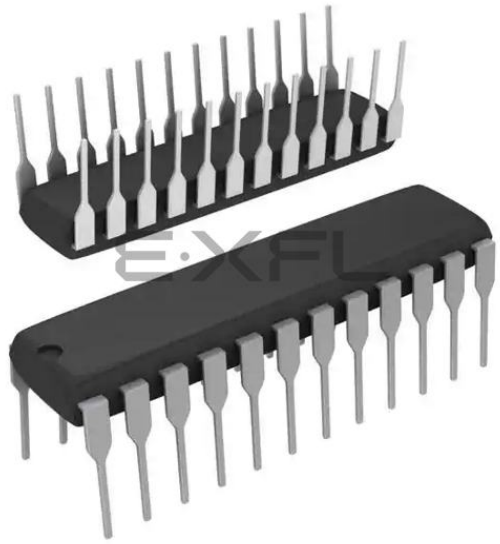


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

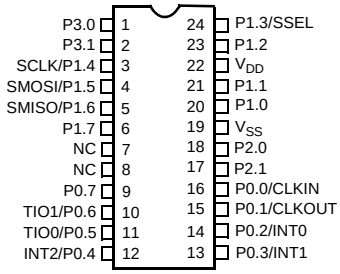
Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	M8C
Core Size	8-Bit
Speed	12MHz
Connectivity	SPI
Peripherals	LVD, POR, WDT
Number of I/O	20
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Through Hole
Package / Case	24-DIP (0.300", 7.62mm)
Supplier Device Package	24-PDIP
Purchase URL	https://www.e-xfl.com/product-detail/infineon-technologies/cy7c60223-pxc

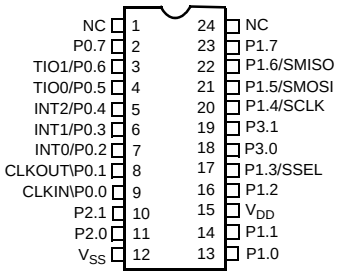
6. Pinouts

Figure 6-1. Package Configurations
Top View

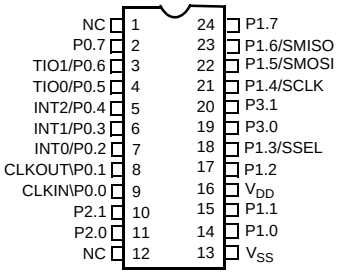
CY7C60223
24-Pin PDIP



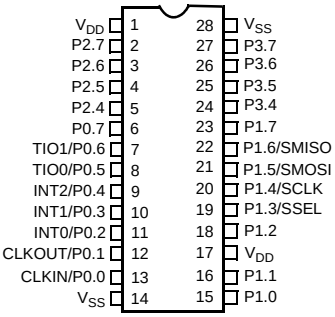
CY7C60223
24-Pin SOIC



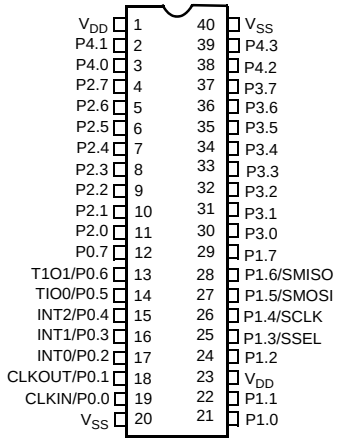
CY7C60223
24-Pin QSOP



CY7C60113
28-Pin SSOP



CY7C60123
40-Pin PDIP



CY7C60123
48-Pin SSOP

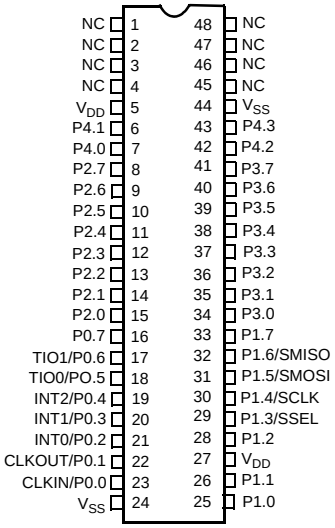


Table 6-1. Pin Assignments (continued)

48 SSOP	40 PDIP	28 SSOP	24 QSOP	24 SOIC	24 PDIP	Name	Description
22	18	12	8	8	15	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	11	7	7	14	P0.2/INT0	GPIO port 0 bit 2—Configured individually Optional rising edge interrupt INT0.
20	16	10	6	6	13	P0.3/INT1	GPIO port 0 bit 3—Configured individually Optional rising edge interrupt INT1.
19	15	9	5	5	12	P0.4/INT2	GPIO port 0 bit 4—Configured individually Optional rising edge interrupt INT2.
18	14	8	4	4	11	P0.5/TIO0	GPIO port 0 bit 5—Configured individually Alternate function timer capture inputs or timer output TIO0.
17	13	7	3	3	10	P0.6/TIO1	GPIO port 0 bit 6—Configured individually Alternate function timer capture inputs or timer output TIO1.
16	12	6	2	2	9	P0.7	GPIO port 0 bit 7—Configured individually
1,2,3, 4			1	1	7	NC	No connect
45,46, 47,48			12	24	8	NC	No connect
5	1	17				V_{DD}	Power
27	23	1	16	15	22		
44	40	14	–	–	–	V_{SS}	Ground
24	20	28	13	12	19		

9.2.5 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 9-11. 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.
```

9.2.6 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 9-12. 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.
```

9.2.7 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 9-13. 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.
```

9.2.8 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 9-14. 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.
```

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 11-2. 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

11.4.1 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 11-3. 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.

11.5 SROM Function Descriptions

11.5.1 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.

11.5.2 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 11-4. 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

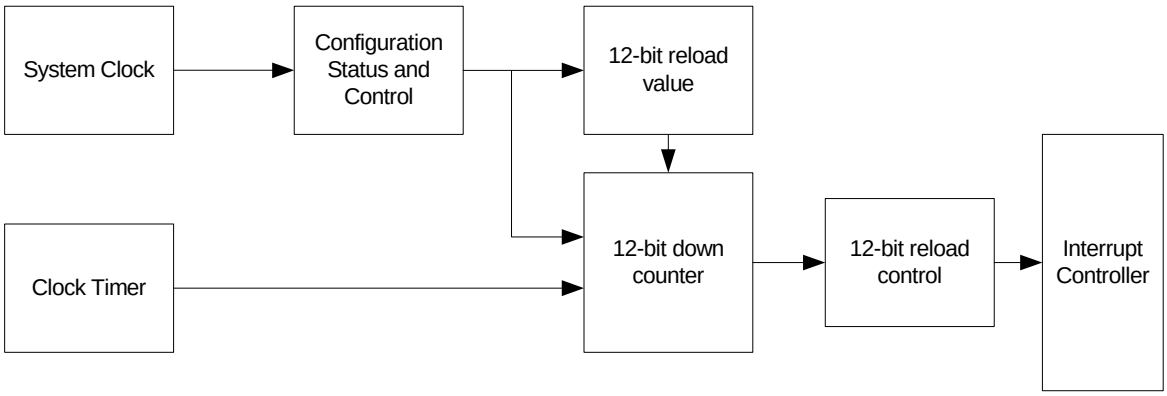
11.6 SROM Table Read Description

The Silicon IDs for enCoRe II LV devices are stored in SROM tables in the part, as shown in Figure 11-3.

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 15, the SROM variables occupy address F8h through FFh in the SRAM. Each of the variables and their definition are given in the section, SROM on page 15.

```
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
11-1 on page 15)
          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 12-2. Programmable Interval Timer Block Diagram



12.2.3 Timer Capture Clock (TCAPCLK)

The Timer Capture clock (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 12-3. Timer Capture Block Diagram

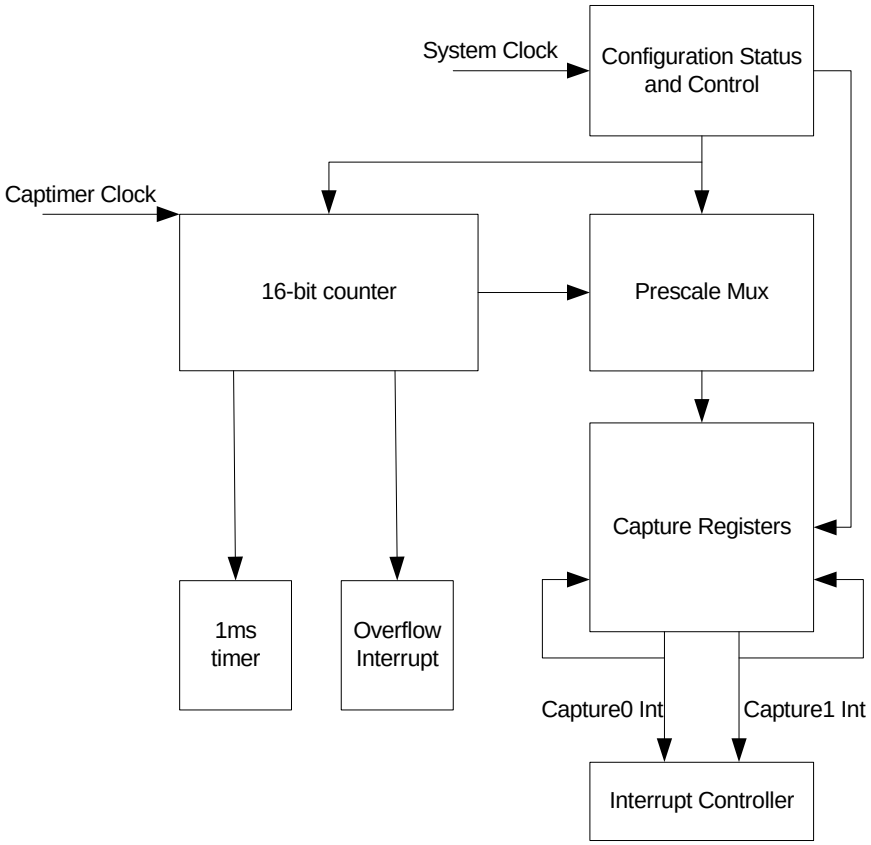


Table 12-1. Timer Clock Configuration (TMRCLKCR) [0x31] [R/W]

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 12-4 on page 25.)

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.

12.2.4 Internal Clock Trim

Table 12-1. 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 3.30V 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

12.2.5 External Clock Trim

Table 12-2. 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

14.1 Sleep Sequence

The Sleep bit is an input into the sleep logic circuit. This circuit is designed to sequence the device into and out of the hardware sleep state. The hardware sequence to put the device to sleep is shown in Figure 14-1 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 14-1 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.

14.1.1 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.

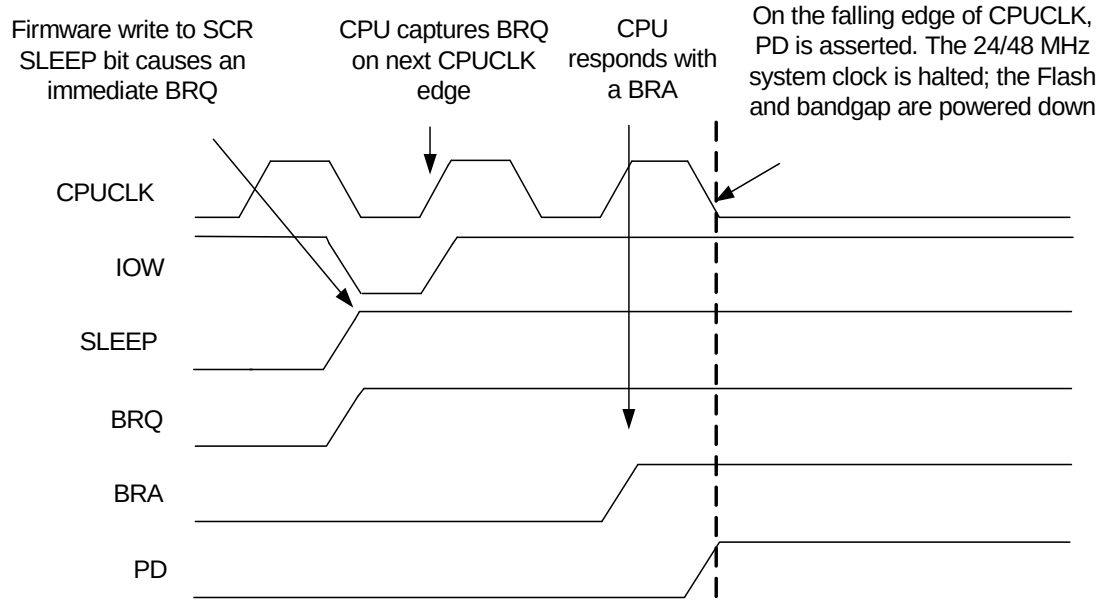
All the other blocks go to the power down mode automatically on suspend.

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 24 MHz or 32 kHz oscillator clocks on to the P01_CLKOUT pin.

Figure 14-1. Sleep Timing



16.1.3 P2 Data

Table 16-3. 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]

16.1.4 P3 Data

Table 16-4. 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]

16.1.5 P4 Data

Table 16-5. 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.

16.2 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.

16.2.1 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 as well.

16.2.2 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.

16.2.3 TTL Thresh

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

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

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

Table 16-10. 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 17-2 on page 45)

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.

16.2.19 P1.7 Configuration

Table 16-11. 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.

16.2.20 P2 Configuration

Table 16-12. 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 8 mA sink drive capability is available on this pin regardless of the setting of the High Sink bit.

16.2.21 P3 Configuration

Table 16-13. 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 8 mA sink drive capability is available on this pin regardless of the setting of the High Sink bit.

16.2.22 P4 Configuration

Table 16-14. 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.

17. 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 Master Mode. SPI is a four pin serial interface comprised of a clock, an enable, and two data pins.

Figure 17-1. SPI Block Diagram

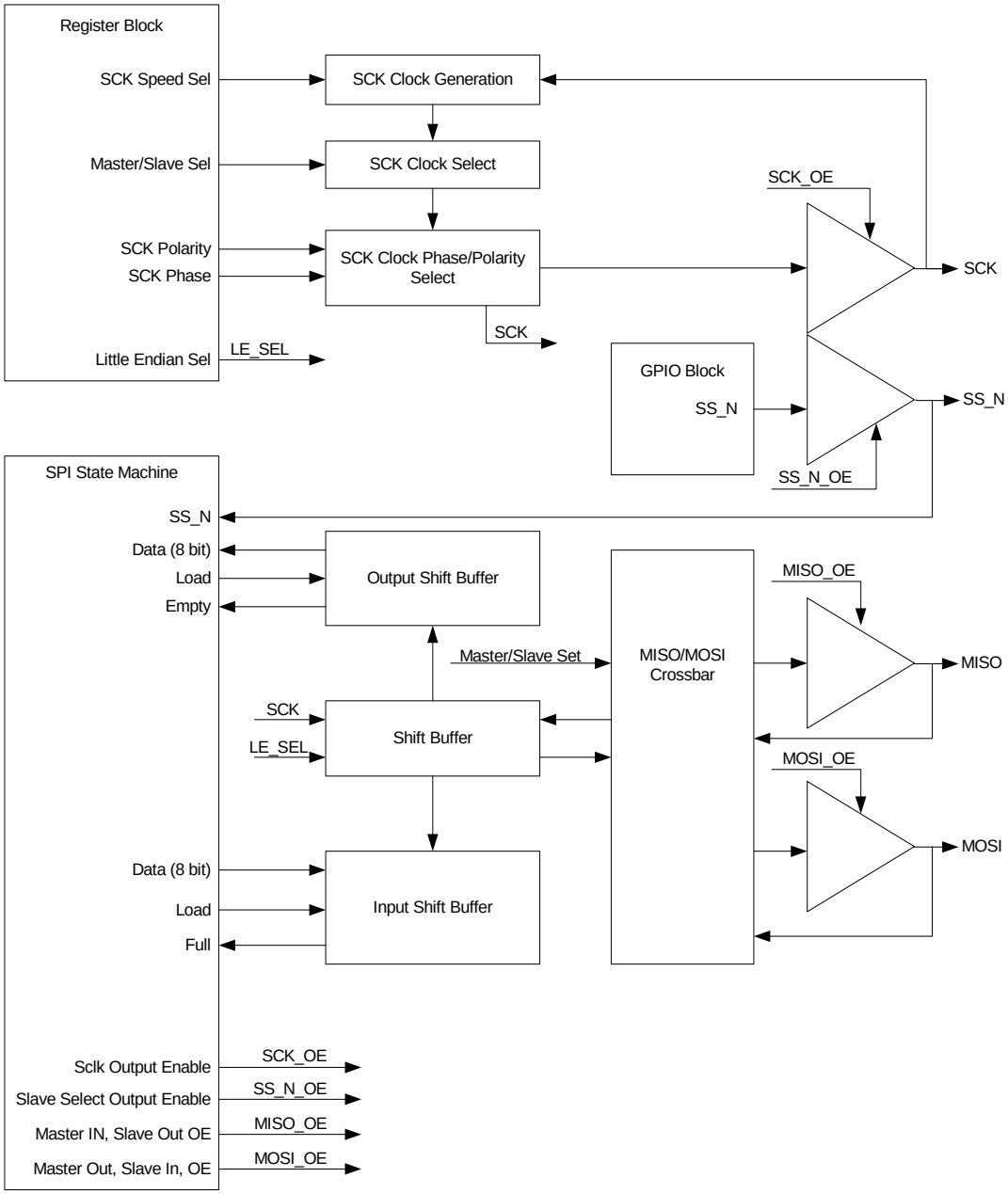


Table 18-2. Free Running Timer High Order Byte (FRTMRH) [0x21] [R/W]

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.

18.1.2 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 18-2. Time Capture Block Diagram

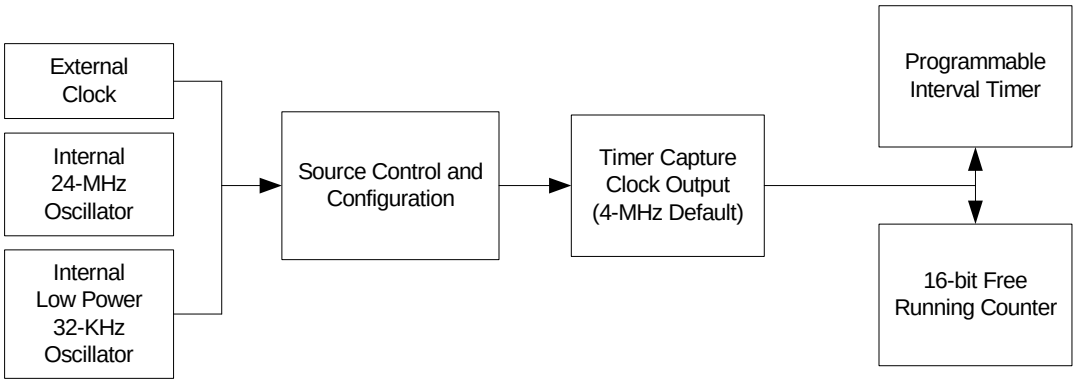


Table 18-1. Timer Configuration (TMRCR) [0x2A] [R/W]

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 18-2. 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 18-3. 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 18-4. 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.

19.1 Architectural Description

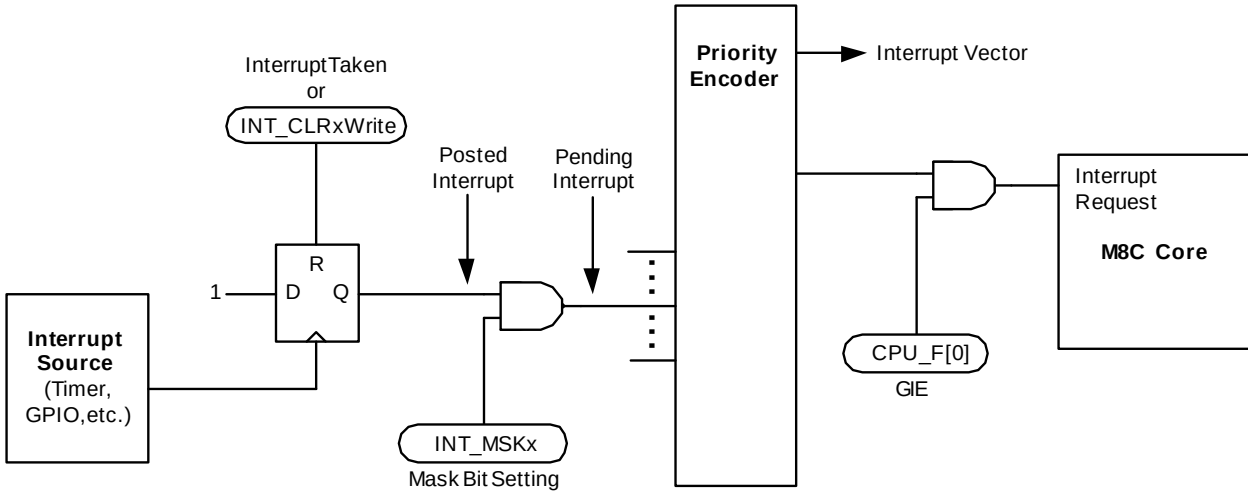
An interrupt is posted when its interrupt conditions occur. This results in the flip-flop in Figure 19-1 clocking in a '1'. The interrupt remains posted until the interrupt is taken or until it is cleared by writing to the appropriate INT_CLRx register.

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 19-1.

Figure 19-1. Interrupt Controller Block Diagram



19.2 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.
2. The current executing instruction finishes.
3. 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).
4. Program execution vectors to the interrupt table. Typically, a LJMP instruction in the interrupt table sends execution to the user's Interrupt Service Routine (ISR) for this interrupt.
5. 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).
6. 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.
7. Execution resumes at the next instruction, after the one that occurred before the interrupt. However, if there are more pending interrupts, the subsequent interrupts are processed before the next normal program instruction.

19.3 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. 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} .....–0.5V to +7.0V
DC Input Voltage ..... –0.5V to + V_{CC} + 0.5V
DC Voltage Applied to Outputs in High-Z State..... –0.5V to + V_{CC} + 0.5V

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 2200V
Latch up Current ..... 200 mA

20.1 DC Characteristics

Parameter	Description	Conditions	Min	Typical	Max	Unit
	General					
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, Vdd = 3.3V, T = 75°C		4.25	11	mA
		CPU =12 MHz, Vdd = 2.7V, T = 25°C		3.25	-	mA
I_{CC2}	V_{CC} Operating Supply Current	CPU = 6 MHz, Vdd = 3.3V, T = 75°C		3.15	9	mA
		CPU = 6 MHz, Vdd = 3.3V, T = 25°C		2.45	-	mA
I_{CC3}	V_{CC} Operating Supply Current	CPU = 3 MHz, Vdd = 2.7V, 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.

Figure 20-4. SPI Slave Timing, CPHA = 1

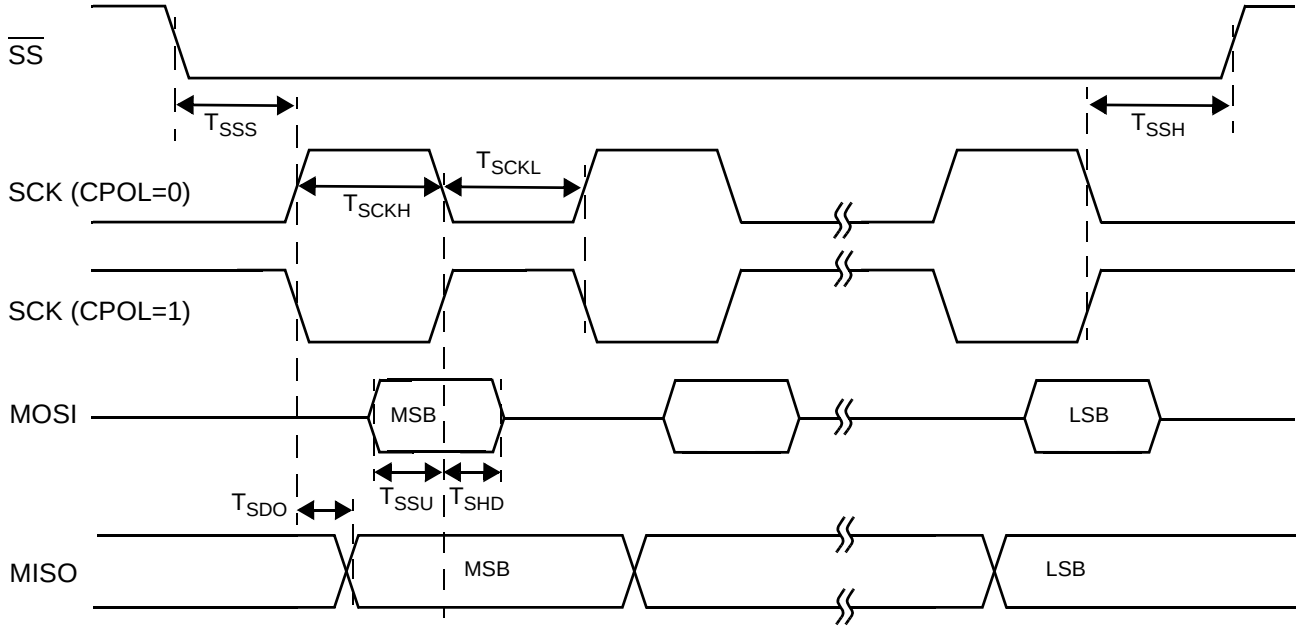


Figure 20-5. SPI Master Timing, CPHA = 0

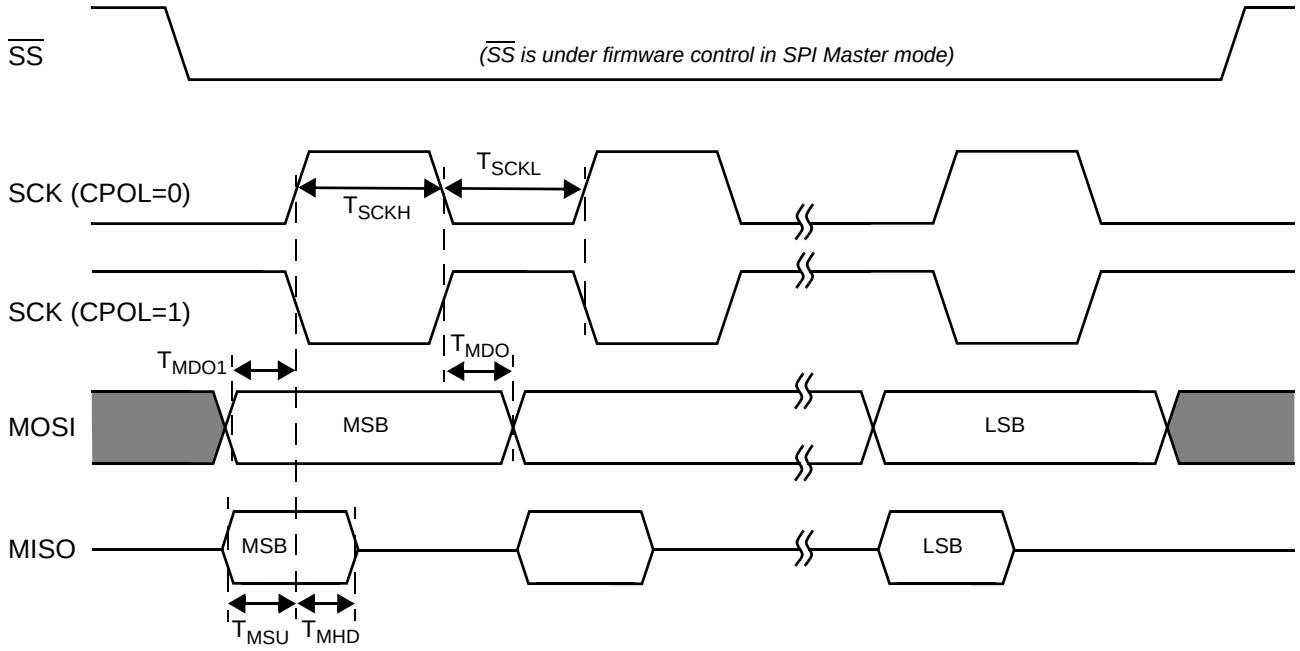


Figure 23-3. 24-Pin QSOP O241

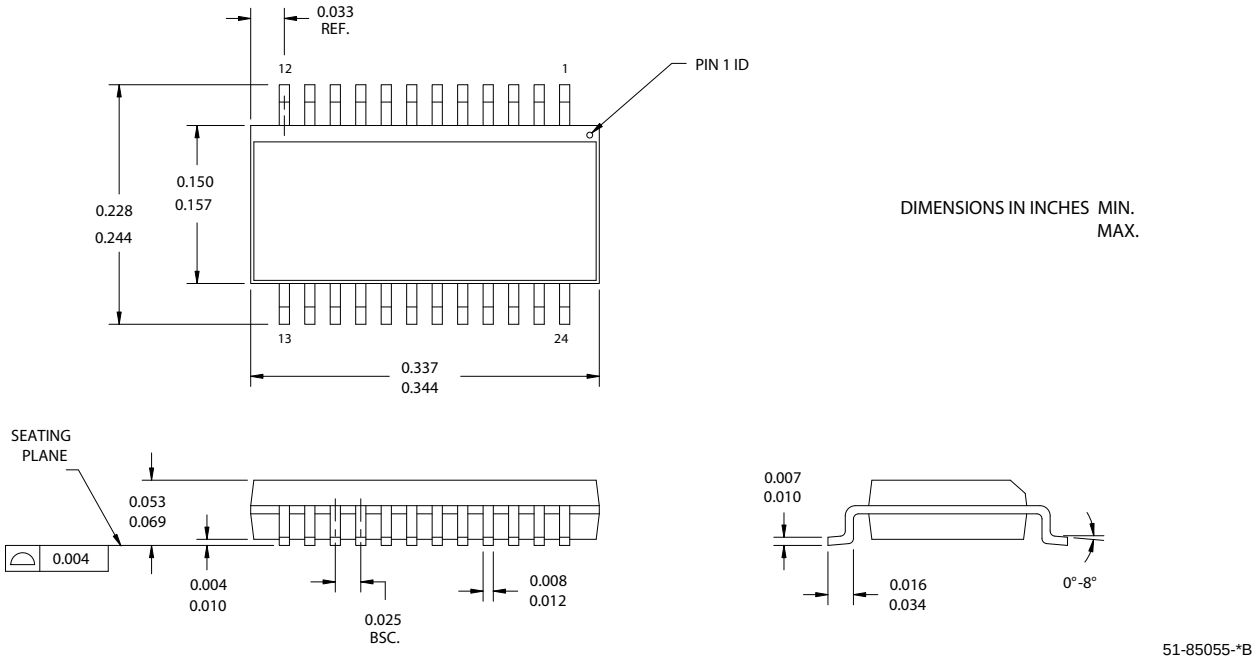
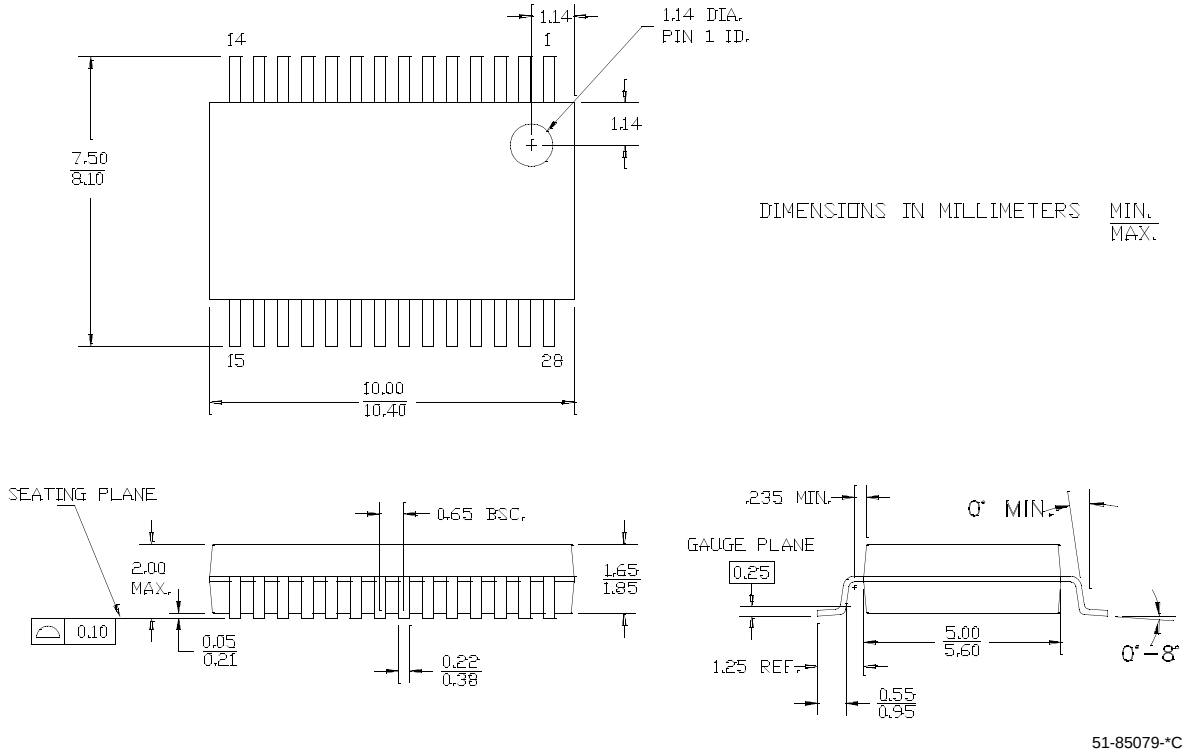


Figure 23-4. 28-Pin (5.3 mm) Shrunk Small Outline Package O28



24. 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.