PSoC CY8C20x34 TRM

PSoC® CY8C20x34
PSoC® CY8C20x24

Technical Reference Manual (TRM)

Document No. 001-13033 Rev. *A

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone (USA): 800.858.1810
Phone (Intnl): 408.943.2600

http://www.cypress.com

Copyrights

PSoC® is a registered trademark and CapSense™, PSoC Designer™, Programmable System-on-Chip™, and PSoC Express™ are trademarks of Cypress Semiconductor Corporation (Cypress), along with Cypress® and Cypress Semiconductor™. All other trademarks or registered trademarks referenced herein are the property of their respective owners.

The information in this document is subject to change without notice and should not be construed as a commitment by Cypress. While reasonable precautions have been taken, Cypress assumes no responsibility for any errors that may appear in this document. No part of this document may be copied or reproduced in any form or by any means without the prior written consent of Cypress. Made in the U.S.A.

Disclaimer

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC Device and User Module Data Sheets contain performance specifications and characterizations for critical parameters. Cypress Semiconductor does not recommend that you use unspecified or uncharacterized functions. If you need a device feature that is not completely specified or characterized, contact the Cypress PSoC Marketing organization through the support web site at http://www.cypress.com/support

Use may be limited by and subject to the applicable Cypress software license agreement.

Flash Code Protection

Note the following details of the Flash code protection features on Cypress devices.

Cypress products meet the specifications contained in their particular Cypress Data Sheets. Cypress believes that its family of products is one of the most secure families of its kind on the market today, regardless of how they are used. There may be methods, unknown to Cypress, that can breach the code protection features. Any of these methods, to our knowledge, would be dishonest and possibly illegal. Neither Cypress 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."

Cypress is willing to work with the customer who is concerned about the integrity of their code. Code protection is constantly evolving. We at Cypress are committed to continuously improving the code protection features of our products.

Contents Overview

Contents

Section A: Overview

The PSoC® family consists of many Mixed-Signal Array with On-Chip Controller devices. As described in this Technical Reference Manual (TRM), the CY8C20x34/24 PSoC device does not have regular digital PSoC blocks and global interconnects that are found in most PSoC devices. The CY8C20x34/24 devices have one analog resource and digital logic in addition to a fast CPU, Flash program memory, and SRAM data memory to support various CapSense™ algorithms.

For the most up-to-date Ordering, Pinout, Packaging, or Electrical Specification information, refer to the PSoC device's data sheet. For the most current technical reference manual information, refer to the addendum. To obtain the newest product documentation, go to the Cypress web site at http://www.cypress.com/psoc. This section contains this chapter:

- Pin Information on page 19.

Document Organization

This manual is organized into sections and chapters, according to ***PSoC*** functionality. Each section contains a top-level architectural diagram and a register summary (if applicable). Most chapters within the sections have an introduction, an architectural/application description, register definitions, and timing diagrams. The sections are:

- **Overview** – Presents the PSoC top-level architecture, helpful information to get started, and document history and conventions. The PSoC device ***pinouts*** are detailed in the Pin Information chapter.
- **PSoC Core** – Describes the heart of the PSoC device in various chapters, beginning with an architectural overview and a summary list of registers pertaining to the PSoC core.
- **CapSense System** – Describes the configurable PSoC CapSense system in various chapters, beginning with an architectural overview and a summary list of registers pertaining to the CapSense system.
- **System Resources** – Presents additional PSoC system resources, beginning with an overview and a summary list of registers pertaining to system resources.
- **Registers** – Lists all PSoC device registers in register mapping tables, and presents bit-level detail of each PSoC register in its own Register Reference chapter. Where applicable, detailed register descriptions are located in each chapter.
- **Glossary** – Defines the specialized terminology used in this manual. Glossary terms are presented in ***bold, italic font*** throughout this manual.
- **Index** – Lists the location of key topics and elements that constitute and empower the PSoC device.

Top-Level Architecture

The PSoC block diagram on the next page illustrates the top-level architecture of the CY8C20x34/24 PSoC device. Each major grouping in the diagram is covered in this manual in its own section: PSoC Core, CapSense System, and the System Resources. Banding these three main areas together is the communication network of the system ***bus***.

PSoC Core

The PSoC Core is a powerful engine that supports a rich instruction set. It encompasses the ***SRAM*** for data storage, an ***interrupt*** controller for easy program execution to new addresses, sleep and watchdog timers, a regulated 3.0V output option is provided for Port 1 IOs, and multiple ***clock*** sources that include the IMO (internal main oscillator) and ILO (internal low speed oscillator) for precision, programmable clocking.

The CPU core, called the M8C, is a powerful processor with speeds up to 12 MHz. The M8C is a two MIPS 8-***bit*** Harvard architecture microprocessor. Within the CPU core are the ***SROM*** and ***Flash*** memory components that provide flexible programming. The smallest PSoC devices have a slightly different analog configuration.

PSoC GPIOs provide connection to the CPU and the CapSense resources of the device. Each pin's drive mode may be selected from four options, allowing great flexibility in external interfacing. Every pin also has the capability to generate a system interrupt on low level and change from last read.

CapSense System

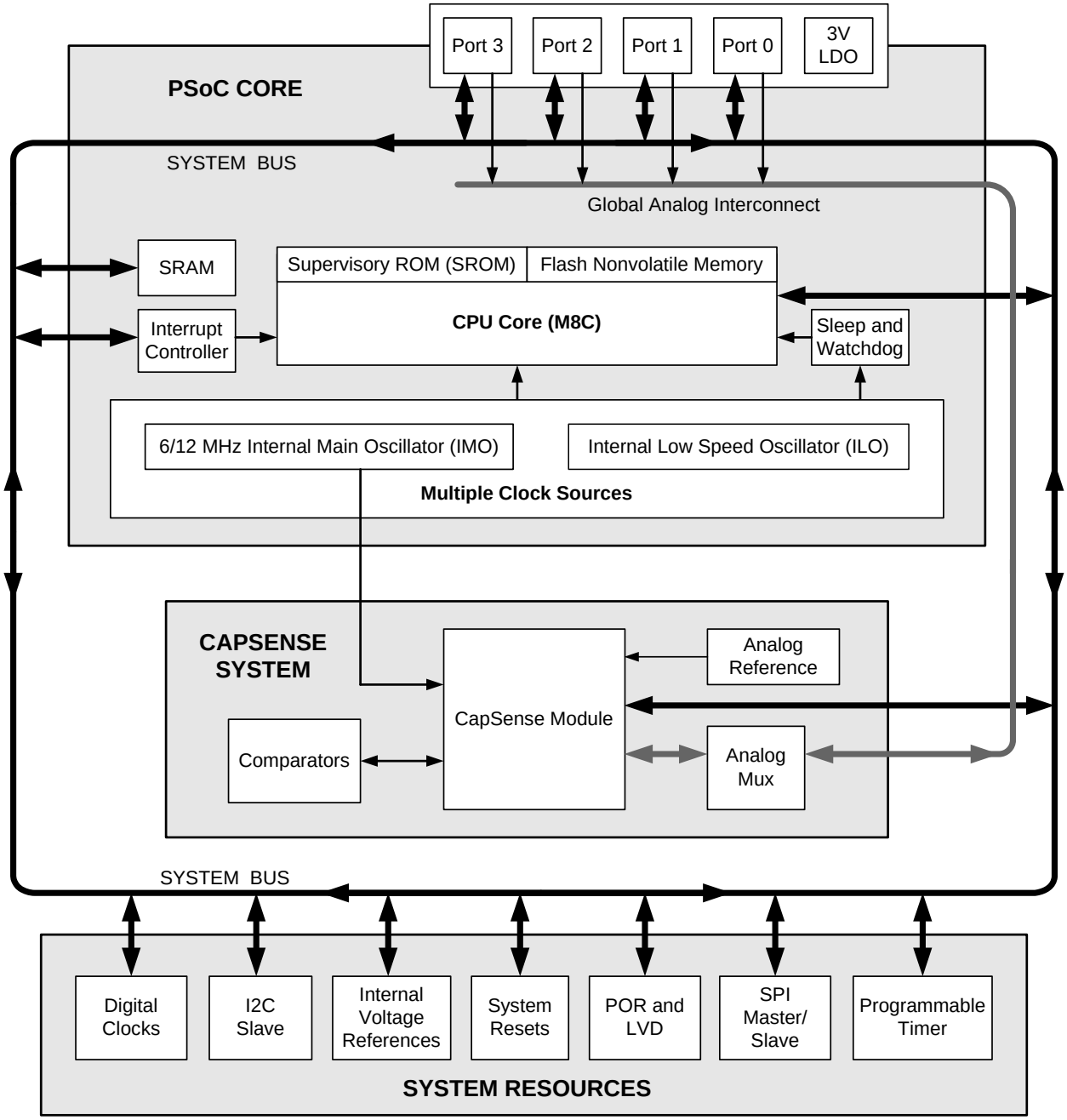
The CapSense System is composed of comparators, reference drivers, IO multiplexers, and digital logic to support various capsensing algorithms. Various reference selections are provided. Digital logic is mainly comprised of counters and timers.

System Resources

The System Resources provide additional PSoC capability. These system resources include:

- Digital clocks to increase the flexibility of the PSoC mixed-signal arrays.
- ***I2C*** functionality for implementing I2C slave.
- Internal voltage references that provide an absolute value of 0.9V, 1.3V, and 1.8V to the CapSense subsystems.
- Various system resets supported by the M8C.
- Power-On-Reset (POR) circuit protection.
- SPI master and slave functionality.
- A programmable timer to provide periodic interrupts.

PSoC Top-Level Block Diagram



Getting Started

The quickest path to understanding PSoC is by reading the PSoC device's data sheet and using the *PSoC Designer Integrated Development Environment (IDE)*. This manual is useful for understanding the details of the PSoC integrated circuit.

Important Note: For the most up-to-date Ordering, Packaging, or Electrical Specification information, refer to the individual PSoC device's data sheet or go to http://www.cypress.com/psoc.

Support

Free support for PSoC products is available online at http://www.cypress.com. Resources include Training Seminars, Discussion Forums, Application Notes, PSoC Consultants, TightLink Technical Support Email/Knowledge Base, and Application Support Technicians.

Technical Support can be reached at http://www.cypress.com/support/.

Product Upgrades

Cypress provides scheduled upgrades and version enhancements for PSoC Designer free of charge. You can order the upgrades from your distributor on CD-ROM or download them directly from http://www.cypress.com under Software and Drivers. Also provided are critical updates to system documentation under Design Support > Design Resources > More Resources or go to http://www.cypress.com.

Development Kits

Development Kits are available from authorized distributors. The Cypress Online Store contains development kits, ***C*** compilers, and all accessories for PSoC development. Go to the Cypress Online Store web site at http://www.cypress.com, click the Online Store shopping cart icon at the bottom of the web page, and click *PSoC (Programmable System-on-Chip)* to view a current list of available items.

Document History

This section serves as a chronicle of the *PSoC Mixed-Signal Array Technical Reference Manual*.

PSoC Technical Reference Manual History

Version/ Release Date	Originator	Description of Change
Version 1.0 September 22, 2006	VED	First release of the PSoC CY8C20x34 Technical Reference Manual. This release encompasses the CY8C20x34 PSoC device.
001-13033 Rev. ** April 24, 2007	VED	Update values in Table 16-1.
001-13033, Rev. *A February 19, 2008	DSG	Added CY8C20x24 parts, revised Table 4-11, corrected IMO Trim voltage ranges in Chapter 7

Documentation Conventions

There are only four distinguishing font types used in this manual, besides those found in the headings.

- The first is the use of *italics* when referencing a document title or file name.
- The second is the use of ***bold italics*** when referencing a term described in the Glossary of this manual.
- The third is the use of Times New Roman font, distinguishing equation examples.
- The fourth is the use of `Courier New` font, distinguishing code examples.

Register Conventions

This table lists the register conventions that are specific to this manual. A more detailed set of register conventions is located in the Register Reference chapter on page 137.

Register Conventions

Convention	Example	Description
'x' in a register name	PRTxIE	Multiple instances/address ranges of the same register
R	R : 00	Read register or bit(s)
W	W : 00	Write register or bit(s)
O	RO : 00	Only a read/write register or bit(s).
L	RL : 00	Logical register or bit(s)
C	RC : 00	Clearable register or bit(s)
00	RW : 00	Reset value is 0x00 or 00h
XX	RW : XX	Register is not reset
0,	0,04h	Register is in bank 0
1,	1,23h	Register is in bank 1
x,	x,F7h	Register exists in register bank 0 and register bank 1
Empty, grayed-out table cell		Reserved bit or group of bits, unless otherwise stated

Numeric Naming

Hexadecimal numbers are represented with all letters in uppercase with an appended lowercase 'h' (for example, '14h' or '3Ah') and ***hexadecimal*** numbers may also be represented by a '0x' prefix, the ***C*** coding convention. Binary numbers have an appended lowercase 'b' (for example, 01010100b' or '01000011b'). Numbers not indicated by an 'h' or 'b' are ***decimal***.

Units of Measure

This table lists the units of measure used in this manual.

Units of Measure

Symbol	Unit of Measure
dB	decibels
Hz	hertz
k	kilo, 1000
K	2^{10}, 1024
KB	1024 bytes
Kbit	1024 bits
kHz	kilohertz (32.000)
MHz	megahertz
µA	microampere
µF	microfarad
µs	microsecond
µV	microvolts
mA	milliampere
ms	millisecond
mV	millivolts
ns	nanosecond
pF	picofarad
ppm	parts per million
V	volts

Acronyms

This table lists the acronyms that are used in this manual.

Acronyms

Acronym	Description
ABUS	analog output bus
AC	alternating current
ADC	analog-to-digital converter
API	Application Programming Interface
BC	broadcast clock
BR	bit rate
BRA	bus request acknowledge
BRQ	bus request
CBUS	comparator bus
CI	carry in
CMP	compare
CO	carry out
CPU	central processing unit
CRC	cyclic redundancy check
CT	continuous time
DAC	digital-to-analog converter
DC	direct current
DI	digital or data input
DMA	direct memory access
DO	digital or data output
ECO	external crystal oscillator
FB	feedback
GIE	global interrupt enable
GPIO	general purpose IO
ICE	in-circuit emulator
IDE	integrated development environment
ILO	internal low speed oscillator
IMO	internal main oscillator
IO	input/output
IOR	IO read
IOW	IO write
IPOR	imprecise power on reset
IRQ	interrupt request
ISR	interrupt service routine
ISSP	in system serial programming
IVR	interrupt vector read
LFSR	linear feedback shift register
LRb	last received bit
LRB	last received byte
LSb	least significant bit
LSB	least significant byte
LUT	look-up table
MISO	master-in-slave-out
MOSI	master-out-slave-in
MSb	most significant bit
MSB	most significant byte
PC	program counter
PCH	program counter high

Acronyms *(continued)*

Acronym	Description
PCL	program counter low
PD	power down
PMA	PSoC™ memory arbiter
POR	power on reset
PPOR	precision power on reset
PRS	pseudo random sequence
PSoC™	Programmable System-on-Chip™
PSSDC	power system sleep duty cycle
PWM	pulse width modulator
RAM	random access memory
RETI	return from interrupt
RO	relaxation oscillator
ROM	read only memory
RW	read/write
SAR	successive approximation register
SC	switched capacitor
SIE	serial interface engine
SE0	single-ended zero
SOF	start of frame
SP	stack pointer
SPI	serial peripheral interconnect
SPIM	serial peripheral interconnect master
SPIS	serial peripheral interconnect slave
SRAM	static random access memory
SROM	supervisory read only memory
SSADC	single slope ADC
SSC	supervisory system call
TC	terminal count
USB	universal serial bus
WDT	watchdog timer
WDR	watchdog reset
XRES	external reset

1. Pin Information

This chapter lists, describes, and illustrates all pins and pinout configurations for the CY8C20x34/24 PSoC device. For up-to-date ordering, pinout, and packaging information, refer to the individual PSoC device's data sheet or go to http://www.cypress.com/psoc.

1.1 Pinouts

The CY8C20x34/24 PSoC devices are available in a variety of packages. Every ***port*** pin (labeled with a "P"), except for ***Vss*** and ***Vdd*** in the following tables and illustrations, is capable of Digital IO.

1.1.1 16-Pin Part Pinout

Table 1-1: 16-Pin Part Pinout (QFN**)

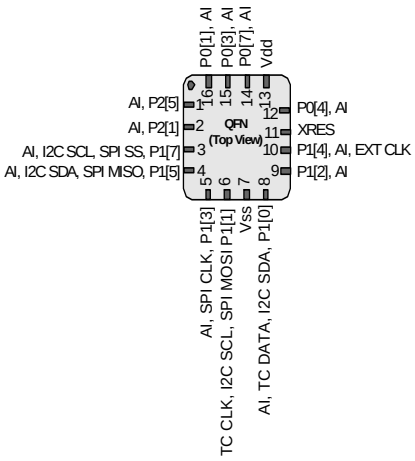
Pin No.	Type		Name	Description
	Digital	Analog		
1	IO	I	P2[5]	
2	IO	I	P2[1]	
3	IO	I	P1[7]	I2C SCL, SPI SS
4	IO	I	P1[5]	I2C SDA, SPI MISO
5	IO	I	P1[3]	SPI CLK
6	IO	I	P1[1]	TC CLK, I2C SCL, SPI MOSI
7	IO	I	Vss	Ground Connection
8	IO	I	P1[0]	TC DATA, I2C SDA
9	IO	I	P1[2]	
10	IO	I	P1[4]	EXTCLK
11	Reset		XRES	Active high external reset with internal pull down
12	IO	I	P0[4]	
13	IO	I	Vdd	Supply Voltage
14	IO	I	P0[7]	
15	IO	I	P0[3]	Integration Cap
16	IO	I	P0[1]	

LEGEND A = Analog, I = Input, O = Output, H = 5 mA High Output Drive.

* These are the ISSP pins, which are not High Z at POR (Power On Reset).

** The center pad on the QFN package should be connected to ground (Vss) for best mechanical, thermal, and electrical performance. If not connected to ground, it should be electrically floated and not connected to any other signal.

CY8C20234, CY8C20224 PSoC Device



1.1.2 24-Pin Part Pinout

Table 1-2. 24-Pin Part Pinout (QFN**)

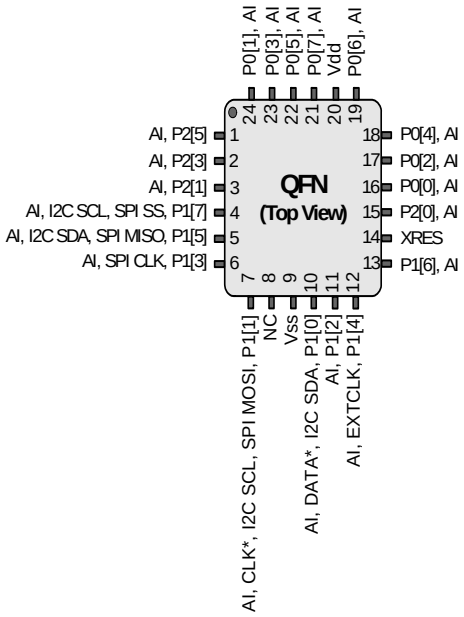
Pin No.	Type		Name	Description
	Digital	Analog		
1	IO	I	P2[5]	
2	IO	I	P2[3]	
3	IO	I	P2[1]	
4	IOH	I	P1[7]	I2C SCL, SPI SS
5	IOH	I	P1[5]	I2C SDA, SPI MISO
6	IOH	I	P1[3]	SPI CLK
7	IOH	I	P1[1]	CLK*, I2C SCL, SPI MOSI
8			NC	No connection
9	Power		Vss	Ground connection
10	IOH	I	P1[0]	DATA*, I2C SDA
11	IOH	I	P1[2]	
12	IOH	I	P1[4]	Optional external clock input (EXTCLK)
13	IOH	I	P1[6]	
14	Input		XRES	Active high external reset with internal pull down
15	IO	I	P2[0]	
16	IO	I	P0[0]	
17	IO	I	P0[2]	
18	IO	I	P0[4]	
19	IO	I	P0[6]	Analog bypass
20	Power		Vdd	Supply voltage
21	IO	I	P0[7]	
22	IO	I	P0[5]	
23	IO	I	P0[3]	Integrating input
24	IO	I	P0[1]	

LEGEND A = Analog, I = Input, O = Output, H = 5 mA High Output Drive.

* These are the ISSP pins, which are not High Z at POR (Power On Reset).

** The center pad on the QFN package should be connected to ground (Vss) for best mechanical, thermal, and electrical performance. If not connected to ground, it should be electrically floated and not connected to any other signal.

CY8C20334, CY8C20324 PSoC Device

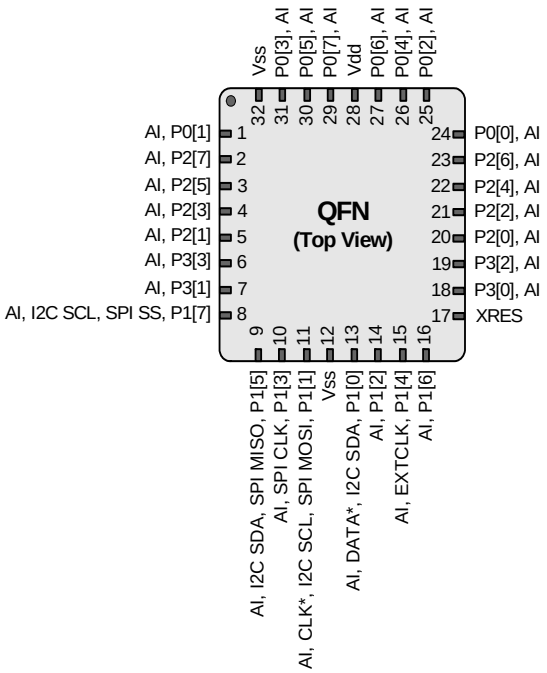


1.1.3 32-Pin Part Pinout

Table 1-3. 32-Pin Part Pinout (QFN**)

Pin No.	Type		Name	Description
	Digital	Analog		
1	IO	I	P0[1]	
2	IO	I	P2[7]	
3	IO	I	P2[5]	
4	IO	I	P2[3]	
5	IO	I	P2[1]	
6	IO	I	P3[3]	
7	IO	I	P3[1]	
8	IOH	I	P1[7]	I2C SCL, SPI SS
9	IOH	I	P1[5]	I2C SDA, SPI MISO
10	IOH	I	P1[3]	SPI CLK
11	IOH	I	P1[1]	CLK*, I2C SCL, SPI MOSI
12	Power		Vss	Ground connection
13	IOH	I	P1[0]	DATA*, I2C SDA
14	IOH	I	P1[2]	
15	IOH	I	P1[4]	Optional external clock input (EXTCLK)
16	IOH	I	P1[6]	
17	Input		XRES	Active high external reset with internal pull down
18	IO	I	P3[0]	
19	IO	I	P3[2]	
20	IO	I	P2[0]	
21	IO	I	P2[2]	
22	IO	I	P2[4]	
23	IO	I	P2[6]	
24	IO	I	P0[0]	
25	IO	I	P0[2]	
26	IO	I	P0[4]	
27	IO	I	P0[6]	Analog bypass
28	Power		Vdd	Supply voltage
29	IO	I	P0[7]	
30	IO	I	P0[5]	
31	IO	I	P0[3]	Integrating input
32	Power		Vss	Ground connection

CY8C20434, CY8C20424 PSoC Device



LEGEND A = Analog, I = Input, O = Output, H = 5 mA High Output Drive.

* These are the ISSP pins, which are not High Z at POR (Power On Reset).

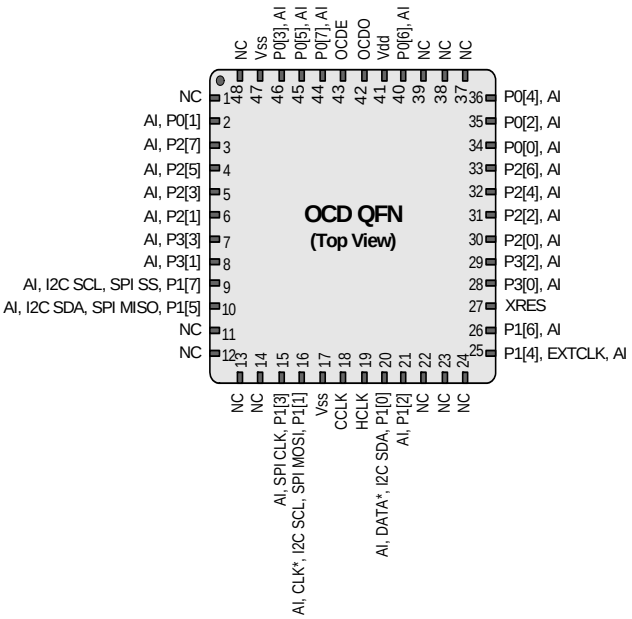
** The center pad on the QFN package should be connected to ground (Vss) for best mechanical, thermal, and electrical performance. If not connected to ground, it should be electrically floated and not connected to any other signal.

1.1.4 48-Pin OCD Part Pinout

Table 1-4. 48-Pin OCD Part Pinout (QFN**)

Pin No.	Digital	Analog	Name	Description
1			NC	No internal connection
2	IO	I	P0[1]	
3	IO	I	P2[7]	
4	IO	I	P2[5]	
5	IO	I	P2[3]	
6	IO	I	P2[1]	
7	IO	I	P3[3]	
8	IO	I	P3[1]	
9	IOH	I	P1[7]	I2C SCL, SPI SS
10	IOH	I	P1[5]	I2C SDA, SPI MISO
11			NC	No internal connection
12			NC	No internal connection
13			NC	No internal connection
14			NC	No internal connection
15	IOH	I	P1[3]	SPI CLK
16	IOH	I	P1[1]	CLK*, I2C SCL, SPI MOSI
17	Power		Vss	Ground connection
18			CCLK	OCD CPU clock output
19			HCLK	OCD high speed clock output
20	IOH	I	P1[0]	DATA*, I2C SDA
21	IOH	I	P1[2]	
22			NC	No internal connection
23			NC	No internal connection
24			NC	No internal connection
25	IOH	I	P1[4]	Optional external clock input (EXTCLK)
26	IOH	I	P1[6]	
27	Input		XRES	Active high external reset with internal pull down
28	IO	I	P3[0]	
29	IO	I	P3[2]	
30	IO	I	P2[0]	
31	IO	I	P2[2]	
32	IO	I	P2[4]	
33	IO	I	P2[6]	
34	IO	I	P0[0]	
35	IO	I	P0[2]	
36	IO	I	P0[4]	
37			NC	No internal connection
38			NC	No internal connection
39			NC	No internal connection
40	IO	I	P0[6]	Analog bypass

CY8C20000 OCD PSoC Device



NOT FOR PRODUCTION

Pin No.	Digital	Analog	Name	Description
41	Power		Vdd	Supply voltage
42			OCDO	OCD even data IO
43			OCDE	OCD odd data output
44	IO	I	P0[7]	
45	IO	I	P0[5]	
46	IO	I	P0[3]	Integrating input
47	Power		Vss	Ground connection
48			NC	No internal connection

LEGEND A = Analog, I = Input, O = Output, NC = No Connection, H = 5 mA High Output Drive.
* ISSP pin which is not HiZ at POR.
** The center pad on the QFN package should be connected to ground (Vss) for best mechanical, thermal, and electrical performance. If not connected to ground, it should be electrically floated and not connected to any other signal.

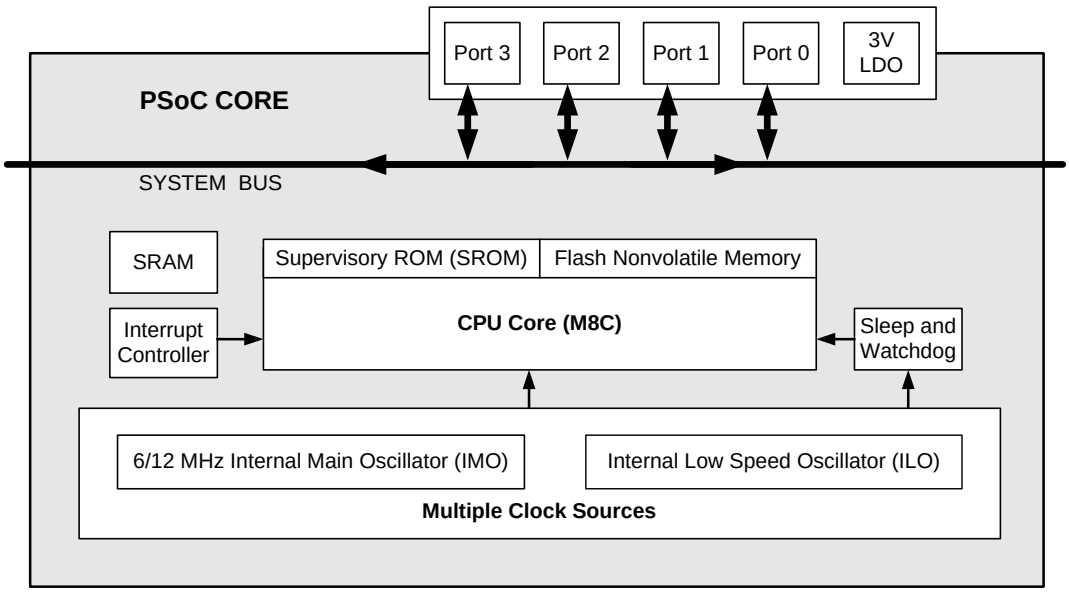
Section B: PSoC Core

The PSoC® Core section discusses the core components of a PSoC device with a base part number of CY8C20x34/24 and the registers associated with those components. The core section covers the heart of the PSoC device, which includes the M8C ***microcontroller***; SROM, interrupt controller, GPIO, and ***SRAM*** paging; multiple clock sources such as IMO and ILO; and sleep and watchdog functionality. This section contains these chapters:

Top Level Core Architecture

The figure below illustrates the top-level architecture of the PSoC's core. Each figure component is discussed in detail in this section.

PSoC Core Block Diagram



Core Register Summary

This table lists all the PSoC registers for the CPU core in ***address*** order within their system resource configuration. The grayed bits are reserved bits. If these bits are written, always write them with a value of '0'. For the core registers, the first 'x' in some ***register*** addresses represents either bank 0 or bank 1. These registers are listed throughout this manual in bank 0, even though they are also available in bank 1.

Summary Table of the Core Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
M8C REGISTER (page 25)										
x,F7h	CPU_F	PgMode[1:0]			XIO		Carry	Zero	GIE	RL : 02
RAM PAGING (SRAM) REGISTERS (page 31)										
0,D0h	CUR_PP								Page Bit	RW : 00
0,D1h	STK_PP								Page Bit	RW : 00
0,D3h	IDX_PP								Page Bit	RW : 00
0,D4h	MVR_PP								Page Bit	RW : 00
0,D5h	MVW_PP								Page Bit	RW : 00
INTERRUPT CONTROLLER REGISTERS (page 45)										
0,DAh	INT_CLR0	I2C	Sleep	SPI	GPIO	Timer	CapSense	Analog	V Monitor	RW : 00
0,E0h	INT_MSK0	I2C	Sleep	SPI	GPIO	Timer	CapSense	Analog	V Monitor	RW : 00
0,E1h	INT_SW_EN								ENSWINT	RW : 00
0,E2h	INT_VC	Pending Interrupt[7:0]								RC : 00
GENERAL PURPOSE IO (GPIO) REGISTERS (page 51)										
0,00h	PRT0DR	Data[7:0]								RW : 00
0,01h	PRT0IE	Interrupt Enables[7:0]								RW : 00
0,04h	PRT1DR	Data[7:0]								RW : 00
0,05h	PRT1IE	Interrupt Enables[7:0]								RW : 00
0,08h	PRT2DR	Data[7:0]								RW : 00
0,09h	PRT2IE	Interrupt Enables[7:0]								RW : 00
0,0Ch	PRT3DR	Data[7:0]								RW : 00
0,0Dh	PRT3IE	Interrupt Enables[7:0]								RW : 00
1,00h	PRT0DM0	Drive Mode 0[7:0]								RW : 00
1,01h	PRT0DM1	Drive Mode 1[7:0]								RW : FF
1,04h	PRT1DM0	Drive Mode 0[7:0]								RW : 00
1,05h	PRT1DM1	Drive Mode 1[7:0]								RW : FF
1,08h	PRT2DM0	Drive Mode 0[7:0]								RW : 00
1,09h	PRT2DM1	Drive Mode 1[7:0]								RW : FF
1,0Ch	PRT3DM0	Drive Mode 0[7:0]								RW : 00
1,0Dh	PRT3DM1	Drive Mode 1[7:0]								RW : FF
1,DCh	IO_CFG							REG_EN	IOINT	RW : 00
INTERNAL MAIN OSCILLATOR (IMO) REGISTER (page 57)										
1,E8h	IMO_TR	Trim[7:0]								W : 00
INTERNAL LOW SPEED OSCILLATOR (ILO) REGISTER (page 59)										
1,E9h	ILO_TR			Bias Trim[1:0]		Freq Trim[3:0]				W : 00
SLEEP AND WATCHDOG REGISTERS (page 61)										
0,E3h	RES_WDT	WDSL_Clear[7:0]								W : 00
1,EBh	SLP_CFG	PSSDC[1:0]								RW : 00

LEGEND
L The and f, expr; or f, expr; and xor f, expr instructions can be used to modify this register.
x An "x" before the comma in the address field indicates that this register can be accessed or written to no matter what bank is used.
C Clearable register or bit(s).
R Read register or bit(s).
W Write register or bit(s).

2. CPU Core (M8C)

This chapter explains the CPU Core, called the M8C, and its associated register. It covers the internal M8C registers, address spaces, ***instruction*** set and formats. For additional information concerning the M8C instruction set, refer to the *PSoC Designer Assembly Language User Guide* available at the Cypress web site (http://www.cypress.com/psoc). For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

2.1 Overview

The ***M8C*** is a two MIPS 8-bit Harvard architecture microprocessor. Selectable processor clock speeds up to 12 MHz allow you to adjust the M8C to a particular application's performance and power requirements. The M8C supports a rich instruction set that allows for efficient low level language support.

2.2 Internal Registers

The M8C has five internal registers that are used in program execution. The registers are:

- Accumulator (A)
- Index (X)
- Program Counter (PC)
- Stack Pointer (SP)
- Flags (F)

All of the internal M8C registers are eight bits in width, except for the PC which is 16 bits wide. When ***reset***, A, X, PC, and SP are reset to 00h. The Flag register (F) is reset to 02h, indicating that the Z ***flag*** is ***set***.

With each ***stack*** operation, the SP is automatically incremented or decremented so that it always points to the next stack ***byte*** in RAM. If the last byte in the stack is at address FFh, the ***stack pointer*** will wrap to RAM address 00h. It is the ***firmware*** developer's responsibility to ensure that the stack does not overlap with user-defined variables in RAM.

With the exception of the F register, the M8C internal registers are not accessible via an explicit register address. The internal M8C registers are accessed using these instructions:

- `MOV A, expr`
- `MOV X, expr`
- `SWAP A, SP`
- `OR F, expr`
- `JMP LABEL`

The F register is read by using address F7h in either register bank.

2.3 Address Spaces

The M8C has three address spaces: ***ROM***, ***RAM***, and registers. The ROM address space includes the supervisory ROM (SROM) and the Flash. The ROM address space is accessed through its own address and ***data bus***.

The ROM address space is composed of the Supervisory ROM and the on-chip Flash program store. Flash is organized into 64-byte blocks. Program store page boundaries are not a concern, since the M8C automatically increments the 16-bit PC on every instruction. This process makes the block boundaries invisible to user code. Instructions occurring on a 256-byte Flash page boundary (with the exception of JMP instructions) incur an extra M8C clock cycle as the upper byte of the PC is incremented.

The register address space is used to configure the PSoC microcontroller's programmable blocks. It consists of two banks of 256 bytes each. To switch between banks, the XIO bit in the Flag register is set or cleared (set for Bank1, cleared for Bank0). The common convention is to leave the bank set to Bank0 (XIO cleared), switch to Bank1 as necessary (set XIO), then switch back to Bank0.

2.4 Instruction Set Summary

The instruction set is summarized in both Table 2-1 and Table 2-2 (in numeric and ***mnemonic*** order, respectively), and serves as a quick reference. If more information is needed, the Instruction Set Summary tables are described in detail in the *PSoC Designer Assembly Language User Guide* (refer to the http://www.cypress.com/psoc web site).

Table 2-1. Instruction Set Summary Sorted Numerically by Opcode

Opcode Hex	Cycles	Bytes	Instruction Format	Flags	Opcode Hex	Cycles	Bytes	Instruction Format	Flags	Opcode Hex	Cycles	Bytes	Instruction Format	Flags
00	15	1	SSC		2D	8	2	OR [X+expr], A	Z	5A	5	2	MOV [expr], X	
01	4	2	ADD A, expr	C, Z	2E	9	3	OR [expr], expr	Z	5B	4	1	MOV A, X	Z
02	6	2	ADD A, [expr]	C, Z	2F	10	3	OR [X+expr], expr	Z	5C	4	1	MOV X, A	
03	7	2	ADD A, [X+expr]	C, Z	30	9	1	HALT		5D	6	2	MOV A, reg[expr]	Z
04	7	2	ADD [expr], A	C, Z	31	4	2	XOR A, expr	Z	5E	7	2	MOV A, reg[X+expr]	Z
05	8	2	ADD [X+expr], A	C, Z	32	6	2	XOR A, [expr]	Z	5F	10	3	MOV [expr], [expr]	
06	9	3	ADD [expr], expr	C, Z	33	7	2	XOR A, [X+expr]	Z	60	5	2	MOV reg[expr], A	
07	10	3	ADD [X+expr], expr	C, Z	34	7	2	XOR [expr], A	Z	61	6	2	MOV reg[X+expr], A	
08	4	1	PUSH A		35	8	2	XOR [X+expr], A	Z	62	8	3	MOV reg[expr], expr	
09	4	2	ADC A, expr	C, Z	36	9	3	XOR [expr], expr	Z	63	9	3	MOV reg[X+expr], expr	
0A	6	2	ADC A, [expr]	C, Z	37	10	3	XOR [X+expr], expr	Z	64	4	1	ASL A	C, Z
0B	7	2	ADC A, [X+expr]	C, Z	38	5	2	ADD SP, expr		65	7	2	ASL [expr]	C, Z
0C	7	2	ADC [expr], A	C, Z	39	5	2	CMP A, expr		66	8	2	ASL [X+expr]	C, Z
0D	8	2	ADC [X+expr], A	C, Z	3A	7	2	CMP A, [expr]		67	4	1	ASR A	C, Z
0E	9	3	ADC [expr], expr	C, Z	3B	8	2	CMP A, [X+expr]	if (A=B) Z=1	68	7	2	ASR [expr]	C, Z
0F	10	3	ADC [X+expr], expr	C, Z	3C	8	3	CMP [expr], expr	if (A<B) C=1	69	8	2	ASR [X+expr]	C, Z
10	4	1	PUSH X		3D	9	3	CMP [X+expr], expr		6A	4	1	RLC A	C, Z
11	4	2	SUB A, expr	C, Z	3E	10	2	MVI A, [[expr]++]	Z	6B	7	2	RLC [expr]	C, Z
12	6	2	SUB A, [expr]	C, Z	3F	10	2	MVI [[expr]++], A		6C	8	2	RLC [X+expr]	C, Z
13	7	2	SUB A, [X+expr]	C, Z	40	4	1	NOP		6D	4	1	RRC A	C, Z
14	7	2	SUB [expr], A	C, Z	41	9	3	AND reg[expr], expr	Z	6E	7	2	RRC [expr]	C, Z
15	8	2	SUB [X+expr], A	C, Z	42	10	3	AND reg[X+expr], expr	Z	6F	8	2	RRC [X+expr]	C, Z
16	9	3	SUB [expr], expr	C, Z	43	9	3	OR reg[expr], expr	Z	70	4	2	AND F, expr	C, Z
17	10	3	SUB [X+expr], expr	C, Z	44	10	3	OR reg[X+expr], expr	Z	71	4	2	OR F, expr	C, Z
18	5	1	POP A	Z	45	9	3	XOR reg[expr], expr	Z	72	4	2	XOR F, expr	C, Z
19	4	2	SBB A, expr	C, Z	46	10	3	XOR reg[X+expr], expr	Z	73	4	1	CPL A	Z
1A	6	2	SBB A, [expr]	C, Z	47	8	3	TST [expr], expr	Z	74	4	1	INC A	C, Z
1B	7	2	SBB A, [X+expr]	C, Z	48	9	3	TST [X+expr], expr	Z	75	4	1	INC X	C, Z
1C	7	2	SBB [expr], A	C, Z	49	9	3	TST reg[expr], expr	Z	76	7	2	INC [expr]	C, Z
1D	8	2	SBB [X+expr], A	C, Z	4A	10	3	TST reg[X+expr], expr	Z	77	8	2	INC [X+expr]	C, Z
1E	9	3	SBB [expr], expr	C, Z	4B	5	1	SWAP A, X	Z	78	4	1	DEC A	C, Z
1F	10	3	SBB [X+expr], expr	C, Z	4C	7	2	SWAP A, [expr]	Z	79	4	1	DEC X	C, Z
20	5	1	POP X		4D	7	2	SWAP X, [expr]		7A	7	2	DEC [expr]	C, Z
21	4	2	AND A, expr	Z	4E	5	1	SWAP A, SP	Z	7B	8	2	DEC [X+expr]	C, Z
22	6	2	AND A, [expr]	Z	4F	4	1	MOV X, SP		7C	13	3	LCALL	
23	7	2	AND A, [X+expr]	Z	50	4	2	MOV A, expr	Z	7D	7	3	LJMP	
24	7	2	AND [expr], A	Z	51	5	2	MOV A, [expr]	Z	7E	10	1	RETI	C, Z
25	8	2	AND [X+expr], A	Z	52	6	2	MOV A, [X+expr]	Z	7F	8	1	RET	
26	9	3	AND [expr], expr	Z	53	5	2	MOV [expr], A		8x	5	2	JMP	
27	10	3	AND [X+expr], expr	Z	54	6	2	MOV [X+expr], A		9x	11	2	CALL	
28	11	1	ROMX	Z	55	8	3	MOV [expr], expr		Ax	5	2	JZ	
29	4	2	OR A, expr	Z	56	9	3	MOV [X+expr], expr		Bx	5	2	JNZ	
2A	6	2	OR A, [expr]	Z	57	4	2	MOV X, expr		Cx	5	2	JC	
2B	7	2	OR A, [X+expr]	Z	58	6	2	MOV X, [expr]		Dx	5	2	JNC	
2C	7	2	OR [expr], A	Z	59	7	2	MOV X, [X+expr]		Ex	7	2	JACC	
										Fx	13	2	INDEX	Z

Note 1 Interrupt acknowledge to Interrupt Vector table = 13 cycles.

Note 2 The number of cycles required by an instruction is increased by one for instructions that span 256 byte page boundaries in the Flash memory space.

Table 2-2. Instruction Set Summary Sorted Alphabetically by Mnemonic

Opcode Hex	Cycles	Bytes	Instruction Format	Flags	Opcode Hex	Cycles	Bytes	Instruction Format	Flags	Opcode Hex	Cycles	Bytes	Instruction Format	Flags
09	4	2	ADC A, expr	C, Z	76	7	2	INC [expr]	C, Z	20	5	1	POP X	
0A	6	2	ADC A, [expr]	C, Z	77	8	2	INC [X+expr]	C, Z	18	5	1	POP A	Z
0B	7	2	ADC A, [X+expr]	C, Z	Fx	13	2	INDEX	Z	10	4	1	PUSH X	
0C	7	2	ADC [expr], A	C, Z	Ex	7	2	JACC		08	4	1	PUSH A	
0D	8	2	ADC [X+expr], A	C, Z	Cx	5	2	JC		7E	10	1	RETI	C, Z
0E	9	3	ADC [expr], expr	C, Z	8x	5	2	JMP		7F	8	1	RET	
0F	10	3	ADC [X+expr], expr	C, Z	Dx	5	2	JNC		6A	4	1	RLC A	C, Z
01	4	2	ADD A, expr	C, Z	Bx	5	2	JNZ		6B	7	2	RLC [expr]	C, Z
02	6	2	ADD A, [expr]	C, Z	Ax	5	2	JZ		6C	8	2	RLC [X+expr]	C, Z
03	7	2	ADD A, [X+expr]	C, Z	7C	13	3	LCALL		28	11	1	ROMX	Z
04	7	2	ADD [expr], A	C, Z	7D	7	3	LJMP		6D	4	1	RRC A	C, Z
05	8	2	ADD [X+expr], A	C, Z	4F	4	1	MOV X, SP		6E	7	2	RRC [expr]	C, Z
06	9	3	ADD [expr], expr	C, Z	50	4	2	MOV A, expr	Z	6F	8	2	RRC [X+expr]	C, Z
07	10	3	ADD [X+expr], expr	C, Z	51	5	2	MOV A, [expr]	Z	19	4	2	SBB A, expr	C, Z
38	5	2	ADD SP, expr		52	6	2	MOV A, [X+expr]	Z	1A	6	2	SBB A, [expr]	C, Z
21	4	2	AND A, expr	Z	53	5	2	MOV [expr], A		1B	7	2	SBB A, [X+expr]	C, Z
22	6	2	AND A, [expr]	Z	54	6	2	MOV [X+expr], A		1C	7	2	SBB [expr], A	C, Z
23	7	2	AND A, [X+expr]	Z	55	8	3	MOV [expr], expr		1D	8	2	SBB [X+expr], A	C, Z
24	7	2	AND [expr], A	Z	56	9	3	MOV [X+expr], expr		1E	9	3	SBB [expr], expr	C, Z
25	8	2	AND [X+expr], A	Z	57	4	2	MOV X, expr		1F	10	3	SBB [X+expr], expr	C, Z
26	9	3	AND [expr], expr	Z	58	6	2	MOV X, [expr]		00	15	1	SSC	
27	10	3	AND [X+expr], expr	Z	59	7	2	MOV X, [X+expr]		11	4	2	SUB A, expr	C, Z
70	4	2	AND F, expr	C, Z	5A	5	2	MOV [expr], X		12	6	2	SUB A, [expr]	C, Z
41	9	3	AND reg[expr], expr	Z	5B	4	1	MOV A, X	Z	13	7	2	SUB A, [X+expr]	C, Z
42	10	3	AND reg[X+expr], expr	Z	5C	4	1	MOV X, A		14	7	2	SUB [expr], A	C, Z
64	4	1	ASL A	C, Z	5D	6	2	MOV A, reg[expr]	Z	15	8	2	SUB [X+expr], A	C, Z
65	7	2	ASL [expr]	C, Z	5E	7	2	MOV A, reg[X+expr]	Z	16	9	3	SUB [expr], expr	C, Z
66	8	2	ASL [X+expr]	C, Z	5F	10	3	MOV [expr], [expr]		17	10	3	SUB [X+expr], expr	C, Z
67	4	1	ASR A	C, Z	60	5	2	MOV reg[expr], A		4B	5	1	SWAP A, X	Z
68	7	2	ASR [expr]	C, Z	61	6	2	MOV reg[X+expr], A		4C	7	2	SWAP A, [expr]	Z
69	8	2	ASR [X+expr]	C, Z	62	8	3	MOV reg[expr], expr		4D	7	2	SWAP X, [expr]	
9x	11	2	CALL		63	9	3	MOV reg[X+expr], expr		4E	5	1	SWAP A, SP	Z
39	5	2	CMP A, expr	if (A=B) Z=1 if (A<B) C=1	3E	10	2	MVI A, [[expr]++]	Z	47	8	3	TST [expr], expr	Z
3A	7	2	CMP A, [expr]		3F	10	2	MVI [[expr]++], A		48	9	3	TST [X+expr], expr	Z
3B	8	2	CMP A, [X+expr]		40	4	1	NOP		49	9	3	TST reg[expr], expr	Z
3C	8	3	CMP [expr], expr		29	4	2	OR A, expr	Z	4A	10	3	TST reg[X+expr], expr	Z
3D	9	3	CMP [X+expr], expr		2A	6	2	OR A, [expr]	Z	72	4	2	XOR F, expr	C, Z
73	4	1	CPL A	Z	2B	7	2	OR A, [X+expr]	Z	31	4	2	XOR A, expr	Z
78	4	1	DEC A	C, Z	2C	7	2	OR [expr], A	Z	32	6	2	XOR A, [expr]	Z
79	4	1	DEC X	C, Z	2D	8	2	OR [X+expr], A	Z	33	7	2	XOR A, [X+expr]	Z
7A	7	2	DEC [expr]	C, Z	2E	9	3	OR [expr], expr	Z	34	7	2	XOR [expr], A	Z
7B	8	2	DEC [X+expr]	C, Z	2F	10	3	OR [X+expr], expr	Z	35	8	2	XOR [X+expr], A	Z
30	9	1	HALT		43	9	3	OR reg[expr], expr	Z	36	9	3	XOR [expr], expr	Z
74	4	1	INC A	C, Z	44	10	3	OR reg[X+expr], expr	Z	37	10	3	XOR [X+expr], expr	Z
75	4	1	INC X	C, Z	71	4	2	OR F, expr	C, Z	45	9	3	XOR reg[expr], expr	Z
										46	10	3	XOR reg[X+expr], expr	Z

Note 1 Interrupt acknowledge to Interrupt Vector table = 13 cycles.

Note 2 The number of cycles required by an instruction is increased by one for instructions that span 256 byte page boundaries in the Flash memory space.

2.5 Instruction Formats

The M8C has a total of seven instruction formats that use instruction lengths of one, two, and three bytes. All instruction bytes are retrieved from the program memory (Flash), using an address and data bus that are independent from the address and data buses used for register and RAM access.

While examples of instructions are given in this section, refer to the *PSoC Designer Assembly Language User Guide* for detailed information on individual instructions.

2.5.1 One-Byte Instructions

Many instructions, such as some of the MOV instructions, have single-byte forms because they do not use an address or data as an operand. As shown in Table 2-3, one-byte instructions use an 8-bit opcode. The set of one-byte instructions can be divided into four categories, according to where their results are stored.

Table 2-3. One-Byte Instruction Format

Byte 0
8-Bit Opcode

The first category of one-byte instructions are those that do not update any registers or RAM. Only the one-byte NOP and SSC instructions fit this category. While the ***program counter*** is incremented as these instructions execute, they do not cause any other internal M8C registers to update, nor do these instructions directly affect the register space or the RAM address space. The SSC instruction causes SROM code to run, which modifies RAM and the M8C internal registers.

The second category contains only the two PUSH instructions. The PUSH instructions are unique because they are the only one-byte instructions that modifies a RAM address. These instructions automatically increment the SP.

The third category contains only the HALT instruction. The HALT instruction is unique because it is the only a one-byte instruction that modifies a user register. The HALT instruction modifies user register space address FFh (CPU_SCR0 register).

The final category for one-byte instructions are those that cause updates of the internal M8C registers. This category holds the largest number of instructions: ASL, ASR, CPL, DEC, INC, MOV, POP, RET, RETI, RLC, ROMX, RRC, SWAP. These instructions can cause the A, X, and SP registers or SRAM to update.

2.5.2 Two-Byte Instructions

The majority of M8C instructions are two bytes in length. While it is possible to divide these instructions into categories identical to the one-byte instructions, this does not provide a useful distinction between the three two-byte instruction formats that the M8C uses.

Table 2-4. Two-Byte Instruction Formats

Byte 0	Byte 1
4-Bit Opcode	12-Bit Relative Address
8-Bit Opcode	8-Bit Data
8-Bit Opcode	8-Bit Address

The first two-byte instruction format, shown in the first row of Table 2-4, is used by short jumps and calls: `CALL`, `JMP`, `JACC`, `INDEX`, `JC`, `JNC`, `JNZ`, `JZ`. This instruction format uses only four bits for the instruction opcode, leaving 12 bits to store the relative destination address in a two's-complement form. These instructions can change program execution to an address relative to the current address by -2048 or +2047.

The second two-byte instruction format, shown in the second row of Table 2-4, is used by instructions that employ the Source Immediate addressing ***mode*** (see the *PSoC Designer Assembly Language User Guide*). The destination for these instructions is an internal M8C register, while the source is a constant value. An example of this type of instruction is `ADD A, 7`.

The third two-byte instruction format, shown in the third row of Table 2-4, is used by a wide range of instructions and addressing modes. Here is a list of the addressing modes that use this third two-byte instruction format:

- Source Direct (`ADD A, [7]`)
- Source Indexed (`ADD A, [X+7]`)
- Destination Direct (`ADD [7], A`)
- Destination Indexed (`ADD [X+7], A`)
- Source Indirect Post Increment (`MVI A, [7]`)
- Destination Indirect Post Increment (`MVI [7], A`)

For more information on addressing modes see the *PSoC Designer Assembly Language User Guide*.

2.5.3 Three-Byte Instructions

The three-byte instruction formats are the second most prevalent instruction formats. These instructions need three bytes because they either move data between two addresses in the user-accessible address space (registers and RAM) or they hold 16-bit absolute addresses as the destination of a long jump or long call.

Table 2-5. Three-Byte Instruction Formats

Byte 0	Byte 1	Byte 2
8-Bit Opcode	16-Bit Address (MSB, LSB)	
8-Bit Opcode	8-Bit Address	8-Bit Data
8-Bit Opcode	8-Bit Address	8-Bit Address

The first instruction format, shown in the first row of Table 2-5, is used by the LJMP and LCALL instructions. These instructions change program execution unconditionally to an absolute address. The instructions use an 8-bit opcode, leaving room for a 16-bit destination address.

The second three-byte instruction format, shown in the second row of Table 2-5, is used by these two addressing modes:

- Destination Direct Source Immediate (`ADD [7], 5`)
- Destination Indexed Source Immediate (`ADD [X+7], 5`)

The third three-byte instruction format, shown in the third row of Table 2-5, is for the Destination Direct Source Direct addressing mode, which is used by only one instruction. This instruction format uses an 8-bit opcode followed by two 8-bit addresses. The first address is the destination address in RAM, while the second address is the source address in RAM. Here is an example of this instruction:

```
MOV [7], [5]
```

2.6 Register Definitions

The register shown here is associated with the CPU Core (M8C). The register description has an associated register table showing the bit structure. The grayed out bits in the table are reserved bits and are not detailed in the register description that follows. Always write reserved bits with a value of '0'.

2.6.1 CPU_F Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
x,F7h	CPU_F	PgMode[1:0]			XIO		Carry	Zero	GIE	RL : 02

LEGEND
L The AND F, expr; OR F, expr; and XOR F, expr flag instructions can be used to modify this register.
x An "x" before the comma in the address field indicates that this register can be read or written to no matter what bank is used.

The M8C Flag Register (CPU_F) provides read access to the M8C flags.

Bits 7 and 6: PgMode[1:0]

PgMode determines how the CUR_PP, STK_PP, and IDX_PP registers are used in forming effective RAM addresses for Direct Address mode and Indexed Address mode operands. PgMode also determines whether the stack page is determined by the STK_PP or IDX_PP register. (See the "Register Definitions" on page 34 in the RAM Paging chapter.)

Bit 4: XIO

The IO Bank Select bit, also known as the register bank select bit, is used to select the register bank that is active for a register read or write. This bit allows the PSoC device to have 512 8-bit registers and can be thought of as the ninth address bit for registers. The address space accessed when the XIO bit is set to '0' is called the ***user space***, while the address space accessed when the XIO bit is set to '1' is called the ***configuration space***.

Bit 2: Carry

The Carry flag bit is set or cleared in response to the actions of several instructions. It can also be manipulated by the flag-logic opcodes (for example, OR F, 4). See the *PSoC Designer Assembly Language User Guide* for more details.

Bit 1: Zero

The Zero flag bit is set or cleared in response to the result of several instructions. It can also be manipulated by the flag-logic opcodes (for example, OR F, 2). See the *PSoC Designer Assembly Language User Guide* for more details.

Bit 0: GIE

The state of the Global Interrupt Enable bit determines whether interrupts (by way of the interrupt request (IRQ)) will be recognized by the M8C. This bit is set or cleared by the user using the flag-logic instructions (for example, OR F, 1). GIE is also cleared automatically when an interrupt is processed, after the flag byte has been stored on the stack, preventing nested interrupts. If desired, the bit can be set in an ***interrupt service routine (ISR)***.

For GIE=1, the M8C samples the IRQ input for each instruction. For GIE=0, the M8C ignores the IRQ.

For additional information, refer to the CPU_F register on page 176.

2.6.2 Related Registers

These registers are related to the M8C block:

- CPU_SCR1 register on page 178.
- CPU_SCR0 register on page 179.

3. RAM Paging

This chapter explains the PSoC device's use of RAM Paging and its associated registers. For a complete table of the RAM paging registers, refer to the "Summary Table of the Core Registers" on page 24. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

3.1 Architectural Description

The M8C is an 8-bit CPU with an 8-bit memory address bus. The memory address bus allows the M8C to access up to 256 bytes of SRAM, to increase the amount of available SRAM and preserve the M8C ***assembly*** language. The CY8C20x34/24 PSoC device has 256 bytes of SRAM with two pages of memory.

To take full advantage of the paged memory architecture of the PSoC device, several registers must be used and two CPU_F register bits must be managed. However, the Power On Reset (POR) value for all of the paging registers and CPU_F bits is zero. This places the PSoC device in a mode identical to existing PSoC devices with only 256 bytes of SRAM. It is not necessary to understand all of the Paging registers to take advantage of the additional SRAM available in some devices. Very simple modifications to the reset state of the memory paging logic can be made to begin to take advantage of the additional SRAM pages.

The memory paging architecture consists of five areas:

- Stack Operations
- Interrupts
- MVI Instructions
- Current Page Pointer
- Indexed Memory Page Pointer

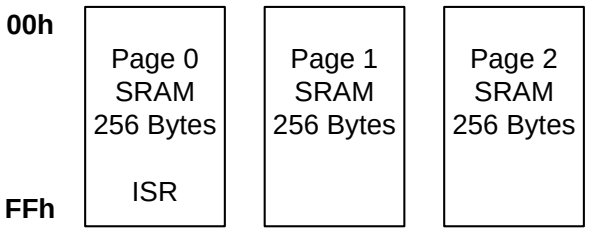
The first three of these areas have no dependency on the CPU_F register's PgMode bits and are covered in the next subsections after Basic Paging. The function of the last two depend on the CPU_F PgMode bits and are covered last.

3.1.1 Basic Paging

To increase the amount of SRAM, the M8C accesses memory page bits. The memory page bits are located in the CUR_PP register and allow for selection of one of eight SRAM pages. In addition to setting the page bits, Page mode must be enabled by setting the CPU_F[7] bit. If Page mode is not enabled, the page bits are ignored and all non-stack memory access is directed to Page 0.

Once Page mode is enabled and the page bits are set, all instructions that operate on memory access the SRAM page indicated by the page bits. The exceptions to this are the instructions that operate on the stack and the MVI instructions: PUSH, POP, LCALL, RETI, RET, CALL, and MVI. See the description of Stack Operations and MVI Instructions below for a more detailed discussion.

Figure 3-1. Data Memory Organization



3.1.2 Stack Operations

As mentioned previously, the paging architecture's reset state puts the PSoC in a mode that is identical to that of a 256 byte PSoC device. Therefore, upon reset, all memory accesses are set to Page 0. The SRAM page that stack operations use is determined by the value of the three least significant bits (LSb) of the stack page pointer register (STK_PP). Stack operations have no dependency on the PgMode bits in the CPU_F register. Stack operations are those that use the Stack Pointer (SP) to calculate their affected address. Refer to the *PSoC Designer Assembly Language User Guide* for more information on all M8C instructions.

Treat stack memory accesses as a special case. If they are not, the stack could be fragmented across several pages. To prevent the stack from fragmenting, all instructions that operate on the stack automatically use the page indicated by the STK_PP register. Therefore, if a CALL is encountered in the program, the PSoC device automatically pushes the program counter onto the stack page indicated by STK_PP. Once the program counter is pushed, the SRAM paging mode automatically switches back to the pre-call mode. All other stack operations, such as RET and POP, follow the same rule as CALL. The stack is confined to a single SRAM page and the Stack Pointer wraps from 00h to FFh and FFh to 00h. The user code must ensure that the stack is not damaged due to stack wrapping.

Because the value of the STK_PP register can be changed at any time, it is theoretically possible to manage the stack in such a way as to allow it to grow beyond one SRAM page or manage multiple stacks. However, the only supported use of the STK_PP register is when its value is set prior to the first stack operation and not changed again.

3.1.3 Interrupts

Interrupts, in a multi-page SRAM PSoC device, operate the same as interrupts in a 256-byte PSoC device. However, because the CPU_F register is automatically set to 0x00 on an interrupt and because of the non-linear nature of interrupts in a system, other parts of the PSoC memory paging architecture can be affected.

Interrupts are an abrupt change in program flow. If no special action is taken on interrupts by the PSoC device, the ***interrupt service routine (ISR)*** could be thrown into any SRAM page. To prevent this problem, the special addressing modes for all memory accesses, except for stack and MVI, are disabled when an ISR is entered. The special addressing modes are disabled when the CUP_F register is cleared. At the end of the ISR, the previous SRAM addressing mode is restored when the CPU_F register value is restored by the RETI instruction.

All interrupt service ***routine*** code starts execution in SRAM Page 0. If it is necessary for the ISR to change to another SRAM page, it can be accomplished by changing the values of the CPU_F[7:6] bits to enable the special SRAM addressing modes. However, any change made to the CUR_PP, IDX_PP, or STK_PP registers persists after the ISR returns. Therefore, the ISR should save the current value of any paging register it modifies and restore its value before the ISR returns.

3.1.4 MVI Instructions

MVI instructions use data page pointers of their own (MVR_PP and MVW_PP). This allows a data buffer to be located away from other program variables, but accessible without changing the Current Page Pointer (CUR_PP).

An MVI instruction performs three memory operations. Both forms of the MVI instruction access an address in SRAM that holds the data pointer (a memory read 1st access), incrementing that value and then storing it back in SRAM (a memory write 2nd access). This pointer value must reside in the current page, just as all other non-stack and non-indexed operations on memory must. However, the third memory operation uses the MVx_PP register. This third memory access can be either a read or a write, depending on which MVI instruction is used. The MVR_PP pointer is used for the MVI instruction that moves data into the accumulator. The MVW_PP pointer is used for the MVI instruction that moves data from the accumulator into SRAM. The MVI pointers are always enabled, regardless of the state of the Flag register page bits (CPU_F register).

3.1.5 Current Page Pointer

The Current Page Pointer is used to determine which SRAM page should be used for all memory accesses. Normal memory accesses are those not covered by other pointers including all non-stack, non-MVI, and non-indexed memory access instructions. The normal memory access instructions have the SRAM page they operate on determined by the value of the CUR_PP register. By default, the CUR_PP register has no affect on the SRAM page that is used for normal memory access, because all normal memory access is forced to SRAM Page 0.

The upper bit of the PgMode bits in the CPU_F register determine whether or not the CUR_PP register affects normal memory access. When the upper bit of the PgMode bits is set to ‘0’, all normal memory access is forced to SRAM Page 0. This mode is automatically enabled when an Interrupt Service Routine (ISR) is entered. This is because, before the ISR is entered, the M8C pushes the current value of the CPU_F register onto the stack and then clears the CPU_F register. Therefore, by default, any normal memory access in an ISR is guaranteed to occur in SRAM Page 0.

When the RETI instruction is executed to end the ISR, the previous value of the CPU_F register is restored, restoring the previous page mode. Note that this ISR behavior is the default and that the PgMode bits in the CPU_F register can be changed while in an ISR. If the PgMode bits are changed while in an ISR, the pre-ISR value is still restored by the RETI; but if the CUR_PP register is changed in the ISR, the ISR is also required to restore the value before executing the RETI instruction.

When the upper bit of the PgMode bits is set to '1', all normal memory access is forced to the SRAM page indicated by the value of the CUR_PP register. Table 3-1 gives a summary of the PgMode bit values and the corresponding Memory Paging mode.

3.1.6 Index Memory Page Pointer

The source indexed and destination indexed addressing modes to SRAM are treated as a unique addressing mode in a PSoC device with more than one page of SRAM. An example of an indexed addressing mode is the MOV A, [X+expr] instruction. Note that register access also has indexed addressing; however, those instructions are not affected by the SRAM paging architecture.

Important Note If you are not using assembly to program a PSoC device, be aware that the ***compiler*** writer may restrict the use of some memory paging modes. Review the conventions in your compiler's user guide for more information on restrictions or conventions associated with memory paging modes.

Indexed SRAM accesses operate in one of three modes:

- Index memory access modes are forced to SRAM Page 0.
- Index memory access modes are directed to the SRAM page indicated by the value in the STK_PP register.
- Index memory access is forced to the SRAM page indicated by the value in the IDX_PP register.

The mode is determined by the value of the PgMode bits in the CPU_F register. However, the final SRAM page that is used also requires setting either the Stack Page Pointer (STK_PP) register or the Index Page Pointer (IDX_PP) register. The table below shows the three indexed memory access modes. The third column of the table is provided for reference only.

Table 3-1. CPU_F PgMode Bit Modes

CPU_F PgMode Bits	Current SRAM Page	Indexed SRAM Page	Typical Use
00b	0	0	ISR*
01b	0	STK_PP	ISR with variables on stack
10b	CUR_PP	IDX_PP	
11b	CUR_PP	STK_PP	

* Mode used by SROM functions initiated by the SSC instruction.

After reset, the PgMode bits are set to 00b. In this mode, index memory accesses are forced to SRAM Page 0, just as they would be in a PSoC device with only 256 bytes of SRAM. This mode is also automatically enabled when an interrupt occurs in a PSoC device and is therefore considered the default ISR mode. This is because before the ISR is entered, the M8C pushes the current value of the CPU_F register on to the stack and then clears the CPU_F register. Therefore, by default, any indexed memory access in an ISR is guaranteed to occur in SRAM Page 0. When the RETI instruction is executed to end the ISR, the previous value of the CPU_F register is restored and the previous page mode is then also restored. Note that this ISR behavior is the default and that the PgMode bits in the CPU_F register may be changed while in an ISR. If the PgMode bits are changed while in an ISR, the pre-ISR value is still restored by the RETI; but if the STK_PP or IDX_PP registers are changed in the ISR, the ISR is also required to restore the values before executing the RETI instruction.

The most likely PgMode bit change, while in an ISR, is from the default value of 00b to 01b. In the 01b mode, indexed memory access is directed to the SRAM page indicated by the value of the STK_PP register. By using the PgMode, the value of the STK_PP register is not required to be modified. The STK_PP register is the register that determines which SRAM page the stack is located on. The 01b paging mode is intended to provide easy access to the stack, while in an ISR, by setting the CPU_X register (just X in the instruction format) equal to the value of SP using the MOV X, SP instruction.

The two previous paragraphs covered two of the three indexed memory access modes: STK_PP and forced to SRAM Page 0. Note, as shown in Table 3-1, that the STK_PP mode for indexed memory access is available under two PgMode settings. The 01b mode is intended for ISR use and the 11b mode is intended for non-ISR use. The third indexed memory access mode requires the PgMode bits to be set to 10b. In this mode indexed memory access is forced to the SRAM page indicated by the value of the IDX_PP register.

3.2 Register Definitions

The registers listed here are associated with RAM Paging and are in address order. The register descriptions have an associated register table showing the bit structure for that register. The grayed out bits in the tables are reserved bits and are not detailed in the register descriptions. Always write reserved bits with a value of '0'. For a complete table of RAM Paging registers, refer to the "Summary Table of the Core Registers" on page 24.

3.2.1 CUR_PP Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,D0h	CUR_PP								Page Bit	RW : 00

The Current Page Pointer Register (CUR_PP) is used to set the effective SRAM page for normal memory accesses in a multi-SRAM page PSoC device.

Bit 0: Page Bit. This bit affects the SRAM page that is accessed by an instruction when the CPU_F[7:0] bits have a value of either 10b or 11b. Source indexed and destination indexed addressing modes, as well as stack instructions, are never affected by the value of the CUR_PP register. (See the STK_PP and IDX_PP registers for more information.)

The source indirect post increment and destination indirect post increment addressing modes, better know as MVI, are only partially affected by the value of the CUR_PP register. For MVI instructions, the pointer address is in the SRAM page indicated by CUR_PP, but the address pointed to may be in another SRAM page.

See the MVR_PP and MVW_PP register descriptions for more information. For additional information, refer to the CUR_PP register on page 162.

3.2.2 STK_PP Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,D1h	STK_PP								Page Bit	RW : 00

The Stack Page Pointer Register (STK_PP) is used to set the effective SRAM page for stack memory accesses in a multi-SRAM page PSoC device.

Bit 0: Page Bit. This bit has the potential to affect two types of memory access.

The purpose of this register is to determine which SRAM page the stack is stored on. In the reset state, this register's value is 0x00 and the stack will therefore be in SRAM Page 0. However, if the STK_PP register value is changed, the next stack operation will occur on the SRAM page indicated by the new STK_PP value. Therefore, the value of this register should be set early in the program and never be changed. If the program changes the STK_PP value after the stack has grown, the program must ensure that the STK_PP value is restored when needed.

Note The impact that the STK_PP register has on the stack is independent of the SRAM Paging bits in the CPU_F register.

The second type of memory accesses that the STK_PP register affects are indexed memory accesses when the CPU_F[7:6] bits are set to 11b. In this mode, source indexed and destination indexed memory accesses are directed to the stack SRAM page, rather than the SRAM page indicated by the IDX_PP register or SRAM Page 0.

For additional information, refer to the STK_PP register on page 163.

3.2.3 IDX_PP Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,D3h	IDX_PP								Page Bit	RW : 00

The Index Page Pointer Register (IDX_PP) is used to set the effective SRAM page for indexed memory accesses in a multi-SRAM page PSoC device.

Bits 0: Page Bit. This bit allows instructions, which use the source indexed and destination indexed address modes, to operate on an SRAM page that is not equal to the current SRAM page. However, the effect this register has on indexed addressing modes is only enabled when the CPU_F[7:6] is set to 10b.

When CPU_F[7:6] is set to 10b and an indexed memory access is made, the access is directed to the SRAM page indicated by the value of the IDX_PP register.

See the STK_PP register description for more information on other indexed memory access modes. For additional information, refer to the IDX_PP register on page 164.

3.2.4 MVR_PP Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,D4h	MVR_PP								Page Bit	RW : 00

The MVI Read Page Pointer Register (MVR_PP) is used to set the effective SRAM page for MVI read memory accesses in a multi-SRAM page PSoC device.

Bit 0: Page Bit. This bit is only used by the MVI A, [expr] instruction, not to be confused with the MVI [expr], A instruction covered by the MVW_PP register. This instruction is considered a read because data is transferred from SRAM to the microprocessor's A register (CPU_A).

When an MVI A, [expr] instruction is executed in a device with more than one page of SRAM, the SRAM address that is read by the instruction is determined by the value of the least significant bits in this register. However, the pointer for the MVI A, [expr] instruction is always located in the current SRAM page. See the *PSoC Designer Assembly Language User Guide* for more information on the MVI A, [expr] instruction.

The function of this register and the MVI instructions are independent of the SRAM Paging bits in the CPU_F register. For additional information, refer to the MVR_PP register on page 165.

3.2.5 MVW_PP Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,D5h	MVW_PP								Page Bit	RW : 00

The MVI Write Page Pointer Register (MVW_PP) is used to set the effective SRAM page for MVI write memory accesses in a multi-SRAM page PSoC device.

Bit 0: Page Bit. This bit is only used by the MVI [expr], A instruction, not to be confused with the MVI A, [expr] instruction covered by the MVR_PP register. This instruction is considered a write because data is transferred from the microprocessor's A register (CPU_A) to SRAM.

When an MVI [expr], A instruction is executed in a device with more than one page of SRAM, the SRAM address that is written by the instruction is determined by the value of the least significant bits in this register. However, the pointer for the MVI [expr], A instruction is always located in the current SRAM page. See the *PSoC Designer Assembly Language User Guide* for more information on the MVI [expr], A instruction.

The function of this register and the MVI instructions are independent of the SRAM Paging bits in the CPU_F register. For additional information, refer to the MVW_PP register on page 166.

3.2.6 Related Registers

- “CPU_F Register” on page 30.

4. Supervisory ROM (SROM)

This chapter discusses the Supervisory ROM (SROM) functions. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

4.1 Architectural Description

The SROM holds code that is used to boot the PSoC device, calibrate circuitry, and perform Flash operations. The functions provided by the SROM are called from code stored in the Flash or by device programmers.

The SROM is used to boot the part and provide ***interface*** functions to the Flash banks. (Table 4-1 lists the SROM functions.) The SROM functions are accessed by executing the Supervisory System Call instruction (SSC) which has an opcode of 00h. Before executing the SSC, the M8C's ***accumulator*** needs to load with the desired SROM function code from Table 4-1.

Attempting to access undefined functions causes a HALT. The SROM functions execute code with calls; therefore, the functions require stack space. With the exception of Reset, all of the SROM functions have a ***parameter block*** in SRAM that you must configure before executing the SSC.

Table 4-2 lists all possible parameter block variables. The meaning of each ***parameter***, with regards to a specific SROM function, is described later in this chapter. Because the SSC instruction clears the CPU_F PgMode bits, all parameter block variable addresses are in SRAM Page 0. The CPU_F value is automatically restored at the end of the SROM function.

The MVR_PP and the MVW_PP pointers are not disabled by clearing the CPU_F PgMode bits. Therefore, the POINTER parameter is interpreted as an address in the page indicated by the MVI page pointers, when the supervisory operation is called. This allows the data ***buffer*** used in the supervisory operation to be located in any SRAM page. (See the RAM Paging chapter on page 31 for more details regarding the MVR_PP and MVW_PP pointers.)

Table 4-1. List of SROM Functions

Function Code	Function Name	Stack Space Needed	Page
00h	SWBootReset	0	38
01h	ReadBlock	7	39
02h	WriteBlock	10	40
03h	EraseBlock	9	40
06h	TableRead	3	41
07h	CheckSum	3	41
08h	Calibrate0	4	41
09h	Calibrate1	3	42
0Ah	WriteAndVerify	7	42
0Fh	HWBootReset	3	39

Note ProtectBlock (described on page 40) and EraseAll (described on page 41) SROM functions are not listed in the table above because they depend upon external programming.

Table 4-2. SROM Function Variables

Variable Name	SRAM Address
KEY1 / RETURN CODE	0,F8h
KEY2	0,F9h
BLOCKID	0,FAh
POINTER	0,FBh
CLOCK	0,FCh
Reserved	0,FDh
DELAY	0,FEh
Reserved	0,FFh

Two important variables that are used for all functions are KEY1 and KEY2. These variables are used to help discriminate between valid SSCs and inadvertent SSCs. KEY1 must always have a value of 3Ah, while KEY2 must have the same value as the stack pointer when the SROM function begins execution. This would be the SP (Stack Pointer) value when the SSC opcode is executed, plus three. For all SROM functions except SWBootReset, if either of the keys do not match the expected values, the M8C will halt. The SWBootReset function does not check the key values. It only checks to see if the accumulator's value is 0x00.

The following code example puts the correct value in KEY1 and KEY2. The code is preceded by a HALT, to force the program to jump directly into the setup code and not accidentally run into it.

```
1.        halt
2. SSCOP: mov [KEY1], 3ah
3.        mov X, SP
4.        mov A, X
5.        add A, 3
6.        mov [KEY2], A
```

4.1.1 Additional SROM Feature

The SROM has this additional feature.

Return Codes: These aid in the determination of success or failure of a particular function. The return code is stored in KEY1's position in the parameter block. The CheckSum and TableRead functions do not have return codes because KEY1's position in the parameter block is used to return other data.

Table 4-3. SROM Return Code Meanings

Return Code Value	Description
00h	Success
01h	Function not allowed due to level of protection on the block.
02h	Software reset without hardware reset.
03h	Fatal error, SROM halted.

Note Read, write, and erase operations may fail if the target block is read or write protected. Block protection levels are set during device programming and cannot be modified from code in the PSoC device.

4.1.2 SROM Function Descriptions

4.1.2.1 SWBootReset Function

The SROM function SWBootReset is responsible for transitioning the device from a reset state to running ***user*** code. See "System Resets" on page 107 for more information on what events causes the SWBootReset function to execute.

The SWBootReset function is executed whenever the SROM is entered with an M8C accumulator value of 00h; the SRAM parameter block is not used as an input to the function. This happens, by design, after a ***hardware*** reset because the M8C's accumulator is reset to 00h or when user code executes the SSC instruction with an accumulator value of 00h.

If the ***checksum*** of the calibration data is valid, the SWBootReset function ends by setting the internal M8C registers to 00h, writing 00h to most SRAM addresses in SRAM Page 0, and then begins to execute user code at address 0000h. (See Table 4-4 and the following paragraphs for more information on which SRAM addresses are modified.) If the checksum is not valid, an internal reset is executed and the boot process starts over. If this condition occurs, the internal reset status bit (IRESS) is set in the CPU_SCR1 register.

In PSoC devices with more than 256 bytes of SRAM, no SRAM is modified by the SWBootReset function in SRAM pages numbered higher than '0'.

Table 4-4 documents the value of all the SRAM addresses in Page 0 after a successful SWBootReset. A value of "xx" indicates that the SRAM address is not modified by the SWBootReset function. A hex value indicates that the address should always have the indicated value after a successful SWBootReset. A "??" indicates that the value, after a SWBootReset, is determined by the value of the IRAMDIS bit in the CPU_SCR1 register. If IRAMDIS is not set, these addresses will be initialized to 00h. If IRAMDIS is set, these addresses will not be modified by a SWBootReset after a watchdog reset.

The IRAMDIS bit allows the preservation of variables even if a watchdog reset (WDR) occurs. The IRAMDIS bit is reset by all system resets except watchdog reset. Therefore, this bit is only useful for watchdog resets and not general resets.

Table 4-4. SRAM Map Post SWBootReset (00h)

Address	0	1	2	3	4	5	6	7
	8	9	A	B	C	D	E	F
0x0_	0x00	0x00	0x00	??	??	??	??	??
	??	??	??	??	??	??	??	??
0x1_	??	??	??	??	??	??	??	??
	??	??	??	??	??	??	??	??
0x2_	??	??	??	??	??	??	??	??
	??	??	??	??	??	??	??	??
0x3_	??	??	??	??	??	??	??	??
	??	??	??	??	??	??	??	??
0x4_	??	??	??	??	??	??	??	??
	??	??	??	??	??	??	??	??
0x5_	??	??	??	??	??	??	??	??
	??	??	??	??	??	??	??	??
0x6_	??	??	??	??	??	??	??	??
	??	??	??	??	??	??	??	??
0x7_	??	??	??	??	??	??	??	??
	??	??	??	??	??	??	??	??
0x8_	??	??	??	??	??	??	??	??
	??	??	??	??	??	??	??	??
0x9_	??	??	??	??	??	??	??	??
	??	??	??	??	??	??	??	??
0xA_	??	??	??	??	??	??	??	??
	??	??	??	??	??	??	??	??
0xB_	??	??	??	??	??	??	??	??
	??	??	??	??	??	??	??	??
0xC_	??	??	??	??	??	??	??	??
	??	??	??	??	??	??	??	??
0xD_	??	??	??	??	??	??	??	??
	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0xE_	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0xF_	0x00	0x00	0x00	0x00	0x00	0x00	??	??
	0x00 0x02	xx	0x00	0x00	0xn	xx	0x00	0x00

Address F8h is the return code byte for all SROM functions (except Checksum and TableRead); for this function, the only acceptable values are 00h and 02h. Address FCh is the fail count variable. After POR (Power on Reset), WDR, or XRES (External Reset), the variable is initialized to 00h by the SROM. Each time the checksum fails, the fail count is incremented. Therefore, if it takes two passes through SWBootReset to get a good checksum, the fail count is 01h.

4.1.2.2 HWBootReset Function

The HWBootReset function forces a hardware reset of the PSoC. A hardware rest causes all registers to return to their POR state. Then, the SROM SWBootReset function executes, followed by Flash code execution beginning at address 0x0000.

The HWBootReset function only requires that the CPU_A, KEY1, and KEY2 be setup correctly. As with all other SROM functions, if the setup is incorrect, the SROM executes a HALT. Then, either a POR, XRES, or WDR is needed to clear the HALT. See the System Resets chapter on page 123 for more information.

Table 4-5. HWBootReset Parameters (0Fh)

Name	Address	Type	Description
KEY1	0,F8h	RAM	3Ah
KEY2	0,F9h	RAM	Stack Pointer value+3, when SSC is executed.

4.1.2.3 ReadBlock Function

The ReadBlock function reads 64 contiguous bytes from Flash: a ***block***. The CY8C20x34/24 PSoC device has 8 KB of Flash and therefore has 128 64-byte blocks. Valid block IDs are 0x00 to 0x7F.

Table 4-6. Flash Memory Organization

PSoC Device	Amount of Flash	Amount of SRAM	Number of Blocks per Bank	Number of Banks
CY8C20x34/24	8 KB	512 Bytes	128	1

The first thing the ReadBlock function does is to check the protection bits to determine if the desired BLOCKID is readable. If read protection is turned on, the ReadBlock function exits setting the accumulator and KEY2 back to 00h. KEY1 has a value of 01h indicating a read failure.

If read protection is not enabled, the function reads 64 bytes from the Flash using a ROMX instruction and stores the results in SRAM using an MVI instruction. The 64 bytes are stored in SRAM, beginning at the address indicated by the value of the POINTER parameter. When the ReadBlock completes successfully, the accumulator, KEY1, and KEY2 will all have a value of 00h.

Note A MVI [expr], A is used to store the Flash block contents in SRAM; thus, you can the MVW_PP register to indicate which SRAM pages receive the data.

Table 4-7. ReadBlock Parameters (01h)

Name	Address	Type	Description
MVW_PP	0,D5h	Register	MVI write page pointer register
KEY1	0,F8h	RAM	3Ah
KEY2	0,F9h	RAM	Stack Pointer value+3, when SSC is executed.
BLOCKID	0,FAh	RAM	Flash block number
POINTER	0,FBh	RAM	Addresses in SRAM where returned data should be stored.

4.1.2.4 WriteBlock Function

The WriteBlock function stores data in the Flash. Data moves 64 bytes at a time from SRAM to Flash using this function. Before doing a write, you must successfully complete an EraseAll or an EraseBlock.

The first thing the WriteBlock function does is check the protection bits and determine if the desired BLOCKID is writeable. If write protection is turned on, the WriteBlock function will exit, setting the accumulator and KEY2 back to 00h. KEY1 will have a value of 01h, indicating a write failure. Write protection is set when the PSoC device is programmed externally and cannot be changed through the SSC function.

The BLOCKID of the ***Flash block***, where the data is stored, must be determined and stored at SRAM address FAh. Valid block IDs are 0x00 to 0x7F.

An MVI A, [expr] instruction is used to move data from SRAM into Flash. Therefore, the MVI read pointer (MVR_PP register) can be used to specify which SRAM page data is pulled from. Using the MVI read pointer and the parameter blocks POINTER value allows the SROM WriteBlock function to move data from any SRAM page into any Flash block.

The SRAM address, of the first of the 64 bytes to be stored in Flash, must be indicated using the POINTER variable in the parameter block (SRAM address FBh).

Finally, the CLOCK and DELAY value must be set correctly. The CLOCK value determines the length of the write ***pulse*** that will be used to store the data in the Flash. The CLOCK and DELAY values are dependent on the CPU speed and must be set correctly. Refer to "Clocking Strategy" on page 42 for additional information.

Table 4-8. WriteBlock Parameters (02h)

Name	Address	Type	Description
MVR_PP	0,D4h	Register	MVI read page pointer register.
KEY1	0,F8h	RAM	3Ah
KEY2	0,F9h	RAM	Stack Pointer value+3, when SSC is executed.
BLOCKID	0,FAh	RAM	Flash block number
POINTER	0,FBh	RAM	First of 64 addresses in SRAM, where the data to be stored in Flash is located prior to calling WriteBlock.
CLOCK	0,FCh	RAM	Clock divider used to set the write pulse width.
DELAY	0,FEh	RAM	For a CPU speed of 12 MHz set to 56h.

4.1.2.5 EraseBlock Function

The EraseBlock function is used to erase a block of 64 contiguous bytes in Flash.

The first thing the EraseBlock function does is check the protection bits and determine if the desired BLOCKID is writeable. If write protection is turned on, the EraseBlock function exits, setting the accumulator and KEY2 back to 00h. KEY1 has a value of 01h, indicating a write failure.

To set up the parameter block for the EraseBlock function, store the correct key values in KEY1 and KEY2. The block number to erased must be stored in the BLOCKID variable, and the CLOCK and DELAY values must be set based on the current CPU speed. For more information on setting the CLOCK and DELAY values, see "Clocking Strategy" on page 42.

Table 4-9. EraseBlock Parameters (03h)

Name	Address	Type	Description
KEY1	0,F8h	RAM	3Ah
KEY2	0,F9h	RAM	Stack Pointer value+3, when SSC is executed.
BLOCKID	0,FAh	RAM	Flash block number
CLOCK	0,FCh	RAM	Clock divider used to set the erase pulse width.
DELAY	0,FEh	RAM	For a CPU speed of 12 MHz set to 56h.

4.1.2.6 ProtectBlock Function

The PSoC devices offer Flash protection on a block-by-block basis. Table 4-10 lists the protection modes available. In the table, ER and EW are used to indicate the ability to perform external reads and writes (that is, by an external programmer). For internal writes, IW is used. Internal reading is always permitted by way of the ROMX instruction. The ability to read by way of the SROM ReadBlock function is indicated by SR.

In this table, note that all protection is removed by EraseAll.

Table 4-10. Protect Block Modes

Mode	Settings	Description	In PSoC Designer
00b	SR ER EW IW	Unprotected	U = Unprotected
01b	$\overline{SR}$ ER EW IW	Read protect	F = Factory upgrade
10b	$\overline{SR}$ ER $\overline{EW}$ IW	Disable external write	R = Field upgrade
11b	$\overline{SR}$ $\overline{ER}$ $\overline{EW}$ $\overline{IW}$	Disable internal write	W = Full protection

4.1.2.7 TableRead Function

The TableRead function gives the user access to part-specific data stored in the Flash during manufacturing. The Flash for these tables is separate from the program Flash and is not directly accessible.

One of the uses of the TableRead function is to retrieve the values needed to optimize Flash programming for temperature. More information about how to use these values is in the section titled “Clocking Strategy” on page 42.

4.1.2.8 EraseAll Function

The EraseAll function performs a series of steps that destroys the user data in the Flash banks and resets the protection block in each Flash bank to all zeros (the unprotected state). This function is only executed by an external programmer. If EraseAll is executed from code, the M8C will HALT without touching the Flash or protections. See Table 4-11.

Table 4-11. Flash Tables with Assigned Values in Flash Bank 0

	F8h	F9h	FAh	FBh	FCh	FDh	FEh	FFh
Table 0	Silicon ID							
Table 1	Voltage Reference Trim for 3.3V reg[1,EA]	IMO Trim for 3.3V reg[1,E8]	Room Temperature Calibration for 3.3V	Hot Temperature Calibration for 3.3V	Voltage Reference Trim for 5V reg[1,EA]	IMO Trim for 5V reg[1,E8]	Room Temperature Calibration for 5V	Hot Temperature Calibration for 5V
Table 2	Voltage Reference Trim for 2.7V reg[1,EA]	IMO Trim 12 MHz Vdd = 2.7V	Room Temperature Calibration for 2.7V	Hot Temperature Calibration for 2.7V	IMO Slow Trim 6 MHz Vdd = 3.3V	IMO Slow Trim 6 MHz Vdd = 2.7V	IMO Slow Trim 6 MHz Vdd = 5.0V	
Table 3	M (cold)	B (cold)	Mult (cold)	M (hot)	B (hot)	Mult (hot)	00h	01h

4.1.2.9 Checksum Function

The Checksum function calculates a 16-bit checksum over a user specifiable number of blocks, within a single ***Flash bank*** starting at block zero. The BLOCKID parameter is used to pass in the number of blocks to checksum. A BLOCKID value of ‘1’ calculates the checksum of only block 0, while a BLOCKID value of ‘0’ calculates the checksum of the entire Flash bank.

The 16-bit checksum is returned in KEY1 and KEY2. The parameter KEY1 holds the lower 8 bits of the checksum and the parameter KEY2 holds the upper 8 bits of the checksum.

Table 4-12. Checksum Parameters (07h)

Name	Address	Type	Description
KEY1	0,F8h	RAM	3Ah
KEY2	0,F9h	RAM	Stack Pointer value+3, when SSC is executed.
BLOCKID	0,FAh	RAM	Number of Flash blocks to calculate checksum on.

4.1.2.10 Calibrate0 Function

The Calibrate0 function transfers the calibration values stored in a special area of the Flash to their appropriate registers. This function may be executed at any time to set all calibration values back to their 5V values. However, it is unnecessary to call this function. This function is simply documented for completeness. 3.3V calibration values are accessed by way of the TableRead function, which is described in the section titled “TableRead Function” on page 41.

Table 4-13. Calibrate0 Parameters (08h)

Name	Address	Type	Description
KEY1	0,F8h	RAM	3Ah
KEY2	0,F9h	RAM	Stack Pointer value+3, when SSC is executed.

4.1.2.11 Calibrate1 Function

While the Calibrate1 function is a completely separate function from Calibrate0, they perform the same task, which is to transfer the calibration values stored in a special area of the Flash to their appropriate registers. What is unique about Calibrate1 is that it calculates a checksum of the calibration data and, if that checksum is determined as invalid, Calibrate1 causes a ***hardware reset*** by generating an internal reset. If this occurs, it is indicated by setting the Internal Reset Status bit (IRESS) in the CPU_SCR1 register.

The Calibrate1 function uses SRAM to calculate a checksum of the calibration data. The POINTER value is used to indicate the address of a 30-byte buffer used by this function. When the function completes, the 30 bytes are set to 00h.

An MVI A, [expr] and an MVI [expr], A instruction are used to move data between SRAM and Flash. Therefore, the MVI write pointer (MVW_PP) and the MVI read pointer (MVR_PP) must be specified to the same SRAM page to control the page of RAM used for the operations.

Calibrate1 was created as a subfunction of SWBootReset and the Calibrate1 function code was added to provide ***direct access***. For more information on how Calibrate1 works, see the "SWBootReset Function" on page 38.

This function may be executed at any time to set all calibration values back to their 5V values. However, it is unnecessary to call this function. This function is simply documented for completeness. This function has no argument to select between 5V and 3.3V calibration values; therefore, it always defaults to 5V values. 3.3V calibration values are accessed by way of the TableRead function, which is described in the section titled "TableRead Function" on page 41.

Table 4-14. Calibrate1 Parameters (09h)

Name	Address	Type	Description
KEY1	0,F8h	RAM	3Ah
KEY2	0,F9h	RAM	Stack Pointer value+3, when SSC is executed.
POINTER	0,FBh	RAM	First of 30 SRAM addresses used by this function.
MVR_PP	0,D4h	Register	MVI write page pointer
MVW_PP	0,D5h	Register	MVI read page pointer

4.1.2.12 WriteAndVerify Function

The WriteAndVerify function works exactly the same as the WriteBlock function with one exception. Once the write operation has completed, the SROM will then read back the contents of Flash and compare those values against the values in SRAM thus verifying that the write was successful. The write and verify is one SROM operation; therefore, the SROM is not exited until the verify is completed.

The parameters for this block are identical to the WriteBlock (see "WriteBlock Parameters (02h)" on page 40). If the verify operation fails, the 0x04 error code will be returned at SRAM address 0xF8. If the write fails, the 0x01 error code returns at SRAM address 0xF8.

Table 4-15. WriteAndVerify Parameters (0Ah)

Name	Address	Type	Description
KEY1	0,F8h	RAM	3Ah
KEY2	0,F9h	RAM	Stack Pointer value+3, when SSC is executed.
BLOCKID	0,FAh	RAM	Flash block number.
POINTER	0,FBh	RAM	First of 64 addresses in SRAM, where the data to be stored in Flash is located prior to calling WriteBlock.
CLOCK	0,FCh	RAM	Clock divider used to set the write pulse width.
DELAY	0,FEh	RAM	For a CPU speed of 12 MHz set to 56h.

4.2 Register Definitions

This chapter has no register detail information because there are no registers directly assigned to the Supervisory ROM.

4.2.1 Related Registers

- "STK_PP Register" on page 34.
- "MVR_PP Register" on page 35.
- "MVW_PP Register" on page 36.
- "CPU_SCR1 Register" on page 108.

4.3 Clocking Strategy

Successful programming and erase operations, on the Flash, require you to set the CLOCK and DELAY parameters correctly. To determine the proper value for the DELAY parameter only, you must consider CPU speed. Use three factors to determine the proper value for CLOCK: operating temperature, CPU speed, and characteristics of the individual device. Equations and additional information on calculating the DELAY and CLOCK values follow.

4.3.1 DELAY Parameter

To determine the proper value for the DELAY parameter, you must consider CPU speed during a Flash operation. Equation 1 displays the equation for calculating DELAY based on a CPU speed value. In this equation the units for CPU are hertz (Hz).

$$DELAY = \frac{100 \times 10^{-6} \cdot CPU - 80}{13}, \qquad \textbf{Equation 1}$$

$$3MHz \le CPU \le 12MHz$$

Equation 2 shows the calculation of the DELAY value for a CPU speed of 12 MHz. The numerical result of this calculation should be rounded to the nearest whole number. In the case of a 12 MHz CPU speed, the correct value for DELAY is 86 (0x56).

$$DELAY = \frac{100 \times 10^{-6} \cdot 12 \times 10^{6} - 80}{13}$$ **Equation 2**

4.3.2 CLOCK Parameter

The CLOCK parameter must be calculated using different equations for erase and write operations. The erase value for CLOCK must be calculated first. In Equation 3, the erase CLOCK value is indicated by a subscript E after the word CLOCK. In Equation 5, the write CLOCK value is indicated by a subscript W after the word CLOCK.

Before either CLOCK value can be calculated, the values for M, B, and Mult must be determined. These are device specific values that are stored in the Flash Table 3 and are accessed by way of the TableRead SROM function (see the "TableRead Function" on page 41). If the operating temperature is at or below 0°C, use the cold values. For operating temperatures at or above 0°C, use the hot values. See Table 4-11 for more information.

Equations for calculating the correct value of CLOCK for write operations are first introduced with the assumption that the CPU speed is 12 MHz. The equation for calculating the CLOCK value for an erase Flash operation is shown in Equation 3. In this equation the T has units of °C.

$$CLOCK_E = B - \frac{2M \cdot T}{256}$$ **Equation 3**

Using the correct values for B, M, and T, in the equation above, is required to achieve the endurance specifications of the Flash. However, for device programmers where this calculation is difficult to perform, the equation is simplified by setting T to 0°C and using the hot value for B and M. This simplification is acceptable only if the total number of erase write cycles are kept to less than 10 and the operation is performed near room temperature. When T is set to '0', Equation 3 simplifies to.

$$CLOCK_E = B$$ **Equation 4**

Once a value for the erase CLOCK value is determined, the write CLOCK value can be calculated. The equation to calculate the CLOCK value for a write is.

$$CLOCK_W = \frac{CLOCK_E \cdot Mult}{64}$$ **Equation 5**

In this equation, the correct value for Mult must be determined, based upon temperature, in the same way that the B and M values were determined for Equation 3.

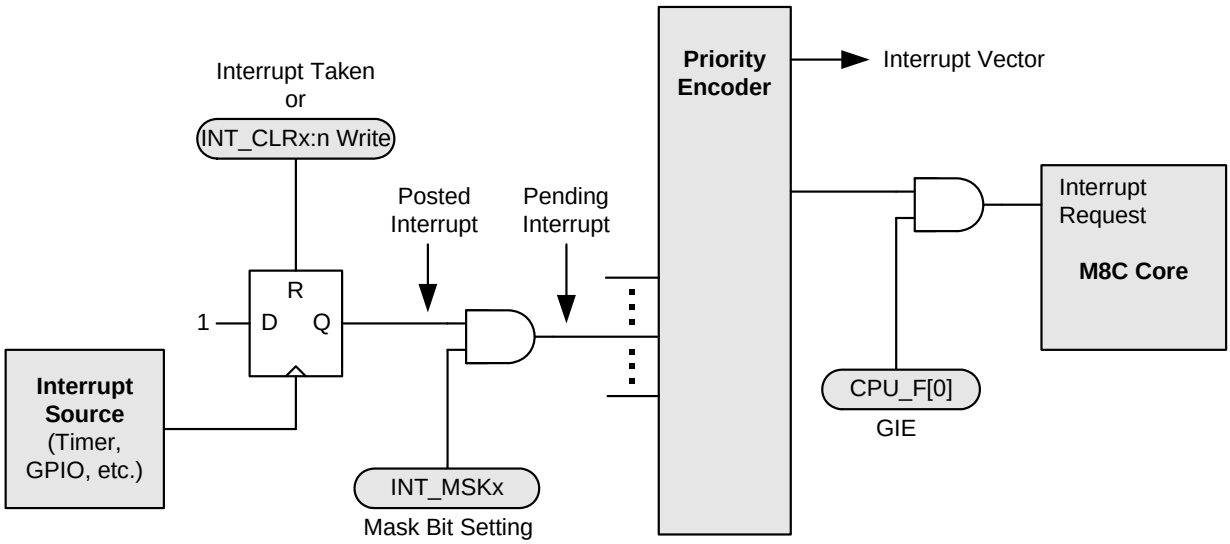
5. Interrupt Controller

This chapter presents the Interrupt Controller and its associated registers. The interrupt controller provides a mechanism for a hardware resource in PSoC mixed-signal array devices, to change program execution to a new address without regard to the current task being performed by the code being executed. For a complete table of the Interrupt Controller registers, refer to the "Summary Table of the Core Registers" on page 24. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

5.1 Architectural Description

A block diagram of the PSoC Interrupt Controller is shown in Figure 5-1, illustrating the concepts of ***posted interrupts*** and ***pending interrupts***.

Figure 5-1. Interrupt Controller Block Diagram



The sequence of events that occur during interrupt processing are.

1. An interrupt becomes active, either because (a) the interrupt condition occurs (for example, a timer expires), (b) a previously posted interrupt is enabled through an update of an interrupt ***mask*** register, or (c) an interrupt is pending and GIE is set from '0' to '1' in the CPU Flag register.
2. The current executing instruction finishes.
3. The internal interrupt routine executes, taking 13 cycles. During this time, these actions occur:
   - The PCH, PCL, and Flag register (CPU_F) are pushed onto the stack (in that order).
   - The CPU_F register is then cleared. Since this clears the GIE bit to '0', additional interrupts are temporarily disabled.
   - The PCH (PC[15:8]) is cleared to zero.
   - The interrupt vector is read from the interrupt controller and its value is placed into PCL (PC[7:0]). This sets the program counter to point to the appropriate address in the interrupt table (for example, 001Ch for the GPIO interrupt).
4. Program execution vectors to the interrupt table. Typically, a LJMP instruction in the interrupt table sends execution to the user's interrupt service routine (ISR) for this interrupt. (See "Instruction Set Summary" on page 26.)
5. The ISR executes. Note that interrupts are disabled since GIE = 0. In the ISR, interrupts can be re-enabled if desired by setting GIE = 1 (take care to avoid stack overflow in this case).
6. The ISR ends with a RETI instruction. This pops the Flag register, PCL, and PCH from the stack, restoring those registers. The restored Flag register re-enables interrupts since GIE = 1 again.
7. Execution resumes at the next instruction, after the one that occurred before the interrupt. However, if there are more pending interrupts, the subsequent interrupts are processed before the next normal program instruction.

Interrupt Latency. The time between the assertion of an enabled interrupt and the start of its ISR is calculated using this equation:

$$\begin{aligned} Latency = \quad & \textbf{Equation 1} \\ & Time\ for\ current\ instruction\ to\ finish + \\ & Time\ for\ M8C\ to\ change\ program\ counter\ to\ interrupt\ address + \\ & Time\ for\ LJMP\ instruction\ in\ interrupt\ table\ to\ execute. \end{aligned}$$

For example, if the 5-cycle JMP instruction is executing when an interrupt becomes active, the total number of CPU clock cycles before the ISR begins are:

$$\begin{aligned} & (1\ to\ 5\ cycles\ for\ JMP\ to\ finish) + \quad \textbf{Equation 2} \\ & (13\ cycles\ for\ interrupt\ routine) + \\ & (7\ cycles\ for\ LJMP) = 21\ to\ 25\ cycles. \end{aligned}$$

In the example above, at 24 MHz, 25 clock cycles take 1.042 µs.

Interrupt Priority. Interrupt priorities only come into consideration if more than one interrupt is pending during the same instruction cycle. In this case, the priority encoder (see Figure 5-1) generates an interrupt vector for the highest pendingpriority interrupt.

5.1.1 Posted versus Pending Interrupts

An interrupt is posted when its interrupt conditions occur. This results in the flip-flop in Figure 5-1 clocking in a '1'. The interrupt remains posted until the interrupt is taken or until it is cleared by writing to the appropriate INT_CLRx register.

A posted interrupt is not pending unless it is enabled by setting its interrupt mask bit (in the appropriate INT_MSKx register). All pending interrupts are processed by the Priority Encoder to determine the highest priority interrupt which is taken by the M8C if the Global Interrupt Enable bit is set in the CPU_F register.

Disabling an interrupt by clearing its interrupt mask bit (in the INT_MSKx register) does not clear a posted interrupt, nor does it prevent an interrupt from being posted. It simply prevents a posted interrupt from becoming pending.

It is especially important to understand the functionality of clearing posted interrupts, if the configuration of the PSoC device is changed by the application.

For example, if a block has a posted interrupt when it is enabled, and then disabled, the posted interrupt remains. It is good practice to use the INT_CLR register to clear posted interrupts before enabling or re-enabling a block.

5.2 Application Overview

The interrupt controller and its associated registers allow the user's code to respond to an interrupt from almost every functional block in the PSoC devices. Interrupts for all the digital blocks and each of the analog columns are available, as well as interrupts for supply voltage, sleep, variable clocks, and a general GPIO (pin) interrupt.

The registers associated with the interrupt controller allow interrupts to be disabled either globally or individually. The registers also provide a mechanism by which a user can ***clear*** all pending and posted interrupts, or clear individual posted or pending interrupts. A ***software*** mechanism is provided to set individual interrupts. Setting an interrupt by way of software is very useful during code development, when one may not have the complete hardware system necessary to generate a real interrupt.

This table lists the interrupts and priorities that are available in the PSoC devices.

Table 5-1. PSoC Device Interrupt Table

Interrupt Priority	Interrupt Address	Interrupt Name
0 (Highest)	0000h	Reset
1	0004h	Supply Voltage Monitor
2	0008h	Analog
3	000Ch	CapSense
4	0010h	Timer
5	0014h	GPIO
6	0018h	SPI
7	001Ch	I2C
8 (Lowest)	0020h	Sleep Timer

5.3 Register Definitions

These registers are associated with the Interrupt Controller and are listed in address order. The register descriptions have an associated register table showing the bit structure for that register. The grayed out bits in the tables are reserved bits and are not detailed in the register descriptions that follow. Always write reserved bits with a value of '0'. For a complete table of Interrupt Controller registers, refer to the "Summary Table of the Core Registers" on page 24.

5.3.1 INT_CLR0 Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,DAh	INT_CLR0	I2C	Sleep	SPI	GPIO	Timer	CapSense	Analog	V Monitor	RW : 00

The Interrupt Clear Register 0 (INT_CLR0) is used to enable the individual interrupt sources' ability to clear posted interrupts.

The INT_CLR0 register is similar to the INT_MSK0 register in that it holds a bit for each interrupt source. Functionally the INT_CLR0 register is similar to the INT_VC register, although its operation is completely independent. When the INT_CLR0 register is read, any bits that are set indicates an interrupt has been posted for that hardware resource. Therefore, reading this register gives the user the ability to determine all posted interrupts.

The Enable Software Interrupt (ENSWINT) bit in the INT_SW_EN register determines the way an individual bit value, written to an INT_CLR0 register, is interpreted. When ENSWINT is cleared (the default state), writing 1's to the INT_CLR0 register has no effect. However, writing 0's to the INT_CLR0 register, when ENSWINT is cleared, will cause the corresponding interrupt to clear. If the ENSWINT bit is set, any 0's written to the INT_CLR0 register is ignored. However, 1's written to the INT_CLR0 register, while ENSWINT is set, will cause an interrupt to post for the corresponding interrupt.

Software interrupts can aid in debugging interrupt service routines by eliminating the need to create system level interactions that are sometimes necessary to create a hardware-only interrupt.

Bit 7: I2C. This bit allows posted I2C interrupts to be read, cleared, or set.

Bit 6: Sleep. This bit allows posted sleep interrupts to be read, cleared, or set.

Bit 5: SPI. This bit allows posted SPI interrupts to be read, cleared, or set.

Bit 4: GPIO. This bit allows posted GPIO interrupts to be read, cleared, or set.

Bit 3: Timer. This bit allows posted Timer interrupts to be read, cleared, or set.

Bit 2: CapSense. This bit allows posted CapSense interrupts to be read, cleared, or set.

Bit 1: Analog. This bit allows posted analog interrupts to be read, cleared, or set.

Bit 0: V Monitor. This bit allows posted voltage monitor interrupts to be read, cleared, or set.

For additional information, refer to the INT_CLR0 register on page 170.

5.3.2 INT_MSK0 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,E0h	INT_MSK0	I2C	Sleep	SPI	GPIO	Timer	CapSense	Analog	V Monitor	RW : 00

The Interrupt Mask Register (INT_MSK0) is used to enable the individual interrupt sources' ability to create pending interrupts.

If cleared, each bit in an INT_MSK0 register prevents a posted interrupt from becoming a pending interrupt (input to the priority encoder). However, an interrupt can still post even if its mask bit is zero. All INT_MSK0 bits are independent of all other INT_MSK0 bits.

If an INT_MSK0 bit is set, the interrupt source associated with that mask bit may generate an interrupt that will become a pending interrupt. For example, if INT_MSK0[4] is set and at least one GPIO pin is configured to generate an interrupt, the interrupt controller will allow a GPIO interrupt request to post and become a pending interrupt for the M8C to respond to. If a higher priority interrupt is generated before the M8C responds to the GPIO interrupt, the higher priority interrupt is responded to before the GPIO interrupt.

Each interrupt source may require configuration at a block level. Refer to the corresponding chapter for each interrupt for any additional configuration information.

Bit 7: I2C. This bit allows I2C interrupts to be enabled or masked.

Bit 6: Sleep. This bit allows sleep interrupts to be enabled or masked.

Bit 5: SPI. This bit allows SPI interrupts to be enabled or masked.

Bit 4: GPIO. This bit allows GPIO interrupts to be enabled or masked.

Bit 3: Timer. This bit allows Timer interrupts to be enabled or masked.

Bit 2: CapSense. This bit allows CapSense interrupts to be enabled or masked.

Bit 1: Analog. This bit allows analog interrupts to be enabled or masked.

Bit 0: V Monitor. This bit allows voltage monitor interrupts to be enabled or masked.

For additional information, refer to the INT_MSK0 register on page 172.

5.3.3 INT_SW_EN Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,E1h	INT_SW_EN								ENSWINT	RW : 00

The Interrupt Software Enable Register (INT_SW_EN) is used to enable software interrupts.

Bit 0: ENSWINT. This bit is a special non-mask bit that controls the behavior of the INT_CLR0 register. See the INT_CLR0 register in this section for more information.

For additional information, refer to the INT_SW_EN register on page 173.

5.3.4 INT_VC Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,E2h	INT_VC	Pending Interrupt[7:0]								RC : 00

LEGEND
C Clearable register or bits.

The Interrupt Vector Clear Register (INT_VC) returns the next pending interrupt and clears all pending interrupts when written.

Bits 7 to 0: Pending Interrupt[7:0]. When the register is read, the ***least significant byte (LSB)*** of the highest priority pending interrupt is returned. For example, if the GPIO and I2C interrupts were pending and the INT_VC register was read, the value 14h is read. However, if no interrupts were pending, the value 00h is returned. This is the reset vector in the interrupt table; however, reading 00h from the INT_VC register should not be considered an indication that a system reset is pending. Rather, reading 00h from the INT_VC register simply indicates that there are no pending interrupts. The highest priority interrupt, indicated by the value returned by a read of the INT_VC register, is removed from the list of pending interrupts when the M8C services an interrupt.

Reading the INT_VC register has limited usefulness. If interrupts are enabled, a read to the INT_VC register would not be able to determine that an interrupt was pending before the interrupt was actually taken. However, while in an interrupt service routine, a user may wish to read the INT_VC register to see what the next interrupt are. When the INT_VC register is written, with any value, all pending and posted interrupts are cleared by asserting the clear line for each interrupt.

For additional information, refer to the INT_VC register on page 174.

5.3.5 Related Registers

- “CPU_F Register” on page 30.

6. General Purpose IO (GPIO)

This chapter discusses the General Purpose IO (GPIO) and its associated registers, which is the circuit responsible for interfacing to the IO pins of a PSoC device. The GPIO blocks provide the interface between the M8C core and the outside world. They offer a large number of configurations to support several types of input/output (IO) operations for both digital and analog systems. For a complete table of the GPIO registers, refer to the "Summary Table of the Core Registers" on page 24. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

6.1 Architectural Description

The GPIO in the CY8C20x34/24 PSoC device is uniform, except the Port 1 GPIO has stronger high drive and an option for regulated output level. These distinctions are discussed in more detail in the section "Port 1 Distinctions" on page 52.

6.1.1 General Description

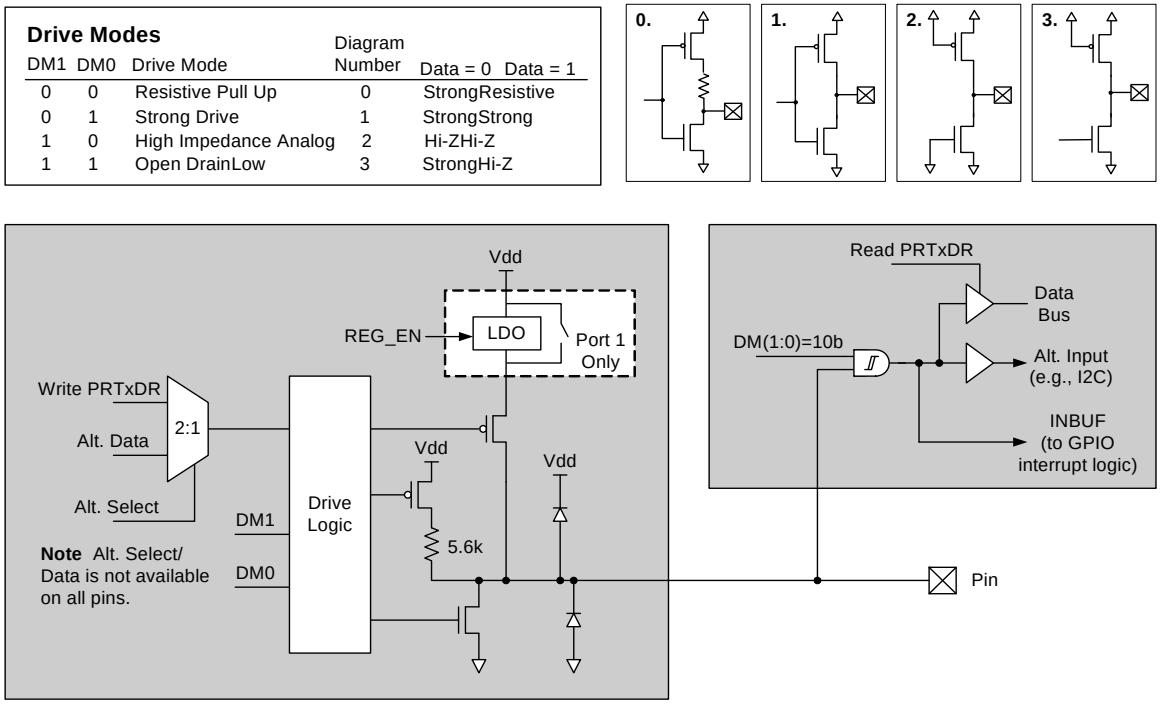
The GPIO contains input buffers, output drivers, and configuration logic for connecting the PSoC device to the outside world.

IO Ports are arranged with (up to) 8 bits per port. Each full port contains eight identical GPIO blocks. Each GPIO block can be used for the following types of IO:

- Digital IO (digital input and output controlled by software)
- Analog IO

Each IO pin also has several drive modes, as well as interrupt capabilities. All GPIO pins provide both digital IO and analog input capability.

Figure 6-1. GPIO Block Diagram



All IO contain the capability to connect to an internal analog bus. This is described in detail in the IO Analog Multiplexer chapter on page 81.

Certain pins contain an option to bypass the normal data path and output from an internal source. An example is I2C outputs. These are described in “Data Bypass” on page 54.

6.1.2 Digital IO

One of the basic operations of the GPIO ports is to allow the M8C to send information out of the PSoC device and get information into the M8C from outside the PSoC device. This is accomplished by way of the port data register (PRTxDR). Writes from the M8C to the PRTxDR register store the data state, one bit per GPIO. In the standard non-bypass mode, the pin drivers drive the pin in response to this data bit, with a drive strength determined by the Drive mode setting. The actual voltage on the pin depends upon the Drive mode and the external load.

The M8C reads the value of a port by reading the PRTxDR register address. When the M8C reads the PRTxDR register address, the current value of the pin voltage is translated into a logic value and returned to the M8C. Note that the pin voltage can represent a different logic value than the last value written to the PRTxDR register. This is an important distinction to remember in situations such as the use of a read modify write to a PRTxDR register. Examples of read modify write instructions include AND, OR, and XOR.

Here is an example of how a read modify write, to a PRTxDR register, could have an unexpected and even indeterminate result in certain systems. Consider a scenario where all bits of Port 1 on the PSoC device are in the strong 0 resistive 1 Drive mode; so that in some cases, the system the PSoC is in may pull down one of the bits by an external driver.

```
mov   reg[PRT1DR], 0xFF
and   reg[PRT1DR], 0x7F
```

In the first line of code above, writing a 0xFF to the port causes the PSoC to drive all pins high through a resistor. This does not affect any bits that happen to be strongly driven low by the system the PSoC is in. However, in the second line of code, it cannot guarantee that only bit 7 is the one set to a strong 0 (zero). Because the AND instruction will first read the port, any bits that are currently driven low externally will be read as a ‘0’. These zeros will then be written back to the port. When this happens, the pin will go in to a strong 0 state; therefore, if the external low drive condition ends in the system, the PSoC will keep the pin value at a logic 0.

6.1.3 Analog and Digital Inputs

Analog signals can pass into the PSoC device core from PSoC device pins through a resistive path. For analog signals, the GPIO block is typically configured into a High Impedance Analog Drive mode (High-Z). This mode turns off the Schmitt trigger on the input path, which may reduce power consumption and decrease internal switching noise when using a particular IO as an analog input.

All modes, except High Impedance Analog, allow digital inputs. The most useful digital input modes are Resistive Pull Up (DM1, DM0 = 00b with Data = 1) or a fully high impedance input using open drain (DM1, DM0 = 11b with Data = 1).

6.1.4 Port 1 Distinctions

Port 1 has two differences from the other GPIO ports. It has stronger high drive and it has an option for regulating all outputs to a 3V level when in strong drive mode. Refer to the device datasheet for the different current sourcing specifications of Port 1.

By setting the REG_EN bit in the IO_CFG register, Port 1 can be configured to drive strong high to a regulated 3V level, when device Vdd is above 3V. If REG_EN is set low, Port 1 pins drive to Vdd in strong drive mode.

In Resistive High Drive mode ([DM1, DM0] = 00), the pins pull up to the ***chip*** Vdd level regardless of the regulator setting for this port. Only Strong Drive mode allows for the outputs to be driven to the regulated level. When the REG_EN bit is set high, pins configured for strong drive will drive to 3V, while those in resistive pull-up mode will drive to Vdd.

6.1.5 GPIO Block Interrupts

Each GPIO pin can be individually configured for interrupt capability. Pins are configured by pin interrupt enables and also by a chip-wide selection for interrupt state with this global selection. Pins can be set to interrupt when the pin is low or when it changes from the last time it was read. The block provides an open-drain interrupt output (INTO) that is connected to other GPIO blocks in a wire-OR fashion.

All pin interrupts that are wire-OR’ed together are tied to the same system GPIO interrupt. Therefore, if interrupts are enabled on multiple pins, the user’s interrupt service routine must provide a mechanism to determine which pin was the source of the interrupt.

Using a GPIO interrupt requires the following steps:

1. Set the Interrupt mode (IOINT bit in the IO_CFG register).
2. Enable the bit interrupt in the GPIO block.
3. Set the mask bit for the (global) GPIO interrupt.
4. Assert the overall Global Interrupt Enable.

The first step sets a common interrupt mode for all pins.

The second step, bit interrupt enable, is set at the GPIO pin level (that is, at each port pin), by way of the PRTxIE registers.

The last two steps are common to all interrupts and are described in the Interrupt Controller chapter on page 45.

At the GPIO block level, asserting the INTO line depends only on the bit interrupt enable and the state of the pin relative to the chosen Interrupt mode. At the PSoC device level, due to their wire-OR nature, the GPIO interrupts are neither true edge-sensitive interrupts nor true level-sensitive interrupts. They are considered edge-sensitive for asserting, but level-sensitive for release of the wire-OR interrupt line.

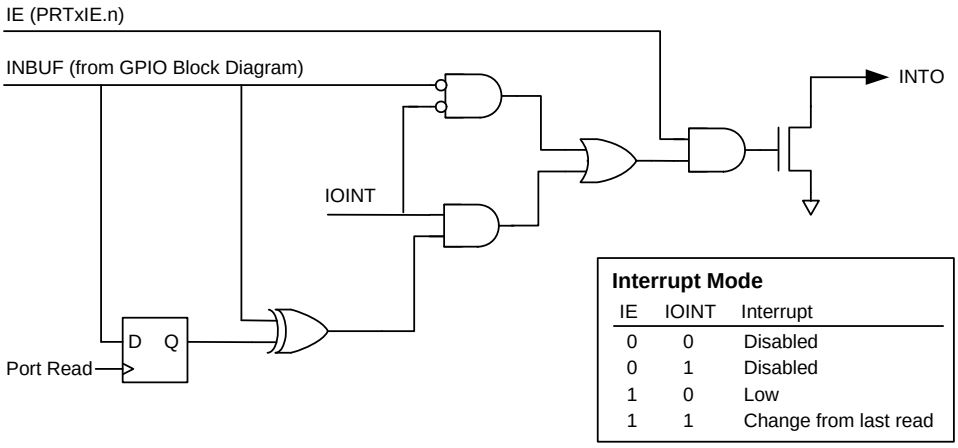
If no GPIO interrupts are asserting, a GPIO interrupt will occur whenever a GPIO pin interrupt enable is set and the GPIO pin transitions (if not already transitioned) appropriately high or low to match the interrupt mode configuration. Once this happens, the INTO line will pull low to assert the GPIO interrupt. This assumes the other system-level enables are on, such as setting the global GPIO interrupt enable and the Global Interrupt Enable. Setting the pin interrupt enable may immediately assert INTO, if the Interrupt mode conditions are already being met at the pin.

Once INTO pulls low, it will continue to hold INTO low until one of these conditions change: (a) the pin interrupt enable is cleared; (b) the voltage at pin transitions to the opposite state; (c) in interrupt-on-change mode, the GPIO data register is read thus setting the local interrupt level to the opposite state; or (d) the Interrupt mode is changed so that the current pin state does not create an interrupt. Once one of these conditions is met, the INTO releases. At this point, another GPIO pin (or this pin again) could assert its INTO pin, pulling the common line low to assert a new interrupt.

Note the following behavior from this level-release feature. If one pin is asserting INTO and then a second pin asserts its INTO, when the first pin releases its INTO, the second pin is already driving INTO and thus no change is seen (that is, no new interrupt would be asserted on the GPIO interrupt). Care must be taken, using polling or the states of the GPIO pin and Global Interrupt Enables, to catch all interrupts among a set of wire-OR GPIO blocks.

Figure 6-2 shows the interrupt logic portion of the block.

Figure 6-2. GPIO Interrupt Logic Diagram



Interrupt Mode

IE	IOINT	Interrupt
0	0	Disabled
0	1	Disabled
1	0	Low
1	1	Change from last read

6.1.5.1 *Interrupt Modes*

GPIO interrupts use the IOINT bit from the IO_CFG register. The setting of IOINT determines the interrupt mode for all GPIO.

Interrupt mode IOINT=0 means that the block will assert the GPIO interrupt line (INTO) when the pin voltage is low, providing the block's bit interrupt enable line is set (high).

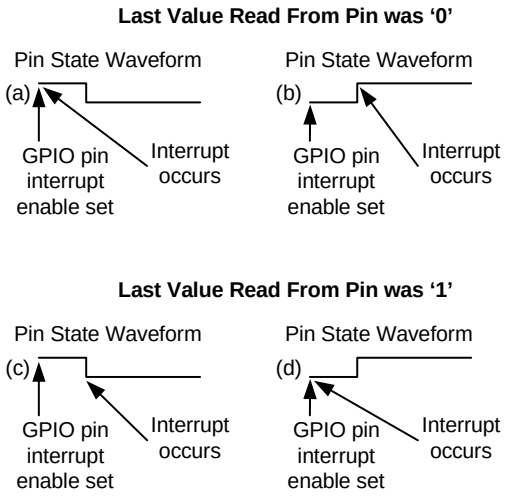
Interrupt mode IOINT=1 means that the block will assert the interrupt line (INTO) when the pin voltage is the opposite of the last state read from the pin, providing the block's bit interrupt enable line is set high. This mode switches between low mode and high mode, depending on the last value that was read from the port during reads of the data register (PRTxDR). If the last value read from the GPIO was '0', the GPIO pin will subsequently be in Interrupt High mode. If the last value read from the GPIO was '1', the GPIO will then be in Interrupt Low mode.

Table 6-1. GPIO Interrupt Modes

IE	IOINT	Description
0	0	Bit interrupt disabled, INTO de-asserted
0	1	Bit interrupt disabled, INTO de-asserted
1	0	Assert INTO when PIN = low
1	1	Assert INTO when PIN = change from last read

Figure 6-3 assumes that the GIE is set, GPIO interrupt mask is set, and that the IOINT bit has been set to high. The Change Interrupt mode relies on the value of an internal read register to determine if the pin state has changed. Therefore, the port that contains the GPIO in question must be read during every interrupt service routine. If the port is not read, the Interrupt mode will act as if it is in high mode when the latch value is '0' and low mode when the latch value is '1'.

Figure 6-3. GPIO Interrupt Mode IOINT = 1



6.1.6 Data Bypass

GPIO pins can be configured to either output data through CPU writes to the PRTxDR registers, or to bypass the port's data register and output data from internal functions instead. The bypass path is shown in Figure 6-1 by the Alt Data input, which is selected by the Alt Select input. These data bypass options are selected in one of two ways.

For internal functions such as I2C and SPI, the hardware automatically selects the bypass mode for the required pins when the function is enabled. In addition, some bypass outputs are selected by the user through the OUT_P1 register. For these, the pin is configured for data bypass when the register bit is set high, which allows an internal signal to be driven to the pin.

For all bypass modes, the desired drive mode of the pin must be configured separately for each pin, with the PRTxDM1 and PRTxDM0 registers.

6.2 Register Definitions

The following registers are associated with the General Purpose IO (GPIO) and are listed in address order. The register descriptions have an associated register table showing the bit structure for that register. The bits in the tables that are grayed out are reserved bits and are not detailed in the register descriptions that follow. Reserved bits should always be written with a value of '0'. For a complete table of GPIO registers, refer to the "Summary Table of the Core Registers" on page 24.

For a selected GPIO block, the individual registers are addressed in the "Summary Table of the Core Registers" on page 24. In the register names, the 'x' is the port number, configured at the PSoC device level (x = 0 to 7 typically). All register values are readable, except for the PRTxDR register; reads of this register return the pin state instead of the register bit state.

6.2.1 PRTxDR Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,xxh	PRTxDR	Data[7:0]								RW : 00

LEGEND
xx An "x" after the comma in the address field indicates that there are multiple instances of the register. For an expanded address listing of these registers, refer to the "Core Register Summary" on page 24.

The Port Data Register (PRTxDR) allows for write or read access of the current logical equivalent of the voltage on the pin.

Bits 7 to 0: Data[7:0]. Writing the PRTxDR register bits set the output drive state for the pin to high (for Data = 1) or low (Data = 0), unless a bypass mode is selected (see "Data Bypass" on page 54).

Reading the PRTxDR register returns the actual pin state, as seen by the input buffer. This may not be the same as the expected output state, if the load pulls the pin more strongly than the pin's configured output drive. See "Digital IO" on page 52 for a detailed discussion of digital IO.

For additional information, refer to the PRTxDR register on page 138.

6.2.2 PRTxIE Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,xxh	PRTxIE	Interrupt Enables[7:0]								RW : 00

LEGEND
xx An "x" after the comma in the address field indicates that there are multiple instances of the register. For an expanded address listing of these registers, refer to the "Core Register Summary" on page 24.

The Port Interrupt Enable Register (PRTxIE) is used to enable/disable interrupts from individual GPIO pins.

Bits 7 to 0: Interrupt Enables[7:0]. A '1' enables the INTO output at the block and a '0' disables INTO so it is only High-Z. In the enabled state, the type of GPIO edge that actually causes an interrupt is set by the IOINT bit in the IO_CFG register.

For additional information, refer to the PRTxIE register on page 139.

6.2.3 PRTxDMx Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,xxh	PRTxDM0	Drive Mode 0[7:0]								RW : 00
1,xxh	PRTxDM1	Drive Mode 1[7:0]								RW : FF

LEGEND
xx An "x" after the comma in the address field indicates that there are multiple instances of the register. For an expanded address listing of these registers, refer to the "Core Register Summary" on page 24.

The Port Drive Mode Bit Registers (PRTxDM0 and PRTxDM1) are used to specify the Drive mode for GPIO pins.

Bits 7 to 0: Drive Mode x[7:0]. In the PRTxDMx registers there are four possible drive modes for each port pin. Two mode bits are required to select one of these modes, and these two bits are spread into two different registers (PRTxDM0 and PRTxDM1). The bit position of the effected port pin (for example, Pin[2] in Port 0) is the same as the bit position of each of the two drive mode register bits that control the Drive mode for that pin (for example, bit[2] in PRT0DM0 and bit[2] in PRT0DM1). The two bits from the two registers are treated as a group. These are referred to as DM1 and DM0, or together as DM[1:0]. Drive modes are shown in the table shown below.

For analog IO, the Drive mode should be set to the High-Z analog mode, 10b. The 10b mode disables the block's digital input buffer so no crowbar current flows, even when the analog input is not close to either power rail. If the 10b Drive mode is used, the pin will always be read as a zero by the CPU and the pin will not be able to generate a useful interrupt. (It is not strictly required that a High-Z mode be selected for analog operation.)

When digital inputs are needed on the same pin as analog inputs, the 11b Drive mode should be used with the corresponding data bit (in the PRTxDR register) set high.

Drive Modes		Pin State	Description
DM1	DM0		
0	0	Resistive pull up	Resistive high, strong low
0	1	Strong drive	Strong high, strong low
1	0	High impedance, analog (**reset state**)	High-Z high and low, digital input disabled (for zero power) (**reset state**)
1	1	Open drain low	High-Z high (digital input enabled), strong low.

The GPIO provides a default Drive mode of high impedance, analog (High-Z). This is achieved by forcing the reset state of all PRTxDM1 registers to FFh.

For additional information, refer to the PRTxDM0 register on page 180, and the PRTxDM1 register on page 181.

6.2.4 IO_CFG Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,DCh	IO_CFG							REG_EN	IOINT	RW : 00

The Input/Output Configuration Register (IO_CFG) is used to configure the Port 1 output regulator and set the Interrupt mode for all GPIO.

Bit 1: REG_EN. The Register Enable bit (REG_EN) controls the regulator on Port 1 outputs.

Bit 0: IOINT. This bit sets the GPIO Interrupt mode for all pins in the CY8C20x34/24 PSoC devices. GPIO interrupts are controlled at each pin by the PRTxIE registers, and also by the global GPIO bit in the INT_MSK0 register.

For additional information, refer to the IO_CFG register on page 184.

7. Internal Main Oscillator (IMO)

This chapter presents the Internal Main Oscillator (IMO) and its associated register. The IMO produces clock signals of 6 MHz and 12 MHz. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

7.1 Architectural Description

The Internal Main Oscillator (IMO) outputs two clocks: a SYSCLK (that can be the internal 6/12 MHz clock or an external clock) and a 12/24 MHz clock called SYSCLKx2 that runs at twice the SYSCLK frequency. In the absence of a high-precision input source from a crystal oscillator, the accuracy of the internal 6/12 MHz clocks will be ±5% over temperature and voltage variation for the 4.75V to 5.25V or the 3.0V to 3.6V voltage ranges. The accuracy will be ±8.3% over temperature and voltage variation in the 2.4V to 3.0V range. This accuracy is only achieved in the selected voltage range. No external components are required to achieve this level of accuracy.

The IMO can be disabled when using an external clocking source. Registers for controlling these operations are found in the Digital Clocks chapter on page 89.

Lower frequency SYSCLK settings are available by setting the Slow IMO (SLIMO) bit in the CPU_SCR1 register. With this bit set and the corresponding factory trim value applied to the IMO_TR register, SYSCLK can be lowered to 6 MHz. This offers lower device power consumption for systems that can operate with the reduced system clock. Slow IMO mode is discussed further in the “Application Overview” on page 57.

7.2 Application Overview

To save power, the IMO frequency can be reduced from 12 MHz to 6 MHz using the SLIMO bit in the CPU_SCR1 register, in conjunction with the Trim values in the IMO_TR register. Both methods are described below.

7.2.1 Trimming the IMO

An 8-bit register (IMO_TR) is used to trim the IMO. Bit 0 is the LSB and bit 7 is the MSB. The trim step size is approximately 80 kHz.

A factory trim setting is loaded into the IMO_TR register at boot time for 4.75V to 5.25V operation. For operation in the voltage ranges below 4.75V, user code must modify the contents of this register with values stored in Flash bank 0 as shown in Table 4-11 on page 41. This is done with a Table Read command to the Supervisory ROM.

7.2.2 Engaging Slow IMO

Forcing the CPU_SCR1 register bit 4 high engages the Slow IMO feature. The IMO will immediately drop to a lower frequency. Factory trim settings are stored in Flash bank 0 as shown in Table 4-11 on page 41 for the following voltage/frequency combinations.

Table 7-1. Slow IMO

Voltage	Normal IMO Frequency	Slow IMO Frequency
4.75V to 5.25 V	12 MHz	6 MHz
3.0V to 3.3V	12 MHz	6 MHz
2.4V to 3.0V	12 MHz	6 MHz

A TableRead command to the Supervisory ROM is performed to set the IMO to the different frequencies. See the “TableRead Function” on page 41.

7.3 Register Definitions

The following register is associated with the Internal Main Oscillator (IMO). The register description has an associated register table showing the bit structure for that register.

7.3.1 IMO_TR Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,E8h	IMO_TR	Trim[7:0]								W : 00

The Internal Main Oscillator Trim Register (IMO_TR) is used to manually center the oscillator's output to a target frequency.

The PSoC device specific value for 5.0V operation is loaded into the IMO_TR register at boot time. The Internal Main Oscillator will operate within specified tolerance over a voltage range of 4.75V to 5.25V, with no modification of this register. If the PSoC device is operated at a lower voltage, user code must modify the contents of this register. For operation in the voltage range of 3.3V ±0.3V or 2.7V ±0.3V, this is accomplished with a TableRead command to the Supervisory ROM, which will supply a trim value for operation in this range. For operation between these voltage ranges, user code can interpolate the best value using both available factory trim values.

It is strongly recommended that the user not alter the register value, unless Slow IMO mode is used.

Bits 7 to 0: Trim[7:0]. These bits are used to trim the Internal Main Oscillator. A larger value in this register increases the speed of the oscillator.

For additional information, refer to the IMO_TR register on page 190.

7.3.2 Related Registers

- "OSC_CR2 Register" on page 94.
- "CPU_SCR1 Register" on page 108.

8. Internal Low Speed Oscillator (ILO)

This chapter briefly explains the Internal Low Speed Oscillator (ILO) and its associated register. The Internal Low Speed Oscillator produces a 32 kHz clock. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

8.1 Architectural Description

The Internal Low Speed Oscillator (ILO) is an oscillator with a nominal frequency of 32 kHz. It is used to generate sleep wake-up interrupts and watchdog resets. This oscillator can also be used as a clocking source for the digital PSoC blocks.

The oscillator operates in three modes: normal power, low power, and off. The Normal Power mode consumes more current to produce a more accurate frequency. The Low Power mode is always used when the part is in a power down (sleep) state.

8.2 Register Definitions

The following register is associated with the Internal Low Speed Oscillator (ILO). The register description has an associated register table showing the bit structure. The bits in the table that are grayed out are reserved bits and are not detailed in the register description that follows. Always write reserved bits with a value of '0'.

8.2.1 ILO_TR Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,E9h	ILO_TR			Bias Trim[1:0]		Freq Trim[3:0]				W : 00

The Internal Low Speed Oscillator Trim Register (ILO_TR) sets the adjustment for the internal low speed oscillator.

The device-specific value, placed in the trim bits of this register at boot time, is based on factory testing. ***It is strongly recommended that the user not alter the values in the register***.

Bits 5 and 4: Bias Trim[1:0]. These bits are used to set the bias current in the PTAT Current Source. Bit 5 gets inverted, so that a medium bias is selected when both bits are '0'. The ***bias current*** is set according to the information in this table:

Bias Current	Bias Trim [1:0]
Medium Bias	00b
Maximum Bias	01b
Minimum Bias	10b
Reserved	11b

Bits 3 to 0: Freq Trim[3:0]. These bits are used to trim the frequency. Bit 0 is the LSb and bit 3 is the MSb. Bit 3 gets inverted inside the register.

For additional information, refer to the ILO_TR register on page 191.

9. Sleep and Watchdog

This chapter discusses the Sleep and Watchdog operations and their associated registers. For a complete table of the Sleep and Watchdog registers, refer to the "Summary Table of the Core Registers" on page 24. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

9.1 Architectural Description

Device components that are involved in Sleep and Watchdog operation are the selected 32 kHz clock, the sleep timer, the Sleep bit in the CPU_SCR0 register, the sleep circuit (to sequence going into and coming out of sleep), the bandgap refresh circuit (to periodically refresh the reference voltage during sleep), and the ***watchdog timer***.

The goal of Sleep operation is to reduce average power consumption as much as possible. The system has a sleep state that can be initiated under firmware control. In this state, the CPU is stopped at an instruction boundary and the 6/12 MHz oscillator (IMO), the Flash memory module, and bandgap voltage reference are powered down. The only blocks that remain in operation are the 32 kHz oscillator, ***PSoC blocks*** clocked from the 32 kHz clock selection, and the supply voltage monitor circuit.

The system can only wake up from sleep as a result of an interrupt or reset event. The sleep timer can provide periodic interrupts to allow the system to wake up, poll peripherals, or do real-time functions, and then go to sleep again. The GPIO (pin) interrupt, supply monitor interrupt, and analog interrupt are examples of ***asynchronous*** interrupts that can also be used to wake the system up.

The Watchdog Timer (WDT) circuit is designed to assert a ***hardware reset*** to the device after a pre-programmed interval, unless it is periodically serviced in firmware. In the event that an unexpected execution path is taken through the code, this functionality serves to reboot the system. It can also restart the system from the CPU halt state.

Once the WDT is enabled, it can only be disabled by an External Reset (XRES) or a Power On Reset (POR). A WDT reset will leave the WDT enabled. Therefore, if the WDT is used in an application, all code (including initialization code) must be written as though the WDT is enabled.

9.1.1 Sleep Timer

The Sleep Timer is a 15-bit up counter clocked by the 32 kHz clock source. This timer is always enabled. The exception to this is within an ***ICE*** (in-circuit ***emulator***) in ***debugger*** mode and when the Stop bit in the CPU_SCR0 is set; the sleep timer is disabled, so that the user will not get continual watchdog resets when a breakpoint is hit in the debugger environment.

If the associated sleep timer interrupt is enabled, a periodic interrupt to the CPU is generated based on the sleep interval selected from the OSC_CR0 register. The sleep timer functionality does not need to be directly associated with the sleep state. It can be used as a general purpose timer interrupt regardless of sleep state.

The reset state of the sleep timer is a count value of all zeros. There are two ways to reset the sleep timer. Any hardware reset, (that is, POR, XRES, or Watchdog Reset (WDR) will reset the sleep timer. There is also a method that allows the user to reset the sleep timer in firmware. A write of 38h to the RES_WDT register clears the sleep timer.

Note Any write to the RES_WDT register also clears the watchdog timer.

Clearing the sleep timer may be done at anytime to synchronize the sleep timer operation to CPU processing. A good example of this is after POR. The CPU hold-off, due to voltage ramp and others, may be significant. In addition, a significant amount of program initialization may be required. However, the sleep timer starts counting immediately after POR and will be at an arbitrary count when user code begins execution. In this case, it may be desirable to clear the sleep timer before enabling the sleep interrupt initially to ensure that the first sleep period is a full interval.

9.2 Application Overview

The following are notes regarding sleep as it relates to firmware and application issues.

Note 1 If an interrupt is pending, enabled, and scheduled to be taken at the instruction boundary after the write to the sleep bit, the system will not go to sleep. The instruction will still execute, but it will not be able to set the Sleep bit in the CPU_SCR0 register. Instead, the interrupt will be taken and the effect of the sleep instruction is ignored.

Note 2 The Global Interrupt Enable (CPU_F register) does not need to be enabled to wake the system out of sleep state. Individual interrupt enables, as set in the interrupt mask registers, are sufficient. If the Global Interrupt Enable is not set, the CPU will not service the ISR associated with that interrupt. However, the system will wake up and continue executing instructions from the point at which it went to sleep. In this case, the user must manually clear the pending interrupt or subsequently enable the Global Interrupt Enable bit and let the CPU take the ISR. If a pending interrupt is not cleared, it will be continuously asserted. Although the Sleep bit may be written and the sleep sequence executed as soon as the device enters Sleep mode, the Sleep bit is cleared by the pending interrupt and Sleep mode is exited immediately.

Note 3 On wake up, the instruction immediately after the sleep instruction is executed before the interrupt service routine (if enabled). The instruction after the sleep instruction is pre-fetched before the system actually goes to sleep. Therefore, when an interrupt occurs to wake the system up, the pre-fetched instruction is executed and then the interrupt service routine is executed. (If the Global Interrupt Enable is not set, instruction execution continues where it left off before sleep.)

Note 4 Analog power must be turned off by firmware before going to sleep, to achieve the smallest sleep current. The system sleep state does not control the analog array. There are individual power controls for each analog block and global power controls in the reference block. These power controls must be manipulated by firmware.

Note 5 If the Global Interrupt Enable bit is disabled, it can be safely enabled just before the instruction that writes the sleep bit. It is usually undesirable to get an interrupt on the instruction boundary, just before writing the sleep bit. This means that on the return from interrupt, the sleep command will be executed, possibly bypassing any firmware preparations that must be made in order to go to sleep. To prevent this, disable interrupts before preparations are made. After sleep preparations, enable global interrupts and write the sleep bit with the two consecutive instructions as follows.

```
and f,~01h          // disable global interrupts
                    // (prepare for sleep, could
                    // be many instructions)
or f,01h            // enable global interrupts
mov reg[ffh],08h    // Set the sleep bit
```

Due to the timing of the Global Interrupt Enable instruction, it is not possible for an interrupt to occur immediately after that instruction. The earliest the interrupt could occur is after the next instruction (write to the Sleep bit) has been executed. Therefore, if an interrupt is pending, the sleep instruction is executed; but as described in Note 1, the sleep instruction will be ignored. The first instruction executed after the ISR is the instruction after sleep.

9.3 Register Definitions

The following registers are associated with Sleep and Watchdog and are listed in address order. Each register description has an associated register table showing the bit structure for that register. The bits that are grayed out in the tables below are reserved bits and are not detailed in the register descriptions. Always write reserved bits with a value of '0'. For a complete table of the Sleep and Watchdog registers, refer to the "Summary Table of the Core Registers" on page 24.

9.3.1 RES_WDT Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,E3h	RES_WDT	WDSL_Clear[7:0]								W : 00

The Reset Watchdog Timer Register (RES_WDT) is used to clear the watchdog timer (a write of any value) and clear both the watchdog timer and the sleep timer (a write of 38h).

Bits 7 to 0: WDSL_Clear[7:0]. The Watchdog Timer (WDT) write-only register is designed to timeout at three roll-over events of the sleep timer. Therefore, if only the WDT is cleared, the next Watchdog Reset (WDR) will occur anywhere from two to three times the current sleep interval setting. If the sleep timer is near the beginning of its count, the watchdog timeout will be closer to three times. However, if the sleep timer is very close to its ***terminal count***, the watchdog timeout will be closer to two times. To ensure a full three times timeout, both the WDT and the sleep timer may be cleared. In applications that need a real-time clock, and thus cannot reset the sleep timer when clearing the WDT, the duty cycle at which the WDT must be cleared should be no greater than two times the sleep interval.

For additional information, refer to the RES_WDT register on page 175.

9.3.2 SLP_CFG Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,EBh	SLP_CFG	PSSDC[1:0]								RW : 00

The Sleep Configuration Register (SLP_CFG) is used to set the sleep duty cycle.

The value placed in this register is based on factory testing. ***It is strongly recommended that the user not alter the register value.***

Bits 7 and 6: PSSDC[1:0]. The Power System Sleep Duty Cycle bits are used to set the sleep duty cycle. These bits should not be altered.

For additional information, refer to the SLP_CFG register on page 193.

9.3.3 Related Registers

- "INT_MSK0 Register" on page 49.
- "OSC_CR0 Register" on page 93.
- "ILO_TR Register" on page 59.
- "CPU_SCR0 Register" on page 109.
- "CPU_SCR1 Register" on page 108.

9.4 Timing Diagrams

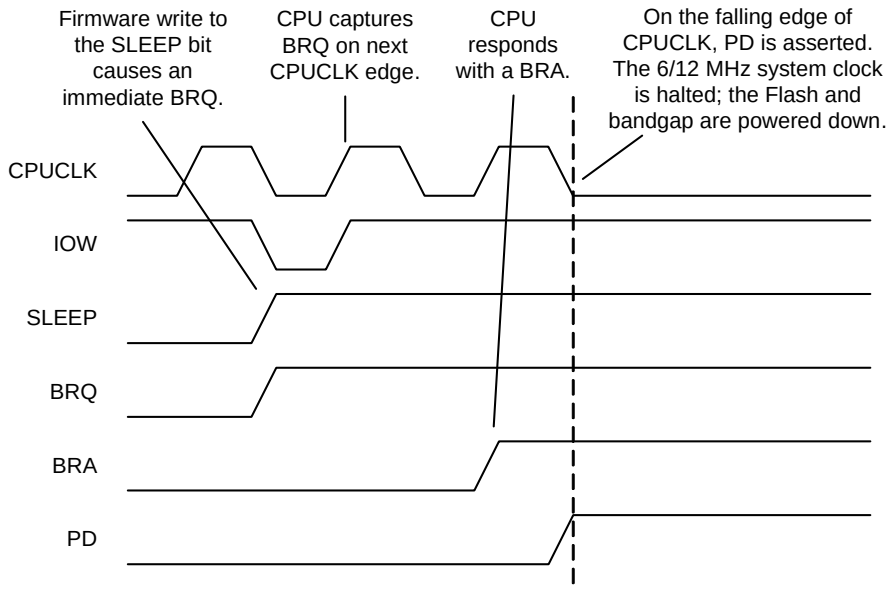
9.4.1 Sleep Sequence

The Sleep bit, in the CPU_SCR0 register, is an input into the sleep logic circuit. This circuit is designed to sequence the device into and out of the hardware sleep state. The hardware sequence to put the device to sleep is shown in Figure 9-1 and is defined as follows.

1. Firmware sets the Sleep bit in the CPU_SCR0 register. The Bus Request (BRQ) signal to the CPU is immediately asserted: This is a request by the system to halt CPU operation at an instruction boundary.
2. The CPU issues a Bus Request Acknowledge (BRA) on the following ***positive edge*** of the CPU clock.
3. The sleep logic waits for the following ***negative edge*** of the CPU clock and then asserts a system-wide Power Down (PD) signal. In Figure 9-1, the CPU is halted and the system-wide power down signal is asserted.

The system-wide PD signal controls three major circuit blocks: the Flash memory module, the Internal Main Oscillator (6/12 MHz oscillator that is also called the IMO), and the bandgap voltage reference. These circuits transition into a zero power state. The only operational circuits on the PSoC device are the ILO, the bandgap refresh circuit, and the supply voltage monitor circuit.

Figure 9-1. Sleep Sequence



9.4.2 Wakeup Sequence

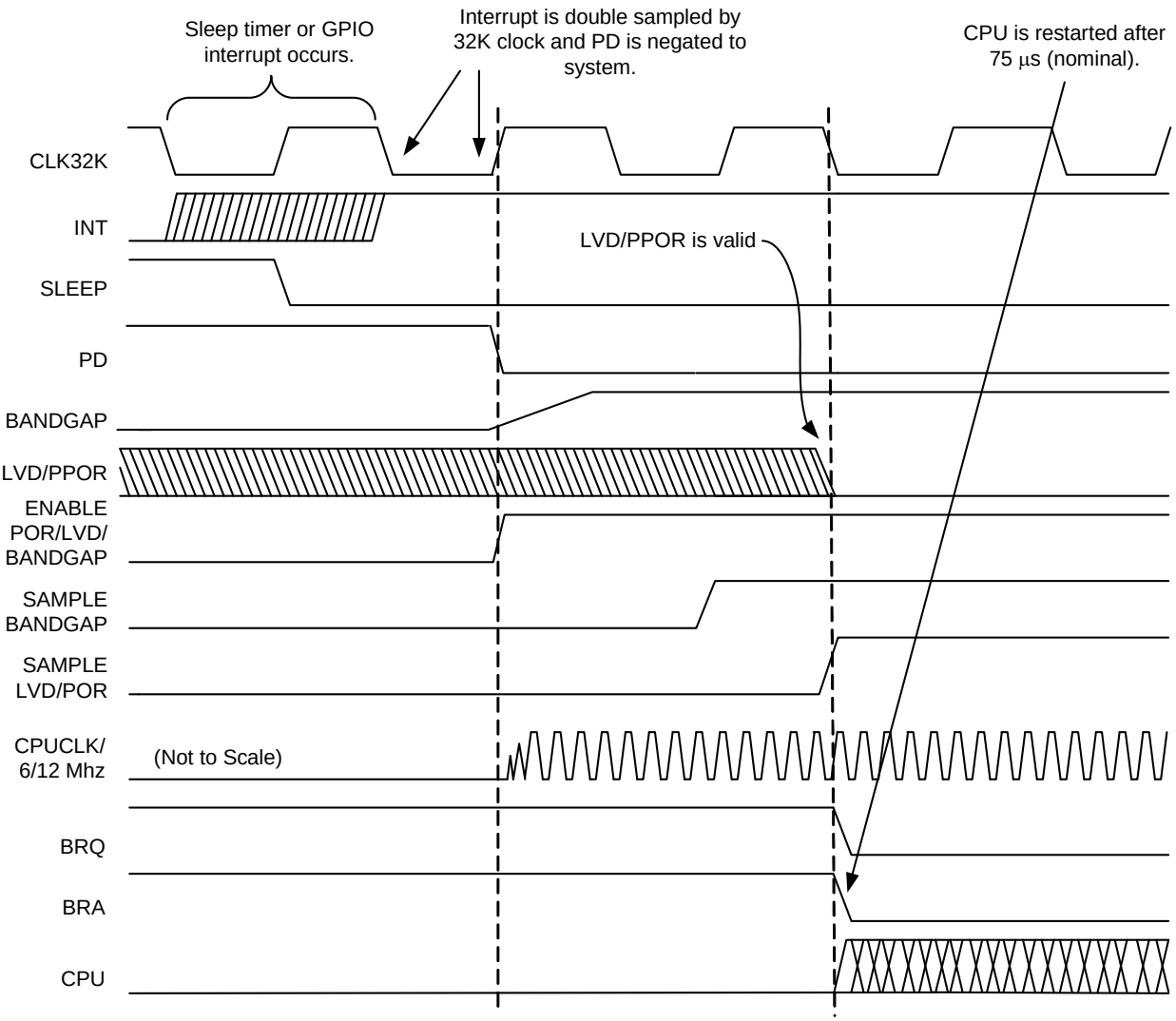
Once asleep, the only event that can wake the system up is an interrupt. The Global Interrupt Enable of the CPU flag register does not need to be set. Any unmasked interrupt will wake the system up. It is optional for the CPU to actually take the interrupt after the wakeup sequence.

The wakeup sequence is synchronized to the 32 kHz clock for purposes of sequencing a startup delay, to allow the Flash memory module enough time to power up before the CPU asserts the first read access. Another reason for the delay is to allow the IMO, bandgap, and LVD/POR circuits time to settle before actually being used in the system. As shown in Figure 9-2, the wakeup sequence is as follows.

1. The wakeup interrupt occurs and is synchronized by the negative edge of the 32 kHz clock.
2. At the following positive edge of the 32 kHz clock, the system-wide PD signal is negated. The Flash memory module, IMO, and bandgap any POR/LVD circuits are all powered up to a normal operating state.
3. At the next positive edge of the 32 kHz clock, the values of the bandgap are settled and sampled.
4. At the following negative edge of the 32 kHz clock (after about 15 µs, nominal), the values of the POR/LVD signals have settled and are sampled. The BRQ signal is negated by the sleep logic circuit. On the following CPU clock, BRA is negated by the CPU and instruction execution resumes.

The wakeup times (interrupt to CPU operational) ranges from two to three 32 kHz cycles or 61 - 92 µs (nominal).

Figure 9-2. Wakeup Sequence

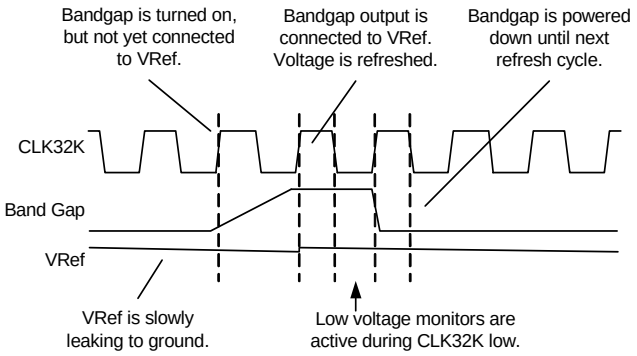


9.4.3 Bandgap Refresh

During normal operation, the bandgap circuit provides a voltage reference (VRef) to the system, for use in the analog blocks, Flash, and ***low voltage detect (LVD)*** circuitry. Normally, the bandgap output is connected directly to the VRef signal. However, during sleep, the ***bandgap reference*** generator block and LVD circuits are completely powered down. The bandgap and LVD blocks are periodically re-enabled during sleep in order to monitor for low voltage conditions. This is accomplished by turning on the bandgap periodically, allowing it time to start up for a full 32 kHz clock period, and connecting it to VRef to refresh the reference voltage for the following 32 kHz clock period as shown in Figure 9-3.

During the second 32 kHz clock period of the refresh cycle, the LVD circuit is allowed to settle during the ***high time*** of the 32 kHz clock. During the low period of the second 32 kHz clock, the LVD interrupt is allowed to occur.

Figure 9-3. Bandgap Refresh Operation



The rate at which the refresh occurs is related to the 32 kHz clock and controlled by the Power System Sleep Duty Cycle (PSSDC). Table 9-1 enumerates the available selections. The default setting (256 sleep timer counts) is applicable for many applications, giving a typical average device current under 5 µA.

Table 9-1. Power System Sleep Duty Cycle Selections

PSSDC	Sleep Timer Counts	Period (Nominal)
00b (default)	256	8 ms
01b	1024	31.2 ms
10b	64	2 ms
11b	16	500 µs

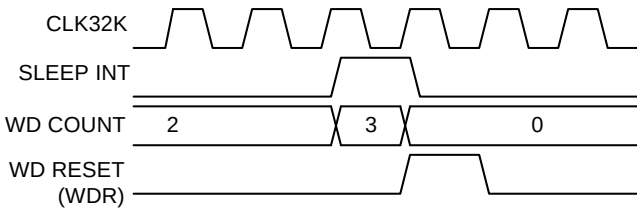
9.4.4 Watchdog Timer

On device boot up, the Watchdog Timer (WDT) is initially disabled. The PORS bit in the system control register controls the enabling of the WDT. On boot, the PORS bit is initially set to '1', indicating that either a POR or XRES event has occurred. The WDT is enabled by clearing the PORS bit. Once this bit is cleared and the watchdog timer is enabled, it cannot be subsequently disabled. (The PORS bit cannot be set to '1' in firmware; it can only be cleared.)

The only way to disable the Watchdog function, after it is enabled, is through a subsequent POR or XRES. Although the WDT is disabled during the first time through initialization code after a POR or XRES, all code should be written as if it is enabled (that is, the WDT should be cleared periodically). This is because, in the initialization code after a WDR event, the watchdog timer is enabled so all code must be aware of this.

The watchdog timer is three counts of the sleep timer interrupt output. The watchdog interval is three times the selected sleep timer interval. The available selections for the watchdog interval are shown in Table 9-1. When the sleep timer interrupt is asserted, the watchdog timer increments. When the counter reaches three, a terminal count is asserted. This terminal count is registered by the 32 kHz clock. Therefore, the WDR (Watchdog Reset) signal will go high after the following edge of the 32 kHz clock and be held asserted for one cycle (30 µs nominal). The ***flip-flop*** that registers the WDT terminal count is not reset by the WDR signal when it is asserted, but is reset by all other resets.

Figure 9-4. Watchdog Reset Timing



Once enabled, the WDT must be periodically cleared in firmware. This is accomplished with a write to the RES_WDT register. This write is data independent, so any write will clear the watchdog timer. (Note that a write of 38h will also clear the sleep timer.) If for any reason the firmware fails to clear the WDT within the selected interval, the circuit will assert WDR to the device. WDR is equivalent in effect to any other reset. All internal registers are set to their reset state, see the table titled "Details of Functionality for Various Resets" on page 112. An important aspect to remember about WDT resets is that RAM initialization can be disabled (IRAMDIS in the CPU_SCR1 register). In this case, the SRAM contents are unaffected; so that when a WDR occurs, program variables are persistent through this reset.

In practical application, it is important to know that the watchdog timer interval can be anywhere between two and three times the sleep timer interval. The only way to guarantee that the WDT interval is a full three times that of the sleep interval is to clear the sleep timer (write 38h) when clearing the WDT register. However, this is not possible in applications that use the sleep timer as a real-time clock. In the case where firmware clears the WDT register without clearing the sleep timer, this can occur at any point in a given sleep timer interval. If it occurs just before the terminal count of a sleep timer interval, the resulting WDT interval will be just over two times that of the sleep timer interval.

9.5 Power Modes

Sleep mode power consumption consists of the items in the following tables.

In Table 9-2, the typical block currents shown do not represent maximums. These currents do not include any analog block currents that may be on during Sleep mode.

Table 9-2. Continuous Currents

IPOR	1 µA
ICLK32K (ILO/ECO)	1 µA

While the CLK32K can be turned off in Sleep mode, this mode is not useful since it makes it impossible to restart unless an imprecise power on reset (IPOR) occurs. (The Sleep bit cannot be cleared without CLK32K.) During the sleep mode buzz, the bandgap is on for two cycles and the LVD circuitry is on for one cycle. Time-averaged currents from periodic sleep mode ‘buzz’, with periodic count of N, are listed in Table 9-3.

Table 9-3. Time Averaged Currents

IBG (Bandgap)	(2/N) * 60 µA
ILVD (LVD comparators)	(2/N) * 50 µA

Table 9-4 lists example currents for N=256 and N=1024. Device leakage currents add to the totals in the table.

Table 9-4. Example Currents

	N=256	N=1024
IPOR	1	1
CLK32K	1	1
IBG	0.46	0.12
ILVD	0.4	0.1
Total	2.9 µA	2.2 µA

Section C: CapSense System

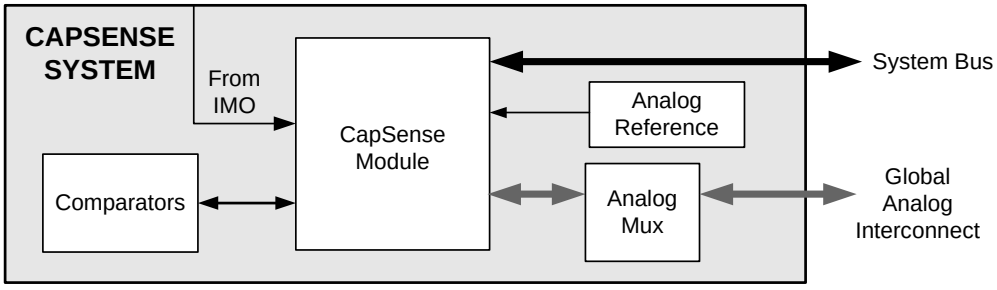
The configurable CapSense System section discusses the CapSense and analog components of the PSoC device and the registers associated with those components. This section encompasses the following chapters:

- CapSense Module on page 71.
- IO Analog Multiplexer on page 81.
- Comparators on page 83.

Top Level CapSense Architecture

The figure below illustrates the top level architecture of the PSoC's CapSense system. Each component of the figure is discussed in detail in this section.

PSoC CapSense System



CapSense Register Summary

This table lists all the PSoC registers for the CapSense system in address order within their system resource configuration. The bits that are grayed out are reserved bits. If these bits are written, always write them with a value of '0'.

Summary Table of the CapSense Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
CAPSENSE MODULE REGISTERS (page 71)										
0,A0h	CS_CR0	CSOUT[1:0]					MODE[1:0]		EN	RW : 00
0,A1h	CS_CR1	CHAIN	CLKSEL[1:0]		RLOSEL	INV	INSEL[2:0]			RW : 00
0,A2h	CS_CR2	IRANGE[1:0]		IDACDIR	IDAC_EN		PXD_EN		RO_EN	RW : 00
0,A3h	CS_CR3	IBOOST	REFMUX	REFMODE	REF_EN	LPFilt[1:0]		LPF_EN[1:0]		RW : 00
0,A4h	CS_CNTL	Data[7:0]								R : 00
0,A5h	CS_CNTH	Data[7:0]								R : 00
0,A6h	CS_STAT	INS	COLS	COHS	PPS	INM	COLM	COHM	PPM	# : 00
0,A7h	CS_TIMER			Timer Count Value[5:0]						RW : 00
0,A8h	CS_SLEW	FastSlew[6:0]							FS_EN	RW : 00
0,FDh	IDAC_D	IDACDATA[7:0]								RW : 00
IO ANALOG MULTIPLEXER REGISTERS (page 81)										
0,61h	AMUX_CFG					ICAPEN[1:0]		INTCAP[1:0]		RW : 00
1,D8h	MUX_CR0	ENABLE[7:0]								RW : 00
1,D9h	MUX_CR1	ENABLE[7:0]								RW : 00
1,DAh	MUX_CR2	ENABLE[7:0]								RW : 00
1,DBh	MUX_CR3	ENABLE[7:0]								RW : 00
COMPARATOR REGISTERS (page 83)										
0,78h	CMP_RDC			CMP1D	CMP0D			CMP1L	CMP0L	# : 00
0,79h	CMP_MUX	INP1[1:0]		INN1[1:0]		INP0[1:0]		INN0[1:0]		RW : 00
0,7Ah	CMP_CR0			CMP1R	CMP1EN			CMP0R	CMP0EN	RW : 00
0,7Bh	CMP_CR1	CINT1	CPIN1	CRST1	CDS1	CINT0	CPIN0	CRST0	CDS0	RW : 00
0,7Ch	CMP_LUT	LUT1[3:0]				LUT0[3:0]				RW : 00

LEGEND
Access is bit specific. Refer to the Register Reference chapter on page 137 for additional information.
R Read register or bit(s).
W Write register or bit(s).

10. CapSense Module

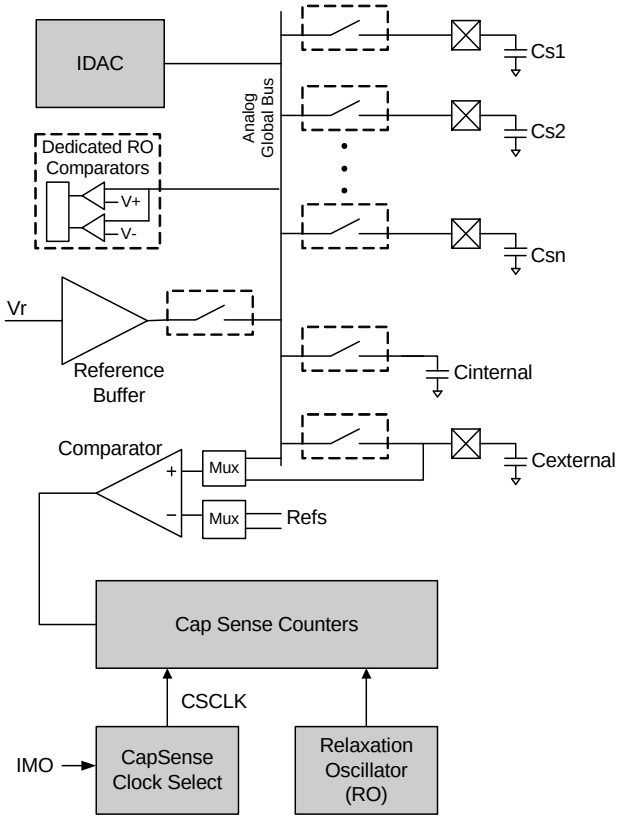
This chapter presents the CapSense™ Module and its associated registers. For a complete table of the CapSense Module registers, refer to the "Summary Table of the CapSense Registers" on page 70. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

10.1 Architectural Description

10.1.1 Types of CapSense Approaches

The CY8C20x34/24 PSoC device contains hardware support for a numbers of different capacitive sensing approaches. A block diagram of the overall capacitive sensing architecture is shown in Figure 10-1. Cs1 through Csn are the capacitors being measured. The various sensing approaches use different subsets of this hardware.

Figure 10-1. CapSense Module Block Diagram



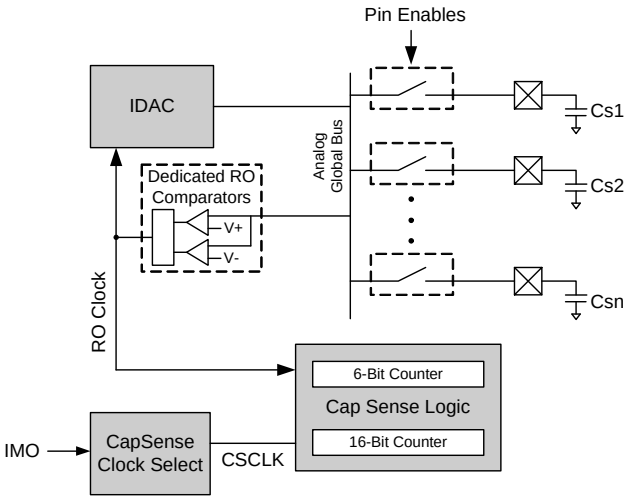
10.1.1.1 Relaxation Oscillator

The relaxation oscillator (RO) method operates by forming an oscillator using the sense capacitance. The IDAC, sense capacitance, and comparator (switching between two references) form the RO.

In the RO method the RO is compared to the frequency of an internal oscillator over a predetermined interval.

This interval is set by a number of cycles of the RO using a 6-bit counter. During this interval, the IMO clocks a 16-bit counter and the final count gives a measure of capacitance. (See Figure 10-2).

Figure 10-2. Relaxation Oscillator Block Diagram



On each (rising/falling) edge of the relaxation oscillator, an option allows the edge to operate with two different slew rates. A faster charging rate can be set for a fixed time, followed by a slower rate until the waveform reaches the target threshold voltage. This approach can lead to increased capacitance measurement sensitivity. The CS_SLEW register controls this mode.

10.1.1.2 IDAC

The internal current DAC provides a ***bias current*** for use with the relaxation oscillator (RO), or for capacitance measurement in the proximity detect mode. It can also be set to supply a sinking or sourcing current to any IO pin through the analog global bus connection.

The IDAC current is set by the 8-bit IDAC_D register. In addition, the two IRANGE bits in the CS_CR2 register provide additional prescaling range.

Note At very low currents a low frequency noise exists on the IDAC output. To avoid generating this low frequency noise, do not use IDAC_D settings equal to or less than three for capacitive sensing applications.

10.1.2 CapSense Counter

The CapSense Counter block (see Figure 10-3) is optimized to implement the relaxation oscillator algorithm. The hardware consists of two 8-bit up-counters with capture that can be optionally chained into a single 16-bit capture counter and an additional 6-bit timer.

In the relaxation algorithm, a 6-bit timer is clocked by the relaxation oscillator. A 16-bit chained counter is formed and clocked by CSCLK, a divided version of the internal main oscillator (IMO). In this configuration, the counters are enabled simultaneously with a write to the enable bit. On terminal count of the 6-bit RO counter, the contents of the 16-bit counter are captured. Changes in this count then indicate capacitance changes.

The CapSense Counter block is optimized to implement the relaxation oscillator algorithm. The hardware consists of two 8-bit counters capable of being chained into a 16-bit counter. For a clear understanding of this architecture see the block diagram in Figure 10-3.

Figure 10-3. CapSense Counter Block Diagram

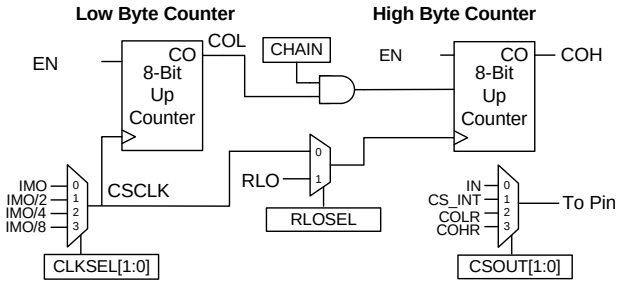
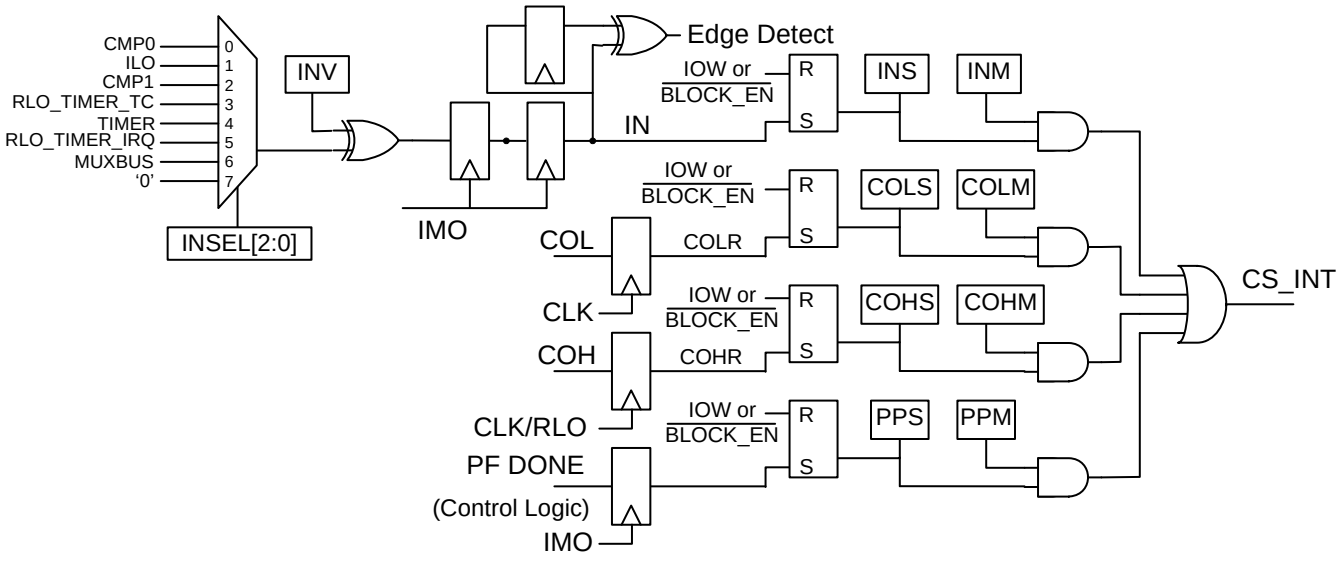


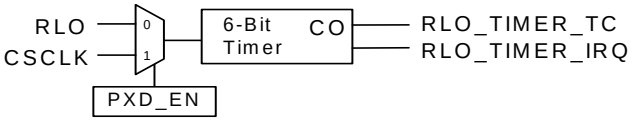
Figure 10-4 illustrates the variety of interrupt options for the block.

Figure 10-4. Block Interrupt Options Diagram



This diagram illustrates the 6-bit timer.

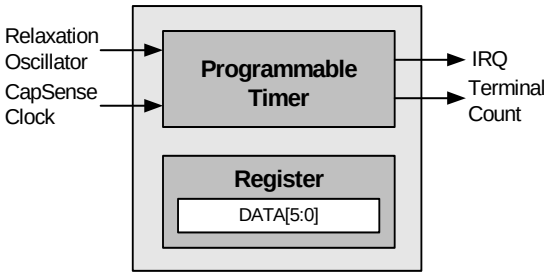
Figure 10-5. 6-Bit Timer Diagram



10.1.3 Timer

The programmable timer is a 6-bit down counter with a terminal count output. This timer has one data register associated with it. The timer is started when the CapSense block is enabled. The enable signal is double synchronized to the timer's clock domain. When started, the timer always starts counting down from the value loaded into its data registers (CS_TIMER). This timer only has a one shot mode, in which the timer completes one full count cycle and stops. Disabling and re-enabling the CapSense block restarts the timer.

Figure 10-6. RLO Timer Block Diagram



The timer's clock is either the RLO clock or the CapSense count clock, depending on the value of the PXD_EN bit in the CS_CR2 register. See the "CS_CR2 Register" on page 75 for details.

10.1.3.1 Operation

When started, the timer loads the value contained in its data register and counts down to its terminal count of zero. The timer outputs an active high terminal count pulse for one clock cycle upon reaching the terminal count. The low time of the terminal count pulse is equal to the loaded decimal count value multiplied by the clock period. (TC_{pw} = COUNT $VALUE_{decimal}$ * CLK_{period}). The period of the terminal count output is the pulse width of the terminal count plus one clock period. (TC_{period} = TC_{pw} + CLK_{period}). Refer to the timing diagram in Figure 10-11.

10.2 Register Definitions

The following registers are associated with the CapSense Module and are listed in address order. The register descriptions have an associated register table showing the bit structure for that register. The bits in the tables that are grayed out are reserved bits and are not detailed in the register descriptions that follow. Always write reserved bits with a value of '0'. For a complete table of CapSense Module registers, refer to the "Summary Table of the CapSense Registers" on page 70.

10.2.1 CS_CR0 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,A0h	CS_CR0	CSOUT[1:0]					MODE[1:0]		EN	RW : 00

The CapSense Control Register 0 (CS_CR0) controls the operation of the CapSense counters. Bits [7:1] should never be written to while the block is enabled.

Bits 7 and 6: CSOUT[1:0]. These bits select between a number of CapSense signals that can be driven to an output pin. Refer to Figure 10-3 on page 72 for the COL and COH, and to Figure 10-7 on page 79 for IN and CS_INT.

CSOUT[1:0]	Description
00	IN
01	CS_INT
10	COL
11	COH

Bits 2 and 1: MODE[1:0]. These bits specify the operating mode of the counter logic. The modes are shown in this table.

MODE[1:0]	Description
00	**Stop On Event** In this mode, the block starts counting when the EN bit is set, and stops counting on the selected interrupt event. This mode allows the user to read the counter results in firmware. Counting can be started again by disabling and re-enabling the block using the EN bit.
01	**Pulse Width** In this mode, after the EN bit is set, the block waits for a positive edge on the data input selection to start the counter, and then stops the counter on the following negative edge of the data input. Polarity can be adjusted with the INV bit (CS_CR1). Counting can be started again by disabling and re-enabling the block using the EN bit.
10	**Period** In this mode, after the EN bit is set, the block waits for a positive edge on the data input selection to start the counter, and then stops the counter on the following positive edge of the data input. Polarity can be adjusted with the INV bit (CS_CR1). Counting can be started again by disabling and re-enabling the block using the EN bit.
11	**Continuous** In this mode, the counter can be used to generate a periodic interrupt. The period is set by the input clock selection in conjunction with using one 8-bit counter (period=100h) or the chained 16-bit counter (period = 10000h).

Bit 0: EN. When this bit is written to '1', the counters are enabled for counting. When this bit is written to '0', counting is stopped and all counter values are reset to '0'. If the counting mode is stopped in conjunction with an event (see MODE[1:0]), the current count is held and can be subsequently read from the counter registers. The EN bit must be toggled to '0' and then back to '1' to start a new count.

For additional information, refer to the CS_CR0 register on page 150.

10.2.2 CS_CR1 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,A1h	CS_CR1	CHAIN	CLKSEL[1:0]		RLOSEL	INV	INSEL[2:0]			RW : 00

The CapSense Control Register 1 (CS_CR1) contains additional CapSense system control options. This register should never be written to while the block is enabled.

Bit 7: CHAIN. When this bit is a ‘0’, the two 8-bit counters operate independently. When this bit is a ‘1’, the counters are chained to operate as a 16-bit counter.

Bits 6 and 5: CLKSEL[1:0]. These bits select the CapSense module frequency of operation according to the information in the following table.

CLKSEL[1:0]	Frequency of Operation
00	IMO
01	IMO/2
10	IMO/4
11	IMO/8

Bit 4: RLOSEL. When this bit is a ‘0’, the entire CapSense system runs at the frequency specified in the CLKSEL[1:0] bits. When this bit is a ‘1’, the High Counter is clocked independently by the CapSense RLO clock.

Bit 3: INV. Input Invert. When this bit is a ‘1’, the data input select is inverted. When this bit is a ‘0’, the input polarity is unchanged.

Bits 2 to 0: INSEL[2:0]. Input Selection. These bits control the selection of input signals for event control according to the information in the following table.

INSEL[1:0]	Selected Input
000	Comparator 0
001	ILO
010	Comparator 1
011	RLO Timer Terminal Count
100	Internal Timer
101	RLO Timer IRQ
110	Analog Global Mux Bus
111	‘0’

For additional information, refer to the CS_CR1 register on page 151.

10.2.3 CS_CR2 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,A2h	CS_CR2	IRANGE[1:0]		IDACDIR	IDAC_EN		PXD_EN		RO_EN	RW : 00

The CapSense Control Register 2 (CS_CR2) contains additional CapSense system control options.

Bits 7 and 6: IRANGE. These bits scale the IDAC current output.

	Frequency of Operation
00	1X range
01	2X range
10	4X range
11	8X range

Bit 5: IDACDIR. This bit determines whether the IDAC sinks or sources current to the analog global bus when enabled.

	Frequency of Operation
0	IDAC Sources
1	IDAC Sink

Bit 4: IDAC_EN. This bit enables manual connection of the IDAC to the analog global bus.

Bit 2: PXD_EN. This bit drives a clock to each IO pin that is enabled for connection to the analog global bus. This clock alternately connects the pin to the bus, then connects the pin to ground. The clock rate is selected by the CLKSEL bits in the CS_CR1 register. In addition, the IDAC sources current to the bus. The programmable timer is clocked by this same clock.

Bit 0: RO_EN. This bit enables the relaxation oscillator. The internal RO is connected to the analog global bus, and the capacitance of any connected pins will affect the RO frequency. The oscillator current is set by the value of the IDAC_D register.

For additional information, refer to the CS_CR2 register on page 152.

10.2.4 CS_CR3 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,A3h	CS_CR3	IBOOST	REFMUX	REFMODE	REF_EN	LPFilt[1:0]		LPF_EN[1:0]		RW : 00

The CapSense Control Register 3 (CS_CR3) contains control bits primarily for the low pass filter and reference buffer.

Bit 7: IBOOST. This bit adds an offset current to all IDAC settings, so a zero value in the IDAC_D register does not give zero current out. This affects all IDAC functions, including the relaxation oscillator.

Bit 6: REFMUX. This bit selects between VREF and REFHI for the reference buffer input.

Bit 5: REFMODE. This bit is used for manual connection of the reference buffer to the analog global bus. If either the CI_EN or RO_EN bits are set high in the CS_CR2 register, this bit has no effect.

Bit 4: REF_EN. This bit enables the reference buffer to drive to the analog global bus.

Bits 3 and 2: LPFilt[1:0]. These bits control the time constant of the low pass filter that connects to the analog bus.

LPFilt[1:0]	Frequency of Operation
00	1 ms
01	2 ms
10	5 ms
11	10 ms

Bits 1 and 0: LPF_EN[1:0]. These bits are used to connect a low pass filter into the input of either comparator channel.

For additional information, refer to the CS_CR3 register on page 153.

10.2.5 CS_CNTL Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,A4h	CS_CNTL	Data[7:0]								RO : 00

The CapSense Counter Low Byte Register (CS_CNTL) contains the current count for the low byte counter.

Bits 7 to 0: Data[7:0]. This value contains the current count for the counter low block. The block must be stopped to read a valid value.

For additional information, refer to the CS_CNTL register on page 154.

10.2.6 CS_CNTH Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,A5h	CS_CNTH	Data[7:0]								RO : 00

The CapSense Counter High Byte Register (CS_CNTH) contains the current count value for the high byte counter.

Bits 7 to 0: Data[7:0]. This value contains the current count for the counter high block. The block must be stopped to read a valid value.

For additional information, refer to the CS_CNTH register on page 155.

10.2.7 CS_STAT Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,A6h	CS_STAT	INS	COLS	COHS	PPS	INM	COLM	COHM	PPM	# : 00

LEGEND
Access is bit specific.

The CapSense Status Register (CS_STAT) controls CapSense counter options.

Status Bits 7 to 4 – The posted CapSense interrupts are the corresponding status bits in this register. Interrupt clearing is performed by clearing the associated status bit. Status can only be updated while the block is enabled and running. All status bits are cleared when the block is disabled.

Bit 7: INS. Input Status. Reading a ‘1’ indicates a rising edge on the selected input was detected. Reading a ‘0’ indicates that this event did not occur. This bit is cleared by writing a ‘0’ to this bit position. Writing a ‘1’ has no effect.

Bit 6: COLS. Counter Carry Out Low Status. Reading a ‘1’ indicates an overflow occurred in the Counter Low block. Reading a ‘0’ indicates that this event did not occur. This bit is cleared by writing a ‘0’ to this bit position. Writing a ‘1’ has no effect.

Bit 5: COHS. Counter Carry Out High Status. Reading a ‘1’ indicates an overflow occurred in the Counter High block. Reading a ‘0’ indicates that this event did not occur. This bit is cleared by writing a ‘0’ to this bit position. Writing a ‘1’ has no effect.

Bit 4: PPS. Pulse Width/Period Status. Reading a ‘1’ indicates the completion of a pulse width or period measurement (as defined by the MODE[1:0] bits in CS_CR0). This bit is cleared by writing a ‘0’ to this bit position. Writing a ‘1’ has no effect.

Mask Bits 3 to 0 – The interrupt mask bits should never be modified while the block is enabled. If modification to bits 3 to 0 is necessary while the block is enabled, then special attention must be paid to ensure that the status bits, bits 7 to 4, are not accidentally cleared. This can be done by writing a ‘1’ to all of the status bits when writing to the mask bits.

Bit 3: INM. Input Interrupt Mask. When this bit is a ‘1’, a rising edge event on the input will assert the block interrupt. When this bit is a ‘0’, this event is masked.

Bit 2: COLM. Counter Carry Out Low Mask. When this bit is a ‘1’, a carry out from the counter low block will assert the block interrupt. When this bit is a ‘0’, this event is masked.

Bit 1: COHM. Counter Carry Out High Mask. When this bit is a ‘1’, a carry out from the counter high block will assert the block interrupt. When this bit is a ‘0’, this event is masked.

Bit 0: PPM. Pulse Width/Period Mask When this bit is a ‘1’, the completion of a pulse width or period measurement will assert the block interrupt. When this bit is a ‘0’, this event is masked.

For additional information, refer to the CS_STAT register on page 156.

10.2.8 CS_TIMER Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,A7h	CS_TIMER			Timer Count Value[5:0]						RW : 00

The CapSense Timer Register (CS_TIMER) sets the timer count value.

Bits 5 to 0: Timer Count Value[5:0]. The 6-bit value in this register sets the initial count value for the timer.

For additional information, refer to the CS_TIMER register on page 157.

10.2.9 CS_SLEW Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,A8h	CS_SLEW	FastSlew[6:0]							FS_EN	RW : 00

The CapSense Slew Control Register (CS_SLEW) enables and controls a fast slewing mode for the relaxation oscillator.

Bits 7 to 1: FastSlew[6:0]. This 7-bit count sets the time interval, in IMO cycles, for a faster slew rate on the relaxation oscillator edges. The interval applies to both rising and falling edges. This timer value has no effect unless the FS_EN bit is set high.

Bit 0: FS_EN. This bit enables the fast slewing interval on each edge of the relaxation oscillator.

For additional information, refer to the CS_SLEW register on page 158.

10.2.10 IDAC_D Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,FDh	IDAC_D	IDACDATA[7:0]								RW : 00

The Current DAC Data Register (IDAC_D) specifies the 8-bit multiplying factor that determines the output DAC current.

Bits 7 to 0: IDACDATA[7:0]. The 8-bit value in this register sets the current driven onto the analog global mux bus when the current DAC mode is enabled.

Note At very low currents a low frequency noise exists on the IDAC output. To avoid generating this low frequency noise, do not use IDAC_D settings equal to or less than three for capacitive sensing applications.

For additional information, refer to the IDAC_D register on page 177.

10.3 Timing Diagrams

Figure 10-7. Event Timing (Mode = 00)

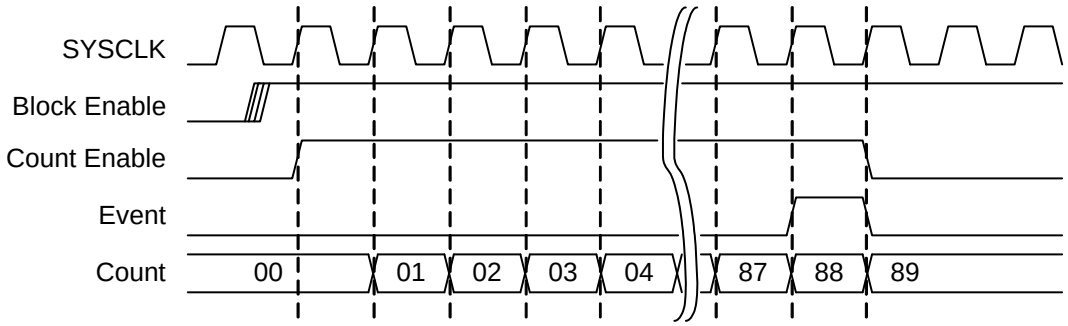


Figure 10-8. Pulse Width Frequency Timing (Mode = 01/10)

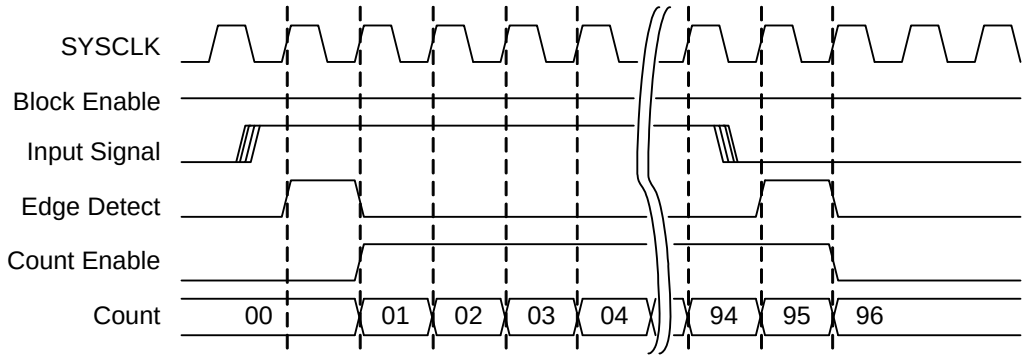


Figure 10-9. Continuous Timing (Mode = 11)

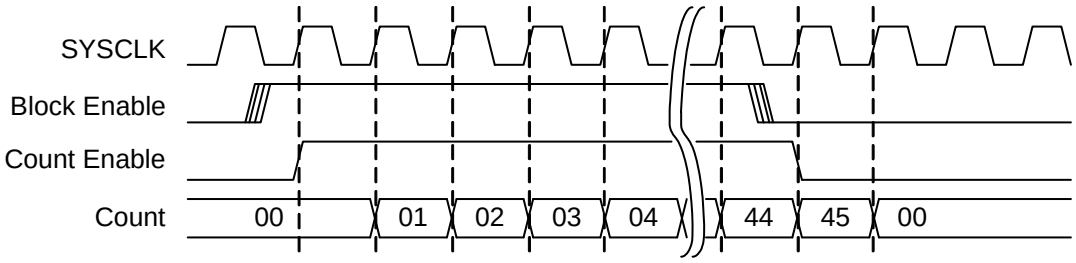


Figure 10-10. High Byte Counter Timing (RLO clock selected)

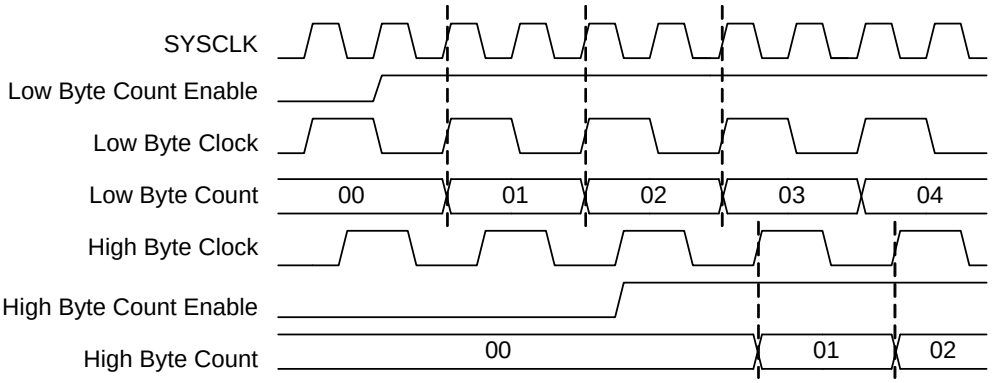
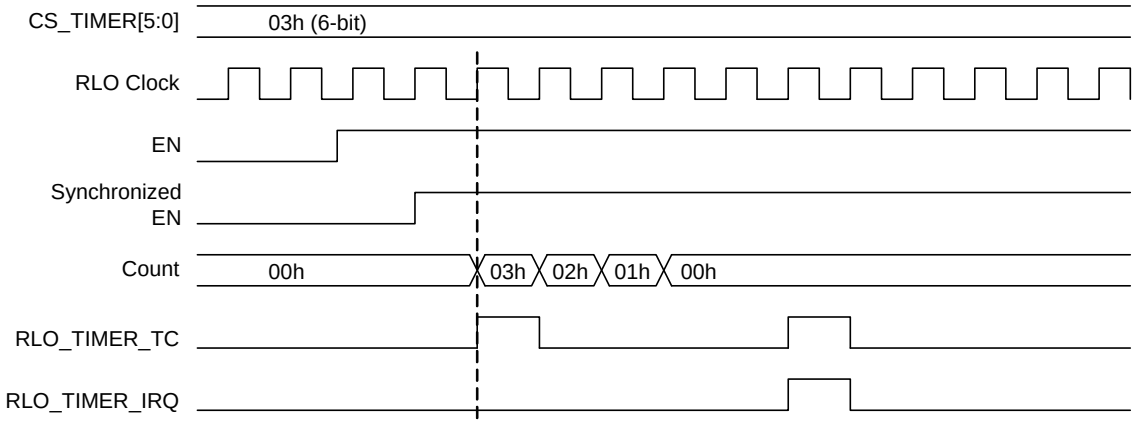


Figure 10-11. 6-Bit RLO Timer Operation



11. IO Analog Multiplexer

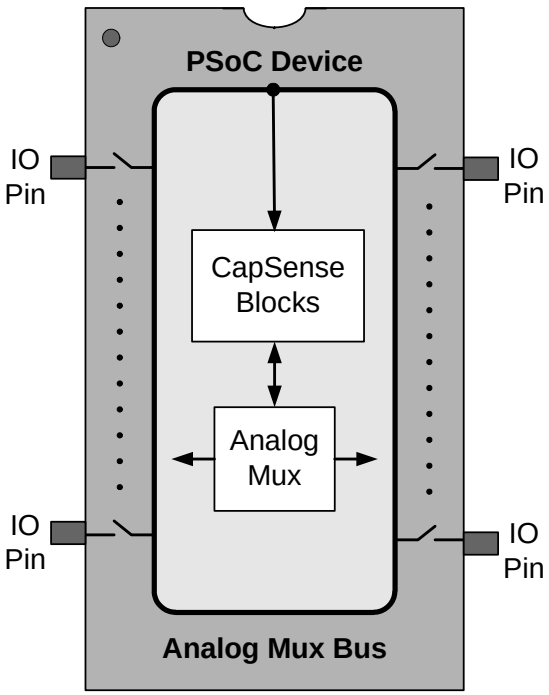
This chapter explains the chip-wide IO Analog Multiplexer for the CY8C20x34/24 PSoC device and its associated registers. For a complete table of the IO Analog Multiplexer registers, refer to the "Summary Table of the CapSense Registers" on page 70. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

11.1 Architectural Description

The CY8C20x34/24 PSoC device contains an enhanced analog multiplexer (mux) capability. This function allows many IO pins to connect to a common internal analog global bus.

Any number of pins can be connected simultaneously, and dedicated support circuitry allows selected pins to be alternately charged high or connected to the bus. The analog global bus can be connected as a comparator input. Figure 11-1 shows a block diagram of the IO analog mux system.

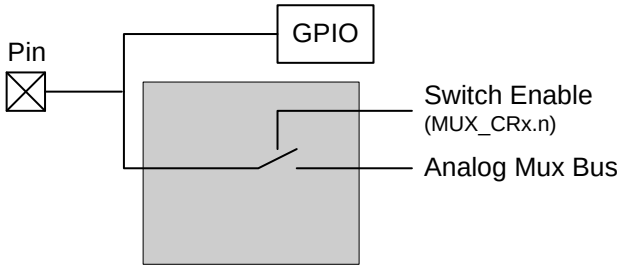
Figure 11-1. IO Analog Mux System



For each pin, the mux capability exists in parallel with the normal GPIO cell, shown in Figure 11-2. Normally, the associated GPIO pin is put into a high-impedance state for these applications, although there are cases where the GPIO cell is configured by the user to briefly drive pin initialization states as described below.

Pins are individually connected to the internal bus by setting the corresponding bits in the MUX_CRx registers. Any number of pins can be enabled at the same time. At reset, all of these mux connections are open (disconnected).

Figure 11-2. IO Pin Configuration for the CY8C20x34/24



11.2 Application Overview

The analog mux circuitry enables capacitive sensing and internal capacitance.

Capacitive Sensing. The analog mux supports capacitive sensing applications through the use of the IO analog multiplexer and its control circuitry. Refer to the "Architectural Description" on page 71 in the CapSense Module chapter for more details.

Internal Capacitance. An internal filter capacitance can be connected to the analog global bus, using the ICAPEN bits in the AMUX_CFG register.

11.3 Register Definitions

The following registers are only associated with the Analog Bus Mux in the CY8C20x34/24 PSoC device and are listed in address order. For a complete table of the IO Analog Multiplexer registers, refer to the “Summary Table of the CapSense Registers” on page 70. Each register description has an associated register table showing the bit structure for that register. Register bits that are grayed out throughout this document are reserved bits and are not detailed in the register descriptions that follow.Always write reserved bits with a value of ‘0’.

11.3.1 AMUX_CFG Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,61h	AMUX_CFG					ICAPEN[1:0]		INTCAP[1:0]		RW : 00

The Analog Mux Configuration Register (AMUX_CFG) is used to configure the integration capacitor pin connections to the analog global bus.

Bits 3 and 2: ICAPEN[1:0]. Setting these bits connect an internal capacitor (up to approximately 100 pF) to the analog global bus.

Bits 1 and 0: INTCAP[1:0]. These bits select between the P0[1] and P0[3] pins for the integration capacitor for charge integration capacitive sensing.

For additional information, refer to the AMUX_CFG register on page 143.

11.3.2 MUX_CRx Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,D8h	MUX_CR0	ENABLE[7:0]								RW : 00
1,D9h	MUX_CR1	ENABLE[7:0]								RW : 00
1,DAh	MUX_CR2	ENABLE[7:0]								RW : 00
1,DBh	MUX_CR3	ENABLE[7:0]								RW : 00

The Analog Mux Port Bit Enable Registers (MUX_CR0, MUX_CR1, MUX_CR2, and MUX_CR3) are used to control the connection between the analog mux bus and the corresponding pin.

Bits 7 to 0: ENABLE[7:0]. The bits in these registers enable connection of individual pins to the analog mux bus. Each IO port has a corresponding MUX_CRx register.

For additional information, refer to the MUX_CRx register on page 183.

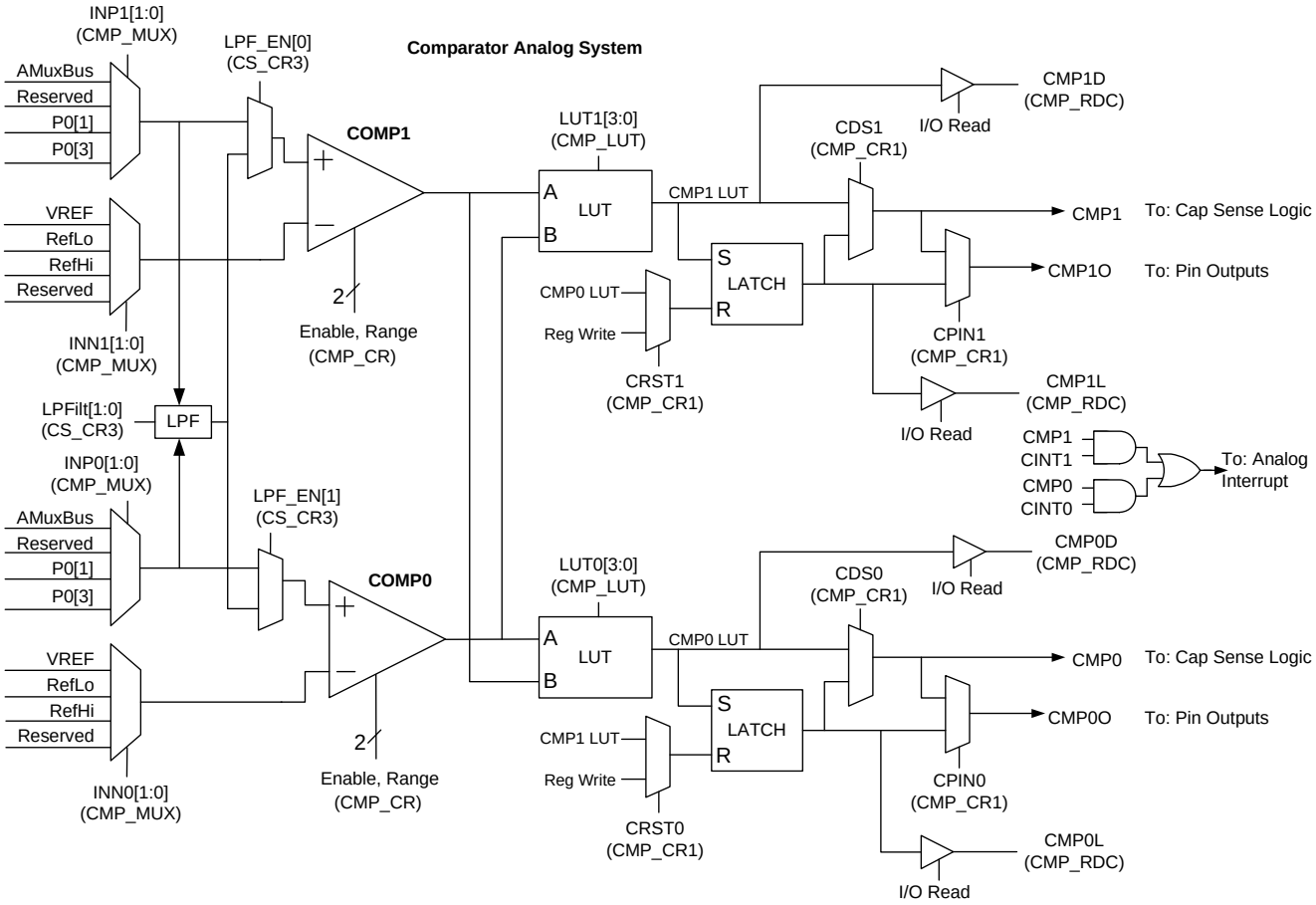
12. Comparators

This chapter explains the Comparators for the CY8C20x34/24 PSoC device and its associated registers. For a complete table of the comparator registers, refer to the "Summary Table of the CapSense Registers" on page 70. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

12.1 Architectural Description

The CY8C20x34/24 PSoC device contains two comparators designed to support capacitive sensing or other general purpose uses. Figure 12-1 shows a block diagram of the comparator system.

Figure 12-1. Comparators Block Diagram



The comparator digital interface performs logic processing on one or more comparator signals, provides a latching capability, and routes the result to other chip subsystems. The comparator signal is routed through a look-up table (LUT) function. The other input to the LUT is the neighboring comparator output. The LUT implements 1 of 16 functions on the two inputs, as selected by the CMP_LUT register. The LUT output also feeds the set input on an reset/set (RS) latch. The latch is cleared by writing a '0' to the appropriate bit in the CMP_RDC register, or by a rising edge from the other comparator LUT.

The primary output for each comparator is the LUT output or its latched version. These are routed to the CapSense logic and to the interrupt controller. The comparator LUT output state and latched state may be directly read by the CPU through the CMP_RDC register. A selection of comparator state may also be driven to an output pin.

When disabled, the comparators consume no power. Two active modes provide a full rail-to-rail input range, or a somewhat lower power option with limited input range.

12.2 Register Definitions

The following registers are only associated with the Comparators in the CY8C20x34/24 PSoC device and are listed in address order. For a complete table of the comparator registers, refer to the "Summary Table of the CapSense Registers" on page 70. Each register description has an associated register table showing the bit structure for that register. Register bits that are grayed out throughout this document are reserved bits and are not detailed in the register descriptions that follow. Always write reserved bits with a value of '0'.

12.2.1 CMP_RDC Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,78h	CMP_RDC			CMP1D	CMP0D			CMP1L	CMP0L	# : 00

LEGEND
Access is bit specific.

The Comparator Read/Clear Register (CMP_RDC) is used to read the state of the comparator data signal and the latched state of the comparator.

Bit 5: CMP1D. Comparator 1 Data State. This bit is a read-only bit and returns the dynamically changing state of the comparator.

Bit 4: CMP0D. Comparator 0 Data State. This bit is a read-only bit and returns the dynamically changing state of the comparator.

Bit 1: CMP1L. Comparator 1 Latched State. This bit is set and held high whenever the comparator 1 LUT goes high since the last time this register was read. Refer to the CRST1 bit in the CMP_CR1 register for information on how the latch is cleared.

Bit 0: CMP0L. Comparator 0 Latched State. This bit is set and held high whenever the comparator 0 LUT goes high since the last time this register was read. Refer to the CRST0 bit in the CMP_CR1 register for information on how the latch is cleared.

For additional information, refer to the CMP_RDC register on page 144.

12.2.2 CMP_MUX Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,79h	CMP_MUX	INP1[1:0]		INN1[1:0]		INP0[1:0]		INN0[1:0]		RW : 00

The Comparator Multiplexer Register (CMP_MUX) contains control bits for input selection of comparators 0 and 1.

Bits 7 and 6: INP1[1:0]. These bits select the positive input data source for comparator 1. The selections are shown in the table.

Bits 5 and 4: INN1[1:0]. These bits select the negative input data source for comparator 1. The selections are shown in the table.

Bits 3 and 2: INP0[1:0]. These bits select the positive input data source for comparator 0. The selections are shown in the table.

Bits 1 and 0: INN0[1:0]. These bits select the negative input data source for comparator 0. The selections are shown in the table below.

INPx[1:0]		INNx[1:0]	
00	Analog Global Bus (Mix-ups	00	1.3V Reference
01	Reserved	01	RefLo Reference
10	P0[1] pad	10	RefHi Reference
11	P0[3] pad	11	Reserved

For additional information, refer to the CMP_MUX register on page 145.

12.2.3 CMP_CR0 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,7Ah	CMP_CR0			CMP1R	CMP1EN			CMP0R	CMP0EN	RW : 00

The Comparator Control Register 0 (CMP_CR0) is used to enable and configure the input range of the comparators.

Bit 5: CMP1R. This bit selects the input range for comparator 1. Setting the bit high selects a somewhat lower power mode that does not operate rail-to-rail.

Bit 4: CMP1EN. This bit enables comparator 1.

Bit 1: CMP0R. This bit selects the input range for comparator 1. Setting the bit high selects a somewhat lower power mode that does not operate rail-to-rail.

Bit 0: CMP0EN. This bit enables comparator 0.

For additional information, refer to the CMP_CR0 register on page 146.

12.2.4 CMP_CR1 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,7Bh	CMP_CR1	CINT1	CPIN1	CRST1	CDS1	CINT0	CPIN0	CRST0	CDS0	RW : 00

The Comparator Control Register 1 (CMP_CR1) is used to configure the comparator output options.

Bit 7: CINT1. This bit connects the comparator 1 output to the analog output.

Bit 6: CPIN1. This bit selects whether the comparator 1 LUT output or the latched output can be routed to a GPIO pin.

Bit 5: CRST1. This bit selects whether the comparator 1 latch is reset on register write or by a rising edge from the comparator 0 LUT output.

Bit 4: CDS1. This bit selects between the comparator 1 LUT and the latched output, for the main comparator output, that drives to the capacitive sense and interrupt logic.

Bit 3: CINT0. This bit connects the comparator 0 output to the analog output.

Bit 2: CPIN0. This bit selects whether the comparator 0 LUT output or the latched output can be routed to a GPIO pin.

Bit 1: CRST0. This bit selects whether the comparator 0 latch is reset on register write or by a rising edge from the comparator 1 LUT output.

Bit 0: CDS0. This bit selects between the comparator 0 LUT and the latched output, for the main comparator output, that drives to the capacitive sense and interrupt logic.

For additional information, refer to the CMP_CR1 register on page 147.

12.2.5 CMP_LUT Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,7Ch	CMP_LUT	LUT1[3:0]				LUT0[3:0]				RW : 00

The Comparator LUT Control Register (CMP_LUT) is used to select the logic function.

Bits 7 to 4: LUT1[3:0]. These bits control the selection of the LUT 1 logic functions that may be selected for the comparator channel 1.

Bits 3 to 0: LUT0[3:0]. These bits control the selection of LUT 0 logic functions that may be selected for the comparator channel 0.

CLUTx[3:0]	0h: 0000: FALSE 1h: 0001: A .AND. $\overline{B}$ 2h: 0010: A .AND. $\overline{B}$ 3h: 0011: $\overline{A}$ 4h: 0100: $\overline{A}$ .AND. B 5h: 0101: B 6h: 0110: A .XOR. B 7h: 0111: A .OR. B 8h: 1000: A .NOR. B 9h: 1001: $\overline{A}$ .XNOR. B Ah: 1010: $\overline{B}$ Bh: 1011: $\overline{A}$ .OR. $\overline{B}$ Ch: 1100: $\overline{A}$ Dh: 1101: $\overline{A}$ .OR. B Eh: 1110: A. NAND. B Fh: 1111: TRUE

For additional information, refer to the CMP_LUT register on page 149.

Section D: System Resources

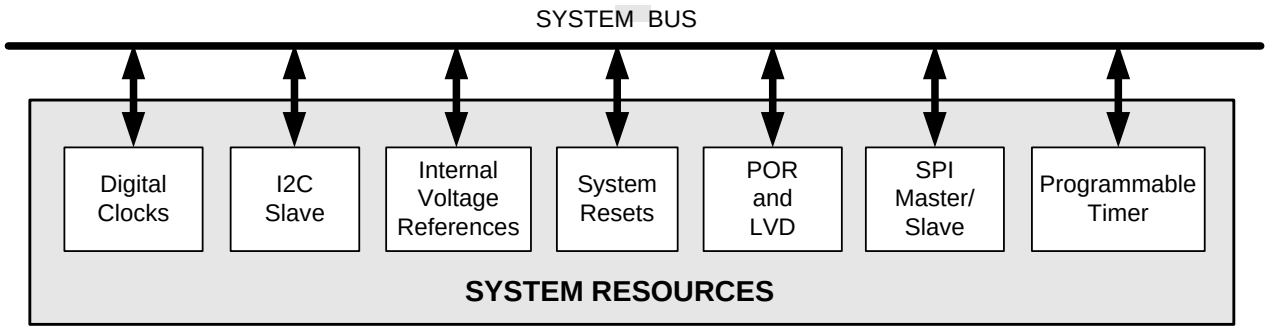
The System Resources section discusses the system resources that are available for the PSoC device and the registers associated with those resources. This section encompasses the following chapters:

- Digital Clocks on page 89.
- I2C Slave on page 95.
- Internal Voltage References on page 105.
- System Resets on page 107.
- POR and LVD on page 113.
- Serial Peripheral Interface on page 115.
- Programmable Timer on page 129.

Top Level System Resources Architecture

The figure below illustrates the top level architecture of the PSoC's system resources. Each component of the figure is discussed in detail in this section.

PSoC System Resources



System Resources Register Summary

The table below lists all the PSoC registers for the system resources, in address order, within their system resource configuration. The bits that are grayed out are reserved bits. If these bits are written, they should always be written with a value of '0'.

Summary Table of the System Resource Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
DIGITAL CLOCK REGISTERS (page 89)										
1,DDh	OUT_P1	P16D	P16EN	P14D	P14EN	P12D	P12EN	P10D	P10EN	RW : 00
1,E0h	OSC_CR0		Disable Buzz	No Buzz	Sleep[1:0]		CPU Speed[2:0]			RW : 01
1,E2h	OSC_CR2						EXTCLKEN	IMODIS		RW : 00
I2C SLAVE REGISTERS (page 95)										
0,D6h	I2C_CFG		PSelect		Stop IE	Clock Rate[1:0]			Enable	RW : 00
0,D7h	I2C_SCR	Bus Error		Stop Status	ACK	Address	Transmit	LRB	Byte Complete	R : 00
0,D8h	I2C_DR	Data[7:0]								RW : 00
INTERNAL VOLTAGE REFERENCES REGISTER (page 106)										
1,EAh	BDG_TR	TC[2:0]			V[4:0]					RW : 50
SYSTEM RESET REGISTERS (page 107)										
0,FEh	CPU_SCR1	IRESS			SLIMO				IRAMDIS	# : 00
0,FFh	CPU_SCR0	GIES		WDRS	PORS	Sleep			STOP	# : XX
POR REGISTERS (page 113)										
1,E3h	VLT_CR			PORLEV[1:0]		LVDTBEN	VM[2:0]			RW : 00
1,E4h	VLT_CMP					NoWrite		LVD	PPOR	R : 00
SPI REGISTERS (page 115)										
0,29h	SPI_TXR	Data[7:0]								W : 00
0,2Ah	SPI_RXR	Data[7:0]								R : 00
0,2Bh	SPI_CR	LSb First	Overrun	SPI Complete	TX Reg Empty	RX Reg Full	Clock Phase	Clock Polarity	Enable	# : 00
1,29h	SPI_CFG	Clock Sel[2:0]			Bypass	SS_	SS_EN_	Int Sel	Slave	RW : 00
PROGRAMMABLE TIMER REGISTERS (page 129)										
0,B0h	PT_CFG							One Shot	START	RW : 00
0,B1h	PT_DATA1				Data[4:0]					RW : 00
0,B2h	PT_DATA0	Data[7:0]								RW : 00

LEGEND
X The value after power on reset is unknown.
R Read register or bit(s).
W Write register or bit(s).
Access is bit specific. Refer to the Register Details chapter for additional information.

13. Digital Clocks

This chapter discusses the Digital Clocks and their associated registers. It serves as an overview of the clocking options available in the PSoC devices. For detailed information on specific oscillators, see the individual oscillator chapters in the section called "PSoC Core" on page 23. For a complete table of the digital clock registers, refer to the "Summary Table of the System Resource Registers" on page 88. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

13.1 Architectural Description

The PSoC M8C core has a large number of clock sources that increase the flexibility of the PSoC mixed-signal array, as listed in Table 13-1 and illustrated in Figure 13-1.

Table 13-1. System Clocking Signals and Definitions

Signal	Definition
SYSCLK	Either the direct output of the Internal Main Oscillator or the direct input of the EXTCLK pin while in external clocking mode.
CPUCLK	SYSCLK is divided down to one of eight possible frequencies, to create CPUCLK which determines the speed of the M8C. See OSC_CR0 in the Register Definitions section of this chapter.
CLK32K	The Internal Low Speed Oscillators output. See OSC_CR0 in the Register Definitions section of this chapter.
CLK12M	The internally generated 12 MHz clock by the IMO. By default, this clock drives SYSCLK; however, an external clock may be used by enabling EXTCLK mode. Also, the IMO may be put into a slow mode using the SLIMO bit which will change the speed of the IMO and the CLK24M to 6 MHz or 12 MHz.
SLEEP	One of four sleep intervals may be selected from 1.95 ms to 1 second. See OSC_CR0 in the Register Definitions section of this chapter.

13.1.1 Internal Main Oscillator

The Internal Main Oscillator (IMO) is the foundation upon which almost all other clock sources in the PSoC mixed-signal array are based. The default mode of the IMO creates a 12 MHz reference clock that is used by many other circuits in the PSoC device. The PSoC device has an option to replace the IMO with an externally supplied clock that will become the base for all of the clocks the IMO normally serves. The internal base clock net is called SYSCLK and may be driven by either the IMO or an external clock (EXTCLK).

Whether the external clock or the internal main oscillator is selected, all PSoC device functions are clocked from a derivative of SYSCLK or are resynchronized to SYSCLK. All external asynchronous signals, as well as the internal low speed oscillator, are resynchronized to SYSCLK for use in the digital PSoC blocks.

Some PSoC devices contain the option to lower the internal oscillator's system clock from 12 MHz to 6 MHz. See the "Architectural Description" on page 57, in the Internal Main Oscillator chapter, for more information.

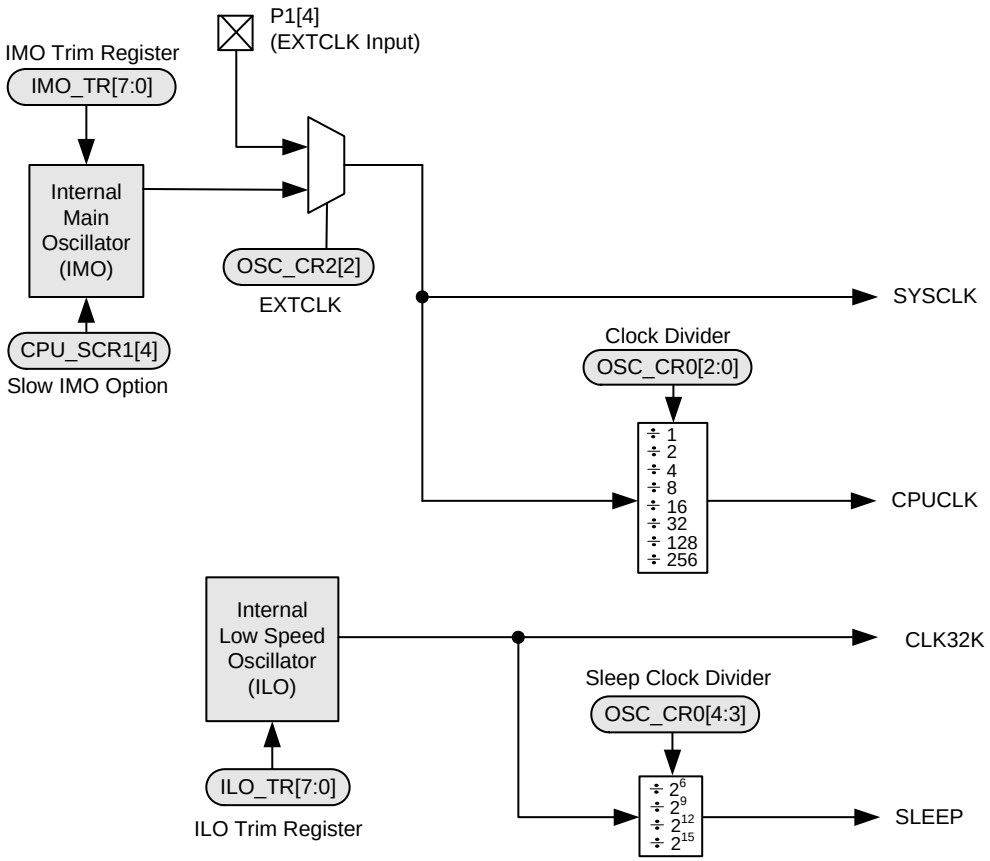
The IMO is discussed in detail in the chapter "Internal Main Oscillator (IMO)" on page 57.

13.1.2 Internal Low Speed Oscillator

The Internal Low Speed Oscillator (ILO) is always on. The ILO is available as a general clock, but is also the clock source for the sleep and watchdog timers.

The ILO is discussed in detail in the chapter "Internal Low Speed Oscillator (ILO)" on page 59.

Figure 13-1. Overview of PSoC Clock Sources



13.1.3 External Clock

The ability to replace the 12 MHz internal main oscillator (IMO), as the device master system clock (SYSCLK) with an externally supplied clock, is a feature in the PSoC mixed-signal array (see Figure 13-1).

Pin P1[4] is the input pin for the external clock. If P1[4] is selected as the external clock source, the drive mode of the pin must be set to High-Z (not High-Z analog).

An external clock with a frequency between 1 MHz and 12 MHz can be supplied. The reset state of the EXTCLKEN bit is ‘0’; therefore, the device always boots up under the control of the IMO. There is no way to start the system from a reset state with the external clock.

When the EXTCLKEN bit is set, the external clock becomes the source for the internal clock tree, SYSCLK, which drives most PSoC device clocking functions. All external and internal signals, including the 1 kHz clock, are synchronized to this clock source.

13.1.3.1 Switch Operation

Switching between the IMO and the external clock may be done in firmware at any time and is transparent to the user. Since all PSoC device resources run on clocks derived from or synchronized to SYSCLK, when the switch is made, analog and digital functions may be momentarily interrupted.

When a switch is made from the IMO to the external clock, the IMO may be turned off to save power. This is done by setting the IMODIS bit and may be done immediately after the instruction that sets the EXTCLKEN bit. However, the IMO must not be disabled if the external clock is slower than 6 MHz. When switching back from an external clock to the IMO, the IMODIS bit must be cleared and a firmware delay implemented. This gives the IMO sufficient start-up time before the EXTCLKEN bit is cleared.

Switch timing depends on whether the CPU clock divider is set for divide by 1, or divide by 2 or greater. In the case where the CPU clock divider is set for divide by 2 or greater, as shown in Figure 13-2, the setting of the EXTCLKEN bit occurs shortly after the rising edge of SYSCLK. The SYSCLK output is then disabled after the next falling edge of SYSCLK, but before the next rising edge. This ensures a glitch-free transition and provides a full cycle of setup time from SYSCLK to output disable. Once the current clock selection is disabled, the enable of the newly selected clock is double synchronized to that clock. After synchronization, on the subsequent negative edge, SYSCLK is enabled to output the newly selected clock.

In the 12 MHz case, as shown in Figure 13-3, the assertion of IOW_ and thus the setting of the EXTCLKEN bit occurs on the falling edge of SYSCLK. Since SYSCLK is already low, the output is immediately disabled. Therefore, the setup time from SYSCLK to disable is one half SYSCLK.

Figure 13-2. Switch from IMO to the External Clock with a CPU Clock Divider of Two or Greater

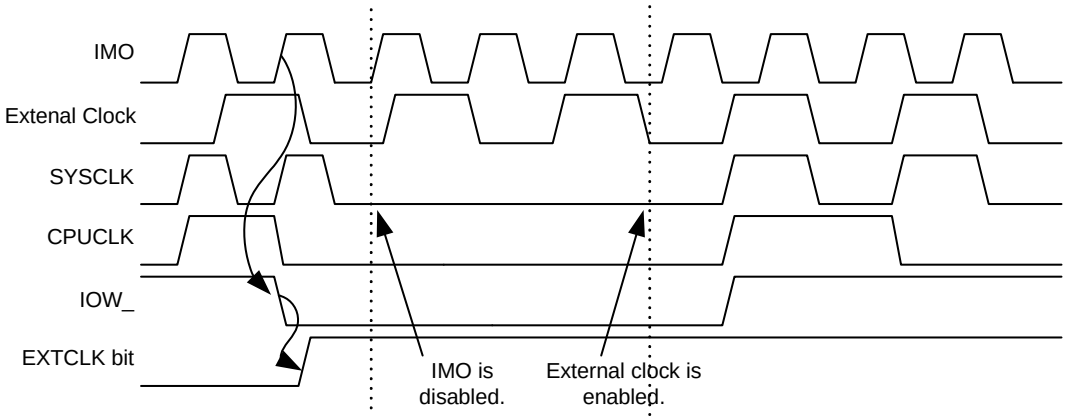
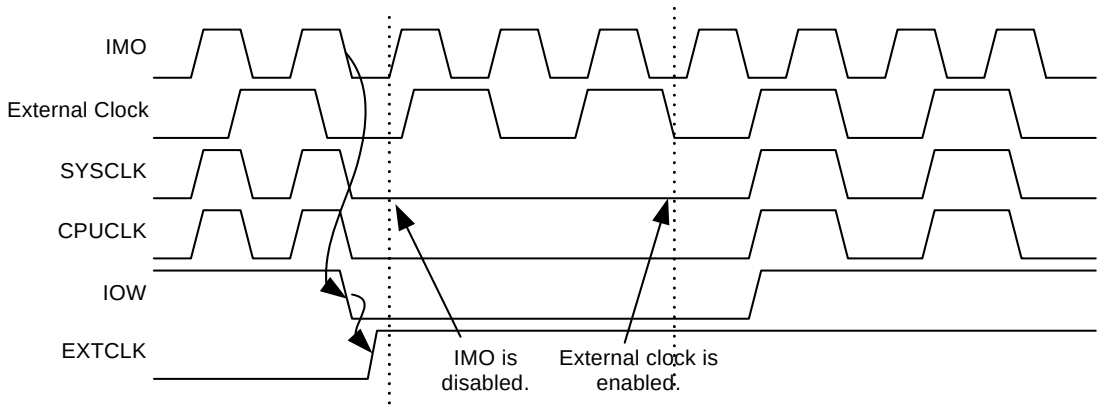


Figure 13-3. Switch from IMO to External Clock with the CPU Running with a CPU Clock Divider of One



13.2 Register Definitions

The following registers are associated with the Digital Clocks and are listed in address order. Each register description has an associated register table showing the bit structure for that register. The bits in the tables that are grayed out throughout this manual are reserved bits and are not detailed in the register descriptions that follow. Always write reserved bits with a value of '0'. For a complete table of digital clock registers, refer to the "Summary Table of the System Resource Registers" on page 88.

13.2.1 OUT_P1 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,DDh	OUT_P1	P16D	P16EN	P14D	P14EN	P12D	P12EN	P10D	P10EN	RW : 00

The Output Override to Port 1 Register (OUT_P1) enables specific internal signals to be output to Port 1 pins. If any other function, such as I2C, is enabled for output on these pins, that function has higher priority than the OUT_P1 signals.

Bit 7: P16D. This bit selects either the TIMEROUT or CLK32 signals for output on P1[6]. P16EN must be high for the signal to be output on that pin.

Bit 6: P16EN. This bit enables pin P1[6] for output of the signal selected by the P16D bit.

Bit 5: P14D. This bit selects either the RO or CMP1 signals for output on P1[4]. P14EN must be high for the signal to be output on that pin.

Bit 4: P14EN. This bit enables pin P1[4] for output of the signal selected by the P14D bit.

Bit 3: P12D. This bit selects either the SYSCLK or CS signals for output on P1[2]. P12EN must be high for the signal to be output on that pin.

Bit 2: P12EN. This bit enables pin P1[2] for output of the signal selected by the P12D bit.

Bit 1: P10D. This bit selects either the SLPINT or CMP0 signals for output on P1[0]. P10EN must be high for the signal to be output on that pin.

Bit 0: P10EN. This bit enables pin P1[0] for output of the signal selected by the P10D bit.

For additional information, refer to the OUT_P1 register on page 185.

13.2.2 OSC_CR0 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,E0h	OSC_CR0		Disable Buzz	No Buzz	Sleep[1:0]		CPU Speed[2:0]			RW : 01

The Oscillator Control Register 0 (OSC_CR0) is used to configure various features of internal clock sources and clock nets.

Bit 6: Disable Buzz. Setting this bit causes the bandgap and POR/LVD systems to remain powered off continuously during sleep. In this case, there is no periodic “buzz” (brief wakeup) of these functions during sleep. This bit has no effect when the No Buzz bit is set high.

Bit 5: No Buzz. Normally, when the Sleep bit is set in the CPU_SCR register, all PSoC device systems are powered down, including the bandgap reference. However, to facilitate the detection of POR and LVD events at a rate higher than the sleep interval, the bandgap circuit is powered up periodically (for about 60 µs) at the Sleep System Duty Cycle, which is independent of the sleep interval and typically higher. When the No Buzz bit is set, the Sleep System Duty Cycle value is overridden and the bandgap circuit is forced to be on during sleep. This results in faster response to an LVD or POR event (continuous detection as opposed to periodic), at the expense of higher average sleep current.

Bits 4 and 3: Sleep[1:0]. The available sleep interval selections are shown in the table below. Sleep intervals are approximate based on the accuracy of the internal low speed oscillator.

Sleep Interval OSC_CR[4:3]	Sleep Timer Clocks	Sleep Period (nominal)	Watchdog Period (nominal)
00b (default)	64	1.95 ms	6 ms
01b	512	15.6 ms	47 ms
10b	4096	125 ms	375 ms
11b	32,768	1 sec	3 sec

Bits 2 to 0: CPU Speed[2:0]. The PSoC M8C may operate over a range of CPU clock speeds as illustrated in the table below, allowing the M8C’s performance and power requirements to be tailored to the application.

The reset value for the CPU speed bits is 001b. Therefore, the default CPU speed is one fourth of the clock source. The internal main oscillator is the default clock source for the CPU speed circuit; therefore, the default CPU speed is 3.0 MHz. See “External Clock” on page 90 for more information on the supported frequencies for externally supplied clocks.

The CPU frequency is changed with a write to the OSC_CR0 register. There are eight frequencies generated from a power-of-two divide circuit which are selected by a 3-bit code. At any given time, the CPU 8-to-1 clock mux is selecting one of the available frequencies, which is resynchronized to the 12 MHz master clock at the output.

A slow IMO option is also supported, as discussed in the IMO chapter in the “Architectural Description” on page 57. This offers an option to lower both system and CPU clock speed in order to save power.

Bits	6 MHz Internal Main Oscillator	12 MHz Internal Main Oscillator	External Clock
000b	750 kHz	1.5 MHz	EXTCLK/ 8
001b	1.5 MHz	3.0 MHz	EXTCLK/ 4
010b	3 MHz	6.0 MHz	EXTCLK/ 2
011b	6 MHz	12.0 MHz	EXTCLK/ 1
100b	375 kHz	750 kHz	EXTCLK/ 16
101b	187.5 kHz	375 kHz	EXTCLK/ 32
110b	46.8 kHz	93.7 kHz	EXTCLK/ 128
111b	23.4 kHz	46.8 kHz	EXTCLK/ 256

An automatic protection mechanism is available for systems that need to run at peak CPU clock speed but cannot guarantee a high enough supply voltage for that clock speed. See the LVDTBEN bit in the “VLT_CR Register” on page 113 for more information.

For additional information, refer to the OSC_CR0 register on page 186.

13.2.3 OSC_CR2 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,E2h	OSC_CR2						EXTCLKEN	IMODIS		RW : 00

The Oscillator Control Register 2 (OSC_CR2) is used to configure various features of internal clock sources and clock nets.

Bit 2: EXTCLKEN. When the EXTCLKEN bit is set, the external clock becomes the source for the internal clock tree, SYSCLK, which drives most PSoC device clocking functions. All external and internal signals, including the low speed oscillator, are synchronized to this clock source. The external clock input is located on P1[4]. When using this input, the pin drive mode should be set to High-Z (not High-Z analog), such as drive mode 11b with PRT1DR bit 4 set high.

Bit 1: IMODIS. When set, the Internal Main Oscillator (IMO) is disabled.

For additional information, refer to the OSC_CR2 register on page 187.

13.2.4 Related Registers

- “INT_CLR0 Registers” on page 48.
- “INT_MSK0 Register” on page 49.

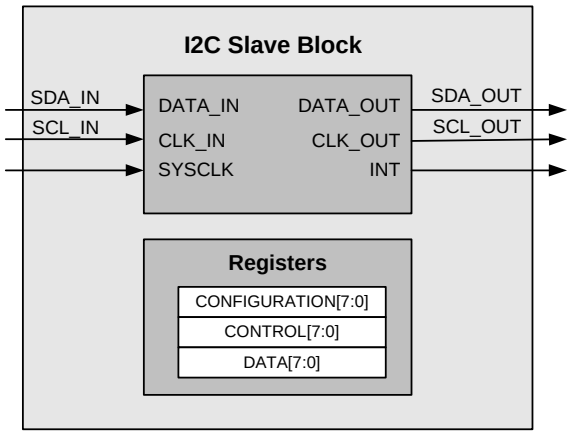
14. I²C Slave

This chapter explains the I²C™ Slave block and its associated registers. The I2C communications block is a serial processor designed to implement a complete I2C slave. For a complete table of the I2C registers, refer to the "Summary Table of the System Resource Registers" on page 88. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

14.1 Architectural Description

The I2C communications block is a serial to parallel processor, designed to interface the PSoC device to a two-wire I2C serial communications bus. To eliminate the need for excessive M8C microcontroller intervention and overhead, the block provides I2C specific support for status detection and generation of framing bits.

Figure 14-1. I²C Slave Block Diagram



The I2C block controls the data (SDA) and the clock (SCL) to the external I2C interface, through direct connections to two dedicated GPIO pins. When I2C is enabled, these GPIO pins are not available for general purpose use. The PSoC device firmware interacts with the block through IO (input/output) register reads and writes, and firmware synchronization will be implemented through polling and/or interrupts.

PSoC I2C features include:

- Slave, Transmitter/Receiver operation
- Byte processing for low CPU overhead
- Interrupt or polling CPU interface
- 7- or 10-bit addressing (through firmware support)
- SMBus operation (through firmware support)

Hardware functionality provides basic I2C control, data, and status primitives. A combination of hardware support and firmware command sequencing provides a high degree of flexibility for implementing the required I2C functionality.

Hardware limitations in regards to I2C are:

1. There is no hardware support for automatic address comparison. Every slave address will cause the block to interrupt the PSoC device and possibly stall the bus.
2. Since receive and transmitted data are not buffered, there is no support for automatic receive acknowledge. The M8C microcontroller must intervene at the boundary of each byte and either send a byte or ACK received bytes.

The I2C block is designed to support a set of primitive operations and detect a set of status conditions specific to the I2C protocol. These primitive operations and conditions are manipulated and combined at the firmware level to support the required data transfer modes. The CPU will set up control options and issue commands to the unit through IO writes and obtain status through IO reads and interrupts.

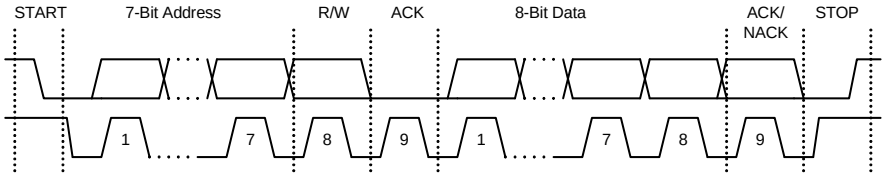
The block operates as a slave. In Slave mode, the unit is always listening for a Start condition, or sending or receiving data.

14.1.1 Basic I²C Data Transfer

Figure 14-2 shows the basic form of data transfers on the I2C bus with a 7-bit address format. (For a more detailed description, see the Philips Semiconductors' I²C™ Specification, version 2.1.)

A Start condition (generated by the master) is followed by a data byte, consisting of a 7-bit slave address (there is also a 10-bit address mode) and a Read/Write (RW) bit. The RW bit sets the direction of data transfer. The addressed slave is required to acknowledge (ACK) the bus by pulling the data line low during the ninth bit time. If the ACK is received, the transfer may proceed and the master can transmit or receive an indeterminate number of bytes, depending on the RW direction. If the slave does not respond with an ACK for any reason, a Stop condition is generated by the master to terminate the transfer or a Restart condition may be generated for a retry attempt.

Figure 14-2. Basic I²C Data Transfer with 7-Bit Address Format



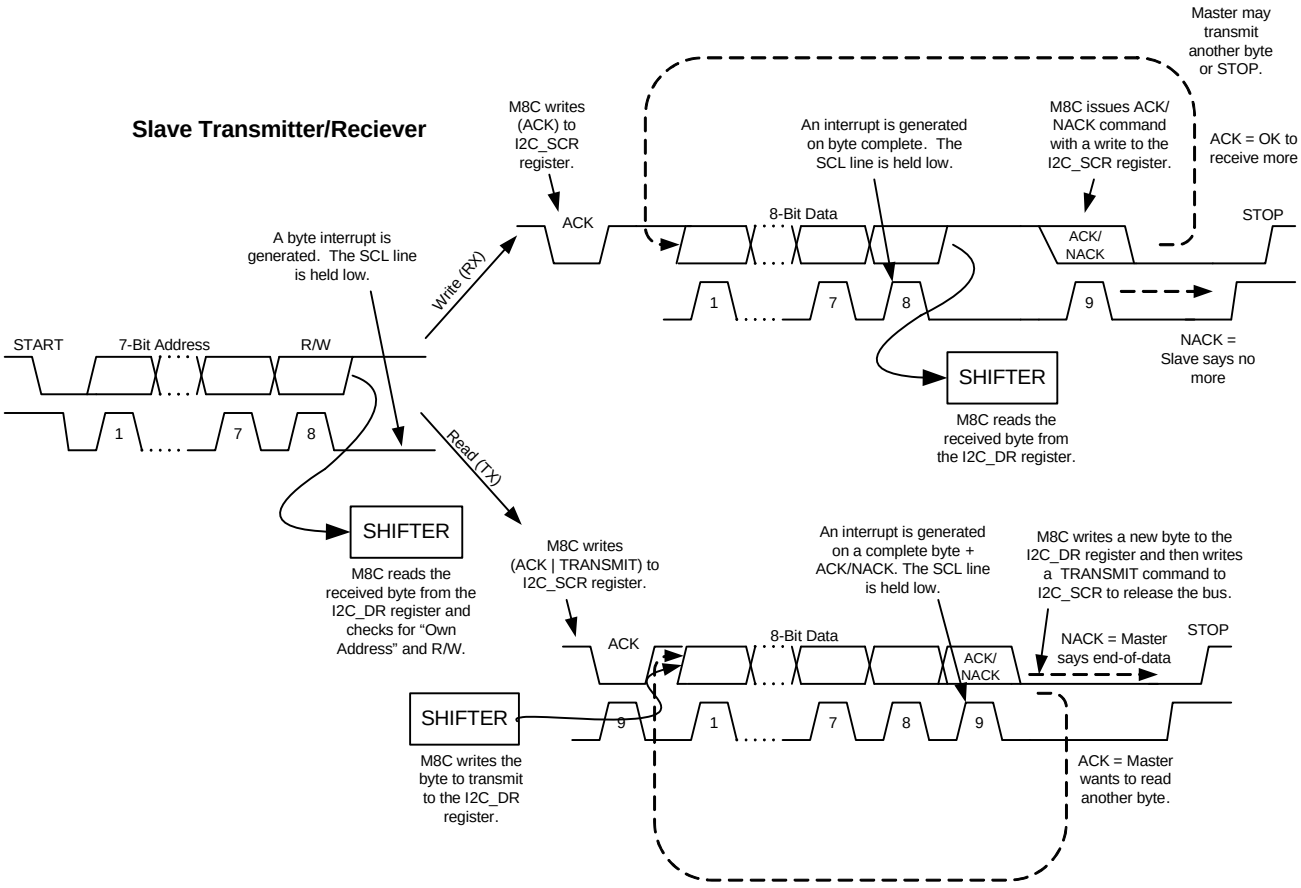
14.2 Application Overview

14.2.1 Slave Operation

When the IC slave operation is enabled, it is continually listening to or on the bus for a Start condition. When detected, the transmitted Address/RW byte is received and read from the I2C block by firmware. At the point where eight bits of the address/RW byte have been received, a byte complete interrupt is generated. On the following low of the clock, the bus is stalled by holding the SCL line low, until the PSoC device has had a chance to read the address byte and compare it to its own address. It will issue an ACK or NACK command based on that comparison.

If there is an address match, the RW bit determines how the PSoC device will sequence the data transfer in Slave mode, as shown in the two branches of Figure 14-3. I2C handshaking methodology (slave holds the SCL line low to "stall" the bus) will be used as necessary, to give the PSoC device time to respond to the events and conditions on the bus. Figure 14-3 is a graphical representation of a typical data transfer from the slave perspective.

Figure 14-3. Slave Operation



14.3 Register Definitions

The following registers are associated with I2C Slave and are listed in address order. Each register description has an associated register table showing the bit structure for that register. The bits in the tables that are grayed out are reserved bits and are not detailed in the register descriptions that follow. Always write reserved bits with a value of '0'. For a complete table of I2C registers, refer to the "Summary Table of the System Resource Registers" on page 88.

14.3.1 I2C_CFG Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,D6h	I2C_CFG		PSelect		Stop IE	Clock Rate[1:0]			Enable	RW : 00

The I2C Configuration Register (I2C_CFG) is used to set the basic operating modes, baud rate, and selection of interrupts.

The bits in this register control baud rate selection and optional interrupts. The values are typically set once for a given configuration. The bits in this register are all RW.

Bit	Access	Description	Mode
6	RW	I2C Pin Select 0 = P1[7], P1[5] 1 = P1[1], P1[0]	Slave
4	RW	Stop IE Stop interrupt enable. 0 = Disabled. 1 = Enabled. An interrupt is generated on the detection of a Stop Condition.	Slave
3:2	RW	Clock Rate 00 = 100K Standard Mode 01 = 400K Fast Mode 10 = 50K Standard Mode 11 = Reserved	Slave

Bit 6: PSelect. Pin Select. With the default value of zero, the I2C pins are P1[7] for clock and P1[5] for data. When this bit is set, the pins for I2C switch to P1[1] for clock and P1[0] for data. This bit may not be changed while the Enable bit is set. However, the PSelect bit may be set at the same time as the enable bits. The two sets of pins that may be used on I2C are not equivalent. The default set, P1[7] and P1[5], are the preferred set. The alternate set, P1[1] and P1[0], are provided so that I2C may be used with 8-pin PSoC devices.

If In-circuit System Serial Programming (ISSP®) is used and the alternate I2C pin set is also used, it is necessary to take into account the interaction between the PSoC Test Controller and the I2C bus. The interface requirements for ISSP should be reviewed to ensure that they are not violated.

Even if ISSP is not used, pins P1[1] and P1[0] will respond differently to a POR or XRES event than other IO pins. After an XRES event, both pins are pulled down to ground by going into the resistive zero drive mode, before reaching the High-Z Drive mode. After a POR event, P1[0] will drive out a one, then go to the resistive zero state for some time, and finally reach the High-Z drive mode state. After POR, P1[1] will go into a resistive zero state for a while, before going to the High-Z Drive mode.

Bit 4: Stop IE. Stop Interrupt Enable. When this bit is set, a slave can interrupt on Stop detection. The status bit associated with this interrupt is the Stop Status bit in the I2C_SCR register. When the Stop Status bit transitions from '0' to '1', the interrupt is generated. It is important to note that the Stop Status bit is not automatically cleared. Therefore, if it is already set, no new interrupts are generated until it is cleared by firmware.

Bits 3 and 2: Clock Rate[1:0]. These bits offer a selection of three sampling and bit rates. All block clocking is based on the SYSCLK input, which is nominally 12 MHz or 6 MHz (unless the PSoC device is in external clocking mode). The sampling rate and the baud rate are determined as follows:

- Sample Rate = SYSCLK/Pre-scale Factor
- Baud Rate = 1/(Sample Rate x Samples per Bit)

The nominal values, when using the internal 12 MHz or 6 MHz oscillator, are shown in this table:.

Clock Rate [1:0]	SLIMO	I2C Mode	SYSCLK Pre-scale Factor	Samples per Bit	Internal Sampling Freq./Period (12 MHz)	Master Baud Rate (nominal)	Start/Stop Hold Time (8 clocks)
00b	0	Standard	/8	16	1.5 MHz/ 667 ns	93.75 kHz	5.3 µs
	1		/4				
01b	0	Fast	/2	16	6 MHz/ 167 ns	375 kHz	1.33 µs
	1		/1				
10b	0	Standard	/8	32	1.5 MHz/ 667 ns	46.8 kHz	10.7 µs
	1		/4				
11b	0	Reserved					
	1						

When clocking the input with a frequency other than 6/12 MHz (for example, clocking the PSOC device with an external clock), the baud rates and sampling rates will scale accordingly. Whether the block works in a Standard Mode or Fast Mode system depends upon the sample rate. The sample rate must be sufficient to resolve bus events, such as Start and Stop conditions. (See the Philips Semiconductors' I^2C™ Specification, version 2.1, for minimum Start and Stop hold times.)

Bit 0: Enable. When the slave is enabled, the block generates an interrupt on any Start condition and an address byte that it receives indicating the beginning of an I2C transfer. The block is clocked from an external master. Therefore, the block works at any frequency up to the maximum defined by the currently selected clock rate. The internal clock is only used to ensure that there is adequate setup time from data output to the next clock on the release of a slave stall. When the Enable bit is '0', the block is held in reset and all status is cleared. Block enable will be synchronized to the SYSCLK clock input (see "Timing Diagrams" on page 101).

Enable	Block Operation
No	Disabled The block is disconnected from the GPIO pins, P1[5] and P1[7]. (The pins may be used as general purpose IO.) When the slave is enabled, the GPIO pins are under control of the I2C hardware and are unavailable. All internal registers (except I2C_CFG) are held in reset.
Yes	Slave Mode Any external Start condition will cause the block to start receiving an address byte. Regardless of the current state, any Start resets the interface and initiates a Receive operation. Any Stop will cause the block to revert to an idle state

For additional information, refer to the I2C_CFG register on page 167.

14.3.2 I2C_SCR Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,D7h	I2C_SCR	Bus Error		Stop Status	ACK	Address	Transmit	LRB	Byte Complete	# : 00

LEGEND
Access is bit specific.

The I2C Status and Control Register (I2C_SCR) is used by the slave to control the flow of data bytes and to keep track of the bus state during a transfer.

This register contains status bits, for determining the state of the current I2C transfer, and control bits, for determining the actions for the next byte transfer. At the end of each byte transfer, the I2C hardware interrupts the M8C microcontroller and stalls the I2C bus on the subsequent low of the clock, until the PSoC device intervenes with the next command. This register may be read as many times as necessary; but on a subsequent write to this register, the bus stall is released and the current transfer will continue.

There are six status bits: Byte Complete, LRB, Address, Stop Status, Lost Arb, and Bus Error. These bits have Read/Clear (RC) access, which means that they are set by hardware but may be cleared by a write of '0' to the bit position. Under certain conditions, status is cleared automatically by the hardware. These cases are noted in the table shown below.

There are two control bits: Transmit and ACK. These bits have RW access and may be cleared by hardware.

Bit 7: Bus Error. The Bus Error status detects misplaced Start or Stop conditions on the bus. These may be due to noise, rogue devices, or other devices that are not yet synchronized with the I2C bus traffic. According to the I2C specification, all compatible devices must reset their interface on a received Start or Stop. This is a natural thing to do in Slave mode because a Start will initiate an address reception and a Stop will idle the slave.

A bus error is defined as follows. A Start is only valid if the block is idle or a Slave receiver is ready to receive the first bit of a new byte after an ACK. Any other timing for a Start condition causes the Bus Error bit to be set. A Stop is only valid if the block is idle or a Slave receiver is ready to receive the first bit of a new byte after an ACK. Any other timing for a Stop condition causes the Bus Error bit to be set.

Bit	Access	Description
7	RC	Bus Error 1 = A misplaced Start or Stop condition was detected. This status bit must be cleared by firmware with a write of '0' to the bit position. It is never cleared by the hardware.
5	RC	Stop Status 1 = A Stop condition was detected. This status bit must be cleared by firmware with a write of '0' to the bit position. It is never cleared by the hardware.
4	RW	ACK: Acknowledge Out 0 = NACK the last received byte. 1 = ACK the last received byte. This bit is automatically cleared by hardware on the following Byte Complete event.
3	RC	Address 1 = The transmitted or received byte is an address. This status bit must be cleared by firmware with a write of '0' to the bit position.
2	RW	Transmit 0 = Receive Mode. 1 = Transmit Mode. This bit is set by firmware to define the direction of the byte transfer. Any Start detect will automatically clear this bit.
1	RC	LRB: Last Received Bit The value of the ninth bit in a Transmit sequence, which is the acknowledge bit from the receiver. 0 = Last transmitted byte was ACK'ed by the receiver. 1 = Last transmitted byte was NACK'ed by the receiver. Any Start detect will automatically clear this bit.
0	RC	Byte Complete Transmit Mode: 1 = 8 bits of data have been transmitted and an ACK or NACK has been received. Receive Mode: 1 = 8 bits of data have been received. Any Start detect will automatically clear this bit.

Bit 5: Stop Status. Stop status is set on detection of an I2C Stop condition. This bit is sticky, which means that it will remain set until a '0' is written back to it by the firmware. This bit may only be cleared if the Byte Complete status bit is set. If the Stop Interrupt Enable bit is set, an interrupt is also generated on Stop detection. It is never automatically cleared.

Using this bit, a slave can distinguish between a previous Stop or Restart on a given address byte interrupt.

Bit 4: ACK. This control bit defines the acknowledge data bit that is transmitted out in response to a received byte. When receiving, a Byte Complete interrupt is generated after the eighth data bit is received. On the subsequent write to this register to continue (or terminate) the transfer, the state of this bit will determine the next bit of data that is transmitted. It is ***active high***. A '1' will send an ACK and a '0' will send a NACK. A Slave receiver sends a NACK to inform the master that it cannot receive any more bytes.

Bit 3: Address. This bit is set when an address has been received. This consists of a Start or Restart, and an address byte.

In Slave mode, when this status is set, firmware will read the received address from the data register and compare it with its own address. If the address does not match, the firmware will write a NACK indication to this register. No further interrupts will occur until the next address is received. If the address does match, firmware must ACK the received byte, then Byte Complete interrupts are generated on subsequent bytes of the transfer.

Bit 2: Transmit. This bit sets the direction of the shifter for a subsequent byte transfer. The shifter is always shifting in data from the I2C bus, but a write of '1' enables the output of the shifter to drive the SDA output line. Since a write to this register initiates the next transfer, data must be written to the data register prior to writing this bit. In Receive mode, the previously received data must have been read from the data register before this write. Firmware derives this direction from the RW bit in the received slave address.

This direction control is only valid for data transfers. The direction of address bytes is determined by the hardware.

Bit 1: LRB. Last Received Bit. This is the last received bit in response to a previously transmitted byte. In Transmit mode, the hardware will send a byte from the data register and clock in an acknowledge bit from the receiver. On the subsequent byte complete interrupt, firmware will check the value of this bit. A '0' is the ACK value and a '1' is a NACK value. The meaning of the LRB depends on the current operating mode.

'0': ACK. The master wants to read another byte. The slave should load the next byte into the I2C_DR register and set the transmit bit in the I2C_SCR register to continue the transfer.

'1': NACK. The master is done reading bytes. The slave will revert to IDLE state on the subsequent I2C_SCR write (regardless of the value written).

Bit 0: Byte Complete. The I2C hardware operates on a byte basis. In Transmit mode, this bit is set and an interrupt is generated at the end of nine bits (the transmitted byte + the received ACK). In Receive mode, the bit is set after the eight bits of data are received. When this bit is set, an interrupt is generated at these data sampling points, which are associated with the SCL input clock rising (see details in "Timing Diagrams" on page 101). If the PSoC device responds with a write back to this register before the subsequent falling edge of SCL (which is approximately one-half bit time), the transfer will continue without interruption. However, if the PSoC device is unable to respond within that time, the hardware will hold the SCL line low, stalling the I2C bus. A subsequent write to the I2C_SCR register will release the stall.
For additional information, refer to the I2C_SCR register on page 168.

14.3.3 I2C_DR Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,D8h	I2C_DR	Data[7:0]								RW : 00

The I2C Data Register (I2C_DR) provides read/write access to the Shift register.

Bits 7 to 0: Data[7:0]. This register is not buffered; therefore, writes and valid data reads may only occur at specific points in the transfer. These cases are outlined as follows.

- **Slave Receiver** – Data in the I2C_DR register is only valid for reading when the Byte Complete status bit is set. Data bytes must be read from the I2C_DR register before writing to the I2C_SCR register, which continues the transfer.
- **Slave Transmitter** – Data bytes must be written to the I2C_DR register before the transmit bit is set in the I2C_SCR register, which continues the transfer.

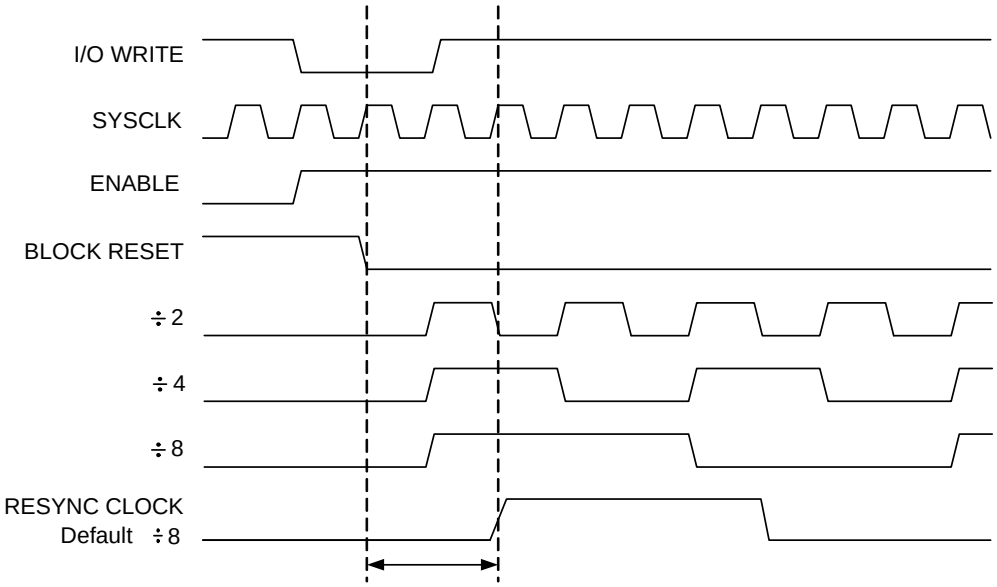
For additional information, refer to the I2C_DR register on page 169.

14.4 Timing Diagrams

14.4.1 Clock Generation

Figure 14-4 illustrates the I2C input clocking scheme. The SYSCLK pin is an input into a three-stage ripple divider that provides the baud rate selections. When the block is disabled, all internal state is held in a reset state. When the Enable bit in the I2C_CFG register is set, the reset is synchronously released and the clock generation is enabled. All three taps from the ***ripple divider*** are selectable (/2, /4, /8) from the clock rate bits in the I2C_CFG register. If any of the three divider taps is selected, that clock is resynchronized to SYSCLK. The resulting clock is routed to all of the synchronous elements in the design.

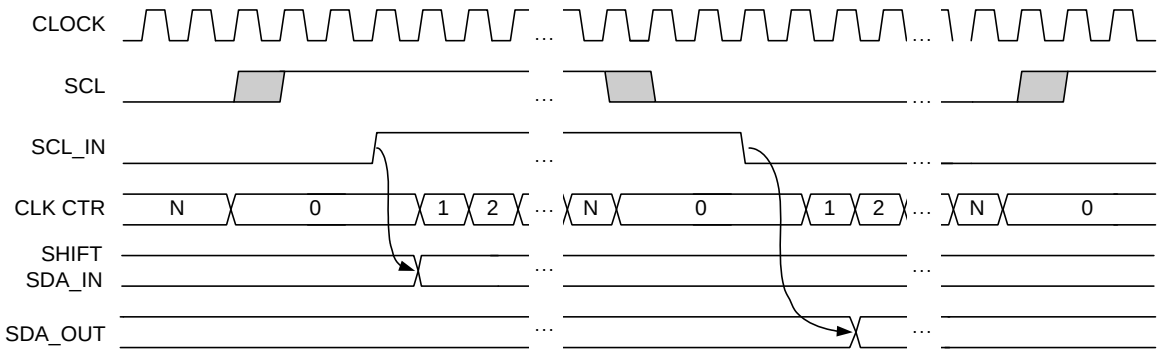
Figure 14-4. I^2C Input Clocking



14.4.2 Basic IO Timing

Figure 14-5 illustrates basic input output timing that is valid for both 16 times sampling and 32 times sampling. For 16 times sampling, N=4; for 32 times sampling, N=12. N is derived from the half-bit rate sampling of eight and 16 clocks, respectively, minus the input latency of three (count of 4 and 12 correspond to 5 and 13 clocks).

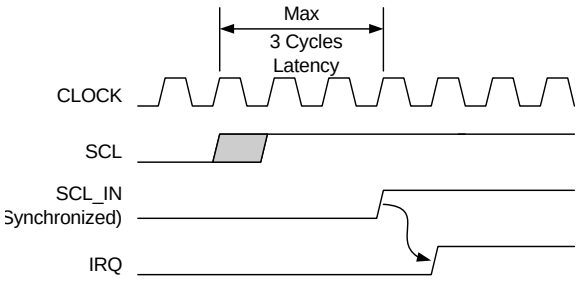
Figure 14-5. Basic Input/Output Timing



14.4.3 Status Timing

Figure 14-6 illustrates the interrupt timing for Byte Complete, which occurs on the positive edge of the ninth clock (byte + ACK/NACK) in Transmit mode and on the positive edge of the eighth clock in Receive mode. There is a maximum of three cycles of latency due to the input synchronizer/filter circuit. As shown, the interrupt occurs on the clock following a valid SCL positive edge input transition (after the synchronizers). The Address bit is set with the same timing but only after a slave address has been received. The LRB (Last Received Bit) status is also set with the same timing but only on the ninth bit after a transmitted byte.

Figure 14-6. Byte Complete, Address, LRB Timing



Transmit: Ninth positive edge SCL
Receive: Eighth positive edge SCL

Figure 14-7 shows the timing for Stop Status. This bit is set (and the interrupt occurs) two clocks after the synchronized and filtered SDA line transitions to a '1', when the SCL line is high.

Figure 14-7. Stop Status and Interrupt Timing

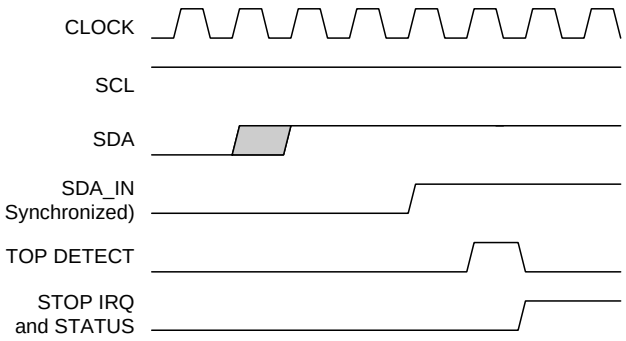
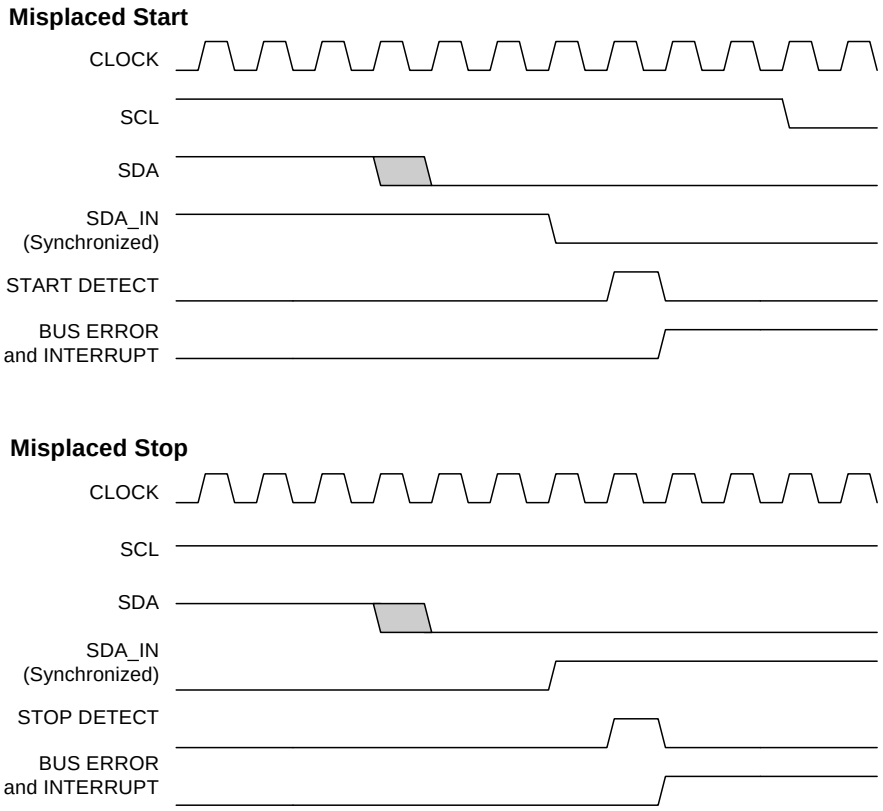


Figure 14-8 illustrates the timing for bus error interrupts. Bus Error status (and Interrupt) occurs one cycle after the internal Start or Stop Detect (two cycles after the filtered and synchronized SDA input transition).

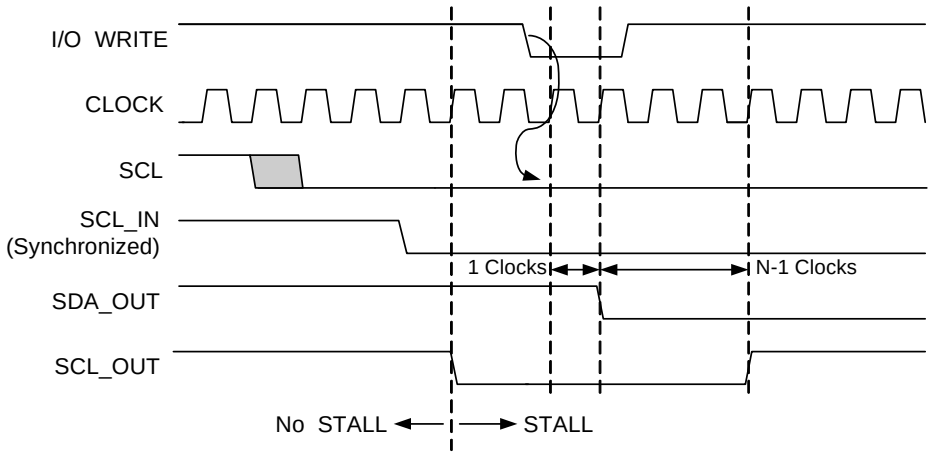
Figure 14-8. Bus Error Interrupt Timing



14.4.4 Slave Stall Timing

When a Byte Complete interrupt occurs, the PSoC device firmware must respond with a write to the I2C_SCR register to continue the transfer (or terminate the transfer). The interrupt occurs two clocks after the rising edge of SCL_IN (see "Status Timing" on page 102). As illustrated in Figure 14-9, firmware has until one clock after the falling edge of SCL_IN to write to the I2C_SCR register; otherwise, a stall occurs. Once stalled, the IO write releases the stall. The setup time between data output and the next rising edge of SCL is always N-1 clocks.

Figure 14-9. Slave Stall Timing



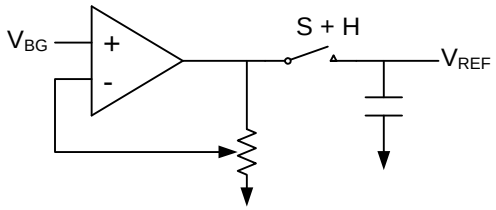
15. Internal Voltage References

This chapter discusses the Internal Voltage References and their associated register. The internal voltage references provide an absolute value of 1.3V to a variety of subsystems in the PSoC device. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

15.1 Architectural Description

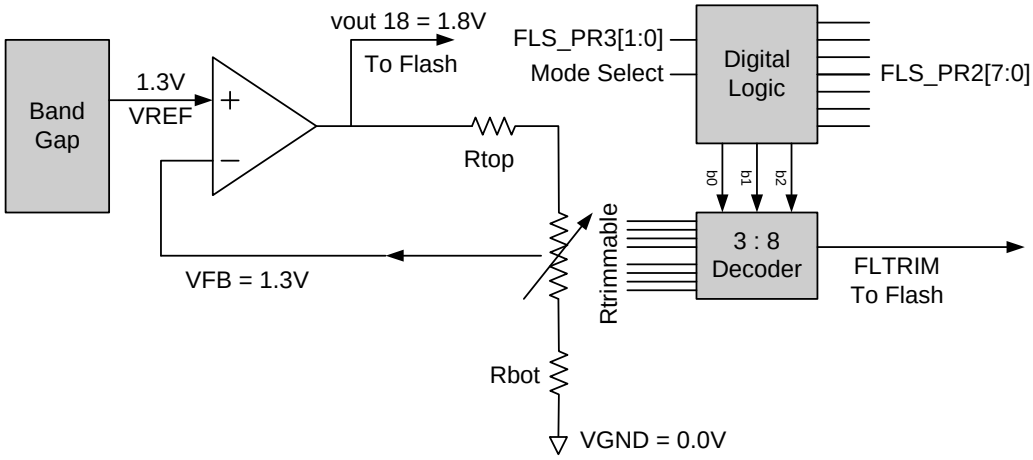
The internal voltage references consist of two blocks: a bandgap voltage generator and a buffer with sample and hold. The bandgap generator is typically a (V_{BE} + K V_T) design, where K is a numerical constant determined by circuit parameters.The buffer circuit provides ***gain*** to the 1.2V bandgap voltage, to produce a 1.3V reference. A simplified ***schematic*** is illustrated in Figure 15-1. The connection between amplifier and capacitor is made through a CMOS switch, allowing the reference voltage to be used by the system while the reference circuit is powered down. The voltage reference is trimmed to 1.30V at room temperature.

Figure 15-1. Voltage Reference Schematic



Another block, which is associated with the internal voltage reference circuitry, is the Flash trim buffer. Figure 15-2 shows the conceptual block diagram for the Flash trim buffer.

Figure 15-2. Flash Trim Buffer



15.2 Register Definitions

This register is associated with the Internal Voltage References. The Internal Voltage References are trimmed for gain and temperature coefficient using the BDG_TR register. The register description below has an associated register table showing the bit structure.

15.2.1 BDG_TR Register

Add.	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,EAh	BDG_TR	TC[2:0]			V[4:0]					RW : 50

The Bandgap Trim Register (BDG_TR) is used to adjust the bandgap and add an RC filter to Agnd.

It is strongly recommended that the user not alter the value of the bits in this register.

Bits 7 to 5: TC[2:0]. These bits are for setting the temperature coefficient inside the bandgap voltage generator.

The value of these bits is used to trim the temperature coefficient. Their value is set to the best value for the device during boot. The value of these bits should not be changed.

Bits 4 to 0: V[4:0]. These bits are for setting the gain in the reference buffer. 32 steps of 2.129 mV are available.

The value of these bits is used to trim the bandgap reference. Their value is set to the best value for the device during boot. The value of these bits should not be changed.

For additional information, refer to the BDG_TR register on page 192.

16. System Resets

This chapter discusses the System Resets and their associated registers. PSoC devices support several types of resets. The various resets are designed to provide error-free operation during power up for any voltage ramping profile, to allow for user-supplied external reset and to provide recovery from errant code operation. For a complete table of the System Reset registers, refer to the "Summary Table of the System Resource Registers" on page 88. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

16.1 Architectural Description

When reset is initiated, all registers are restored to their default states. In the Register Reference chapter on page 137, this is indicated by the POR row in the register tables and elsewhere it is indicated in the Access column, values on the right side of the colon, in the register tables. Minor exceptions are explained below.

The following types of resets can occur in the PSoC device:

- Power on Reset (POR). This occurs at low supply voltage and is comprised of multiple sources.
- External Reset (XRES). This active high reset is driven into the PSoC device on parts that contain an XRES pin.
- Watchdog Reset (WDR). This optional reset occurs when the watchdog timer expires before being cleared by user firmware. Watchdog reset defaults to off.
- Internal Reset (IRES). This occurs during the boot sequence if the SROM code determines that Flash reads are not valid.

The occurrence of a reset is recorded in the Status and Control registers (CPU_SCR0 for POR, XRES, and WDR) or in the System Status and Control Register 1 (CPU_SCR1 for IRESS). Firmware can interrogate these registers to determine the cause of a reset.

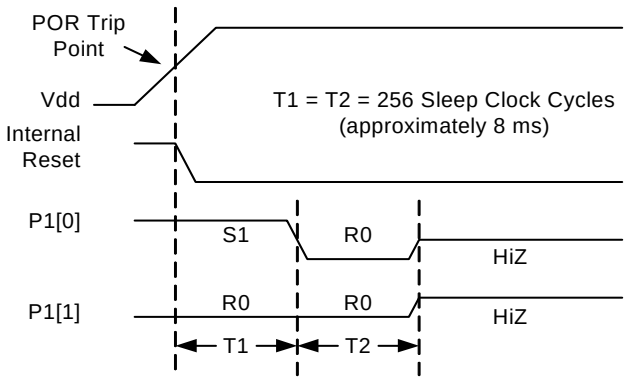
16.2 Pin Behavior During Reset

Power on Reset and External Reset cause toggling on two GPIO pins, P1[0] and P1[1], as described below and illustrated in Figure 16-1 and Figure 16-2. This allows programmers to synchronize with the PSoC device. All other GPIO pins are placed in a high impedance state during and immediately following reset.

16.2.1 GPIO Behavior on Power Up

At power up, the internal POR causes P1[0] to initially drive a strong high (1) while P1[1] drives a resistive low (0). After 256 sleep oscillator cycles (approximately 8 ms), the P1[0] signal transitions to a resistive low state. After an additional 256 sleep oscillator clocks, both pins transition to a high impedance state and normal CPU operation begins. This is illustrated in Figure 16-1.

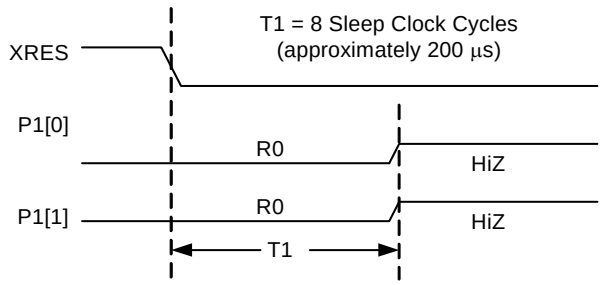
Figure 16-1. P1[1:0] Behavior on Power Up



16.2.2 GPIO Behavior on External Reset

During External Reset (XRES=1), both P1[0] and P1[1] drive resistive low (0). After XRES de-asserts, these pins continue to drive resistive low for another 8 sleep clock cycles (approximately 200 µs). After this time, both pins transition to a high impedance state and normal CPU operation begins. This is illustrated in Figure 16-2.

Figure 16-2. P1[1:0] Behavior on External Reset (XRES)



16.3 Register Definitions

The following registers are associated with the PSoC System Resets and are listed in address order. Each register description has an associated register table showing the bit structure for that register. The bits in the tables that are grayed out are reserved bits and are not detailed in the register descriptions that follow. Always write reserved bits with a value of '0'. For a complete table of system reset registers, refer to the "Summary Table of the System Resource Registers" on page 88.

16.3.1 CPU_SCR1 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
x,FEh	CPU_SCR1	IRESS			SLIMO				IRAMDIS	# : 00

LEGEND
x An "x" before the comma in the address field indicates that this register can be read or written to no matter what bank is used.
Access is bit specific. Refer to the Register Reference chapter on page 137 for additional information.

The System Status and Control Register 1 (CPU_SCR1) is used to convey the status and control of events related to internal resets and watchdog reset.

Bit 7: IRESS. Internal Reset Status. This bit is a read only bit that may be used to determine if the booting process occurred more than once.

When this bit is set, it indicates that the SROM SWBootReset code was executed more than once. If this bit is not set, the SWBootReset was executed only once. In either case, the SWBootReset code will not allow execution from code stored in Flash until the M8C core is in a safe operating mode with respect to supply voltage and Flash operation. There is no need for concern when this bit is set. It is provided for systems which may be sensitive to boot time, so that they can determine if the normal one-pass boot time was exceeded. For more information on the SWBootReest code see the Supervisory ROM (SROM) chapter on page 37.

Bit 4: SLIMO. Slow IMO. When set, this bit allows the active power dissipation of the PSoC device to be reduced by slowing down the IMO from 12 MHz to 6 MHz. The IMO trim value must also be changed when SLIMO is set (see "Engaging Slow IMO" on page 57). When not in external clocking mode, the IMO is the source for SYSCLK; therefore, when the speed of the IMO changes so will SYSCLK.

Bit 0: IRAMDIS. Initialize RAM Disable. This bit is a control bit that is readable and writeable. The ***default value*** for this bit is '0', which indicates that the maximum amount of SRAM should be initialized on watchdog reset to a value of 00h. When the bit is '1', the minimum amount of SRAM is initialized after a watchdog reset. For more information on this bit, see the "SROM Function Descriptions" on page 38.

For additional information, refer to the CPU_SCR1 register on page 178.

16.3.2 CPU_SCR0 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,FFh	CPU_SCR0	GIES		WDRS	PORS	Sleep			STOP	# : XX

LEGEND
Access is bit specific. Refer to register detail for additional information.
XX The reset value is 10h after POR/XRES and 20h after a watchdog reset.

The System Status and Control Register 0 (CPU_SCR0) is used to convey the status and control of events for various functions of a PSoC device.

Bit 7: GIES. Global Interrupt Enable Status. This bit is a read only status bit and its use is discouraged. The GIES bit is a legacy bit which was used to provide the ability to read the GIE bit of the CPU_F register. However, the CPU_F register is now readable. When this bit is set, it indicates that the GIE bit in the CPU_F register is also set which, in turn, indicates that the microprocessor will service interrupts.

Bit 5: WDRS. WatchDog Reset Status. This bit may not be set. It is normally '0' and automatically set whenever a watchdog reset occurs. The bit is readable and clearable by writing a zero to its bit position in the CPU_SCR0 register.

Bit 4: PORS. Power On Reset Status. This bit, which is the watchdog enable bit, is set automatically by a POR or External Reset (XRES). If the bit is cleared by user code, the watchdog timer is enabled. Once cleared, the only way to reset the PORS bit is to go through a POR or XRES. Thus, there is no way to disable the watchdog timer other than to go through a POR or XRES.

Bit 3: Sleep. This bit is used to enter Low Power Sleep mode when set. To wake up the system, this register bit is cleared asynchronously by any enabled interrupt. There are two special features of this bit that ensures proper Sleep operation. First, the write to set the register bit is blocked, if an interrupt is about to be taken on that instruction boundary (immediately after the write). Second, there is a hardware interlock to ensure that, once set, the Sleep bit may not be cleared by an incoming interrupt until the sleep circuit has finished performing the sleep sequence and the system-wide power down signal has been asserted. This prevents the sleep circuit from being interrupted in the middle of the process of system power down, possibly leaving the system in an indeterminate state.

Bit 0: STOP. This bit is readable and writeable. When set, the PSoC M8C will stop executing code until a reset event occurs. This can be either a POR, WDR, or XRES. If an application wants to stop code execution until a reset, the preferred method would be to use the HALT instruction rather than a register write to this bit.

For additional information, refer to the CPU_SCR0 register on page 179.

16.4 Timing Diagrams

16.4.1 Power On Reset

A Power on Reset (POR) is triggered whenever the supply voltage is below the POR trip point. POR ends once the supply voltage rises above this voltage. Refer to the POR and LVD chapter on page 113 for more information on the operation of the POR block.

POR consists of two pieces: an imprecise POR (IPOR) and a Precision POR (PPOR). “POR” refers to the OR of these two functions. IPOR has coarser accuracy and its trip point is typically lower than PPOR’s trip point. PPOR is derived from a circuit that is calibrated (during boot) for a very accurate location of the POR trip point.

During POR (POR=1), the IMO is powered off for low power during start-up. Once POR de-asserts, the IMO is started (see Figure 16-3).

POR configures register reset status bits as shown in Table 16-1 on page 112. PPOR does not affect the Band-Gap Trim register (BDG_TR), but IPOR does reset this register.

16.4.2 External Reset

An External Reset (XRES) is caused by pulling the XRES pin high. The XRES pin has an always-on, pull down resistor, so it does not require an external pull down for operation and can be tied directly to ground or left open. Behavior after XRES is similar to POR.

During XRES (XRES=1), the IMO is powered off for low power during start-up. Once XRES de-asserts, the IMO is started (see Figure 16-3).

How the XRES configures register reset status bits is shown in Table 16-1 on page 112.

16.4.3 Watchdog Timer Reset

The user has the option to enable the Watchdog Timer Reset (WDR), by clearing the PORS bit in the CPU_SCR0 register. Once the PORS bit is cleared, the watchdog timer cannot be disabled. The only exception to this is if a POR/XRES event takes place which will disable the WDR. Note that a WDR does not clear the Watchdog timer. See “Watchdog Timer” on page 66 for details of the Watchdog operation.

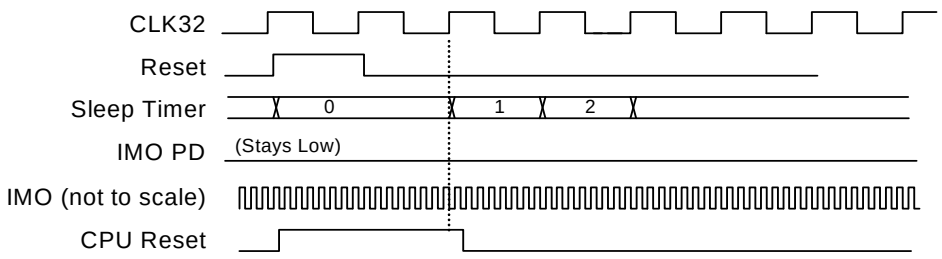
When the watchdog timer expires, a watchdog event occurs resulting in the reset sequence. Some characteristics unique to the WDR are as follows.

- PSoC device reset asserts for one cycle of the CLK32K clock (at its reset state).
- The IMO is not halted during or after WDR (that is, the part does not go through a low power phase).
- CPU operation restarts one CLK32K cycle after the internal reset de-asserts (see Figure 16-3).

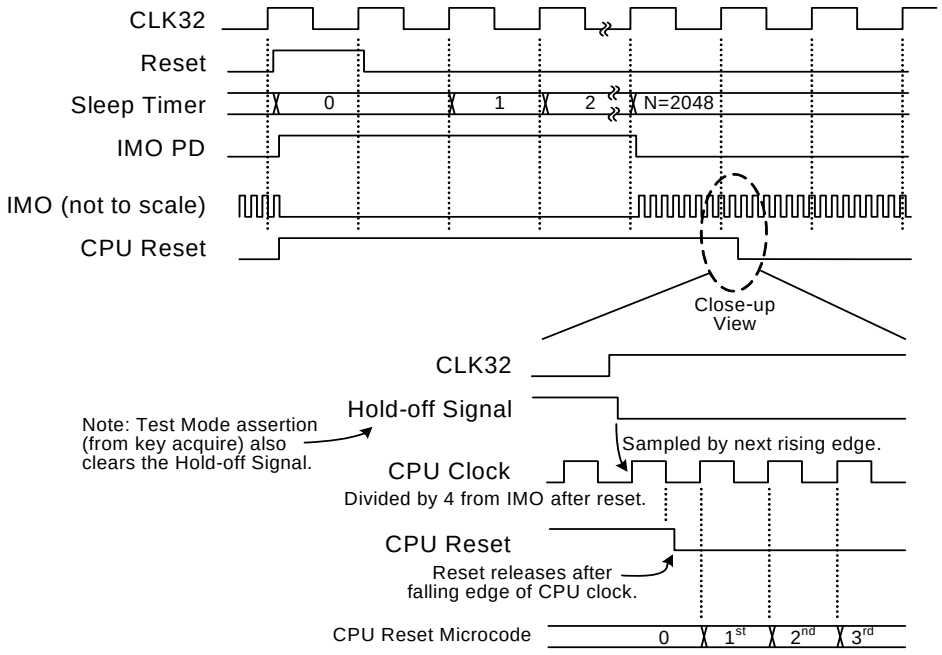
How the WDR configures register reset status bits is shown in Table 16-1 on page 112.

Figure 16-3. Key Signals During WDR and IRES Key Signals During POR and XRES

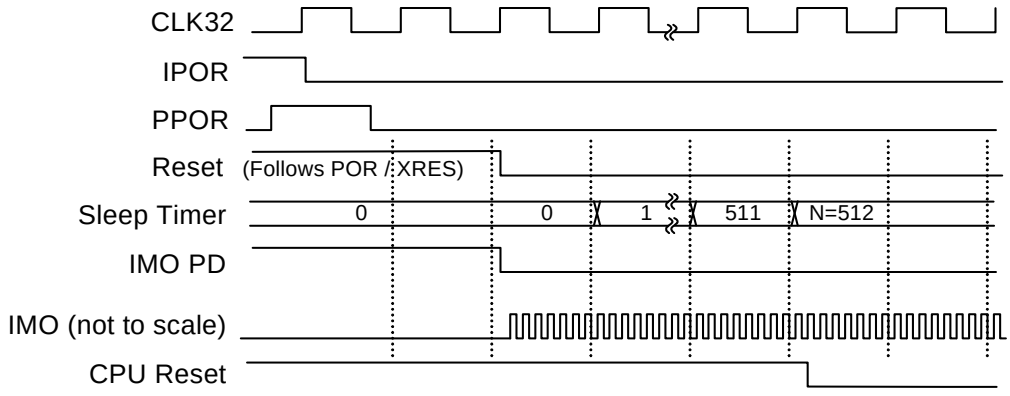
WDR: Reset 1 cycle, then one additional cycle before CPU reset released.



IRES: Reset 1 cycle, then 2048 additional cycles low power hold-off, and then 1 cycle with IMO on before CPU reset released.



POR (IPOR followed by PPOR): Reset while POR is high (IMO off), then 511(+) cycles (IMO on), and then the CPU reset is released. **XRES** is the same, with N=8.



16.4.4 Reset Details

Timing and functionality details are summarized in Table 16-1. Figure 16-3 shows some of the relevant signals for IPOR, PPOR, and XRES, while Figure 16-3 shows signaling for WDR and IRES.

Table 16-1. Details of Functionality for Various Resets

Item	IPOR (Part of POR)	PPOR (Part of POR)	XRES	WDR
Reset Length	While POR=1	While PPOR=1, plus 30-60 µs (1-2 clocks)	While XRES=1	30 µs (1 clock)
Low Power (IMO Off) During Reset?	Yes	Yes	Yes	No
Low Power Wait Following Reset?	No	No	No	No
CLK32K Cycles from End of Reset to CPU Reset De-asserts[a]	512	1	8	1
Register Reset (See next line for CPU_SCR0, CPU_SCR1)	All	All, except PPOR does not reset Bandgap Trim register	All	All
Reset Status Bits in CPU_SCR0, CPU_SCR1	Set PORS, Clear WDRS, Clear IRAMDIS	Set PORS, Clear WDRS, Clear IRAMDIS	Set PORS, Clear WDRS, Clear IRAMDIS	Clear PORS, Set WDRS, IRAMDIS unchanged
Bandgap Power	On	On	On	On
Boot Time[b]	2.0 ms Min 5.5 ms Max	2.0 ms Min 5.5 ms Max	2.0 ms Min 5.5 ms Max	2.0 ms Min 5.5 ms Max

a. CPU reset is released after synchronization with the CPU clock.
b. Measured from CPU reset release to execution of the code at Flash address 0x0000.

16.5 Power Modes

The ILO block drives the CLK32K clock used to time most events during the reset sequence. This clock is powered down by IPOR but not by any other reset. The sleep timer provides interval timing.

While POR or XRES assert, the IMO is powered off to reduce start-up power consumption.

During and following IRES (for 64 ms nominally), the IMO is powered off for low average power during slow supply ramps.

During and after POR or XRES, the bandgap circuit is powered up.

Following IRES, the bandgap circuit is only powered up occasionally to refresh the sampled bandgap voltage value. This sampling follows the same process used during sleep mode.

The IMO is always on for at least one CLK32K cycle before CPU reset is deasserted.

17. POR and LVD

This chapter briefly discusses the Power on Reset (POR) and Low Voltage Detect (LVD) circuits and their associated registers. For a complete table of the POR registers, refer to the “Summary Table of the System Resource Registers” on page 88. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

17.1 Architectural Description

The Power on Reset (POR) and Low Voltage Detect (LVD) circuits provide protection against low voltage conditions. The POR function senses Vdd and holds the system in reset until the magnitude of Vdd will support operation to specification. The LVD function senses Vdd and provides an interrupt to the system when Vdd falls below a selected threshold. Other outputs and status bits are provided to indicate important voltage trip levels. Refer to Section 16.2 Pin Behavior During Reset for a description of GPIO pin behavior during power up.

17.2 Register Definitions

The following registers are associated with the POR and LVD, and are listed in address order. The register descriptions below have an associated register table showing the bit structure. The bits that are grayed out in the register tables are reserved bits and are not detailed in the register descriptions that follow. Always write reserved bits with a value of ‘0’. For a complete table of the POR registers, refer to the “Summary Table of the System Resource Registers” on page 88.

17.2.1 VLT_CR Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,E3h	VLT_CR			PORLEV[1:0]		LVDTBEN	VM[2:0]			RW : 00

The Voltage Monitor Control Register (VLT_CR) is used to set the trip points for POR and LVD.

The VLT_CR register is cleared by all resets. This can cause reset cycling during very slow supply ramps to 5V when the POR range is set for the 5V range. This is because the reset clears the POR range setting back to 3V and a new boot/start-up occurs (possibly many times). The user can manage this with Sleep mode and/or reading voltage status bits if such cycling is an issue.

Bits 5 and 4: PORLEV[1:0]. These bits set the Vdd level at which PPOR switches to one of three valid values. Note that 11b is a reserved value and should not be used.

The three valid settings for these bits are:

- 00b (2.4V operation)
- 01b (2.7V operation)
- 10b (3.0V operation)

See the “DC POR and LVD Specifications” table in the Electrical Specifications section of the PSoC device data sheet for voltage tolerances for each setting.

Bit 3: LVDTBEN. This bit is AND’ed with LVD to produce a throttle-back signal that reduces CPU clock speed when low voltage conditions are detected. When the throttle-back signal is asserted, the CPU speed bits in the OSC_CR0 register are reset forcing the CPU speed to its reset state.

Bits 2 to 0: VM[2:0]. These bits set the Vdd level at which the LVD Comparator switches.

See the “DC POR and LVD Specifications” table in the Electrical Specifications section of the PSoC device data sheet for voltage tolerances for each setting.

For additional information, refer to the VLT_CR register on page 188.

17.2.2 VLT_CMP Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,E4h	VLT_CMP					NoWrite		LVD	PPOR	RW : 00

The Voltage Monitor Comparators Register (VLT_CMP) is used to read the state of internal supply voltage monitors.

Bit 3: NoWrite. This bit is only used in PSoC devices with a 2.4V minimum POR. It reads the state of the Flash write voltage monitor.

Bit 1: LVD. This bit reads the state of the low voltage detect comparator. The trip point for the LVD is set by VM[2:0] in the VLT_CR register.

Bit 0: PPOR. This bit reads back the state of the PPOR output. This can only be meaningfully read with PORLEV[1:0] set to disable PPOR. In that case, the PPOR status bit shows the comparator state directly.

For additional information, refer to the VLT_CMP register on page 189.

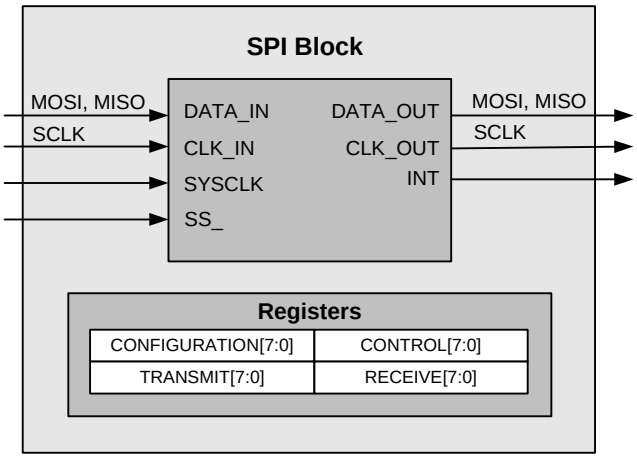
18. Serial Peripheral Interface

This chapter presents the Serial Peripheral Interconnect (SPI) and its associated registers. For a complete table of the SPI registers, refer to the "Summary Table of the System Resource Registers" on page 88. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

18.1 Architectural Description

The Serial Peripheral Interconnect (SPI) block is a dedicated master or slave SPI. The SPI slave function requires three inputs: Clock, Data, and SS_ (unless the SS_ is forced active with the SS_bit in the configuration register).

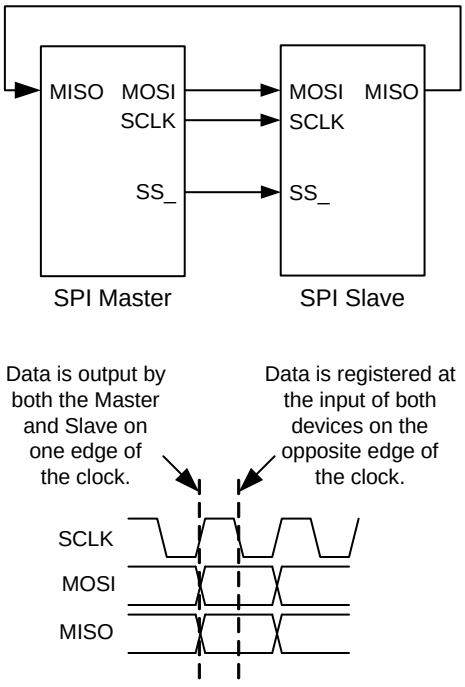
Figure 18-1. SPI Block Diagram



18.1.1 SPI Protocol Function

The SPI is a Motorola™ specification for implementing full-duplex synchronous serial communication between devices. The 3-wire protocol uses both edges of the clock to enable synchronous communication without the need for stringent setup and hold requirements. Figure 18-2 shows the basic signals in a simple connection.

Figure 18-2. Basic SPI Configuration



A device can be a master or slave. A master outputs clock and data to the ***slave device*** and inputs slave data. A slave device inputs clock and data from the master device and outputs data for input to the master. Together, the master and slave are essentially a circular shift register, where the master is generating the clocking and initiating data transfers.

A basic data transfer occurs when the master sends eight bits of data, along with eight clocks. In any transfer, both master and slave are transmitting and receiving simultaneously. If the master is only sending data, the received data from the slave is ignored. If the master wishes to receive data from the slave, the master must send dummy bytes to generate the clocking for the slave to send data back.

18.1.1.1 SPI Protocol Signal Definitions

The SPI protocol signal definitions are located in Table 18-1. The use of the SS_ signal varies according to the capability of the slave device.

Table 18-1. SPI Protocol Signal Definitions

Name	Function	Description
MOSI	Master Out Slave In	Master data output.
MISO	Master In Slave Out	Slave data output.
SCLK	Serial Clock	Clock generated by the master.
SS_	Slave Select (active low)	This signal is provided to enable multi-slave connections to the MISO pin. The MOSI and SCLK pins can be connected to multiple slaves, and the SS_ input selects which slave will receive the input data and drive the MISO line.

18.1.2 SPI Master Function

The SPI Master (SPIM) offers SPI operating modes 0-3. By default, the most significant bit (MSb) of the data byte is shifted out first. An additional option can be set to reverse the direction and shift the data byte out the least significant bit (LSb) first. (Refer to the timing diagrams for this function on page 122.)

The SPI protocol requires data to be registered at the device input, on the opposite edge of the clock that operates the output shifter.

The SPIM controls data transmission between master and slave because it generates the bit clock for internal clocking and for clocking the SPIS. The bit clock is derived from the CLK input selection.

There are four control bits and four status bits in the control register (SPI_CR) that provide for PSoC device interfacing and synchronization.

The SPIM hardware has no support for driving the Slave Select (SS_) signal. The behavior and use of this signal is application and PSoC device dependent and, if required, must be implemented in firmware.

18.1.2.1 Usability Exceptions

The following are usability exceptions for the SPI Protocol function.

1. The SPI_RXR (Rx Buffer) register is not writeable.
2. The SPI_TXR (Tx Buffer) register is not readable.

18.1.2.2 Block Interrupt

The SPIM block has a selection of two interrupt sources: interrupt on TX Reg Empty (default) or interrupt on SPI Complete. Mode bit 1 in the function register controls the selection. These modes are discussed in detail in "SPIM Timing" on page 122.

If SPI Complete is selected as the block interrupt, the control register must be read in the interrupt routine so that this status bit is cleared; otherwise, no subsequent interrupts are generated.

18.1.3 SPI Slave Function

The SPI Slave (SPIS) offers SPI operating modes 0-3. By default, the MSb of the data byte is shifted out first. An additional option can be set to reverse the direction and shift the data byte out LSb first. (Refer to the timing diagrams for this function on page 125.)

The SPI protocol requires data to be registered at the device input, on the opposite edge of the clock that operates the output shifter.

The SPIS function derives all clocking from the SCLK input (typically an external SPI Master). This means that the master must initiate all transmissions. For example, to read a byte from the SPIS, the master sends a byte.

There are four control bits and four status bits in the control register (SPI_CR) that provide for PSoC device interfacing and synchronization.

There is an additional data input in the SPIS, Slave Select (SS_), which is an active low signal. SS_ must be asserted to enable the SPIS to receive and transmit. SS_ has two high level functions: 1) To allow for the selection of a given slave in multi-slave environment, and 2) To provide additional clocking for TX data queuing in SPI modes 0 and 1.

SS_ may be controlled from an external pin or can be controlled by way of user firmware.

When SS_ is negated, the SPIS ignores any MOSI/SCLK input from the master. In addition, the SPIS state machine is reset and the MISO output is forced to idle at logic 1. This allows for a wired-AND connection in a multi-slave environment. Note that if High-Z output is required when the slave is not selected, this behavior must be implemented in firmware with IO writes to the port drive register.

18.1.3.1 Usability Exceptions

The following are usability exceptions for the SPI Slave function.

1. The SPI_RXR (Rx Buffer) register is not writeable.
2. The SPI_TXR (Tx Buffer) register is not readable.

18.1.3.2 Block Interrupt

The SPIS block has a selection of two interrupt sources: Interrupt on TX Reg Empty (default) or interrupt on SPI Complete (same selection as the SPIM). Mode bit 1 in the function register controls the selection.

If SPI Complete is selected as the block interrupt, the control register must still be read in the interrupt routine so that this status bit is cleared; otherwise, no subsequent interrupts are generated.

18.1.4 Input Synchronization

All pin inputs are double synchronized to SYSCLK by default. Synchronization can be bypassed by setting the BYPS bit in the SPI_CFG register.

18.2 Register Definitions

These registers are associated with the SPI and are listed in address order. The register descriptions have an associated register table showing the bit structure for that register. For a complete table of SPI registers, refer to the "Summary Table of the System Resource Registers" on page 88.

Data Registers

18.2.1 SPI_TXR Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,29h	SPI_TXR	Data[7:0]								W : 00

The SPI Transmit Data Register (SPI_TXR) is the SPI's transmit data register.

Bits 7 to 0: Data[7:0]. These bits encompass the SPI Transmit register. They are discussed by function type in Table 18-1 and Table 18-2.

For additional information, refer to the SPI_TXR register on page 140.

18.2.2 SPI_RXR Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,2Ah	SPI_RXR	Data[7:0]								R : 00

The SPI Receive Data Register (SPI_RXR) is the SPI's receive data register. A write to this register will clear the RX Reg Full status bit in the Control register (SPI_CR).

Bits 7 to 0: Data[7:0]. These bits encompass the SPI Receive register. They are discussed by function type in Table 18-1 and Table 18-2.

For additional information, refer to the SPI_RXR register on page 141.

18.2.2.1 SPI Master Data Register Definitions

There are two 8-bit Data registers and one 8-bit Control/Status register. This table explains the meaning of the Transmit and Receive registers in the context of SPIM operation.

Name	Function	Description
SPI_TXR	TX Buffer	Write only register. If no transmission is in progress and this register is written to, the data from this register is loaded into the Shift register, on the following clock edge, and a transmission is initiated. If a transmission is currently in progress, this register serves as a buffer for TX data. This register should only be written to when TX Reg Empty status is set and the write clears the TX Reg Empty status bit in the Control register. When the data is transferred from this register to the Shift register, then TX Reg Empty status is set.
SPI_RXR	RX Buffer	Read only register. When a byte transmission/reception is complete, the data in the shifter is transferred into the RX Buffer register and RX Reg Full status is set in the Control register. A read from this register clears the RX Reg Full status bit in the Control register.

18.2.2.2 SPI Slave Data Register Definitions

There are two 8-bit Data registers and one 8-bit Control/Status register. The table shown below explains the meaning of the Transmit and Receive registers in the context of SPIS operation.

Table 18-2. SPIS Data Register Descriptions

Name	Function	Description
SPI_TXR	TX Buffer	Write only register. This register should only be written to when TX Reg Empty status is set and the write clears the TX Reg Empty status bit in the Control register. When the data is transferred from this register to the Shift register, then TX Reg Empty status is set.
SPI_RXR	RX Buffer	Read only register. When a byte transmission/reception is complete, the data in the shifter is transferred into the RX Buffer register and RX Reg Full status is set in the Control register. A read from this register clears the RX Reg Full status bit in the Control register.

Control Register

18.2.3 SPI_CR Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,2Bh	SPI_CR	LSb First	Overrun	SPI Complete	TX Reg Empty	RX Reg Full	Clock Phase	Clock Polarity	Enable	# : 00

LEGEND
Access is bit specific. Refer to the register detail for additional information.

The SPI Control Register (SPI_CR) is the SPI's control register.

Bit 7: LSb First. This bit determines how the serial data is shifted out, either LSb or MSb first.

Bit 6: Overrun. This status bit indicates whether or not there was a Receive Buffer overrun. A read from the Receive buffer, after each received byte, must be performed before the reception of the next byte to avoid an overrun condition.

Bit 5: SPI Complete. This status bit indicates the completion of a transaction. A read from this register clears this bit.

Bit 4: TX Reg Empty. This status bit indicates whether or not the transmit register is empty.

Bit 3: RX Reg Full. This status bit indicates a receive register full condition.

Bit 2: Clock Phase. This bit determines which edge (rising and falling) that the data changes on.

Bit 1: Clock Polarity. This bit determines the logic level the clock codes to in its idle state.

Bit 0: Enable. This bit enables the SPI block.

For additional information, refer to the SPI_CR register on page 142.

18.2.3.1 SPI Control Register Definitions

Bit #	Name	Access	Description
7	LSb First	Read/Write	0 = Data shifted out MSb First. 1 = Data shifted out LSb First.
6	Overrun	Read Only	0 = No overrun. 1 = Indicates new byte received before previous one is read.
5	SPI Complete	Read Only	0 = Transaction in progress. 1 = Transaction is complete. Reading SPI_CR clears this bit.
4	TX Reg Empty	Read Only	0 = TX register is full. 1 = TX register is empty. Writing SPI_TXR register clears this bit.
3	TX Reg Full	Read Only	0 = RX register is not full. 1 = RX register is full. Reading SPI_RXR register clears this bit.
2	Clock Phase	Read/Write	0 = Data changes on trailing edge. 1 = Data changes on leading clock edge.
1	Clock Polarity	Read/Write	0 = Non-inverted, clock idles low (modes 0,2) 1 = Inverted, clock idles high (modes 1,3)
0	Enable	Read/Write	0 = Disable SPI function. 1 = Enable SPI function.

Configuration Register

The configuration block contains 1 register. Do not change this register while the block is enabled. Note that the SPI Configuration register is located in bank 1 of the PSoC device's memory map.

18.2.4 SPI_CFG Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
1,29h	SPI_CFG	Clock Sel[2:0]			Bypass	SS_	SS_EN_	Int Sel	Slave	RW : 00

The SPI Configuration Register (SPI_CFG) is used to configure the SPI.

Bits 7 to 5: Clock Sel. Clock Selection. These bits determine the operating frequency of the SPI Master.

Bit 4: Bypass. This bit determines whether or not the inputs are synchronized to SYSCLK.

Bit 3: SS_. Slave Select. This bit determines the logic value of the SS_ signal when the SS_EN_ signal is asserted (SS_EN_ = 0).

Bit 2: SS_EN_. Slave Select Enable. This active low bit determines if the slave select (SS_) signal is driven internally. If it is driven internally, its logic level is determined by the SS_ bit. If it is driven externally, its logic level is determined by the external pin.

Bit 1: Int Sel. Interrupt Select. This bit selects which condition produces an interrupt.

Bit 0: Slave. This bit determines whether the block functions as a master or slave.

For additional information, refer to the SPI_CFG register on page 182.

18.2.4.1 SPI Configuration Register Definitions

Bit #	Name	Access	Mode	Description
7:5	Clock Sel	Read/Write	Master	SYSCLK 000b / 2 001b / 4 010b / 8 011b / 16 100b / 32 101b / 64 110b / 128 111b / 256
4	Bypass	Read/Write	Master/Slave	0 = All pin unputs are doubled, synchronized 1 = Input synchronization is bypassed.
3	SS_	Read/Write	Slave	0 = Slave selected 1 = Slave selection is determined from external SS_ pin.
2	SS_EN_	Read/Write	Slave	0 = Slave selection determined from SS_ bit. 1 = Slave selection determined from external SS_ pin.
1	Int Sel	Read/Write	Master/Slave	0 = Interrupt on TX Reg Empty 1 = Interrupt on SPI Complete
0	Slave	Read/Write	Master/Slave	0 = Operates as a master. 1 = Operates as a slave.

18.3 Timing Diagrams

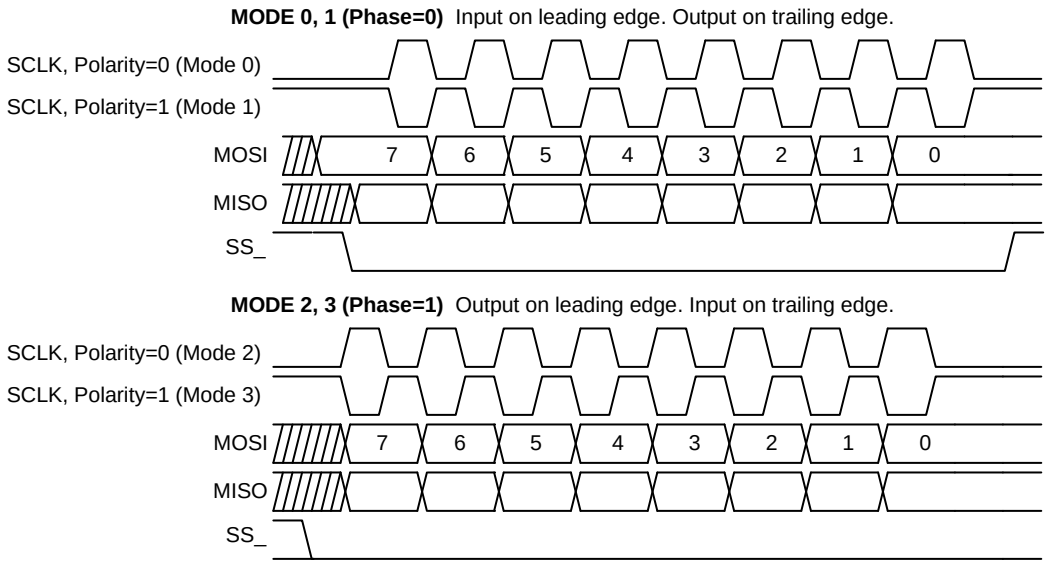
18.3.1 SPI Mode Timing

Figure 18-3 shows the SPI modes which are typically defined as 0,1, 2, or 3. These mode numbers are an encoding of two control bits: Clock Phase and Clock Polarity.

Clock phase indicates the relationship of the clock to the data. When the clock phase is '0', it means that the data is registered as an input on the leading edge of the clock and the next data is output on the trailing edge of the clock. When the clock phase is '1', it means that the next data is output on the leading edge of the clock and that data is registered as an input on the trailing edge of the clock.

Clock polarity controls clock inversion. When clock polarity is set to '1', the clock idle state is high.

Figure 18-3. SPI Mode Timing



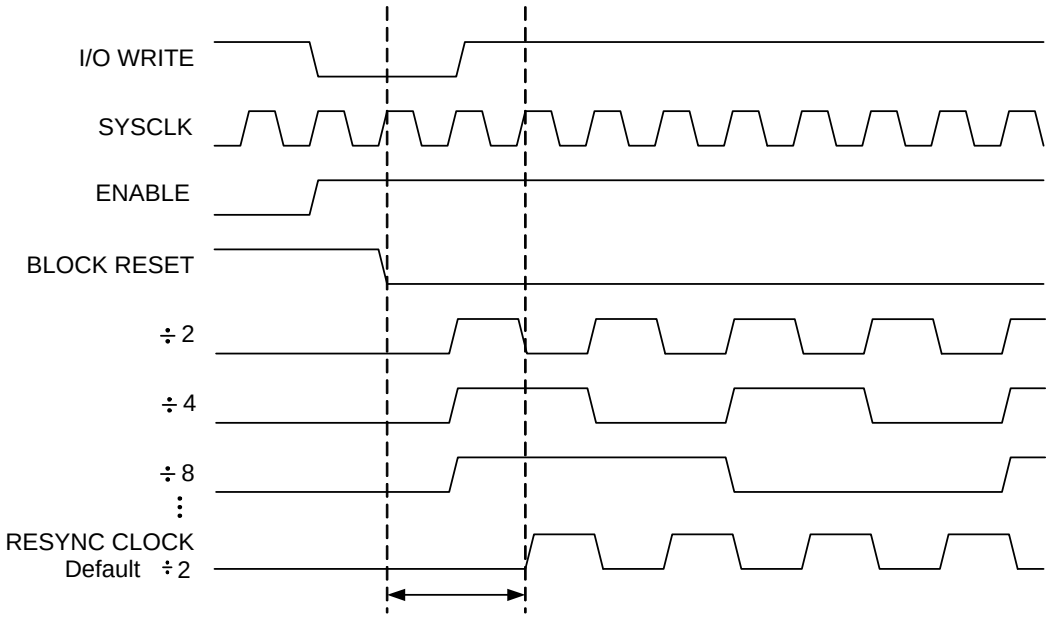
18.3.2 SPIM Timing

Enable/Disable Operation. As soon as the block is configured for SPIM, the primary output is the MSb or LSb of the Shift register, depending on the LSb First configuration in bit 7 of the Control register. The auxiliary output is '1' or '0', depending on the idle clock state of the SPI mode. This is the idle state.

Clock Generation. Figure 18-4 illustrates the SPIM input clocking scheme. The SYSCLK pin is an input into an eight-stage ripple divider that provides the baud rate selections. When the block is disabled, all internal state is held in a reset state. When the Enable bit in the SPI_CR register is set, the reset is synchronously released and the clock generation is enabled. All eight taps from the ripple divider are selectable (/2, /4, /8, /16, /32, /64, /128, /256) from the Clock Sel bits in the SPI_CFG register. The selected divider tap is resynchronized to SYSCLK. The resulting clock is routed to all of the synchronous elements in the design.

When the block is disabled, the SCLK and MOSI outputs revert to their idle state. All internal state is reset (including CR0 status) to its configuration specific reset state.

Figure 18-4. SPI Input Clocking



Normal Operation. Typical timing for a SPIM transfer is shown in Figure 18-5 and Figure 18-6. The user initially writes a byte to transmit when TX Reg Empty status is true. If no transmission is currently in progress, the data is loaded into the shifter and the transmission is initiated. The TX Reg Empty status is asserted again and the user is allowed to write the next byte to be transmitted to the TX Buffer register. After the last bit is output, if TX Buffer data is available with one-half clock setup time to the next clock, a new byte transmission will be initiated. A SPIM block receives a byte at the same time that it sends one. Use the SPI Complete or RX Reg Full to determine when the input byte was received.

Figure 18-5. Typical SPIM Timing in Mode 0 and 1

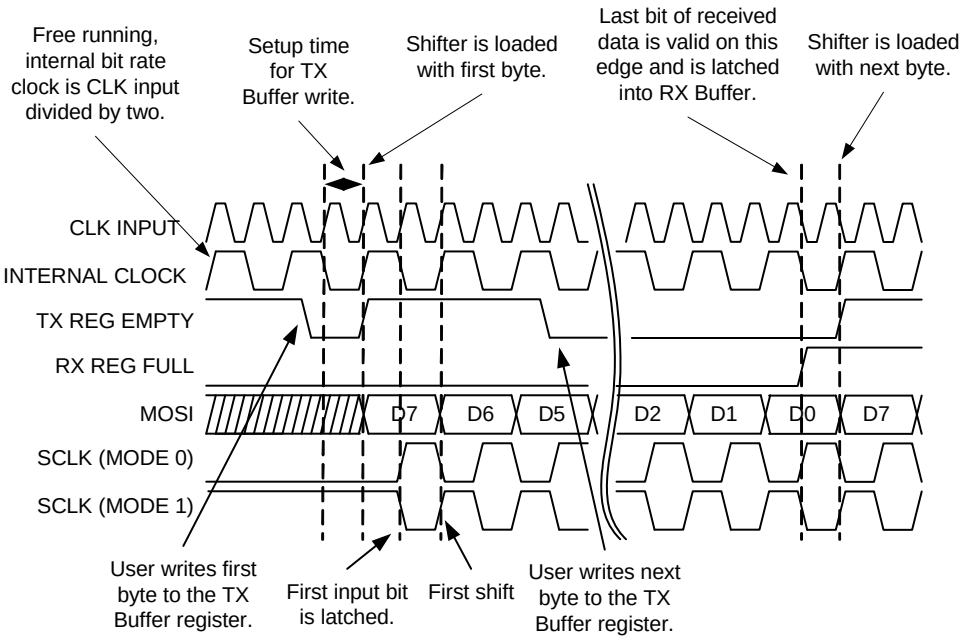
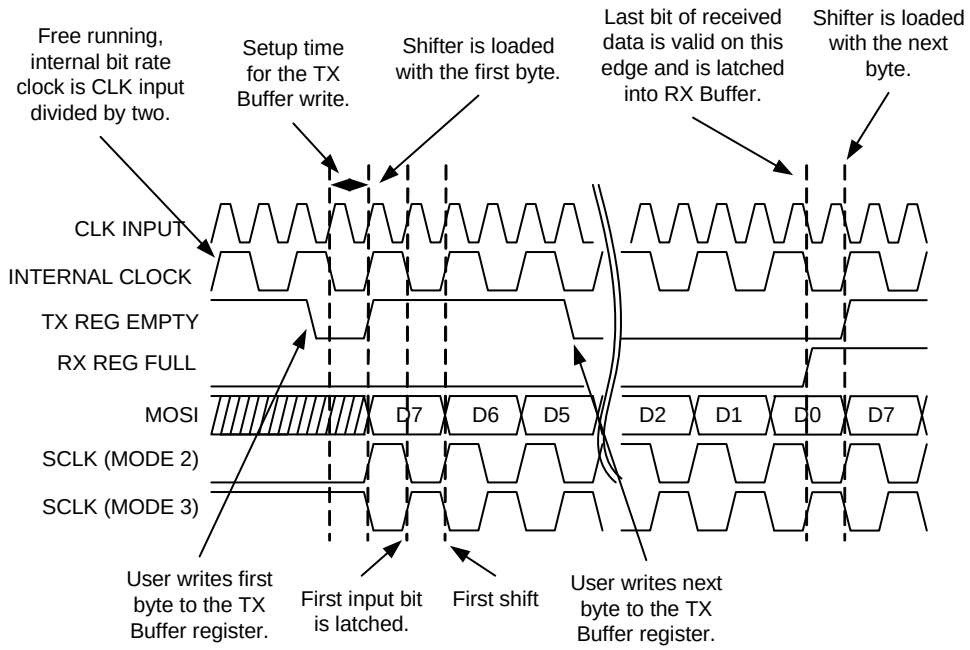


Figure 18-6. Typical SPIM Timing in Mode 2 and 3



Status Generation and Interrupts

There are four status bits in an SPI Block: TX Reg Empty, RX Reg Full, SPI Complete, and Overrun.

TX Reg Empty indicates that a new byte can be written to the TX Buffer register. When the block is enabled, this status bit is immediately asserted. This status bit is cleared when the user writes a byte of data to the TX Buffer register. TX Reg Empty is a control input to the state machine and, if a transmission is not already in progress, the assertion of this control signal initiates one. This is the default SPIM block interrupt. However, an initial interrupt is not generated when the block is enabled. The user must write a byte to the TX Buffer register and that byte must be loaded into the shifter before interrupts generated from the TX Reg Empty status bit are enabled.

RX Reg Full is asserted on the edge that captures the eighth bit of receive data. This status bit is cleared when the user reads the RX Buffer register (DR2).

SPI Complete is an optional interrupt and is generated when eight bits of data and clock have been sent. In modes 0 and 1, this occurs one-half cycle after RX Reg Full is set; because in these modes, data is latched on the leading edge of the clock and there is an additional one-half cycle remaining to complete that clock. In modes 2 and 3, this occurs at the same edge that the receive data is latched. This signal may be used to read the received byte or it may be used by the SPIM to disable the block after data transmission is complete.

Overrun status is set, if RX Reg Full is still asserted from a previous byte when a new byte is about to be loaded into the RX Buffer register. Because the RX Buffer register is implemented as a latch, Overrun status is set one-half bit clock before RX Reg Full status.

See Figure 18-7 and Figure 18-8 for status timing relationships.

Figure 18-7. SPI Status Timing for Modes 0 and 1

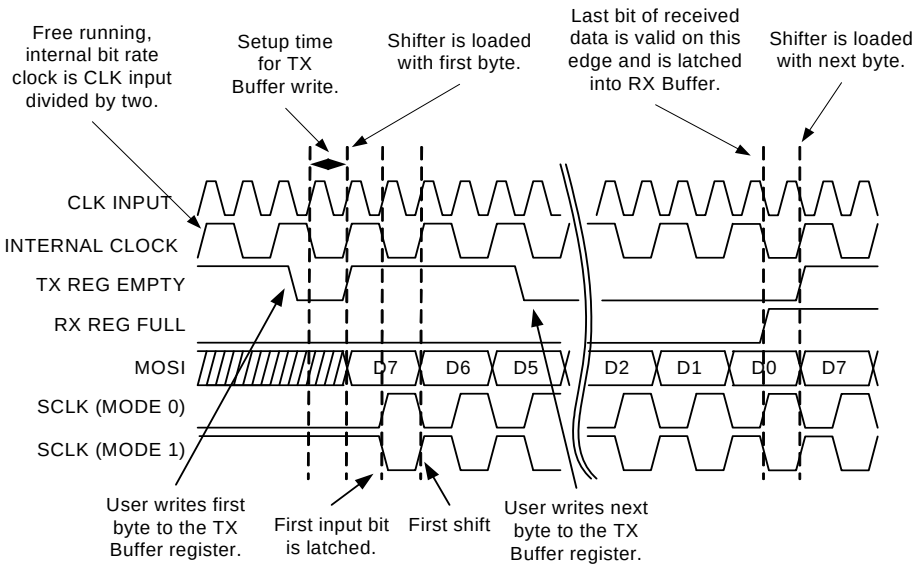
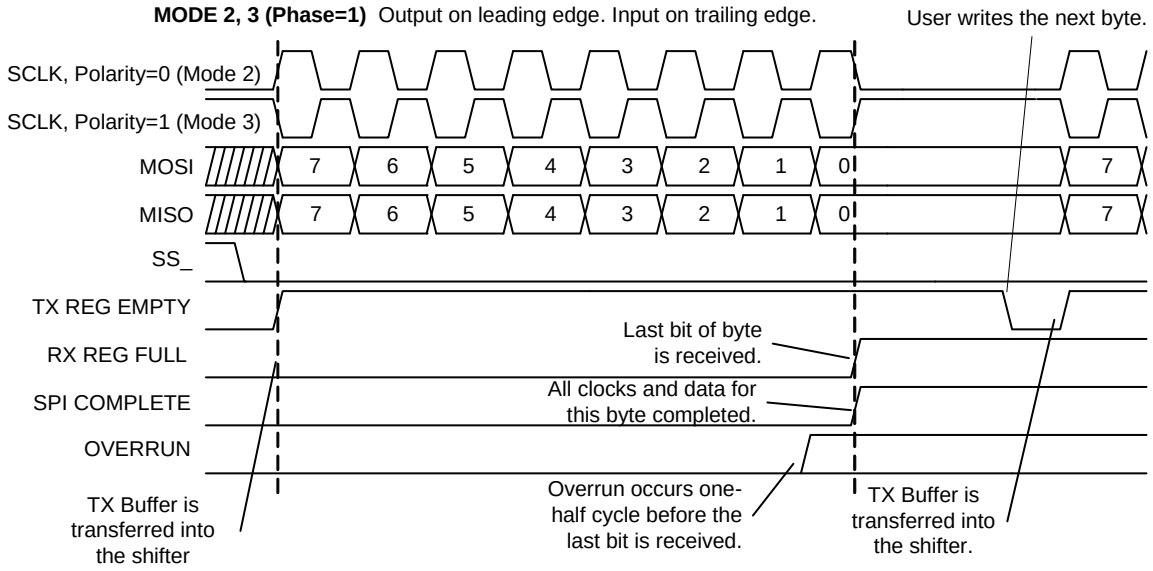


Figure 18-8. SPI Status Timing for Modes 2 and 3



18.3.3 SPIS Timing

Enable/Disable Operation. As soon as the block is configured for SPI Slave and before enabling, the MISO output is set to idle at logic 1. Both the enable bit must be set and the SS_ asserted (either driven externally or forced by firmware programming) for the block to output data. When enabled, the primary output is the MSb or LSb of the shift register, depending on the LSb First configuration in bit 7 of the Control register. The auxiliary output of the SPIS is always forced into tri-state.

Since the SPIS has no internal clock, it must be enabled with setup time to any external master supplying the clock. Setup time is also required for a TX Buffer register write, before the first edge of the clock or the first falling edge of SS_ depending on the mode. This setup time must be assured through the protocol and an understanding of the timing between the master and slave in a system.

When the block is disabled, the MISO output reverts to its idle '1' state. All internal state is reset (including CR0 status) to its configuration specific reset state.

Normal Operation. Typical timing for a SPIS transfer is shown in Figure 18-9 and Figure 18-10. If the SPIS is primarily used as a receiver, the RX Reg Full (polling only) or SPI Complete (polling or interrupt) status may be used to determine when a byte has been received. In this way, the SPIS operates identically with the SPIM. However, there are two main areas in which the SPIS operates differently: 1) SPIS behavior related to the SS_ signal, and 2) TX data queuing (loading the TX Buffer register).

Figure 18-9. Typical SPIS Timing in Modes 0 and 1

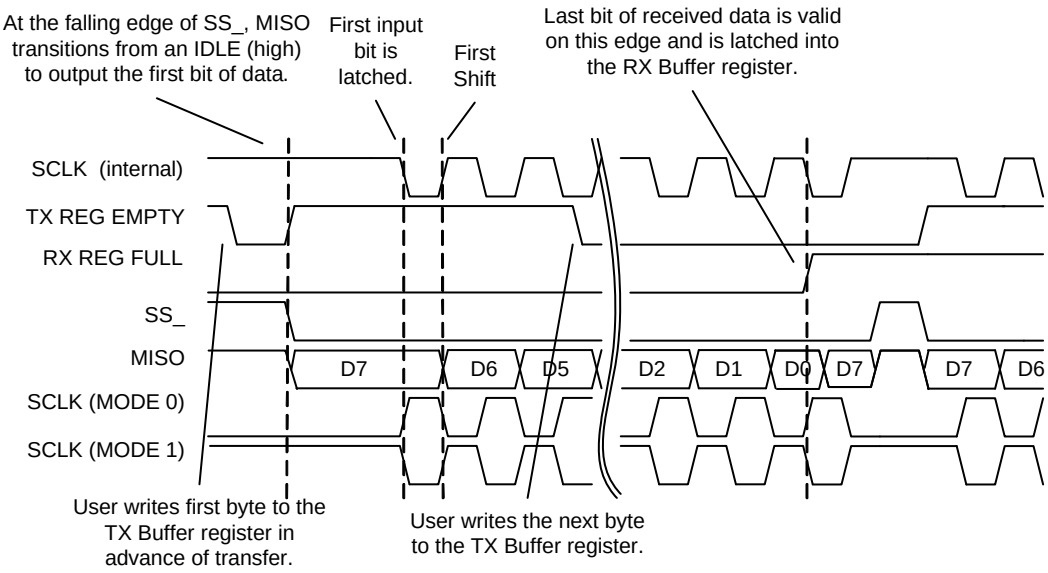
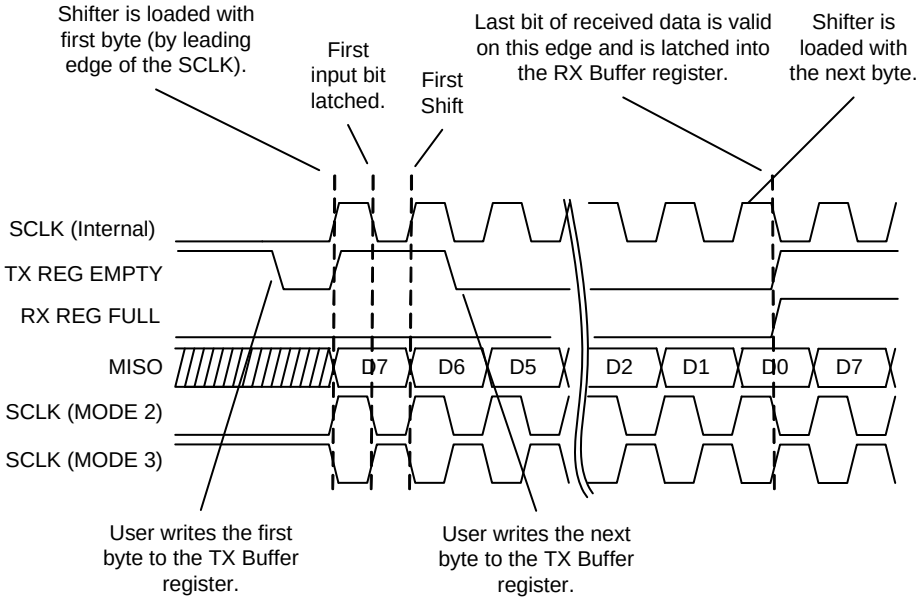


Figure 18-10. Typical SPIS Timing in Modes 2 and 3



Slave Select (SS_, active low). Slave Select must be asserted to enable the SPIS for receive and transmit. There are two ways to do this:

1. Drive the auxiliary input from a pin (selected by the Aux IO Select bits in the output register). This gives the SPI master control of the slave selection in a multi-slave environment.
2. SS_ may be controlled in firmware with register writes to the output register. When Aux IO Enable = 1, Aux IO Select bit 0 becomes the SS_ input. This allows the user to save an input pin in single slave environments.

When SS_ is negated (whether from an external or internal source), the SPIS state machine is reset and the MISO output is forced to idle at logic 1. In addition, the SPIS ignores any incoming MOSI/SCLK input from the master.

Status Generation and Interrupts. There are four status bits in the SPIS Block: TX Reg Empty, RX Reg Full, SPI Complete, and Overrun. The timing of these status bits are identical to the SPIM, with the exception of TX Reg Empty which is covered in the section on TX data queuing.

Status Clear On Read. Refer to the same subsection in "SPIM Timing" on page 122.

TX Data Queuing. Most SPI applications call for data to be sent back from the slave to the master. Writing firmware to accomplish this requires an understanding of how the Shift register is loaded from the TX Buffer register.

All modes use the following mechanism: 1) If there is no transfer in progress, 2) if the shifter is empty, and 3) if data is available in the TX Buffer register, the byte is loaded into the shifter.

The only difference between the modes is that the definition of "transfer in progress" is slightly different between modes 0 and 1, and modes 2 and 3.

Figure 18-11 illustrates TX data loading in modes 0 and 1. A transfer in progress is defined to be from the falling edge of SS_ to the point at which the RX Buffer register is loaded with the received byte. This means that in order to send a byte in the next transfer, it must be loaded into the TX Buffer register before the falling edge of SS_. This ensures a minimum setup time for the first bit, since the leading edge of the first SCLK must latch in the received data. If SS_ is not toggled between each byte or is forced low through the configuration register, the leading edge of SCLK is used to define the start of transfer. However, in this case, the user must provide the required setup time (one-half clock minimum before the leading edge) with a knowledge of system latencies and response times.

Figure 18-11. Mode 0 and 1 Transfer in Progress

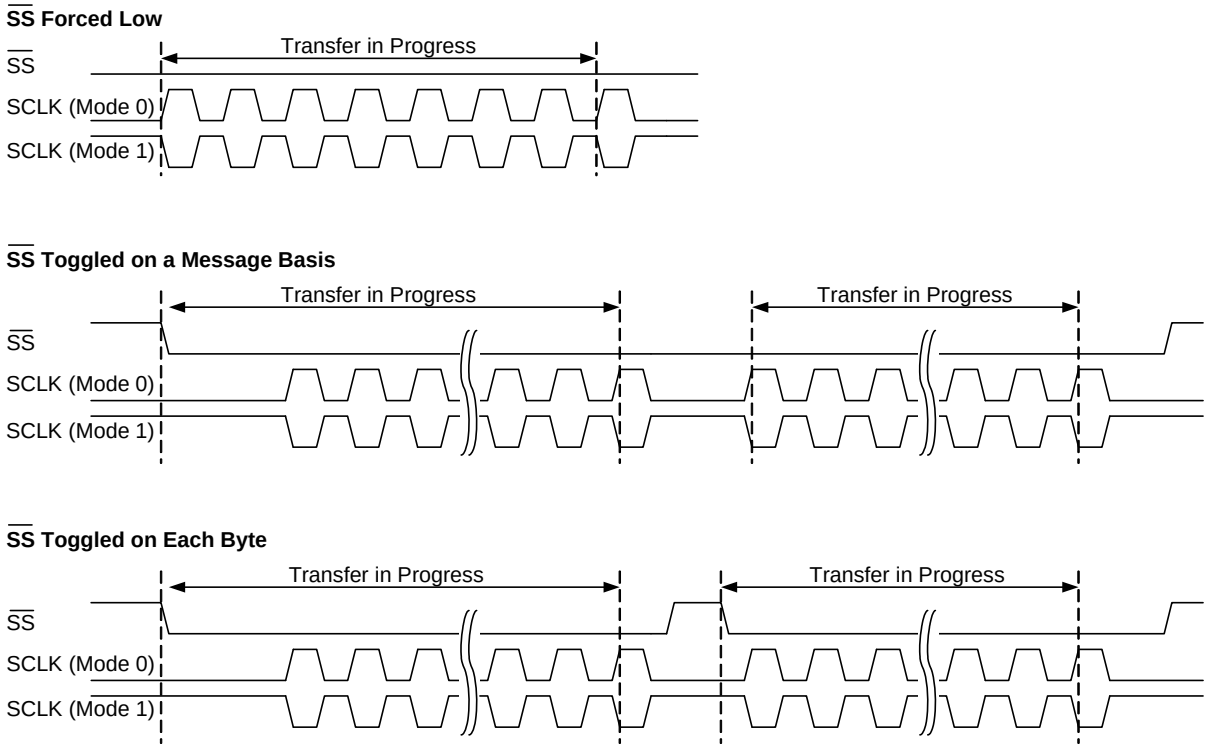
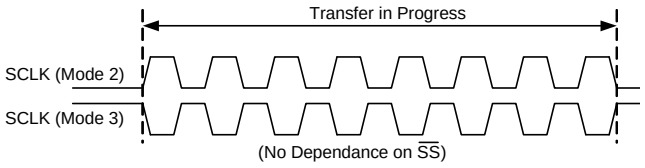


Figure 18-12 illustrates TX data loading in modes 2 and 3. In this case, a transfer in progress is defined to be from the leading edge of the first SCLK, to the point at which the RX Buffer register is loaded with the received byte. Loading the shifter by the leading edge of the clock has the effect of providing the required one-half clock setup time, as the data is latched into the receiver on the trailing edge of the SCLK in these modes.

Figure 18-12. Mode 2 and 3 Transfer in Progress



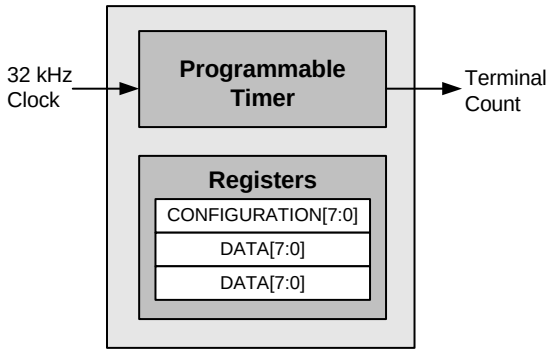
19. Programmable Timer

This chapter presents the Programmable Timer and its associated registers. For a complete table of the programmable timer registers, refer to the “Summary Table of the System Resource Registers” on page 88. For a quick reference of all PSoC registers in address order, refer to the Register Reference chapter on page 137.

19.1 Architectural Description

The programmable timer is a 13-bit down counter with a terminal count output. The timer has one configuration and two data registers associated with it. It is started by setting the START bit in its configuration register (PT_CFG). When started, the timer always starts counting down from the value loaded into its data registers (PT_DATA1, PT_DATA0). The timer has a one shot mode, in which the timer completes one full count cycle and stops. In one-shot mode the START bit in the configuration register is cleared after completion of one full count cycle. Setting the START bit will restart the timer.

Figure 19-1. Programmable Timer Block Diagram



19.1.1 Operation

When started, the programmable timer loads the value contained in its data registers and counts down to its terminal count of zero. The timer outputs an active high terminal count pulse for one clock cycle upon reaching the terminal count. The low time of the terminal count pulse is equal to the loaded decimal count value, multiplied by the clock period. (TC_{pw} = COUNT VALUE$_{decimal}$ * CLK_{period}). The period of the terminal count output is the pulse width of the terminal count, plus one clock period. (TC_{period} = TC_{pw} + CLK_{period}). Refer to Figure 19-2 and Figure 19-3.

Figure 19-2. Continuous Operation Example

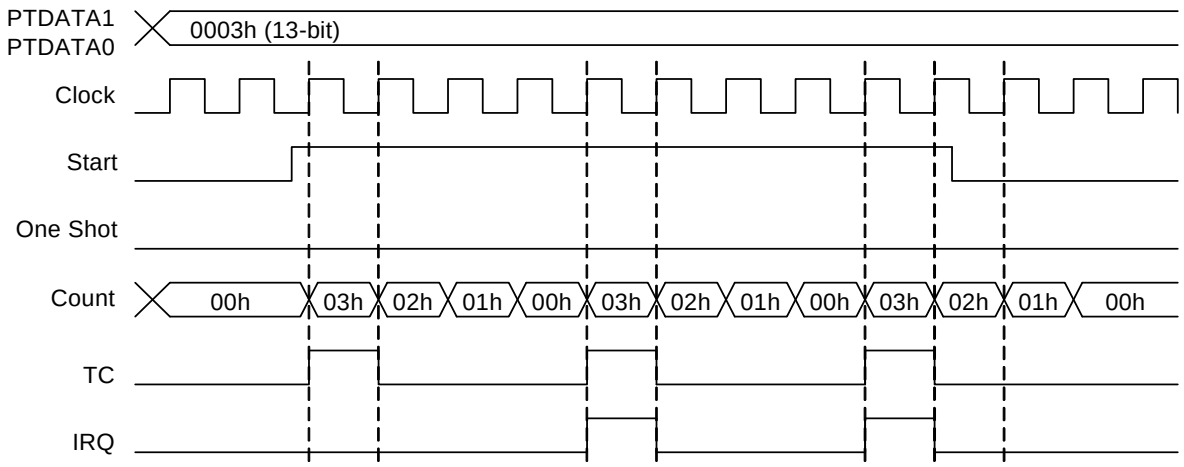
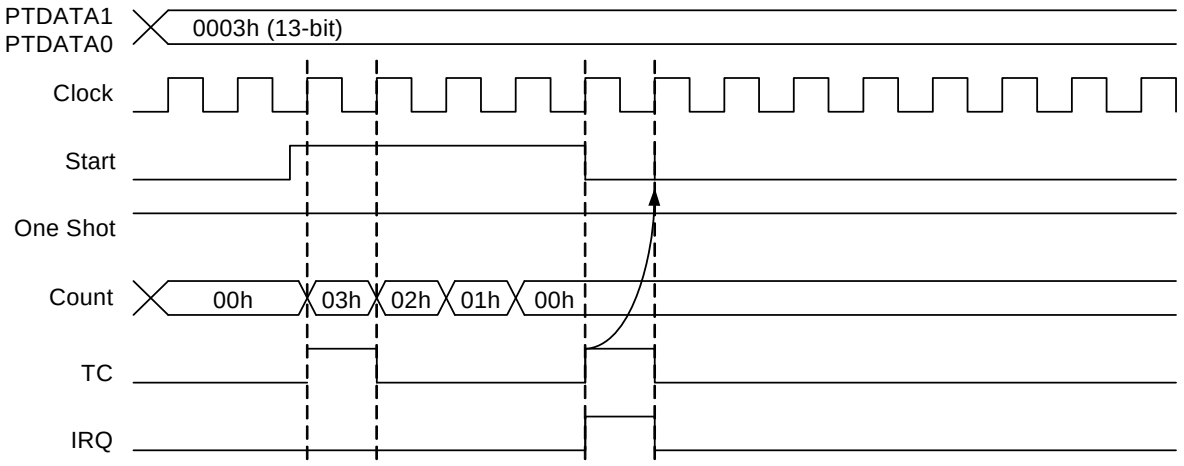


Figure 19-3. One Shot Operation Example



19.2 Register Definitions

The following registers are associated with the Programmable Timer and are listed in address order. The register descriptions have an associated register table showing the bit structure for that register. The bits in the tables that are grayed out are reserved bits and are not detailed in the register descriptions that follow. Always write reserved bits with a value of '0'. For a complete table of programmable timer registers, refer to the "Summary Table of the System Resource Registers" on page 88.

19.2.1 PT_CFG Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,B0h	PT_CFG							One Shot	START	RW : 00

The Programmable Timer Configuration Register (PT_CFG) configures the PSoC's programmable timer.

Bit 1: One Shot. This bit determines if the timer runs in one shot mode or continuous mode. In one-shot mode the timer completes one full count cycle and terminates. Upon termination, the START bit in this register is cleared. In continuous mode, the timer reloads the count value each time upon completion of its count cycle and repeats.

Bit 0: START. This bit starts the timer counting from a full count. The full count is determined by the value loaded into the DATA registers. This bit is cleared when the timer is running in one shot mode upon completion of a full count cycle.

For additional information, refer to the PT_CFG register on page 159.

19.2.2 PT_DATA1 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,B1h	PT_DATA1				Data[4:0]					RW : 00

The Programmable Timer Data Register 1 (PT_DATA1) holds the upper 5 bits of the progammable timer's count value.

Bits 4 to 0: Data[4:0]. These bits hold the upper 5 bits of the timer's 13-bit count value.

For additional information, refer to the PT_DATA1 register on page 160.

19.2.3 PT_DATA0 Register

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,B2h	PT_DATA0	Data[7:0]								RW : 00

The Programmable Timer Data Register 0 (PT_DATA0) holds the lower 8 bits of the progammable timer's count value.

Bits 7 to 0: Data[7:0]. These bits hold the lower 8 bits of the timer's13-bit count value.

For additional information, refer to the PT_DATA0 register on page 161.

Section E: Registers

The Registers section discusses the registers of the PSoC CY8C20x34/24 device. It lists all the registers in mapping tables, in address order. For easy reference, each register is linked to the page of a detailed description located in the next chapter. This section encompasses the following chapter:

- Register Reference on page 137.

Register General Conventions

The register conventions specific to this section and the Register Reference chapter are listed in the following table.

Register Conventions

Convention	Description
Empty, grayed-out table cell	Illustrates a reserved bit or group of bits.
‘x’ before the comma in an address	Indicates the register exists in register bank 1 and register bank 2.
‘x’ in a register name	Indicates that there are multiple instances/address ranges of the same register.
R	Read register or bit(s)
W	Write register or bit(s)
O	Only a read/write register or bit(s).
L	Logical register or bit(s)
C	Clearable register or bit(s)
#	Access is bit specific

Register Mapping Tables

The PSoC device has a total register address space of 512 bytes. The register space is also referred to as IO space and is broken into two parts: Bank 0 (user space) and Bank 1 (configuration space). The XIO bit in the Flag register (CPU_F) determines which bank the user is currently in. When the XIO bit is set, the user is said to be in the “extended” address space or the “configuration” registers.

Refer to the individual PSoC device data sheets for device-specific register mapping information.

Register Map Bank 0 Table: User Space

Name	Addr (0,Hex)	Access	Page	Name	Addr (0,Hex)	Access	Page	Name	Addr (0,Hex)	Access	Page	Name	Addr (0,Hex)	Access	Page
PRT0DR	00	RW	138		40				80				C0		
PRT0IE	01	RW	139		41				81				C1		
	02				42				82				C2		
	03				43				83				C3		
PRT1DR	04	RW	138		44				84				C4		
PRT1IE	05	RW	139		45				85				C5		
	06				46				86				C6		
	07				47				87				C7		
PRT2DR	08	RW	138		48				88				C8		
PRT2IE	09	RW	139		49				89				C9		
	0A				4A				8A				CA		
	0B				4B				8B				CB		
PRT3DR	0C	RW	138		4C				8C				CC		
PRT3IE	0D	RW	139		4D				8D				CD		
	0E				4E				8E				CE		
	0F				4F				8F				CF		
	10				50				90			CUR_PP	D0	RW	162
	11				51				91			STK_PP	D1	RW	163
	12				52				92				D2		
	13				53				93			IDX_PP	D3	RW	164
	14				54				94			MVR_PP	D4	RW	165
	15				55				95			MVW_PP	D5	RW	166
	16				56				96			I2C_CFG	D6	RW	167
	17				57				97			I2C_SCR	D7	#	168
	18				58				98			I2C_DR	D8	RW	169
	19				59				99				D9		
	1A				5A				9A			INT_CLR0	DA	RW	170
	1B				5B				9B				DB		
	1C				5C				9C				DC		
	1D				5D				9D				DD		
	1E				5E				9E				DE		
	1F				5F				9F				DF		
	20				60			CS_CR0	A0	RW	150	INT_MSK0	E0	RW	172
	21			AMUX_CFG	61	RW	143	CS_CR1	A1	RW	151	INT_SW_EN	E1	RW	173
	22				62			CS_CR2	A2	RW	152	INT_VC	E2	RC	174
	23				63			CS_CR3	A3	RW	153	RES_WDT	E3	W	175
	24				64			CS_CNTL	A4	RW	154		E4		
	25				65			CS_CNTH	A5	RW	155		E5		
	26				66			CS_STAT	A6	#	156		E6		
	27				67			CS_TIMER	A7	RW	157		E7		
	28				68			CS_SLEW	A8	RW	158		E8		
SPI_TXR	29	W	140		69				A9				E9		
SPI_RXR	2A	R	141		6A				AA				EA		
SPI_CR	2B	#	142		6B				AB				EB		
	2C				6C				AC				EC		
	2D				6D				AD				ED		
	2E				6E				AE				EE		
	2F				6F				AF				EF		
	30				70			PT_CFG	B0	RW	159		F0		
	31				71			PT_DATA1	B1	RW	160		F1		
	32				72			PT_DATA0	B2	RW	161		F2		
	33				73				B3				F3		
	34				74				B4				F4		
	35				75				B5				F5		
	36				76				B6				F6		
	37				77				B7			CPU_F	F7	RL	176
	38			CMP_RDC	78	#	144		B8				F8		
	39			CMP_MUX	79	RW	145		B9				F9		
	3A			CMP_CR0	7A	RW	146		BA				FA		
	3B			CMP_CR1	7B	RW	147		BB				FB		
	3C			CMP_LUT	7C	RW	149		BC				FC		
	3D				7D				BD			IDAC_D	FD	RW	177
	3E				7E				BE			CPU_SCR1	FE	#	178
	3F				7F				BF			CPU_SCR0	FF	#	179

Gray fields are reserved. # Access is bit specific.

Register Map Bank 1 Table: Configuration Space

Name	Addr (1,Hex)	Access	Page	Name	Addr (1,Hex)	Access	Page	Name	Addr (1,Hex)	Access	Page	Name	Addr (1,Hex)	Access	Page
PRT0DM0	00	RW	180		40				80				C0		
PRT0DM1	01	RW	181		41				81				C1		
	02				42				82				C2		
	03				43				83				C3		
PRT1DM0	04	RW	180		44				84				C4		
PRT1DM1	05	RW	181		45				85				C5		
	06				46				86				C6		
	07				47				87				C7		
PRT2DM0	08	RW	180		48				88				C8		
PRT2DM1	09	RW	181		49				89				C9		
	0A				4A				8A				CA		
	0B				4B				8B				CB		
PRT3DM0	0C	RW	180		4C				8C				CC		
PRT3DM1	0D	RW	181		4D				8D				CD		
	0E				4E				8E				CE		
	0F				4F				8F				CF		
	10				50				90				D0		
	11				51				91				D1		
	12				52				92				D2		
	13				53				93				D3		
	14				54				94				D4		
	15				55				95				D5		
	16				56				96				D6		
	17				57				97				D7		
	18				58				98			MUX_CR0	D8	RW	183
	19				59				99			MUX_CR1	D9	RW	183
	1A				5A				9A			MUX_CR2	DA	RW	183
	1B				5B				9B			MUX_CR3	DB	RW	183
	1C				5C				9C			IO_CFG	DC	RW	184
	1D				5D				9D			OUT_P1	DD	RW	185
	1E				5E				9E				DE		
	1F				5F				9F				DF		
	20				60				A0			OSC_CR0	E0	RW	186
	21				61				A1				E1		
	22				62				A2			OSC_CR2	E2	RW	187
	23				63				A3			VLT_CR	E3	RW	188
	24				64				A4			VLT_CMP	E4	R	189
	25				65				A5				E5		
	26				66				A6				E6		
	27				67				A7				E7		
	28				68				A8			IMO_TR	E8	W	190
SPI_CFG	29	RW	182		69				A9			ILO_TR	E9	W	191
	2A				6A				AA			BDG_TR	EA	RW	192
	2B				6B				AB			SLP_CFG	EB	RW	193
	2C				6C				AC				EC		
	2D				6D				AD				ED		
	2E				6E				AE				EE		
	2F				6F				AF				EF		
	30				70				B0				F0		
	31				71				B1				F1		
	32				72				B2				F2		
	33				73				B3				F3		
	34				74				B4				F4		
	35				75				B5				F5		
	36				76				B6				F6		
	37				77				B7			CPU_F	F7	RL	176
	38				78				B8				F8		
	39				79				B9				F9		
	3A				7A				BA				FA		
	3B				7B				BB				FB		
	3C				7C				BC				FC		
	3D				7D				BD				FD		
	3E				7E				BE			CPU_SCR1	FE	#	178
	3F				7F				BF			CPU_SCR0	FF	#	179

Gray fields are reserved. # Access is bit specific.

20. Register Reference

This chapter is a reference for all the PSoC device registers in address order, for Bank 0 and Bank 1. The most detailed descriptions of the PSoC registers are in the Register Definitions section of each chapter. The registers that are in both banks are incorporated with the Bank 0 registers, designated with an ‘x’, rather than a ‘0’ preceding the comma in the address. Bank 0 registers are listed first and begin on page 138. Bank 1 registers are listed second and begin on page 180. A condensed view of all the registers is shown in the register mapping tables starting on page 133.

20.1 Maneuvering Around the Registers

For ease-of-use, this chapter is formatted so that there is one register per page, although some registers use two pages. On each page, from top to bottom, there are four sections:

1. Register name and address (from lowest to highest).
2. Register table showing the bit organization, with reserved bits grayed out.
3. Written description of register specifics or links to additional register information.
4. Detailed register bit descriptions.

Use the register tables, in addition to the detailed register bit descriptions, to determine which bits are reserved. Reserved bits are grayed table cells and are not described in the bit description section. Reserved bits should always be written with a value of ‘0’. For all registers, an ‘x’ before the comma in the address field indicates that the register can be accessed or written to no matter what bank is used. For example, the M8C flag register’s (CPU_F) address is ‘x,F7h’ meaning it is located in bank 0 and bank 1 at F7h.

20.2 Register Conventions

This table lists the register conventions that are specific to this chapter.

Register Conventions

Convention	Example	Description
‘x’ in a register name	PRTxIE	Multiple instances/address ranges of the same register
R	R : 00	Read register or bit(s)
W	W : 00	Write register or bit(s)
O	RO : 00	Only a read/write register or bit(s).
L	RL : 00	Logical register or bit(s)
C	RC : 00	Clearable register or bit(s)
00	RW : 00	Reset value is 0x00 or 00h
XX	RW : XX	Register is not reset
0,	0,04h	Register is in bank 0
1,	1,23h	Register is in bank 1
x,	x,F7h	Register exists in register bank 0 and register bank 1
Empty, grayed-out table cell		Reserved bit or group of bits, unless otherwise stated

20.3 Bank 0 Registers

The following registers are all in bank 0 and are listed in address order. An ‘x’ before the comma in the register’s address indicates that the register can be accessed in Bank 0 and Bank 1, independent of the XIO bit in the CPU_F register. Registers that are in both Bank 0 and Bank 1 are listed in address order in Bank 0. For example, the CPU_F register has an address of X,F7h and is listed only in Bank 0 but is accessed in both Bank 0 and Bank 1.

20.3.1 PRTxDR

Port Data Register

Individual Register Names and Addresses:

PRT0DR : 0,00h PRT1DR : 0,04h PRT2DR : 0,08h PRT3DR : 0,0Ch

	7	6	5	4	3	2	1	0
Access : POR	RW : 00							
Bit Name	Data[7:0]							

This register allows for write or read access, of the current logical equivalent, of the voltage on the pin.

For PRT3DR, the upper ***nibble*** of this register will return the last data bus value when read and should be masked off prior to using this information.

For additional information, refer to the Register Definitions on page 55 in the GPIO chapter.

Bit	Name	Description
7:0	**Data[7:0]**	Write value to port or read value from port. Reads return the state of the pin, not the value in the PRTxDR register.

20.3.2 PRTxIE

Port Interrupt Enable Register

Individual Register Names and Addresses:

PRT0IE : 0,01h PRT1IE : 0,05h PRT2IE : 0,09h PRT3IE : 0,0Dh

	7	6	5	4	3	2	1	0
Access : POR	RW : 00							
Bit Name	Interrupt Enables[7:0]							

This register enables or disables interrupts from individual GPIO pins.

For PRT3DR, the upper nibble of this register returns the last data bus value when read and should be masked off prior to using this information.

For additional information, refer to the Register Definitions on page 55 in the GPIO chapter.

Bit	Name	Description
7:0	**Interrupt Enables[7:0]**	Bits enable the corresponding port pin interrupt. 0 Port pin interrupt disabled for the corresponding pin. 1 Port pin interrupt enabled for the corresponding pin. Interrupt mode is determined by the IOINT bit in the IO_CFG register.

20.3.3 SPI_TXR

SPI Transmit Data Register

Individual Register Names and Addresses:

SPI_TXR : 0,29h

	7	6	5	4	3	2	1	0
Access : POR	W : 00							
Bit Name	Data[7:0]							

This register is the SPI's transmit data register.

For additional information, refer to the Register Definitions on page 117 in the SPI chapter.

Bit	Name	Description
7:0	**Data[7:0]**	Data for selected function.

20.3.4 SPI_RXR

SPI Receive Data Register

Individual Register Names and Addresses:

SPI_RXR : 0,2Ah

	7	6	5	4	3	2	1	0
Access : POR	R : 00							
Bit Name	Data[7:0]							

This register is the SPI's receive data register.

For additional information, refer to the Register Definitions on page 117 in the SPI chapter.

Bit	Name	Description
7:0	**Data[7:0]**	Data for selected function.

20.3.5 SPI_CR

SPI Control Register

Individual Register Names and Addresses:

SPI_CR : 0,2Bh

	7	6	5	4	3	2	1	0
Access : POR	RW : 0	R : 0	R : 0	R : 1	R : 0	RW : 0	RW : 0	RW : 0
Bit Name	LSb First	Overrun	SPI Complete	TX Reg Empty	RX Reg Full	Clock Phase	Clock Polarity	Enable

This register is the SPI control register.

The LSb First, Clock Phase, and Clock Polarity bits are configuration bits and should never be changed once the block is enabled. These bits can be set at the same time that the block is enabled.

For additional information, refer to the Register Definitions on page 117 in the SPI chapter.

Bit	Name	Description
7	**LSb First**	Do not change this bit during an SPI transfer. 0 Data is shifted out MSb first. 1 Data is shifted out LSb first.
6	**Overrun**	0 No overrun has occurred. 1 Overrun has occurred. Indicates that a new byte is received and loaded into the RX Buffer before the previous one is read. It is cleared on a read of this (CR0) register.
5	**SPI Complete**	0 Indicates that a byte may still be in the process of shifting out, or no transmission is active. 1 Indicates that a byte is shifted out and all associated clocks are generated. It is cleared on a read of this (CR0) register. Optional interrupt.
4	**TX Reg Empty**	Reset state and the state when the block is disabled is '1'. 0 Indicates that a byte is currently buffered in the TX register. 1 Indicates that a byte is written to the TX register and cleared on write of the TX Buffer (DR1) register. This is the default interrupt. This status is initially asserted on block enable; however, the TX Reg Empty interrupt will occur only after the first data byte is written and transferred into the shifter.
3	**RX Reg Full**	0 RX register is empty. 1 A byte is received and loaded into the RX register. It is cleared on a read of the RX Buffer (DR2) register.
2	**Clock Phase**	0 Data is latched on the leading clock edge. Data changes on the trailing edge (modes 0, 1). 1 Data changes on the leading clock edge. Data is latched on the trailing edge (modes 2, 3).
1	**Clock Polarity**	0 Non-inverted, clock idles low (modes 0, 2). 1 Inverted, clock idles high (modes 1, 3).
0	**Enable**	0 SPI function is not enabled. 1 SPI function is enabled.

20.3.6 AMUX_CFG

Analog Mux Configuration Register

Individual Register Names and Addresses:

AMUX_CFG : 0,61h

	7	6	5	4	3	2	1	0
Access : POR					RW : 0		RW : 0	
Bit Name					ICAPEN[1:0]		INTCAP[1:0]	

This register configures the integration capacitor pin connections to the analog global bus.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 82 in the IO Analog Mux chapter.

Bits	Name	Description
3:2	**ICAPEN[1:0]**	Bits connect internal capacitance to the analog global bus. 00b No capacitance 01b Approximately 25 pF connected 10b Approximately 50 pF connected 11b Approximately 100 pF connected
1:0	**INTCAP[1:0]**	Select pins to enable connection of external integration capacitor in the charge integration mode. 00b Neither P0[3] or P0[1] enabled 01b P0[1] pin enabled 10b P0[3] pin enabled 11b Both P0[3] and P0[1] pins enabled

20.3.7 CMP_RDC

Comparator Read/Clear Register

Individual Register Names and Addresses:

CMP_RDC : 0,78h

	7	6	5	4	3	2	1	0
Access : POR			R : 0	R : 0			RC : 0	RC : 0
Bit Name			CMP1D	CMP0D			CMP1L	CMP0L

This register is used to read the state of the comparator data signal, and the latched state of the comparator.

In the table above, reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 84 in the Comparators chapter.

Bit	Name	Description
5	**CMP1D**	Read-only bit that returns the dynamically changing state of comparator 1. This bit reads zero whenever the comparator is disabled.
4	**CMP0D**	Read-only bit that returns the dynamically changing state of comparator 0. This bit reads zero whenever the comparator is disabled.
1	**CMP1L**	Bit reads the latch output for comparator 1. This bit is cleared by either a write of '0' to this bit, or by a rising edge of the comparator 0 LUT, depending on the state of the CRST1 bit in the CMP_CR1 register.
0	**CMP0L**	Bit reads the latch output for comparator 0. This bit is cleared by either a write of '0' to this bit, or by a rising edge of the comparator 1 LUT, depending upon the state of the CRST0 bit in the CMP_CR1 register.

20.3.8 CMP_MUX

Comparator Multiplexer Register

Individual Register Names and Addresses:

CMP_MUX : 0,79h

	7	6	5	4	3	2	1	0
Access : POR	RW : 0		RW : 0		RW : 0		RW : 0	
Bit Name	INP1[1:0]		INN1[1:0]		INP0[1:0]		INN0[1:0]	

This register contains control bits for input selection of comparators 0 and 1.

For additional information, refer to the Register Definitions on page 84 in the Comparators chapter.

Bit	Name	Description
7:6	**INP1[1:0]**	Comparator 1 Positive Input Select 00b Analog Global Mux Bus 01b Reserved 10b P0[1] pin 11b P0[3] pin
5:4	**INN1[1:0]**	Comparator 1 Negative Input Select 00b VREF (1.3V) 01b Ref Lo (approximately 0.9V) 10b Ref Hi (approximately 1.8V) 11b Reserved
3:2	**INP0[1:0]**	Comparator 0 Positive Input Select 00b Analog Global Mux Bus 01b Reserved 10b P0[1] pin 11b P0[3] pin
1:0	**INN0[1:0]**	Comparator 0 Negative Input Select 00b VREF (1.3V) 01b Ref Lo (approximately 0.9V) 10b Ref Hi (approximately 1.8V) 11b Reserved

20.3.9 CMP_CR0

Comparator Control Register 0

Individual Register Names and Addresses:

CMP_CR0 : 0,7Ah

	7	6	5	4	3	2	1	0
Access : POR			RW : 0	RW : 0			RW : 0	RW : 0
Bit Name			CMP1R	CMP1EN			CMP0R	CMP0EN

This register enables and configures the input range of the comparators.

In the table above, reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 84 in the Comparators chapter.

Bit	Name		Description
5	**CMP1R**	0	Comparator 1 set to rail-to-rail input range, with approximately 20 µA cell current.
		1	Comparator 1 set to limited input range (Vss to Vdd - 1V), with approximately 10 µA cell current.
4	**CMP1EN**	0	Comparartor 1 disabled, powered off.
		1	Comparartor 1 enabled.
1	**CMP0R**	0	Comparator 0 set to rail-to-rail input range, with approximately 20 µA cell current.
		1	Comparator 0 set to limited input range (Vss to Vdd - 1V), with approximately 10 µA cell current.
0	**CMP0EN**	0	Comparartor 0 disabled, powered off.
		1	Comparartor 0 enabled.

20.3.10 CMP_CR1

Comparator Control Register 1

Individual Register Names and Addresses:

CMP_CR1 : 0,7Bh

	7	6	5	4	3	2	1	0
Access : POR	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0
Bit Name	CINT1	CPIN1	CRST1	CDS1	CINT0	CPIN0	CRST0	CDS0

This register is used to configure the comparator output options.

For additional information, refer to the Register Definitions on page 84 in the Comparators chapter.

Bit	Name	Description
7	**CINT1**	This bit selects comparator 1 for input to the analog interrupt. Note that if both CINT1 and CINT0 are set high, a rising edge on either comparator output may cause an interrupt. 0 Comparator 1 does not connect to the analog interrupt. 1 Comparartor 1 connects to the analog interrupt. A rising edge will assert that interrupt, if it is enabled in the INT_MSK0 register.
6	**CPIN1**	This bit selects the Comparator 1 signal for possible connection to the GPIO pin. Connection to the pin also depends on the configuration of the OUT_P1 register. 0 Select Comparator 1 LUT output 1 Select Comparator 1 Latch output
5	**CRST1**	This bit selects the source for resetting the Comparator 1 latch. 0 Reset by writing a ‘0’ to the CMP_RDC register’s CMP1L bit 1 Reset by rising edge of Comparator 0 LUT output
4	**CDS1**	This bit selects the data output for the comparator 1 channel, for routing to the capacitive sense logic and comparator 1 interrupt. 0 Select the Comparator 1 LUT output 1 Select the Comparator 1 latch output
3	**CINT0**	This bit selects comparator 0 for input to the analog interrupt. Note that if both CINT1 and CINT0 are set high, a rising edge on either comparator output may cause an interrupt. 0 Comparator 0 does not connect to the analog interrupt. 1 Comparartor 0 connects to the analog interrupt. A rising edge will assert that interrupt, if it is enabled in the INT_MSK0 register.
2	**CPIN0**	This bit selects the Comparator 0 signal for possible connection to the GPIO pin. Connection to the pin also depends on the configuration of the OUT_P1 register. 0 Select Comparator 0 LUT output 1 Select Comparator 0 Latch output
1	**CRST0**	This bit selects the source for resetting the Comparator 0 latch. 0 Reset by writing a ‘0’ to the CMP_RDC register’s CMP0L bit 1 Reset by rising edge of Comparator 1 LUT output

(continued on next page)

20.3.10 CMP_CR1 *(continued)*

0	**CDS0**	This bit selects the data output for the comparator 0 channel, for routing to the capacitive sense logic and comparator 0 interrupt. 0 Select the Comparator 0 LUT output 1 Select the Comparator 0 latch output

20.3.11 CMP_LUT

Comparator LUT Register

Individual Register Names and Addresses:

CMP_LUT: 0,7Ch

	7	6	5	4	3	2	1	0
Access : POR	RW : 0				RW : 0			
Bit Name	LUT1[3:0]				LUT0[3:0]			

This register selects the logic function.

For additional information, refer to the Register Definitions on page 84 in the Comparators chapter.

Bits	Name	Description
7:4	**LUT1[3:0]**	Select 1 of 16 logic functions for output of comparator bus 1. A = Comp1 output, B = Comp0 output. **Function** 0h FALSE 1h A AND $\overline{B}$ 2h A AND $\overline{B}$ 3h $\overline{A}$ 4h $\overline{A}$ AND B 5h B 6h A XOR B 7h A OR B 8h A NOR B 9h $\overline{A}$ XNOR B Ah $\overline{B}$ Bh $\overline{A}$ OR $\overline{B}$ Ch $\overline{A}$ Dh $\overline{A}$ OR B Eh A NAND B Fh TRUE
3:0	**LUT0[3:0]**	Select 1 of 16 logic functions for output of comparator bus 0. A = Comp0 output, B = Comp1 output. **Function** 0h FALSE 1h A AND $\overline{B}$ 2h A AND $\overline{B}$ 3h $\overline{A}$ 4h $\overline{A}$ AND B 5h B 6h A XOR B 7h A OR B 8h A NOR B 9h $\overline{A}$ XNOR B Ah $\overline{B}$ Bh $\overline{A}$ OR $\overline{B}$ Ch $\overline{A}$ Dh $\overline{A}$ OR B Eh A NAND B Fh TRUE

20.3.12 CS_CR0

CapSense Control Register 0

Individual Register Names and Addresses:

CS_CR0 : 0,A0h

	7	6	5	4	3	2	1	0
Access : POR	RW : 0					RW : 0		RW : 0
Bit Name	CSOUT[1:0]					MODE[1:0]		EN

This register controls the operation of the CapSense counters.

Never write to bits [7:1] while the block is enabled.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 74 in the CapSense Module chapter.

Bit	Name	Description
7:6	**CSOUT[1:0]**	CapSense Output 00b Selected Input 01b CapSense Interrupt 10b Carry Out Low Byte 11b Carry Out High Byte
2:1	**MODE[1:0]**	CapSense Counter Mode 00b Event mode. Start in Enable, stop on interrupt event. 01b Pulse Width mode. Start on positive edge of next input. Stop on negative edge of input. 10b Period mode. Start on positive edge of input. Stop on next positive edge of input. 11b Start in Enable, continuous operation until disable.
0	**EN**	0 Counting is stopped and all counter values are reset to '0'. 1 Counters are enabled for counting.

20.3.13 CS_CR1

CapSense Control Register 1

Individual Register Names and Addresses:

CS_CR1 : 0,A1h

	7	6	5	4	3	2	1	0
Access : POR	RW : 0	RW : 0		RW : 0	RW : 0	RW : 0		
Bit Name	CHAIN	CLKSEL[1:0]		RLOSEL	INV	INSEL[2:0]		

This register contains additional CapSense system control options.

Never write to this register while the block is enabled.

For additional information, refer to the Register Definitions on page 74 in the CapSense Module chapter.

Bit	Name	Description
7	**CHAIN**	Counter Chain Control 0 8-bit high/low counters operate independently 1 High/low counters operate as a 16-bit synchronous block
6:5	**CLKSEL[1:0]**	CapSense Clock (CSCLK) Selection 00b IMO 01b IMO/2 10b IMO/4 11b IMO/8
4	**RLOSEL**	Relaxation Oscillator Clock (RLO) Select 0 High byte counter runs on the selected IMO-based frequency. 1 High byte counter runs on the RLO clock frequency.
3	**INV**	Input Invert 0 Selected input is not inverted. 1 Selected input is inverted.
2:0	**INSEL[2:0]**	Input Selection 000b Comparator 0 001b ILO 010b Comparator 1 011b RLO Timer Terminal Count 100b Interval Timer 101b RLO Timer IRQ 110b Analog Global Mux Bus 111b ‘0’

20.3.14 CS_CR2

CapSense Control Register 2

Individual Register Names and Addresses:

CS_CR2 : 0,A2h

	7	6	5	4	3	2	1	0
Access : POR	RW : 0		RW : 0	RW : 0		RW : 0		RW : 0
Bit Name	IRANGE[1:0]		IDACDIR	IDAC_EN		PXD_EN		RO_EN

This register contains additional CapSense system control options.

For additional information, refer to the Register Definitions on page 74 in the CapSense Module chapter.

Bit	Name	Description
7:6	**IRANGE**[1:0]	Bits scale the IDAC current output. The IDAC_D register sets the base current in the IDAC. 00 IDAC output scaled to 1X range. 01 IDAC output scaled to 2X range. 10 IDAC output scaled to 4X range. 11 IDAC output scaled to 8X range.
5	**IDACDIR**	Bit determines the source/sink state of the IDAC when enabled (IDAC_EN = 1 or PXD_EN = 1). 0 IDAC sources current to analog global bus. 1 IDAC sinks current from analog global bus.
4	**IDAC_EN**	Bit provides manual connection of the IDAC to the analog global bus. The IDAC is automatically connected when RO_EN = 1 or PXD_EN = 1. 0 No manual connection 1 IDAC is connected to analog global bus.
2	**PXD_EN**	0 No clock to I/O pins 1 Enabled pins switch between ground and the analog global bus. Clock rate selected by the CLKSEL bits in the CS_CR1 register. Selected clock drives CapSense timer.
0	**RO_EN**	0 Relaxation oscillator disabled. 1 Relaxation oscillator enabled. Charging currents are set by the IRANGE bits and the IDAC_D register value.

20.3.15 CS_CR3

CapSense Control Register 3

Individual Register Names and Addresses:

CS_CR3 : 0,A3h

	7	6	5	4	3	2	1	0
Access : POR	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0		RW : 0	
Bit Name	IBOOST	REFMUX	REFMODE	REF_EN	LPFilt[1:0]		LPF_EN[1:0]	

This register contains control bits primarily for the proximity detection algorithm.

For additional information, refer to the Register Definitions on page 74 in the CapSense Module chapter.

Bit	Name	Description
7	**IBOOST**	This bit adds a fixed boost current to the IDAC output. This affects all functions using the IDAC, such as the relaxation oscillator. The boost size is approximately equal to 200 (decimal) counts of the IDAC_D register. 0 No boost current 1 Boost current is added to IDAC output.
6	**REFMUX**	This bit selects the reference voltage for the input of the reference buffer. 0 Select REFHI (1.8V) 1 Select VREF (1.3V)
5	**REFMODE**	This bit allows manual connection of the reference buffer output to the analog global bus. If either CI_EN=1 or RO_EN=1 in the CS_CR2 register, this bit has no effect (reference buffer connection is off or controlled by other settings). 0 No connection 1 Reference buffer connected to the analog global bus.
4	**REF_EN**	This bit enables the reference buffer to drive the analog global bus. 0 Reference buffer is disabled, powered down. 1 Reference buffer is enabled. Connection to the analog global bus is controlled by the REFMODE bit in this register, and by the CI_EN and RO_EN bits in the CS_CR2 register.
3:2	**LPFilt[1:0]**	Low pass filter approximate time constant. 00b 1 µs 01b 2 µs 10b 5 µs 11b 10 µs
1:0	**LPF_EN[1:0]**	These bits enable the low pass filter. 00b No connection of either comparator channel to low pass filter 01b Connect comparator channel 0 through the low pass filter 10b Connect comparator channel 1 through the low pass filter 11b Connect both comparator channel inputs together, and through the low pass filter

20.3.16 CS_CNTL

CapSense Counter Low Byte Register

Individual Register Names and Addresses:

CS_CNTL : 0,A4h

	7	6	5	4	3	2	1	0
Access : POR	R : 00							
Bit Name	Data[7:0]							

This register contains the current count for the low byte counter.

For additional information, refer to the Register Definitions on page 74 in the CapSense Module chapter.

Bit	Name	Description
7:0	**Data[7:0]**	On a read of this register, the current count is returned. It is only read when the counter is stopped. **Note** The counter must be stopped by the configured event. When the counter is disabled, the count is reset to 00h.

20.3.17 CS_CNTH

CapSense Counter High Byte Register

Individual Register Names and Addresses:

CS_CNTH : 0,A5h

	7	6	5	4	3	2	1	0
Access : POR	R : 00							
Bit Name	Data[7:0]							

This register contains the current count value for the high byte counter.

For additional information, refer to the Register Definitions on page 74 in the CapSense Module chapter.

Bit	Name	Description
7:0	**Data[7:0]**	On a read of this register, the current count is returned. It is only read when the counter is stopped. **Note** The counter must be stopped by the configured event. When the counter is disabled, the count is reset to 00h.

20.3.18 CS_STAT

CapSense Status Register

Individual Register Names and Addresses:

CS_STAT : 0,A6h

	7	6	5	4	3	2	1	0
Access : POR	RC : 0	RC : 0	RC : 0	RC : 0	RW : 0	RW : 0	RW : 0	RW : 0
Bit Name	INS	COLS	COHS	PPS	INM	COLM	COHM	PPM

This register controls the CapSense counter options.

Never modify the interrupt mask bits while the block is enabled. If modification to bits 3 to 0 is necessary while the block is enabled, then pay attention to ensure that the status bits, bits 7 to 4, are not accidentally cleared. This is done by writing a '1' to all of the status bits when writing to the mask bits.

For additional information, refer to the Register Definitions on page 74 in the CapSense Module chapter.

Bit	Name	Description
7	**INS**	Input Status 0 No event detected 1 A rising edge on the selected input was detected. Cleared by writing a '0' to this bit.
6	**COLS**	Counter Carry Out Low Status 0 No event detected 1 A carry out from the low byte counter was detected. Cleared by writing a '0' back to this bit.
5	**COHS**	Counter Carry Out High Status 0 No event detected 1 A carry out from the high byte counter was detected. Cleared by writing a '0' back to this bit.
4	**PPS**	Pulse Width/Period Measurement Status 0 No event detected 1 A pulse width or period measurement was completed. Cleared by writing a '0' back to this bit.
3	**INM**	Input Interrupt/Mask 0 Disabled 1 Input event is enabled to assert the block interrupt.
2	**COLM**	Counter Carry Out Low Interrupt Mask 0 Disabled 1 Counter carry out low is enabled to assert the block interrupt.
1	**COHM**	Counter Carry Out High Interrupt Mask 0 Disabled 1 Counter carry out high is enabled to assert the block interrupt.
0	**PPM**	Pulse Width/Period Measurement Interrupt Mask 0 Disabled 1 Completion of a pulse width or period measurement is enabled to assert the block interrupt.

20.3.19 CS_TIMER

CapSense Timer Register

Individual Register Names and Addresses:

CS_TIMER : 0,A7h

	7	6	5	4	3	2	1	0
Access : POR			RW : 00					
Bit Name			Timer Count Value[5:0]					

This register sets the timer count value.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of ‘0’. For additional information, refer to the Register Definitions on page 74 in the CapSense Module chapter.

Bit	Name	Description
5:0	**Timer Count Value[5:0]**	Holds the timer count value.

20.3.20 CS_SLEW

CapSense Slew Control Register

Individual Register Names and Addresses:

CS_SLEW : 0,A8h

	7	6	5	4	3	2	1	0
Access : POR	RW : 0							RW : 0
Bit Name	FastSlew[6:0]							FS_EN

This register enables and controls a fast slewing mode for the relaxation oscillator.

For additional information, refer to the Register Definitions on page 74 in the CapSense Module chapter.

Bit	Name	Description
7:1	**FastSlew[6:0]**	This 7-bit value sets a counter, clocked at the IMO frequency. While the counter is counting down from this value, the relaxation oscillator edge slews at the maximum gain setting. During this interval, the IRANGE bits in the CS_CR2 register are internally set to maximum (11b). At the end of the interval, the user-defined IRANGE level is restored, so that the relaxation oscillator continues slewing with a slower edge rate to the target voltage threshold. If the FS_EN bit is low, the FastSlew setting has no effect. After each edge of the relaxation oscillator, the counter is re-loaded and the fast slewing interval re-occurs, followed by the slower edge rate at the end of the count down. Note that the IRANGE bits in the CS_CR2 register will always read the user-defined setting. Because the IRANGE value is forced to maximum during this interval, the increase in the edge rate can be 1X, 2X, 4X, or 8X, depending on the programmed value of the IRANGE bits. 0000000b No fast edge rate interval 0000001b Minimum fast edge rate interval (1 IMO period) … 1111111b Maximum fast edge rate interval (127 IMO period)
0	**FS_EN**	Enable bit for the Fast Slew mode 0 Fast slew mode disabled. 1 Fast slew mode enabled. After each relaxation oscillator transition, the relaxation oscillator runs with a higher current for a time controlled by the FastSlew bits.

20.3.21 PT_CFG

Programmable Timer Configuration Register

Individual Register Names and Addresses:

PT_CFG : 0,B0h

	7	6	5	4	3	2	1	0
Access : POR							RW : 0	RW : 0
Bit Name							One Shot	START

This register configures the programmable timer.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 131 in the Programmable Timer chapter.

Bit	Name		Description
1	**One Shot**	0	Continuous count mode. Timer reloads the count value from the data registers at each terminal count, and continues counting.
		1	One-shot mode. Timer goes through one complete count period and then stops. At completion, the START bit in this register is cleared.
0	**START**	0	Timer held in reset.
		1	Timer counts down from a full count determined from its data registers (PT_DATA1, PT_DATA0). When complete, it either stops or reloads and continues, based on the One Shot bit in this register.

20.3.22 PT_DATA1

Programmable Timer Data Register 1

Individual Register Names and Addresses:

PT_DATA1 : 0,B1h

	7	6	5	4	3	2	1	0
Access : POR				RW : 00				
Bit Name				DATA[4:0]				

This register provides the programmable timer with its upper 5 bits of the count value.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 131 in the Programmable Timer chapter.

Bit	Name	Description
4:0	**Data[4:0]**	Holds upper 5 bits of 13-bit count value.

20.3.23 PT_DATA0

Programmable Timer Data Register 0

Individual Register Names and Addresses:

PT_DATA0 : 0,B2h

	7	6	5	4	3	2	1	0
Access : POR	RW : 00							
Bit Name	Data[7:0]							

This register is used to provide the programmable timer with its lower 8 bits of the count value.

For additional information, refer to the Register Definitions on page 131 in the Programmable Timer chapter.

Bit	Name	Description
7:0	**Data[7:0]**	Holds lower 8 bits of 13-bit count value.

20.3.24 CUR_PP

Current Page Pointer Register

Individual Register Names and Addresses:

CUR_PP: 0,D0h

	7	6	5	4	3	2	1	0
Access : POR								RW : 0
Bit Name								Page Bit

This register sets the effective SRAM page for normal memory accesses in a multi-SRAM page PSoC device.

This register is only used when a device has more than one page of SRAM.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 34 in the RAM Paging chapter.

Bit	Name	Description
0	**Page Bit**	This bit determines which SRAM page is used for generic SRAM access. See the RAM Paging chapter on page 31 for more information. 0b SRAM Page 0 1b SRAM Page 1 **Note** A value beyond the available SRAM, for a specific PSoC device, should not be set.

20.3.25 STK_PP

Stack Page Pointer Register

Individual Register Names and Addresses:

STK_PP: 0,D1h

	7	6	5	4	3	2	1	0
Access : POR								RW : 0
Bit Name								Page Bit

This register sets the effective SRAM page for stack memory accesses in a multi-SRAM page PSoC device.

This register is only used when a device has more than one page of SRAM.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of ‘0’. For additional information, refer to the Register Definitions on page 34 in the RAM Paging chapter.

Bit	Name	Description
0	**Page Bit**	This bit determines which SRAM page is used to hold the stack. See the RAM Paging chapter on page 31 for more information. 0b SRAM Page 0 1b SRAM Page 1 **Note** A value beyond the available SRAM, for a specific PSoC device, should not be set.

20.3.26 IDX_PP

Indexed Memory Access Page Pointer Register

Individual Register Names and Addresses:

IDX_PP: 0,D3h

	7	6	5	4	3	2	1	0
Access : POR								RW : 0
Bit Name								Page Bit

This register sets the effective SRAM page for indexed memory accesses in a multi-SRAM page PSoC device.

This register is only used when a device has more than one page of SRAM.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 34 in the RAM Paging chapter.

Bit	Name	Description
0	**Page Bit**	This bit determines which SRAM page an indexed memory access operates on. See the Register Definitions on page 34 for more information on when this register is active. 0b SRAM Page 0 1b SRAM Page 1 **Note** A value beyond the available SRAM, for a specific PSoC device, should not be set.

20.3.27 MVR_PP

MVI Read Page Pointer Register

Individual Register Names and Addresses:

MVR_PP: 0,D4h

	7	6	5	4	3	2	1	0
Access : POR								RW : 0
Bit Name								Page Bit

This register sets the effective SRAM page for MVI read memory accesses in a multi-SRAM page PSoC device.

This register is only used when a device has more than one page of SRAM.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 34 in the RAM Paging chapter.

Bit	Name	Description
0	**Page Bit**	This bit determines on which SRAM page an MVI Read instruction operates. 0b SRAM Page 0 1b SRAM Page 1 **Note** A value beyond the available SRAM, for a specific PSoC device, is not set.

20.3.28 MVW_PP

MVI Write Page Pointer Register

Individual Register Names and Addresses:

MVW_PP: 0,D5h

	7	6	5	4	3	2	1	0
Access : POR								RW : 0
Bit Name								Page Bit

This register sets the effective SRAM page for MVI write memory accesses in a multi-SRAM page PSoC device.

This register is only used when a device has more than one page of SRAM.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 34 in the RAM Paging chapter.

Bit	Name	Description
0	**Page Bit**	This bit determines on which SRAM page aa MVI Write instruction operates. 0b SRAM Page 0 1b SRAM Page 1 **Note** A value beyond the available SRAM, for a specific PSoC device, should not be set.

20.3.29 I2C_CFG

I²C Configuration Register

Individual Register Names and Addresses:

I2C_CFG: 0,D6h

	7	6	5	4	3	2	1	0
Access : POR		RW : 0		RW : 0	RW : 0			RW : 0
Bit Name		PSelect		Stop IE	Clock Rate[1:0]			Enable

This register sets the basic operating modes, baud rate, and selection of interrupts.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 98 in the I2C Slave chapter.

Bit	Name	Description
6	**PSelect**	I2C Pin Select 0 P1[5] and P1[7] 1 P1[0] and P1[1] **Note** Read the I2C Slave chapter for a discussion of the side effects of choosing the P1[0] and P1[1] pair of pins.
4	**Stop IE**	Stop Interrupt Enable 0 Disabled 1 Enabled. An interrupt is generated on the detection of a Stop condition.
3:2	**Clock Rate[1:0]**	00b 100K Standard Mode 01b 400K Fast Mode 10b 50K Standard Mode 11b Reserved
0	**Enable**	0 Disabled 1 Enabled

20.3.30 I2C_SCR

I²C Status and Control Register

Individual Register Names and Addresses:

I2C_SCR: 0,D7h

	7	6	5	4	3	2	1	0
Access : POR	RC : 0		RC : 0	RW : 0	RC : 0	RW : 0	RC : 0	RC : 0
Bit Name	Bus Error		Stop Status	ACK	Address	Transmit	LRB	Byte Complete

The slave uses this register to control the flow of data bytes and to keep track of the bus state during a transfer.

Bits in this register are held in reset until one of the enable bits in I2C_CFG is set.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 98 in the I2C Slave chapter.

Bit	Name	Description
7	**Bus Error**	0 Status bit. It must be cleared by firmware by writing a '0' to the bit position. It is never cleared by the hardware. 1 A misplaced Start or Stop condition was detected.
5	**Stop Status**	0 Status bit. It must be cleared by firmware with write of '0' to the bit position. It is never cleared by the hardware. 1 A Stop condition was detected.
4	**ACK**	Acknowledge Out. Bit is automatically cleared by hardware on a Byte Complete event. 0 NACK the last received byte. 1 ACK the last received byte
3	**Address**	0 Status bit. It must be cleared by firmware with write of '0' to the bit position. 1 The received byte is a slave address.
2	**Transmit**	This bit is set by firmware to define the direction of the byte transfer. Any Start detect or a write to the Start or Restart generate bits, when operating in Master mode, will also clear the bit. 0 Receive mode 1 Transmit mode
1	**LRB**	This is the Last Received Bit. The value of the 9th bit in a Transmit sequence, which is the acknowledge bit from the receiver. Any Start detect or a write to the Start or Restart generate bits, when operating in Master mode, will also clear the bit. 0 Last transmitted byte was ACK'ed by the receiver. 1 Last transmitted byte was NACK'ed by the receiver.
0	**Byte Complete**	Transmit/Receive Mode: 0 No completed transmit/receive since last cleared by firmware. Any Start detect or a write to the Start or Restart generate bits, when operating in Master mode, will also clear the bit. Transmit Mode: 1 Eight bits of data were transmitted and an ACK or NACK has been received. Receive Mode: 1 Eight bits of data were received.

20.3.31 I2C_DR

I²C Data Register

Individual Register Names and Addresses:

I2C_DR: 0,D8h

	7	6	5	4	3	2	1	0
Access : POR	RW : 00							
Bit Name	Data[7:0]							

This register provides read/write access to the Shift register.

This register is read only for received data and write only for transmitted data.

For additional information, refer to the Register Definitions on page 98 in the I2C Slave chapter.

Bit	Name	Description
7:0	**Data[7:0]**	Read received data or write data to transmit

20.3.32 INT_CLR0

Interrupt Clear Register 0

Individual Register Names and Addresses:

INT_CLR0: 0,DAh

	7	6	5	4	3	2	1	0
Access : POR	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0
Bit Name	I2C	Sleep	SPI	GPIO	Timer	CapSense	Analog	V Monitor

This register enables the individual interrupt sources' ability to clear posted interrupts.

When bits in this register are read, a '1' is returned for every bit position that has a corresponding posted interrupt. When bits in this register are written with a '0' and ENSWINT is not set, posted interrupts are cleared at the corresponding bit positions. If there was not a posted interrupt, there is no effect. When bits in this register are written with a '1' and ENSWINT is set, an interrupt is posted in the interrupt controller.

For additional information, refer to the Register Definitions on page 48 in the Interrupt Controller chapter.

Bit	Name	Description	
7	**I2C**	Read 0	No posted interrupt for I2C.
		Read 1	Posted interrupt present for I2C.
		Write 0 AND ENSWINT = 0	Clear posted interrupt if it exists.
		Write 1 AND ENSWINT = 0	No effect
		Write 0 AND ENSWINT = 1	No effect
		Write 1 AND ENSWINT = 1	Post an interrupt for I2C.
6	**Sleep**	Read 0	No posted interrupt for sleep timer.
		Read 1	Posted interrupt present for sleep timer.
		Write 0 AND ENSWINT = 0	Clear posted interrupt if it exists.
		Write 1 AND ENSWINT = 0	No effect
		Write 0 AND ENSWINT = 1	No effect
		Write 1 AND ENSWINT = 1	Post an interrupt for sleep timer.
5	**SPI**	Read 0	No posted interrupt for SPI.
		Read 1	Posted interrupt present for SPI.
		Write 0 AND ENSWINT = 0	Clear posted interrupt if it exists.
		Write 1 AND ENSWINT = 0	No effect
		Write 0 AND ENSWINT = 1	No effect
		Write 1 AND ENSWINT = 1	Post an interrupt for SPI.
4	**GPIO**	Read 0	No posted interrupt for general purpose inputs and outputs (pins).
		Read 1	Posted interrupt present for GPIO (pins).
		Write 0 AND ENSWINT = 0	Clear posted interrupt if it exists.
		Write 1 AND ENSWINT = 0	No effect
		Write 0 AND ENSWINT = 1	No effect
		Write 1 AND ENSWINT = 1	Post an interrupt for general purpose inputs and outputs (pins).

(continued on next page)

20.3.32 INT_CLR0 *(continued)*

3	**Time0r**	Read 0	No posted interrupt for Timer.
		Read 1	Posted interrupt present for Timer.
		Write 0 AND ENSWINT = 0	Clear posted interrupt if it exists.
		Write 1 AND ENSWINT = 0	No effect
		Write 0 AND ENSWINT = 1	No effect
		Write 1 AND ENSWINT = 1	Post an interrupt for Timer.
2	**CapSense**	Read 0	No posted interrupt for CapSense.
		Read 1	Posted interrupt present for CapSense.
		Write 0 AND ENSWINT = 0	Clear posted interrupt if it exists.
		Write 1 AND ENSWINT = 0	No effect
		Write 0 AND ENSWINT = 1	No effect
		Write 1 AND ENSWINT = 1	Post an interrupt for CapSense.
1	**Analog**	Read 0	No posted interrupt for analog.
		Read 1	Posted interrupt present for analog.
		Write 0 AND ENSWINT = 0	Clear posted interrupt if it exists.
		Write 1 AND ENSWINT = 0	No effect
		Write 0 AND ENSWINT = 1	No effect
		Write 1 AND ENSWINT = 1	Post an interrupt for analog.
0	**V Monitor**	Read 0	No posted interrupt for supply voltage monitor.
		Read 1	Posted interrupt present for supply voltage monitor.
		Write 0 AND ENSWINT = 0	Clear posted interrupt if it exists.
		Write 1 AND ENSWINT = 0	No effect
		Write 0 AND ENSWINT = 1	No effect
		Write 1 AND ENSWINT = 1	Post an interrupt for supply voltage monitor.

20.3.33 INT_MSK0

Interrupt Mask Register 0

Individual Register Names and Addresses:

INT_MSK0: 0,E0h

	7	6	5	4	3	2	1	0
Access : POR	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0
Bit Name	I2C	Sleep	SPI	GPIO	Timer	CapSense	Analog	V Monitor

This register enables the individual sources' ability to create pending interrupts.

When an interrupt is masked off, the mask bit is '0'. The interrupt still posts in the interrupt controller. Therefore, clearing the mask bit only prevents a posted interrupt from becoming a pending interrupt.

For additional information, refer to the Register Definitions on page 48 in the Interrupt Controller chapter.

Bit	Name	Description	
7	**I2C**	0 1	Mask I2C interrupt Unmask I2C interrupt
6	**Sleep**	0 1	Mask sleep interrupt Unmask sleep interrupt
5	**SPI**	0 1	Mask SPI interrupt Unmask SPI interrupt
4	**GPIO**	0 1	Mask GPIO interrupt Unmask GPIO interrupt
3	**Timer**	0 1	Mask Timer interrupt Unmask Timer interrupt
2	**CapSense**	0 1	Mask CapSense interrupt Unmask CapSense interrupt
1	**Analog**	0 1	Mask analog interrupt Unmask analog interrupt
0	**V Monitor**	0 1	Mask voltage monitor interrupt Unmask voltage monitor interrupt

20.3.34 INT_SW_EN

Interrupt Software Enable Register

Individual Register Names and Addresses:

INT_SW_EN: 0,E1h

	7	6	5	4	3	2	1	0
Access : POR								RW : 0
Bit Name								ENSWINT

This register enables software interrupts.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of ‘0’. For additional information, refer to the Register Definitions on page 48 in the Interrupt Controller chapter.

Bit	Name		Description
0	**ENSWINT**	0	Disable software interrupts
		1	Enable software interrupts

20.3.35 INT_VC

Interrupt Vector Clear Register

Individual Register Names and Addresses:

INT_VC: 0,E2h

	7	6	5	4	3	2	1	0
Access : POR	RC : 00							
Bit Name	Pending Interrupt[7:0]							

This register returns the next pending interrupt and clears all pending interrupts when written.

For additional information, refer to the Register Definitions on page 48 in the Interrupt Controller chapter.

Bit	Name	Description	
7:0	**Pending Interrupt[7:0]**	Read	Returns vector for highest priority pending interrupt.
		Write	Clears all pending and posted interrupts.

20.3.36 RES_WDT

Reset Watchdog Timer Register

Individual Register Names and Addresses:

RES_WDT: 0,E3h

	7	6	5	4	3	2	1	0
Access : POR	W : 00							
Bit Name	WDSL_Clear[7:0]							

This register clears the watchdog timer alone, or clear both the watchdog timer and the sleep timer together.

For additional information, refer to the Register Definitions on page 63 in the Sleep and Watchdog chapter.

Bit	Name	Description
7:0	**WDSL_Clear[7:0]**	Any write clears the watchdog timer. A write of 38h clears both the watchdog and sleep timers.

20.3.37 CPU_F

M8C Flag Register

Individual Register Names and Addresses:

CPU_F: x,F7h

	7	6	5	4	3	2	1	0
Access : POR	RL : 0			RL : 0		RL : 0	RL : 0	RL : 0
Bit Name	PgMode[1:0]			XIO		Carry	Zero	GIE

This register provides read access to the M8C flags.

The AND f, expr; OR f, expr; and XOR f, expr flag instructions can be used to modify this register.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 30 in the M8C chapter.

Bit	Name	Description	
7:6	**PgMode[1:0]**	00b	Direct Address mode and Indexed Address mode operands are referred to RAM Page 0, regardless of the values of CUR_PP and IDX_PP. Note that this condition prevails on entry to an Interrupt Service Routine when the CPU_F register is cleared.
		01b	Direct Address mode instructions are referred to Page 0. Indexed Address mode instructions are referred to the RAM page specified by the stack page pointer, STK_PP.
		10b	Direct Address mode instructions are referred to the RAM page specified by the current page pointer, CUR_PP. Indexed Address mode instructions are referred to the RAM page specified by the index page pointer, IDX_PP.
		11b	Direct Address mode instructions are referred to the RAM page specified by the current page pointer, CUR_PP. Indexed Address mode instructions are referred to the RAM page specified by the stack page pointer, STK_PP.
4	**XIO**	0	Normal register address space
		1	Extended register address space. Primarily used for configuration.
2	**Carry**	Set by the M8C CPU Core to indicate whether there has been a carry in the previous logical/arithmetic operation.	
		0	No carry
		1	Carry
1	**Zero**	Set by the M8C CPU Core to indicate whether there has been a zero result in the previous logical/arithmetic operation.	
		0	Not equal to zero
		1	Equal to zero
0	**GIE**	0	M8C processes no interrupts.
		1	Interrupt processing enabled.

20.3.38 IDAC_D

Current DAC Data Register

Individual Register Names and Addresses:

IDAC_D : 0,FDh

	7	6	5	4	3	2	1	0
Access : POR	RW : 00							
Bit Name	IDACDATA[7:0]							

This register specifies the 8-bit multiplying factor that determines the output IDAC current.

For additional information, refer to the Register Definitions on page 74 in the CapSense Module chapter.

Bits	Name	Description
7:0	**IDACDATA[7:0]**	The eight-bit value that selects the number of current units that combine to form the IDAC current. This current then drives the analog mux bus when IDAC mode is enabled. For example, a setting of 80h means that the charging current is 128 current units. The current size also depends upon the IRANGE setting in the CS_CR2 register. This setting supplies the charging current for the relaxation oscillator. This current and the external capacitance connected to the analog global bus determines the RO frequency. This register is also used to set the charging current in the proximity detect mode. Step size is approximately 330 nA/bit for default IRANGE state 00b. 00h Smallest current (nominally zero unless IBOOST bit is set in the CS_CR3 register). FFh Largest current **Note** At very low currents a low frequency noise exists on the IDAC output. To avoid generating this low frequency noise, do not use IDAC_D settings equal to or less than three for capacitive sensing applications.

20.3.39 CPU_SCR1

System Status and Control Register 1

Individual Register Names and Addresses:

CPU_SCR1: x,FEh

	7	6	5	4	3	2	1	0
Access : POR	R : 0			RW : 0				RW : 0
Bit Name	IRESS			SLIMO				IRAMDIS

This register conveys the status and control of events related to internal resets and watchdog reset.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 108 in the System Resets chapter.

Bit	Name	Description
7	**IRESS**	This bit is read only. 0 Boot phase only executed once. 1 Boot phase occurred multiple times.
4	**SLIMO**	This bit reduces frequency of the internal main oscillator (IMO) and is reserved on PSoC devices that do not support the slow IMO (see the Architectural Description on page 57). 0 IMO produces 12 MHz 1 Slow IMO (6 MHz)
0	**IRAMDIS**	0 SRAM is initialized to 00h after POR, XRES, and WDR. 1 Addresses 03h - D7h of SRAM Page 0 are not modified by WDR.

20.3.40 CPU_SCR0

System Status and Control Register 0

Individual Register Names and Addresses:

CPU_SCR0: x,FFh

	7	6	5	4	3	2	1	0
Access : POR	R : 0		RC : 0	RC : 1	RW : 0			RW : 0
Bit Name	GIES		WDRS	PORS	Sleep			STOP

This register conveys the status and control of events for various functions of a PSoC device.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 108 in the System Resets chapter.

Bit	Name	Description
7	**GIES**	Global Interrupt Enable Status. It is recommended that the user read the Global Interrupt Enable Flag bit from the CPU_F register on page 176. This bit is read only for GIES. Its use is discouraged, because the Flag register is now readable at address x,F7h (read only).
5	**WDRS**	Watchdog Reset Status. This bit may not be set by user code; however, it may be cleared by writing a '0'. 0 No Watchdog reset has occurred. 1 Watchdog reset has occurred.
4	**PORS**	Power On Reset Status. This bit may not be set by user code; however, it may be cleared by writing a '0'. 0 Power On Reset has not occurred and watchdog timer is enabled. 1 Will be set after external reset or Power On Reset.
3	**Sleep**	Set by the user to enable the CPU sleep state. CPU will remain in Sleep mode until any interrupt is pending. 0 Normal operation 1 Sleep
0	**STOP**	0 M8C is free to execute code. 1 M8C is halted. Can only be cleared by POR, XRES, or WDR.

20.4 Bank 1 Registers

The following registers are all in bank 1 and are listed in address order. Registers that are in both Bank 0 and Bank 1 are listed in address order in the section titled Bank 0 Registers on page 138.

20.4.1 PRTxDM0

Port Drive Mode Bit Register 0

Individual Register Names and Addresses:

PRT0DM0 : 1,00h PRT1DM0 : 1,04h PRT2DM0 : 1,08h PRT3DM0 : 1,0Ch

	7	6	5	4	3	2	1	0
Access : POR	RW : 00							
Bit Name	Drive Mode 0[7:0]							

This register is one of two registers whose combined value determines the unique drive mode of each bit in a GPIO port.

In register PRTxDM0 there are four possible drive modes for each port pin. Two mode bits are required to select one of these modes, and these two bits are spread into two different registers (PRTxDM0 and PRTxDM1 on page 181). The bit position of the affected port pin (for example, Pin[2] in Port 0) is the same as the bit position of each of the two Drive Mode register bits that control the Drive mode for that pin (for example, bit[2] in PRT0DM0 and bit[2] in PRT0DM1). The two bits from the two registers are treated as a group. These are referred to as DM1 and DM0, or together as DM[1:0].

All Drive mode bits are shown in the subtable below ([1**0**] refers to the combination (in order) of bits in a given bit position); however, this register only controls the ***least significant bit (LSb)*** of the Drive mode.

The upper nibble of the PRT3DM0 register returns the last data bus value when read and should be masked off before using this information.

For additional information, refer to the Register Definitions on page 55 in the GPIO chapter.

Bit	Name	Description				
7:0	**Drive Mode 0[7:0]**	Bit 0 of the Drive mode, for each of 8-port pins, for a GPIO port.				
		[10**]**		**Pin Output High**	**Pin Output Low**	**Notes**
		00b	Pull up	Resistive	Strong	
		0**1**b	Strong	Strong	Strong	
		1**0**b	ITIZ	High-Z	High-Z	Reset state. Digital input disabled for zero power.
		1**1**b	Open Drain Low	High-Z	Strong	I2C compatible mode. For digital inputs, use this mode with data bit PRTxDR register) set high.
		Note A bold digit, in the table above, signifies that the digit is used in this register.				

20.4.2 PRTxDM1

Port Drive Mode Bit Register 1

Individual Register Names and Addresses:

PRT0DM1 : 1,01h PRT1DM1 : 1,05h PRT2DM1 : 1,09h PRT3DM1 : 1,0Dh

	7	6	5	4	3	2	1	0
Access : POR	RW : FF							
Bit Name	Drive Mode 1[7:0]							

This register is one of three registers whose combined value determines the unique Drive mode of each bit in a GPIO port.

In register PRTxDM1 there are four possible drive modes for each port pin. Two mode bits are required to select one of these modes, and these two bits are spread into two different registers (PRTxDM1 and PRTxDM0 on page 180). The bit position of the effected port pin (for example, Pin[2] in Port 0) is the same as the bit position of each of the two Drive Mode register bits that control the Drive mode for that pin (for example, bit[2] in PRT0DM0 and bit[2] in PRT0DM1). The two bits from the two registers are treated as a group. These are referred to as DM1 and DM0, or together as DM[1:0].

All Drive mode bits are shown in the subtable below ([**1**0] refers to the combination (in order) of bits in a given bit position); however, this register only controls the most significant bit (MSb) of the Drive mode.

The upper nibble of the PRT3DM1 register returns the last data bus value when read and should be masked off prior to using this information.

For additional information, refer to the Register Definitions on page 55 in the GPIO chapter.

Bit	Name	Description				
7:0	**Drive Mode 1[7:0]**	Bit 1 of the Drive mode, for each of 8-port pins, for a GPIO port.				
		[10**]**		**Pin Output High**	**Pin Output Low**	**Notes**
		00b	Pull up	Resistive	Strong	
		0**1**b	Strong	Strong	Strong	
		1**0**b	ITIZ	High-Z	High-Z	Reset state. Digital input disabled for zero power.
		1**1**b	Open Drain Low	High-Z	Strong	I2C compatible mode. For digital inputs, use this mode with data bit (PRTxDR register) set high.
		Note A bold digit, in the table above, signifies that the digit is used in this register.				

20.4.3 SPI_CFG

SPI Configuration Register

Individual Register Names and Addresses:

SPI_CFG : 1,29h

	7	6	5	4	3	2	1	0
Access : POR	RW : 0			RW : 0	RW : 0	RW : 0	RW : 0	RW : 0
Bit Name	Clock Sel[2:0]			Bypass	SS_	SS_EN_	Int Sel	Slave

This register configures the SPI.

Do not change the values in this register while the block is enabled.

For additional information, refer to the Register Definitions on page 117 in the SPI chapter.

Bit	Name	Description
7:5	**Clock Sel[2:0]**	SYSCLK in Master mode 000b / 2 001b / 4 010b / 8 011b / 16 100b / 32 101b / 64 110b / 128 111b / 256
4	**Bypass**	Bypass Synchronization 0 All pin unputs are doubled, synchronized 1 Input synchronization is bypassed.
3	**SS_**	Slave Select in Slave mode 0 Slave selected 1 Slave not selected
2	**SS_EN_**	Internal Slave Select Enable 0 Slave selection determined from SS_ bit. 1 Slave selection determined from external SS_ pin.
1	**Int Sel**	Interrupt Select 0 Interrupt on TX Reg Empty. 1 Interrupt on SPI Complete.
0	**Slave**	0 Operates as a master. 1 Operates as a slave.

20.4.4 MUX_CRx

Analog Mux Port Bit Enables Register

Individual Register Names and Addresses:

MUX_CR0 : 1,D8h MUX_CR1 : 1,D9h MUX_CR2 : 1,DAh MUX_CR3 : 1,DBh

	7	6	5	4	3	2	1	0
Access : POR	RW : 00							
Bit Name	ENABLE[7:0]							

This register controls the connection between the analog mux bus and the corresponding pin.

Port 3 is a 4-bit port, so the upper 4 bits of the MUX_CR3 register are reserved and returns an undefined value when read.

For additional information, refer to the Register Definitions on page 82 in the IO Analog Multiplexer chapter.

Bits	Name	Description
7:0	**ENABLE[7:0]**	Each bit controls the connection between the analog mux bus and the corresponding port pin. For example, MUX_CR2[3] controls the connection to bit 3 in Port 2. Any number of pins may be connected at the same time. Note that if a discharge clock is selected in the AMUX_CFG register, the connection to the mux bus will be switched on and off under hardware control. 0 No connection between port pin and analog mux bus. 1 Connect port pin to analog mux bus.

20.4.5 IO_CFG

Input/Output Configuration Register

Individual Register Names and Addresses:

IO_CFG : 1,DCh

	7	6	5	4	3	2	1	0
Access : POR							RW : 0	RW : 0
Bit Name							REG_EN	IOINT

This register configures the Port 1 output regulator and set the interrupt mode for all GPIO.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 82 in the GPIO chapter.

Bits	Name	Description
1	**REG_EN**	Controls the regulator on Port 1 outputs. 0 Regulator disabled, so Port 1 strong outputs drive to Vdd. 1 Regulator enabled, so Port 1 strong outputs drive to approximately 3V (for Vdd > 3V).
0	**IOINT**	Sets the GPIO interrupt mode for all pins in the PSoC device. GPIO interrupts are also controlled at each pin by the PRTxIE registers, and by the global GPIO bit in the INT_MSK0 register. 0 GPIO interrupt configured for interrupt when pin is low. 1 GPIO interrupt configured for interrupt when pin state changes from last time port was read.

20.4.6 OUT_P1

Output Override to Port 1 Register

Individual Register Names and Addresses:

OUT_P1: 1,DDh

	7	6	5	4	3	2	1	0
Access : POR	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0	RW : 0
Bit Name	P16D	P16EN	P14D	P14EN	P12D	P12EN	P10D	P10EN

This register enables specific internal signals to be output to Port 1 pins.

Note that the GPIO drive modes must be specified to support the desired output mode (registers PRT1DM1 and PRT1DM0). If a pin is enabled for output by a bit in this register, the corresponding signal has priority over any other internal function that may be configured to output to that pin.

For additional information, refer to the Register Definitions on page 92 in the Digital Clocks chapter.

Bit	Name	Description
7	**P16D**	This bit selects the data output to P1[6] when P16EN is high. 0 Select Timer output (TIMEROUT) 1 Select CLK32
6	**P16EN**	This bit enables pin P1[6] for output of the signal selected by the P16D bit. 0 No internal signal output to P1[6] 1 Output the signal selected by P16D to P1[6]
5	**P14D**	This bit selects the data output to P1[4] when P14EN is high. 0 Select Relaxation Oscillator (RO) 1 Select Comparator 1 Output (CMP1)
4	**P14EN**	This bit enables pin P1[4] for output of the signal selected by the P14D bit. 0 No internal signal output to P1[4] 1 Output the signal selected by P14D to P1[4]
3	**P12D**	This bit selects the data output to P1[2] when P12EN is high. 0 Select Main System Clock (SYSCLK) 1 Select CapSense output signal (CS). This signal is selected by the CSOUT[1:0] bits in the CS_CR0 register.
2	**P12EN**	This bit enables pin P1[2] for output of the signal selected by the P12D bit. 0 No internal signal output to P1[2] 1 Output the signal selected by P12D to P1[2]
1	**P10D**	This bit selects the data output to P1[0] when P10EN is high. 0 Select Sleep Interrupt (SLPINT) 1 Select Comparator 0 Output (CMP0)
0	**P10EN**	This bit enables pin P1[0] for output of the signal selected by the P10D bit. 0 No internal signal output to P1[0] 1 Output the signal selected by P10D to P1[0]

20.4.7 OSC_CR0

Oscillator Control Register 0

Individual Register Names and Addresses:

OSC_CR0: 1,E0h

	7	6	5	4	3	2	1	0
Access : POR		RW : 0	RW : 0	RW : 0		RW : 001b		
Bit Name		Disable Buzz	No Buzz	Sleep[1:0]		CPU Speed[2:0]		

This register configures various features of internal clock sources and clock nets.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 92 in the Digital Clocks chapter.

Bit	Name	Description
6	**Disable Buzz**	This bit has lower priority than the No Buzz bit. Therefore if No Buzz = 1, the Disable Buzz bit has no effect. 0 No effect on buzz modes 1 Buzz is disabled during sleep, with bandgap powered down. No periodic wakeup of the bandgap during sleep.
5	**No Buzz**	0 BUZZ bandgap during power down 1 Bandgap is always powered even during sleep.
4:3	**Sleep[1:0]**	Sleep Interval 00b 1.95 ms (512 Hz) 01b 15.6 ms (64 Hz) 10b 125 ms (8 Hz) 11b 1 s (1 Hz)
2:0	**CPU Speed[2:0]**	Bits set the CPU clock speed, based on the system clock (SYSCLK). SYSTOLE is 12 MHz by default, but it can optionally be set to 6 MHz or be driven from an external clock.

	6 MHz IMO	12 MHz IMO	External Clock	
000b	750 kHz	1.5 MHz	EXTCLK/8	
001b	1.5 MHz	3 MHz	EXTCLK/4	Reset State
010b	3 MHz	6 MHz	EXTCLK/2	
011b	6 MHz	12 MHz	EXTCLK/1	
100b	375 kHz	750 kHz	EXTCLK/16	
101b	187.5 kHz	375 kHz	EXTCLK/32	
110b	46.9 kHz	93.7 kHz	EXTCLK/128	
111b	23.4 kHz	46.8 kHz	EXTCLK/256	

20.4.8 OSC_CR2

Oscillator Control Register 2

Individual Register Names and Addresses:

OSC_CR2: 1,E2h

	7	6	5	4	3	2	1	0
Access : POR						RW : 0	RW : 0	
Bit Name						EXTCLKEN	IMODIS	

This register configures various features of internal clock sources and clock nets.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 92 in the Digital Clocks chapter.

Bit	Name	Description
2	**EXTCLKEN**	External Clock Mode Enable 0 Disabled. Operate from internal main oscillator. 1 Enabled. Operate from clock supplied at P1[4].
1	**IMODIS**	Internal Oscillator Disable. Set this bit to save power when using an external clock on P1[4]. 0 Enabled. Internal oscillator enabled. 1 Disabled. **Note** This bit must not be set high in the same instruction that sets EXTCLKEN high, but it can be set in the next instruction. Also, this bit must not be set high if the external clock frequency is less than 6 MHz. When switching from external clock to internal clock, enable the IMO for at least 10 µs before the transition to internal clock. Refer to Switch Operation on page 90.

20.4.9 VLT_CR

Voltage Monitor Control Register

Individual Register Names and Addresses:

VLT_CR: 1,E3h

	7	6	5	4	3	2	1	0
Access : POR			RW : 0		RW : 0	RW : 0		
Bit Name			PORLEV[1:0]		LVDTBEN	VM[2:0]		

This register sets the trip points for the POR and LVD.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 113 in the POR and LVD chapter.

Bit	Name	Description
5:4	**PORLEV[1:0]**	These bits set the POR level per the DC electrical specifications in the PSoC device data sheet. 00b POR level for 2.4 V operation (refer to the PSoC device data sheet) 01b POR level for 2.7V operation (refer to the PSoC device data sheet) 10b POR level for 3.0V operation 11b Reserved
3	**LVDTBEN**	This bit enables reset of CPU speed register by LVD comparator output. 0 Disables CPU speed throttle-back. 1 Enables CPU speed throttle-back.
2:0	**VM[2:0]**	These bits set the LVD levels according to the DC electrical specifications in the PSoC device data sheet, for those PSoC devices with this feature. 000b Lowest voltage setting 001b 010b . 011b . 100b . 101b 110b 111b Highest voltage setting

20.4.10 VLT_CMP

Voltage Monitor Comparators Register

Individual Register Names and Addresses:

VLT_CMP: 1,E4h

	7	6	5	4	3	2	1	0
Access : POR					R : 0		R : 0	R : 0
Bit Name					NoWrite		LVD	PPOR

This register reads the state of internal supply voltage monitors.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 113 in the POR and LVD chapter.

Bit	Name	Description
3	**NoWrite**	This bit reads the state of the Flash write voltage monitor. 0 Sufficient voltage for Flash write. 1 Insufficient voltage for Flash write.
1	**LVD**	This bit reads state of LVD comparator. 0 Vdd is above LVD trip point. 1 Vdd is below LVD trip point.
0	**PPOR**	This bit reads state of Precision POR comparator. This is only useful with PPOR reset disabled, with PORLEV[1:0] in the VLT_CR register set to 11b. 0 Vdd is above PPOR trip voltage. 1 Vdd is below PPOR trip voltage.

20.4.11 IMO_TR

Internal Main Oscillator Trim Register

Individual Register Names and Addresses:

IMO_TR: 1,E8h

	7	6	5	4	3	2	1	0
Access : POR	W : 00							
Bit Name	Trim[7:0]							

This register manually centers the Internal Main Oscillator's (IMO) output to a target frequency.

It is strongly recommended that the user not alter this register's values unless Slow IMO mode is used. Do not change the value in this register.

For additional information, refer to the Register Definitions on page 58 in the Internal Main Oscillator chapter.

Bit	Name	Description
7:0	**Trim[7:0]**	The value of this register is used to trim the Internal Main Oscillator. Its value is set to the best value for the device during boot. ***The value of these bits should not be changed unless Slow IMO mode is used***. 00h Lowest frequency setting 01h ... ... 7Fh 80h Design center setting 81h ... ... FEh FFh Highest frequency setting

20.4.12 ILO_TR

Internal Low Speed Oscillator Trim Register

Individual Register Names and Addresses:

ILO_TR: 1,E9h

	7	6	5	4	3	2	1	0
Access : POR			W : 0		W : 0			
Bit Name			Bias Trim[1:0]		Freq Trim[3:0]			

This register sets the adjustment for the Internal Low Speed Oscillator (ILO).

It is strongly recommended that the user not alter this register's values. The trim bits are set to factory specifications and should not be changed.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 59 in the Internal Low Speed Oscillator chapter.

Bit	Name	Description
5:4	**Bias Trim[1:0]**	This bit is the device specific, best value during boot. ***The value of these bits should not be changed.*** 00b Medium bias 01b Maximum bias (recommended) 10b Minimum bias 11b Intermediate Bias * * About 15% higher than the minimum bias.
3:0	**Freq Trim[3:0]**	This bit is set to the device specific, best value during boot. ***Do not change the these bit values.***

20.4.13 BDG_TR

Bandgap Trim Register

Individual Register Names and Addresses:

BDG_TR: 1,EAh

	7	6	5	4	3	2	1	0
Access : POR	RW : 2			RW : 10h				
Bit Name	TC[2:0]			V[4:0]				

This register adjusts the bandgap and add an RC filter to Agnd.

It is strongly recommended that the user not alter this register's values.

For additional information, refer to the Register Definitions on page 106 in the Internal Voltage References chapter.

Bit	Name	Description
7:5	**TC[2:0]**	These bits are set to the best value for the device during boot. ***Do not change these bit values.***
4:0	**V[4:0]**	These bits are set to the best value for the device during boot. ***Do not change these bit values.***

20.4.14 SLP_CFG

Sleep Configuration Register

Individual Register Names and Addresses:

SLP_CFG: 1,EBh

	7	6	5	4	3	2	1	0
Access : POR	RW : 0							
Bit Name	PSSDC[1:0]							

This register sets the sleep duty cycle.

It is strongly recommended that the user not alter this register's values. The trim bits are set to factory specifications and should not be changed.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Always write reserved bits with a value of '0'. For additional information, refer to the Register Definitions on page 63 in the Sleep and Watchdog chapter.

Bit	Name	Description
7:6	**PSSDC[1:0]**	These bits set the sleep duty cycle. They control the ratios (in numbers of 32.768 kHz clock periods) of "on" time versus "off" time for PORLVD4, bandgap reference. ***Do not change these bit values.*** 00b 1 / 256 (8 ms) 01b 1 / 1024 (31.2 ms) 10b 1 / 64 (2 ms) 11b 1 / 16 (500 ms)

Section F: Glossary

The Glossary section explains the terminology used in this technical reference manual. Glossary terms are characterized in ***bold, italic font*** throughout the text of this manual.

A

accumulator	In a CPU, a register in which intermediate results are stored. Without an accumulator, it would be necessary to write the result of each calculation (addition, subtraction, shift, and so on.) to main memory and read them back. Access to main memory is slower than access to the accumulator, which usually has direct paths to and from the arithmetic and logic unit (ALU).
active high	1. A logic signal having its asserted state as the logic 1 state. 2. A logic signal having the logic 1 state as the higher voltage of the two states.
active low	1. A logic signal having its asserted state as the logic 0 state. 2. A logic signal having its logic 1 state as the lower voltage of the two states: inverted logic.
address	The label or number identifying the memory location (RAM, ROM, or register) where a unit of information is stored.
algorithm	A procedure for solving a mathematical problem in a finite number of steps that frequently involve repetition of an operation.
ambient temperature	The temperature of the air in a designated area, particularly the area surrounding the PSoC device.
analog	As opposed to digital, signals that are on or off or '1' or '0'. Analog signals vary in a continuous manner. See also ***analog signals***.
analog blocks	The basic programmable opamp circuits. These are SC (switched capacitor) and CT (continuous time) blocks. These blocks can be interconnected to provide ADCs, DACs, multi-pole filters, gain stages, and much more.
analog output	An output that is capable of driving any voltage between the supply rails, instead of just a logic 1 or logic 0.
analog signals	A signal represented in a continuous form with respect to continuous times, as contrasted with a digital signal represented in a discrete (discontinuous) form in a sequence of time.
analog-to-digital (ADC)	A device that changes an analog signal to a digital signal of corresponding magnitude. Typically, an ADC converts a voltage to a digital number. The ***digital-to-analog (DAC)*** converter performs the reverse operation.
AND	See ***Boolean Algebra***.

API (Application Programming Interface) A series of software routines that comprise an interface between a computer application and lower-level services and functions (for example, user modules and libraries). APIs serve as building blocks for programmers that create software applications.

array An array, also known as a vector or list, is one of the simplest data structures in computer programming. Arrays hold a fixed number of equally-sized data elements, generally of the same data type. Individual elements are accessed by index using a consecutive range of integers, as opposed to an associative array. Most high level programming languages have arrays as a built-in data type. Some arrays are multi-dimensional, meaning they are indexed by a fixed number of integers; for example, by a group of two integers. One- and two-dimensional arrays are the most common. Also, an array can be a group of capacitors or resistors connected in some common form.

assembly A symbolic representation of the machine language of a specific processor. Assembly language is converted to machine code by an assembler. Usually, each line of assembly code produces one machine instruction, though the use of macros is common. Assembly languages are considered low level languages; where as C is considered a high level language.

asynchronous A signal whose data is acknowledged or acted upon immediately, irrespective of any clock signal.

attenuation The decrease in intensity of a signal as a result of absorption of energy and of scattering out of the path to the detector, but not including the reduction due to geometric spreading. Attenuation is usually expressed in dB.

B

bandgap reference A stable voltage reference design that matches the positive temperature coefficient of V_T with the negative temperature coefficient of V_{BE}, to produce a zero temperature coefficient (ideally) reference.

bandwidth
1. The frequency range of a message or information processing system measured in hertz.
2. The width of the spectral region over which an amplifier (or absorber) has substantial gain (or loss). It is sometimes represented more specifically as, for example, full width at half maximum.

bias
1. A systematic deviation of a value from a reference value.
2. The amount by which the average of a set of values departs from a reference value.
3. The electrical, mechanical, magnetic, or other force (field) applied to a device to establish a reference level to operate the device.

bias current The constant low level DC current that is used to produce a stable operation in amplifiers. This current can sometimes be changed to alter the bandwidth of an amplifier.

binary The name for the base 2 numbering system. The most common numbering system is the base 10 numbering system. The base of a numbering system indicates the number of values that may exist for a particular positioning within a number for that system. For example, in base 2, binary, each position may have one of two values (0 or 1). In the base 10, decimal, each position may have one of ten values (0, 1, 2, 3, 4, 5, 6, 7, 8, and 9).

bit A single digit of a binary number. Therefore, a bit may only have a value of ‘0’ or ‘1’. A group of 8 bits is called a byte. Because the PSoC's M8C is an 8-bit microcontroller, the PSoC's native data chunk size is a byte.

bit rate (BR) The number of bits occurring per unit of time in a bit stream, usually expressed in bits per second (bps).

block
1. A functional unit that performs a single function, such as an oscillator.
2. A functional unit that may be configured to perform one of several functions, such as a digital PSoC block or an analog PSoC block.

Boolean Algebra In mathematics and computer science, Boolean algebras or Boolean lattices, are algebraic structures which “capture the essence” of the logical operations AND, OR and NOT as well as set the theoretic operations union, intersection, and complement. Boolean algebra also defines a set of theorems that describe how Boolean equations can be manipulated. For example, these theorems are used to simplify Boolean equations which will reduce the number of logic elements needed to implement the equation.

The operators of Boolean algebra may be represented in various ways. Often they are simply written as AND, OR, and NOT. In describing circuits, NAND (NOT AND), NOR (NOT OR), XNOR (exclusive NOT OR), and XOR (exclusive OR) may also be used. Mathematicians often use + (for example, A+B) for OR and • for AND (for example, A*B) (since in some ways those operations are analogous to addition and multiplication in other algebraic structures) and represent NOT by a line drawn above the expression being negated (for example, ~A, A_, !A).

break-before-make The elements involved go through a disconnected state entering (‘break”) before the new connected state (“make”).

broadcast net A signal that is routed throughout the microcontroller and is accessible by many blocks or systems.

buffer
1. A storage area for data that is used to compensate for a speed difference, when transferring data from one device to another. Usually refers to an area reserved for IO operations into which data is read or from which data is written.
2. A portion of memory set aside to store data, often before it is sent to an external device or as it is received from an external device.
3. An amplifier used to lower the output impedance of a system.

bus
1. A named connection of nets. Bundling nets together in a bus makes it easier to route nets with similar routing patterns.
2. A set of signals performing a common function and carrying similar data. Typically represented using vector notation; for example, address[7:0].
3. One or more conductors that serve as a common connection for a group of related devices.

byte A digital storage unit consisting of 8 bits.

C

C A high level programming language.

capacitance A measure of the ability of two adjacent conductors, separated by an insulator, to hold a charge when a voltage differential is applied between them. Capacitance is measured in units of Farads.

capture To extract information automatically through the use of software or hardware, as opposed to hand-entering of data into a computer file.

chaining	Connecting two or more 8-bit digital blocks to form 16-, 24-, and even 32-bit functions. Chaining allows certain signals such as Compare, Carry, Enable, Capture, and Gate to be produced from one block to another.
checksum	The checksum of a set of data is generated by adding the value of each data word to a sum. The actual checksum can simply be the result sum or a value that must be added to the sum to generate a pre-determined value.
chip	A single monolithic Integrated Circuit (IC). See also ***integrated circuit (IC)***.
clear	To force a bit/register to a value of logic 0.
clock	The device that generates a periodic signal with a fixed frequency and duty cycle. A clock is sometimes used to synchronize different logic blocks.
clock generator	A circuit that is used to generate a clock signal.
CMOS	The logic gates constructed using ***MOS*** transistors connected in a complementary manner. CMOS is an acronym for complementary metal-oxide semiconductor.
comparator	An electronic circuit that produces an output voltage or current whenever two input levels simultaneously satisfy predetermined amplitude requirements.
compiler	A program that translates a high level language, such as C, into machine language.
configuration	In a computer system, an arrangement of functional units according to their nature, number, and chief characteristics. Configuration pertains to hardware, software, firmware, and documentation. The configuration will affect system performance.
configuration space	In PSoC devices, the register space accessed when the XIO bit, in the CPU_F register, is set to ‘1’.
crowbar	A type of over-voltage protection that rapidly places a low resistance shunt (typically an SCR) from the signal to one of the power supply rails, when the output voltage exceeds a predetermined value.
crystal oscillator	An oscillator in which the frequency is controlled by a piezoelectric crystal. Typically a piezoelectric crystal is less sensitive to ambient temperature than other circuit components.
cyclic redundancy check (CRC)	A calculation used to detect errors in data communications, typically performed using a linear feedback shift register. Similar calculations may be used for a variety of other purposes such as data compression.

D

data bus	A bi-directional set of signals used by a computer to convey information from a memory location to the central processing unit and vice versa. More generally, a set of signals used to convey data between digital functions.
data stream	A sequence of digitally encoded signals used to represent information in transmission.
data transmission	The sending of data from one place to another by means of signals over a channel.

debugger	A hardware and software system that allows the user to analyze the operation of the system under development. A debugger usually allows the developer to step through the firmware one step at a time, set break points, and analyze memory.
dead band	A period of time when neither of two or more signals are in their active state or in transition.
decimal	A base-10 numbering system, which uses the symbols 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9 (called digits) together with the decimal point and the sign symbols + (plus) and - (minus) to represent numbers.
default value	Pertaining to the pre-defined initial, original, or specific setting, condition, value, or action a system will assume, use, or take in the absence of instructions from the user.
device	The device referred to in this manual is the PSoC chip, unless otherwise specified.
die	An unpackaged integrated circuit (IC), normally cut from a wafer.
digital	A signal or function, the amplitude of which is characterized by one of two discrete values: ‘0’ or ‘1’.
digital blocks	The 8-bit logic blocks that can act as a counter, timer, serial receiver, serial transmitter, CRC generator, pseudo-random number generator, or SPI.
digital logic	A methodology for dealing with expressions containing two-state variables that describe the behavior of a circuit or system.
digital-to-analog (DAC)	A device that changes a digital signal to an analog signal of corresponding magnitude. The ***analog-to-digital (ADC)*** converter performs the reverse operation.
direct access	The capability to obtain data from a storage device, or to enter data into a storage device, in a sequence independent of their relative positions by means of addresses that indicate the physical location of the data.
duty cycle	The relationship of a clock period ***high time*** to its ***low time***, expressed as a percent.

E

emulator	Duplicates (provides an emulation of) the functions of one system with a different system, so that the second system appears to behave like the first system.
External Reset (XRES)	An active high signal that is driven into the PSoC device. It causes all operation of the CPU and blocks to stop and return to a pre-defined state.

F

falling edge	A transition from a logic 1 to a logic 0. Also known as a negative edge.
feedback	The return of a portion of the output, or processed portion of the output, of a (usually active) device to the input.
filter	A device or process by which certain frequency components of a signal are attenuated.

firmware The software that is embedded in a hardware device and executed by the CPU. The software may be executed by the end user but it may not be modified.

flag Any of various types of indicators used for identification of a condition or event (for example, a character that signals the termination of a transmission).

Flash An electrically programmable and erasable, non-***volatile*** technology that provides users with the programmability and data storage of EPROMs, plus in-system erasability. Non-volatile means that the data is retained when power is off.

Flash bank A group of Flash ROM blocks where Flash block numbers always begin with '0' in an individual Flash bank. A Flash bank also has its own block level protection information.

Flash block The smallest amount of Flash ROM space that may be programmed at one time and the smallest amount of Flash space that may be protected. A Flash block holds 64 bytes.

flip-flop A device having two stable states and two input terminals (or types of input signals) each of which corresponds with one of the two states. The circuit remains in either state until it is made to change to the other state by application of the corresponding signal.

frequency The number of cycles or events per unit of time, for a periodic function.

G

gain The ratio of output current, voltage, or power to input current, voltage, or power, respectively. Gain is usually expressed in dB.

ground

1. The electrical neutral line having the same potential as the surrounding earth.
2. The negative side of DC power supply.
3. The reference point for an electrical system.
4. The conducting paths between an electric circuit or equipment and the earth, or some conducting body serving in place of the earth.

H

hardware A comprehensive term for all of the physical parts of a computer or embedded system, as distinguished from the data it contains or operates on, and the software that provides instructions for the hardware to accomplish tasks.

hardware reset A reset that is caused by a circuit, such as a POR, watchdog reset, or external reset. A hardware reset restores the state of the device as it was when it was first powered up. Therefore, all registers are set to the POR value as indicated in register tables throughout this manual.

hexadecimal A base 16 numeral system (often abbreviated and called hex), usually written using the symbols 0-9 and A-F. It is a useful system in computers because there is an easy mapping from four bits to a single hex digit. Thus, one can represent every byte as two consecutive hexadecimal digits. Compare the binary, hex, and decimal representations:

bin	=	hex	=	dec
0000b	=	0x0	=	0
0001b	=	0x1	=	1
0010b	=	0x2	=	2
...				
1001b	=	0x9	=	9
1010b	=	0xA	=	10
1011b	=	0xB	=	11
...				
1111b	=	0xF	=	15

So the decimal numeral 79 whose binary representation is 0100 1111b can be written as 4Fh in hexadecimal (0x4F).

high time The amount of time the signal has a value of '1' in one period, for a periodic digital signal.

I

I2C A two-wire serial computer bus by Philips Semiconductors. I2C is an inter-integrated circuit. It is used to connect low-speed peripherals in an embedded system. The original system was created in the early 1980s as a battery control interface, but it was later used as a simple internal bus system for building control electronics. I2C uses only two bi-directional pins, clock and data, both running at +5V and pulled high with resistors. The bus operates at 100 kbits/second in standard mode and 400 kbits/second in fast mode. I^2C™ is a trademark of the Philips Semiconductors.

ICE The in-circuit emulator that allows users to test the project in a hardware environment, while viewing the debugging device activity in a software environment (PSoC Designer).

idle state A condition that exists whenever user messages are not being transmitted, but the service is immediately available for use.

impedance

1. The resistance to the flow of current caused by resistive, capacitive, or inductive devices in a circuit.
2. The total passive opposition offered to the flow of electric current. Note the impedance is determined by the particular combination of resistance, inductive reactance, and capacitive reactance in a given circuit.

input A point that accepts data in a device, process, or channel.

input/output (IO) A device that introduces data into or extracts data from a system.

instruction An expression that specifies one operation and identifies its operands, if any, in a programming language such as C or assembly.

integrated circuit (IC) A device in which components such as resistors, capacitors, diodes, and ***transistors*** are formed on the surface of a single piece of semiconductor.

interface The means by which two systems or devices are connected and interact with each other.

interrupt A suspension of a process, such as the execution of a computer program, caused by an event external to that process and performed in such a way that the process can be resumed.

interrupt service routine (ISR) A block of code that normal code execution is diverted to when the M8C receives a hardware interrupt. Many interrupt sources may each exist with its own priority and individual ISR code block. Each ISR code block ends with the RETI instruction, returning the device to the point in the program where it left normal program execution.

J

jitter

1. A misplacement of the timing of a transition from its ideal position. A typical form of corruption that occurs on serial data streams.
2. The abrupt and unwanted variations of one or more signal characteristics, such as the interval between successive pulses, the amplitude of successive cycles, or the frequency or phase of successive cycles.

K

keeper A circuit that holds a signal to the last driven value, even when the signal becomes un-driven.

L

latency The time or delay that it takes for a signal to pass through a given circuit or network.

least significant bit (LSb) The binary digit, or bit, in a binary number that represents the least significant value (typically the right-hand bit). The bit versus byte distinction is made by using a lower case “b” for bit in LSb.

least significant byte (LSB) The byte in a multi-byte word that represents the least significant value (typically the right-hand byte). The byte versus bit distinction is made by using an upper case “B” for byte in LSB.

Linear Feedback Shift Register (LFSR) A shift register whose data input is generated as an ***XOR*** of two or more elements in the register chain.

load The electrical demand of a process expressed as power (watts), current (amps), or resistance (ohms).

logic function A mathematical function that performs a digital operation on digital data and returns a digital value.

look-up table (LUT) A logic block that implements several logic functions. The logic function is selected by means of select lines and is applied to the inputs of the block. For example: A 2 input LUT with 4 select lines can be used to perform any one of 16 logic functions on the two inputs resulting in a single logic output. The LUT is a combinational device; therefore, the input/output relationship is continuous, that is, not sampled.

low time The amount of time the signal has a value of ‘0’ in one period, for a periodic digital signal.

low voltage detect (LVD) A circuit that senses Vdd and provides an interrupt to the system when Vdd falls below a selected threshold.

M

M8C An 8-bit Harvard Architecture microprocessor. The microprocessor coordinates all activity inside a PSoC by interfacing to the Flash, SRAM, and register space.

macro A programming language macro is an abstraction whereby a certain textual pattern is replaced according to a defined set of rules. The interpreter or compiler automatically replaces the macro instance with the macro contents when an instance of the macro is encountered. Therefore, if a macro is used 5 times and the macro definition required 10 bytes of code space, 50 bytes of code space will be needed in total.

mask
1. To obscure, hide, or otherwise prevent information from being derived from a signal. It is usually the result of interaction with another signal, such as noise, static, jamming, or other forms of interference.
2. A pattern of bits that can be used to retain or suppress segments of another pattern of bits in computing and data processing systems.

master device A device that controls the timing for data exchanges between two devices. Or when devices are cascaded in width, the master device is the one that controls the timing for data exchanges between the cascaded devices and an external interface. The controlled device is called the ***slave device***.

microcontroller An integrated circuit ***chip*** that is designed primarily for control systems and products. In addition to a CPU, a microcontroller typically includes memory, timing circuits, and IO circuitry. The reason for this is to permit the realization of a controller with a minimal quantity of chips, thus achieving maximal possible miniaturization. This, in turn, will reduce the volume and the cost of the controller. The microcontroller is normally not used for general-purpose computation as is a microprocessor.

mixed signal The reference to a circuit containing both analog and digital techniques and components.

mnemonic
1. A tool intended to assist the memory. Mnemonics rely on not only repetition to remember facts, but also on creating associations between easy-to-remember constructs and lists of data.
2. A two to four character string representing a microprocessor instruction.

mode A distinct method of operation for software or hardware. For example, the Digital PSoC block may be in either counter mode or timer mode.

modulation A range of techniques for encoding information on a carrier signal, typically a sine-wave signal. A device that performs modulation is known as a modulator.

Modulator A device that imposes a signal on a carrier.

MOS An acronym for metal-oxide semiconductor.

most significant bit (MSb) The binary digit, or bit, in a binary number that represents the most significant value (typically the left-hand bit). The bit versus byte distinction is made by using a lower case “b” for bit in MSb.

most significant byte (MSB) The byte in a multi-byte word that represents the most significant value (typically the left-hand byte). The byte versus bit distinction is made by using an upper case “B” for byte in MSB.

multiplexer (mux) 1. A logic function that uses a binary value, or address, to select between a number of inputs and conveys the data from the selected input to the output.
2. A technique which allows different input (or output) signals to use the same lines at different times, controlled by an external signal. Multiplexing is used to save on wiring and IO ports.

N

NAND See ***Boolean Algebra***.

negative edge A transition from a logic 1 to a logic 0. Also known as a falling edge.

net The routing between devices.

nibble A group of four bits, which is one-half of a byte.

noise 1. A disturbance that affects a signal and that may distort the information carried by the signal.
2. The random variations of one or more characteristics of any entity such as voltage, current, or data.

NOR See ***Boolean Algebra***.

NOT See ***Boolean Algebra***.

O

OR See ***Boolean Algebra***.

oscillator A circuit that may be crystal controlled and is used to generate a clock frequency.

output The electrical signal or signals which are produced by an analog or digital block.

P

parallel The means of communication in which digital data is sent multiple bits at a time, with each simultaneous bit being sent over a separate line.

parameter Characteristics for a given block that have either been characterized or may be defined by the designer.

parameter block A location in memory where parameters for the SSC instruction are placed prior to execution.

parity A technique for testing transmitting data. Typically, a binary digit is added to the data to make the sum of all the digits of the binary data either always even (even parity) or always odd (odd parity).

path 1. The logical sequence of instructions executed by a computer.
2. The flow of an electrical signal through a circuit.

pending interrupts An interrupt that has been triggered but has not been serviced, either because the processor is busy servicing another interrupt or global interrupts are disabled.

phase	The relationship between two signals, usually the same frequency, that determines the delay between them. This delay between signals is either measured by time or angle (degrees).
Phase-Locked Loop (PLL)	An electronic circuit that controls an ***oscillator*** so that it maintains a constant phase angle relative to a reference signal.
pin	A terminal on a hardware component. Also called lead.
pinouts	The pin number assignment: the relation between the logical inputs and outputs of the PSoC device and their physical counterparts in the printed circuit board (PCB) package. Pinouts will involve pin numbers as a link between schematic and PCB design (both being computer generated files) and may also involve pin names.
port	A group of pins, usually eight.
positive edge	A transition from a logic 0 to a logic 1. Also known as a rising edge.
posted interrupts	An interrupt that has been detected by the hardware but may or may not be enabled by its mask bit. Posted interrupts that are not masked become pending interrupts.
Power On Reset (POR)	A circuit that forces the PSoC device to reset when the voltage is below a pre-set level. This is one type of ***hardware reset***.
program counter	The instruction pointer (also called the program counter) is a register in a computer processor that indicates where in memory the CPU is executing instructions. Depending on the details of the particular machine, it holds either the address of the instruction being executed or the address of the next instruction to be executed.
protocol	A set of rules. Particularly the rules that govern networked communications.
PSoC	Cypress Semiconductor's Programmable System-on-Chip (PSoC) mixed-signal array. PSoC® and Programmable System-on-Chip™ are trademarks of Cypress.
PSoC blocks	*See* ***analog blocks*** and ***digital blocks***.
PSoC Designer	The software for Cypress' Programmable System-on-Chip™ technology.
pulse	A rapid change in some characteristic of a signal (for example, phase or frequency) from a baseline value to a higher or lower value, followed by a rapid return to the baseline value.
pulse width modulator (PWM)	An output in the form of duty cycle which varies as a function of the applied measurand.

R

RAM	An acronym for random access memory. A data-storage device from which data can be read out and new data can be written in.
register	A storage device with a specific capacity, such as a bit or byte.
reset	A means of bringing a system back to a know state. See ***hardware reset*** and ***software reset***.
resistance	The resistance to the flow of electric current measured in ohms for a conductor.

revision ID A unique identifier of the PSoC device.

ripple divider An asynchronous ripple counter constructed of flip-flops. The clock is fed to the first stage of the counter. An n-bit binary counter consisting of n flip-flops that can count in binary from 0 to 2^n - 1.

rising edge *See* ***positive edge***.

ROM An acronym for read only memory. A data-storage device from which data can be read out, but new data cannot be written in.

routine A block of code, called by another block of code, that may have some general or frequent use.

routing Physically connecting objects in a design according to design rules set in the reference library.

runt pulses In digital circuits, narrow pulses that, due to non-zero rise and fall times of the signal, do not reach a valid high or low level. For example, a runt pulse may occur when switching between asynchronous clocks or as the result of a race condition in which a signal takes two separate paths through a circuit. These race conditions may have different delays and are then recombined to form a glitch or when the output of a flip-flop becomes metastable.

S

sampling The process of converting an analog signal into a series of digital values or reversed.

schematic A diagram, drawing, or sketch that details the elements of a system, such as the elements of an electrical circuit or the elements of a logic diagram for a computer.

seed value An initial value loaded into a linear feedback shift register or random number generator.

serial
1. Pertaining to a process in which all events occur one after the other.
2. Pertaining to the sequential or consecutive occurrence of two or more related activities in a single device or channel.

set To force a bit/register to a value of logic 1.

settling time The time it takes for an output signal or value to stabilize after the input has changed from one value to another.

shift The movement of each bit in a word, one position to either the left or right. For example, if the hex value 0x24 is shifted one place to the left, it becomes 0x48. If the hex value 0x24 is shifted one place to the right, it becomes 0x12.

shift register A memory storage device that sequentially shifts a word either left or right to output a stream of serial data.

sign bit The most significant binary digit, or bit, of a signed binary number. If set to a logic 1, this bit represents a negative quantity.

signal A detectable transmitted energy that can be used to carry information. As applied to electronics, any transmitted electrical impulse.

silicon ID A unique identifier of the PSoC silicon.

skew	The difference in arrival time of bits transmitted at the same time, in parallel transmission.
slave device	A device that allows another device to control the timing for data exchanges between two devices. Or when devices are cascaded in width, the slave device is the one that allows another device to control the timing of data exchanges between the cascaded devices and an external interface. The controlling device is called the master device.
software	A set of computer programs, procedures, and associated documentation concerned with the operation of a data processing system (for example, compilers, library routines, manuals, and circuit diagrams). Software is often written first as source code and then converted to a binary format that is specific to the device on which the code will be executed.
software reset	A partial reset executed by software to bring part of the system back to a known state. A software reset will restore the M8C to a know state but not PSoC blocks, systems, peripherals, or registers. For a software reset, the CPU registers (CPU_A, CPU_F, CPU_PC, CPU_SP, and CPU_X) are set to 0x00. Therefore, code execution will begin at Flash address 0x0000.
SRAM	An acronym for static random access memory. A memory device allowing users to store and retrieve data at a high rate of speed. The term static is used because, once a value has been loaded into an SRAM cell, it will remain unchanged until it is explicitly altered or until power is removed from the device.
SROM	An acronym for supervisory read only memory. The SROM holds code that is used to boot the device, calibrate circuitry, and perform Flash operations. The functions of the SROM may be accessed in normal user code, operating from Flash.
stack	A stack is a data structure that works on the principle of Last In First Out (LIFO). This means that the last item put on the stack is the first item that can be taken off.
stack pointer	A stack may be represented in a computer's inside blocks of memory cells, with the bottom at a fixed location and a variable stack pointer to the current top cell.
state machine	The actual implementation (in hardware or software) of a function that can be considered to consist of a set of states through which it sequences.
sticky	A bit in a register that maintains its value past the time of the event that caused its transition has passed.
stop bit	A signal following a character or block that prepares the receiving device to receive the next character or block.
switching	The controlling or routing of signals in circuits to execute logical or arithmetic operations, or to transmit data between specific points in a network.
Switch phasing	The clock that controls a given switch, PHI1 or PHI2, in respect to the switch capacitor (SC) blocks. The PSoC SC blocks have two groups of switches. One group of these switches is normally closed during PHI1 and open during PHI2. The other group is open during PHI1 and closed during PHI2. These switches can be controlled in the normal operation, or in reverse mode if the PHI1 and PHI2 clocks are reversed.
synchronous	1. A signal whose data is not acknowledged or acted upon until the next active edge of a clock signal. 2. A system whose operation is synchronized by a clock signal.

T

tap The connection between two blocks of a device created by connecting several blocks/components in a series, such as a shift register or resistive voltage divider.

terminal count The state at which a counter is counted down to zero.

threshold The minimum value of a signal that can be detected by the system or sensor under consideration.

transistors The transistor is a solid-state semiconductor device used for amplification and switching, and has three terminals. A small current or voltage applied to one terminal controls the current through the other two. It is the key component in all modern electronics. In digital circuits, transistors are used as very fast electrical switches, and arrangements of transistors can function as logic gates, RAM-type memory, and other devices. In analog circuits, transistors are essentially used as amplifiers.

tri-state A function whose output can adopt three states: 0, 1, and Z (high-impedance). The function does not drive any value in the Z state and, in many respects, may be considered to be disconnected from the rest of the circuit, allowing another output to drive the same ***net***.

U

UART A UART or universal asynchronous receiver-transmitter translates between parallel bits of data and serial bits.

user The person using the PSoC device and reading this manual.

user modules Pre-build, pre-tested hardware/firmware peripheral functions that take care of managing and configuring the lower level analog and digital PSoC blocks. User modules also provide high level ***API (Application Programming Interface)*** for the peripheral function.

user space The bank 0 space of the register map. The registers in this bank are more likely to be modified during normal program execution and not just during initialization. Registers in bank 1 are most likely to be modified only during the initialization phase of the program.

V

Vdd A name for a power net meaning “voltage drain.” The most positive power supply signal. Usually 5 or 3.3 volts.

volatile Not guaranteed to stay the same value or level when not in scope.

Vss A name for a power net meaning “voltage source.” The most negative power supply signal.

W

watchdog timer A timer that must be serviced periodically. If it is not serviced, the CPU will reset after a specified period of time.

waveform The representation of a signal as a plot of amplitude versus time.

X

XOR See ***Boolean Algebra***.

Index

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W

X

Z