Welcome to E-XFL.COM

Embedded - Microcontrollers - Application Specific: Tailored Solutions for Precision and Performance

Embedded - Microcontrollers - Application Specific represents a category of microcontrollers designed with unique features and capabilities tailored to specific application needs. Unlike general-purpose microcontrollers, application-specific microcontrollers are optimized for particular tasks, offering enhanced performance, efficiency, and functionality to meet the demands of specialized applications.

What Are Embedded - Microcontrollers - Application Specific?

Application-specific microcontrollers are engineered to

Details	
Product Status	Obsolete
Applications	Touchscreen Controller
Core Processor	M8C
Program Memory Type	FLASH (32kB)
Controller Series	CY8CT
RAM Size	2K x 8
Interface	I²C, SPI, UART/USART, USB
Number of I/O	28
Voltage - Supply	1.8V
Operating Temperature	-40°C ~ 85°C
Mounting Type	Surface Mount
Package / Case	32-VFQFN Exposed Pad
Supplier Device Package	32-QFN
Purchase URL	https://www.e-xfl.com/product-detail/infineon-technologies/cy8ctst200a-32lqxi

Top Level Architecture

The PSoC block diagram on the next page illustrates the top-level architecture of the CY8CTMG20x and CY8CTST200 devices. Each major grouping in the diagram is covered in this manual in its own section: PSoC Core, TrueTouch 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 I/Os, 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 24 MHz. The M8C is a four MIPS 8-***bit*** Harvard architecture microprocessor. Within the CPU core are the ***SROM*** and ***Flash*** memory components that provide flexible programming.

PSoC GPIOs provide connection to the CPU and the TrueTouch resources of the device. Each pin's drive mode is selectable 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.

TrueTouch™ System

The TrueTouch System is composed of comparators, reference drivers, I/O multiplexers, and digital logic to support various capacitive sensing algorithms. Various reference selections are provided. Digital logic is mainly comprised of counters and timers.

System Resources

The System Resources provide additional PSoCcapability. These system resources include:

- Digital clocks to increase the flexibility of the PSoC programmable system-on-chip.
- I2C functionality with "no bus stalling."
- 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.
- Clock boost network providing a stronger signal to switches.
- Full-speed USB interface for USB 2.0 communication with 512 bytes of dedicated buffer memory and an internal 3V regulator.

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

The following 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 187.

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

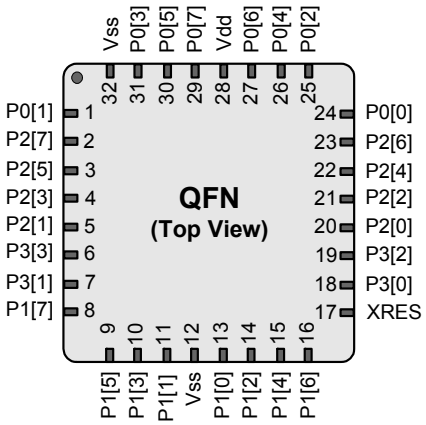
Symbol	Unit of Measure
°C	degree Celsius
dB	decibels
fF	femtofarad
Hz	hertz
k	kilo, 1000
K	2^{10}, 1024
KB	1024 bytes
Kbit	1024 bits
kHz	kilohertz (32.000)
kΩ	kilohm
MHz	megahertz
MΩ	megaohm
µA	microampere
µF	microfarad
µs	microsecond
µV	microvolt
µVrms	microvolts root-mean-square
mA	milliampere
ms	millisecond
mV	millivolt
nA	nanoamphere
ns	nanosecond
nV	nanovolt
Ω	ohm
pF	picofarad
pp	peak-to-peak
ppm	parts per million
sps	samples per second
σ	sigma: one standard deviation
V	volt

1.1.3 CY8CTMG200-32LQXI, CY8CTMG200A-32LQXI, CY8CTST200-32LQXI, CY8CTST200A-32LQXI, CY8CTMG201-32LQXI, CY8CTMG201A-32LQXI PSoC 32-Pin Part Pinout

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

Pin No.	Digital	Analog	Name	Description
1	IOH	I	P0[1]	Integrating input
2	IO	I	P2[7]	
3	IO	I	P2[5]	XTAL Out
4	IO	I	P2[3]	XTAL In
5	IO	I	P2[1]	
6	IO	I	P3[3]	
7	IO	I	P3[1]	
8	IOHR	I	P1[7]	I2C SCL, SPI SS
9	IOHR	I	P1[5]	I2C SDA, SPI MISO
10	IOHR	I	P1[3]	SPI CLK
11	IOHR	I	P1[1]	TC CLK*, I2C SCL, SPI MOSI
12	Power		Vss	Ground pin
13	IOHR	I	P1[0]	TC DATA*, I2C SDA, SPI CLK
14	IOHR	I	P1[2]	
15	IOHR	I	P1[4]	EXTCLK
16	IOHR	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	IOH	I	P0[0]	
25	IOH	I	P0[2]	
26	IOH	I	P0[4]	
27	IOH	I	P0[6]	
28	Power		Vdd	Power pin
29	IOH	I	P0[7]	
30	IOH	I	P0[5]	
31	IOH	I	P0[3]	
32	Power		Vss	Ground pin

CY8CTMG200-32LQXI, CY8CTMG200A-32LQXI, CY8CTST200-32LQXI, CY8CTST200A-32LQXI, CY8CTMG201-32LQXI, CY8CTMG201A-32LQXI PSoC Devices



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

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 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	if (A=B) Z=1 if (A<B) C=1	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]		68	7	2	ASR [expr]	C, Z
0F	10	3	ADC [X+expr], expr	C, Z	3C	8	3	CMP [expr], expr		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 128 byte page boundaries in the Flash memory space.

3. 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 187.

3.1 Architectural Description

The SROM holds code that boots a PSoC device, calibrates circuitry, and performs 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 blocks. (Table 3-1 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 wanted SROM function code from Table 3-1.

Attempting to access undefined functions (Reserved 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 3-2 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 39 for more details regarding the MVR_PP and MVW_PP pointers.)

Table 3-1. List of SROM Functions

Function Code	Function Name	Required Stack Space	Page
00h	SWBootReset	0	34
01h	ReadBlock	7	35
02h	WriteBlock	7	35
03h	EraseBlock	5	36
06h	TableRead	3	36
07h	CheckSum	4	37
08h	Calibrate0	4	37
09h	Calibrate1	3	37
0Ah	WriteAndVerify	7	37
0Fh	HWBootReset	3	38

Note ProtectBlock and EraseAll (described on page 36) SROM functions are not listed in the table above because they are dependent on external programming.

Table 3-2. 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

Note CLOCK and DELAY are ignored and are reserved for future use.

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 is 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 halts. The SWBootReset function does not check the key values. It only checks to see if the accumulator's value is 0x00.

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 252.

5.3.9 Related Registers

- CPU_F on page 254.

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) 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 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]. Drive modes are shown in Table 6-3.

For analog I/O, set the drive mode 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 is always read as a zero by the CPU and the pin cannot generate a useful interrupt. (It is not strictly required that you select High Z mode for analog operation.)

When digital inputs are needed on the same pin as analog inputs, use the 11b Drive mode 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 259, and the PRTxDM1 register on page 260.

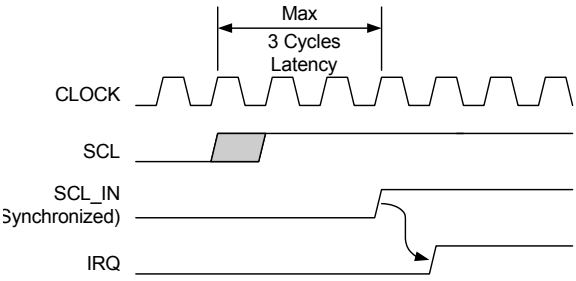
Summary Table of the System Resource Registers *(continued)*

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access
0,46h	EPx_CNT0	Data Toggle	Data Valid						Count MSB	# : 0
0,48h	EPx_CNT0	Data Toggle	Data Valid						Count MSB	# : 0
0,4Ah	EPx_CNT0	Data Toggle	Data Valid						Count MSB	# : 0
0,4Ch	EPx_CNT0	Data Toggle	Data Valid						Count MSB	# : 0
0,4Eh	EPx_CNT0	Data Toggle	Data Valid						Count MSB	# : 0
1,30h	USB_CR1						BusActiv-ity	Enable-Lock	RegEnable	RW : 0
1,34h	PMAx_WA	Write Address[7:0]								RW : 00
1,35h	PMAx_WA	Write Address[7:0]								RW : 00
1,36h	PMAx_WA	Write Address[7:0]								RW : 00
1,37h	PMAx_WA	Write Address[7:0]								RW : 00
1,38h	PMAx_WA	Write Address[7:0]								RW : 00
1,39h	PMAx_WA	Write Address[7:0]								RW : 00
1,3Ah	PMAx_WA	Write Address[7:0]								RW : 00
1,3Bh	PMAx_WA	Write Address[7:0]								RW : 00
1,44h	PMAx_WA	Write Address[7:0]								RW : 00
1,45h	PMAx_WA	Write Address[7:0]								RW : 00
1,46h	PMAx_WA	Write Address[7:0]								RW : 00
1,47h	PMAx_WA	Write Address[7:0]								RW : 00
1,48h	PMAx_WA	Write Address[7:0]								RW : 00
1,49h	PMAx_WA	Write Address[7:0]								RW : 00
1,4Ah	PMAx_WA	Write Address[7:0]								RW : 00
1,4Bh	PMAx_WA	Write Address[7:0]								RW : 00
1,3Ch	PMAx_RA	Read Address[7:0]								RW : 00
1,3Dh	PMAx_RA	Read Address[7:0]								RW : 00
1,3Eh	PMAx_RA	Read Address[7:0]								RW : 00
1,3Fh	PMAx_RA	Read Address[7:0]								RW : 00
1,40h	PMAx_RA	Read Address[7:0]								RW : 00
1,41h	PMAx_RA	Read Address[7:0]								RW : 00
1,42h	PMAx_RA	Read Address[7:0]								RW : 00
1,43h	PMAx_RA	Read Address[7:0]								RW : 00
1,4Ch	PMAx_RA	Read Address[7:0]								RW : 00
1,4Dh	PMAx_RA	Read Address[7:0]								RW : 00
1,4Eh	PMAx_RA	Read Address[7:0]								RW : 00
1,4Fh	PMAx_RA	Read Address[7:0]								RW : 00
1,50h	PMAx_RA	Read Address[7:0]								RW : 00
1,51h	PMAx_RA	Read Address[7:0]								RW : 00
1,52h	PMAx_RA	Read Address[7:0]								RW : 00
1,53h	PMAx_RA	Read Address[7:0]								RW : 00
1,54h	EPx_CR0	Stall		NAK_INT_EN	ACKed Tx	Mode[3:0]				# : 00
1,55h	EPx_CR0	Stall		NAK_INT_EN	ACKed Tx	Mode[3:0]				# : 00
1,56h	EPx_CR0	Stall		NAK_INT_EN	ACKed Tx	Mode[3:0]				# : 00
1,57h	EPx_CR0	Stall		NAK_INT_EN	ACKed Tx	Mode[3:0]				# : 00
1,58h	EPx_CR0	Stall		NAK_INT_EN	ACKed Tx	Mode[3:0]				# : 00
1,59h	EPx_CR0	Stall		NAK_INT_EN	ACKed Tx	Mode[3:0]				# : 00
1,5Ah	EPx_CR0	Stall		NAK_INT_EN	ACKed Tx	Mode[3:0]				# : 00
1,5Bh	EPx_CR0	Stall		NAK_INT_EN	ACKed Tx	Mode[3:0]				# : 00

15.4.3 Status Timing

Figure 15-8 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 15-8. Byte Complete, Address, LRB Timing



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

Figure 15-9 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 15-9. Stop Status and Interrupt Timing

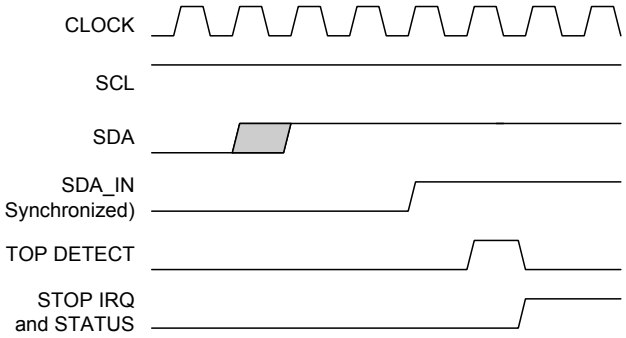


Figure 15-10 illustrates the timing for bus error interrupts. Bus Error status (and interrupt) occurs one cycle after the internal Start or Stop detect (two cycles after the filtered and synchronized SDA input transition).

Figure 15-10. Bus Error Interrupt Timing

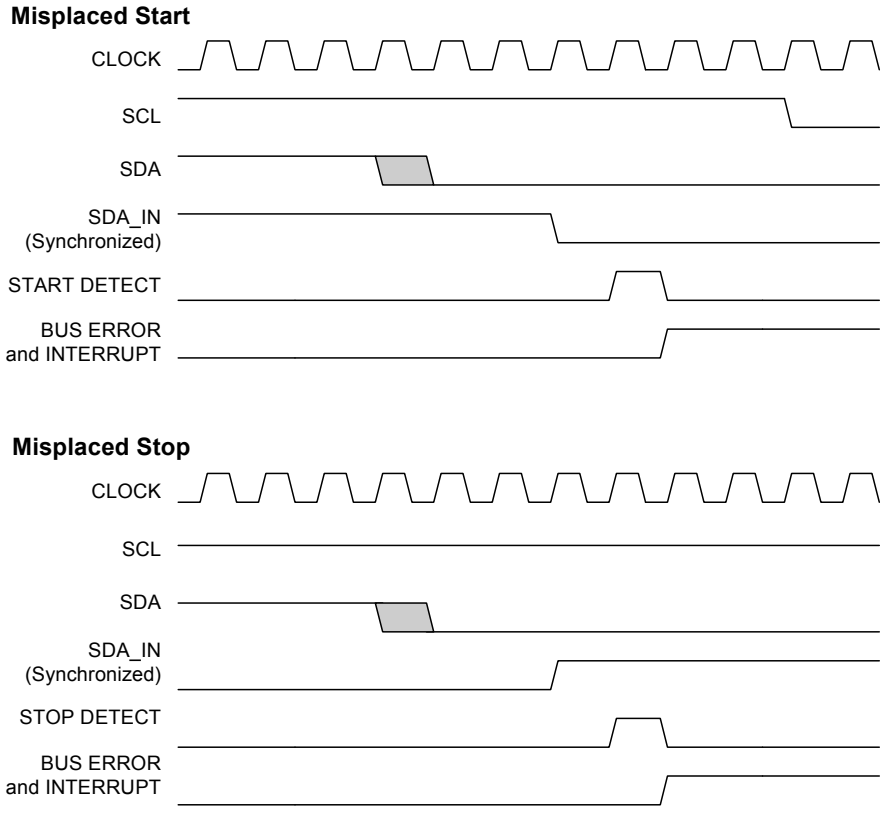
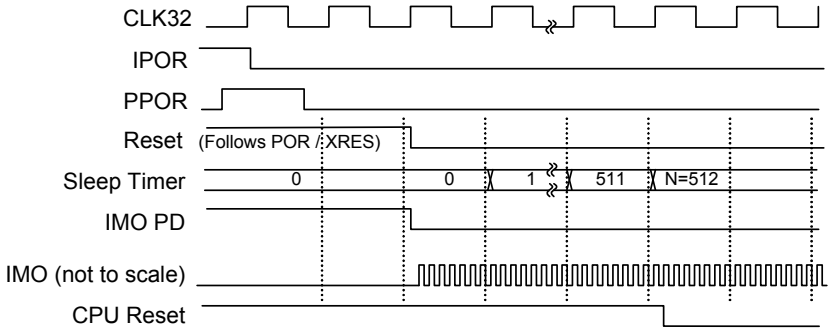
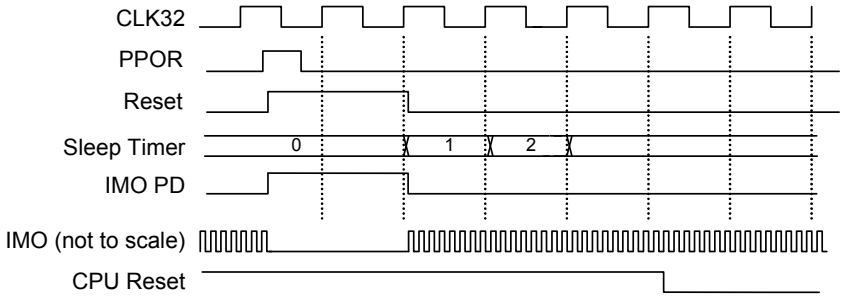


Figure 16-4. Key Signals During POR and XRES

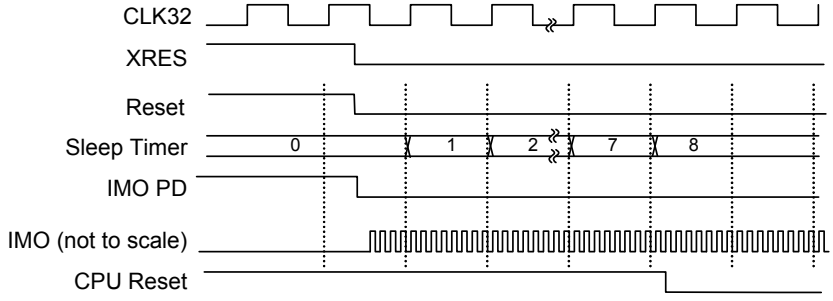
POR (IPOR followed by PPOR): Reset while POR is high (IMO off), then 511(+) cycles (IMO on), and then the CPU reset is released. **XRES** is the same, with N=8.



PPOR (with no IPOR): Reset while PPOR is high and to the end of the next 32K cycle (IMO off); 1 cycle IMO on before the CPU reset is released. Note that at the 3V level, PPOR tends to be brief because the reset clears the POR range register (VLT_CR) back to the default 2.4V setting.



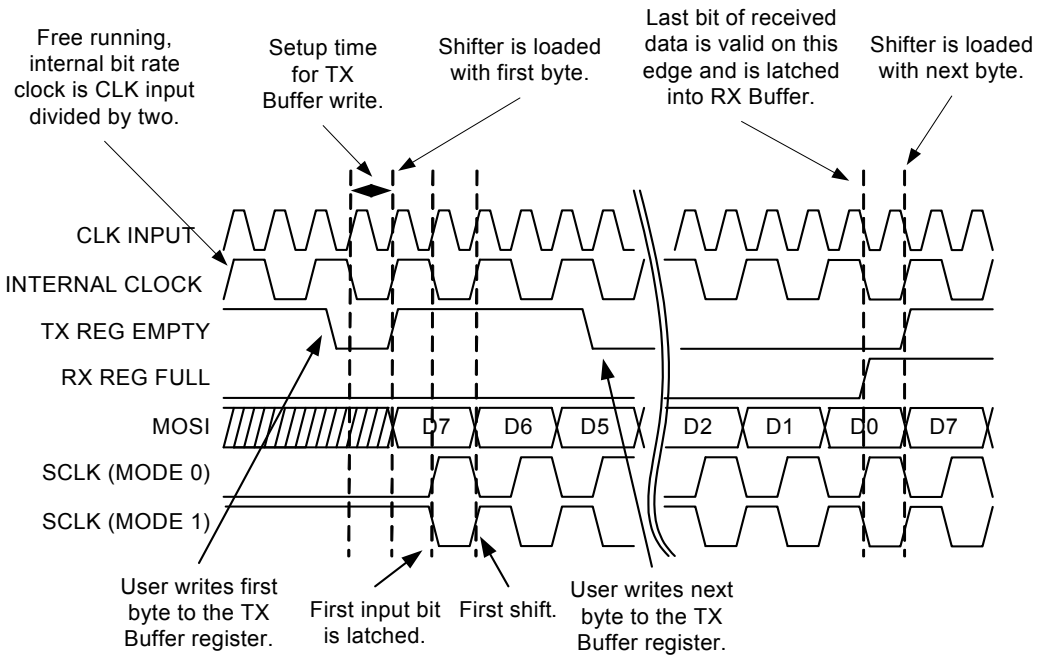
XRES: Reset while XRES is high (IMO off), then 7(+) cycles (IMO on), and then the CPU reset is released.



Normal Operation. Typical timing for an 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 is initiated. An SPIM block receives a byte at the same time that it sends one. The SPI Complete or RX Reg Full can be used to determine when the input byte has been received.

Figure 18-5. Typical SPIM Timing in Mode 0 and 1



(Self-powered devices do not need to go into Suspend mode.) This condition is detected by monitoring the Bus Activity bit in the USB_CR1 register. This bit must be polled periodically. If it reads high (bus activity present), it must be cleared by firmware. If no activity is detected for the desired time (for example, 3 ms), then the device must enter Suspend mode.

Not all sleep modes preserve the USB configuration register states during sleep. The Standby I2C-USB mode is the preferred sleep mode for USB operation because the state of all USB registers is maintained during sleep. The other sleep modes do not preserve all of these registers to save sleep power.

The USB regulator settings must not be changed when entering sleep state, since the regulator automatically enters a low power state for the given regulator mode (pass-through or regulating).

20.2.5.1 Using Standby I2C-USB Sleep Mode for USB Suspend

To enter Standby I2C-USB mode, firmware powers down the desired functions, as it would to enter a Standby I2C-USB mode sleep state as documented in 10.1.1 Sleep Control Implementation Logic on page 74, including writing the SLEEP bit of the CPU_SCR0 register. In this mode, USB configuration registers and data are preserved during sleep.

20.2.5.2 Using Standby or Deep Sleep Modes for USB Suspend

The Standby and Deep Sleep modes do not have hardware supported for USB suspend operation because not all USB registers are preserved in these modes. In USB parts, if there is a need to enter Standby or Deep Sleep mode, then user firmware must save the non-retained registers to SRAM before entering into sleep and restore these registers once the device wakes up. The list of these registers is given below. Some configuration registers retain their values in all sleep modes: USB_CR0, USB_CR1, USBIO_CR0, USBIO_CR1, IMO_TR, IMO_TR1. In addition, the USB SRAM contents are preserved in all sleep modes.

The USB registers that retain state in Standby I2C-USB mode but not in other sleep modes are: the endpoint control registers (EPx_CRx), endpoint PMA write address (PMAx_WA)/read address (PMAx_RA) registers, PMA data registers (PMAx_DR), endpoint count registers (EPx_CNTx), endpoint 0 data register (EP0_DRx) and start of frame registers (USB_SOFx). These registers are reset after the device comes out of sleep.

(An alternative is to simply disconnect from the USB bus before going into one of these sleep modes, and then re-connect to re-initialize the USB system after waking up.)

20.2.5.3 Wakeup from Suspend

The USB wake interrupt must be enabled to allow the device to exit the sleep state when there is activity on the USB bus. This interrupt can be enabled at any time since it only asserts when the device is in the sleep state. Other interrupts may be optionally enabled, such as the sleep interrupt, GPIO, I2C, to periodically wake the device while in USB suspend state. The USB wake interrupt can wake up the device from all the three sleep states (Standby sleep, I2C-USB sleep and Deep Sleep). If D+ is low when the SLEEP bit is being set, the device briefly enters sleep state, and then exits sleep due to the USB wake interrupt.

By carefully using a sleep timer interrupt, the device can wake periodically, monitor the environment, and return to sleep while maintaining a low average current that meets the USB suspend current specification.

If the device needs to issue a resume signal to the USB system, firmware can write to the TEN and TD bits in the USBIO_CR0 register to manually force a K state on the bus. Using these bits produces signaling that meets the USB timing specifications.

When driving a resume, the J state (TD=1) must be driven briefly before driving the K state (TD=0). The steps are summarized as follows:

1. Drive the J state (TEN=1, TD=1) for one instruction.
2. Drive the resume, or K state (TEN=1, TD=0) for the proper time (1 ms to 15 ms).
3. Stop driving the USB bus manually (TEN=0).

20.2.6 Regulator

The transceiver contains a built-in regulator that can be used to power the transceiver from the USB bus voltage or other supply around 5V. The regulator supplies the proper levels for USB signals, which switch between 0V and 3.3V nominally. If the PSoC device is operating with a Vdd supply near 3.3V, then the regulator must be placed into a pass-through mode so that the Vdd voltage is directly supplied to the transceiver, without regulation. The RegEnable bit (bit 0 in the USB_CR1 register) is used to pick between the regulating mode (5V supply) or the passthrough mode (3.3V supply). At power up, the regulator is automatically held in pass-through mode, but the USB transceiver pins are tristated.

21. 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 188. Bank 1 registers are listed second and begin on page 259. A condensed view of all the registers is shown in the register mapping tables starting on page 183.

21.1 Maneuvering Around the Registers

For ease-of-use, this chapter has been 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 must 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.

21.2 Register Conventions

The following 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.

21.3.33 PT0_CFG

Programmable Timer 0 Configuration Register

Individual Register Names and Addresses:

PT0_CFG : 0,B0h

	7	6	5	4	3	2	1	0
Access : POR						RW : 0	RW : 0	RW : 0
Bit Name						CLKSEL	One Shot	START

This register configures the programmable timer 0.

In the table above, note that reserved bits are grayed table cells and are not described in the bit description section below. Reserved bits must always be written with a value of '0'. For additional information, refer to the Register Definitions on page 163 in the Programmable Timer chapter.

Bit	Name		Description
2	**CLKSEL**		This bit determines if the timer runs on the 32 kHz clock or CPU clock. If the bit is set to 1'b1, the timer runs on the CPU clock, otherwise, the timer runs on the 32 kHz clock.
1	**One Shot**	0	Continuous count mode. Timer reloads the count value from the data registers upon each terminal count, and continues counting.
		1	One-shot mode. Timer goes through one complete count period and then stops. Upon 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.

21.3.34 PTx_DATA1

Programmable Timers Data Register 1

Individual Register Names and Addresses:

PT0_DATA1 : 0,B1h PT1_DATA1 : 0,B4h PT2_DATA1 : 0,B7h

	7	6	5	4	3	2	1	0
Access : POR	RW : 00							
Bit Name	DATA[7:0]							

These registers hold the eight bits of the progammable timer's count value for the device.

For additional information, refer to the Register Definitions on page 163 in the Programmable Timer chapter.

Bit	Name	Description
7:0	**DATA[7:0]**	This is the upper byte of a 16-bit timer. The lower byte is in the corresponding PTx_DATA0 register.

21.3.41 I2C_BP

I²C Base Address Pointer Register

Individual Register Names and Addresses:

I2C_BP : 0,CBh

	7	6	5	4	3	2	1	0
Access : POR				R : 00				
Bit Name				I2C Base Pointer[4:0]				

This register contains the base address value of the RAM data buffer and is read only.

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 122 in the I2C Slave chapter.

Bit	Name	Description
4:0	**I2C Base Pointer[4:0]**	In the EZI2C protocol, the first data byte after the slave address transaction in write mode is the base address for subsequent reads and writes and it is transferred directly into this register. If the desired transaction is a master write to the slave, subsequent bytes are written to the RAM buffer starting with this address and auto incremented (see I2C_CP register). In case of a read, a Start or Restart must be issued and the read location starts with this address and again subsequent read addresses are auto incremented as pointed to by the I2C_CP register value. The value of this register is modified only at the beginning of every I2C write transaction. The I2C master must always supply a value for this register in the first byte of data after the slave's address in a given write transaction. If performing reads, the master need not set the value of this register. The current value of this register is also used directly for reads.

21.3.65 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[1:0]				IRAMDIS

This register is used to convey 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 below. Reserved bits must always be written with a value of '0'. For additional information, refer to the Register Definitions on page 137 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.
4:3	**SLIMO[1:0]**	These bits set the frequency range for the IMO. **Note** When changing from the default setting, the corresponding trim value must be loaded into the IMO_TR register for highest frequency accuracy. **SLIMO** **CY8CTMG20x/ CY8CTST200** 00 12 01 6 10 24 11 Reserved
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.

21.4.14 IO_CFG1

Input/Output Configuration Register 1

Individual Register Names and Addresses:

IO_CFG1 : 1,DCh

	7	6	5	4	3	2	1	0
Access : POR	RW : 0		RW : 0		RW : 0	RW : 0	RW : 0	RW : 0
Bit Name	StrongP		Range[1:0]		P1_LOW_ THRS	SPICLK_ ON_P10	REG_EN	IO INT

This register is used to configure 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 below. Reserved bits must always be written with a value of '0'. For additional information, refer to the Register Definitions on page 59 in the GPIO chapter.

Bits	Name	Description
7	**StrongP**	Setting this bit increases the drive strength and edge ratio for high outputs.
5:4	**Range[1:0]**	Selects the high output level for Port 1 outputs. 00 3.0 volts 01 3.0 volts 10 1.8 volts 11 2.5 volts
3	**P1-LOW_THRS**	This bit reduces the threshold voltage of the P1 port input buffers so that there are no compatibility issues when Port 1 is communicating at regulated voltage levels. 0 Standard threshold of VIH, VIL 1 Reduce threshold of VIH, VIL
2	**SPICLK_ON_P10**	When set to '1', the SPI clock is mapped to Port 1 pin 0. Otherwise, it is mapped to Port 1 pin 3.
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	**IO INT**	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.

21.4.22 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	RW : 00							
Bit Name	Trim[7:0]							

This register is used to manually center the Internal Main Oscillator's (IMO) output to a target frequency.

It is strongly recommended that you do not alter this register's values except to load factory trim settings when changing IMO range.

When changing ranges, the new trim value for this range must be read from Flash using a Table Read operation. The new value must be written at the lower frequency range. That is, when moving to a higher frequency range, change the IMO_TR value and then change the range (SLIMO[1:0] in CPU_SCR1). When moving to a lower frequency, change the range first and then update IMO_TR.

For additional information, refer to the Register Definitions on page 64 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.

T

U

V