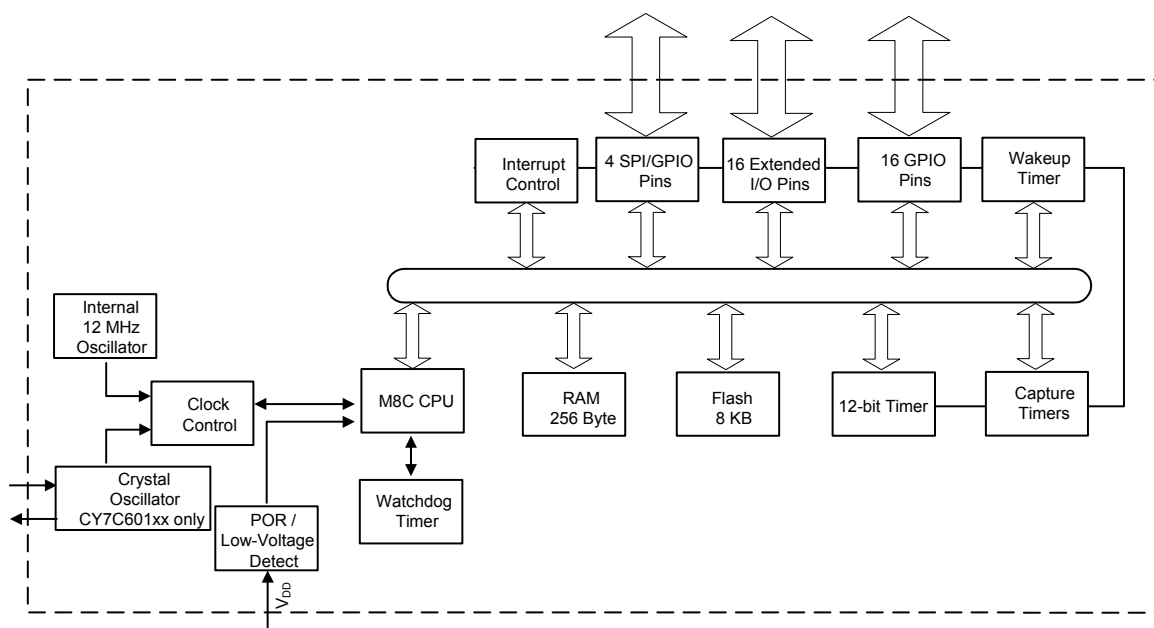# enCoRe™ II Low-Voltage Microcontroller

1. Features

- enCoRe™ II low-voltage (enCoRe II LV) – enhanced component reduction
  - Internal crystalless oscillator with support for optional external clock or external crystal or resonator
  - Configurable I/O for real world interface without external components
- Enhanced 8-bit microcontroller
  - Harvard architecture
  - M8C CPU speed up to 12 MHz or sourced by an external crystal, resonator, or clock signal
- Internal memory
  - 256 bytes of random access memory (RAM)
  - 8 KB of flash including electrically erasable read only memory (EEROM) emulation
- Low power consumption
  - Typically 2.25 mA at 3 MHz
  - 5 µA sleep
- In-system reprogrammability
  - Enables easy firmware update
- General-purpose I/O (GPIO) ports
  - Up to 36 GPIO pins
  - 2-mA source current on all GPIO pins
  - Configurable 8 or 50 mA per pin current sink on designated pins
  - Each GPIO port supports high-impedance inputs, configurable pull-up, open drain output, complementary metal oxide semiconductor (CMOS), and transistor-transistor logic (TTL) inputs, and CMOS output
  - Maskable interrupts on all I/O pins

- SPI serial communication
  - Master or slave operation
  - Configurable up to 2 Mbit per second transfers
  - Supports half-duplex single-data line mode for optical sensors
- 2-channel 8-bit or 1-channel 16-bit capture timer registers, which store both rising and falling edge times
  - Two registers each for two input pins
  - Separate registers for rising and falling edge capture
  - Simplifies interface to radio frequency (RF) inputs for wireless applications
- Internal low-power wakeup timer during suspend mode
  - Periodic wakeup with no external components
- Programmable interval timer interrupts
- Reduced RF emissions at 27 MHz and 96 MHz
- Watchdog timer (WDT)
- Low-voltage detection (LVD) with user-selectable threshold voltages
- Improved output drivers to reduce electromagnetic interference (EMI)
- Operating voltage from 2.7 V to 3.6 V DC
- Operating temperature from 0 °C to 70 °C
- Available in 40-pin plastic dual inline package (PDIP), 24-pin small outline integrated circuit (SOIC), 24-pin quad small outline package (QSOP) and shrink small outline package (SSOP), 48-pin SSOP
- Advanced development tools based on Cypress PSoC® tools
- Industry-standard programmer support

2. Logic Block Diagram

3. Contents

4. Applications

The CY7C601xx and CY7C602xx are targeted for the following applications:

- PC wireless human interface devices (HID)
  - Mice (optomechanical, optical, trackball)
  - Keyboards
  - Presenter tools

- Gaming
  - Joysticks
  - Gamepad

- General-purpose wireless applications
  - Remote controls
  - Barcode scanners
  - POS terminal
  - Consumer electronics
  - Toys

5. Introduction

The enCoRe II LV family brings the features and benefits of the enCoRe II to non-USB applications. The enCoRe II family has an integrated oscillator that eliminates the external crystal or resonator, reducing overall cost. Other external components, such as wakeup circuitry, are also integrated into this chip.

The enCoRe II LV is a low-voltage, low-cost 8-bit flash-programmable microcontroller.

The enCoRe II LV features up to 36 GPIO pins. The I/O pins are grouped into five ports (Port 0 to 4). The pins on ports 0 and 1 are configured individually, when the pins on ports 2, 3, and 4 are only configured as a group. Each GPIO port supports high-impedance inputs, configurable pull-up, open-drain output, CMOS, and TTL inputs, and CMOS output with up to five pins that support programmable drive strength of up to 50-mA sink current. Additionally, each I/O pin is used to generate a GPIO interrupt to the microcontroller. Each GPIO port has its own GPIO interrupt vector with the exception of GPIO port 0. GPIO port 0 has, in addition to the port interrupt vector, three dedicated pins that have independent interrupt vectors (P0.2–P0.4).

The enCoRe II LV features an internal oscillator. Optionally, an external 1-MHz to 24-MHz crystal is used to provide a higher precision reference. The enCoRe II LV also supports external clock.

The enCoRe II LV has 8 KB of flash for user code and 256 bytes of RAM for stack space and user variables.

In addition, enCoRe II LV includes a WDT, a vectored interrupt controller, a 16-bit free-running timer with capture registers, and a 12-bit programmable interval timer. The power on reset (POR) circuit detects when power is applied to the device, resets the logic to a known state, and executes instructions at flash address 0x0000. When power falls below a programmable trip voltage, it generates a reset or is configured to generate an interrupt. There is a LVD circuit that detects when V_{CC} drops below a programmable trip voltage. This is configurable to generate a LVD interrupt to inform the processor about the low-voltage event. POR and LVD share the same interrupt; there is no separate interrupt for each. The WDT ensures the firmware never gets stalled in an infinite loop.

The microcontroller supports 17 maskable interrupts in the vectored interrupt controller. All interrupts can be masked. Interrupt sources include LVR or POR, a programmable interval timer, a nominal 1.024 ms programmable output from the free-running timer, two capture timers, five GPIO ports, three GPIO pins, two SPI, a 16-bit free-running timer wrap, and an internal wakeup timer interrupt. The wakeup timer causes periodic interrupts when enabled. The capture timers interrupt whenever a new timer value is saved due to a selected GPIO edge event. A total of eight GPIO interrupts support both TTL or CMOS thresholds. For additional flexibility, on the edge-sensitive GPIO pins, the interrupt polarity is programmable to be either rising or falling.

The free-running timer generates an interrupt at 1024-μs rate. It also generates an interrupt when the free-running counter overflow occurs – every 16.384 ms. The duration of an event under firmware control is measured by reading the timer at the start and end of an event, then calculating the difference between the two values. The two 8-bit capture timer registers save a programmable 8-bit range of the free-running timer when a GPIO edge occurs on the two capture pins (P0.5 and P0.6). The two 8-bit capture registers are ganged into a single 16-bit capture register.

The enCoRe II LV supports in-system programming by using the P1.0 and P1.1 pins as the serial programming mode interface.

6. Conventions

In this document, bit positions in the registers are shaded to indicate which members of the enCoRe II LV family implement the bits.

☐	Available in all enCoRe II LV family members
■	CY7C601xx only

7. Pinouts

Figure 7-1. Package Configurations

Top View

CY7C60223
24-Pin SOIC

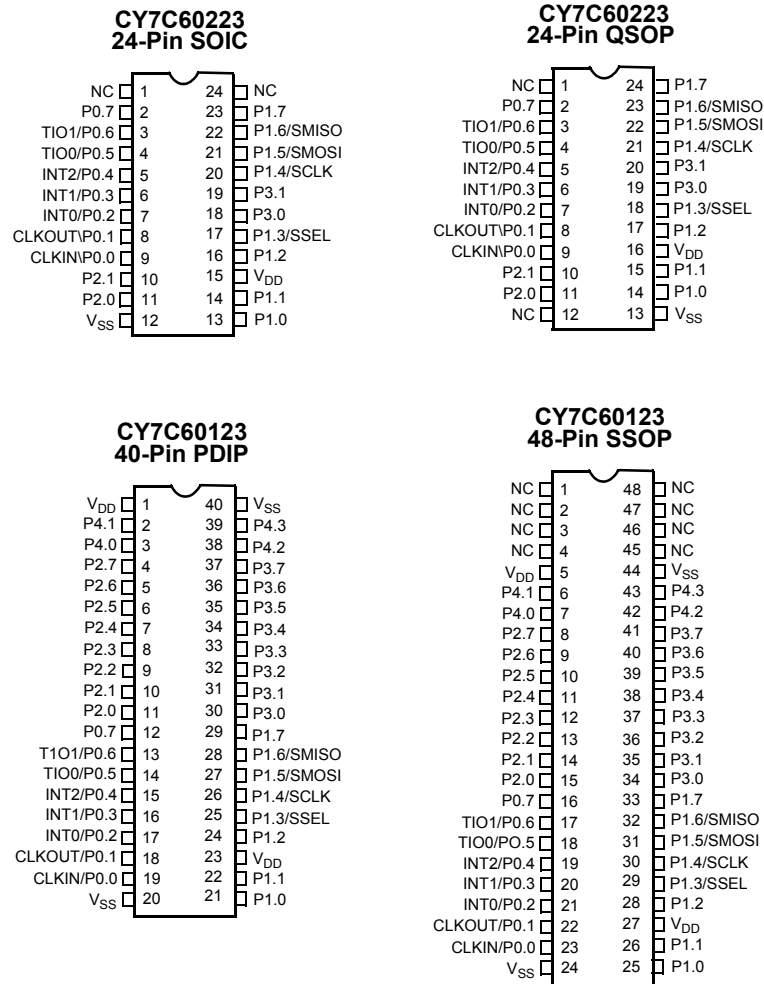|				
NC	1		24	NC
P0.7	2		23	P1.7
TIO1/P0.6	3		22	P1.6/SMISO
TIO0/P0.5	4		21	P1.5/SMOSI
INT2/P0.4	5		20	P1.4/SCLK
INT1/P0.3	6		19	P3.1
INT0/P0.2	7		18	P3.0
CLKOUT\P0.1	8		17	P1.3/SSEL
CLKIN\P0.0	9		16	P1.2
P2.1	10		15	V$_{DD}$
P2.0	11		14	P1.1
V$_{SS}$	12		13	P1.0

CY7C60223
24-Pin QSOP

NC	1		24	P1.7
P0.7	2		23	P1.6/SMISO
TIO1/P0.6	3		22	P1.5/SMOSI
TIO0/P0.5	4		21	P1.4/SCLK
INT2/P0.4	5		20	P3.1
INT1/P0.3	6		19	P3.0
INT0/P0.2	7		18	P1.3/SSEL
CLKOUT\P0.1	8		17	P1.2
CLKIN\P0.0	9		16	V$_{DD}$
P2.1	10		15	P1.1
P2.0	11		14	P1.0
NC	12		13	V$_{SS}$

CY7C60123
40-Pin PDIP

V$_{DD}$	1		40	V$_{SS}$
P4.1	2		39	P4.3
P4.0	3		38	P4.2
P2.7	4		37	P3.7
P2.6	5		36	P3.6
P2.5	6		35	P3.5
P2.4	7		34	P3.4
P2.3	8		33	P3.3
P2.2	9		32	P3.2
P2.1	10		31	P3.1
P2.0	11		30	P3.0
P0.7	12		29	P1.7
T1O1/P0.6	13		28	P1.6/SMISO
TIO0/P0.5	14		27	P1.5/SMOSI
INT2/P0.4	15		26	P1.4/SCLK
INT1/P0.3	16		25	P1.3/SSEL
INT0/P0.2	17		24	P1.2
CLKOUT/P0.1	18		23	V$_{DD}$
CLKIN/P0.0	19		22	P1.1
V$_{SS}$	20		21	P1.0

CY7C60123
48-Pin SSOP

NC	1		48	NC
NC	2		47	NC
NC	3		46	NC
NC	4		45	NC
V$_{DD}$	5		44	V$_{SS}$
P4.1	6		43	P4.3
P4.0	7		42	P4.2
P2.7	8		41	P3.7
P2.6	9		40	P3.6
P2.5	10		39	P3.5
P2.4	11		38	P3.4
P2.3	12		37	P3.3
P2.2	13		36	P3.2
P2.1	14		35	P3.1
P2.0	15		34	P3.0
P0.7	16		33	P1.7
TIO1/P0.6	17		32	P1.6/SMISO
TIO0/PO.5	18		31	P1.5/SMOSI
INT2/P0.4	19		30	P1.4/SCLK
INT1/P0.3	20		29	P1.3/SSEL
INT0/P0.2	21		28	P1.2
CLKOUT/P0.1	22		27	V$_{DD}$
CLKIN/P0.0	23		26	P1.1
V$_{SS}$	24		25	P1.0

7.1 Pin Assignments

Table 7-1. Pin Assignments

48 SSOP	40 PDIP	24 QSOP	24 SOIC	Name	Description
7	3	–	–	P4.0	GPIO port 4—configured as a group (nibble)
6	2	–	–	P4.1	
42	38	–	–	P4.2	
43	39	–	–	P4.3	
34	30	19	18	P3.0	GPIO port 3—configured as a group (byte)
35	31	20	19	P3.1	
36	32	–	–	P3.2	
37	33	–	–	P3.3	
38	34	–	–	P3.4	
39	35	–	–	P3.5	
40	36	–	–	P3.6	
41	37	–	–	P3.7	
15	11	11	11	P2.0	GPIO port 2—configured as a group (byte)
14	10	10	10	P2.1	
13	9	–	–	P2.2	
12	8	–	–	P2.3	
11	7	–	–	P2.4	
10	6	–	–	P2.5	
9	5	–	–	P2.6	
8	4	–	–	P2.7	
25	21	14	13	P1.0	GPIO port 1 bit 0/ISSP-SCLK If this pin is used as a general-purpose output it draws current. It is, therefore, configured as an input to reduce current draw.
26	22	15	14	P1.1	GPIO port 1 bit 1/ISSP-SDATA If this pin is used as a general-purpose output it draws current. It is, therefore, configured as an input to reduce current draw.
28	24	17	16	P1.2	GPIO port 1 bit 2
29	25	18	17	P1.3/SSEL	GPIO port 1 bit 3—Configured individually Alternate function is SSEL signal of the SPI bus.
30	26	21	20	P1.4/SCLK	GPIO port 1 bit 4—Configured individually Alternate function is SCLK signal of the SPI bus.
31	27	22	21	P1.5/SMOSI	GPIO port 1 bit 5—Configured individually Alternate function is SMOSI signal of the SPI bus.
32	28	23	22	P1.6/SMISO	GPIO port 1 bit 6—Configured individually Alternate function is SMISO signal of the SPI bus.
33	29	24	23	P1.7	GPIO port 1 bit 7—Configured individually TTL voltage threshold.
23	19	9	9	P0.0/CLKIN	GPIO port 0 bit 0—Configured individually On CY7C601xx, optional Clock In when external oscillator is disabled or external oscillator input when external oscillator is enabled. On CY7C602xx, oscillator input when configured as Clock In.

Table 7-1. Pin Assignments (continued)

48 SSOP	40 PDIP	24 QSOP	24 SOIC	Name	Description
22	18	8	8	P0.1/CLKOUT	GPIO port 0 bit 1—Configured individually On CY7C601xx, optional Clock Out when external oscillator is disabled or external oscillator output drive when external oscillator is enabled. On CY7C602xx, oscillator output when configured as Clock Out.
21	17	7	7	P0.2/INT0	GPIO port 0 bit 2—Configured individually Optional rising edge interrupt INT0.
20	16	6	6	P0.3/INT1	GPIO port 0 bit 3—Configured individually Optional rising edge interrupt INT1.
19	15	5	5	P0.4/INT2	GPIO port 0 bit 4—Configured individually Optional rising edge interrupt INT2.
18	14	4	4	P0.5/TIO0	GPIO port 0 bit 5—Configured individually Alternate function timer capture inputs or timer output TIO0.
17	13	3	3	P0.6/TIO1	GPIO port 0 bit 6—Configured individually Alternate function timer capture inputs or timer output TIO1.
16	12	2	2	P0.7	GPIO port 0 bit 7—Configured individually
1,2,3,4	–	1	1	NC	No connect
45,46,47,48	–	12	24	NC	No connect
5	1			V_{DD}	Power
27	23	16	15		
44	40	–	–	V_{SS}	Ground
24	20	13	12		

8. Register Summary

Table 8-1. enCoRe II LV Register Summary

The XIO bit in the CPU flags register must be set to access the extended register space for all registers above 0xFF.

Addr	Name	7	6	5	4	3	2	1	0	R/W	Default
00	P0DATA	P0.7	P0.6/TIO1	P0.5/TIO0	P0.4/INT2	P0.3/INT1	P0.2/INT0	P0.1/CLKOUT	P0.0/CLKIN	bbbbbbbb	00000000
01	P1DATA	P1.7	P1.6/SMISO	P1.5/SMOSI	P1.4/SCLK	P1.3/SSEL	P1.2	P1.1	P1.0	bbbbbbbb	00000000
02	P2DATA	P2.7–P2.2						P2.1–P2.0		bbbbbbbb	00000000
03	P3DATA	P3.7–P3.2						P3.1–P3.0		bbbbbbbb	00000000
04	P4DATA	Reserved				P4.3–P4.0				----bbbb	00000000
05	P00CR	Reserved	Int enable	Int act low	TTL thresh	High sink	Open drain	Pull-up enable	Output enable	-bbbbbbb	00000000
06	P01CR	CLK output	Int enable	Int act low	TTL thresh	High sink	Open drain	Pull-up enable	Output enable	bbbbbbbb	00000000
07–09	P02CR–P04CR	Reserved		Int act low	TTL thresh	Reserved	Open drain	Pull-up enable	Output enable	--bb-bbb	00000000
0A–0B	P05CR– P06CR	TIO output	Int enable	Int act low	TTL thresh	Reserved	Open drain	Pull-up enable	Output enable	bbbb-bbb	00000000
0C	P07CR	Reserved	Int enable	Int act low	TTL thresh	Reserved	Open drain	Pull-up enable	Output enable	-bbb-bbb	00000000
0D	P10CR	Reserved	Int enable	Int act low	Reserved				Output enable	-bb----b	00000000
0E	P11CR	Reserved	Int enable	Int act low	Reserved		Open drain	Reserved	Output enable	-bb--b-b	00000000
0F	P12CR	CLK output	Int enable	Int act low	TTL threshold	Reserved	Open drain	Pull-up enable	Output enable	bbbb-bbb	00000000
10	P13CR	Reserved	Int enable	Int act low	Reserved	High sink	Open drain	Pull-up enable	Output enable	-bb-bbbb	00000000
11–13	P14CR– P16CR	SPI use	Int enable	Int act low	Reserved	High sink	Open drain	Pull-up enable	Output enable	bbb-bbbb	00000000
14	P17CR	Reserved	Int enable	Int act low	Reserved	High sink	Open drain	Pull-up enable	Output enable	-bb-bbbb	00000000
15	P2CR	Reserved	Int enable	Int act low	TTL thresh	High sink	Open drain	Pull-up enable	Output enable	-bbbbbbb	00000000
16	P3CR	Reserved	Int enable	Int act low	TTL thresh	High sink	Open drain	Pull-up enable	Output enable	-bbbbbbb	00000000
17	P4CR	Reserved	Int enable	Int act low	TTL thresh	Reserved	Open drain	Pull-up enable	Output enable	-bbb-bbb	00000000
20	FRTMRL	Free-running timer [7:0]								bbbbbbbb	00000000
21	FRTMRH	Free-running timer [15:8]								bbbbbbbb	00000000
22	TCAP0R	Capture 0 rising [7:0]								rrrrrrrr	00000000
23	TCAP1R	Capture 1 rising [7:0]								rrrrrrrr	00000000
24	TCAP0F	Capture 0 falling [7:0]								rrrrrrrr	00000000
25	TCAP1F	Capture 1 falling [7:0]								rrrrrrrr	00000000
26	PITMRL	Prog interval timer [7:0]								rrrrrrrr	00000000
27	PITMRH	Reserved				Prog interval timer [11:8]				----rrrr	00000000
28	PIRL	Prog interval [7:0]								bbbbbbbb	00000000
29	PIRH	Reserved				Prog interval [11:8]				----bbbb	00000000
2A	TMRCR	First edge hold	8-bit capture prescale		Cap0 16-bit enable	Reserved				bbbbb---	00000000
2B	TCAPINTE	Reserved			Cap1 fall active	Cap1 rise active	Cap0 fall active	Cap0 rise active		----bbbb	00000000
2C	TCAPINTS	Reserved			Cap1 fall active	Cap1 rise active	Cap0 fall active	Cap0 rise active		----bbbb	00000000
30	CPUCLKCR	Reserved							CPU CLK select	-------b	00000000
31	TMRCLKCR	TCAPCLK divider		TCAPCLK select		ITMRCLK divider		ITMRCLK select		bbbbbbbb	10001111
32	CLKIOCR	Reserved			XOSC Select	XOSC Enable	EFTB Disabled	CLKOUT select		---bbbbb	00000000

Table 8-1. enCoRe II LV Register Summary (continued)

The XIO bit in the CPU flags register must be set to access the extended register space for all registers above 0xFF.

Addr	Name	7	6	5	4	3	2	1	0	R/W	Default
34	IOSCTR	foffset[2:0]			Gain[4:0]					bbbbbbbb	000ddddd
35	XOSCTR	Reserved			XOSC XGM [2:0]			Reserved	Mode	---bbb-b	000ddddd
36	LPOSCTR	32 kHz low power	Reserved	32 kHz bias trim [1:0]		32 kHz freq trim [3:0]				b-bbbbbb	d-ddddd
3C	SPIDATA	SPIData[7:0]								bbbbbbbb	00000000
3D	SPICR	Swap	LSB first	Comm mode		CPOL	CPHA	SCLK select		bbbbbbbb	00000000
DA	INT_CLR0	GPIO port 1	Sleep timer	INT1	GPIO Port 0	SPI Receive	SPI transmit	INT0	POR/LVD	bbbbbbbb	00000000
DB	INT_CLR1	TCAP0	Prog interval timer	1 ms timer	Reserved					bbb-----	00000000
DC	INT_CLR2	Reserved	GPIO port 4	GPIO port 3	GPIO port 2	Reserved	INT2	16-bit counter wrap	TCAP1	-bbb-bbb	00000000
DE	INT_MSK3	ENSWINT	Reserved							r-------	00000000
DF	INT_MSK2	Reserved	GPIO port 4 int enable	GPIO port 3 int enable	GPIO port 2 int enable	Reserved	INT2 Int enable	16-bit counter wrap int enable	TCAP1 Int enable	-bbb-bbb	00000000
E1	INT_MSK1	TCAP0 int enable	Prog interval timer int enable	1 ms timer int enable	Reserved					bbb-----	00000000
E0	INT_MSK0	GPIO Port 1 int enable	Sleep timer int enable	INT1 int enable	GPIO port 0 int enable	SPI receive int enable	SPI transmit int enable	INT0 int enable	POR/LVD int enable	bbbbbbbb	00000000
E2	INT_VC	Pending interrupt [7:0]								bbbbbbbb	00000000
E3	RESWDT	Reset watchdog timer [7:0]								wwwwwwww	00000000
--	CPU_A	Temporary register T1 [7:0]								--------	00000000
--	CPU_X	X[7:0]								--------	00000000
--	CPU_PCL	Program counter [7:0]								--------	00000000
--	CPU_PCH	Program counter [15:8]								--------	00000000
--	CPU_SP	Stack pointer [7:0]								--------	00000000
F7	CPU_F	Reserved			XIO	Super	Carry	Zero	Global IE	---brbbb	00000010
FF	CPU_SCR	GIES	Reserved	WDRS	PORS	Sleep	Reserved	Reserved	Stop	r-ccb--b	00010100
1E0	OSC_CR0	Reserved		No buzz	Sleep timer [1:0]		CPU speed [2:0]			--bbbbbb	00001000
1E3	LVDCR	Reserved		PORLEV[1:0]		Reserved	VM[2:0]			--bb-bbb	00000000
1EB	ECO_TR	Sleep duty cycle [1:0]		Reserved						bb------	00000000
1E4	VLTCMP	Reserved						LVD	PPOR	------rr	00000000

Note In the R/W column:

  b = Both read and write

  r = Read only

  w = Write only

  c = Read or clear

  d = Calibration value. Must not change during normal use

9. CPU Architecture

This family of microcontrollers is based on a high-performance, 8-bit, Harvard-architecture microprocessor. Five registers control the primary operation of the CPU core. These registers are affected by various instructions, but are not directly accessible through the register space by the user.

Table 9-1. CPU Registers and Register Name

Register	Register Name
Flags	CPU_F
Program counter	CPU_PC
Accumulator	CPU_A
Stack pointer	CPU_SP
Index	CPU_X

The 16-bit program counter register (CPU_PC) directly addresses the full 8 KB of program memory space.

10. CPU Registers

10.1 Flags Register

The flags register is only set or reset with logical instruction.

The accumulator register (CPU_A) is the general-purpose register that holds results of instructions that specify any of the source addressing modes.

The index register (CPU_X) holds an offset value used in the indexed addressing modes. Typically, this is used to address a block of data within the data memory space.

The stack pointer register (CPU_SP) holds the address of the current top-of-stack in the data memory space. It is affected by the PUSH, POP, LCALL, CALL, RETI, and RET instructions, which manage the software stack. It is also affected by the SWAP and ADD instructions.

The flag register (CPU_F) has three status bits: Zero Flag bit [1]; Carry Flag bit [2]; Supervisory State bit [3]. The global interrupt enable bit [0] is used to globally enable or disable interrupts. The user cannot manipulate the supervisory state status bit [3]. The flags are affected by arithmetic, logic, and shift operations. The manner in which each flag is changed is dependent upon the instruction being executed (AND, OR, XOR). See Table 11-1 on page 13.

Table 10-1. CPU Flags Register (CPU_F) [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved			XIO	Super	Carry	Zero	Global IE
Read/Write	–	–	–	R/W	R	R/W	R/W	R/W
Default	0	0	0	0	0	0	1	0

Bit [7:5]: Reserved
Bit 4: XIO
Set by the user to select between the register banks.
0 = Bank 0
1 = Bank 1
Bit 3: Super
Indicates whether the CPU is executing user code or supervisor code. (This code cannot be accessed directly by the user.)
0 = User code
1 = Supervisor code
Bit 2: Carry
Set by CPU to indicate whether there is a carry in the previous logical or arithmetic operation.
0 = No carry
1 = Carry
Bit 1: Zero
Set by CPU to indicate whether there is a zero result in the previous logical or arithmetic operation.
0 = Not equal to zero
1 = Equal to zero
Bit 0: Global IE
Determines whether all interrupts are enabled or disabled.
0 = Disabled
1 = Enabled
Note This register is readable with explicit address 0xF7. The *OR F, expr* and *AND F, expr* are used to set and clear the CPU_F bits.

10.1.1 Accumulator Register

Table 10-2. CPU Accumulator Register (CPU_A)

Bit #	7	6	5	4	3	2	1	0
Field	CPU accumulator [7:0]							
Read/Write	–	–	–	–	–	–	–	–
Default	0	0	0	0	0	0	0	0

Bit [7:0]: CPU accumulator [7:0]
8-bit data value holds the result of any logical or arithmetic instruction that uses a source addressing mode.

10.1.2 Index Register

Table 10-3. CPU X Register (CPU_X)

Bit #	7	6	5	4	3	2	1	0
Field	X [7:0]							
Read/Write	–	–	–	–	–	–	–	–
Default	0	0	0	0	0	0	0	0

Bit [7:0]: X [7:0]
8-bit data value holds an index for any instruction that uses an indexed addressing mode.

10.1.3 Stack Pointer Register

Table 10-4. CPU Stack Pointer Register (CPU_SP)

Bit #	7	6	5	4	3	2	1	0
Field	Stack pointer [7:0]							
Read/Write	–	–	–	–	–	–	–	–
Default	0	0	0	0	0	0	0	0

Bit [7:0]: Stack pointer [7:0]
8-bit data value holds a pointer to the current top-of-stack.

10.1.4 CPU Program Counter High Register

Table 10-5. CPU Program Counter High Register (CPU_PCH)

Bit #	7	6	5	4	3	2	1	0
Field	Program counter [15:8]							
Read/Write	–	–	–	–	–	–	–	–
Default	0	0	0	0	0	0	0	0

Bit [7:0]: Program counter [15:8]
8-bit data value holds the higher byte of the program counter.

10.1.5 CPU Program Counter Low Register

Table 10-6. CPU Program Counter Low Register (CPU_PCL)

Bit #	7	6	5	4	3	2	1	0
Field	Program counter [7:0]							
Read/Write	–	–	–	–	–	–	–	–
Default	0	0	0	0	0	0	0	0

Bit [7:0]: Program counter [7:0]
8-bit data value holds the lower byte of the program counter.

10.2 Addressing Modes

10.2.1 Source Immediate

The result of an instruction using this addressing mode is placed in the A register, the F register, the SP register, or the X register, which is specified as part of the instruction opcode. Operand 1 is an immediate value that serves as a source for the instruction. Arithmetic instructions require two sources; the second source is the A, X, SP, or F register specified in the opcode. Instructions using this addressing mode are two bytes in length.

Table 10-7. Source Immediate

Opcode	Operand 1
Instruction	Immediate value

Examples

ADD   A,   7   ;In this case, the immediate value of 7 is added with the accumulator and the result is placed in the accumulator.

MOV   X,   8   ;In this case, the immediate value of 8 is moved to the X register.

AND   F,   9   ;In this case, the immediate value of 9 is logically ANDed with the F register and the result is placed in the F register.

10.2.2 Source Direct

The result of an instruction using this addressing mode is placed in either the A register or the X register, which is specified as part of the instruction opcode. Operand 1 is an address that points to a location in either the RAM memory space or the register space that is the source for the instruction. Arithmetic instructions require two sources; the second source is the A register or X register specified in the opcode. Instructions using this addressing mode are two bytes in length.

Table 10-8. Source Direct

Opcode	Operand 1
Instruction	Source address

Examples

ADD   A,   [7]   ;In this case, the value in the RAM memory location at address 7 is added with the accumulator, and the result is placed in the accumulator.

MOV   X,   REG[8]   ;In this case, the value in the register space at address 8 is moved to the X register.

10.2.3 Source Indexed

The result of an instruction using this addressing mode is placed in either the A register or the X register, which is specified as part of the instruction opcode. Operand 1 is added to the X register forming an address that points to a location in either the RAM memory space or the register space that is the source for the instruction. Arithmetic instructions require two sources; the second source is the A register or X register specified in the opcode. Instructions using this addressing mode are two bytes in length.

Table 10-9. Source Indexed

Opcode	Operand 1
Instruction	Source index

Examples

ADD   A,   [X+7]   ;In this case, the value in the memory location at address X + 7 is added with the accumulator, and the result is placed in the accumulator.

MOV   X,   REG[X+8]   ;In this case, the value in the register space at address X + 8 is moved to the X register.

10.2.4 Destination Direct

The result of an instruction using this addressing mode is placed within either the RAM memory space or the register space. Operand 1 is an address that points to the location of the result. The source for the instruction is either the A register or the X register, which is specified as part of the instruction opcode. Arithmetic instructions require two sources; the second source is the location specified by Operand 1. Instructions using this addressing mode are two bytes in length.

Table 10-10. Destination Direct

Opcode	Operand 1
Instruction	Destination address

Examples

ADD   [7],   A   ;In this case, the value in the memory location at address 7 is added with the accumulator, and the result is placed in the memory location at address 7. The accumulator is unchanged.

MOV   REG[8],   A   ;In this case, the accumulator is moved to the register space location at address 8. The accumulator is unchanged.

10.2.5 Destination Indexed

The result of an instruction using this addressing mode is placed within either the RAM memory space or the register space. Operand 1 is added to the X register forming the address that points to the location of the result. The source for the instruction is the A register. Arithmetic instructions require two sources; the second source is the location specified by Operand 1 added with the X register. Instructions using this addressing mode are two bytes in length.

Table 10-11. Destination Indexed

Opcode	Operand 1
Instruction	Destination index

Example

```
ADD   [X+7],      A     ;In this case, the value in the memory
                          location at address X+7 is added with the
                          accumulator and the result is placed in the
                          memory location at address X+7. The
                          accumulator is unchanged.
```

10.2.6 Destination Direct Source Immediate

The result of an instruction using this addressing mode is placed within either the RAM memory space or the register space. Operand 1 is the address of the result. The source for the instruction is Operand 2, which is an immediate value. Arithmetic instructions require two sources; the second source is the location specified by Operand 1. Instructions using this addressing mode are three bytes in length.

Table 10-12. Destination Direct Source Immediate

Opcode	Operand 1	Operand 2
Instruction	Destination address	Immediate value

Examples

```
ADD   [7],      5     ;In this case, value in the memory location
                        at address 7 is added to the immediate
                        value of 5, and the result is placed in the
                        memory location at address 7.

MOV   REG[8],   6     ;In this case, the immediate value of 6 is
                        moved into the register space location at
                        address 8.
```

10.2.7 Destination Indexed Source Immediate

The result of an instruction using this addressing mode is placed within either the RAM memory space or the register space. Operand 1 is added to the X register to form the address of the result. The source for the instruction is Operand 2, which is an immediate value. Arithmetic instructions require two sources; the second source is the location specified by Operand 1 added with the X register. Instructions using this addressing mode are three bytes in length.

Table 10-13. Destination Indexed Source Immediate

Opcode	Operand 1	Operand 2
Instruction	Destination index	Immediate value

Examples

```
ADD   [X+7],      5     ;In this case, the value in the memory
                          location at address X+7 is added
                          with the immediate value of 5, and
                          the result is placed in the memory
                          location at address X+7.

MOV   REG[X+8],   6     ;In this case, the immediate value of
                          6 is moved into the location in the
                          register space at address X+8.
```

10.2.8 Destination Direct Source Direct

The result of an instruction using this addressing mode is placed within the RAM memory. Operand 1 is the address of the result. Operand 2 is an address that points to a location in the RAM memory that is the source for the instruction. This addressing mode is only valid on the MOV instruction. The instruction using this addressing mode is three bytes in length.

Table 10-14. Destination Direct Source Direct

Opcode	Operand 1	Operand 2
Instruction	Destination address	Source address

Example

```
MOV   [7],      [8]     ;In this case, the value in the memory location
                          at address 8 is moved to the memory location
                          at address 7.
```

10.2.9 Source Indirect Post Increment

The result of an instruction using this addressing mode is placed in the accumulator. Operand 1 is an address pointing to a location within the memory space, which contains an address (the indirect address) for the source of the instruction. The indirect address is incremented as part of the instruction execution. This addressing mode is only valid on the MVI instruction. The instruction using this addressing mode is two bytes in length. Refer to the *PSoC Designer: Assembly Language User Guide* for further details on MVI instruction.

Table 10-15. Source Indirect Post Increment

Opcode	Operand 1
Instruction	Source address

Example

MVI	A,	[8]	;In this case, the value in the memory location at address 8 is an indirect address. The memory location pointed to by the Indirect address is moved into the accumulator. The indirect address is then incremented.

10.2.10 Destination Indirect Post Increment

The result of an instruction using this addressing mode is placed within the memory space. Operand 1 is an address pointing to a location within the memory space, which contains an address (the indirect address) for the destination of the instruction. The indirect address is incremented as part of the instruction execution. The source for the instruction is the accumulator. This addressing mode is only valid on the MVI instruction. The instruction using this addressing mode is two bytes in length.

Table 10-16. Destination Indirect Post Increment

Opcode	Operand 1
Instruction	Destination address

Example

MVI	[8],	A	;In this case, the value in the memory location at address 8 is an indirect address. The accumulator is moved into the memory location pointed to by the indirect address. The indirect address is then incremented.

11. Instruction Set Summary

The instruction set is summarized in Table 11-1 numerically and serves as a quick reference. The instruction set summary tables are described in detail in the *PSoC Designer: Assembly Language User Guide*.

Table 11-1. Instruction Set Summary Sorted Numerically by Opcode Order

Opcode Hex	Cycles	Bytes	Instruction Format[1, 2]	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

Notes
1. Interrupt routines take 13 cycles before execution resumes at interrupt vector table.
2. The number of cycles required by an instruction is increased by one for instructions that span 256 byte boundaries in the flash memory space.
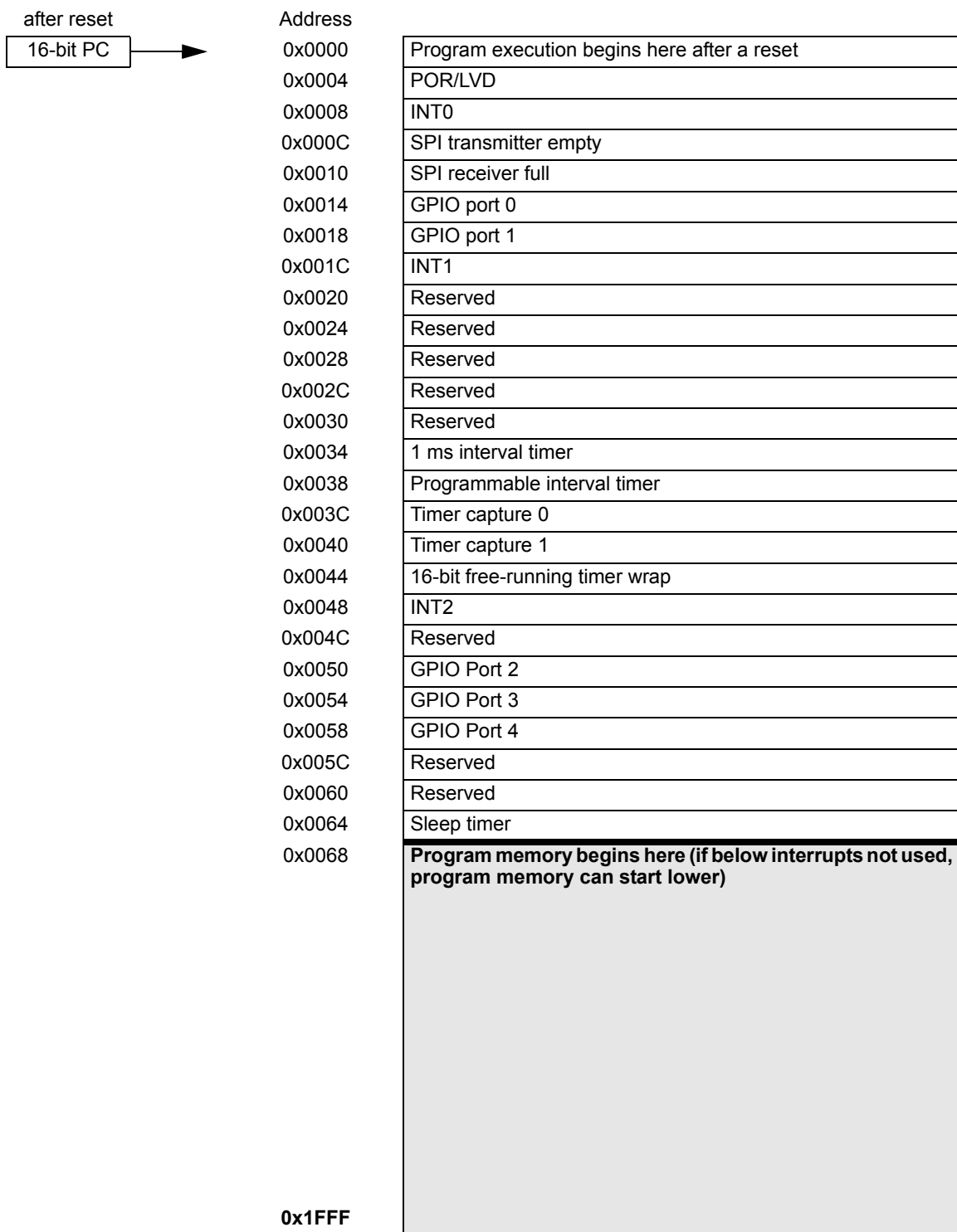
Table 11-1. Instruction Set Summary Sorted Numerically by Opcode Order (continued)

Opcode Hex	Cycles	Bytes	Instruction Format[1, 2]	Flags	Opcode Hex	Cycles	Bytes	Instruction Format	Flags	Opcode Hex	Cycles	Bytes	Instruction Format	Flags
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

12. Memory Organization

12.1 Flash Program Memory Organization

Figure 12-1. Program Memory Space with Interrupt Vector Table

after reset	Address	
16-bit PC →	0x0000	Program execution begins here after a reset
	0x0004	POR/LVD
	0x0008	INT0
	0x000C	SPI transmitter empty
	0x0010	SPI receiver full
	0x0014	GPIO port 0
	0x0018	GPIO port 1
	0x001C	INT1
	0x0020	Reserved
	0x0024	Reserved
	0x0028	Reserved
	0x002C	Reserved
	0x0030	Reserved
	0x0034	1 ms interval timer
	0x0038	Programmable interval timer
	0x003C	Timer capture 0
	0x0040	Timer capture 1
	0x0044	16-bit free-running timer wrap
	0x0048	INT2
	0x004C	Reserved
	0x0050	GPIO Port 2
	0x0054	GPIO Port 3
	0x0058	GPIO Port 4
	0x005C	Reserved
	0x0060	Reserved
	0x0064	Sleep timer
	0x0068	**Program memory begins here (if below interrupts not used, program memory can start lower)**
	0x1FFF	

12.2 Data Memory Organization

The CY7C601xx and CY7C602xx microcontrollers provide up to 256 bytes of data RAM

Figure 12-2. Data Memory Organization

After Reset	Address	
8-bit PSP →	0x00	**Stack begins here and grows upward**
Top of RAM Memory	0xFF	
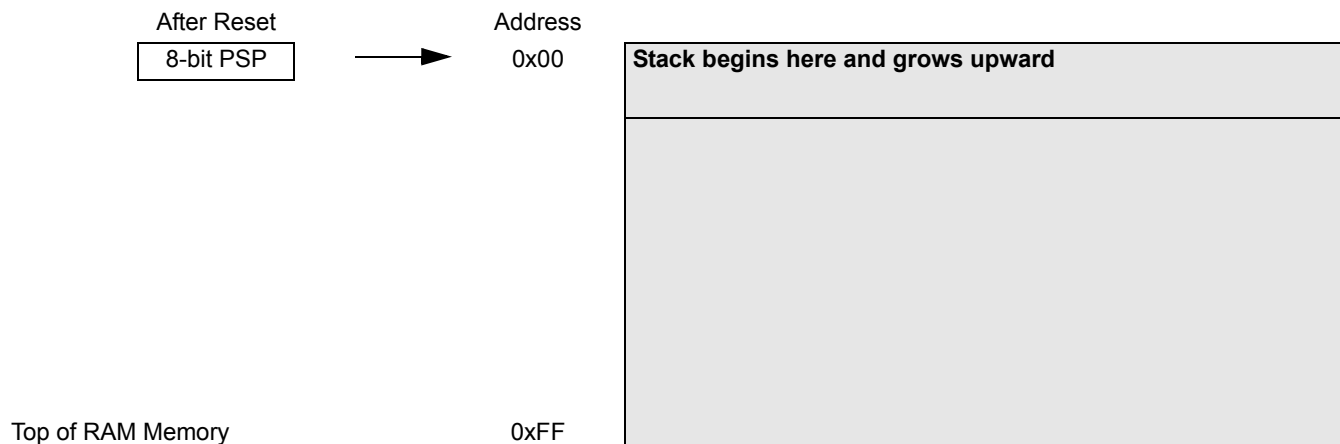
12.3 Flash

This section describes the flash block of enCoRe II LV. Much of the visible flash functionality, including programming and security, are implemented in the M8C supervisory read only memory (SROM). enCoRe II LV flash has an endurance of 1000 erase and write cycles and a ten year data retention capability.

12.3.1 Flash Programming and Security

All flash programming is performed by code in the SROM. The registers that control flash programming are only visible to the M8C CPU when it is executing out of SROM. This makes it impossible to read, write, or erase the flash by avoiding the security mechanisms implemented in the SROM.

Customer firmware only programs flash through SROM calls. The data or code images are sourced through any interface with the appropriate support firmware. This type of programming requires a 'bootloader'—a piece of firmware resident on the flash. For safety reasons, this bootloader is not overwritten during firmware rewrites.

The flash provides four extra auxiliary rows to hold flash block protection flags, boot time calibration values, configuration tables, and any device values. The routines to access these auxiliary rows are documented in the SROM section. The auxiliary rows are not affected by the device erase function.

12.3.2 In-System Programming

enCoRe II LV devices enable in-system programming by using the P1.0 and P1.1 pins as the serial programming mode interface. This allows an external controller to make the enCoRe II LV part enter serial programming mode and then use the test queue to issue flash access functions in the SROM.

12.4 SROM

The SROM holds the code to boot the part, calibrate circuitry, and perform Flash operations (Table 12-1 lists the SROM functions). The functions of the SROM are accessed in normal user code or operating from flash. The SROM exists in a separate memory space from user code. To access SROM functions, the supervisory system call (SSC) instruction is executed, which has an opcode of 00h. Before executing SSC, the M8C's accumulator is loaded with the desired SROM function code from Table 12-1. Undefined functions causes a HALT if called from user code. 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 must be configured before executing the SSC. Table 12-2 on page 17 lists all possible parameter block variables. The meaning of each parameter, with regard to a specific SROM function, is described later in this section.

Table 12-1. SROM Function Codes

Function Code	Function Name	Stack Space
00h	SWBootReset	0
01h	ReadBlock	7
02h	WriteBlock	10
03h	EraseBlock	9
05h	EraseAll	11
06h	TableRead	3
07h	CheckSum	3

Two important variables used for all functions are KEY1 and KEY2. These variables help discriminate between valid and inadvertent SSCs. KEY1 always has a value of 3Ah, while KEY2 has the same value as the stack pointer when the SROM function begins execution. This is the stack pointer value when the SSC opcode is executed, plus three. If either of the keys do not match the expected values, the M8C halts (with the exception of the SWBootReset function). The following code puts the correct value in KEY1 and KEY2. The code starts with a halt, to force the program to jump directly into the setup code and not run into it.

```
halt
SSCOP: mov [KEY1], 3ah
mov X, SP
mov A, X
add A, 3
mov [KEY2], A
```

Table 12-2. SROM Function Parameters

Variable Name	SRAM Address
Key1/Counter/Return Code	0,F8h
Key2/TMP	0,F9h
BlockID	0,FAh
Pointer	0,FBh
Clock	0,FCh
Mode	0,FDh
Delay	0,FEh
PCL	0,FFh

12.4.1 Return Codes

The SROM also features return codes and lockouts.

Return codes determine the 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 12-3. SROM Return Codes

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

Read, write, and erase operations may fail if the target block is read- or write-protected. Block protection levels are set during device programming.

The EraseAll function overwrites data in addition to leaving the entire user flash in the erase state. The EraseAll function loops through the number of flash macros in the product, executing the following sequence: erase, bulk program all zeros, erase. After the user space in all flash macros are erased, a second loop erases and then programs each protection block with zeros.

12.5 SROM Function Descriptions

12.5.1 SWBootReset Function

The SROM function, SWBootReset, is responsible for transitioning the device from a reset state to running user code. 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. The SWBootReset function does not execute when the SSC instruction is executed with a bad key value and a non zero function code. An enCoRe II LV device executes the HALT instruction if a bad value is given for either KEY1 or KEY2.

The SWBootReset function verifies the integrity of the calibration data by way of a 16-bit checksum, before releasing the M8C to run user code.

12.5.2 ReadBlock Function

The ReadBlock function is used to read 64 contiguous bytes from flash: a block.

The function first checks the protection bits and determines 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 first of the 64 bytes is stored in SRAM at the address indicated by the value of the POINTER parameter. When the ReadBlock completes successfully the accumulator, KEY1, and KEY2 all have a value of 00h.

Table 12-4. ReadBlock Parameters

Name	Address	Description
KEY1	0,F8h	3Ah
KEY2	0,F9h	Stack pointer value, when SSC is executed
BLOCKID	0,FAh	Flash block number
POINTER	0,FBh	First of 64 addresses in SRAM where returned data is stored

12.5.3 WriteBlock Function

The WriteBlock function is used to store data in flash. Data is moved 64 bytes at a time from SRAM to flash using this function. The WriteBlock function first checks the protection bits and determines if the desired BLOCKID is writable. If write protection is turned on, the WriteBlock function exits setting the accumulator and KEY2 back to 00h. KEY1 has a value of 01h, indicating a write failure. The configuration of the WriteBlock function is straightforward. The BLOCKID of the flash block, where the data is stored, is determined and stored at SRAM address FAh.

The SRAM address of the first of the 64 bytes to be stored in flash is indicated using the POINTER variable in the parameter block (SRAM address FBh). Finally, the CLOCK and DELAY value are set correctly. The CLOCK value determines the length of the write pulse used to store the data in flash. The CLOCK and DELAY values are dependent on the CPU speed and must be set correctly. Refer to the Clocking section for additional information.

Table 12-5. WriteBlock Parameters

Name	Address	Description
KEY1	0,F8h	3Ah
KEY2	0,F9h	Stack pointer value, when SSC is executing
BLOCK ID	0,FAh	8 KB flash block number (00h–7Fh) 4 KB flash block number (00h–3Fh) 3 KB flash block number (00h–2Fh)
POINTER	0,FBh	First 64 addresses in SRAM where the data is stored in flash is located before calling WriteBlock
CLOCK	0,FCh	Clock divider used to set the write pulse width
DELAY	0,FEh	For a CPU speed of 12 MHz set to 56h

12.5.4 EraseBlock Function

The EraseBlock function is used to erase a block of 64 contiguous bytes in flash. The EraseBlock function first checks the protection bits and determines if the desired BLOCKID is writable. 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. The EraseBlock function is only useful as the first step in programming. Erasing a block does not make data in a block fully unreadable. If the objective is to obliterate data in a block, the best method is to perform an EraseBlock followed by a WriteBlock of all zeros.

To set up the parameter block for EraseBlock, correct key values must be stored in KEY1 and KEY2. The block number to be erased is stored in the BLOCKID variable and the CLOCK and DELAY values are set based on the current CPU speed.

Table 12-6. EraseBlock Parameters

Name	Address	Description
KEY1	0,F8h	3Ah
KEY2	0,F9h	Stack pointer value, when SSC is executed
BLOCKID	0,FAh	Flash block number (00h–7Fh)
CLOCK	0,FCh	Clock divider used to set the erase pulse width
DELAY	0,FEh	For a CPU speed of 12 MHz set to 56h

12.5.5 ProtectBlock Function

The enCoRe II LV devices offer flash protection on a block-by-block basis. Table 12-7 lists the protection modes available. In the table, ER and EW indicate the ability to perform external reads and writes; IW is used for internal writes. Internal reading is always permitted using the ROMX instruction. The ability to read using the SROM ReadBlock function is indicated by SR. The protection level is stored in two bits according to Table 12-7. These bits are bit packed into 64 bytes of the protection block. Therefore, each protection block byte stores the protection level for four flash blocks. The bits are packed into a byte, with the lowest numbered block's protection level stored in the lowest numbered bits in Table 12-7.

The first address of the protection block contains the protection level for blocks 0 through 3; the second address is for blocks 4 through 7. The 64th byte stores the protection level for blocks 252 through 255.

Table 12-7. Protection Modes

Mode	Settings	Description	Marketing
00b	SR ER EW IW	Unprotected	Unprotected
01b	S̶R̶ ER EW IW	Read protect	Factory upgrade
10b	S̶R̶ ER E̶W̶ IW	Disable external write	Field upgrade
11b	S̶R̶ E̶R̶ E̶W̶ I̶W̶	Disable internal write	Full protection

7	6	5	4	3	2	1	0
Block n+3		Block n+2		Block n+1		Block n	

Only an EraseAll decreases the protection level by placing zeros in all locations of the protection block. To set the level of protection, the ProtectBlock function is used. This function takes data from SRAM, starting at address 80h, and ORs it with the current values in the protection block. The result of the OR operation is then stored in the protection block. The EraseBlock function does not change the protection level for a block. Because the SRAM location for the protection data is fixed and there is only one protection block for every flash macro, the ProtectBlock function expects very few variables in the parameter block to be set before calling the function. The parameter block values that are, besides the keys, are the CLOCK and DELAY values.

Table 12-8. ProtectBlock Parameters

Name	Address	Description
KEY1	0,F8h	3Ah
KEY2	0,F9h	Stack pointer value when SSC is executed
CLOCK	0,FCh	Clock divider used to set the write pulse width
DELAY	0,FEh	For a CPU speed of 12 MHz set to 56h

12.5.6 EraseAll Function

The EraseAll function performs a series of steps that destroy the user data in the Flash macros and resets the protection block in each Flash macro to all zeros (the unprotected state). The EraseAll function does not affect the three hidden blocks above the protection block in each flash macro. The first of these four hidden blocks is used to store the protection table for its 8 KB of user data.

The EraseAll function begins by erasing the user space of the flash macro with the highest address range. A bulk program of all zeros is then performed on the same flash macro, to destroy all traces of previous contents. The bulk program is followed by a second erase that leaves the flash macro ready for writing. The erase, program, erase sequence is then performed on the next lowest flash macro in the address space if it exists. Following erase of the user space, the protection block for the flash macro with the highest address range is erased. Following erase of the protection block, zeros are written into every bit of the protection table. The next lowest flash macro in the address space then has its protection block erased and filled with zeros.

The result of the EraseAll function is that all user data in flash is destroyed and the flash is left in an unprogrammed state, ready to accept one of the various write commands. The protection bits for all user data are also reset to the zero state.

Besides the keys, the CLOCK and DELAY parameter block values are also set.

Table 12-9. EraseAll Parameters

Name	Address	Description
KEY1	0,F8h	3Ah
KEY2	0,F9h	Stack pointer value when SSC is executed
CLOCK	0,FCh	Clock divider used to set the write pulse width
DELAY	0,FEh	For a CPU speed of 12 MHz set to 56h

12.5.7 TableRead Function

The TableRead function gives the user access to part specific data stored in the flash during manufacturing. It also returns a Revision ID for the die (not to be confused with the Silicon ID).

Table 12-10. Table Read Parameters

Name	Address	Description
KEY1	0,F8h	3Ah
KEY2	0,F9h	Stack pointer value when SSC is executed.
BLOCKID	0,FAh	Table number to read.

The table space for the enCoRe II LV is simply a 64-byte row broken up into eight tables of eight bytes (see Figure 12-3 on page 21). The tables are numbered zero through seven. All user and hidden blocks in the CY7C601xx/CY7C602xx parts consist of 64 bytes.

An internal table (Table 0) holds the silicon ID and returns the revision ID. The silicon ID is returned in SRAM, while the revision and family IDs are returned in the CPU_A and CPU_X registers. The silicon ID is a value placed in the table by programming the flash and is controlled by Cypress Semiconductor Product Engineering. The revision ID is hard coded into the SROM and also redundantly placed in SROM Table 1. This is discussed in detail later in this section.

SROM Table 1 holds Family/Die ID and revision ID values for the device and returns a one-byte internal revision counter. The internal revision counter starts with a value of zero and is incremented when one of the other revision numbers is not incremented. It is reset to zero when one of the other revision numbers is incremented. The internal revision count is returned in the CPU_A register. The CPU_X register is always set to FFh when Table 1 is read. The CPU_A and CPU_X registers always return a value of FFh when Tables 2 to 7 are read. The BLOCKID value, in the parameter block, indicates which table must be returned to the user. Only the three least significant bits of the BLOCKID parameter are used by TableRead function for enCoRe II LV devices. The upper five bits are ignored. When the function is called, it transfers bytes from the table to SRAM addresses F8h–FFh.

The M8C's A and X registers are used by the TableRead function to return the die's revision ID. The revision ID is a 16-bit value hard coded into the SROM that uniquely identifies the die's design.

The return values for the corresponding table calls are tabulated as shown in Table 12-11.

Table 12-11. Return Values for Table Read

Table Number	Return Value	
	A	X
0	Revision ID	Family ID
1	Internal revision counter	0xFF
2-7	0xFF	0xFF

12.6 SROM Table Read Description

The silicon IDs for enCoRe II LV devices are stored in the SROM tables in the part, as shown in Figure 12-3 on page 21.

The silicon ID can be read out from the part using SROM table reads. This is demonstrated in the following pseudo code. As mentioned in the section SROM on page 16, the SROM variables occupy address F8h through FFh in the SRAM. Each of the variables and their definitions are given in the section SROM on page 16.

```
AREA SSCParmBlkA(RAM,ABS)

    org  F8h // Variables are defined starting at address F8h

SSC_KEY1:                 ; F8h  supervisory key
SSC_RETURNCODE:    blk 1 ; F8h  result code
SSC_KEY2 :         blk 1  ;F9h  supervisory stack ptr key
SSC_BLOCKID:       blk 1 ; FAh  block ID
SSC_POINTER:       blk 1 ; FBh  pointer to data buffer
SSC_CLOCK:         blk 1 ; FCh  Clock
SSC_MODE:          blk 1 ; FDh  ClockW ClockE multiplier
SSC_DELAY:         blk 1 ; FEh  flash macro sequence delay count
SSC_WRITE_ResultCode: blk 1 ; FFh  temporary result code

_main:
        mov   A, 2
        mov   [SSC_BLOCKID], A// To read from Table 2 - trim values for the IMO are stored in table 2
        mov   X, SP      ; copy SP into X
        mov   A, X       ; A temp stored in X
        add   A, 3                ; create 3 byte stack frame (2 + pushed A)
        mov   [SSC_KEY2], A       ; save stack frame for supervisory code

   ; load the supervisory code for flash operations
        mov   [SSC_KEY1], 3Ah  ;FLASH_OPER_KEY - 3Ah

        mov   A,6        ; load A with specific operation. 06h is the code for Table (read Table
12-1 on page 16)
        SSC              ; SSC call the supervisory ROM

// At the end of the SSC command the silicon ID is stored in F8 (MSB) and F9(LSB) of the SRAM

.terminate:
    jmp .terminate
```

Figure 12-3. SROM Table

	F8h	F9h	FAh	FBh	FCh	FDh	FEh	FFh
Table 0	Silicon ID [15-8]	Silicon ID [7-0]						
Table 1	Family / Die ID	Revision ID						
Table 2					24 MHz IOSCTR at 3.30V	24 MHz IOSCTR at 3.00V	24 MHz IOSCTR at 2.85V	24 MHz IOSCTR at 2.70V
Table 3	32 kHz LPOSCTR at 3.30V	32 kHz LPOSCTR at 3.00V	32 kHz LPOSCTR at 2.85V	32 kHz LPOSCTR at 2.70V				
Table 4								
Table 5								
Table 6								
Table 7								

12.6.1 Checksum Function

The Checksum function calculates a 16-bit checksum over a user-specifiable number of blocks, within a single flash macro (Bank) starting from block zero. The BLOCKID parameter is used to pass in the number of blocks to calculate the checksum over. A BLOCKID value of '1' calculates the checksum of only block 0, while a BLOCKID value of '0' calculates the checksum of all 256 user blocks. The 16-bit checksum is returned in KEY1 and KEY2. The parameter KEY1 holds the lower eight bits of the checksum and the parameter KEY2 holds the upper eight bits of the checksum.

The checksum algorithm executes the following sequence of three instructions over the number of blocks times 64 to be checksummed.

```
romx
add [KEY1], A
adc [KEY2], 0
```

Table 12-1. Checksum Parameters

Name	Address	Description
KEY1	0,F8h	3Ah
KEY2	0,F9h	Stack pointer value when SSC is executed
BLOCKID	0,FAh	Number of flash blocks to calculate checksum on

13. Clocking

The enCoRe II LV has two internal oscillators, the internal 24-MHz oscillator and the 32-kHz low-power oscillator.

The internal 24-MHz oscillator is designed such that it is trimmed to an output frequency of 24 MHz over temperature and voltage variation. The internal 24-MHz oscillator accuracy is 24 MHz –22% to +10% (between 0° and 70°C). No external components are required to achieve this level of accuracy.

Firmware is responsible for selecting the correct trim values from the user row to match the power supply voltage in the end application and writing the values to the trim registers IOSCTR and LPOSCTR.

The internal low-speed oscillator (ILO) of nominally 32 kHz provides a slow clock source for the enCoRe II LV in suspend mode. This is used to generate a periodic wakeup interrupt and provide a clock to sequential logic during power-up and power-down events when the main clock is stopped. In addition, this oscillator can be used as a clocking source for the interval timer clock (ITMRCLK) and capture timer clock (TCAPCLK). The 32-kHz low-power oscillator can operate in low-power mode or provide a more accurate clock in normal mode. The internal 32 kHz low-power oscillator accuracy ranges from –53.12% to +56.25%. The 32-kHz low-power oscillator can be calibrated against the internal 24-MHz oscillator or another timing source, if desired.

enCoRe II LV provides the ability to load new trim values for the 24-MHz oscillator based on voltage. This allows V_{DD} to be monitored and have firmware trim the oscillator based on the voltage present. The IOSCTR register is used to set trim values for the 24-MHz oscillator. enCoRe II LV is initialized with 3.30-V trim values at power-on, then firmware is responsible for transferring the correct set of trim values to the trim registers to match the application's actual V_{DD}. The 32-kHz oscillator generally does not require trim adjustments for voltage but trim values for 32 kHz are also stored in supervisory ROM.

To improve the accuracy of the IMO, new trim values are loaded based on supply voltage to the part. For this, firmware needs to make modifications to two registers:

1. The internal oscillator trim register at location 0x34.

2. The gain register at location 0x38.

13.1 Trim Values for the IOSCTR Register

The trim values are stored in SROM tables in the part as shown in Figure 12-3 on page 21.

The trim values are read out from the part based on voltage settings and written to the IOSCTR register at location 0x34. The following pseudo code shows how this is done.

```
_main:
    mov    A, 2
    mov    [SSC_BLOCKID], A
```

Call SROM operation to read the SROM table (refer to the section SROM Table Read Description on page 20).

```
//After this command is executed, the trim
//values for 3.3, 3.0, 2.85 and 2.7 are stored
//at locations FC through FF in the RAM. SROM
//calls are explained in the previous section of
//this data sheet
;    mov    A, [FCh] // trim values for 3.3 V
     mov    A, [FDh] // trim values for 3.0 V
;    mov    A, [FEh] // trim values for 2.85 V
;    mov    A, [FFh] // trim values for 2.70 V
     mov    reg[IOSCTR],A // Loading IOSCTR with
                          // trim values for
                          // 3.0 V
.terminate:
    jmp .terminate
```

Gain value for the register at location [0x38]:

3.3 V = 0x40

3.0 V = 0x40

2.85 V = 0xFF

2.70 V = 0xFF

Load register [0x38] with the gain values corresponding to the appropriate voltage.

Table 13-1. Oscillator Trim Values versus Voltage Settings

Supervisory ROM Table	Function
Table2 FCh	24 MHz IOSCTR at 3.30 V
Table2 FDh	24 MHz IOSCTR at 3.00 V
Table2 FEh	24 MHz IOSCTR at 2.85 V
Table2 FFh	24 MHz IOSCTR at 2.70 V
Table3 F8h	32 kHz LPOSCTR at 3.30 V
Table3 F9h	32 kHz LPOSCTR at 3.00 V
Table3 FAh	32 kHz LPOSCTR at 2.85 V
Table3 FBh	32 kHz LPOSCTR at 2.70 V

When using the 32-kHz oscillator, the PITMRL/H is read until two consecutive readings match before sending and receiving data. The following firmware example assumes the developer is interested in the lower byte of the PIT.

```
Read_PIT_counter:
mov A, reg[PITMRL]
mov [57h], A
mov A, reg[PITMRL]
mov [58h],A
mov [59h], A
mov A, reg[PITMRL]
mov [60h], A
;;;Start comparison
mov A,[60h]
mov X, [59h]
sub A, [59h]
jz done
mov A, [59h]
mov X, [58h]
sub A, [58h]
jz done
mov X, [57h]
;;;correct data is in memory location 57h
done:
mov [57h], X
ret
```

The CY7C601xx part is optionally sourced from an external crystal oscillator. The external clock driving on CLKIN range is from 187 kHz to 24 MHz.

13.2 Clock Architecture Description

The enCoRe II LV clock selection circuitry allows the selection of independent clocks for the CPU, interval timers, and capture timers.

On the CY7C601xx, the external oscillator is sourced by the crystal oscillator. When the crystal oscillator is disabled, it is sourced directly from the CLKIN pin. The external crystal oscillator is fed through the EFTB block, which is optionally bypassed.

13.2.1 CPU Clock

The CPU clock, CPUCLK, is sourced from the external crystal oscillator, the internal 24-MHz oscillator, or the internal 32-kHz low-power oscillator. The selected clock source can optionally be divided by 2^{n-1} where n is 0–7 (see Table 13-2 on page 25).

When it is not being used by the external crystal oscillator, the CLKOUT pin is driven from one of the many sources. This is used for test and also in some applications. The sources that drive the CLKOUT are:

■ CLKIN after the optional EFTB filter

■ Internal 24-MHz oscillator

■ Internal 32-kHz oscillator

■ CPUCLK after the programmable divider

Figure 13-1. CPU Clock Block Diagram
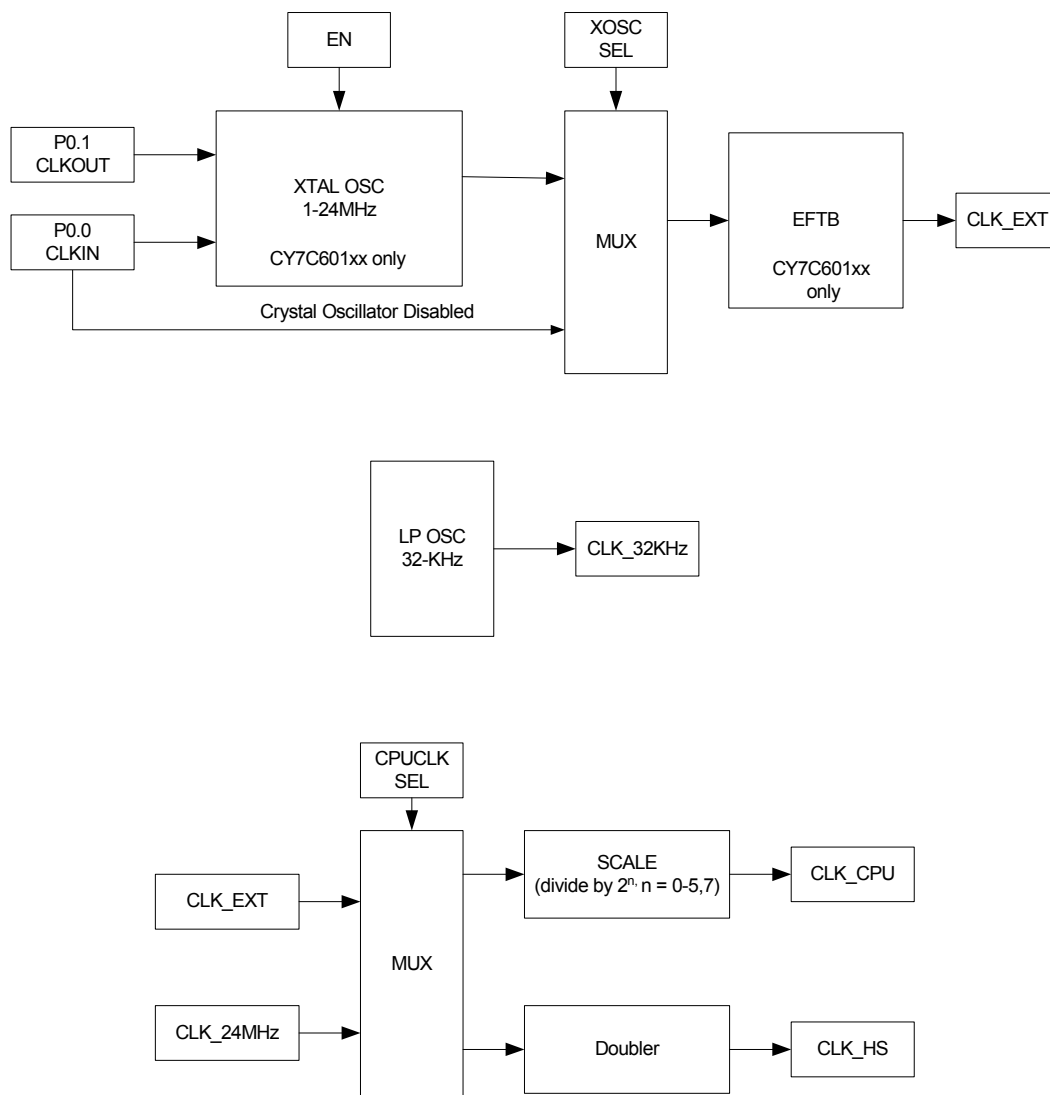
Table 13-1. CPU Clock Configuration (CPUCLKCR) [0x30] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved							CPUCLK Select
Read/Write	–	–	–	–	–	–	–	R/W
Default	0	0	0	0	0	0	0	0

Bit [7:1]: Reserved

Bit 0: CPU CLK select

0 = Internal 24-MHz oscillator

1 = External oscillator source

Note The CPU speed selection is configured using the OSC_CR0 Register (Table 13-2 on page 25).

Table 13-2. OSC Control 0 (OSC_CR0) [0x1E0] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved		No Buzz	Sleep Timer [1:0]		CPU Speed [2:0]		
Read/Write	–	–	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	1	0	0	0

Bit [7:6]: Reserved

Bit 5: No buzz

During sleep (the sleep bit is set in the CPU_SCR register—Table 14-1 on page 31), the LVD and POR detection circuit is turned on periodically to detect any POR and LVD events on the V_{CC} pin (the sleep duty cycle bits in the ECO_TR are used to control the duty cycle—Table 16-3 on page 36). To facilitate the detection of POR and LVD events, the 'No Buzz' bit is used to continuously enable the LVD and POR detection circuit during sleep. This results in a faster response to an LVD or POR event during sleep at the expense of a slightly higher than average sleep current. Obtaining the absolute lowest power usage in sleep mode requires the 'No Buzz' bit to be clear.

0 = The LVD and POR detection circuit is turned on periodically as configured in the sleep duty cycle.

1 = The sleep duty cycle value is overridden. The LVD and POR detection circuit is always enabled.

Note The periodic sleep duty cycle enabling is independent with the sleep interval shown in the Sleep [1:0] bits below.

Bit [4:3]: Sleep timer [1:0]

Sleep Timer [1:0]	Sleep Timer Clock Frequency (Nominal)	Sleep Period (Nominal)	Watchdog Period (Nominal)
00	512 Hz	1.95 ms	6 ms
01	64 Hz	15.6 ms	47 ms
10	8 Hz	125 ms	375 ms
11	1 Hz	1 sec	3 sec

Note Sleep intervals are approximate.

Bit [2:0]: CPU speed [2:0]

The enCoRe II LV operates over a range of CPU clock speeds. The reset value for the CPU Speed bits is zero; therefore, the default CPU speed is 3 MHz.

CPU Speed [2:0]	CPU when Internal Oscillator is selected	External Clock
000	3 MHz (Default)	Clock In/8
001	6 MHz	Clock In/4
010	12 MHz	Clock In/2
011	Reserved	Reserved
100	1.5 MHz	Clock In/16
101	750 kHz	Clock In/32
110	187 kHz	Clock In/128
111	Reserved	Reserved

Note This register exists in the second bank of I/O space. This requires setting the XIO bit in the CPU flags register.

Table 13-3. Clock I/O Configuration (CLKIOCR) [0x32] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved			XOSC Select	XOSC Enable	EFTB Disabled	CLKOUT Select	
Read/Write	–	–	–	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

Bit [7:5]: Reserved

Bit 4: XOSC select

This bit, when set, selects the external crystal oscillator clock as clock source of external clock. When selecting the crystal oscillator clock, first enable the crystal oscillator and wait for few cycles. This is the oscillator stabilization period. Then select the crystal clock as clock source. Similarly, to deselect crystal clock, first deselect crystal clock as clock source then disable the crystal oscillator.

0 = Not select external crystal oscillator clock.

1 = Select the external crystal oscillator clock.

Bit 3: XOSC enable

This bit is only available on the CY7C601xx.

This bit when set enables the external crystal oscillator. The external crystal oscillator shares pads CLKIN and CLKOUT with two GPIOs—P0.0 and P0.1 respectively. When the external crystal oscillator is enabled, the CLKIN signal comes from the external crystal oscillator block and the output enables on the GPIOs for P0.0 and P0.1 are disabled, eliminating the possibility of contention. When the external crystal oscillator is disabled, the source for CLKIN signal comes from the P0.0 GPIO input.

0 = Disable the external oscillator.

1 = Enable the external oscillator.

Note The external crystal oscillator startup time takes up to 2 ms.

Bit 2: EFTB disabled

This bit is only available on the CY7C601xx.

0 = Enable the EFTB filter.

1 = Disable the EFTB filter, causing CLKIN to bypass the EFTB filter.

Bit [1:0]: CLKOUT select

0 0 = Internal 24-MHz oscillator

0 1 = External oscillator source

1 0 = Internal 32-kHz low-power oscillator

1 1 = CPUCLK

13.2.2 Interval Timer Clock (ITMRCLK)

The ITMRCLK is sourced from the external crystal oscillator, internal 24-MHz oscillator, internal 32-kHz low-power oscillator, or the timer capture clock. A programmable prescaler of 1, 2, 3, or 4 then divides the selected source. The 12-bit programmable interval timer is a simple down counter with a programmable reload value. It provides a 1-μs resolution by default. When the down counter reaches zero, the next clock is spent reloading. The reload value is read and written when the counter is running, but ensure that the counter does not unintentionally reload when the 12-bit reload value is only partially stored between two writes of the 12-bit value. The programmable interval timer generates an interrupt to the CPU on each reload.

The parameters to be set show up on the device editor view of PSoC Designer when you place the enCoRe II LV timer user module. The parameters are PITIMER_Source and PITIMER_Divider. The PITIMER_Source is the clock to the timer and the PITIMER_Divider is the value the clock is divided by.

The interval register (PITMR) holds the value that is loaded into the PIT counter on terminal count.

The programmable interval timer resolution is configurable. For example:

TCAPCLK divide by x of CPU clock (for example TCAPCLK divide by 2 of a 24 MHz CPU clock gives a frequency of 12 MHz)

ITMRCLK divide by x of TCAPCLK (for example, ITMRCLK divide by 3 of TCAPCLK is 4 MHz so resolution is 0.25 μs).

Figure 13-2. Programmable Interval Timer Block Diagram
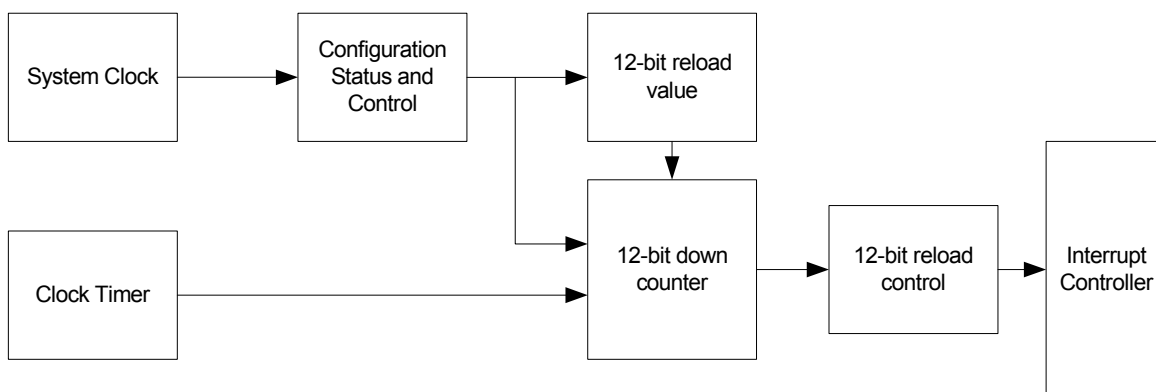
13.2.3 Timer Capture Clock (TCAPCLK)

The TCAPCLK is sourced from the external crystal oscillator, the internal 24-MHz oscillator or the internal 32-kHz low-power oscillator. A programmable prescaler of 2, 4, 6, or 8 then divides the selected source.

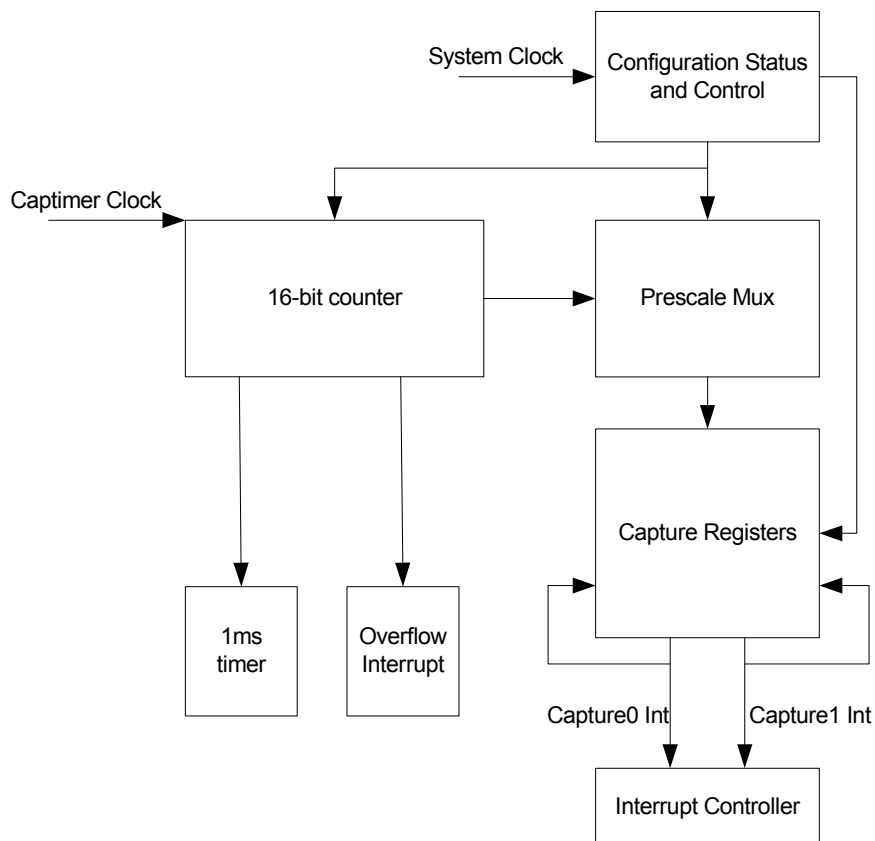**Figure 13-3. Timer Capture Block Diagram**

Table 13-1. Timer Clock Configuration (TMRCLKCR) [0x31] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	TCAPCLK divider		TCAPCLK select		ITMRCLK divider		ITMRCLK select	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	1	0	0	0	1	1	1	1

Bit [7:6]: TCAPCLK divider [1:0]

TCAPCLK Divider controls the TCAPCLK divisor.

0 0 = Divider Value 2

0 1 = Divider Value 4

1 0 = Divider Value 6

1 1 = Divider Value 8

Bit [5:4]: TCAPCLK select

The TCAPCLK Select field controls the source of the TCAPCLK.

0 0 = Internal 24-MHz oscillator

0 1 = External Crystal Oscillator—external crystal oscillator on CLKIN and CLKOUT if the external crystal oscillator is enabled, CLKIN input if the external crystal oscillator is disabled (the XOSC Enable bit of the CLKIOCR Register is cleared—Table 13-3 on page 26.)

1 0 = Internal 32-kHz oscillator

1 1 = TCAPCLK disabled

Note The 1024 μs interval timer is based on the assumption that TCAPCLK is running at 4 MHz. Changes in TCAPCLK frequency cause a corresponding change in the 1024 μs interval timer frequency.

Bit [3:2]: ITMRCLK divider

ITMRCLK Divider controls the ITMRCLK divisor.

0 0 = Divider value of 1

0 1 = Divider value of 2

1 0 = Divider value of 3

1 1 = Divider value of 4

Bit [1:0]: ITMRCLK select

0 0 = Internal 24-MHz oscillator

0 1 = External crystal oscillator—external crystal oscillator on CLKIN and CLKOUT if the external crystal oscillator is enabled, CLKIN input if the external crystal oscillator is disabled.

1 0 = Internal 32-kHz oscillator

1 1 = TCAPCLK

Note Changing the source of TMRCLK requires both the source and destination clocks to be running. It is not possible to change the clock source away from TCAPCLK after that clock is stopped.

13.2.4 Internal Clock Trim

Table 13-2. IOSC Trim (IOSCTR) [0x34] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	foffset[2:0]			Gain[4:0]				
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	D	D	D	D	D

The IOSC calibrate register is used to calibrate the internal oscillator. The reset value is undefined, but during boot the SROM writes a calibration value that is determined during manufacturing test. The 'D' indicates that the default value is trimmed to 24 MHz at 3.30 V at power on.

Bit [7:5]: foffset [2:0]

This value is used to trim the frequency of the internal oscillator. These bits are not used in factory calibration and is zero. Setting each of these bits causes the appropriate fine offset in oscillator frequency.

foffset bit 0 = 7.5 kHz

foffset bit 1 = 15 kHz

foffset bit 2 = 30 kHz

Bit [4:0]: Gain [4:0]

The effective frequency change of the offset input is controlled through the gain input. A lower value of the gain setting increases the gain of the offset input. This value sets the size of each offset step for the internal oscillator. Nominal gain change (kHz/offsetStep) at each bit, typical conditions (24 MHz operation):

Gain bit 0 = –1.5 kHz

Gain bit 1 = –3.0 kHz

Gain bit 2 = –6 kHz

Gain bit 3 = –12 kHz

Gain bit 4 = –24 kHz

13.2.5 External Clock Trim

Table 13-3. XOSC Trim (XOSCTR) [0x35] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved			XOSC XGM [2:0]			Reserved	Mode
Read/Write	–	–	–	R/W	R/W	R/W	–	R/W
Default	0	0	0	D	D	D	–	D

This register is used to calibrate the external crystal oscillator. The reset value is undefined, but during boot the SROM writes a calibration value that is determined during manufacturing test. This is the meaning of 'D' in the default field.

Bit [7:5]: Reserved

Bit [4:2]: XOSC XGM [2:0]

Amplifier transconductance setting. The Xgm settings are recommended for resonators with frequencies of interest for the enCoRe II LV as below:

Resonator	XGM Setting	Worst Case R (Ohms)
6 MHz Crystal	001	403
12 MHz Crystal	011	201
Reserved	111	-
6 MHz Ceramic	001	70.4
12 MHz Ceramic	011	41

Bit 1: Reserved

Bit 0: Mode

0 = Oscillator mode

1 = Fixed maximum bias test mode

13.2.6 LPOSC Trim

Table 13-4. LPOSC Trim (LPOSCTR) [0x36] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	32 kHz Low Power	Reserved	32 kHz Bias Trim [1:0]		32 kHz Freq Trim [3:0]			
Read/Write	R/W	–	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	–	D	D	D	D	D	D

This register is used to calibrate the 32-kHz low-speed oscillator. The reset value is undefined but during boot the SROM writes a calibration value that is determined during manufacturing test. This is the meaning of 'D' in the Default field. The trim value is adjusted vs. voltage as noted in Table 13-1 on page 24.

Bit 7: 32 kHz low-power

0 = The 32-kHz low-speed oscillator operates in normal mode.

1 = The 32-kHz low-speed oscillator operates in a low-power mode. The oscillator continues to function normally but with reduced accuracy.

Bit 6: Reserved

Bit [5:4]: 32 kHz bias trim [1:0]

These bits control the bias current of the low power oscillator.

0 0 = Mid bias

0 1 = High bias

1 0 = Reserved

1 1 = Reserved

Note Do not program the 32-kHz bias trim [1:0] field with the reserved 10b value as the oscillator does not oscillate at all corner conditions with this setting.

Bit [3:0]: 32-kHz Freq Trim [3:0]

These bits are used to trim the frequency of the low-power oscillator.

13.3 CPU Clock During Sleep Mode

When the CPU enters sleep mode the CPUCLK select (Bit 0, Table 13-1 on page 24) is forced to the internal oscillator, and the oscillator is stopped. When the CPU comes out of sleep mode it runs on the internal oscillator. The internal oscillator recovery time is three clock cycles of the internal 32-kHz low-power oscillator.

If the system requires the CPU to run off the external clock after waking from sleep mode, firmware needs to switch the clock source for the CPU. If the external clock source is the external oscillator and the oscillator is disabled, firmware needs to enable the external oscillator, wait for it to stabilize, and then change the clock source.

14. Reset

The microcontroller supports two types of resets: POR and watchdog reset (WDR). When reset is initiated, all registers are restored to their default states and all interrupts are disabled.

The occurrence of a reset is recorded in the system status and control register (CPU_SCR). Bits within this register record the occurrence of POR and WDR reset respectively. The firmware interrogates these bits to determine the cause of a reset.

The microcontroller resumes execution from flash address 0x0000 after a reset. The internal clocking mode is active after a reset, until changed by user firmware.

Note The CPU clock defaults to 3 MHz (an internal 24-MHz oscillator divide-by-8 mode) at POR to guarantee operation at the low V_{CC} that might be present during the supply ramp.

Table 14-1. System Status and Control Register (CPU_SCR) [0xFF] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	GIES	Reserved	WDRS	PORS	Sleep	Reserved	Reserved	Stop
Read/Write	R	–	R/C[3]	R/C[3]	R/W	–	–	R/W
Default	0	0	0	1	0	1	0	0

The bits of the CPU_SCR register are used to convey status and control of events for various functions of an enCoRe II LV device.

Bit 7: GIES

The Global Interrupt Enable Status 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 services interrupts.

0 = Global interrupts disabled

1 = Global interrupt enabled

Bit 6: Reserved

Bit 5: WDRS

The WDRS bit is set by the CPU to indicate that a WDR event has occurred. The user can read this bit to determine the type of reset that has occurred. The user can clear but not set this bit.

0 = No WDR

1 = A WDR event has occurred

Bit 4: PORS

The PORS bit is set by the CPU to indicate that a POR event has occurred. The user can read this bit to determine the type of reset that has occurred. The user can clear but not set this bit.

0 = No POR

1 = A POR event has occurred. (Note that WDR events does not occur until this bit is cleared.)

Bit 3: SLEEP

Set by the user to enable CPU sleep state. CPU remains in sleep mode until any interrupt is pending. The Sleep bit is covered in more detail in the Sleep Mode section.

0 = Normal operation

1 = Sleep

Bit [2:1]: Reserved

Bit 0: STOP

This bit is set by the user to halt the CPU. The CPU remains halted until a reset (WDR, POR, or external reset) takes place. If an application wants to stop code execution until a reset, the preferred method is to use the HALT instruction rather than writing to this bit.

0 = Normal CPU operation

1 = CPU is halted (not recommended)

Note
3. C = Clear. This bit can only be cleared by the user and cannot be set by firmware.

14.1 Power On Reset

POR occurs every time the power to the device is switched on. POR is released when the supply is typically 2.6 V for the upward supply transition, with typically 50 mV of hysteresis during the power on transient. Bit 4 of the system status and control register (CPU_SCR) is set to record this event (the register contents are set to 00010000 by the POR). After a POR, the microprocessor is held off for approximately 20 ms for the V_{CC} supply to stabilize before executing the first instruction at address 0x00 in flash. If the V_{CC} voltage drops below the POR downward supply trip point, POR is reasserted. The V_{CC} supply needs to ramp linearly from 0 to V_{CC} in less than 200 ms.

Note The PORS status bit is set at POR and is only cleared by the user; it cannot be set by firmware.

14.2 Watchdog Timer Reset

The user has the option to enable the WDT. The WDT is enabled by clearing the PORS bit. When the PORS bit is cleared, the WDT cannot be disabled. The only exception to this is if a POR event takes place, which disables the WDT.

The sleep timer is used to generate the sleep time period and the watchdog time period. The sleep timer uses the internal 32-kHz low-power oscillator system clock to produce the sleep time period. The user programs the sleep time period using the sleep timer bits of the OSC_CR0 register (Table 13-2 on page 25). When the sleep time elapses (sleep timer overflows), an interrupt to the sleep timer interrupt vector is generated.

The watchdog timer period is automatically set to be three counts of the sleep timer overflow. This represents between two and three sleep intervals depending on the count in the sleep timer at the previous WDT clear. When this timer reaches three, a WDR is generated. The user either clears the WDT, or the WDT and the sleep timer. Whenever the user writes to the reset WDT register (RES_WDT), the WDT is cleared. If the data written is the hex value 0x38, the sleep timer is also cleared at the same time.

Table 14-2. Reset Watchdog Timer (RESWDT) [0xE3] [W]

Bit #	7	6	5	4	3	2	1	0
Field	Reset Watchdog Timer [7:0]							
Read/Write	W	W	W	W	W	W	W	W
Default	0	0	0	0	0	0	0	0

Any write to this register clears the watchdog timer, a write of 0x38 also clears the sleep timer.

Bit [7:0]: Reset watchdog timer [7:0]

15. Sleep Mode

The CPU is put to sleep only by the firmware. This is accomplished by setting the sleep bit in the system status and control register (CPU_SCR). This stops the CPU from executing instructions, and the CPU remains asleep until an interrupt is pending, or there is a reset event (either a POR or a WDT reset).

The low-voltage detection (LVD) circuit drops into fully functional power reduced states, and the latency for the LVD is increased. The actual latency is traded against power consumption by changing sleep duty cycle field of the ECO_TR register.

The internal 32-kHz low-speed oscillator remains running. Before entering the suspend mode, firmware optionally configures the 32-kHz low-speed oscillator to operate in a low-power mode to help reduce the overall power consumption (using the 32-kHz low-power bit, as shown in Table 13-4 on page 30). This helps to save approximately 5 μA; however, the trade off is that the 32-kHz low-speed oscillator is less accurate (–53.12% to +56.25% deviation).

All interrupts remain active. Only the occurrence of an interrupt wakes the part from sleep. The stop bit in the system status and control register (CPU_SCR) is cleared for a part to resume out of sleep. The global interrupt enable bit of the CPU flags register (CPU_F) does not have any effect. Any unmasked interrupt wakes the system. As a result, any interrupt not intended for waking is disabled through the interrupt mask registers.

When the CPU enters sleep mode the CPUCLK select (Bit 1, Table 13-1 on page 24) is forced to the internal oscillator. The internal oscillator recovery time is three clock cycles of the internal 32-kHz low-power oscillator. The internal 24-MHz oscillator restarts immediately on exiting sleep mode. If the external crystal oscillator is used, the firmware needs to switch the clock source for the CPU.

Unlike the internal 24-MHz oscillator, the external oscillator is not automatically shut down during sleep. Systems that need the external oscillator disabled in sleep mode must disable the external oscillator before entering sleep mode. In systems where the CPU runs off the external oscillator, the firmware needs to switch the CPU to the internal oscillator before disabling the external oscillator.

On exiting sleep mode, after the clock is stable and the delay time has expired, the instruction immediately following the sleep instruction is executed before the interrupt service routine (if enabled).

The sleep interrupt allows the microcontroller to wake up periodically and poll system components while maintaining very low average power consumption. The sleep interrupt is also used to provide periodic interrupts during non-sleep modes.

15.1 Sleep Sequence

The sleep bit is an input into the sleep logic circuit. This circuit is designed to sequence the device in and out of the hardware sleep state. The hardware sequence to put the device to sleep is shown in Figure 15-1 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. The CPU samples BRQ on the positive edge of CPUCLK.

2. Due to the specific timing of the register write, the CPU issues a bus request acknowledge (BRA) on the following positive edge of the CPU clock. 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 15-1 the CPU is halted and the system wide power down signal is asserted.

3. The system wide PD signal controls several major circuit blocks: the flash memory module, the internal 24-MHz oscillator, the EFTB filter, and the bandgap voltage reference. These circuits transition into a zero power state. The only operational circuits on chip are the low-power oscillator, the bandgap refresh circuit, and the supply voltage monitor (POR/LVD) circuit.

The external crystal oscillator on enCoRe II LV devices is not automatically powered down when the CPU enters the sleep state. Firmware must explicitly disable the external crystal oscillator to reduce power to levels specified.

15.1.1 Low-Power in Sleep Mode

To achieve the lowest possible power consumption during suspend or sleep, the following conditions are observed in addition to considerations for the sleep timer and external crystal oscillator:

■ All GPIOs are set to outputs and driven low

■ Clear P11CR[0], P10CR[0]

■ Set P10CR[1]

■ Make sure the 32-kHz oscillator clock is not selected as clock source to ITMRCLK, TCAPCLK, and not even as clock output source onto P01_CLKOUT pin.

All the other blocks go to the power-down mode automatically on suspend.

The following steps are user-configurable and help in reducing the average suspend mode power consumption.

1. Configure the power supply monitor at a large regular intervals, control register bits are 1,EB[7:6] (power system sleep duty cycle PSSDC[1:0]).
2. Configure the low-power oscillator into low power mode, control register bit is LOPSCTR[7].

For low-power considerations during sleep when external clock is used as the CPUCLK source, the clock source must be held low to avoid unintentional leakage current. If the clock is held high, then there may be a leakage through M8C. To avoid current consumption make sure ITMRCLK and TCPCLK are not sourced by either low-power 32-kHz oscillator or 24-MHz crystal-less oscillator. Do not select the 24-MHz or 32-kHz oscillator clocks on to the P01_CLKOUT pin.

Figure 15-1. Sleep Timing

Firmware write to SCR SLEEP bit causes an immediate BRQ

CPU captures BRQ on next CPUCLK edge

CPU responds with a BRA

On the falling edge of CPUCLK, PD is asserted. The 24/48 MHz system clock is halted; the Flash and bandgap are powered down
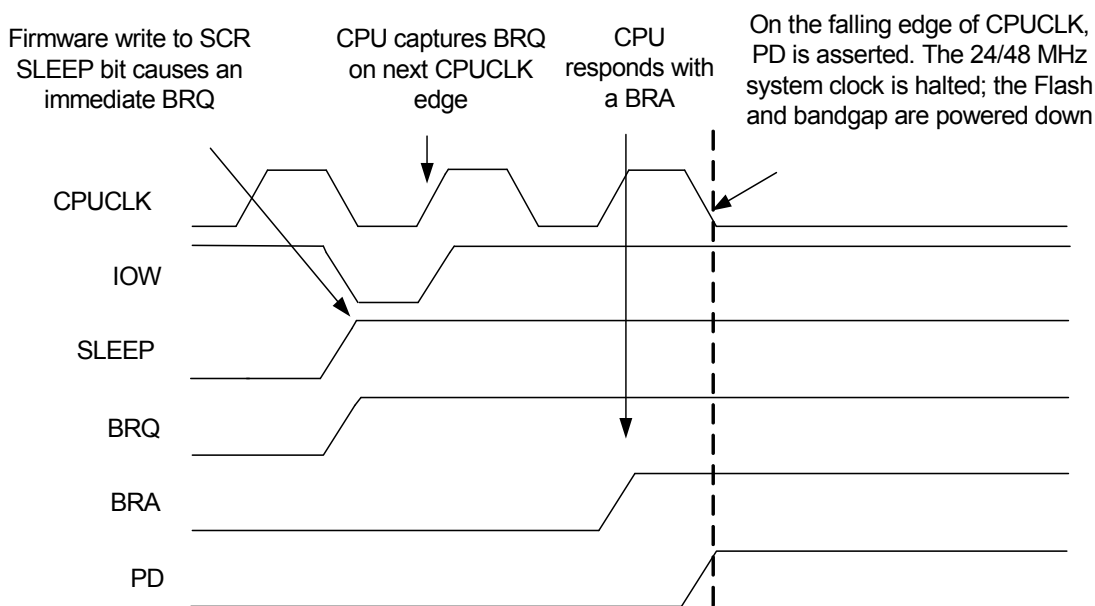
15.2 Wakeup Sequence

When asleep, the only event that wakes the system up is an interrupt. The global interrupt enable of the CPU flag register need not be set. Any unmasked interrupt wakes 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. This is done to sequence a startup delay and 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 enable the oscillator, bandgap, and LVD and POR circuits time to settle before actually being used in the system. As shown in Figure 15-2, 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, internal oscillator, EFTB, and bandgap circuit are all powered up to a normal operating state.

3. At the following positive edge of the 32-kHz clock, the current values for the precision POR and LVD have settled and are sampled.

4. At the following negative edge of the 32-kHz clock (after about 15 µs nominal), the BRQ signal is negated by the sleep logic circuit. On the following CPUCLK, BRA is negated by the CPU and instruction execution resumes. Note that in Figure 15-2 fixed function blocks, such as flash, internal oscillator, EFTB, and bandgap, have about 15 µs start-up. The wakeup times (interrupt to CPU operational) range from 75 µs to 105 µs.

Figure 15-2. Wakeup Timing
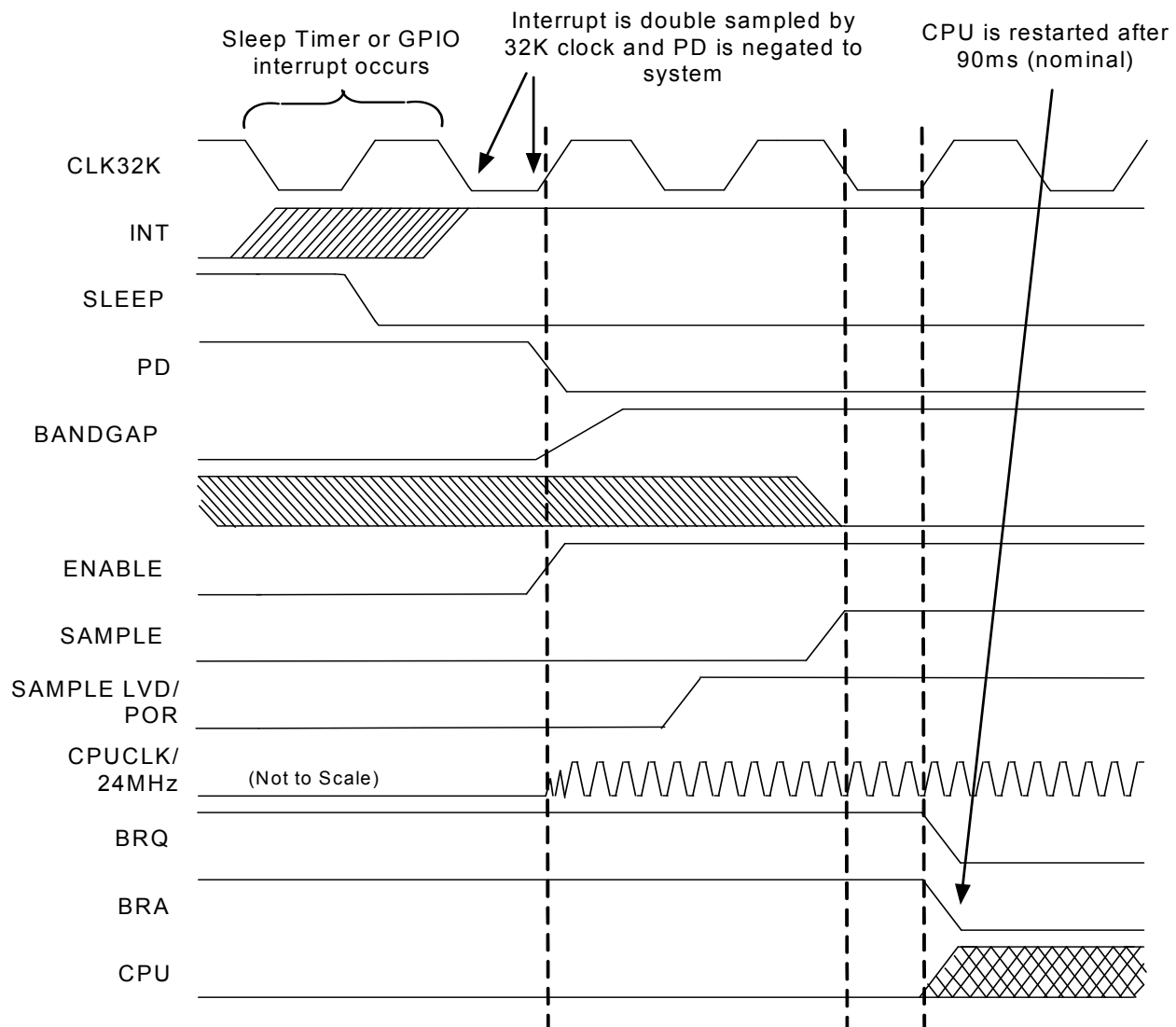
16. Low-Voltage Detect Control

Table 16-1. Low-Voltage Control Register (LVDCR) [0x1E3] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved		PORLEV[1:0]		Reserved	VM[2:0]		
Read/Write	–	–	R/W	R/W	–	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

This register controls the configuration of the POR and LVD circuit. This register is accessed only in the second bank of I/O space. This requires setting the XIO bit in the CPU flags register.

Bit [7:6]: Reserved

Bit [5:4]: PORLEV[1:0]

This field controls the level below which the precision power on-reset (PPOR) detector generates a reset.

0 0 = 2.7 V range (trip near 2.6 V)

0 1 = 3 V range (trip near 2.9 V)

1 0 = Reserved

1 1 = PPOR does not generate a reset, but values read from the voltage monitor comparators register (Table 16-2 on page 36) give the internal PPOR comparator state with trip point set to the 3-V range setting.

Bit 3: Reserved

Bit [2:0]: VM[2:0]

This field controls the level below which the low-voltage-detect trips—possibly generating an interrupt and the level at which Flash is enabled for operation.

Note This register exists in the second bank of I/O space. This requires setting the XIO bit in the CPU flags register.

VM[2:0]	LVD Trip Point (V)		
	Min	Max	Typical
000	2.69	2.72	2.7
001	2.90	2.94	2.92
010	3.00	3.04	3.02
011	3.10	3.15	3.13
100	Reserved		
101	Reserved		
110	Reserved		
111	Reserved		

16.1 POR Compare State

Table 16-2. Voltage Monitor Comparators Register (VLTCMP) [0x1E4] [R]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved						LVD	PPOR
Read/Write	–	–	–	–	–	–	R	R
Default	0	0	0	0	0	0	0	0

This read-only register allows reading the current state of the LVD and PPOR comparators.

Bit [7:2]: Reserved

Bit 1: LVD

This bit is set to indicate that the LVD comparator has tripped, indicating that the supply voltage has gone below the trip point set by VM[2:0] (See Table 16-1 on page 35).

0 = No low-voltage-detect event

1 = A low-voltage-detect has tripped

Bit 0: PPOR

This bit is set to indicate that the PPOR comparator has tripped, indicating that the supply voltage is below the trip point set by PORLEV[1:0].

0 = No PPOR event

1 = A PPOR event has occurred

Note This register exists in the second bank of I/O space. This requires setting the XIO bit in the CPU flags register.

16.2 ECO Trim Register

Table 16-3. ECO (ECO_TR) [0x1EB] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Sleep duty cycle [1:0]		Reserved					
Read/Write	R/W	R/W	–	–	–	–	–	–
Default	0	0	0	0	0	0	0	0

This register controls the ratios (in numbers of 32 kHz clock periods) of "on" time versus "off" time for LVD and POR detection circuit.

Bit [7:6]: Sleep duty cycle [1:0]

0 0 = 1/128 periods of the Internal 32 kHz low speed oscillator.

0 1 = 1/512 periods of the Internal 32 kHz low speed oscillator.

1 0 = 1/32 periods of the Internal 32 kHz low speed oscillator.

1 1 = 1/8 periods of the Internal 32 kHz low speed oscillator.

Note This register is only accessed in the second bank of I/O space. This requires setting the XIO bit in the CPU flags register.

17. General-Purpose I/O Ports

17.1 Port Data Registers

17.1.1 P0 Data

Table 17-1. P0 Data Register (P0DATA)[0x00] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	P0.7	P0.6/TIO1	P0.5/TIO0	P0.4/INT2	P0.3/INT1	P0.2/INT0	P0.1/CLKOUT	P0.0/CLKIN
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

This register contains the data for Port 0. Writing to this register sets the bit values to be output on output enabled pins. Reading from this register returns the current state of the Port 0 pins.

Bit 7: P0.7 Data

Bit [6:5]: P0.6–P0.5 Data/TIO1 and TIO0

Beside their use as the P0.6–P0.5 GPIOs, these pins are also used for alternate functions as the Capture Timer input or timer output pins (TIO1 and TIO0). To configure the P0.5 and P0.6 pins, refer to the P0.5/TIO0–P0.6/TIO1 Configuration Register (Table 17-4 on page 41).

Bit [4:2]: P0.4–P0.2 Data/INT2–INT0

Beside their use as the P0.4–P0.2 GPIOs, these pins are also used for the alternate functions as the interrupt pins (INT0–INT2). To configure the P0.4–P0.2 pins, refer to the P0.2/INT0–P0.4/INT2 Configuration Register (Table 17-3 on page 40).

Bit 1: P0.1/CLKOUT

Beside its use as the P0.1 GPIO, this pin is also used for the alternate function as the CLK OUT pin. To configure the P0.1 pin, refer to the P0.1/CLKOUT Configuration Register (Table 17-2 on page 40).

Bit 0: P0.0/CLKIN

Beside its use as the P0.0 GPIO, this pin is also used for the alternate function as the CLKIN pin. To configure the P0.0 pin, refer to the P0.0/CLKIN Configuration Register (Table 17-1 on page 39).

17.1.2 P1 Data

Table 17-2. P1 Data Register (P1DATA) [0x01] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	P1.7	P1.6/SMISO	P1.5/SMOSI	P1.4/SCLK	P1.3/SSEL	P1.2	P1.1	P1.0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

This register contains the data for Port 1. Writing to this register sets the bit values to be output on output enabled pins. Reading from this register returns the current state of the Port 1 pins.

Bit 7: P1.7 data

Bit [6:3]: P1.6–P1.3 Data/SPI Pins (SMISO, SMOSI, SCLK, SSEL)

Beside their use as the P1.6–P1.3 GPIOs, these pins are also used for the alternate function as the SPI interface pins. To configure the P1.6–P1.3 pins, refer to the P1.3–P1.6 configuration register (Table 17-9 on page 42).

Bit [2:0]: P1.2–P1.0

17.1.3 P2 Data

Table 17-3. P2 Data Register (P2DATA) [0x02] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	P2.7–P2.2						P2.1–P2.0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

This register contains the data for Port 2. Writing to this register sets the bit values to be output on output enabled pins. Reading from this register returns the current state of the Port 2 pins.

Bit [7:2]: P2 Data [7:2]

Bit [1:0]: P2 Data [1:0]

17.1.4 P3 Data

Table 17-4. P3 Data Register (P3DATA) [0x03] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	P3.7–P3.2						P3.1–P3.0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

This register contains the data for Port 3. Writing to this register sets the bit values to be output on output enabled pins. Reading from this register returns the current state of the Port 3 pins.

Bit [7:2]: P3 Data [7:2]

Bit [1:0]: P3 Data [1:0]

17.1.5 P4 Data

Table 17-5. P4 Data Register (P4DATA) [0x04] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved				P4.3–P4.0			
Read/Write	–	–	–	–	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

This register contains the data for Port 4. Writing to this register sets the bit values to be output on output enabled pins. Reading from this register returns the current state of the Port 2 pins.

Bit [7:4]: Reserved

Bit [3:0]: P4 data [3:0]

P4.3–P4.0 only exist in the CY7C601xx.

17.2 GPIO Port Configuration

All GPIO configuration registers have common configuration controls. By default all GPIOs are configured as inputs. To prevent the inputs from floating, pull-up resistors are enabled. Firmware configures each of the GPIOs before use. The following are bit definitions of the GPIO configuration registers.

17.2.1 Int Enable

When set, the Int Enable bit allows the GPIO to generate interrupts. Interrupt generate occurs regardless of whether the pin is configured for input or output. All interrupts are edge-sensitive. However, for interrupts that are shared by multiple sources (ports 2, 3, and 4), all inputs are deasserted before a new interrupt occurs.

When clear, the corresponding interrupt is disabled on the pin.

It is possible to configure GPIOs as outputs, enable the interrupt on the pin, and then generate the interrupt by driving the appropriate pin state. This is useful in test and may find value in applications too.

17.2.2 Int Act Low

When clear, the corresponding interrupt is active HIGH. When set, the interrupt is active LOW. For P0.2–P0.4 Int Act Low makes interrupts active on the rising edge. Int Act Low set makes interrupts active on the falling edge.

17.2.3 TTL Thresh

When set, the input has TTL threshold. When clear, the input has standard CMOS threshold.

Note The GPIOs default to CMOS threshold. The user's firmware must configure the threshold to TTL mode if necessary.

17.2.4 High Sink

When set, the output sinks up to 50 mA.

When clear, the output sinks up to 8 mA.

On the CY7C601xx, only the P3.7, P2.7, P0.1, and P0.0 have a 50 mA sink drive capability. Other pins have a 8-mA sink drive capability.

On the CY7C602xx, only the P1.7–P1.3 have a 50-mA sink drive capability. Other pins have an 8-mA sink drive capability.

17.2.5 Open Drain

When set, the output on the pin is determined by the port data register. If the corresponding bit in the port data register is set, the pin is in high-impedance state; if it is clear, the pin is driven low.

When clear, the output is driven low or high.

17.2.6 Pull-up Enable

When set the pin has a 7 K pull-up to V_{DD}.

When clear, the pull-up is disabled.

17.2.7 Output Enable

When set, the output driver of the pin is enabled.

When clear, the output driver of the pin is disabled.

For pins with shared functions there are some special cases.

P0.0(CLKIN) and P0.1(CLKOUT) are not output-enabled when the crystal oscillator is enabled. Output enables for these pins are overridden by XOSC Enable.

17.2.8 SPI Use

The P1.3(SSEL), P1.4(SCLK), P1.5(SMOSI), and P1.6(SMISO) pins are used for their dedicated functions or for GPIO. To enable the pin for GPIO, clear the corresponding SPI Use bit. The SPI function controls the output enable for its dedicated function pins when their GPIO enable bit is clear.

Figure 17-1. GPIO Block Diagram
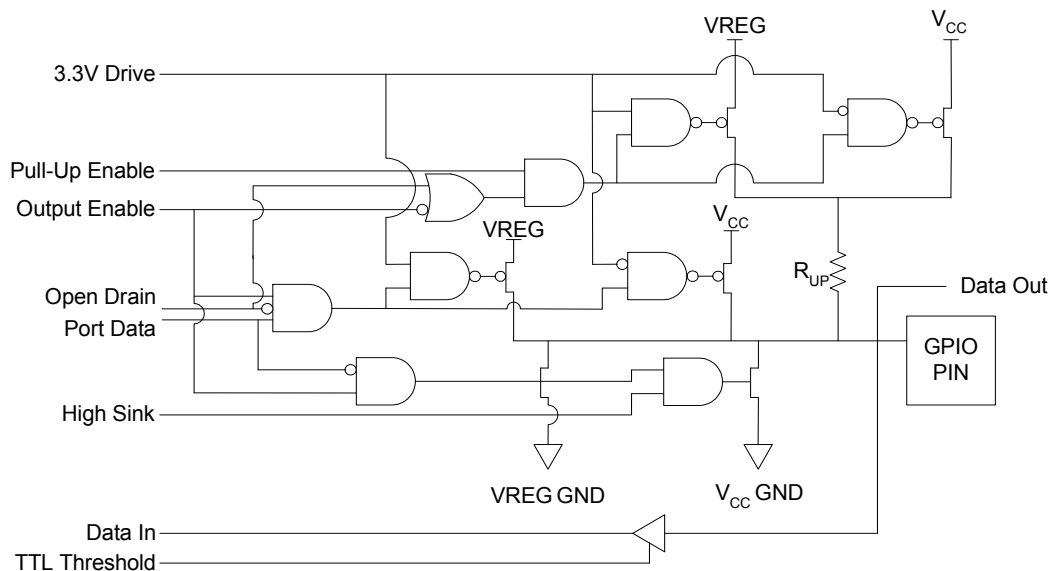
17.2.9 P0.0/CLKIN Configuration

Table 17-1. P0.0/CLKIN Configuration (P00CR) [0x05] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved	Int Enable	Int Act Low	TTL Thresh	High Sink	Open Drain	Pull-up Enable	Output Enable
Read/Write	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

This pin is shared between the P0.0 GPIO use and the CLKIN pin for the external crystal oscillator. When the external oscillator is enabled the settings of this register are ignored.

The alternate function of the pin as the CLKIN is only available in the CY7C601xx. When the external oscillator is enabled (the XOSC Enable bit of the CLKIOCR Register is set—Table 13-3 on page 26), the GPIO function of the pin is disabled.

The 50-mA sink drive capability is only available in the CY7C601xx. In the CY7C602xx, only an 8-mA sink drive capability is available on this pin regardless of the setting of the high sink bit.

17.2.10 P0.1/CLKOUT Configuration

Table 17-2. P0.1/CLKOUT Configuration (P01CR) [0x06] R/W]

Bit #	7	6	5	4	3	2	1	0
Field	CLK Output	Int Enable	Int Act Low	TTL Thresh	High Sink	Open Drain	Pull-up Enable	Output Enable
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

This pin is shared between the P0.1 GPIO use and the CLKOUT pin for the external crystal oscillator. When the external oscillator is enabled the settings of this register are ignored. When CLK output is set, the internally selected clock is sent out onto P0.1CLKOUT pin.

The alternate function of the pin as the CLKOUT is only available in the CY7C601xx. When the external oscillator is enabled (the XOSC enable bit of the CLKIOCR register is set—Table 13-3 on page 26), the GPIO function of the pin is disabled.

The 50 mA sink drive capability is only available in the CY7C601xx. In the CY7C602xx, only 8 mA sink drive capability is available on this pin regardless of the setting of the high sink bit.

Bit 7: CLK output

0 = The clock output is disabled.

1 = The clock selected by the CLK Select field (Bit [1:0] of the CLKIOCR register—Table 13-3 on page 26) is driven out to the pin.

17.2.11 P0.2/INT0–P0.4/INT2 Configuration

Table 17-3. P0.2/INT0–P0.4/INT2 Configuration (P02CR–P04CR) [0x07–0x09] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved		Int Act Low	TTL Thresh	Reserved	Open Drain	Pull-up Enable	Output Enable
Read/Write	–	–	R/W	R/W	–	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

These registers control the operation of pins P0.2–P0.4 respectively. These pins are shared between the P0.2–P0.4 GPIOs and the INT0–INT2. The INT0–INT2 interrupts are different from all other GPIO interrupts. These pins are connected directly to the interrupt controller to provide three edge-sensitive interrupts with independent interrupt vectors. These interrupts occur on a rising edge when Int Act Low is clear and on a falling edge when Int Act Low is set. These pins are enabled as interrupt sources in the interrupt controller registers (Table 20-7 on page 59 and Table 20-5 on page 58).

To use these pins as interrupt inputs, configure them as inputs by clearing the corresponding Output Enable. If the INT0–INT2 pins are configured as outputs with interrupts enabled, firmware generates an interrupt by writing the appropriate value to the P0.2, P0.3, and P0.4 data bits in the P0 Data Register.

Regardless of whether the pins are used as interrupt or GPIO pins, the Int Enable, Int Act Low, TTL Threshold, Open Drain, and pull-up enable bits control the behavior of the pin.

The P0.2/INT0–P0.4/INT2 pins are individually configured with the P02CR (0x07), P03CR (0x08), and P04CR (0x09) respectively.

Note Changing the state of the Int Act Low bit generates an unintentional interrupt. When configuring these interrupt sources, follow this procedure:

1. Disable interrupt source
2. Configure interrupt source
3. Clear any pending interrupts from the source
4. Enable interrupt source

17.2.12 P0.5/TIO0–P0.6/TIO1 Configuration

Table 17-4. P0.5/TIO0–P0.6/TIO1 Configuration (P05CR–P06CR) [0x0A–0x0B] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	TIO Output	Int Enable	Int Act Low	TTL Thresh	Reserved	Open Drain	Pull-up Enable	Output Enable
Read/Write	R/W	R/W	R/W	R/W	–	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

These registers control the operation of pins P0.5 through P0.6 respectively.

P0.5 and P0.6 are shared with TIO0 and TIO1 respectively. To use these pins as capture timer inputs, configure them as inputs by clearing the corresponding Output Enable. To use TIO0 and TIO1 as timer outputs, set the TIOx Output and Output Enable bits. If these pins are configured as outputs and the TIO output bit is clear, the firmware controls the TIO0 and TIO1 inputs by writing the value to the P0.5 and P0.6 data bits in the P0 data register.

Regardless of whether either pin is used as a TIO or GPIO pin the Int Enable, Int Act Low, TTL threshold, open drain, and pull-up enable control the behavior of the pin.

TIO0(P0.5) when enabled outputs a positive pulse from the 1024 μs interval timer. This is the same signal that is used internally to generate the 1024 μs timer interrupt. This signal is not gated by the interrupt enable state. The pulse is active for one cycle of the capture timer clock.

TIO1(P0.6) when enabled outputs a positive pulse from the programmable interval timer. This is the same signal that is used internally to generate the programmable timer interval interrupt. This signal is not gated by the interrupt enable state. The pulse is active for one cycle of the interval timer clock.

The P0.5/TIO0 and P0.6/TIO1 pins are individually configured with the P05CR (0x0A) and P06CR (0x0B), respectively.

17.2.13 P0.7 Configuration

Table 17-5. P0.7 Configuration (P07CR) [0x0C] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved	Int Enable	Int Act Low	TTL Thresh	Reserved	Open Drain	Pull-up Enable	Output Enable
Read/Write	–	R/W	R/W	R/W	–	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

This register controls the operation of pin P0.7.

17.2.14 P1.0 Configuration

Table 17-6. P1.0 Configuration (P10CR) [0x0D] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved	Int Enable	Int Act Low	Reserved			P1.0 and P1.1 Pull-up Enable	Output Enable
Read/Write	R/W	R/W	R/W	–	–	–	–	R/W
Default	0	0	0	0	0	0	0	0

This register controls the operation of the P1.0 pin.

Bit1: P1.0 and P1.1 Pull-up enable

0 = Disable the P1.0 and P1.1 pull-up resistors.

1 = Enable the internal pull-up resistors for both the P1.0 and P1.1. Each of the P1.0 and P1.1 pins is pulled up with R_{UP1} (see DC Characteristics on page 60).

Note There is no 2 mA sourcing capability on this pin. The pin can only sink 5 mA at V_{OL3} (see DC Characteristics on page 60)

The P1.0 is an open drain only output. It actively drives a signal low, but cannot actively drive a signal high.

If this pin is used as a general purpose output, it draws current. It is therefore configured as an input to reduce current draw.

17.2.15 P1.1 Configuration

Table 17-7. P1.1 Configuration (P11CR) [0x0E] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved	Int Enable	Int Act Low	Reserved		Open Drain	Reserved	Output Enable
Read/Write	–	R/W	R/W	–	–	R/W	–	R/W
Default	0	0	0	0	0	0	0	0

This register controls the operation of the P1.1 pin.

The pull-up resistor on this pin is enabled by the P10CR Register.

Note There is no 2 mA sourcing capability on this pin. The pin can only sink 5 mA at V_{OL3} (see DC Characteristics on page 60)

If this pin is used as a general purpose output, it draws current. It is, therefore, configured as an input to reduce current draw.

17.2.16 P1.2 Configuration

Table 17-8. P1.2 Configuration (P12CR) [0x0F] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	CLK Output	Int Enable	Int Act Low	TTL Threshold	Reserved	Open Drain	Pull-up Enable	Output Enable
Read/Write	R/W	R/W	R/W	R/W	–	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

This register controls the operation of the P1.2.

Bit 7: CLK Output

0 = The internally selected clock is not sent out onto P1.2 pin.

1 = This CLK Output is used to observe connected external crystal oscillator clock connected in CY7C601xx. When CLK Output is set, the internally selected clock is sent out onto P1.2 pin.

Note: Table 13-3 on page 26 is used to select the external or internal clock in enCoRe II devices

17.2.17 P1.3 Configuration (SSEL)

Table 17-9. P1.3 Configuration (P13CR) [0x10] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved	Int Enable	Int Act Low	Reserved	High Sink	Open Drain	Pull-up Enable	Output Enable
Read/Write	–	R/W	R/W	–	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

This register controls the operation of the P1.3 pin. This register exists in all enCoRe II LVparts.

The P1.3 GPIO's threshold is always set to TTL.

When the SPI hardware is enabled or disabled, the pin is controlled by the Output Enable bit and the corresponding bit in the P1 data register.

Regardless of whether the pin is used as an SPI or GPIO pin the Int Enable, Int act Low, high sink, open drain, and pull-up enable control the behavior of the pin.

17.2.18 P1.4–P1.6 Configuration (SCLK, SMOSI, SMISO)

Table 17-10. P1.4–P1.6 Configuration (P14CR–P16CR) [0x11–0x13] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	SPI Use	Int Enable	Int Act Low	Reserved	High Sink	Open Drain	Pull-up Enable	Output Enable
Read/Write	R/W	R/W	R/W	–	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

These registers control the operation of pins P1.4–P1.6, respectively. These registers exist in all enCoRe II LV parts.

Bit 7: SPI Use

0 = Disable the SPI alternate function. The pin is used as a GPIO

1 = Enable the SPI function. The SPI circuitry controls the output of the pin

The P1.4–P1.6 GPIO's threshold is always set to TTL.

When the SPI hardware is enabled, pins that are configured as SPI Use have their output enable and output state controlled by the SPI circuitry. When the SPI hardware is disabled or a pin has its SPI Use bit clear, the pin is controlled by the Output Enable bit and the corresponding bit in the P1 data register.

Regardless of whether any pin is used as an SPI or GPIO pin the Int Enable, Int act low, high sink, open drain, and pull-up enable control the behavior of the pin.

Note for Comm Modes 01 or 10 (SPI Master or SPI Slave, see Table 18-2 on page 46)

When configured for SPI (SPI Use = 1 and Comm Modes [1:0] = SPI master or SPI slave mode), the input and output direction of pins P1.5, and P1.6 is set automatically by the SPI logic. However, pin P1.4's input and output direction is not automatically set; it must be explicitly set by firmware. For SPI master mode, pin P1.4 must be configured as an output; for SPI slave mode, pin P1.4 must be configured as an input.

17.2.19 P1.7 Configuration

Table 17-11. P1.7 Configuration (P17CR) [0x14] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved	Int Enable	Int Act Low	Reserved	High Sink	Open Drain	Pull-up Enable	Output Enable
Read/Write	–	R/W	R/W	–	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

This register controls the operation of pin P1.7.

The 50 mA sink drive capability is only available in CY7C602xx. In CY7C601xx, only 8 mA sink drive capability is available on this pin regardless of the setting of the high sink bit.

The P1.7 GPIO's threshold is always set to TTL.

17.2.20 P2 Configuration

Table 17-12. P2 Configuration (P2CR) [0x15] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved	Int Enable	Int Act Low	TTL Thresh	High Sink	Open Drain	Pull-up Enable	Output Enable
Read/Write	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

In CY7C602xx, this register controls the operation of pins P2.0–P2.1. In CY7C601xx, this register controls the operation of pins P2.0–P2.7.

The 50-mA sink drive capability is only available on pin P2.7 and only on CY7C601xx. In CY7C602xx, only an 8-mA sink drive capability is available on this pin regardless of the setting of the high sink bit.

17.2.21 P3 Configuration

Table 17-13. P3 Configuration (P3CR) [0x16] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved	Int Enable	Int Act Low	TTL Thresh	High Sink	Open Drain	Pull-up Enable	Output Enable
Read/Write	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

In CY7C602xx, this register controls the operation of pins P3.0–P3.1. In CY7C601xx, this register controls the operation of pins P3.0–P3.7.

The 50-mA sink drive capability is only available on pin P3.7 and only on CY7C601xx. In CY7C602xx, only an 8-mA sink drive capability is available on this pin regardless of the setting of the high sink bit.

17.2.22 P4 Configuration

Table 17-14. P4 Configuration (P4CR) [0x17] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved	Int Enable	Int Act Low	TTL Thresh	High Sink	Open Drain	Pull-up Enable	Output Enable
Read/Write	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

This register exists only in CY7C601xx. This register controls the operation of pins P4.0–P4.3.

18. Serial Peripheral Interface (SPI)

The SPI master and slave interface core logic runs on the SPI clock domain. The SPI clock is a divider off of the CPUCLK when in the master mode. SPI is a four pin serial interface comprised of a clock, an enable, and two data pins.

Figure 18-1. SPI Block Diagram
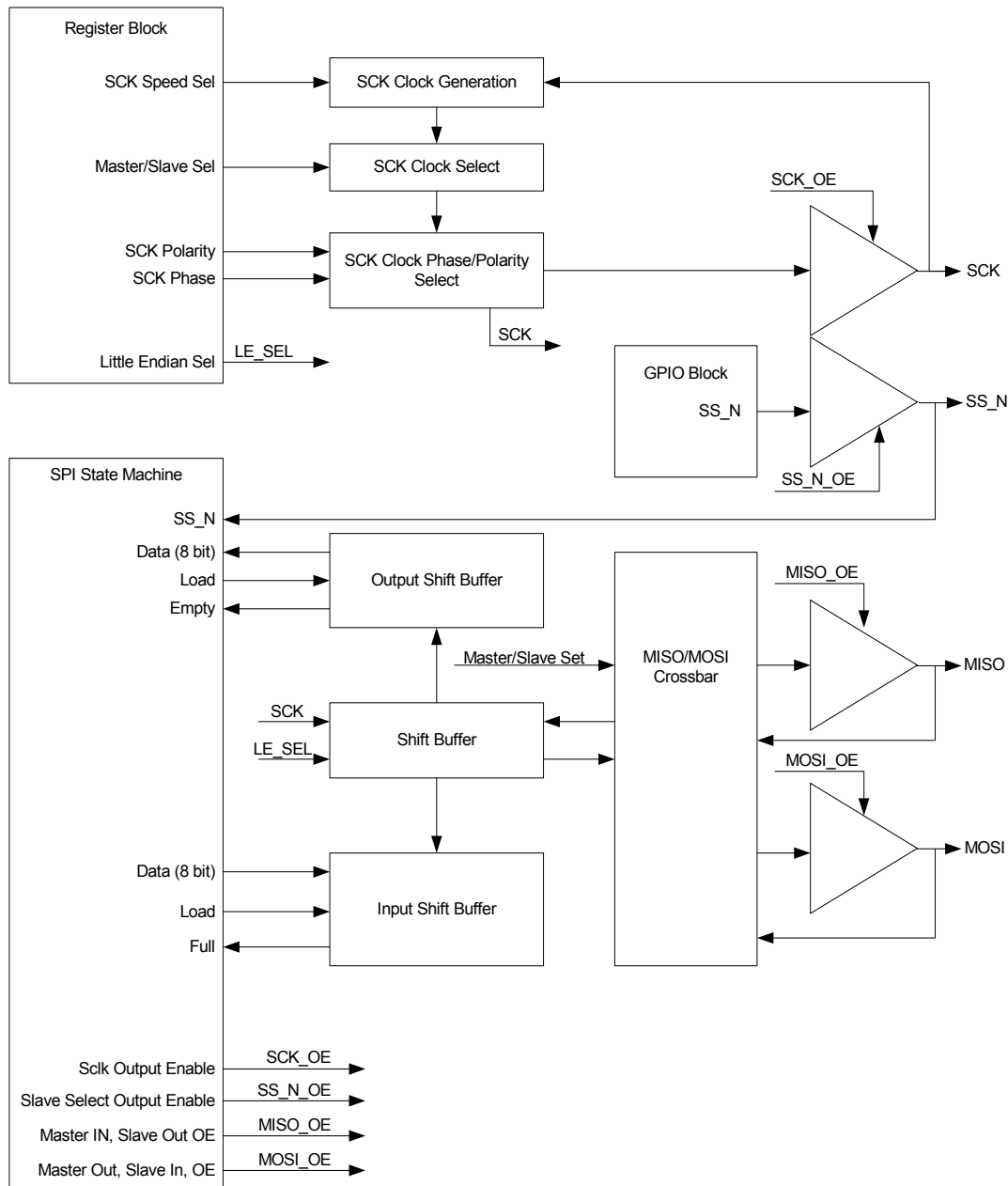
18.1 SPI Data Register

Table 18-1. SPI Data Register (SPIDATA) [0x3C] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	SPIData[7:0]							
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

When read, this register returns the contents of the receive buffer. When written, it loads the transmit holding register.

Bit [7:0]: SPI Data [7:0]

When an interrupt occurs to indicate to firmware that a byte of receive data is available or the transmitter holding register is empty, firmware has seven SPI clocks to manage the buffers—to empty the receiver buffer or to refill the transmit holding register. Failure to meet this timing requirement results in incorrect data transfer.

18.2 SPI Configure Register

Table 18-2. SPI Configure Register (SPICR) [0x3D] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Swap	LSB First	Comm Mode		CPOL	CPHA	SCLK Select	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

Bit 7: Swap

0 = Swap function disabled

1 = The SPI block swaps its use of SMOSI and SMISO. Among other things, this is useful to implement single wire communications similar to SPI.

Bit 6: LSB first

0 = The SPI transmits and receives the MSB (Most Significant Bit) first.

1 = The SPI transmits and receives the LSB (Least Significant Bit) first.

Bit [5:4]: Comm mode [1:0]

0 0: All SPI communication disabled

0 1: SPI master mode

1 0: SPI slave mode

1 1: Reserved

Bit 3: CPOL

This bit controls the SPI clock (SCLK) idle polarity.

0 = SCLK idles low

1 = SCLK idles high

Bit 2: CPHA

The Clock Phase bit controls the phase of the clock on which data is sampled. Table 18-3 on page 47 shows the timing for various combinations of LSB First, CPOL, and CPHA.

Bit [1:0]: SCLK Select

This field selects the speed of the master SCLK. When in master mode, SCLK is generated by dividing the base CPUCLK

Important Note for Comm Modes 01b or 10b (SPI Master or SPI Slave)
When configured for SPI, (SPI Use = 1 – Table 17-10 on page 43), the input and output direction of pins P1.3, P1.5, and P1.6 is set automatically by the SPI logic. However, pin P1.4's input and output direction is NOT automatically set; it must be explicitly set by firmware. For SPI Master mode, pin P1.4 is configured as an output; for SPI Slave mode, pin P1.4 is configured as an input.

Table 18-3. SPI Mode Timing vs. LSB First, CPOL, and CPHA
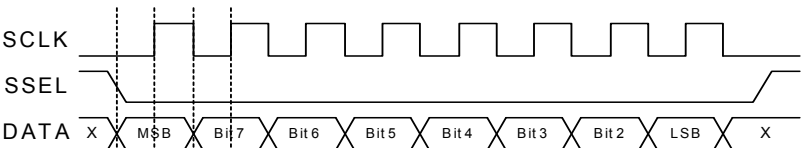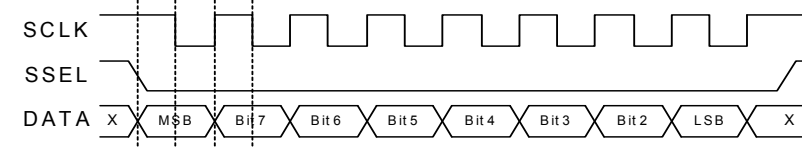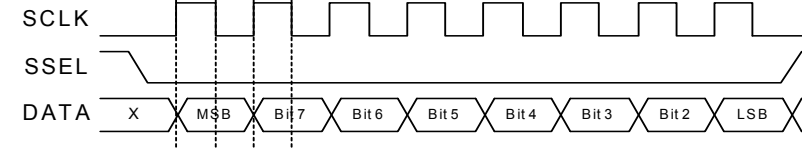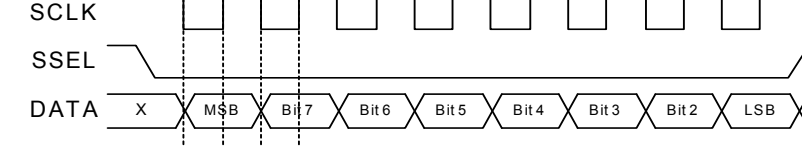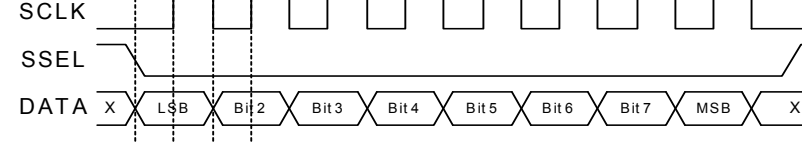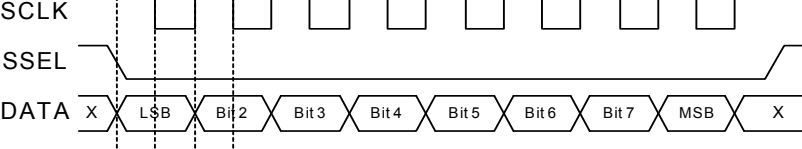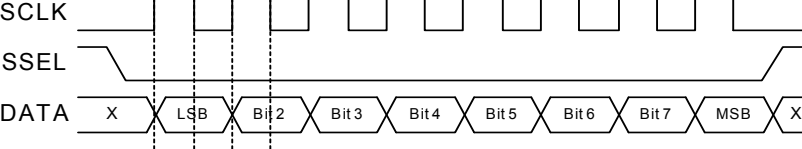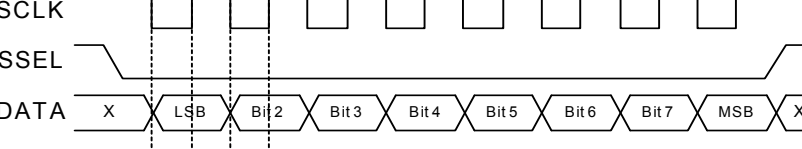
LSB First	CPHA	CPOL	Diagram
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Table 18-4. SPI SCLK Frequency

SCLK Select	CPUCLK Divisor	SCLK Frequency when CPUCLK = 12 MHz
00	6	2 MHz
01	12	1 MHz
10	48	250 kHz
11	96	125 kHz

18.3 SPI Interface Pins

The SPI interface uses the P1.3–P1.6 pins. These pins are configured using the P1.3 and P1.4–P1.6 configuration.

19. Timer Registers

All timer functions of the enCoRe II LV are provided by a single-timer block. The timer block is asynchronous from the CPU clock. The 16-bit free-running counter is used as the time base for timer captures and also as a general time base by software.

19.1 Registers

19.1.1 Free-Running Counter

The 16-bit free-running counter is clocked by the timer capture clock (TCAPCLK). It is read in software for use as a general-purpose time base. When reading the low-order byte, the high-order byte is registered. Reading the high-order byte reads this register allowing the CPU to read the 16-bit value atomically (loads all bits at one time). The free-running timer generates an interrupt at 1024 μs rate when clocked by a 4-MHz source. It also generates an interrupt when the free-running counter overflow occurs – every 16.384 ms (with a 4-MHz source). This extends the length of the timer.

Figure 19-1. 16-Bit Free-Running Counter Block Diagram
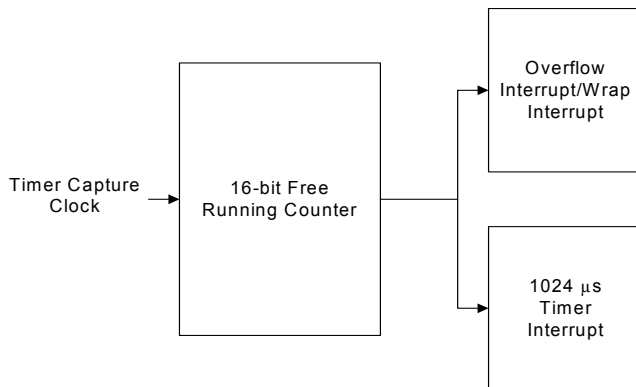
Table 19-1. Free-Running Timer Low-Order Byte (FRTMRL) [0x20] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field				Free-running Timer [7:0]				
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

Bit [7:0]: Free-running timer [7:0]

This register holds the low-order byte of the 16-bit free-running timer. Reading this register moves the high-order byte into a holding register allowing an automatic read of all 16 bits simultaneously.

For reads, the actual read occurs in the cycle when the low-order is read. For writes, the actual time the write occurs is the cycle when the high-order is written.

When reading the free-running timer, the low-order byte is read first and the high-order second. When writing, the low-order byte is written first then the high-order byte.

Table 19-2. Free-Running Timer High-Order Byte (FRTMRH) [0x21] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Free-running timer [15:8]							
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

Bit [7:0]: Free-running timer [15:8]
When reading the free-running timer, the low-order byte is read first and the high-order second. When writing, the low-order byte is written first, then the high-order byte.

19.1.2 Time Capture

enCoRe II LV has two 8-bit captures. Each capture has a separate register for rising and falling time. The two 8-bit captures can be configured as a single 16-bit capture. When configured in this way, the capture 1 registers hold the high-order byte of the 16-bit timer capture value. Each of the four capture registers can be programmed to generate an interrupt when it is loaded.

Figure 19-2. Time Capture Block Diagram
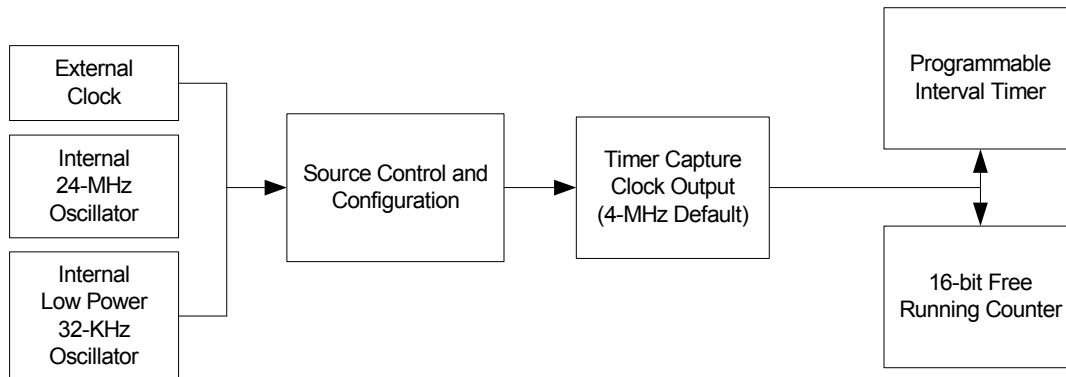
Table 19-1. Timer Configuration (TMRCR) [0x2A] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	First Edge Hold	8-bit Capture Prescale [2:0]			Cap0 16-bit Enable	Reserved		
Read/Write	R/W	R/W	R/W	R/W	R/W	–	–	–
Default	0	0	0	0	0	0	0	0

Bit 7: First edge hold
The first edge hold function applies to all four capture timers.
0 = The time of the most recent edge is held in the capture timer data register. If multiple edges have occurred since reading the capture timer, the time for the most recent one is read.
1 = The time of the first occurrence of an edge is held in the capture timer data register until the data is read. Subsequent edges are ignored until the capture timer data register is read.
Bit [6:4]: 8-bit capture prescale [2:0]
This field controls which eight bits of the 16 free-running timer are captured when in bit mode.
0 0 0 = capture timer[7:0]
0 0 1 = capture timer[8:1]
0 1 0 = capture timer[9:2]
0 1 1 = capture timer[10:3]
1 0 0 = capture timer[11:4]
1 0 1 = capture timer[12:5]
1 1 0 = capture timer[13:6]
1 1 1 = capture timer[14:7]
Bit 3: Cap0 16-bit Enable
0 = Capture 0 16-bit mode is disabled
1 = Capture 0 16-bit mode is enabled. Capture 1 is disabled and the Capture 1 rising and falling registers are used as an extension to the Capture 0 registers—extending them to 16 bits.
Bit [2:0]: Reserved

Table 19-2. Capture Interrupt Enable (TCAPINTE) [0x2B] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field		Rese	rved		Cap1 Fall Enable	Cap1 Rise Enable	Cap0 Fall Enable	Cap0 Rise Enable
Read/Write	–	–	–	–	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

Bit [7:4]: Reserved

Bit 3: Cap1 fall enable

0 = Disable the capture 1 falling edge interrupt

1 = Enable the capture 1 falling edge interrupt

Bit 2: Cap1 rise enable

0 = Disable the capture 1 rising edge interrupt

1 = Enable the capture 1 rising edge interrupt

Bit 1: Cap0 fall enable

0 = Disable the capture 0 falling edge interrupt

1 = Enable the capture 0 falling edge interrupt

Bit 0: Cap0 rise enable

0 = Disable the capture 0 rising edge interrupt

1 = Enable the capture 0 rising edge interrupt

Table 19-3. Timer Capture 0 Rising (TCAP0R) [0x22] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field				Capture 0	Rising [7:0]			
Read/Write	R	R	R	R	R	R	R	R
Default	0	0	0	0	0	0	0	0

Bit [7:0]: Capture 0 Rising [7:0]

This register holds the value of the free-running timer when the last rising edge occurred on the TIO0 input. When Capture 0 is in 8-bit mode, the bits that are stored here are selected by the Prescale [2:0] bits in the Timer Configuration register. When Capture 0 is in 16-bit mode this register holds the lower order eight bits of the 16-bit timer.

Table 19-4. Timer Capture 1 Rising (TCAP1R) [0x23] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field				Capture 1	Rising [7:0]			
Read/Write	R	R	R	R	R	R	R	R
Default	0	0	0	0	0	0	0	0

Bit [7:0]: Capture 1 Rising [7:0]

This register holds the value of the free-running timer when the last rising edge occurred on the TIO1 input. The bits that are stored here are selected by the Prescale [2:0] bits in the Timer Configuration register. When Capture 0 is in 16-bit mode this register holds the high-order eight bits of the 16-bit timer from the last TIO0 rising edge.

Table 19-5. Timer Capture 0 Falling (TCAP0F) [0x24] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field				Capture 0	Falling [7:0]			
Read/Write	R	R	R	R	R	R	R	R
Default	0	0	0	0	0	0	0	0

Bit [7:0]: Capture 0 Falling [7:0]

This register holds the value of the free-running timer when the last falling edge occurred on the TIO0 input. When Capture 0 is in 8-bit mode, the bits that are stored here are selected by the Prescale [2:0] bits in the Timer Configuration register. When Capture 0 is in 16-bit mode this register holds the lower order eight bits of the 16-bit timer.

Table 19-6. Timer Capture 1 Falling (TCAP1F) [0x25] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Capture 1 Falling [7:0]							
Read/Write	R	R	R	R	R	R	R	R
Default	0	0	0	0	0	0	0	0

Bit [7:0]: Capture 1 Falling [7:0]

This register holds the value of the free-running timer when the last falling edge occurred on the TIO1 input. The bits stored here are selected by the Prescale [2:0] bits in the Timer Configuration register. When capture 0 is in 16-bit mode this register holds the high-order eight bits of the 16-bit timer from the last TIO0 falling edge.

Table 19-7. Capture Interrupt Status (TCAPINTS) [0x2C] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved				Cap1 Fall Active	Cap1 Rise Active	Cap0 Fall Active	Cap0 Rise Active
Read/Write	–	–	–	–	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

These four bits contains the status bits for the four timer captures for the four timer block capture interrupt sources. Writing any of these bits with 1 clears that interrupt.

Bit [7:4]: Reserved

Bit 3: Cap1 fall active

0 = No event

1 = A falling edge has occurred on TIO1

Bit 2: Cap1 rise active

0 = No event

1 = A rising edge has occurred on TIO1

Bit 1: Cap0 Fall Active

0 = No event

1 = A falling edge has occurred on TIO0

Bit 0: Cap0 Rise Active

0 = No event

1 = A rising edge has occurred on TIO0

Note The interrupt status bits are cleared by firmware to enable subsequent interrupts. This is achieved by writing a '1' to the corresponding Interrupt status bit.

19.1.3 Programmable Interval Timer

Table 19-8. Programmable Interval Timer Low (PITMRL) [0x26] [R]

Bit #	7	6	5	4	3	2	1	0
Field	Prog Interval Timer [7:0]							
Read/Write	R	R	R	R	R	R	R	R
Default	0	0	0	0	0	0	0	0

Bit [7:0]: Prog Interval Timer [7:0]

This register holds the low-order byte of the 12-bit programmable interval timer. Reading this register moves the high-order byte into a holding register allowing an automatic read of all 12 bits simultaneously.

Table 19-9. Programmable Interval Timer High (PITMRH) [0x27] [R]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved				Prog Interval Timer [11:8]			
Read/Write	--	--	--	--	R	R	R	R
Default	0	0	0	0	0	0	0	0

Bit [7:4]: Reserved

Bit [3:0]: Prog Internal Timer [11:8]

This register holds the high-order nibble of the 12-bit programmable interval timer. Reading this register returns the high-order nibble of the 12-bit timer at the instant when the low-order byte was last read.

Table 19-10. Programmable Interval Reload Low (PIRL) [0x28] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Prog Interval [7:0]							
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

Bit [7:0]: Prog Interval [7:0]

This register holds the lower eight bits of the timer. When writing into the 12-bit reload register, write lower byte first then the higher nibble.

Table 19-11. Programmable Interval Reload High (PIRH) [0x29] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved				Prog Interval[11:8]			
Read/Write	--	--	--	--	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

Bit [7:4]: Reserved

Bit [3:0]: Prog Interval [11:8]

This register holds the higher 4 bits of the timer. When writing into the 12-bit reload register, write lower byte first then the higher nibble.

Figure 19-3. Timer Functional Sequence Diagram
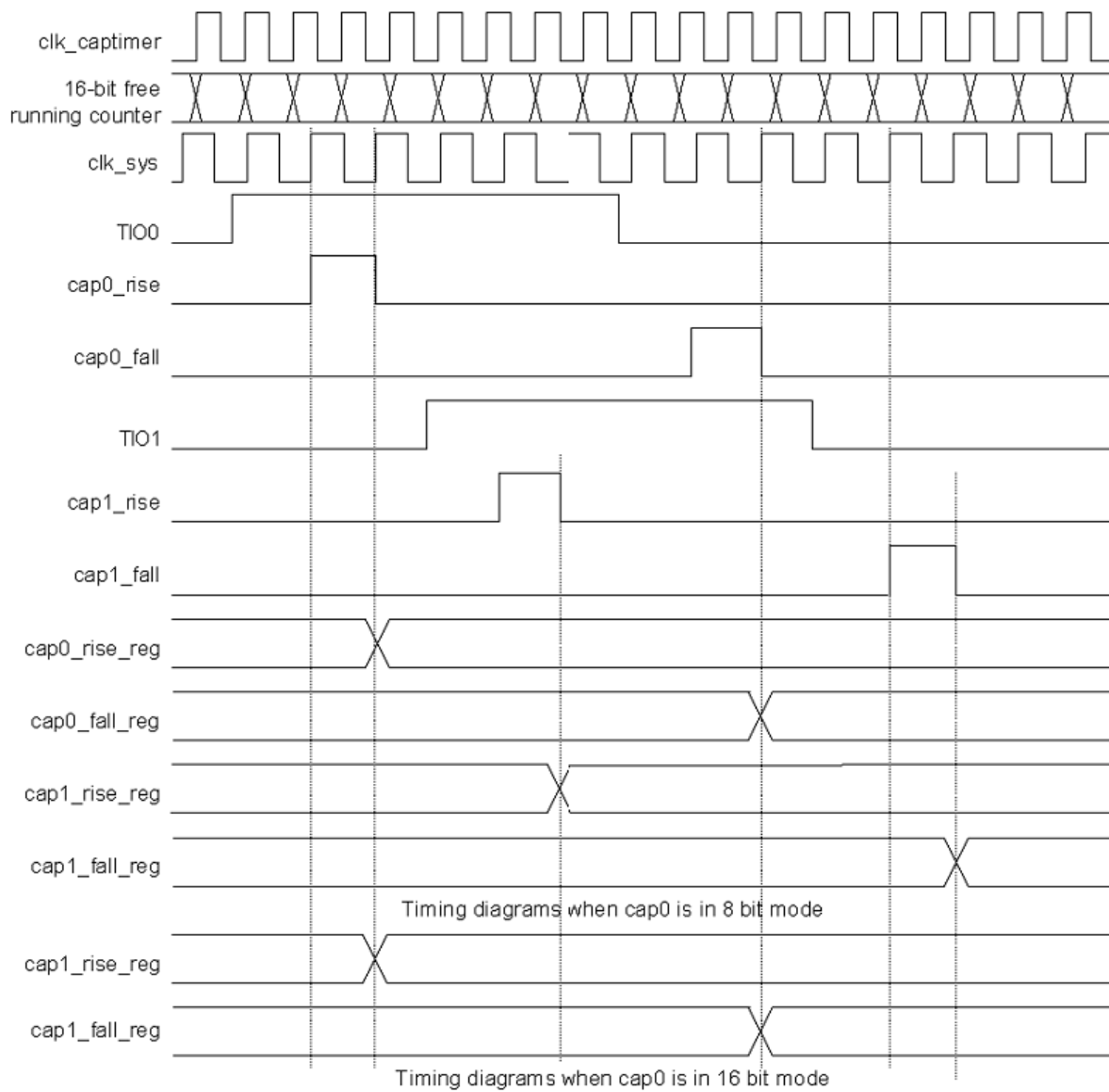
Figure 19-4. 16-Bit Free-Running Counter Loading Timing Diagram

16-bit free running counter loading timing
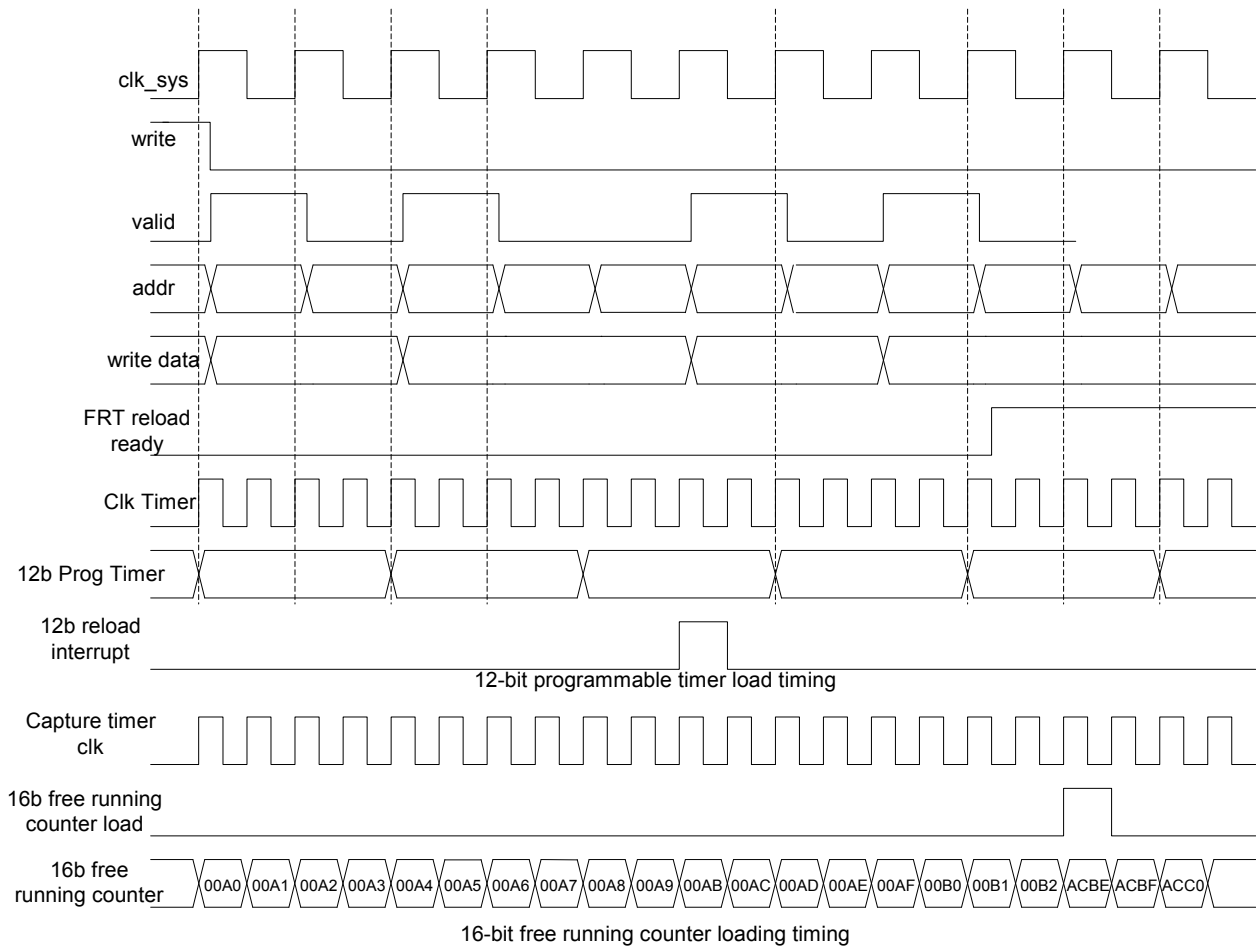
Figure 19-5. Memory Mapped Registers Read and Write Timing Diagram

Memory mapped registers Read/Write timing diagram
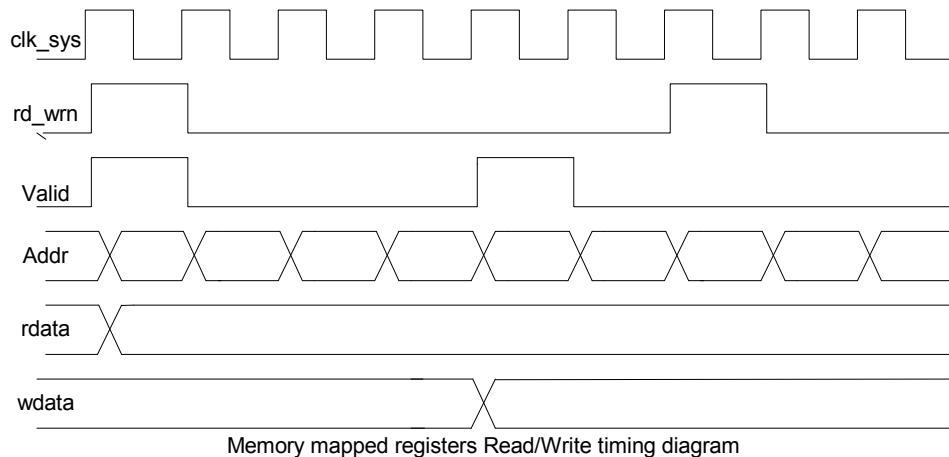
20. Interrupt Controller

The interrupt controller and its associated registers allow the user's code to respond to an interrupt from almost every functional block in the enCoRe II LV devices. The registers associated with the interrupt controller are disabled either globally or individually. The registers also provide a mechanism for users to clear all pending and posted interrupts or clear individual posted or pending interrupts.

Table 20-1 lists all interrupts and the priorities that are available in the enCoRe II LV devices.

Table 20-1. Interrupt Priorities, Address, and Name

Interrupt Priority	Interrupt Address	Name
0	0000h	Reset
1	0004h	POR/LVD
2	0008h	INT0
3	000Ch	SPI transmitter empty
4	0010h	SPI receiver full
5	0014h	GPIO Port 0
6	0018h	GPIO Port 1
7	001Ch	INT1
8	0020h	Reserved
9	0024h	Reserved
10	0028h	Reserved
11	002Ch	Reserved
12	0030h	Reserved
13	0034h	1 mS interval timer
14	0038h	Programmable interval timer
15	003Ch	Timer Capture 0
16	0040h	Timer Capture 1
17	0044h	16-bit free-running timer wrap
18	0048h	INT2
19	004Ch	Reserved
20	0050h	GPIO Port 2
21	0054h	GPIO Port 3
22	0058h	GPIO Port 4
23	005Ch	Reserved
24	0060h	Reserved
25	0064h	Sleep timer

20.1 Architectural Description

An interrupt is posted when its interrupt conditions occur. This results in the flip-flop in Figure 20-1 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.

Nested interrupts are accomplished by reenabling interrupts inside an interrupt service routine. To do this, set the IE bit in the flag register. A block diagram of the enCoRe II LV interrupt controller is shown in Figure 20-1.

Figure 20-1. Interrupt Controller Block Diagram
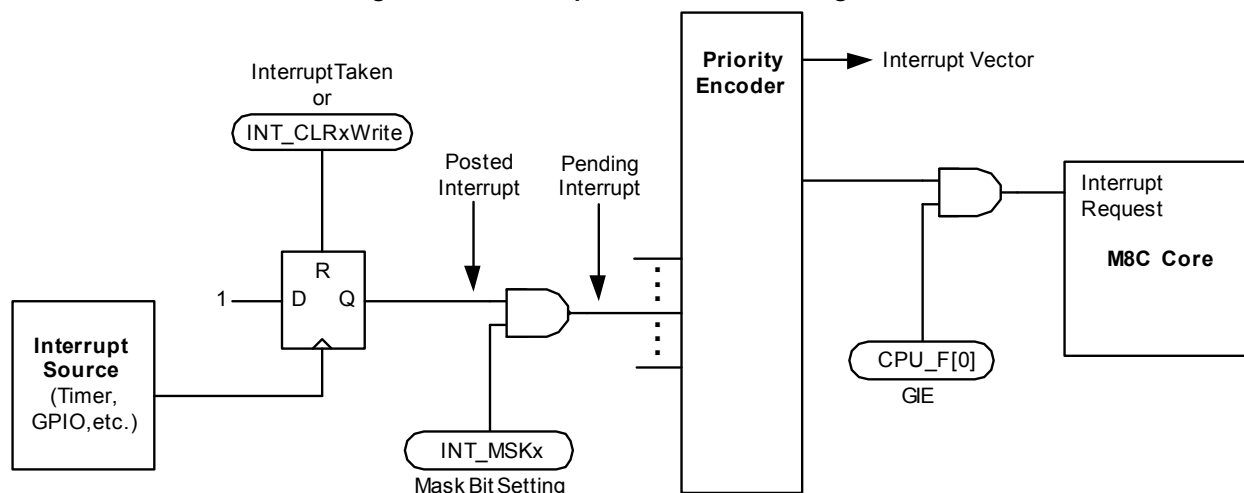
20.2 Interrupt Processing

The sequence of events that occur during interrupt processing is as follows:

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.
   c. An interrupt is pending and GIE is set from 0 to 1 in the CPU Flag register.
1. The current executing instruction finishes.
2. The internal interrupt is dispatched, taking 13 cycles. During this time, the following actions occur:
   a. The MSB and LSB of program counter and flag registers (CPU_PC and CPU_F) are stored onto the program stack by an automatic CALL instruction (13 cycles) generated during the interrupt acknowledge process.
   b. The PCH, PCL, and flag register (CPU_F) are stored onto the program stack (in that order) by an automatic CALL instruction (13 cycles) generated during the interrupt acknowledge process.
   c. The CPU_F register is then cleared. Because this clears the GIE bit to 0, additional interrupts are temporarily disabled.
   d. The PCH (PC[15:8]) is cleared to zero.
   e. The interrupt vector is read from the interrupt controller and its value placed into PCL (PC[7:0]). This sets the program counter to point to the appropriate address in the interrupt table (for example, 0004h for the POR and LVD interrupt).

1. 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.
2. The ISR executes. Note that interrupts are disabled because GIE =0. In the ISR, interrupts are re-enabled if desired, by setting GIE = 1 (avoid stack overflow).
3. The ISR ends with a RETI instruction, which restores the program counter and flag registers (CPU_PC and CPU_F). The restored flag register re-enables interrupts, because GIE = 1 again.
4. 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.

20.3 Interrupt Latency

The time between the assertion of an enabled interrupt and the start of its ISR is calculated from the following equation.

Latency = Time for current instruction to finish + time for internal interrupt routine to execute + time for LJMP instruction in interrupt table to execute.

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 is as follows:

(1 to 5 cycles for JMP to finish) + (13 cycles for interrupt routine) + (7 cycles for LJMP) = 21 to 25 cycles.

In the example above, at 12 MHz, 25 clock cycles take 2.08 µs.

20.4 Interrupt Registers

20.4.1 Interrupt Clear Register

The interrupt clear registers (INT_CLRx) are used to enable the individual interrupt sources' ability to clear posted interrupts.

When an INT_CLRx register is read, any bits that are set indicates an interrupt has been posted for that hardware resource. Therefore, reading these registers enables the user to determine all posted interrupts.

Table 20-1. Interrupt Clear 0 (INT_CLR0) [0xDA] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	GPIO Port 1	Sleep timer	INT1	GPIO Port 0	SPI receive	SPI Transmit	INT0	POR/LVD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

When reading this register,

0 = There is no posted interrupt for the corresponding hardware.

1 = There is a posted interrupt for the corresponding hardware.

Writing a '0' to the bits clears the posted interrupts for the corresponding hardware. Writing a '1' to the bits and to the ENSWINT (Bit 7 of the INT_MSK3 Register) posts the corresponding hardware interrupt.
The GPIO interrupts are edge-triggered.

Table 20-2. Interrupt Clear 1 (INT_CLR1) [0xDB] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	TCAP0	Prog Interval Timer	1-ms Program-mable Interrupt	Reserved				
Read/Write	R/W	R/W	R/W	–	–	–	–	–
Default	0	0	0	0	0	0	0	0

When reading this register,

0 = There is no posted interrupt for the corresponding hardware.

1 = There is a posted interrupt for the corresponding hardware.

Writing a '0' to the bits clears the posted interrupts for the corresponding hardware. Writing a '1' to the bits AND to the ENSWINT (Bit 7 of the INT_MSK3 Register) posts the corresponding hardware interrupt.

Table 20-3. Interrupt Clear 2 (INT_CLR2) [0xDC] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved	GPIO Port 4	GPIO Port 3	GPIO Port 2	Reserved	INT2	16-bit Counter Wrap	TCAP1
Read/Write	–	R/W	R/W	R/W	–	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

When reading this register,

0 = There is no posted interrupt for the corresponding hardware.

1 = There is a posted interrupt for the corresponding hardware.

Writing a '0' to the bits clears the posted interrupts for the corresponding hardware. Writing a '1' to the bits AND to the ENSWINT (Bit 7 of the INT_MSK3 Register) posts the corresponding hardware interrupt.

20.4.2 Interrupt Mask Registers

The interrupt mask registers (INT_MSKx) enable the individual interrupt sources' ability to create pending interrupts.

There are four interrupt mask registers (INT_MSK0, INT_MSK1, INT_MSK2, and INT_MSK3) which are referred to in general as INT_MSKx. If cleared, each bit in an INT_MSKx 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_MSKx bits are independent of all other INT_MSKx bits.

If an INT_MSKx bit is set, the interrupt source associated with that mask bit generates an interrupt that becomes a pending interrupt.

The enable software interrupt (ENSWINT) bit in INT_MSK3[7] determines the way an individual bit value written to an INT_CLRx register is interpreted. When cleared, writing 1s to an INT_CLRx register has no effect. However, writing 0s to an INT_CLRx register, when ENSWINT is cleared, causes the corresponding interrupt to clear. If the ENSWINT bit is set, 0s written to the INT_CLRx registers are ignored. However, 1s written to an INT_CLRx register, when ENSWINT is set, causes an interrupt to post for the corresponding interrupt.

Software interrupts 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.

Table 20-4. Interrupt Mask 3 (INT_MSK3) [0xDE] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	ENSWINT	Reserved						
Read/Write	R	–	–	–	–	–	–	–
Default	0	0	0	0	0	0	0	0

Bit 7: Enable Software Interrupt (ENSWINT)

0= Disable. Writing 0s to an INT_CLRx register, when ENSWINT is cleared, clears the corresponding interrupt.

1= Enable. Writing 1s to an INT_CLRx register, when ENSWINT is set, posts the corresponding interrupt.

Bit [6:0]: Reserved

Table 20-5. Interrupt Mask 2 (INT_MSK2) [0xDF] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Reserved	GPIO Port 4 Int Enable	GPIO Port 3 Int Enable	GPIO Port 2 Int Enable	Reserved	INT2 Int Enable	16-bit Counter Wrap Int Enable	TCAP1 Int Enable
Read/Write	–	R/W	R/W	R/W	–	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

Bit 7: Reserved

Bit 6: GPIO Port 4 Interrupt Enable

0 = Mask GPIO Port 4 interrupt

1 = Unmask GPIO Port 4 interrupt

Bit 5: GPIO Port 3 Interrupt Enable

0 = Mask GPIO Port 3 interrupt

1 = Unmask GPIO Port 3 interrupt

Bit 4: GPIO Port 2 Interrupt Enable

0 = Mask GPIO Port 2 interrupt

1 = Unmask GPIO Port 2 interrupt

Bit 3: Reserved

Bit 2: INT2 Interrupt Enable

0 = Mask INT2 interrupt

1 = Unmask INT2 interrupt

Bit 1: 16-bit Counter Wrap Interrupt Enable

0 = Mask 16-bit counter wrap interrupt

1 = Unmask 16-bit counter wrap interrupt

Bit 0: TCAP1 Interrupt Enable

0 = Mask TCAP1 interrupt

1 = Unmask TCAP1 interrupt

The GPIO interrupts are edge-triggered.

Table 20-6. Interrupt Mask 1 (INT_MSK1) [0xE1] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	TCAP0 Int Enable	Prog Interval Timer Int Enable	1-ms Timer Int Enable	Reserved				
Read/Write	R/W	R/W	R/W	–	–	–	–	–
Default	0	0	0	0	0	0	0	0

Bit 7: TCAP0 Interrupt Enable
0 = Mask TCAP0 interrupt
1 = Unmask TCAP0 interrupt
Bit 6: Prog Interval Timer Interrupt Enable
0 = Mask prog interval timer interrupt
1 = Unmask prog interval timer interrupt
Bit 5: 1 ms Timer Interrupt Enable
0 = Mask 1 ms interrupt
1 = Unmask 1 ms interrupt
Bit [4:0]: Reserved

Table 20-7. Interrupt Mask 0 (INT_MSK0) [0xE0] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	GPIO Port 1 Int Enable	Sleep Timer Int Enable	INT1 Int Enable	GPIO Port 0 Int Enable	SPI Receive Int Enable	SPI Transmit Int Enable	INT0 Int Enable	POR/LVD Int Enable
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

Bit 7: GPIO Port 1 Interrupt Enable
0 = Mask GPIO Port 1 interrupt
1 = Unmask GPIO Port 1 interrupt
Bit 6: Sleep Timer Interrupt Enable
0 = Mask sleep timer interrupt
1 = Unmask sleep timer interrupt
Bit 5: INT1 Interrupt Enable
0 = Mask INT1 interrupt
1 = Unmask INT1 interrupt
Bit 4: GPIO Port 0 Interrupt Enable
0 = Mask GPIO Port 0 interrupt
1 = Unmask GPIO Port 0 interrupt
Bit 3: SPI Receive Interrupt Enable
0 = Mask SPI receive interrupt
1 = Unmask SPI receive interrupt
Bit 2: SPI Transmit Enable
0 = Mask SPI transmit interrupt
1 = Unmask SPI transmit interrupt
Bit 1: INT0 Interrupt Enable
0 = Mask INT0 interrupt
1 = Unmask INT0 interrupt
Bit 0: POR/LVD Interrupt Enable
0 = Mask POR/LVD interrupt
1 = Unmask POR/LVD interrupt

20.4.3 Interrupt Vector Clear Register

Table 20-8. Interrupt Vector Clear Register (INT_VC) [0xE2] [R/W]

Bit #	7	6	5	4	3	2	1	0
Field	Pending Interrupt [7:0]							
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

The Interrupt Vector Clear Register (INT_VC) holds the interrupt vector for the highest priority pending interrupt when read, and when written clears all pending interrupts.
Bit [7:0]: Pending Interrupt [7:0]
8-bit data value holds the interrupt vector for the highest priority pending interrupt. Writing to this register clears all pending interrupts.

21. Absolute Maximum Ratings

Storage temperature .... –40 °C to +90 °C

Ambient temperature with power applied .... –0 °C to +70 °C

Supply voltage on V_{CC} relative to V_{SS}..........–0.5 V to +7.0 V

DC input voltage .... –0.5 V to + V_{CC} + 0.5 V

DC voltage applied to outputs in
high-Z state.................................... –0.5 V to + V_{CC} + 0.5 V

Maximum total sink output current into Port 0
and 1 and pins .... 70 mA

Maximum total source output current
into GPIO pins .... 30 mA

Maximum on-chip power dissipation
on any GPIO pin .... 50 mW

Power dissipation .... 300 mW

Static discharge voltage .... 2200 V

Latch-up current .... 200 mA

21.1 DC Characteristics

Parameter	Description / General	Conditions	Min	Typical	Max	Unit
V_{CC1}	Operating voltage	CPU speed <= 12 MHz	2.7		3.6	V
T_{FP}	Operating temperature	Flash programming	0		70	°C
I_{CC1}	V_{CC} operating supply current	CPU =12 MHz, V_{DD} = 3.3 V, T = 75 °C		4.25	11	mA
		CPU =12 MHz, V_{DD} = 2.7 V, T = 25 °C		3.25	-	mA
I_{CC2}	V_{CC} operating supply current	CPU = 6 MHz, V_{DD} = 3.3 V, T = 75 °C		3.15	9	mA
		CPU = 6 MHz, V_{DD} = 3.3 V, T = 25 °C		2.45	-	mA
I_{CC3}	V_{CC} operating supply current	CPU = 3 MHz, V_{DD} = 2.7 V, T = 25 °C		2.0	-	mA
I_{SB1}	Standby current	Internal and external oscillators, bandgap, flash, CPU clock, timer clock all disabled			10	μA
Low-voltage detect						
V_{LVD}	Low-voltage detect trip voltage	LVDCR [2:0] set to 000	2.681		2.7	V
General-purpose I/O interface						
R_{UP}	Pull-up resistance		4		12	KΩ
V_{ICR}	Input threshold voltage low, CMOS mode	Low to high edge	40%		65%	V_{CC}
V_{ICF}	Input threshold voltage low, CMOS mode	High to low edge	30%		55%	V_{CC}
V_{HC}	Input hysteresis voltage, CMOS mode	High to low edge	3%		10%	V_{CC}
V_{ILTTL}	Input low-voltage, TTL mode				0.72	V
V_{IHTTL}	Input HIGH voltage, TTL mode		1.6			V
V_{OL1}	Output low-voltage, high drive[4]	I_{OL1} = 50 mA			1.4	V
V_{OL2}	Output low-voltage, high drive[4]	I_{OL1} = 25 mA			0.4	V
V_{OL3}	Output low-voltage, low drive	I_{OL2} = 8 mA			0.8	V
V_{OH}	Output high voltage[4]	I_{OH} = 2 mA	V_{CC} – 0.5			V

Note
4. Available only on CY7C601xx P2.7, P3.7, P0.0, P0.1; CY7C602xx P1.3, P1.4, P1.5, P1.6, P1.7.

21.2 AC Characteristics

Parameter	Description	Conditions	Min	Typical	Max	Unit
Clock						
T_{ECLKDC}	External clock duty cycle		45		55	%
T_{ECLK2}	External clock frequency		1		24	MHz
F_{IMO}	Internal main oscillator (IMO) frequency	With proper trim values loaded[5]	18.72		26.4	MHz
F_{ILO}	Internal low-power oscillator (ILO)	With proper trim values loaded[5]	15.0001		50.0	KHz
GPIO Timing						
T_{R_GPIO}	Output rise time	Measured between 10 and 90% V_{DD} and Vreg with 50 pF load			50	ns
T_{F_GPIO}	Output fall time	Measured between 10 and 90% V_{DD} and Vreg with 50 pF load			15	ns
SPI Timing						
T_{SMCK}	SPI master clock rate	$F_{CPUCLK}/6$			2	MHz
T_{SSCK}	SPI slave clock rate				2.2	MHz
T_{SCKH}	SPI clock high time	High for CPOL = 0, Low for CPOL = 1	125			ns
T_{SCKL}	SPI clock low time	Low for CPOL = 0, High for CPOL = 1	125			ns
T_{MDO}	Master data output time[6]	SCK to data valid	−25		50	ns
T_{MDO1}	Master data output time, First bit with CPHA = 0	Time before leading SCK edge	100			ns
T_{MSU}	Master input data setup time		50			ns
T_{MHD}	Master input data hold time		50			ns
T_{SSU}	Slave input data setup time		50			ns
T_{SHD}	Slave input data hold time		50			ns
T_{SDO}	Slave data output time	SCK to data valid			100	ns
T_{SDO1}	Slave data output time, First bit with CPHA = 0	Time after $\overline{SS}$ LOW to data valid			100	ns
T_{SSS}	Slave select setup time	Before first SCK edge	150			ns
T_{SSH}	Slave select hold time	After last SCK edge	150			ns

Figure 21-1. Clock Timing
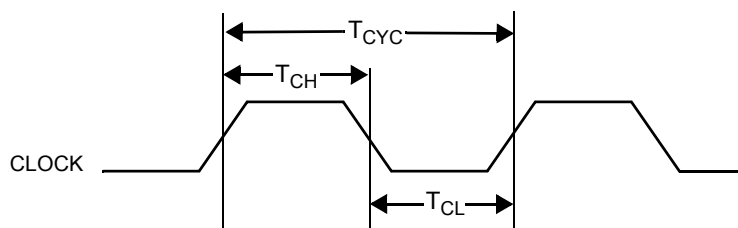
Notes
5. Refer to Clocking on page 22 for details on loading proper trim values.
6. In Master mode, first bit is available 0.5 SPICLK cycle before Master clock edge is available on the SCLK pin.

Figure 21-2. GPIO Timing Diagram
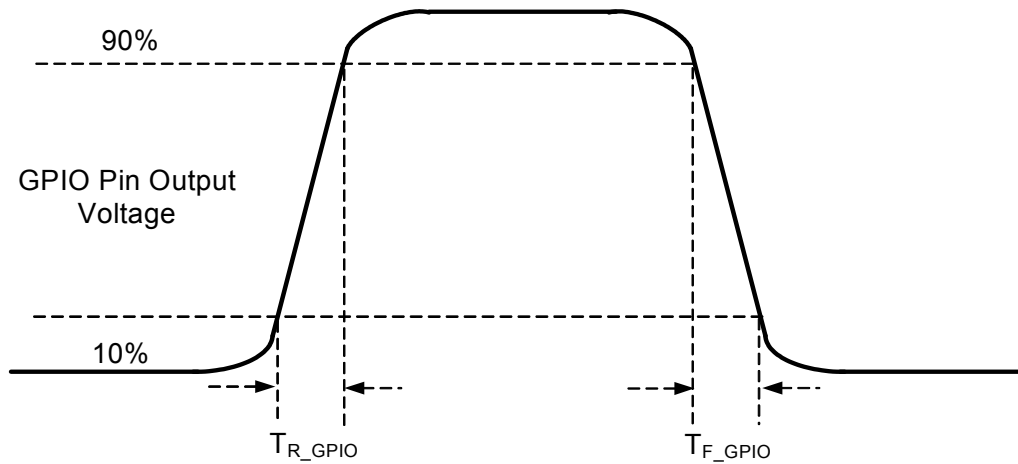
Figure 21-3. SPI Master Timing, CPHA = 1
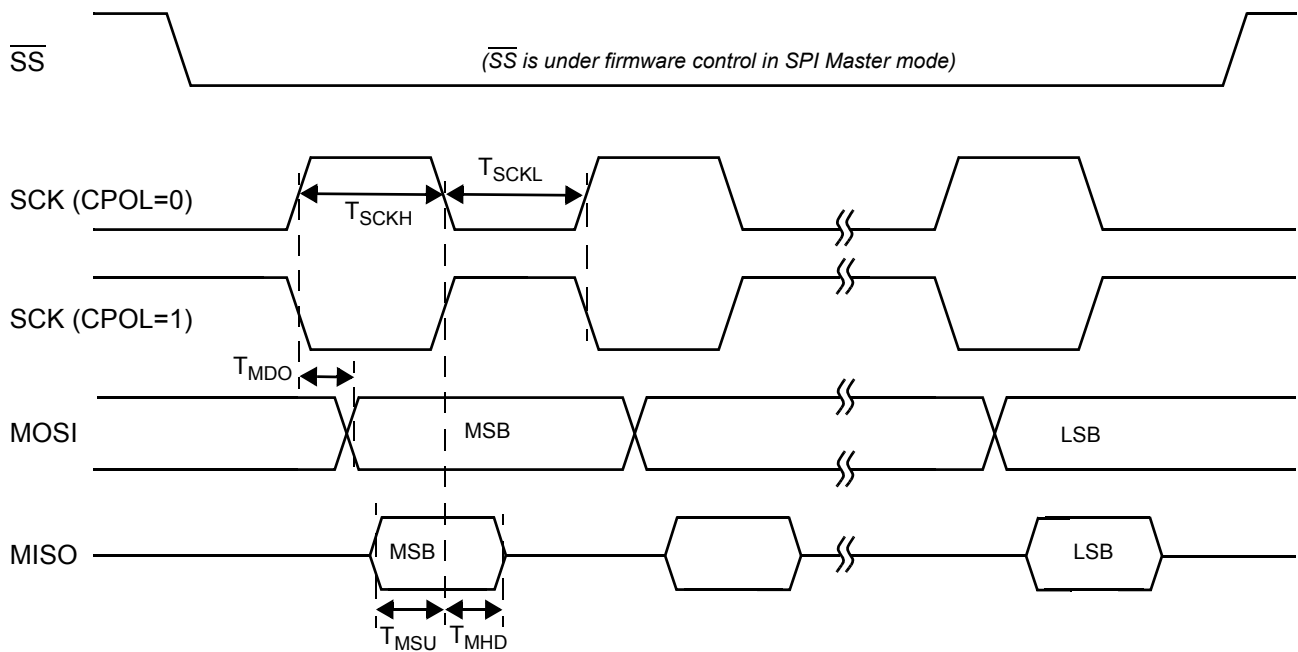
Figure 21-4. SPI Slave Timing, CPHA = 1
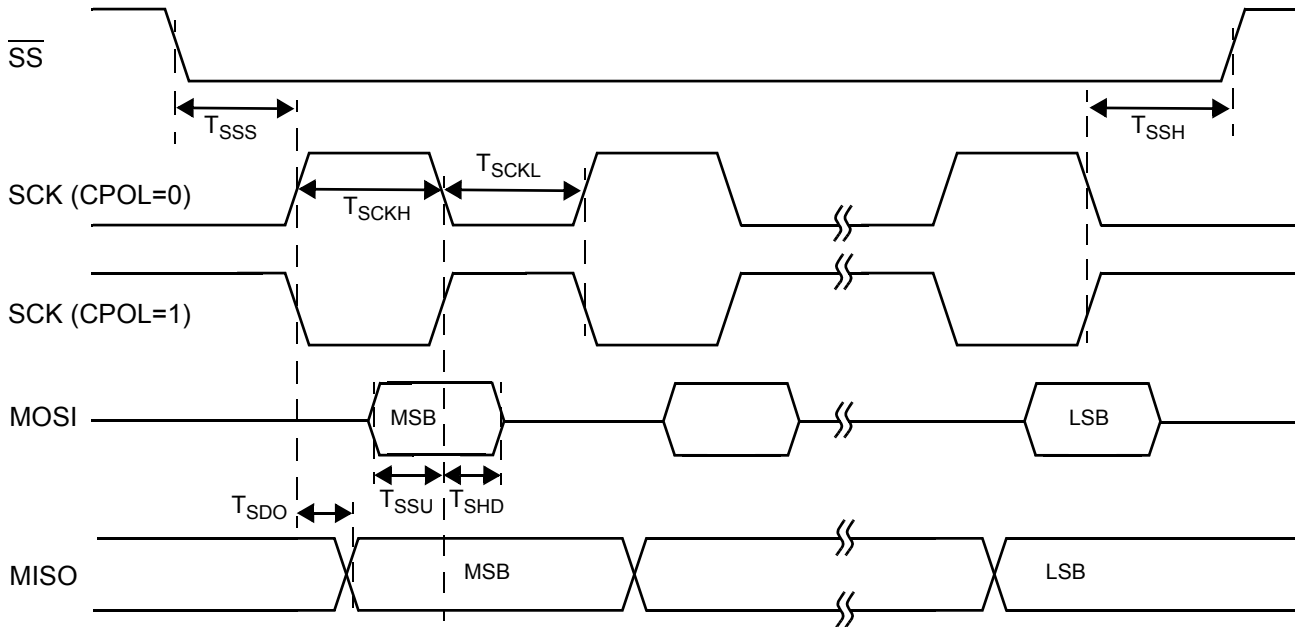
Figure 21-5. SPI Master Timing, CPHA = 0
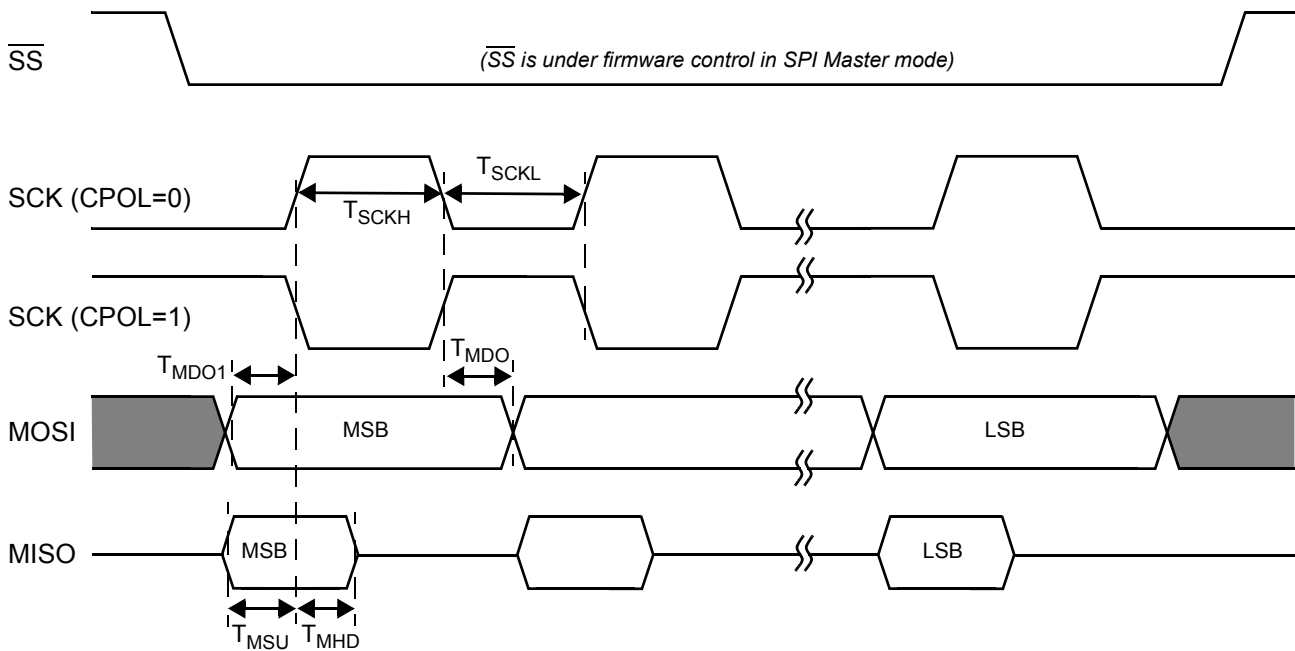
Figure 21-6. SPI Slave Timing, CPHA = 0
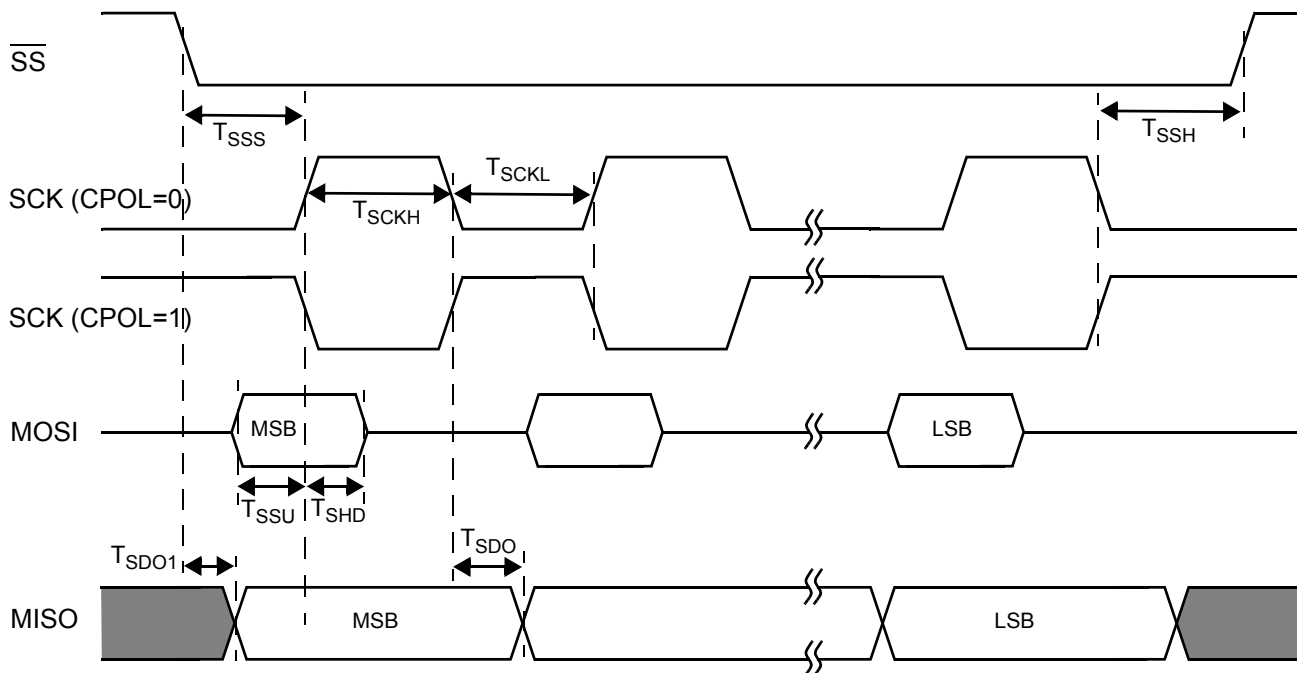
22. Ordering Information

Ordering Code	Flash Size (KB)	RAM Size (Bytes)	Package Type
CY7C60123-PVXC	8	256	48-pin SSOP
CY7C60123-PXC	8	256	40-pin PDIP
CY7C60223-SXC	8	256	24-pin SOIC
CY7C60223-QXC	8	256	24-pin QSOP

23. Package Handling

Some IC packages require baking before they are soldered to a PCB to remove moisture that may have been absorbed after leaving the factory. A label on the packaging has details about actual bake temperature and the minimum bake time to remove this moisture. The maximum bake time is the aggregate time that the parts are exposed to the bake temperature. Exceeding this exposure time may degrade device reliability.

Parameter	Description	Min	Typical	Max	Unit
$T_{BAKETEMP}$	Bake temperature	–	125	See package label	°C
$T_{BAKETIME}$	Bake time	See package label	–	72	hours

24. Package Diagrams

Figure 24-1. 24-Pin (300-Mil) SOIC S13

NOTE :
1. JEDEC STD REF MO-119
2. BODY LENGTH DIMENSION DOES NOT INCLUDE MOLD PROTRUSION/END FLASH,BUT DOES INCLUDE MOLD MISMATCH AND ARE MEASURED AT THE MOLD PARTING LINE. MOLD PROTRUSION/END FLASH SHALL NOT EXCEED 0.010 in (0.254 mm) PER SIDE
3. DIMENSIONS IN INCHES $\frac{MIN.}{MAX.}$
4. PACKAGE WEIGHT 0.65gms

PART #	
S24.3	STANDARD PKG.
SZ24.3	LEAD FREE PKG.

51-85025 *F
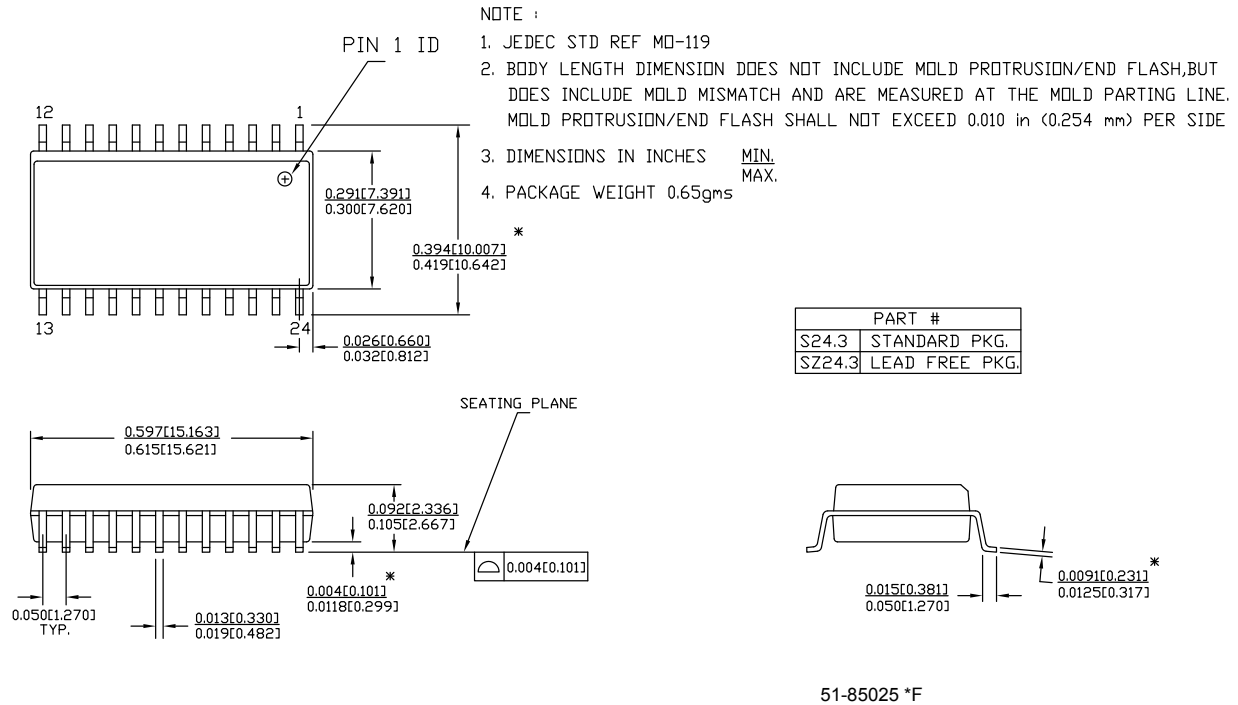
Figure 24-2. 24-Pin QSOP O241

DIMENSIONS IN INCHES $\frac{MIN.}{MAX.}$

51-85055 *D
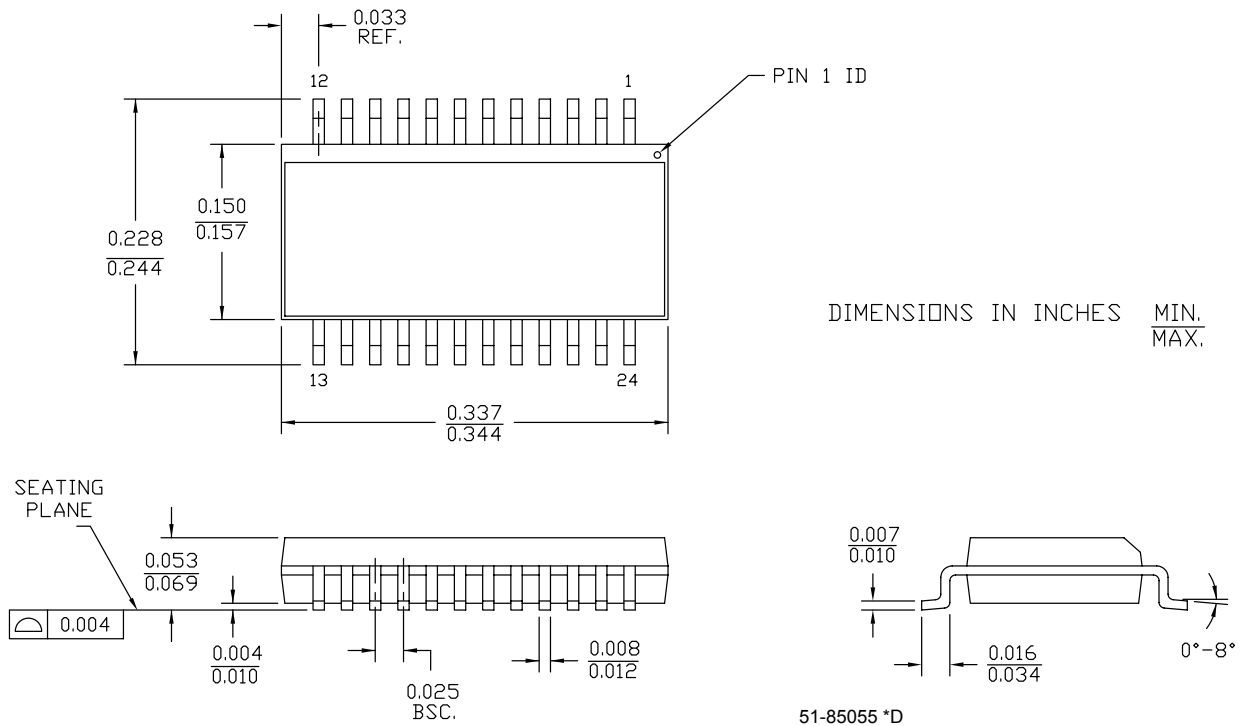
Figure 24-3. 40-Pin (600-Mil) Molded DIP P17

DIMENSIONS IN INCHES $\frac{\text{MIN.}}{\text{MAX.}}$
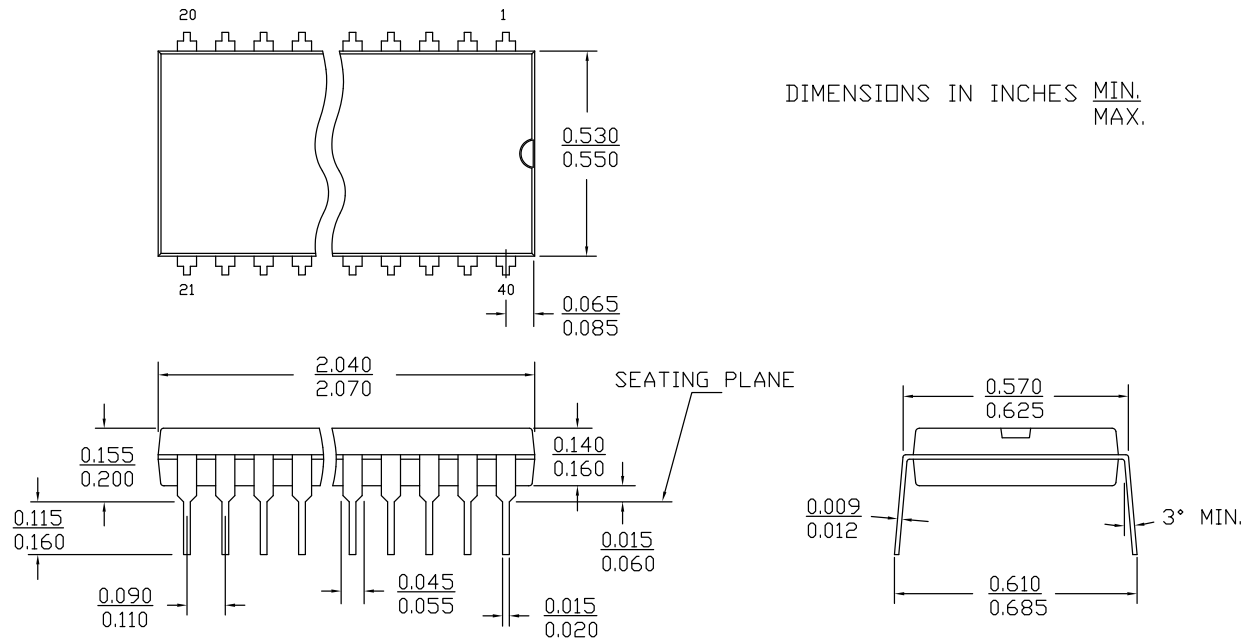
51-85019 *C

Figure 24-4. 48-Pin Shrunk Small Outline Package O48

DIMENSIONS IN INCHES $\frac{\text{MIN.}}{\text{MAX.}}$
PKG. WEIGHT: REFER TO PMDD SPEC.
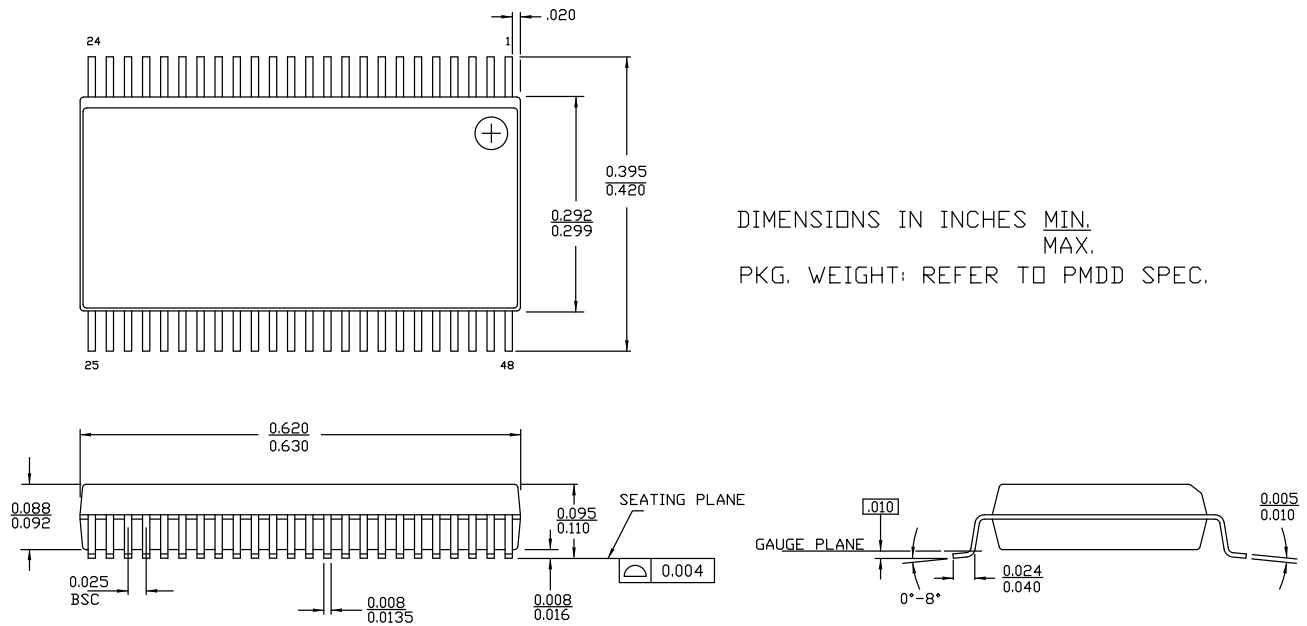
51-85061 *F

25. Document History Page

Document Title: CY7C601xx/CY7C602xx, enCoRe™ II Low-Voltage Microcontroller				
Document Number: 38-16016				
Rev.	ECN	Orig. of Change	Submission Date	Description of Change
**	327601	BON	See ECN	New data sheet
*A	400134	BHA	See ECN	Updated Power consumption values Corrected Pin Assignment Table for 24 QSOP, 24 PDIP and 28 SSOP packages Minor text changes for clarification purposes Corrected INT_MSK0 and INT_MSK1 register address Corrected register bit definitions Corrected Protection Mode Settings in Table 10-7 Updated LVD Trip Point values Added Block diagrams for Timer functional timing Replaced TBD's with actual values Added SPI Block Diagram Added Timing Block Diagrams Removed CY7C60123 DIE from Figure 5-1 Removed CY7C60123-WXC from Section 22.0 Ordering Information Updated internal 24 MHz oscillator accuracy information Added information on sending/receiving data when using 32 KHz oscillator
*B	505222	TYJ	See ECN	Minor text changes GPIO capacitance and timing diagram included Method to clear Capture Interrupt Status bit discussed Sleep and Wakeup sequence documented PIT Timer registers' R/W capability corrected to read only Modified Free-running Counter text in section 17.1.1
*C	524104	KKVTMP	See ECN	Change title from Wireless enCoRe II to enCoRe II Low Voltage
*D	1821746	VGT / FSU / AESA	See ECN	Changed "High current drive" on GPIO pins to "2 mA source current on all GPIO pins". Changed the storage temperature from -40C to 90C in "Absolute Maximum ratings" section. Added the line "The GPIOs interrupts are edge-triggered." in Tables 19-2 and 19-6. Made timing changes in Table 43. Added Figure 12-1 (SROM Table) and text after it. Also modified Table 12-1 based on Figure 12-1 (SROM Table). Changed "CAPx" to "TIOx" in Tables 18-8 and 18-9. Changed "Capturex" to "TIOx" in Figure 18-3.
*E	2620679	CMCC / PYRS	12/12/08	Added Package Handling information Formatted code in Clocking section, Removed reference to external crystal oscillator in Tables 12-2 and 12-4
*F	2761532	DVJA	09/09/2009	Changed default value of the Sleep Timer from 00(512 Hz) to 01(64 Hz) in the OSC_CR0 [0x1E0] register.
*G	2899862	XUT	03/26/10	Removed obsolete parts from the ordering information table Updated package diagrams
*H	2978027	DATT	07/12/2010	Sunset review; no technical updates. Updated content to meet style guide and template requirements.
*I	2999570	MLIM	08/03/2010	Minor change to correct revision in the document footer.
*J	3275367	NXZ	06/06/2011	Removed "CY7C60223 24-pin PDIP and CY7C60113 28-pin SSOP" from Figure 7-1. Removed "28 SSOP" and "24 PDIP" columns from Table 7-1. Removed Figure 24-2 (24-pin PDIP) and Figure 24-4 (28-pin SSOP) Updated description field of P1.0 and P1.1 in Table 7-1 on page 5

Document Title: CY7C601xx/CY7C602xx, enCoRe™ II Low-Voltage Microcontroller
Document Number: 38-16016

Rev.	ECN	Orig. of Change	Submission Date	Description of Change
*K	4499620	SELV	09/11/2014	Updated Package Diagrams: spec 51-85025 – Changed revision from *E to *F. spec 51-85055 – Changed revision from *C to *D. spec 51-85019 – Changed revision from *B to *C. spec 51-85061 – Changed revision from *D to *F. Updated in new template. Completing Sunset Review.

Sales, Solutions, and Legal Information

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

Automotive	cypress.com/go/automotive
Clocks & Buffers	cypress.com/go/clocks
Interface	cypress.com/go/interface
Lighting & Power Control	cypress.com/go/powerpsoc
	cypress.com/go/plc
Memory	cypress.com/go/memory
PSoC	cypress.com/go/psoc
Touch Sensing	cypress.com/go/touch
USB Controllers	cypress.com/go/USB
Wireless/RF	cypress.com/go/wireless

PSoC® Solutions

psoc.cypress.com/solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

Cypress Developer Community

Community | Forums | Blogs | Video | Training

Technical Support

cypress.com/go/support

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

[Cypress Semiconductor](#):
  CY7C60123-PVXC CY7C60223-QXC CY7C60223-SXC CY7C60123-PXC