING PWM TIMING PARAMETERS, BLINKING MODE

Name	Description	MIN	TYP	MAX	Units
t1	Period	7.8ms		32sec	
t_f	Frequency	0.03125		128	Hz
t2	High Time	0		16	sec
t3	Low Time	0		16	sec
t_d	Duty cycle	0		100	%

TABLE 38-12: BLINKING/BREATHING PWM TIMING PARAMETERS, GENERAL PURPOSE

Name	Description	MIN	TYP	MAX	Units
t1	Period	5.3µs		21.8ms	
t_f	Frequency	45.8Hz		187.5kHz	
t2	High Time	0		10.9	ms
t3	Low Time	0		10.9	ms
t_d	Duty cycle	0		100	%

38.12 I2C/SMBus Timing

FIGURE 38-12: I2C/SMBUS TIMING

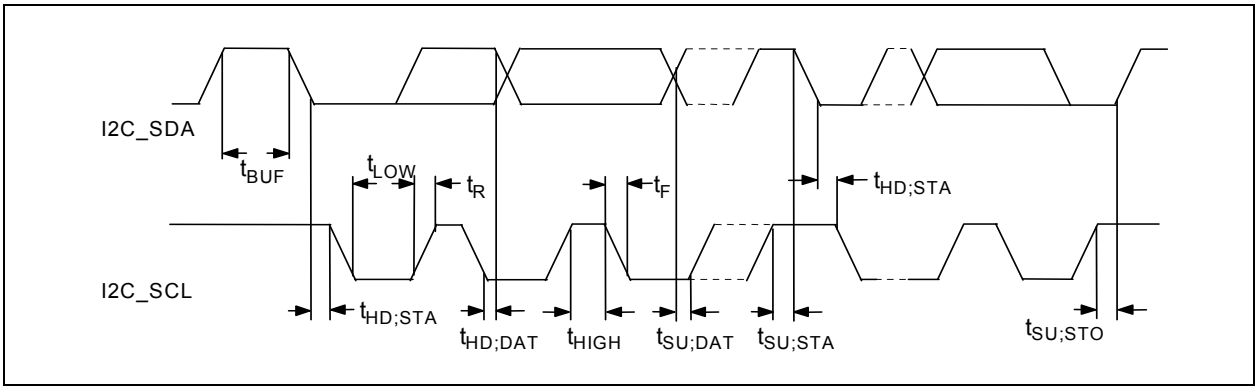


TABLE 38-13: I2C/SMBUS TIMING PARAMETERS

Symbol	Parameter	Standard-Mode		Fast-Mode		Fast-Mode Plus		Units
		MIN	MAX	MIN	MAX	MIN	MAX	
f_{SCL}	SCL Clock Frequency		100		400		1000	kHz
t_{BUF}	Bus Free Time	4.7		1.3		0.5		µs
$t_{SU;STA}$	START Condition Set-Up Time	4.7		0.6		0.26		µs
$t_{HD;STA}$	START Condition Hold Time	4.0		0.6		0.26		µs
t_{LOW}	SCL LOW Time	4.7		1.3		0.5		µs
t_{HIGH}	SCL HIGH Time	4.0		0.6		0.26		µs
t_R	SCL and SDA Rise Time		1.0		0.3		0.12	µs
t_F	SCL and SDA Fall Time		0.3		0.3		0.12	µs
$t_{SU;DAT}$	Data Set-Up Time	0.25		0.1		0.05		µs
$t_{HD;DAT}$	Data Hold Time	0		0		0		µs
$t_{SU;STO}$	STOP Condition Set-Up Time	4.0		0.6		0.26		µs

38.13 Quad SPI Master Controller - Serial Peripheral Interface (QMSPI) Timings

FIGURE 38-13: SPI CLOCK TIMING

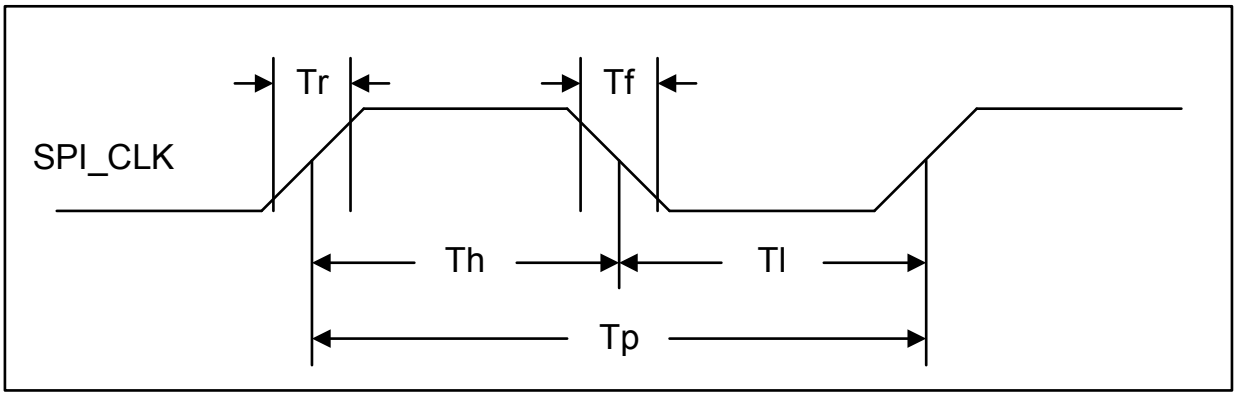
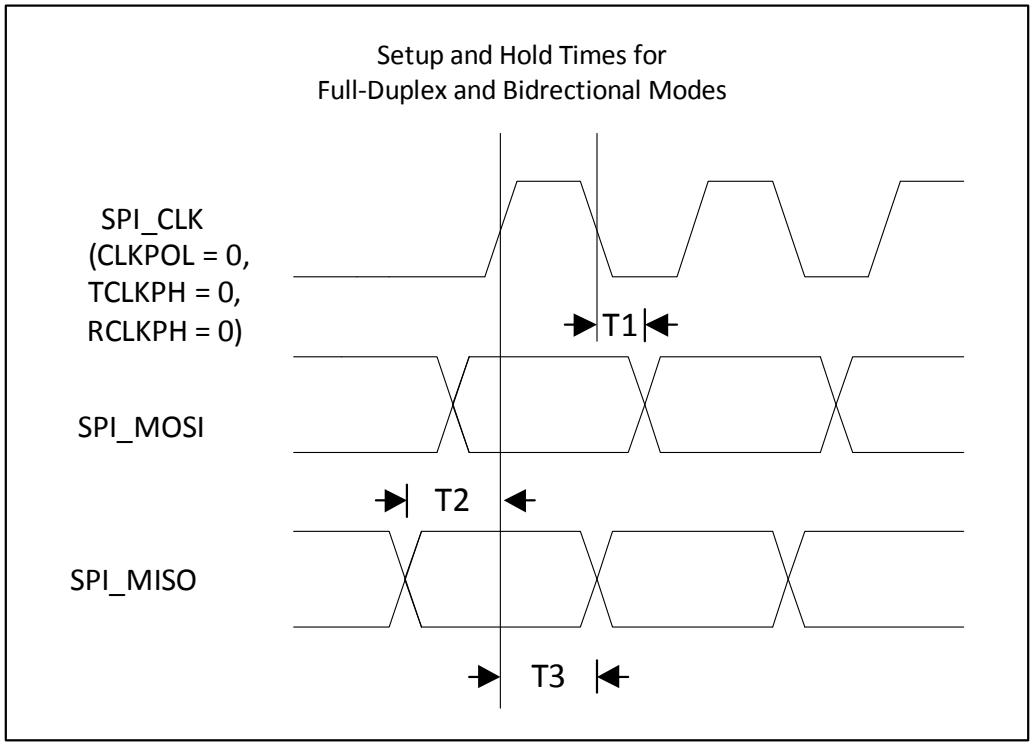


TABLE 38-14: SPI CLOCK TIMING PARAMETERS

Name	Description	MIN	TYP	MAX	Units
Tr	SPI Clock Rise Time. Measured from 10% to 90%.			3	ns
Tf	SPI Clock Fall Time. Measured from 90% to 10%.			3	ns
Th/Tl	SPI Clock High Time/SPI Clock Low Time	40% of SPCLK Period	50% of SPCLK Period	60% of SPCLK Period	ns
Tp	SPI Clock Period – As selected by SPI Clock Generator Register	20.8		5,333	ns
Note:	Test conditions are as follows: output load is C_L=30pF, pin drive strength setting is 4mA and slew rate setting is slow.				

FIGURE 38-14: SPI SETUP AND HOLD TIMES



Note:	SPI_IO[3:0] obey the SPI_MOSI and SPI_MISO timing. In the 2-pin SPI Interface implementation, SPI_IO0 pin is the SPI Master-Out/Slave-In (MOSI) pin and the SPI_IO1 pin is the Master-In/Slave-out (MISO) pin.

TABLE 38-15: SPI SETUP AND HOLD TIMES PARAMETERS

Name	Description	MIN	TYP	MAX	Units
T1	Data Output Delay			2	ns
T2	Data IN Setup Time	5.5			ns
T3	Data IN Hold Time	0			ns
Note:	Test conditions are as follows: output load is C_L=30pF, pin drive strength setting is 4mA and slew rate setting is slow				

38.14 General Purpose Serial Peripheral Interface (GP-SPI) Timings

Note that the following timing applies to all of the CEC1702 Serial Peripheral Interface functions.

FIGURE 38-15: SPI CLOCK TIMING

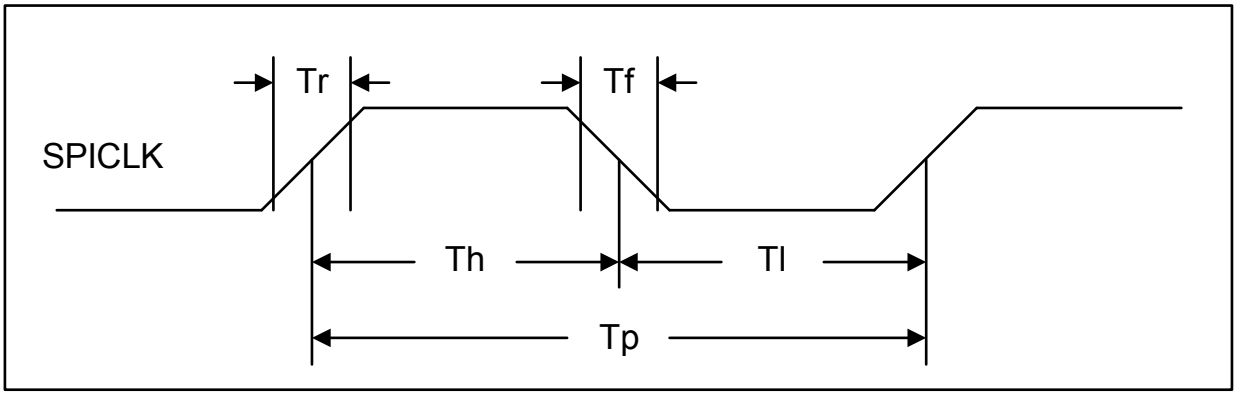
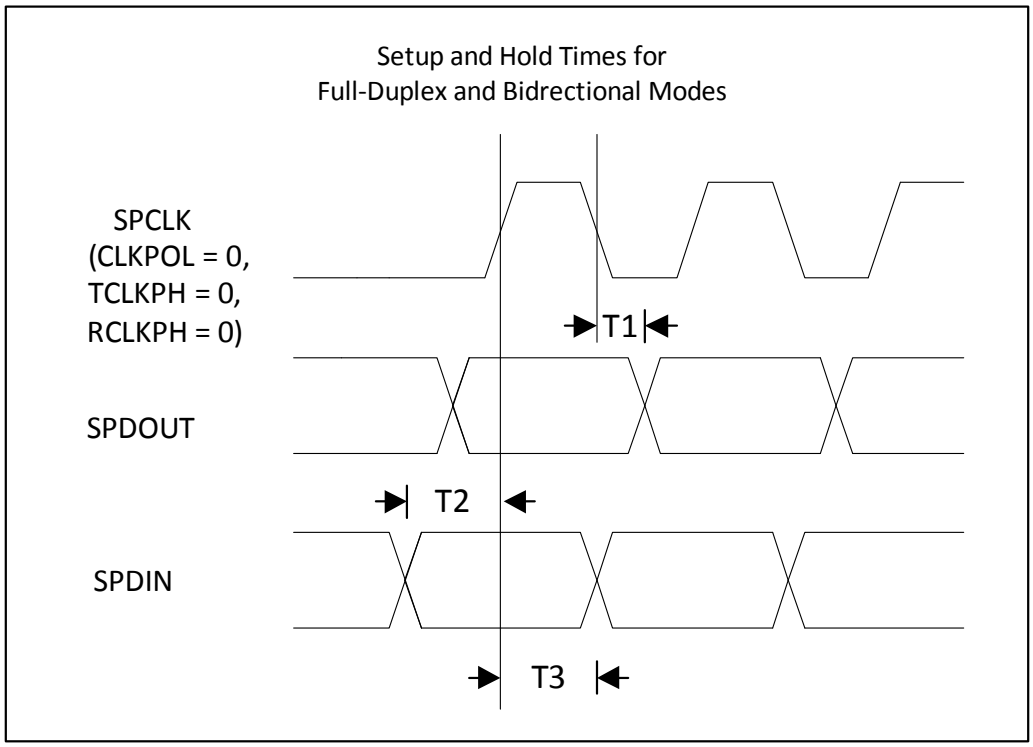


TABLE 38-16: SPI CLOCK TIMING PARAMETERS

Name	Description	MIN	TYP	MAX	Units
Tr	SPI Clock Rise Time. Measured from 10% to 90%.			3	ns
Tf	SPI Clock Fall Time. Measured from 90% to 10%.			3	ns
Th/Tl	SPI Clock High Time/SPI Clock Low Time	40% of SPCLK Period	50% of SPCLK Period	60% of SPCLK Period	ns
Tp	SPI Clock Period – As selected by SPI Clock Generator Register	20.8		62500	ns
Note:	Test conditions are as follows: output load is C_L=30pF, pin drive strength setting is 4mA and slew rate setting is slow.				

FIGURE 38-16: SPI SETUP AND HOLD TIMES



Note:	SPI IO[3:0] obey the SPI_MOSI and SPI_MISO timing. In the 2-pin SPI Interface implementation, SPI_IO0 pin is the SPI Master-Out/Slave-In (MOSI) pin and the SPI_IO1 pin is the Master-In/Slave-out (MISO) pin.

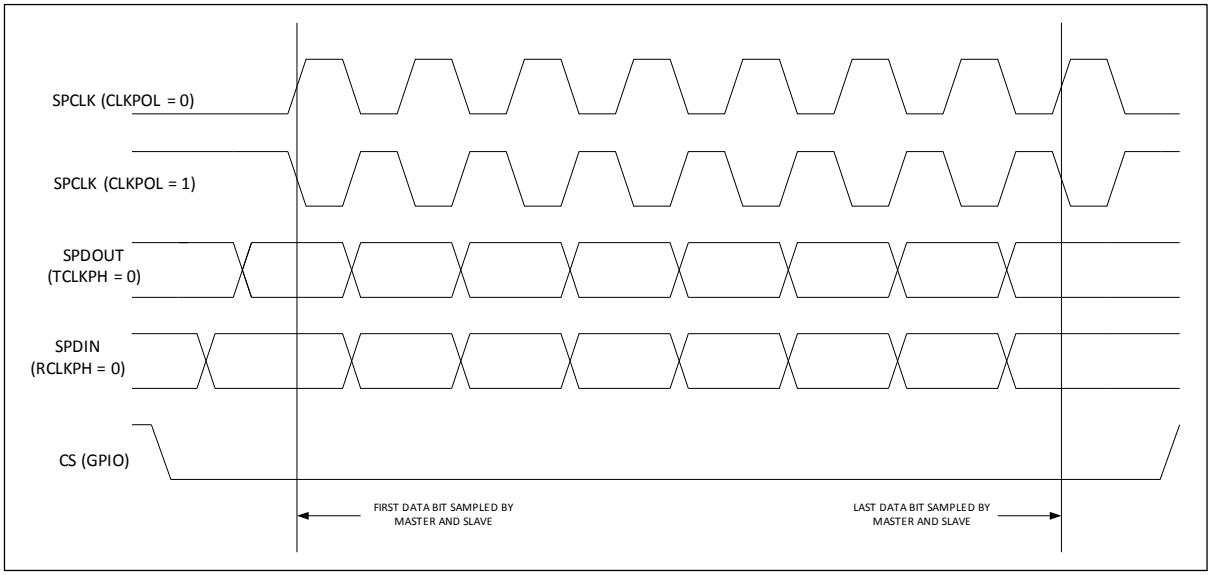
TABLE 38-17: SPI SETUP AND HOLD TIMES PARAMETERS

Name	Description	MIN	TYP	MAX	Units
T1	Data Output Delay			2	ns
T2	Data IN Setup Time	5.5			ns
T3	Data IN Hold Time	0			ns
Note:	Test conditions are as follows: output load is C_L=30pF, pin drive strength setting is 4mA and slew rate setting is slow				

38.14.1 SPI INTERFACE TIMINGS

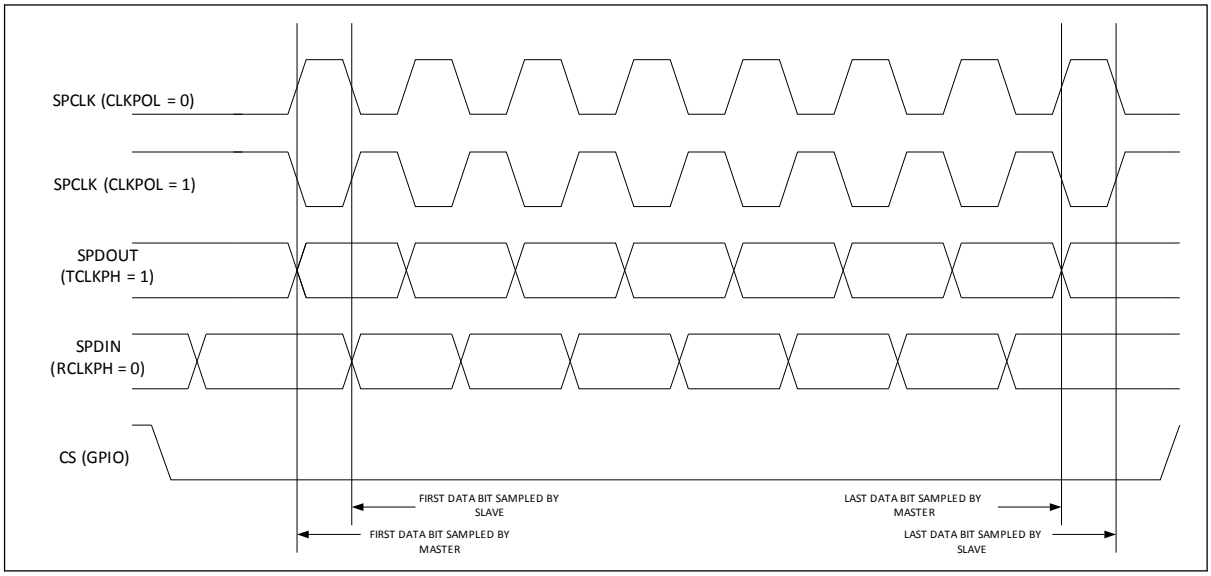
The following timing diagrams represent a single-byte transfer over the SPI interface using different SPCLK phase settings. Data bits are transmitted in bit order starting with the MSB (LSBF='0') or the LSB (LSBF='1'). See the SPI Control Register for information on the LSBF bit. The CS signal in each diagram is a generic bit-controlled chip select signal required by most peripheral devices. This signal and additional chip selects can be GPIO controlled. Note that these timings for Full Duplex Mode are also applicable to Half Duplex (or Bi-directional) mode.

FIGURE 38-17: INTERFACE TIMING, FULL DUPLEX MODE (TCLKPH = 0, RCLKPH = 0)



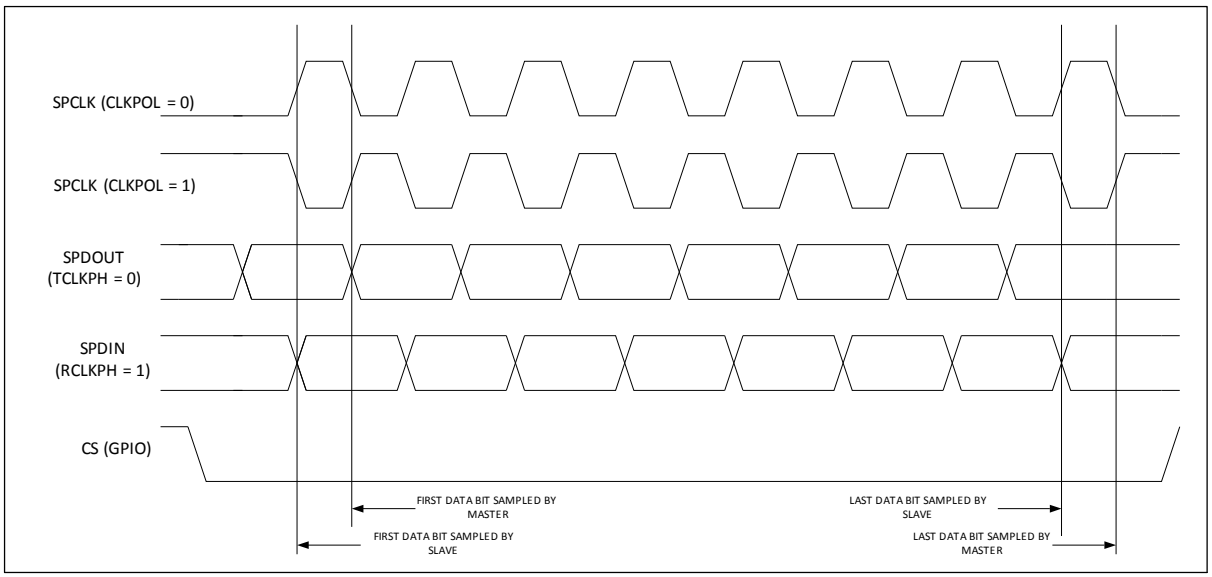
In this mode, data is available immediately when a device is selected and is sampled on the first and following odd SPCLK edges by the master and slave.

FIGURE 38-18: SPI INTERFACE TIMING, FULL DUPLEX MODE (TCLKPH = 1, RCLKPH = 0)



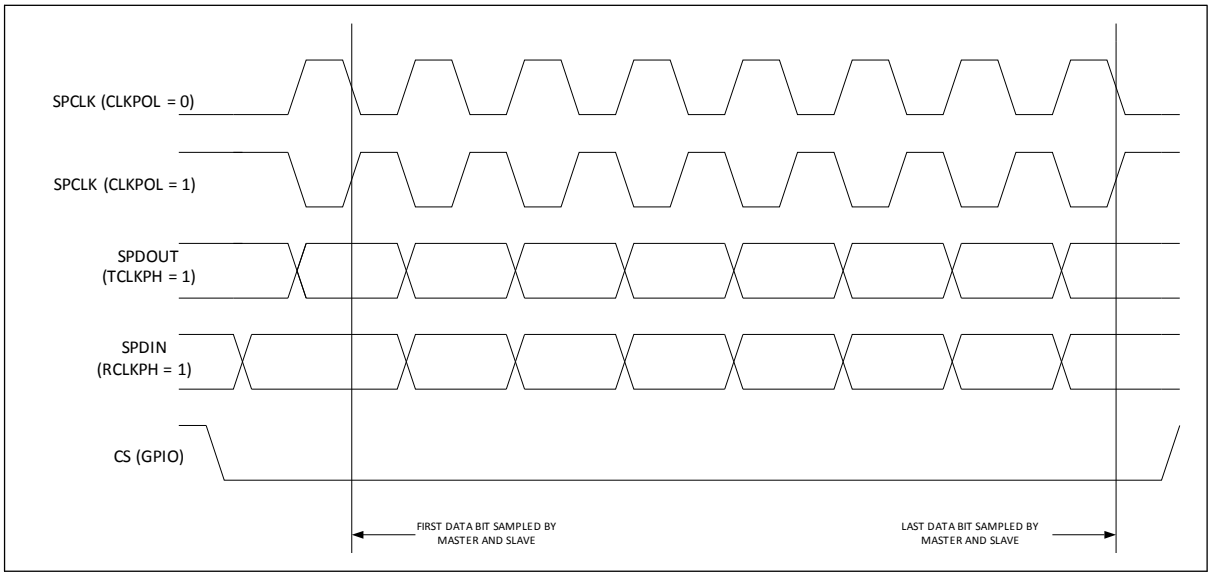
In this mode, the master requires an initial SPCLK edge before data is available. The data from slave is available immediately when the slave device is selected. The.data is sampled on the first and following odd edges by the master. The data is sampled on the second and following even SPCLK edges by the slave.

FIGURE 38-19: SPI INTERFACE TIMING, FULL DUPLEX MODE (TCLKPH = 0, RCLKPH = 1)



In this mode, the data from slave is available immediately when the slave device is selected. The slave device requires an initial SPCLK edge before data is available. The data is sampled on the second and following even SPCLK edges by the master. The data is sampled on the first and following odd edges by the slave.

FIGURE 38-20: SPI INTERFACE TIMING - FULL DUPLEX MODE (TCLKPH = 1, RCLKPH = 1)



In this mode, the master and slave require an initial SPCLK edge before data is available. Data is sampled on the second and following even SPCLK edges by the master and slave.

38.15 Serial Debug Port Timing

FIGURE 38-21: SERIAL DEBUG PORT TIMING PARAMETERS

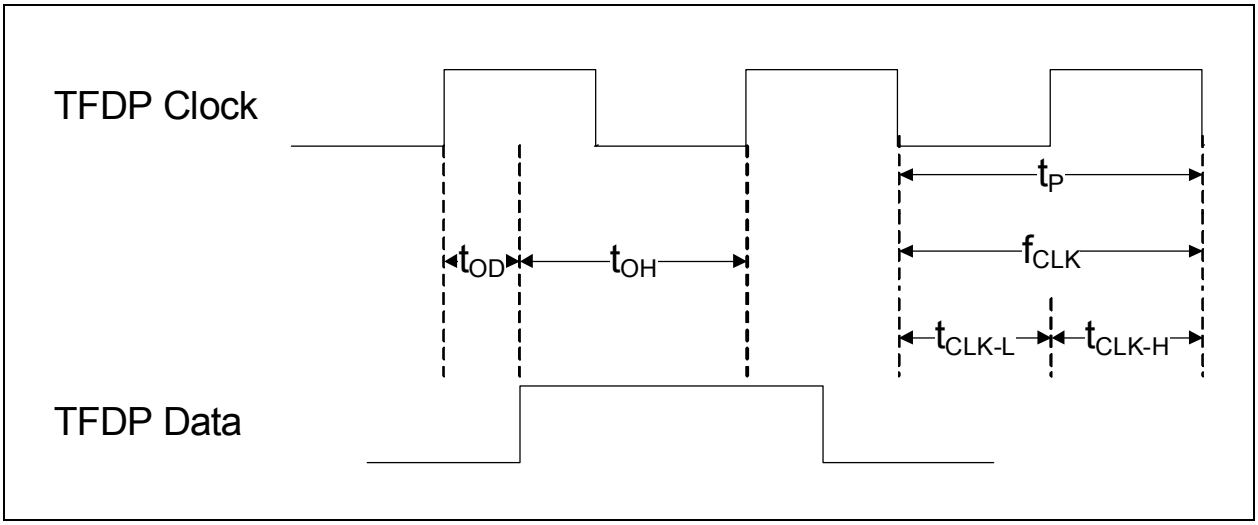


TABLE 38-18: SERIAL DEBUG PORT INTERFACE TIMING PARAMETERS

Name	Description	MIN	TYP	MAX	Units
f_{clk}	TFDP Clock frequency (see note)	2.5	-	24	MHz
t_P	TFDP Clock Period.	1/fclk			µs
t_{OD}	TFDP Data output delay after falling edge of TFDP_CLK.			5	nsec
t_{OH}	TFDP Data hold time after falling edge of TFDP Clock	$t_P - t_{OD}$			nsec
t_{CLK-L}	TFDP Clock Low Time	$t_P/2 - 3$		$t_P/2 + 3$	nsec
t_{CLK-H}	TFDP Clock high Time (see Note 1)	$t_P/2 - 3$		$t_P/2 + 3$	nsec

Note 1: When the clock divider for the embedded controller is an odd number value greater than 2h, then $t_{CLK-L} = t_{CLK-H} + 15$ ns. When the clock divider for the embedded controller is 0h, 1h, or an even number value greater than 2h, then $t_{CLK-L} = t_{CLK-H}$.

38.16 JTAG Interface Timing

FIGURE 38-22: JTAG POWER-UP & ASYNCHRONOUS RESET TIMING

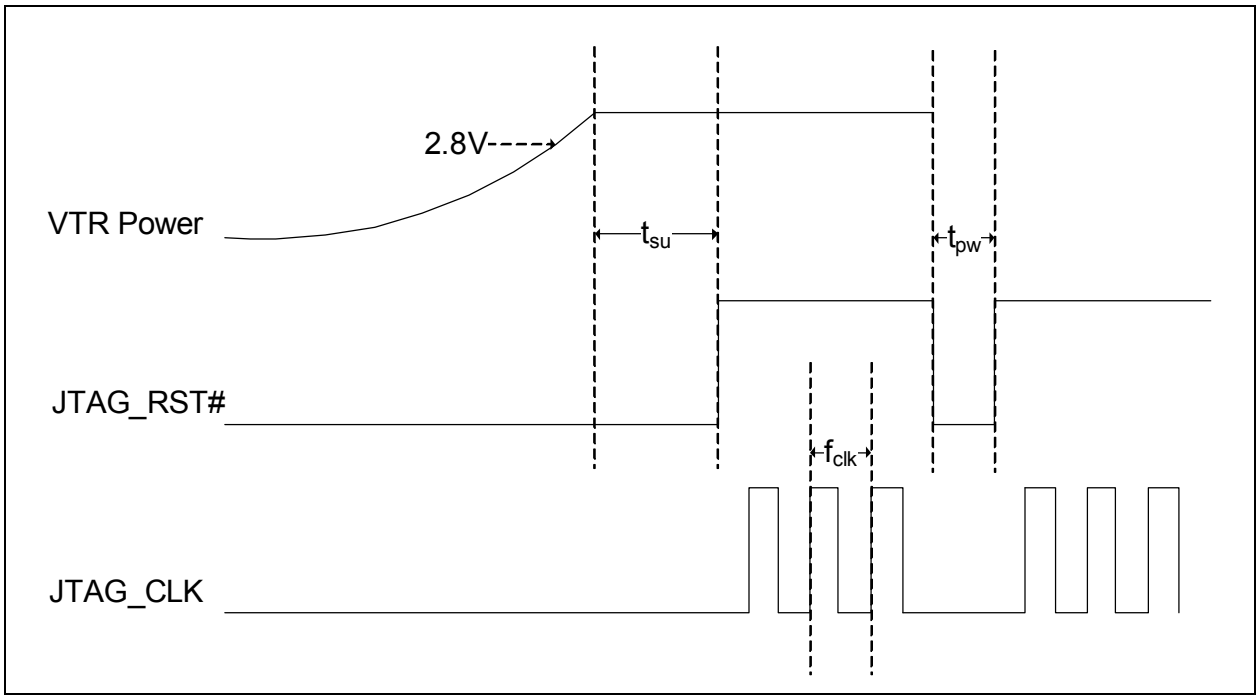


FIGURE 38-23: JTAG SETUP & HOLD PARAMETERS

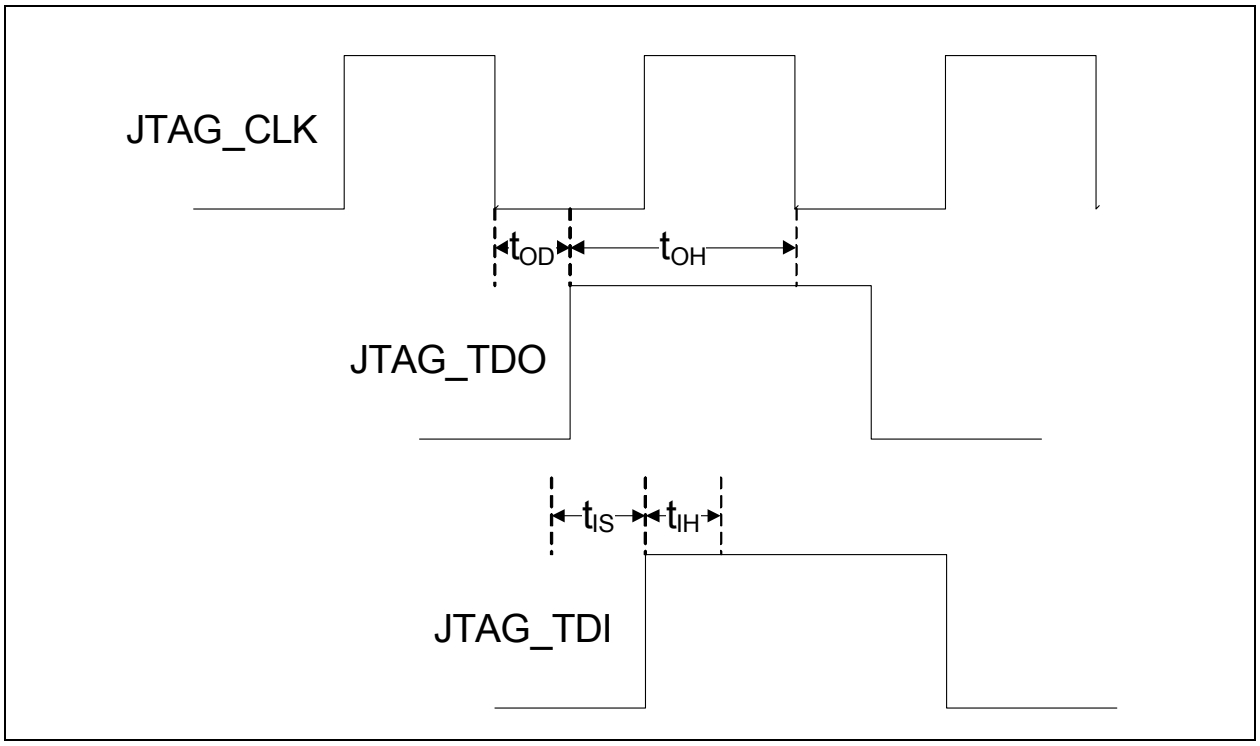


TABLE 38-19: JTAG INTERFACE TIMING PARAMETERS

Name	Description	MIN	TYP	MAX	Units
t_{su}	JTAG_RST# de-assertion after VTR power is applied	5			ms
t_{pw}	JTAG_RST# assertion pulse width	500			nsec
f_{clk}	JTAG_CLK frequency (see note)			48	MHz
t_{OD}	TDO output delay after falling edge of TCLK.	5		10	nsec
t_{OH}	TDO hold time after falling edge of TCLK	1 TCLK - t_{OD}			nsec
t_{IS}	TDI setup time before rising edge of TCLK.	5			nsec
t_{IH}	TDI hold time after rising edge of TCLK.	5			nsec

Note: f_{clk} is the maximum frequency to access a JTAG Register.

39.0 ARM M4F BASED EMBEDDED CONTROLLER

39.1 Introduction

This chapter contains a description of the ARM M4F Embedded Controller (EC).

The EC is built around an ARM® Cortex®-M4F Processor provided by Arm Ltd. (the “ARM M4F IP”). The ARM Cortex® M4F is a full-featured 32-bit embedded processor, implementing the ARMv7-M THUMB instruction set and FPU instruction set in hardware.

The ARM M4F IP is configured as a Von Neumann, Byte-Addressable, Little-Endian architecture. It provides a single unified 32-bit byte-level address, for a total direct addressing space of 4GByte. It has multiple bus interfaces, but these express priorities of access to the chip-level resources (Instruction Fetch vs. Data RAM vs. others), and they do not represent separate addressing spaces.

The ARM M4F is configured as follows.

- **Little-Endian** byte ordering is selected at all times
- **Bit Banding** is included for efficient bit-level access
- **Floating-Point Unit (FPU)** is included, to implement the Floating-Point instruction set in hardware
- **Debug** features are included at “Ex+” level, defined as follows:
  - **DWT** Unit provides 4 Data Watchpoint comparators and Execution Monitoring
  - **FPB** Unit provides HW Breakpointing with 6 Instruction and 2 Literal (Read-Only Data) address comparators. The FPB comparators are also available for Patching: remapping Instruction and Literal Data addresses.
- **Trace** features are included at “Full” level, defined as follows:
  - **DWT** for reporting breakpoints and watchpoints
  - **ITM** for profiling and to timestamp and output messages from instrumented firmware builds
  - **ETM** for instruction tracing, and for enhanced reporting of Core and DWT events
  - The ARM-defined **HTM** trace feature is **not included**
- **NVIC** Interrupt controller with 8 priority levels and up to 240 individually-vectored interrupt inputs
  - A Microchip-defined Interrupt Aggregator function (at chip level) may be used to group multiple interrupts onto single NVIC inputs
  - The ARM-defined **WIC** feature is **not included**. The Microchip Interrupt Aggregator function (at chip level) provides Wake control
- **MPU** (Memory Protection Unit) is included for memory access control
- Single entry **Write Buffer** is incorporated

39.2 References

1. ARM Limited: Cortex®-M4 Technical Reference Manual, DDI0439C, 29 June 2010
2. ARM Limited: ARM®v7-M Architecture Reference Manual, DDI0403D, November 2010
3. NOTE: Filename DDI0403D_arm_architecture_v7m_reference_manual_errata_markup_1_0.pdf
4. ARM® Generic Interrupt Controller Architecture version 1.0 Architecture Specification, IHI0048A, September 2008
5. ARM Limited: AMBA® Specification (Rev 2.0), IHI0011A, 13 May 1999
6. ARM Limited: AMBA® 3 AHB-Lite Protocol Specification, IHI0033A, 6 June 2006
7. ARM Limited: AMBA® 3 ATB Protocol Specification, IHI0032A, 19 June 2006
8. ARM Limited: Cortex-M™ System Design Kit Technical Reference Manual, DDI0479B, 16 June 2011
9. ARM Limited: CoreSight™ v1.0 Architecture Specification, IHI0029B, 24 March 2005
10. ARM Limited: CoreSight™ Components Technical Reference Manual, DDI0314H, 10 July 2009
11. ARM Limited: ARM® Debug Interface v5 Architecture Specification, IHI0031A, 8 February 2006
12. ARM Limited: ARM® Debug Interface v5 Architecture Specification ADIv5.1 Supplement, DSA09-PRDC-008772, 17 August 2009
13. ARM Limited: Embedded Trace Macrocell™ (ETMv1.0 to ETMv3.5) Architecture Specification, IHI0014Q, 23 September 2011
14. ARM Limited: CoreSight™ ETM™-M4 Technical Reference Manual, DDI0440C, 29 June 2010

39.3 Terminology

39.3.1 ARM IP TERMS AND ACRONYMS

- AHB

  Advanced High-Performance Bus, a system-level on-chip **AMBA 2** bus standard. See Reference[5], ARM Limited: AMBA® Specification (Rev 2.0), IHI0011A, 13 May 1999.

- AHB-AP

  AHB Access Port, the **AP** option selected by Microchip for the **DAP**

- AHB-Lite

  A Single-Master subset of the **AHB** bus standard: defined in the **AMBA 3** bus standard. See Reference[6], ARM Limited: AMBA® 3 AHB-Lite Protocol Specification, IHI0033A, 6 June 2006.

- AMBA

  The collective term for bus standards originated by ARM Limited.

  AMBA 3 defines the IP's **AHB-Lite** and **ATB** bus interfaces.

  AMBA 2 (AMBA Rev. 2.0) defines the EC's **AHB** bus interface.

- AP

  Any of the ports on the **DAP** subblock for accessing on-chip resources on behalf of the Debugger, independent of processor operations. A single **AHB-AP** option is currently selected for this function.

- APB

  Advanced Peripheral Bus, a limited 32-bit-only bus defined in **AMBA 2** for I/O register accesses. This term is relevant only to describe the **PPB** bus internal to the EC core. See Reference [5], ARM Limited: AMBA® Specification (Rev 2.0), IHI0011A, 13 May 1999.

- ARMv7

  The identifying name for the general architecture implemented by the **Cortex-M** family of IP products.

  The **ARMv7** architecture has no relationship to the older "ARM 7" product line, which is classified as an "ARMv3" architecture, and is very different.

- ATB

  Interface standard for Trace data to the **TPIU** from **ETM** and/or **ITM** blocks, Defined in **AMBA 3**. See Reference[7], ARM Limited: AMBA® 3 ATB Protocol Specification, IHI0032A, 19 June 2006.

- Cortex-M4F

  The ARM designation for the specific IP selected for this product: a Cortex M4 processor core containing a hardware Floating Point Unit (FPU).

- DAP

  Debug Access Port, a subblock consisting of **DP** and **AP** subblocks.

- DP

  Any of the ports in the **DAP** subblock for connection to an off-chip Debugger. A single **SWJ-DP** option is currently selected for this function, providing **JTAG** connectivity.

- DWT

  Data Watchdog and Trace subblock. This contains comparators and counters used for data watchpoints and Core activity tracing.

- ETM

  Embedded Trace Macrocell subblock. Provides enhancements for Trace output reporting, mostly from the **DWT** subblock. It adds enhanced instruction tracing, filtering, triggering and timestamping.

- FPB

  FLASH Patch Breakpoint subblock. Provides either Remapping (Address substitution) or Breakpointing (Exception or Halt) for a set of Instruction addresses and Data addresses. See Section 8.3 of Reference [1], ARM Limited: Cortex®-M4 Technical Reference Manual, DDI0439C, 29 June 2010.

- FPU

  Floating-Point Unit: a subblock included in the Core for implementing the Floating Point instruction set in hardware.

- HTM

  AHB Trace Macrocell. This is an optional subblock that is **not included**.

- ITM

  Instrumentation Trace Macrocell subblock. Provides a HW Trace interface for “printf”-style reports from instrumented firmware builds, with timestamping also provided.

- MEM-AP

  A generic term for an **AP** that connects to a memory-mapped bus on-chip. For this product, this term is synonymous with the AHB Access Port, **AHB-AP.**

- MPU

  Memory Protection Unit.

- NVIC

  Nested Vectored Interrupt Controller subblock. Accepts external interrupt inputs. See References [2], ARM Limited: ARM®v7-M Architecture Reference Manual, DDI0403D, November 2010 and [4], ARM® Generic Interrupt Controller Architecture version 1.0 Architecture Specification, IHI0048A, September 2008.

- PPB

  Private Peripheral Bus: A specific **APB** bus with local connectivity within the EC.

- ROM Table

  A ROM-based data structure in the Debug section that allows an external Debugger and/or a FW monitor to determine which of the Debug features are present.

- SWJ-DP

  Serial Wire / **JTAG** Debug Port, the **DP** option selected by Microchip for the **DAP**.

- TPA

  Trace Port Analyzer: any off-chip device that uses the TPIU output.

- TPIU

  Trace Port Interface Unit subblock. Multiplexes and buffers Trace reports from the ETM and ITM subblocks.

- WIC

  Wake-Up Interrupt Controller. This is an optional subblock that is **not included**.

39.3.2 MICROCHIP TERMS AND ACRONYMS

- Interrupt Aggregator

  This is a module that may be present at the chip level, which can combine multiple interrupt sources onto single interrupt inputs at the EC, causing them to share a vector.

- PMU

  Processor Memory Unit, this is a module that may be present at the chip level containing any memory resources that are closely-coupled to the CEC1702 EC. It manages accesses from both the EC processor and chip-level bus masters.

39.4 ARM M4F IP Interfaces

This section defines only the interfaces to the ARM IP itself. For the interfaces of the entire block, see Section 39.5, "Block External Interfaces".

The CEC1702 IP has the following major external interfaces, as shown in Figure 39-1, "ARM M4F Based Embedded Controller I/O Block Diagram":

- ICode AHB-Lite Interface
- DCode AHB-Lite Interface
- System AHB-Lite Interface

- Debug (JTAG) Interface
- Trace Port Interface
- Interrupt Interface

The EC operates on the model of a single 32-bit addressing space of byte addresses (4Gbytes, Von Neumann architecture) with Little-Endian byte ordering. On the basis of an internal decoder (part of the Bus Matrix shown in Figure 39-1), it routes Read/Write/Fetch accesses to one of three external interfaces, or in some cases internally (shown as the PPB interface).

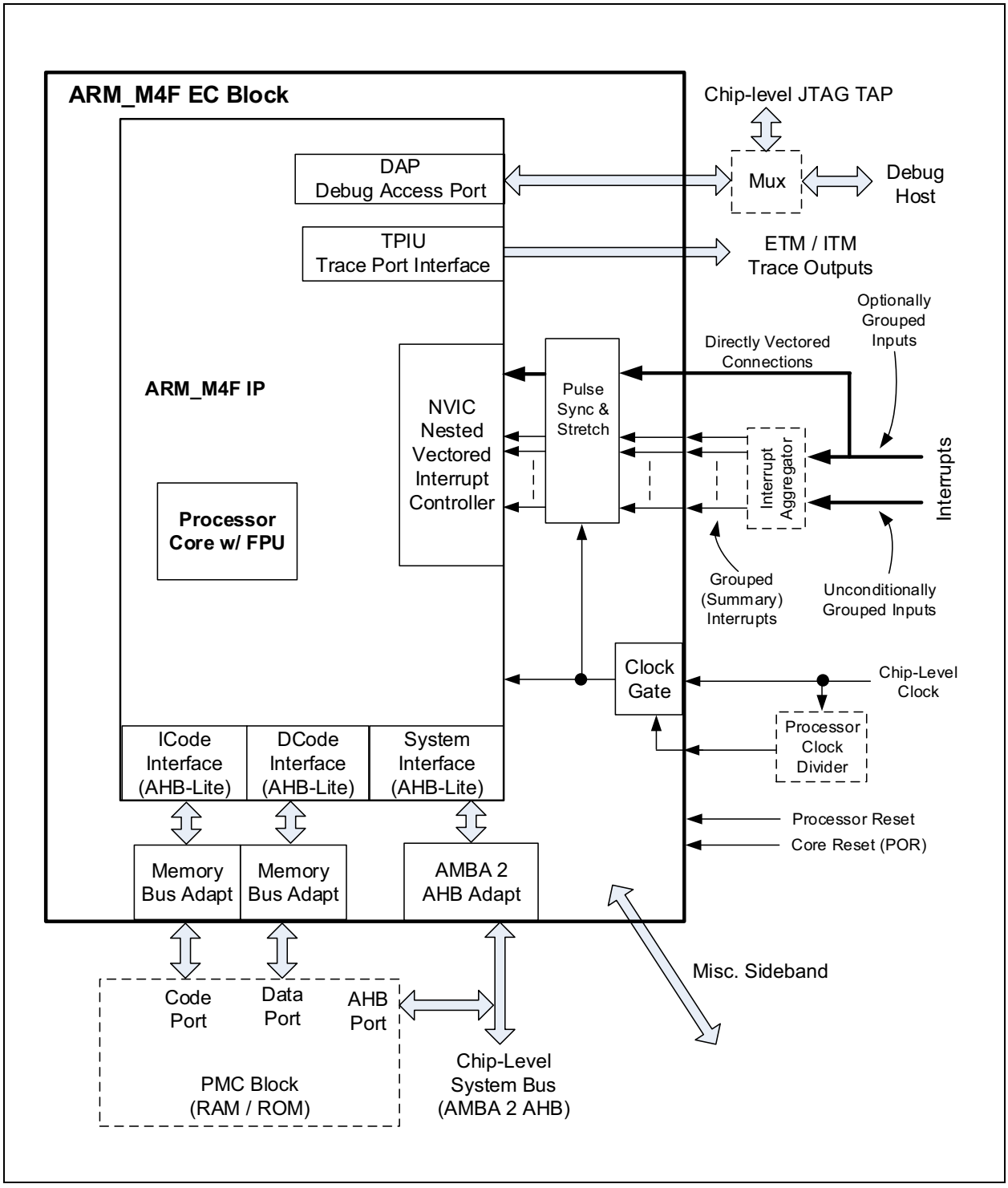
The EC executes instructions out of closely-coupled memory via the ICode Interface. Data accesses to closely-coupled memory are handled via the DCode Interface. The EC accesses the rest of the on-chip address space via the System AHB-Lite interface. The Debugger program in the host can probe the EC and all EC addressable memory via the JTAG debug interface.

Aliased addressing spaces are provided at the chip level so that specific bus interfaces can be selected explicitly where needed. For example, the EC's Bit Banding feature uses the System AHB-Lite bus to access resources normally accessed via the DCode or ICode interface.

Note:	The EC executes most instructions in 1 clock cycle. If an instruction accesses code and data that are in different RAM blocks, then it takes one clock cycle to access both code and data (done in parallel). However, if the code and data blocks are in the same RAM block, then it takes two clock cycles (one clock for code access and one clock for data access) since it must do it sequentially.

39.5 Block External Interfaces

FIGURE 39-1: ARM M4F BASED EMBEDDED CONTROLLER I/O BLOCK DIAGRAM



39.6 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

39.6.1 POWER DOMAINS

TABLE 39-1: POWER SOURCES

Name	Description
VTR	The ARM M4F Based Embedded Controller is powered by VTR.

39.6.2 CLOCK INPUTS

39.6.2.1 Basic Clocking

The basic clocking comes from a free-running Clock signal provided from the chip level.

TABLE 39-2: CLOCK INPUTS

Name	Description
48MHz	The clock source to the EC. Division of the clock rate is determined by the PROCESSOR_CLOCK_DIVIDE field in the Processor Clock Control Register.

39.6.2.2 System Tick Clocking

The System Tick clocking is controlled by a signal from chip-level logic. It is the 48MHz divided by the following:

- ((PROCESSOR_CLOCK_DIVIDE)x2)+1

39.6.2.3 Debug JTAG Clocking

The Debug JTAG clocking comes from chip-level logic, which may multiplex or gate this clock. See Section 39.10, "Debugger Access Support".

39.6.2.4 Trace Clocking

The Clock for the Trace interface is identical to the 48MHz input.

39.6.3 RESETS

The reset interface from the chip level is given below.

TABLE 39-3: RESET SIGNALS

Name	Description
RESET_EC	The ARM M4F Based Embedded Controller is reset by RESET_EC.

39.7 Interrupts

The ARM M4F Based Embedded Controller is equipped with an Interrupt Interface to respond to interrupts. These inputs go to the IP's NVIC block after a small amount of hardware processing to ensure their detection at varying clock rates. See Figure 39-1, "ARM M4F Based Embedded Controller I/O Block Diagram".

As shown in Figure 39-1, an Interrupt Aggregator block may exist at the chip level, to allow multiple related interrupts to be grouped onto the same NVIC input, and so allowing them to be serviced using the same vector. This may allow the same interrupt handler to be invoked for a group of related interrupt inputs. It may also be used to expand the total number of interrupt inputs that can be serviced.

The NMI (Non-Maskable Interrupt) connection is tied off and not used.

39.7.1 NVIC INTERRUPT INTERFACE

The NVIC interrupt unit can be wired to up to 240 interrupt inputs from the chip level. The interrupts that are actually connected from the chip level are defined in the Interrupt section.

All NVIC interrupt inputs can be programmed as either pulse or level triggered. They can also be individually masked, and individually assigned to their own hardware-managed priority level.

39.7.2 NVIC RELATIONSHIP TO EXCEPTION VECTOR TABLE ENTRIES

The Vector Table consists of 4-byte entries, one per vector. Entry 0 is not a vector, but provides an initial Reset value for the Main Stack Pointer. Vectors start with the Reset vector, at Entry #1. Entries up through #15 are dedicated for internal exceptions, and do not involve the NVIC.

NVIC entries in the Vector Table start with Entry #16, so that NVIC Interrupt #0 is at Entry #16, and all NVIC interrupt numbers are incremented by 16 before accessing the Vector Table.

The number of connections to the NVIC determines the necessary minimum size of the Vector Table, as shown below. It can extend as far as 256 entries (255 vectors, plus the non-vector entry #0).

A Vector entry is used to load the Program Counter (PC) and the EPSR.T bit. Since the Program Counter only expresses code addresses in units of two-byte Halfwords, bit[0] of the vector location is used to load the EPSR.T bit instead, selecting THUMB mode for exception handling. Bit[0] must be '1' in all vectors, otherwise a UsageFault exception will be posted (INVSTATE, unimplemented instruction set). If the Reset vector is at fault, the exception posted will be HardFault instead.

TABLE 39-4: EXCEPTION AND INTERRUPT VECTOR TABLE LAYOUT

Table Entry	Exception Number	Exception
Special Entry for Reset Stack Pointer		
0	(none)	Holds Reset Value for the Main Stack Pointer. Not a Vector.
Core Internal Exception Vectors start here		
1	1	Reset Vector (PC + EPSR.T bit)
2	2	NMI (Non-Maskable Interrupt) Vector
3	3	HardFault Vector
4	4	MemManage Vector
5	5	BusFault Vector
6	6	UsageFault Vector
7	(none)	(Reserved by ARM Ltd.)
8	(none)	(Reserved by ARM Ltd.)
9	(none)	(Reserved by ARM Ltd.)
10	(none)	(Reserved by ARM Ltd.)
11	11	SVCall Vector
12	12	Debug Monitor Vector
13	(none)	(Reserved by ARM Ltd.)
14	14	PendSV Vector
15	15	SysTick Vector
NVIC Interrupt Vectors start here		
16	16	NVIC Interrupt #0 Vector
. . .	. . .	. . .
n + 16	n + 16	NVIC Interrupt #n Vector
. . .	. . .	. . .
max + 16	max + 16	NVIC Interrupt #max Vector (Highest-numbered NVIC connection.)
. . .	. . .	. Table size may (but need not) extend further. . .
255	255	NVIC Interrupt #239 (Architectural Limit of Exception Table)

39.8 Low Power Modes

The ARM processor can enter Sleep or Deep Sleep modes internally. This action will cause an output signal Clock Required to be turned off, allowing clocks to be stopped from the chip level. However, Clock Required will still be held active, or set to active, unless all of the following conditions exist:

- No interrupt is pending.
- An input signal Sleep Enable from the chip level is active.
- The Debug JTAG port is inactive (reset or configured not present).

In addition, regardless of the above conditions, a chip-level input signal Force Halt may halt the processor and remove Clock Required.

39.9 Description

39.9.1 BUS CONNECTIONS

There are three bus connections used from CEC1702 EC block, which are directly related to the IP bus ports. See Figure 39-1, "ARM M4F Based Embedded Controller I/O Block Diagram".

For the mapping of addresses at the chip level, see Section 3.0, "Device Inventory".

39.9.1.1 Closely Coupled Instruction Fetch Bus

As shown in Figure 39-1, the AHB-Lite ICode port from the IP is converted to a more conventional SRAM memory-style bus and connected to the on-chip memory resources with routing priority appropriate to Instruction Fetches.

39.9.1.2 Closely Coupled Data Bus

As shown in Figure 39-1, the AHB-Lite DCode port from the IP is converted to a more conventional SRAM memory-style bus and connected to the on-chip memory resources with routing priority appropriate to fast Data Read/Write accesses.

39.9.1.3 Chip-Level System Bus

As shown in Figure 39-1, the AHB-Lite System port from the IP is converted from AHB-Lite to fully arbitrated multi-master capability (the AMBA 2 defined AHB bus: see Reference [5], ARM Limited: AMBA® Specification (Rev 2.0), IHI0011A, 13 May 1999). Using this bus, all addressable on-chip resources are available. The multi-mastering capability supports the Microchip DMA and EMI features if present, as well as the Bit-Banding feature of the IP itself.

As also shown in Figure 39-1, the Closely-Coupled memory resources are also available through this bus connection using aliased addresses. This is required in order to allow Bit Banding to be used in these regions, but it also allows them to be accessed by DMA and other bus masters at the chip level.

Note:	Registers with properties such as Write-1-to-Clear (W1C), Read-to-Clear and FIFOs need to be handled with appropriate care when being used with the bit band alias addressing scheme. Accessing such a register through a bit band alias address will cause the hardware to perform a read-modify-write, and if a W1C-type bit is set, it will get cleared with such an access. For example, using a bit band access to the Interrupt Aggregator, including the Interrupt Enables and Block Interrupt Status to clear an IRQ will clear all active IRQs.

39.9.2 INSTRUCTION PIPELINING

There are no special considerations except as defined by ARM documentation.

39.10 Debugger Access Support

An external Debugger accesses the chip through a JTAG standard interface. The ARM Debug Access Port supports both the 2-pin SWD (Serial Wire Debug) interface and the 4-pin JTAG interface.

As shown in Figure 39-1, "ARM M4F Based Embedded Controller I/O Block Diagram", other resources at the chip level that share the JTAG port pins; for example chip-level Boundary Scan.

By default, debug access is disabled when the EC begins executing code. EC code enables debugging by writing the Debug Enable Register in the EC Subsystem Registers block.

TABLE 39-5: ARM JTAG ID

ARM Debug Mode	JTAG ID
SW-DP (2-wire)	0x2BA01477
JTAG (4-wire)	0x4BA00477

39.10.1 DEBUG AND ACCESS PORTS (SWJ-DP AND AHB-AP SUBBLOCKS)

These two subblocks work together to provide access to the chip for the Debugger using the Debug JTAG connection, as described in Chapter 4 of the ARM Limited: ARM® Debug Interface v5 Architecture Specification, IHI0031A, 8 February 2006.

39.10.2 BREAKPOINT, WATCHPOINT AND TRACE SUPPORT

See References [11], ARM Limited: ARM® Debug Interface v5 Architecture Specification, IHI0031A, 8 February 2006 and [12], ARM Limited: ARM® Debug Interface v5 Architecture Specification ADIv5.1 Supplement, DSA09-PRDC-008772, 17 August 2009. A summary of functionality follows.

Breakpoint and Watchpoint facilities can be programmed to do one of the following:

- Halt the processor. This means that the external Debugger will detect the event by periodically polling the state of the EC.
- Transfer control to an internal Debug Monitor firmware routine, by triggering the Debug Monitor exception (see Table 39-4, "Exception and Interrupt Vector Table Layout").

39.10.2.1 Instrumentation Support (ITM Subblock)

The Instrumentation Trace Macrocell (ITM) is for profiling software. This uses non-blocking register accesses, with a fixed low-intrusion overhead, and can be added to a Real-Time Operating System (RTOS), application, or exception handler. If necessary, product code can retain the register access instructions, avoiding probe effects.

39.10.2.2 HW Breakpoints and ROM Patching (FPB Subblock)

The Flash Patch and Breakpoint (FPB) block. This block can remap sections of ROM, typically Flash memory, to regions of RAM, and can set breakpoints on code in ROM. This block can be used for debug, and to provide a code or data patch to an application that requires field updates to a product in ROM.

39.10.2.3 Data Watchpoints and Trace (DWT Subblock)

The Debug Watchpoint and Trace (DWT) block provides watchpoint support, program counter sampling for performance monitoring, and embedded trace trigger control.

39.10.2.4 Trace Interface (ETM and TPIU)

The Embedded Trace Macrocell (ETM) provides instruction tracing capability. For details of functionality and usage, see References [13], ARM Limited: Embedded Trace Macrocell™ (ETMv1.0 to ETMv3.5) Architecture Specification, IHI0014Q, 23 September 2011 and [14], ARM Limited: CoreSight™ ETM™-M4 Technical Reference Manual, DDI0440C, 29 June 2010.

The Trace Port Interface Unit (TPIU) provides the external interface for the ITM, DWT and ETM.

39.11 Delay Register

39.11.1 DELAY REGISTER

Offset	1000_0000h			
Bits	**Description**	**Type**	**Default**	**Reset Event**
31:5	Reserved	R	-	-
4:0	DELAY Writing a value *n,* from 0h to 31h, to this register will cause the ARM processor to stall for (*n*+1) microseconds (that is, from 1µS to 32µS). Reads will return the last value read immediately. There is no delay.	R/W	0h	RESET_SYS

APPENDIX A: DATA SHEET REVISION HISTORY

Revision	Section/Figure/Entry	Correction
DS00002207E (08-23-19)	Several template pages updated.	
	Updated **Section 3.7 "Register Map"**	Address 4000FC38h marked reserved.
	Updated Section 33.8.5	OTP Lock Register exposed to customer
	Updated Section	exposed to customer
	Updated Section 27.12.3	Added ACTIVE bit explanation in Section 27.12.3
DS00002207D (03-13-19)	Updated DC Electrical Characteristics definition	Fixed the Symbol of Pull up current and Pull down current to Rup and Rdn in Table 37-3
	Updated BGPO Power Register definition	"The GPIO input register for the GPIO that is multiplexed with the BGPO always returns a '1b'" is replaced with "The GPIO Input register for the GPIO that is multiplexed with the BGPO always returns the value of the pin associated with BGPO[*i*]."
	Table 4-1 updated	Device Revision Register name updated in Table 4-1. Register name will not contain Hard Wire
	Updated VCI_OUT logic figure	Updated Figure 30-2 and removed latch from the diagram. Design does not contain the latch.
	Updated Ball Map table	Updated Ball Map number and missing information in the tables.
	Added Note in eFuse chapter	Added Note to storing Encrypted data in eFuse and VBAT registers for best security.
DS00002207C (07-18-18)	Updated note following Table 4-1, "Chip-Level (Global) Control/Configuration Registers"	"Device ID for Device Revision values are in Table 4-1. Refer to errata sheet for the Device Revision value."
	2.7 "Signal Description by Interface"	Changed VFLT_PLL definition to "Filter for PLL".
	TABLE 5-1: "Power Source Definitions"	Removed VFLT_PLL as it is not a power pin.
	TABLE 37-12: "VBAT Supply Current, I_VBAT (VBAT=3.0V)"	Updated eSPI power numbers to table.
	FIGURE 38-5: "Clock Timing Diagram"	Updated Clock Timing diagram. removed $T_{ADJ.}$
	TABLE 38-4: "Clock Timing Parameters"	Updated t_{su} description. Updated Start-up delay from 0k Hz to Operating Frequency.
	Table 37-10: "Thermal Packaging Characteristics"	Moved XFBGA to WFBGA, Added package information SZ, TN and XY.
	Chapter 2	Changed nSYS_RST to RESET_SYS. Port 80 Host R/W Access Type table updated.

Revision	Section/Figure/Entry	Correction
	WDT_EVENT_COUNT description updated	In Section 33.8.6, "WDT Event Count Register", updated the description of WDT_EVENT_COUNT register.
	Updated PCR Power Reset Status Register bit definitions	JTAG_RST#, RESET_SYS_STATUS and VBAT_RESET_STATUS register field definition updated in Section 5.9.8, "PCR Power Reset Status Register".
	Added Note in Efuse programming section	Added Note in Section 36.10.1, "Control Register" to make sure that both FSOURCE_EN_READ and FSOURCE_EN_PRGM change state together.
DS00002207C (10-31-17)	Document Features and Section 34.10.2, "Cryptographic Hashing," on page 353	Added SHA-384.
	Product Identification System on page 417	Changed CEC1702 Product Identification to “Cryptographic Embedded Controller” and B version to “Not Provisioned Version”.
	Section 37.2.3, "Capacitive Loading Specifications," on page 373	Changed “VCC” to “VTR”.
	Section 36.7.1, "eFUSE Programming Sequence," on page 365	Programming voltage corrected to be in the range of 1.52V to 1.60.
	Section 5.4.2, "Battery Circuit Requirements," on page 69	“Ground” changed to “Discharge”.
	Removed “BGND” signal from data sheet.	
	Section 2.4.1, "Default State," on page 9	Changed default of GPIO064 to “LRESET#”.
	Section 36.9, "eFuse Memory Map," on page 366	Modified eFuse block.
	Section 33.8.7, "GPIO Bank Power Register," on page 347	Added note below section.
	Table 37-3, “DC Electrical Characteristics,” on page 374	Updated I0L and I0H values.
	Table 9-5, “Reset Signals,” on page 112 and Section 9.11.2, "Configuration Select Register," on page 127	Removed “RESET_VCC” from table. Changed bit1 to reserved and added note to customer to change this register to “0”.
DS00002207B (03-03-17)	Updated to reflect Functional Rev C	
DS00002207A (07-19-16)	Document Release	

THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at www.microchip.com. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or field application engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://microchip.com/support

PRODUCT IDENTIFICATION SYSTEM

Not all of the possible combinations of Device, Temperature Range and Package may be offered for sale. To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

PART NO.[1]	-	X	-	XX	-	X[4]/XXX[2]	-	[X][3]
Device		Total SRAM		Version/ Revision		Temp Range/ Package		Tape and Reel Option

Device:	CEC1702[1]	Cryptographic Embedded Controller
Total SRAM	Q	480KB
Version/ Revision:	B#	B = Non Provisioned Version, # = Version Revision Number
Temperature Range	Blank = 0°C to +70°C (Commercial) I/ = -40°C to +85°C (Industrial)	
Package:	SX	84 pin WFBGA[2], 7mm x 7mm body, 0.65mm pitch
Tape and Reel Option:	Blank = Tray packaging TR = Tape and Reel[3]	

Examples:

a) CEC1702Q-B2-I/SX = CEC1702, 480KB total SRAM, Non Provisioned part, Version Revision 2, Industrial Temp ,84-WFBGA, Tray packaging.

Note 1: These products meet the halogen maximum concentration values per IEC61249-2-21.

2: All package options are RoHS compliant. For RoHS compliance and environmental information, please visit http://www.microchip.com/pagehandler/en-us/aboutus/ehs.html

3: Tape and Reel identifier only appears in the catalog part number description. This identifier is used for ordering purposes and is not printed on the device package. Check with your Microchip Sales Office for package availability with the Tape and Reel option.

4: Industrial Temperature is supported by CEC1702.

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

ISBN: 9781522449447

For information regarding Microchip's Quality Management Systems, please visit www.microchip.com/quality.

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Austin, TX
Tel: 512-257-3370

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453
Tel: 317-536-2380

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608
Tel: 951-273-7800

Raleigh, NC
Tel: 919-844-7510

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110
Tel: 408-436-4270

Canada - Toronto
Tel: 905-695-1980
Fax: 905-695-2078

ASIA/PACIFIC

Australia - Sydney
Tel: 61-2-9868-6733

China - Beijing
Tel: 86-10-8569-7000

China - Chengdu
Tel: 86-28-8665-5511

China - Chongqing
Tel: 86-23-8980-9588

China - Dongguan
Tel: 86-769-8702-9880

China - Guangzhou
Tel: 86-20-8755-8029

China - Hangzhou
Tel: 86-571-8792-8115

China - Hong Kong SAR
Tel: 852-2943-5100

China - Nanjing
Tel: 86-25-8473-2460

China - Qingdao
Tel: 86-532-8502-7355

China - Shanghai
Tel: 86-21-3326-8000

China - Shenyang
Tel: 86-24-2334-2829

China - Shenzhen
Tel: 86-755-8864-2200

China - Suzhou
Tel: 86-186-6233-1526

China - Wuhan
Tel: 86-27-5980-5300

China - Xian
Tel: 86-29-8833-7252

China - Xiamen
Tel: 86-592-2388138

China - Zhuhai
Tel: 86-756-3210040

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444

India - New Delhi
Tel: 91-11-4160-8631

India - Pune
Tel: 91-20-4121-0141

Japan - Osaka
Tel: 81-6-6152-7160

Japan - Tokyo
Tel: 81-3-6880- 3770

Korea - Daegu
Tel: 82-53-744-4301

Korea - Seoul
Tel: 82-2-554-7200

Malaysia - Kuala Lumpur
Tel: 60-3-7651-7906

Malaysia - Penang
Tel: 60-4-227-8870

Philippines - Manila
Tel: 63-2-634-9065

Singapore
Tel: 65-6334-8870

Taiwan - Hsin Chu
Tel: 886-3-577-8366

Taiwan - Kaohsiung
Tel: 886-7-213-7830

Taiwan - Taipei
Tel: 886-2-2508-8600

Thailand - Bangkok
Tel: 66-2-694-1351

Vietnam - Ho Chi Minh
Tel: 84-28-5448-2100

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

Finland - Espoo
Tel: 358-9-4520-820

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Garching
Tel: 49-8931-9700

Germany - Haan
Tel: 49-2129-3766400

Germany - Heilbronn
Tel: 49-7131-72400

Germany - Karlsruhe
Tel: 49-721-625370

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Germany - Rosenheim
Tel: 49-8031-354-560

Israel - Ra'anana
Tel: 972-9-744-7705

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Italy - Padova
Tel: 39-049-7625286

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Norway - Trondheim
Tel: 47-7288-4388

Poland - Warsaw
Tel: 48-22-3325737

Romania - Bucharest
Tel: 40-21-407-87-50

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Gothenberg
Tel: 46-31-704-60-40

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820