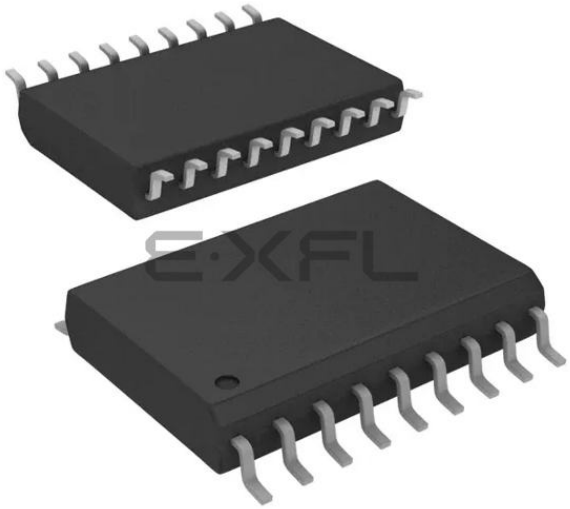


Welcome to E-XFL.COM

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

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

What Are Embedded - Microcontrollers - Application Specific?

Application-specific microcontrollers are engineered to

Details	
Product Status	Obsolete
Applications	USB Microcontroller
Core Processor	M8B
Program Memory Type	OTP (8kB)
Controller Series	CY7C637xx
RAM Size	256 x 8
Interface	PS/2, USB
Number of I/O	10
Voltage - Supply	3.5V ~ 5.5V
Operating Temperature	0°C ~ 70°C
Mounting Type	Surface Mount
Package / Case	18-SOIC (0.295", 7.50mm Width)
Supplier Device Package	18-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/infineon-technologies/cy7c63723c-sxct

Functional Overview

enCoRe USB—The New USB Standard

Cypress has reinvented its leadership position in the low-speed USB market with a new family of innovative microcontrollers. Introducing...enCoRe USB—“enhanced Component Reduction.” Cypress has leveraged its design expertise in USB solutions to create a new family of low-speed USB microcontrollers that enables peripheral developers to design new products with a minimum number of components. At the heart of the enCoRe USB technology is the breakthrough design of a crystalless oscillator. By integrating the oscillator into our chip, an external crystal or resonator is no longer needed. We have also integrated other external components commonly found in low-speed USB applications such as pull-up resistors, wake-up circuitry, and a 3.3 V regulator. All of this adds up to a lower system cost.

The CY7C637xxC is an 8-bit RISC one-time-programmable (OTP) microcontroller. The instruction set has been optimized specifically for USB and PS/2 operations, although the microcontrollers can be used for a variety of other embedded applications.

The CY7C637xxC features up to 16 GPIO pins to support USB, PS/2 and other applications. The I/O pins are grouped into two ports (Port 0 to 1) where each pin can be individually configured as inputs with internal pull-ups, open drain outputs, or traditional CMOS outputs with programmable drive strength of up to 50 mA output drive. Additionally, each I/O pin can be used to generate a GPIO interrupt to the microcontroller. Note the GPIO interrupts all share the same “GPIO” interrupt vector.

The CY7C637xxC microcontrollers feature an internal oscillator. With the presence of USB traffic, the internal oscillator can be set to precisely tune to USB timing requirements (6 MHz ±1.5%). Optionally, an external 6-MHz ceramic resonator can be used to provide a higher precision reference for USB operation. This clock generator reduces the clock-related noise emissions (EMI). The clock generator provides the 6- and 12-MHz clocks that remain internal to the microcontroller.

The CY7C637xxC has 8 Kbytes of EPROM and 256 bytes of data RAM for stack space, user variables, and USB FIFOs.

These parts include low-voltage reset logic, a Watchdog timer, a vectored interrupt controller, a 12-bit free-running timer, and capture timers. The low-voltage reset (LVR) logic detects when power is applied to the device, resets the logic to a known state, and begins executing instructions at EPROM address 0x0000. LVR will also reset the part when V_{CC} drops below the operating voltage range. The Watchdog timer can be used to ensure the firmware never gets stalled for more than approximately 8 ms.

The microcontroller supports 10 maskable interrupts in the vectored interrupt controller. Interrupt sources include the USB Bus-Reset, the 128-µs and 1.024-ms outputs from the free-running timer, three USB endpoints, two capture timers, an internal wake-up timer and the GPIO ports. The timers bits cause periodic interrupts when enabled. The USB endpoints interrupt after USB transactions complete on the bus. The capture timers interrupt whenever a new timer value is saved due to a selected GPIO edge event. The GPIO ports have a level of masking to select which GPIO inputs can cause a GPIO interrupt. For additional flexibility, the input transition polarity that causes an interrupt is programmable for each GPIO pin. The interrupt polarity can be either rising or falling edge [1].

The free-running 12-bit timer clocked at 1 MHz provides two interrupt sources as noted above (128 µs and 1.024 ms). The timer can be used to measure the duration of an event under firmware control by reading the timer at the start and end of an event, and subtracting the two values. The four capture timers save a programmable 8 bit range of the free-running timer when a GPIO edge occurs on the two capture pins (P0.0, P0.1).

The CY7C637xxC includes an integrated USB serial interface engine (SIE) that supports the integrated peripherals. The hardware supports one USB device address with three endpoints. The SIE allows the USB host to communicate with the function integrated into the microcontroller. A 3.3V regulated output pin provides a pull-up source for the external USB resistor on the D– pin.

The USB D+ and D– USB pins can alternately be used as PS/2 SCLK and SDATA signals, so that products can be designed to respond to either USB or PS/2 modes of operation. PS/2 operation is supported with internal pull-up resistors on SCLK and SDATA, the ability to disable the regulator output pin, and an interrupt to signal the start of PS/2 activity. No external components are necessary for dual USB and PS/2 systems, and no GPIO pins need to be dedicated to switching between modes. Slow edge rates operate in both modes to reduce EMI.

Note
1. Errata: When a falling edge interrupt is enabled for a GPIO pin, reading the GPIO Port 1 coincident to a rising edge of that GPIO signal may generate a false GPIO interrupt. In similar manner when a rising edge interrupt is enabled for a GPIO pin, reading the GPIO Port 1 coincident to a falling edge of that GPIO signal may generated a false GPIO interrupt. For more information, see the “Errata” on page 53.

page 44 for the value of t_{START}). Program execution begins from address 0x0000 after this t_{START} delay period. This provides time for V_{CC} to stabilize before the part executes code. See Low-voltage Reset (LVR) on page 14 for more details.

1 = Disables the LVR circuit.

0 = Enables the LVR circuit.

Bit 2: Precision USB Clocking Enable

The Precision USB Clocking Enable only affects operation in internal oscillator mode. **In that mode, this bit must be set to 1 to cause the internal clock to automatically precisely tune to USB timing requirements (6 MHz ±1.5%)**. The frequency may have a looser initial tolerance at power-up, but all USB transmissions from the chip will meet the USB specification.

1 = Enabled. The internal clock accuracy is **6 MHz ±1.5%** after USB traffic is received**.**

0 = Disabled. The internal clock accuracy is 6 MHz ±5%.

Bit 1: Internal Clock Output Disable

The Internal Clock Output Disable is used to keep the internal clock from driving out to the XTALOUT pin. This bit has no effect in the external oscillator mode.

1 = Disable internal clock output. XTALOUT pin will drive HIGH.

0 = Enable the internal clock output. The internal clock is driven out to the XTALOUT pin.

Bit 0: External Oscillator Enable

At power-up, the chip operates from the internal clock by default. Setting the External Oscillator Enable bit HIGH disables the internal clock, and halts the part while the external resonator/crystal oscillator is started. Clearing this bit has no immediate effect, although the state of this bit is used when waking out of suspend mode to select between internal and external clock. In internal clock mode, XTALIN pin will be configured as an input with a weak pull-down and can be used as a GPIO input (P2.1).

1 = Enable the external oscillator. The clock is switched to external clock mode, as described in Internal/External Oscillator Operation on page 13.

0 = Enable the internal oscillator.

Internal/External Oscillator Operation

The internal oscillator provides an operating clock, factory set to a nominal frequency of 6 MHz. This clock requires no external components. At power-up, the chip operates from the internal clock. In this mode, the internal clock is buffered and driven to the XTALOUT pin by default, and the state of the XTALIN pin can be read at Port 2.1. While the internal clock is enabled, its output can be disabled at the XTALOUT pin by setting the Internal Clock Output Disable bit of the Clock Configuration Register.

Setting the External Oscillator Enable bit of the Clock Configuration Register HIGH disables the internal clock, and halts the part while the external resonator/crystal oscillator is started. The steps involved in switching from Internal to External Clock mode are as follows:

1. At reset, chip begins operation using the internal clock.
2. Firmware sets Bit 0 of the Clock Configuration Register. For example,

```
mov A, 1h      ; Set Bit 0 HIGH (External Oscillator
                 Enable bit). Bit 7 cleared gives
                 faster start-up
iowr F8h       ; Write to Clock Configuration
                 Register
```

3. Internal clocking is halted, the internal oscillator is disabled, and the external clock oscillator is enabled.
4. After the external clock becomes stable, chip clocks are re-enabled using the external clock signal. (Note that the time for the external clock to become stable depends on the external resonating device; see next section.)
5. After an additional delay the CPU is released to run. This delay depends on the state of the Ext. Clock Resume Delay bit of the Clock Configuration Register. The time is 128 µs if the bit is 0, or 4 ms if the bit is 1.
6. Once the chip has been set to external oscillator, it can only return to internal clock when waking from suspend mode. Clearing bit 0 of the Clock Configuration Register will not re-enable internal clock mode until suspend mode is entered. See Suspend Mode on page 15 for more details on suspend mode operation.

If the Internal Clock is enabled, the XTALIN pin can serve as a general purpose input, and its state can be read at Port 2, Bit 1 (P2.1). Refer to Figure 13 on page 19 for the Port 2 Data Register. In this mode, there is a weak pull-down at the XTALIN pin. This input cannot provide an interrupt source to the CPU.

External Oscillator

The user can connect a low-cost ceramic resonator or an external oscillator to the XTALIN/XTALOUT pins to provide a precise reference frequency for the chip clock, as shown in Figure 3 on page 12. The external components required are a ceramic resonator or crystal and any associated capacitors. To run from the external resonator, the External Oscillator Enable bit of the Clock Configuration Register must be set to 1, as explained in the previous section.

Start-up times for the external oscillator depend on the resonating device. Ceramic resonator based oscillators typically start in less than 100 µs, while crystal based oscillators take longer, typically 1 to 10 ms. Board capacitance should be minimized on the XTALIN and XTALOUT pins by keeping the traces as short as possible.

An external 6-MHz clock can be applied to the XTALIN pin if the XTALOUT pin is left open.

Reset

The USB Controller supports three types of resets. The effects of the reset are listed below. The reset types are:

1. Low-voltage Reset (LVR)
2. Brown Out Reset (BOR)
3. Watchdog Reset (WDR)

The occurrence of a reset is recorded in the Processor Status and Control Register (Figure 34 on page 30). Bits 4 (Low-voltage or Brown-out Reset bit) and 6 (Watchdog Reset bit) are used to

Bit [7:0]: P0[7:0] Mode 0

1 = Port 0 Mode 0 is logic HIGH

0 = Port 0 Mode 0 is logic LOW

Figure 10. GPIO Port 0 Mode1 Register (Address 0x0B)

Bit #	7	6	5	4	3	2	1	0
Bit Name	P0[7:0] Mode1							
Read/Write	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit [7:0]: P0[7:0] Mode 1

1 = Port Pin Mode 1 is logic HIGH

0 = Port Pin Mode 1 is logic LOW

Figure 11. GPIO Port 1 Mode0 Register (Address 0x0C)

Bit #	7	6	5	4	3	2	1	0
Bit Name	P1[7:0] Mode0							
Read/Write	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit [7:0]: P1[7:0] Mode 0

1 = Port Pin Mode 0 is logic HIGH

0 = Port Pin Mode 0 is logic LOW

Figure 12. GPIO Port 1 Mode1 Register (Address 0x0D)

Bit #	7	6	5	4	3	2	1	0
Bit Name	P1[7:0] Mode1							
Read/Write	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit [7:0]: P1[7:0] Mode 1

1 = Port Pin Mode 1 is logic HIGH

0 = Port Pin Mode 1 is logic LOW

Each pin can be independently configured as high-impedance inputs, inputs with internal pull-ups, open drain outputs, or traditional CMOS outputs with selectable drive strengths.

The driving state of each GPIO pin is determined by the value written to the pin's Data Register and by its associated Mode0 and Mode1 bits. *Table 3* lists the configuration states based on these bits. The GPIO ports default on reset to all Data and Mode Registers cleared, so the pins are all in a high-impedance state. The available GPIO output drive strength are:

- **Hi-Z Mode** (Mode1 = 0 and Mode0 = 0)

  Q1, Q2, and Q3 (Figure 6 on page 17) are OFF. The GPIO pin is not driven internally. Performing a read from the Port Data Register return the actual logic value on the port pins.

- **Low Sink Mode** (Mode1 = 1, Mode0 = 0, and the pin's Data Register = 0)

  Q1 and Q3 are OFF. Q2 is ON. The GPIO pin is capable of sinking 2 mA of current.

- **Medium Sink Mode** (Mode1 = 0, Mode0 = 1, and the pin's Data Register = 0)

  Q1 and Q3 are OFF. Q2 is ON. The GPIO pin is capable of sinking 8 mA of current.

- **High Sink Mode** (Mode1 = 1, Mode0 = 1, and the pin's Data Register = 0)

  Q1 and Q3 are OFF. Q2 is ON. The GPIO pin is capable of sinking 50 mA of current.

- **High Drive Mode** (Mode1 = 0 or 1, Mode0 = 1, and the pin's Data Register = 1)

  Q1 and Q2 are OFF. Q3 is ON. The GPIO pin is capable of sourcing 2 mA of current.

- **Resistive Mode** (Mode1 = 1, Mode0 = 0, and the pin's Data Register = 1)

  Q2 and Q3 are OFF. Q1 is ON. The GPIO pin is pulled up with an internal 14-kΩ resistor.

Note that open drain mode can be achieved by fixing the Data and Mode1 Registers LOW, and switching the Mode0 register.

Input thresholds are CMOS, or TTL as shown in the table (See DC Characteristics on page 42 for the input threshold voltage in TTL or CMOS modes). Both input modes include hysteresis to minimize noise sensitivity. In suspend mode, if a pin is used for a wake-up interrupt using an external R-C circuit, CMOS mode is preferred for lowest power.

Table 3. Ports 0 and 1 Output Control Truth Table

Data Register	Mode1	Mode0	Output Drive Strength	Input Threshold
0	0	0	Hi-Z	CMOS
1			Hi-Z	TTL
0	0	1	Medium (8 mA) Sink	CMOS
1			High Drive	CMOS
0	1	0	Low (2 mA) Sink	CMOS
1			Resistive	CMOS
0	1	1	High (50 mA) Sink	CMOS
1			High Drive	CMOS

Auxiliary Input Port

Port 2 serves as an auxiliary input port as shown in Figure 13. The Port 2 inputs all have TTL input thresholds.

Figure 15. USB Device Address Register (Address 0x10)

Bit #	7	6	5	4	3	2	1	0
Bit Name	Device Address Enable	Device Address						
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

In either USB or PS/2 mode, this register is cleared by both hardware resets and the USB bus reset. See Interrupt Sources on page 32 for more information on the USB Bus Reset – PS/2 interrupt.

Bit 7: Device Address Enable

This bit must be enabled by firmware before the serial interface engine (SIE) will respond to USB traffic at the address specified in Bit [6:0].

1 = Enable USB device address.

0 = Disable USB device address.

Bit [6:0]: Device Address Bit [6:0]

These bits must be set by firmware during the USB enumeration process (i.e., SetAddress) to the non-zero address assigned by the USB host.

USB Control Endpoint

All USB devices are required to have an endpoint number 0 (EP0) that is used to initialize and control the USB device. EP0 provides access to the device configuration information and allows generic USB status and control accesses. EP0 is bidirectional as the device can both receive and transmit data. EP0 uses an 8-byte FIFO at SRAM locations 0xF8-0xFF, as shown in Data Memory Organization on page 10.

The EP0 endpoint mode register uses the format shown in Figure .

Figure 16. Endpoint 0 Mode Register (Address 0x12)

Bit #	7	6	5	4	3:0
Bit Name	SETUP Received	IN Received	OUT Received	ACKed Transaction	Mode Bit
Read/ Write	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0 0 0 0

The SIE provides a locking feature to prevent firmware from overwriting bits in the USB Endpoint 0 Mode Register. Writes to the register have no effect from the point that Bit[6:0] of the register are updated (by the SIE) until the firmware reads this register. The CPU can unlock this register by reading it.

Because of these hardware-locking features, firmware should perform an read after a write to the USB Endpoint 0 Mode Register and USB Endpoint 0 Count Register (Figure 18) to verify that the contents have changed as desired, and that the SIE has not updated these values.

Bit [7:4] of this register are cleared by any non-locked write to this register, regardless of the value written.

Bit 7: SETUP Received

1 = A valid SETUP packet has been received. This bit is forced HIGH from the start of the data packet phase of the SETUP transaction until the start of the ACK packet returned by the SIE. The CPU is prevented from clearing this bit during this interval. While this bit is set to ‘1’, the CPU cannot write to the EP0 FIFO. This prevents firmware from overwriting an incoming SETUP transaction before firmware has a chance to read the SETUP data.

0 = No SETUP received. This bit is cleared by any non-locked writes to the register.

Bit 6: IN Received

1 = A valid IN packet has been received. This bit is updated to ‘1’ after the last received packet in an IN transaction. This bit is cleared by any non-locked writes to the register.

0 = No IN received. This bit is cleared by any non-locked writes to the register.

Bit 5: OUT Received

1 = A valid OUT packet has been received. This bit is updated to ‘1’ after the last received packet in an OUT transaction. This bit is cleared by any non-locked writes to the register.

0 = No OUT received. This bit is cleared by any non-locked writes to the register.

Bit 4: ACKed Transaction

The ACKed Transaction bit is set whenever the SIE engages in a transaction to the register's endpoint that completes with an ACK packet.

1 = The transaction completes with an ACK.

0 = The transaction does not complete with an ACK.

Bit [3:0]: Mode Bit[3:0]

The endpoint modes determine how the SIE responds to USB traffic that the host sends to the endpoint. For example, if the endpoint Mode Bits [3:0] are set to 0001 which is NAK IN/OUT mode as shown in *Table 8*, the SIE will send NAK handshakes in response to any IN or OUT token sent to this endpoint. In this NAK IN/OUT mode, the SIE will send an ACK handshake when the host sends a SETUP token to this endpoint. The mode encoding is shown in *Table 8*. Additional information on the mode bits can be found in *Table 9* and *Table 10*. These modes give the firmware total control on how to respond to different tokens sent to the endpoints from the host.

In addition, the Mode Bits are automatically changed by the SIE in response to many USB transactions. For example, if the Mode Bit [3:0] are set to 1011 which is ACK OUT-NAK IN mode as shown in *Table 8*, the SIE will change the endpoint Mode Bit [3:0] to NAK IN/OUT (0001) mode after issuing an ACK handshake in response to an OUT token. Firmware needs to update the mode for the SIE to respond appropriately.

USB signaling in the case where the VREG pin is used as an input, and an external regulator is provided for the USB pull-up resistor. This also limits the swing on the D– and D+ pins to about 1V above the internal regulator voltage, so the Device Address Enable bit normally should only be set for USB operating modes.

The regulator output is only designed to provide current for the USB pull-up resistor. In addition, the output voltage at the VREG pin is effectively disconnected when the CY7C637xxC device transmits USB from the internal SIE. This means that the VREG pin does not provide a stable voltage during transmits, although this does not affect USB signaling.

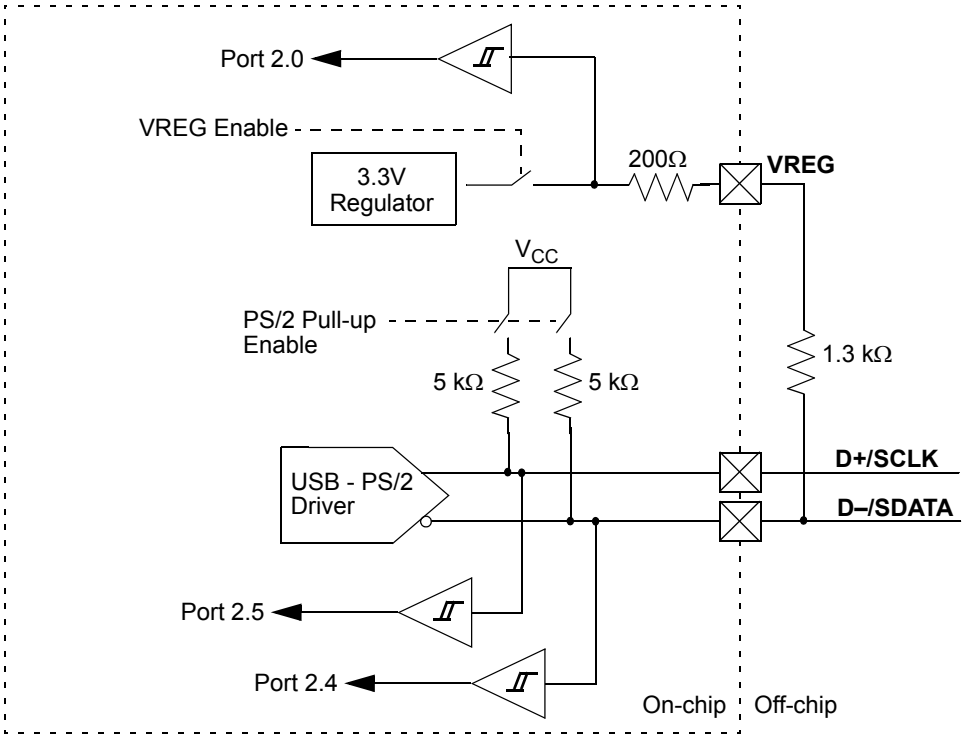
PS/2 Operation

The CY7C637xxC parts are optimized for combination USB or PS/2 devices, through the following features:

1. USB D+ and D– lines can also be used for PS/2 SCLK and SDATA pins, respectively. With USB disabled, these lines can be placed in a high-impedance state that will pull up to V_{CC}. (Disable USB by clearing the Address Enable bit of the USB Device Address Register, Figure 15).
2. An interrupt is provided to indicate a long LOW state on the SDATA pin. This eliminates the need to poll this pin to check for PS/2 activity. Refer to USB Port Status and Control on page 19 for more details.
3. Internal PS/2 pull-up resistors can be enabled on the SCLK and SDATA lines, so no GPIO pins are required for this task (bit 7, USB Status and Control Register, *Figure 14*).
4. The controlled slew rate outputs from these pins apply to both USB and PS/2 modes to minimize EMI.
5. The state of the SCLK and SDATA pins can be read, and can be individually driven LOW in an open drain mode. The pins are read at bits [5:4] of Port 2, and are driven with the Control Bits [2:0] of the USB Status and Control Register.
6. The V_{REG} pin can be placed into a high-impedance state, so that a USB pull-up resistor on the D–/SDATA pin will not interfere with PS/2 operation (bit 6, USB Status and Control Register).

The PS/2 on-chip support circuitry is illustrated in Figure 19.

Figure 19. Diagram of USB-PS/2 System Connections



Master SCK Selection

The Master's SCK is programmable to one of four clock settings, as shown in Figure 20. The frequency is selected with the Clock Select Bits of the SPI control register. The hardware provides 8 output clocks on the SCK pin (P0.7) for each byte transfer. Clock phase and polarity are selected by the CPHA and CPOL control bits (see Figure 20 and Figure 23).

The master SCK duty cycle is nominally 33% in the fastest (2 Mbps) mode, and 50% in all other modes.

Operation as an SPI Slave

In slave mode, the chip receives SCK from an external master on pin P0.7. Data from the master is shifted in on the MOSI pin (P0.5), while data is being shifted out of the slave on the MISO pin (P0.6). In addition, the active LOW Slave Select must be asserted to enable the slave for transmit. The Slave Select pin is P0.4. These pins must be configured in appropriate GPIO modes, with the GPIO data register set to 1 to enable bypass mode selected for the MISO pin.

In Slave mode, writes to the SPI Data Register load the Transmit buffer. If the Slave Select is asserted ($\overline{SS}$ LOW) and the shift register is not busy shifting a previous byte, the transmit buffer contents will be automatically transferred into the shift register. If the shift register is busy, the new byte will be loaded into the shift register only after the active byte has finished and is transferred to the receive buffer. The new byte is then ready to be shifted out (shifting waits for SCK from the Master). If the Slave Select is not active when the transmit buffer is loaded, data is not transferred to the shift register until Slave Select is asserted. The Transmit Buffer Full (TBF) bit will be set to '1' until the transmit buffer's data-byte is transferred to the shift register. Writing to the transmit buffer while the TBF bit is HIGH will overwrite the old byte in the Transmit Buffer.

If the Slave Select is deasserted before a byte transfer is complete, the transfer is aborted and no interrupt is generated. Whenever Slave Select is asserted, the transmit buffer is automatically reloaded into the shift register.

Clock phase and polarity must be selected to match the SPI master, using the CPHA and CPOL control bits (see Figure 22 and Figure 23).

The SPI slave logic continues to operate in suspend, so if the SPI interrupt is enabled, the device can go into suspend during a SPI slave transaction, and it will wake up at the interrupt that signals the end of the byte transfer.

SPI Status and Control

The SPI Control Register is shown in Figure 22. The timing diagram in Figure 23 shows the clock and data states for the various SPI modes.

Figure 22. SPI Control Register (Address 0x61)

Bit #	7	6	5	4	3	2	1	0
Bit Name	TCMP	TBF	Comm Mode[1:0]		CPOL	CPHA	SCK Select	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 7: TCMP

1 = TCMP is set to 1 by the hardware when 8-bit transfer is complete. The SPI interrupt is asserted at the same time TCMP is set to 1.

0 = This bit is only cleared by firmware.

Bit 6: TBF

Transmit Buffer Full bit.

1 = Indicates data in the transmit buffer has not transferred to the shift register.

0 = Indicates data in the transmit buffer has transferred to the shift register.

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

00 = All communications functions disabled (default).

01 = SPI Master Mode.

10 = SPI Slave Mode.

11 = Reserved.

Bit 3: CPOL

SPI Clock Polarity bit.

1 = SCK idles HIGH.

0 = SCK idles LOW.

Bit 2: CPHA

SPI Clock Phase bit (see Figure 23)

Bit [1:0]: SCK Select

Master mode SCK frequency selection (no effect in Slave Mode):

00 = 2 Mbit/s

01 = 1 Mbit/s

10 = 0.5 Mbit/s

11 = 0.0625 Mbit/s

12-bit Free-running Timer

The 12-bit timer operates with a 1-µs tick, provides two interrupts (128-µs and 1.024-ms) and allows the firmware to directly time events that are up to 4 ms in duration. The lower eight bits of the timer can be read directly by the firmware. Reading the lower eight bits latches the upper four bits into a temporary register. When the firmware reads the upper four bits of the timer, it is actually reading the count stored in the temporary register. The effect of this is to ensure a stable 12-bit timer value can be read, even when the two reads are separated in time.

Figure 24. Timer LSB Register (Address 0x24)

Bit #	7	6	5	4	3	2	1	0
Bit Name	Timer [7:0]							
Read/Write	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit [7:0]: Timer lower eight bits

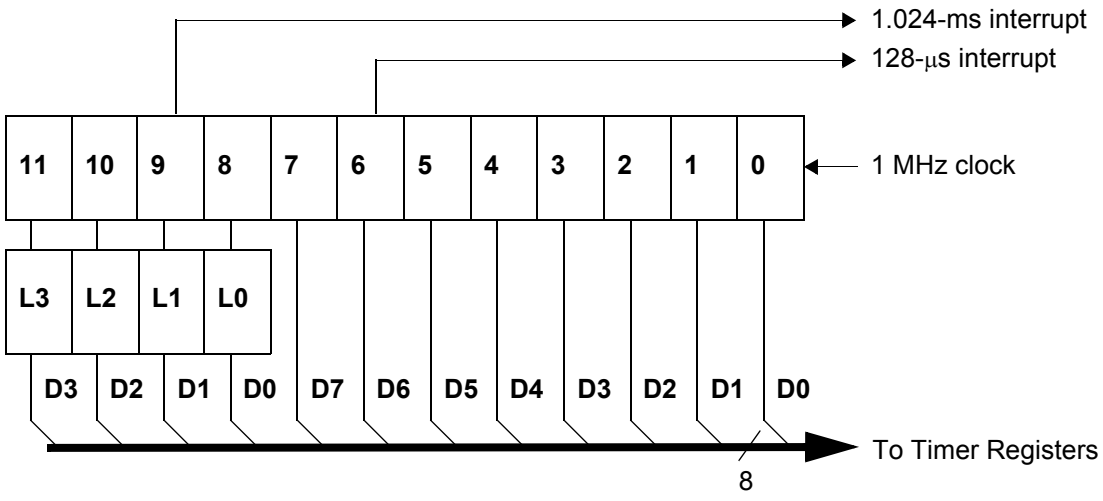
Figure 25. Timer MSB Register (Address 0x25)

Bit #	7	6	5	4	3	2	1	0
Bit Name	Reserved				Timer [11:8]			
Read/Write	-	-	-	-	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit [7:4]: Reserved

Bit [3:0]: Timer upper four bits

Figure 26. Timer Block Diagram



The four Capture Timer Data Registers are read-only, and are shown in Figure 28 through Figure 31.

Out of the 12-bit free running timer, the 8-bit captured in the Capture Timer Data Registers are determined by the Prescale Bit [2:0] in the Capture Timer Configuration Register (Figure 33).

Figure 28. Capture Timer A-Rising, Data Register (Address 0x40)

Bit #	7	6	5	4	3	2	1	0
Bit Name	Capture A Rising Data							
Read/Write	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Figure 29. Capture Timer A-Falling, Data Register (Address 0x41)

Bit #	7	6	5	4	3	2	1	0
Bit Name	Capture A Falling Data							
Read/Write	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Figure 30. Capture Timer B-Rising, Data Register (Address 0x42)

Bit #	7	6	5	4	3	2	1	0
Bit Name	Capture B Rising Data							
Read/Write	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Figure 31. Capture Timer B-Falling, Data Register (Address 0x43)

Bit #	7	6	5	4	3	2	1	0
Bit Name	Capture B Falling Data							
Read/Write	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Figure 32. Capture Timer Status Register (Address 0x45)

Bit #	7	6	5	4	3	2	1	0
Bit Name	Reserved				Capture B Falling Event	Capture B Rising Event	Capture A Falling Event	Capture A Rising Event
Read/ Write	-	-	-	-	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit [7:4]: Reserved.

Bit [3:0]: Capture A/B, Falling/Rising Event

These bits record the occurrence of any rising or falling edges on the capture GPIO pins. Bits in this register are cleared by reading the corresponding data register.

1 = A rising or falling event that matches the pin's rising/falling condition has occurred.

0 = No event that matches the pin's rising or falling edge condition.

Because both Capture A events (rising and falling) share an interrupt, user's firmware needs to check the status of both Capture A Falling and Rising Event bits to determine what caused the interrupt. This is also true for Capture B events.

Figure 33. Capture Timer Configuration Register (Address 0x44)

Bit #	7	6	5	4	3	2	1	0
Bit Name	First Edge Hold	Prescale Bit [2:0]			Capture B Falling Int Enable	Capture B Rising Int Enable	Capture A Falling Int Enable	Capture A Rising Int Enable
Read/ Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 7: First Edge Hold

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.

0 = The time of the most recent edge is held in the Capture Timer Data Register. That is, if multiple edges have occurred before reading the capture timer, the time for the last one will be read (default state).

The First Edge Hold function applies globally to all four capture timers.

Bit [6:4]: Prescale Bit [2:0]

Three prescaler bits allow the capture timer clock rate to be selected among 5 choices, as shown in Table 6 below.

Bit [3:0]: Capture A/B, Rising/Falling Interrupt Enable

Each of the four Capture Timer registers can be individually enabled to provide interrupts.

Both Capture A events share a common interrupt request, as do the two Capture B events. In addition to the event enables, the main Capture Interrupt Enables bit in the Global Interrupt Enable register (Interrupt Sources on page 32) must be set to activate a capture interrupt.

1 = Enable interrupt

0 = Disable interrupt

Table 6. Capture Timer Prescalar Settings (Step size and range for F_{CLK} = 6 MHz)

Prescale 2:0	Captured Bits	LSB Step Size	Range
000	Bits 7:0 of free-running timer	1 µs	256 µs
001	Bits 8:1 of free-running timer	2 µs	512 µs
010	Bits 9:2 of free-running timer	4 µs	1.024 ms
011	Bits 10:3 of free-running timer	8 µs	2.048 ms
100	Bits 11:4 of free-running timer	16 µs	4.096 ms

Table 6. Capture Timer Prescalar Settings (Step size and range for F_{CLK} = 6 MHz)

Processor Status and Control Register

Figure 34. Processor Status and Control Register (Address 0xFF)

Bit #	7	6	5	4	3	2	1	0
Bit Name	IRQ Pending	Watchdog Reset	Bus Interrupt Event	LVR/BOR Reset	Suspend	Interrupt Enable Sense	Reserved	Run
Read/Write	R	R/W	R/W	R/W	R/W	R	-	R/W
Reset	0	1	0	1	0	0	0	1

Bit 7: IRQ Pending

When an interrupt is generated, it is registered as a pending interrupt. The interrupt will remain pending until its interrupt enable bit is set (Figure 35 and Figure 36) and interrupts are globally enabled (Bit 2, Processor Status and Control Register). At that point the internal interrupt handling sequence will clear the IRQ Pending bit until another interrupt is detected as pending. This bit is only valid if the Global Interrupt Enable bit is disabled.

1 = There are pending interrupts.

0 = No pending interrupts.

Bit 6: Watchdog Reset

The Watchdog Timer Reset (WDR) occurs when the internal Watchdog timer rolls over. The timer will roll over and WDR will occur if it is not cleared within t_{WATCH} (see Switching Characteristics on page 44 for the value of t_{WATCH}). This bit is cleared by an LVR/BOR. Note that a Watchdog reset can occur with a POR/LVR/BOR event, as discussed at the end of this section.

1 = A Watchdog reset occurs.

0 = No Watchdog reset

Bit 5: Bus Interrupt Event

The Bus Reset Status is set whenever the event for the USB Bus Reset or PS/2 Activity interrupt occurs. The event type (USB or PS/2) is selected by the state of the USB-PS/2 Interrupt Mode bit in the USB Status and Control Register (see Figure 14). The details on the event conditions that set this bit are given in Interrupt Sources on page 32. In either mode, this bit is set as soon as the event has lasted for 128–256 µs, and the bit will be set even if the interrupt is not enabled. The bit is only cleared by firmware or LVR/WDR.

1 = A USB reset occurred or PS/2 Activity is detected, depending on USB-PS/2 Interrupt Select bit.

0 = No event detected since last cleared by firmware or LVR/WDR.

Bit 4: LVR/BOR Reset

The Low-voltage or Brown-out Reset is set to ‘1’ during a power-on reset. Firmware can check bits 4 and 6 in the reset handler to determine whether a reset was caused by a LVR/BOR condition or a Watchdog timeout. This bit is not affected by WDR. Note that a LVR/BOR event may be followed by a Watchdog reset before firmware begins executing, as explained at the end of this section.

1 = A POR or LVR has occurred.

0 = No POR nor LVR since this bit last cleared.

Bit 3: Suspend

Writing a '1' to the Suspend bit will halt the processor and cause the microcontroller to enter the suspend mode that significantly reduces power consumption. An interrupt or USB bus activity will cause the device to come out of suspend. After coming out of suspend, the device will resume firmware execution at the instruction following the IOWR which put the part into suspend. When writing the suspend bit with a resume condition present (such as non-idle USB activity), the suspend state will still be entered, followed immediately by the wake-up process (with appropriate delays for the clock start-up). See Suspend Mode on page 15 for more details on suspend mode operation.

1 = Suspend the processor.

0 = Not in suspend mode. Cleared by the hardware when resuming from suspend.

Bit 2: Interrupt Enable Sense

This bit shows whether interrupts are enabled or disabled. Firmware has no direct control over this bit as writing a zero or one to this bit position will have no effect on interrupts. This bit is further gated with the bit settings of the Global Interrupt Enable Register (Figure 35) and USB Endpoint Interrupt Enable Register (Figure 36). Instructions DI, EI, and RETI manipulate the state of this bit.

1 = Interrupts are enabled.

0 = Interrupts are masked off.

Bit 1: Reserved. Must be written as a 0.

Bit 0: Run

This bit is manipulated by the HALT instruction. When Halt is executed, the processor clears the run bit and halts at the end of the current instruction. The processor remains halted until a reset occurs (low-voltage, brown-out, or Watchdog). This bit should normally be written as a '1'.

During power-up, or during a low-voltage reset, the Processor Status and Control Register is set to 00010001, which indicates a LVR/BOR (bit 4 set) has occurred and no interrupts are pending (bit 7 clear). Note that during the t_{START} ms partial suspend at start-up (explained in Reset on page 13), a Watchdog Reset will also occur. When a WDR occurs during the power-up suspend interval, firmware would read 01010001 from the Status and Control Register after power-up. Normally the LVR/BOR bit should be cleared so that a subsequent WDR can be clearly identified. *Note that if a USB bus reset (long SE0) is received before firmware examines this register, the Bus Interrupt Event bit would also be set.*

During a Watchdog Reset, the Processor Status and Control Register is set to 01XX0001, which indicates a Watchdog Reset (bit 4 set) has occurred and no interrupts are pending (bit 7 clear).

Interrupts

Interrupts can be generated by the GPIO lines, the internal free-running timer, the SPI block, the capture timers, on various USB events, PS/2 activity, or by the wake-up timer. All interrupts are maskable by the Global Interrupt Enable Register and the USB End Point Interrupt Enable Register. Writing a '1' to a bit position enables the interrupt associated with that bit position. During a reset, the contents of the interrupt enable registers are cleared, along with the Global Interrupt enable bit of the CPU, effectively disabling all interrupts.

The interrupt controller contains a separate flip-flop for each interrupt. See Figure 37 for the logic block diagram of the interrupt controller. When an interrupt is generated it is first registered as a pending interrupt. It will stay pending until it is serviced or a reset occurs. A pending interrupt will only generate an interrupt request if it is enabled by the corresponding bit in the interrupt enable registers. The highest priority interrupt request will be serviced following the completion of the currently executing instruction.

When servicing an interrupt, the hardware will first disable all interrupts by clearing the Global Interrupt Enable bit in the CPU (the state of this bit can be read at Bit 2 of the Processor Status and Control Register). Next, the flip-flop of the current interrupt is cleared. This is followed by an automatic CALL instruction to the ROM address associated with the interrupt being serviced (i.e., the Interrupt Vector, see Interrupt Vectors on page 31). The instruction in the interrupt table is typically a JMP instruction to the address of the Interrupt Service Routine (ISR). The user can re-enable interrupts in the interrupt service routine by executing an EI instruction. Interrupts can be nested to a level limited only by the available stack space.

The Program Counter value as well as the Carry and Zero flags (CF, ZF) are stored onto the Program Stack by the automatic CALL instruction generated as part of the interrupt acknowledge process. The user firmware is responsible for ensuring that the processor state is preserved and restored during an interrupt. The PUSH A instruction should typically be used as the first command in the ISR to save the accumulator value and the POP A instruction should be used just before the RETI instruction to restore the accumulator value. The program counter, CF and ZF are restored and interrupts are enabled when the RETI instruction is executed.

The DI and EI instructions can be used to disable and enable interrupts, respectively. These instructions affect only the Global Interrupt Enable bit of the CPU. If desired, EI can be used to re-enable interrupts while inside an ISR, instead of waiting for the RETI that exits the ISR. While the global interrupt enable bit is cleared, the presence of a pending interrupt can be detected by examining the IRQ Sense bit (Bit 7 in the Processor Status and Control Register).

Interrupt Vectors

The Interrupt Vectors supported by the device are listed in Table 7. The highest priority interrupt is #1 (USB Bus Reset / PS/2 activity), and the lowest priority interrupt is #11 (Wake-up Timer). Although Reset is not an interrupt, the first instruction executed after a reset is at ROM address 0x0000, which corresponds to the first entry in the Interrupt Vector Table. Interrupt vectors occupy two bytes to allow for a two-byte JMP instruction to the appropriate Interrupt Service Routine (ISR).

Table 7. Interrupt Vector Assignments

Interrupt Vector No.	ROM Address	Function
not applicable	0x0000	Execution after Reset begins here
1	0x0002	USB Bus Reset or PS/2 Activity interrupt
2	0x0004	128-µs timer interrupt
3	0x0006	1.024-ms timer interrupt
4	0x0008	USB Endpoint 0 interrupt
5	0x000A	USB Endpoint 1 interrupt
6	0x000C	USB Endpoint 2 interrupt
7	0x000E	SPI Interrupt
8	0x0010	Capture Timer A interrupt

Table 7. Interrupt Vector Assignments

9	0x0012	Capture Timer B interrupt
10	0x0014	GPIO interrupt
11	0x0016	Wake-up Timer interrupt

Interrupt Latency

Interrupt latency can be calculated from the following equation:

Interrupt Latency = (Number of clock cycles remaining in the current instruction)
+ (10 clock cycles for the CALL instruction)
+ (5 clock cycles for the JMP instruction)

For example, if a 5 clock cycle instruction such as JC is being executed when an interrupt occurs, the first instruction of the Interrupt Service Routine will execute a minimum of 16 clocks (1+10+5) or a maximum of 20 clocks (5+10+5) after the interrupt is issued. With a 6-MHz external resonator, internal CPU clock speed is 12 MHz, so 20 clocks take 20/12 MHz = 1.67 µs.

Interrupt Sources

The following sections provide details on the different types of interrupt sources.

Figure 35. Global Interrupt Enable Register (Address 0x20)

Bit #	7	6	5	4	3	2	1	0
Bit Name	Wake-up Interrupt Enable	GPIO Interrupt Enable	Capture Timer B Intr. Enable	Capture Timer A Intr. Enable	SPI Interrupt Enable	1.024-ms Interrupt Enable	128-µs Interrupt Enable	USB Bus Reset / PS/2 Activity Intr. Enable
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 7: Wake-up Interrupt Enable

The internal wake-up timer is normally used to wake the part from suspend mode, but it can also provide an interrupt when the part is awake. The wake-up timer is cleared whenever the Wake-up Interrupt Enable bit is written to a 0, and runs whenever that bit is written to a 1. When the interrupt is enabled, the wake-up timer provides periodic interrupts at multiples of period, as described in Wake-up Timer on page 16.

1 = Enable wake-up timer for periodic wake-up.

0 = Disable and power-off wake-up timer.

Bit 6: GPIO Interrupt Enable

Each GPIO pin can serve as an interrupt input. During a reset, GPIO interrupts are disabled by clearing all GPIO interrupt enable registers. Writing a '1' to a GPIO Interrupt Enable bit enables GPIO interrupts from the corresponding input pin. These registers are shown in Figure 38 for Port 0 and Figure 39 for Port 1. In addition to enabling the desired individual pins for interrupt, the main GPIO interrupt must be enabled, as explained in General Purpose I/O Ports on page 17.

The polarity that triggers an interrupt is controlled independently for each GPIO pin by the GPIO Interrupt Polarity Registers. Setting a Polarity bit to '0' allows an interrupt on a falling GPIO edge, while setting a Polarity bit to '1' allows an interrupt on a rising GPIO edge. The Polarity Registers reset to 0 and are shown in Figure 38 for Port 0 and Figure 40 for Port 1.

All of the GPIO pins share a single interrupt vector, which means the firmware will need to read the GPIO ports with enabled interrupts to determine which pin or pins caused an interrupt.The GPIO interrupt structure is illustrated in Figure 41.

Note that if one port pin triggered an interrupt, no other port pins can cause a GPIO interrupt until that port pin has returned to its inactive (non-trigger) state or its corresponding port interrupt enable bit is cleared. The CY7C637xxC does not assign interrupt priority to different port pins and the Port Interrupt Enable Registers are not affected by the interrupt acknowledge process.

1 = Enable

0 = Disable

Bit [5:4]: Capture Timer A and B Interrupts

There are two capture timer interrupts, one for each associated pin. Each of these interrupts occurs on an enabled edge of the selected GPIO pin(s). For each pin, rising and/or falling edge capture interrupts can be in selected. Refer to Timer Capture Registers on page 28. These interrupts are independent of the GPIO interrupt, described in the next section.

1 = Enable

0 = Disable

Bit 3: SPI Interrupt Enable

The SPI interrupt occurs at the end of each SPI byte transaction, at the final clock edge, as shown in Figure 23. After the interrupt, the received data byte can be read from the SPI Data Register, and the TCMP control bit will be high

1 = Enable

0 = Disable

Bit 2: 1.024-ms Interrupt Enable

The 1.024-ms interrupts are periodic timer interrupts from the free-running timer (based on the 6-MHz clock). The user should disable this interrupt before going into the suspend

mode to avoid possible conflicts between servicing the timer interrupts (128-µs interrupt and 1.024-ms interrupt) first or the suspend request first when waking up.

1 = Enable. Periodic interrupts will be generated approximately every 1.024 ms.

0 = Disable.

Bit 1: 128-µs Interrupt Enable

The 128-µs interrupt is another source of timer interrupt from the free-running timer. The user should disable both timer interrupts (128-µs and 1.024-ms) before going into the suspend mode to avoid possible conflicts between servicing the timer interrupts first or the suspend request first when waking up.

1 = Enable. Periodic interrupts will be generated approximately every 128 µs.

0 = Disable.

Bit 0: USB Bus Reset - PS/2 Interrupt Enable

The function of this interrupt is selectable between detection of either a USB bus reset condition, or PS/2 activity. The selection is made with the USB-PS/2 Interrupt Mode bit in the USB Status and Control Register (Figure 14). In either case, the interrupt will occur if the selected condition exists for 256 µs, and may occur as early as 128 µs.

A USB bus reset is indicated by a single ended zero (SE0) on the USB D+ and D– pins. The USB Bus Reset interrupt occurs when the SE0 condition ends. PS/2 activity is indicated by a continuous LOW on the SDATA pin. The PS/2 interrupt occurs as soon as the long LOW state is detected.

During the entire interval of a USB Bus Reset or PS/2 interrupt event, the USB Device Address register is cleared.

The Bus Reset/PS/2 interrupt may occur 128 µs after the bus condition is removed.

1 = Enable

0 = Disable

Figure 36. Endpoint Interrupt Enable Register (Address 0x21)

Bit #	7	6	5	4	3	2	1	0
Bit Name	Reserved					EP2 Interrupt Enable	EP1 Interrupt Enable	EP0 Interrupt Enable
Read/Write	-	-	-	-	-	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit [7:3]: Reserved.

Bit [2:1]: EP2,1 Interrupt Enable

There are two non-control endpoint (EP2 and EP1) interrupts. If enabled, a non-control endpoint interrupt is generated when:

- The USB host writes valid data to an endpoint FIFO. However, if the endpoint is in ACK OUT modes, an interrupt is generated regardless of data packet validity (i.e., good CRC). Firmware must check for data validity.
- The device SIE sends a NAK or STALL handshake packet to the USB host during the host attempts to read data from the endpoint (INs).
- The device receives an ACK handshake after a successful read transaction (IN) from the host.
- The device SIE sends a NAK or STALL handshake packet to the USB host during the host attempts to write data (OUTs) to the endpoint FIFO.

1 = Enable

0 = Disable

Refer to Table 8 for more information.

Bit 0: EP0 Interrupt Enable

If enabled, a control endpoint interrupt is generated when:

- The endpoint 0 mode is set to accept a SETUP token.
- After the SIE sends a 0-byte packet in the status stage of a control transfer.
- The USB host writes valid data to an endpoint FIFO. However, if the endpoint is in ACK OUT modes, an interrupt is generated regardless of what data is received. Firmware must check for data validity.
- The device SIE sends a NAK or STALL handshake packet to the USB host during the host attempts to read data from the endpoint (INs).
- The device SIE sends a NAK or STALL handshake packet to the USB host during the host attempts to write data (OUTs) to the endpoint FIFO.

1 = Enable EP0 interrupt

0 = Disable EP0 interrupt

Figure 40. Port 1 Interrupt Polarity Register (Address 0x07)

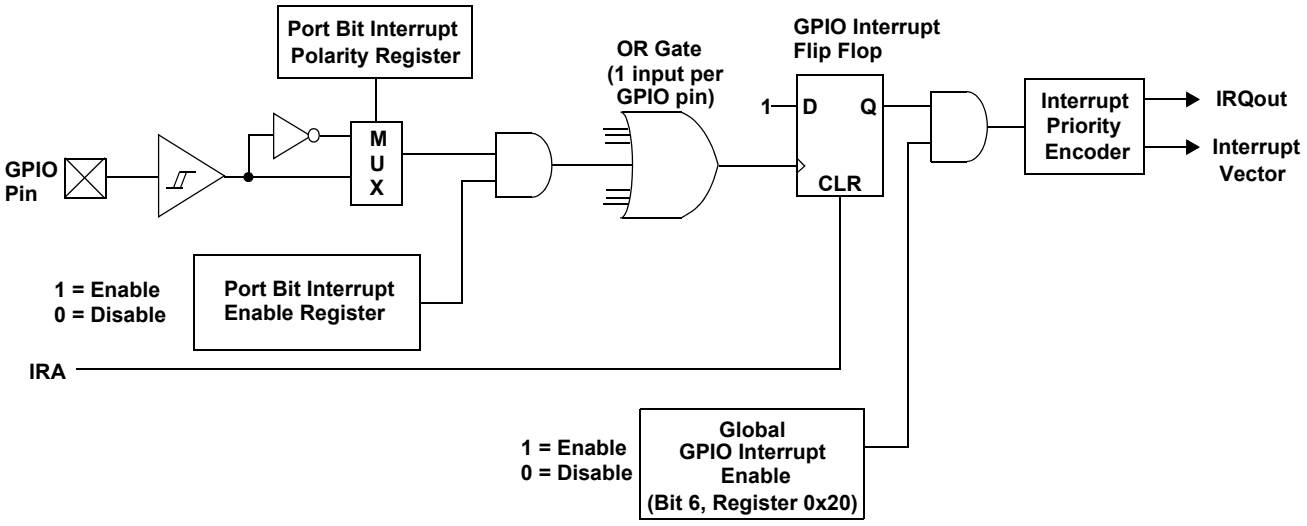
Bit #	7	6	5	4	3	2	1	0
Bit Name	P1 Interrupt Polarity							
Read/Write	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit [7:0]: P1[7:0] Interrupt Polarity

1 = Rising GPIO edge

0 = Falling GPIO edge

Figure 41. GPIO Interrupt Diagram



USB Mode Tables

The following tables give details on mode setting for the USB Serial Interface Engine (SIE) for both the control endpoint (EP0) and non-control endpoints (EP1 and EP2).

Table 8. USB Register Mode Encoding for Control and Non-Control Endpoints

Mode	Encoding	SETUP	IN	OUT	Comments
Disable	**0000**	Ignore	Ignore	Ignore	Ignore all USB traffic to this endpoint
NAK IN/OUT	**0001**	Accept	NAK	NAK	On Control endpoint, after successfully sending an ACK handshake to a SETUP packet, the SIE forces the endpoint mode (from modes other than 0000) to 0001. The mode is also changed by the SIE to 0001 from mode 1011 on issuance of ACK handshake to an OUT.
Status OUT Only	**0010**	Accept	STALL	Check	For Control endpoints
STALL IN/OUT	**0011**	Accept	STALL	STALL	For Control endpoints
Ignore IN/OUT	**0100**	Accept	Ignore	Ignore	For Control endpoints
Reserved	**0101**	Ignore	Ignore	Always	Reserved
Status IN Only	**0110**	Accept	TX 0 Byte	STALL	For Control Endpoints
Reserved	**0111**	Ignore	TX Count	Ignore	Reserved
NAK OUT	**1000**	Ignore	Ignore	NAK	In mode 1001, after sending an ACK handshake to an OUT, the SIE changes the mode to 1000
ACK OUT(STALL[4]=0) ACK OUT(STALL[4]=1)	**1001** **1001**	Ignore Ignore	Ignore Ignore	ACK STALL	This mode is changed by the SIE to mode 1000 on issuance of ACK handshake to an OUT
NAK OUT - Status IN	**1010**	Accept	TX 0 Byte	NAK	
ACK OUT - NAK IN	**1011**	Accept	NAK	ACK	This mode is changed by the SIE to mode 0001 on issuance of ACK handshake to an OUT
NAK IN	**1100**	Ignore	NAK	Ignore	An ACK from mode 1101 changes the mode to 1100
ACK IN(STALL[4]=0) ACK IN(STALL[4]=1)	**1101** **1101**	Ignore Ignore	TX Count STALL	Ignore Ignore	This mode is changed by the SIE to mode 1100 on issuance of ACK handshake to an IN
NAK IN - Status OUT	**1110**	Accept	NAK	Check	An ACK from mode 1111 changes the mode to 1110
ACK IN - Status OUT	**1111**	Accept	TX Count	Check	This mode is changed by the SIE to mode 1110 on issuance of ACK handshake to an IN

Note
4. STALL bit is the bit 7 of the USB Non-Control Device Endpoint Mode registers. Refer to USB Non-control Endpoints on page 22 for more explanation.

Mode Column:

The 'Mode' column contains the mnemonic names given to the modes of the endpoint. The mode of the endpoint is determined by the four-bit binaries in the 'Encoding' column as discussed below. The Status IN and Status OUT modes represent the status IN or OUT stage of the control transfer.

Encoding Column:

The contents of the 'Encoding' column represent the Mode Bits [3:0] of the Endpoint Mode Registers (Figure 16 and Figure 17). The endpoint modes determine how the SIE responds to different tokens that the host sends to the endpoints. For example, if the Mode Bits [3:0] of the Endpoint 0 Mode Register (Figure 16) are set to '0001', which is NAK IN/OUT mode as shown in Table 8 above, the SIE of the part will send an ACK handshake in response to SETUP tokens and NAK any IN or OUT tokens. For more information on the functionality of the Serial Interface Engine (SIE), see USB Serial Interface Engine (SIE) on page 19.

SETUP, IN, and OUT Columns:

Depending on the mode specified in the 'Encoding' column, the 'SETUP', 'IN', and 'OUT' columns contain the device SIE's responses when the endpoint receives SETUP, IN, and OUT tokens respectively.

A 'Check' in the Out column means that upon receiving an OUT token the SIE checks to see whether the OUT is of zero length and has a Data Toggle (Data1/0) of 1. If these conditions are true, the SIE responds with an ACK. If any of the above conditions is not met, the SIE will respond with either a STALL or Ignore. Table 10 gives a detailed analysis of all possible cases.

A 'TX Count' entry in the IN column means that the SIE will transmit the number of bytes specified in the Byte Count Bit [3:0] of the Endpoint Count Register (Figure 18) in response to any IN token.

A 'TX 0 Byte' entry in the IN column means that the SIE will transmit a zero byte packet in response to any IN sent to the endpoint. Sending a 0 byte packet is to complete the status stage of a control transfer.

An 'Ignore' means that the device sends no handshake tokens.

An 'Accept' means that the SIE will respond with an ACK to a valid SETUP transaction.

Comments Column:

Some Mode Bits are automatically changed by the SIE in response to many USB transactions. For example, if the Mode

Table 10. Details of Modes for Differing Traffic Conditions (continued)

1	0	1	1	IN	x	UC	x	UC	UC	UC	UC	1	UC	UC	No-Change	NAK	yes
NAK OUT/Status IN																	
1	0	1	0	OUT	<= 10	UC	valid	UC	UC	UC	UC	UC	1	UC	No-Change	NAK	yes
1	0	1	0	OUT	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	No-Change	Ignore	no
1	0	1	0	OUT	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	No-Change	Ignore	no
1	0	1	0	IN	x	UC	x	UC	UC	UC	UC	1	UC	1	No-Change	TX 0 Byte	yes
Status IN Only																	
0	1	1	0	OUT	<= 10	UC	valid	UC	UC	UC	UC	UC	1	UC	0	0	1	1	STALL	yes
0	1	1	0	OUT	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	No-Change	Ignore	no
0	1	1	0	OUT	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	No-Change	Ignore	no
0	1	1	0	IN	x	UC	x	UC	UC	UC	UC	1	UC	1	No-Change	TX 0 Byte	yes
Control Read																	
ACK IN/Status OUT																	
1	1	1	1	OUT	2	UC	valid	1	1	updates	UC	UC	1	1	No-Change	ACK	yes
1	1	1	1	OUT	2	UC	valid	0	1	updates	UC	UC	1	UC	0	0	1	1	STALL	yes
1	1	1	1	OUT	!=2	UC	valid	updates	1	updates	UC	UC	1	UC	0	0	1	1	STALL	yes
1	1	1	1	OUT	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	No-Change	Ignore	no
1	1	1	1	OUT	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	No-Change	Ignore	no
1	1	1	1	IN	x	UC	x	UC	UC	UC	UC	1	UC	1	1	1	1	0	ACK (back)	yes
NAK IN/Status OUT																	
1	1	1	0	OUT	2	UC	valid	1	1	updates	UC	UC	1	1	No-Change	ACK	yes
1	1	1	0	OUT	2	UC	valid	0	1	updates	UC	UC	1	UC	0	0	1	1	STALL	yes
3	2	1	0	token	count	buffer	dval	DTOG	DVAL	COUNT	SETUP	IN	OUT	ACK	3	2	1	0	response	int
1	1	1	0	OUT	!=2	UC	valid	updates	1	updates	UC	UC	1	UC	0	0	1	1	STALL	yes
1	1	1	0	OUT	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	No-Change	Ignore	no
1	1	1	0	OUT	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	No-Change	Ignore	no
1	1	1	0	IN	x	UC	x	UC	UC	UC	UC	1	UC	UC	No-Change	NAK	yes
Status OUT Only																	
0	0	1	0	OUT	2	UC	valid	1	1	updates	UC	UC	1	1	No-Change	ACK	yes
0	0	1	0	OUT	2	UC	valid	0	1	updates	UC	UC	1	UC	0	0	1	1	STALL	yes
0	0	1	0	OUT	!=2	UC	valid	updates	1	updates	UC	UC	1	UC	0	0	1	1	STALL	yes
0	0	1	0	OUT	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	No-Change	Ignore	no
0	0	1	0	OUT	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	No-Change	Ignore	no
0	0	1	0	IN	x	UC	x	UC	UC	UC	UC	1	UC	UC	0	0	1	1	STALL	yes

Register Summary

	Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Read/Write/ Both/	Default/ Reset
GPIO CONFIGURATION PORTS 0, 1 AND 2	0x00	Port 0 Data	P0								BBBBBBBB	00000000
	0x01	Port 1 Data	P1								BBBBBBBB	00000000
	0x02	Port 2 Data	Reserved		D+(SCLK) State	D- (SDATA) State	Reserved		P2.1 (Int Clk Mode Only	VREG Pin State	--RR--RR	00000000
	0x0A	GPIO Port 0 Mode 0	P0[7:0] Mode0								WWWWWWWW	00000000
	0x0B	GPIO Port 0 Mode 1	P0[7:0] Mode1								WWWWWWWW	00000000
	0x0C	GPIO Port 1 Mode 0	P1[7:0] Mode0								WWWWWWWW	00000000
	0x0D	GPIO Port 1 Mode 1	P1[7:0] Mode1								WWWWWWWW	00000000
	0x04	Port 0 Interrupt Enable	P0[7:0] Interrupt Enable								WWWWWWWW	00000000
	0x05	Port 1 Interrupt Enable	P1[7:0] Interrupt Enable								WWWWWWWW	00000000
	0x06	Port 0 Interrupt Polarity	P0[7:0] Interrupt Polarity								WWWWWWWW	00000000
	0x07	Port 1 Interrupt Polarity	P1[7:0] Interrupt Polarity								WWWWWWWW	00000000
Clock Config.	0xF8	Clock Configuration	Ext. Clock Resume Delay	Wake-up Timer Adjust Bit [2:0]			Low-voltage Reset Disable	Precision USB Clocking Enable	Internal Clock Output Disable	External Oscillator Enable	BBBBBBBB	00000000
ENDPOINT 0, I AND 2 CONFIGURATION	0x10	USB Device Address	Device Address Enable	Device Address							BBBBBBBB	00000000
	0x12	EP0 Mode	SETUP Received	IN Received	OUT Received	ACKed Transaction	Mode Bit				BBBBBBBB	00000000
	0x14, 0x16	EP1, EP2 Mode Register	STALL	Reserved		ACKed Transaction	Mode Bit				B--BBBBB	00000000
	0x11, 0x13, and 0x15	EP0,1, and 2 Counter	Data 0/1 Toggle	Data Valid	Reserved		Byte Count				BB--BBBB	00000000
USB-SC	0x1F	USB Status and Control	PS/2 Pull-up Enable	VREG Enable	USB Reset-PS/2 Activity Interrupt Mode	Reserved	USB Bus Activity	D+/D- Forcing Bit			BBB-BBBB	00000000
INTERRUPT	0x20	Global Interrupt Enable	Wake-up Interrupt Enable	GPIO Interrupt Enable	Capture Timer B Intr. Enable	Capture Timer A Intr. Enable	SPI Interrupt Enable	1.024 ms Interrupt Enable	128 µs Interrupt Enable	USB Bus Reset-PS/2 Activity Intr. Enable	BBBBBBBB	00000000
	0x21	Endpoint Interrupt Enable	Reserved					EP2 Interrupt Enable	EP1 Interrupt Enable	EP0 Interrupt Enable	-----BBB	00000000
TIMER	0x24	Timer LSB	Timer Bit [7:0]								RRRRRRRR	00000000
	0x25	Timer (MSB)	Reserved				Timer Bit [11:8]				----RRRR	00000000
SPI	0x60	SPI Data	Data I/O								BBBBBBBB	00000000
	0x61	SPI Control	TCMP	TBF	Comm Mode [1:0]		CPOL	CPHA	SCK Select		BBBBBBBB	00000000
CAPTURE TIMER	0x40	Capture Timer A-Rising, Data Register	Capture A Rising Data								RRRRRRRR	00000000
	0x41	Capture Timer A-Falling, Data Register	Capture A Falling Data								RRRRRRRR	00000000
	0x42	Capture Timer B-Rising, Data Register	Capture B Rising Data								RRRRRRRR	00000000
	0x43	Capture Timer B-Falling, Data Register	Capture B Falling Data								RRRRRRRR	00000000
	0x44	Capture Timer Configuration	First Edge Hold	Prescale Bit [2:0]			Capture B Falling Intr Enable	Capture B Rising Intr Enable	Capture A Falling Intr Enable	Capture A Rising Intr Enable	BBBBBBBB	00000000
	0x45	Capture Timer Status	Reserved				Capture B Falling Event	Capture B Rising Event	Capture A Falling Event	Capture A Rising Event	----BBBB	00000000
PROC SC.	0xFF	Process Status & Control	IRQ Pending	Watch Dog Reset	Bus Interrupt Event	LVR/BOR Reset	Suspend	Interrupt Enable Sense	Reserved	Run	RBBBBR-B	See Processor Status and Control Register

Figure 47. SPI Master Timing, CPHA = 0

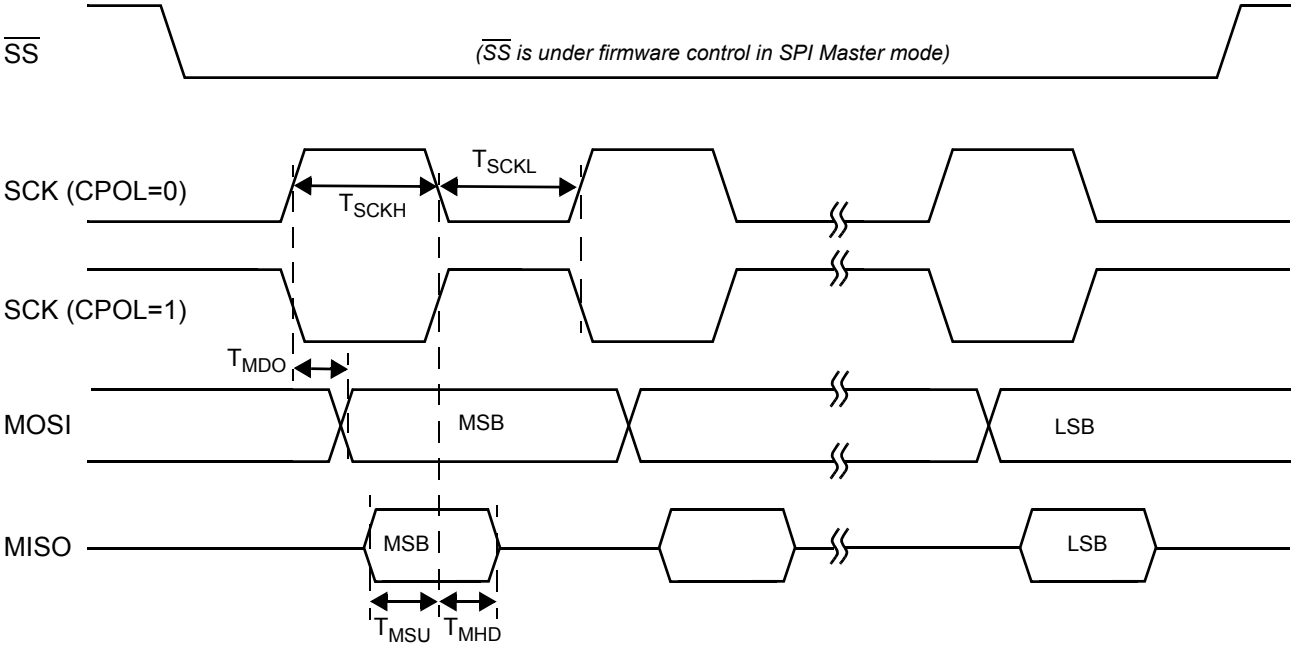


Figure 48. SPI Slave Timing, CPHA = 0

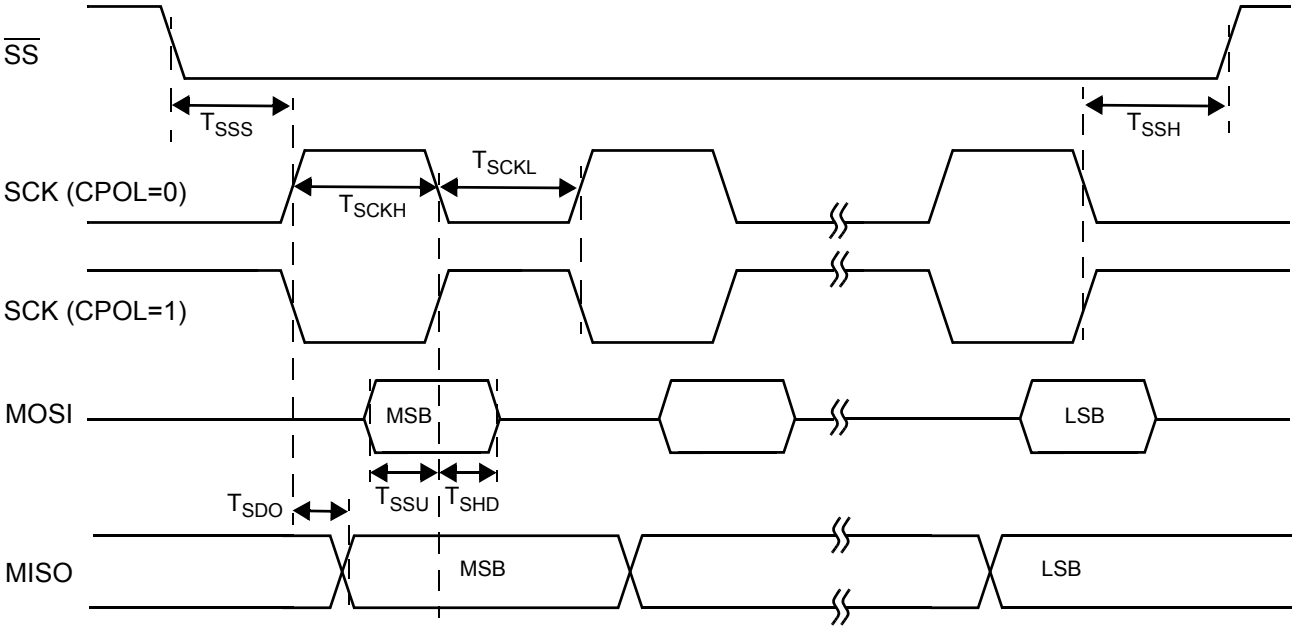


Figure 56. Die Form

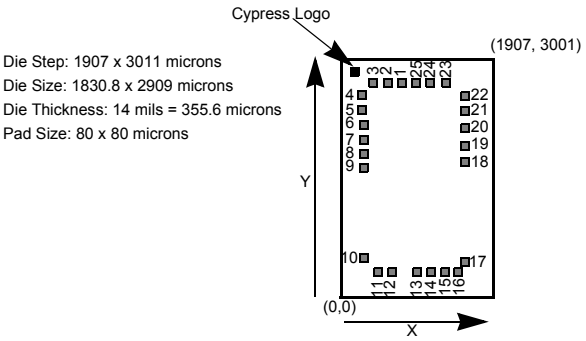


Table 11 below shows the die pad coordinates for the CY7C63722C-XC. The center location of each bond pad is relative to the bottom left corner of the die which has coordinate (0,0).

Table 11. CY7C63722C-XC Probe Pad Coordinates in microns ((0,0) to bond pad centers)

Pad Number	Pin Name	X (microns)	Y (microns)
1	P0.0	788.95	2843.15
2	P0.1	597.45	2843.15
3	P0.2	406.00	2843.15
4	P0.3	154.95	2687.95
5	P1.0	154.95	2496.45
6	P1.2	154.95	2305.05
7	P1.4	154.95	2113.60
8	P1.6	154.95	1922.05
9	Vss	154.95	1730.90
10	Vss	154.95	312.50
11	Vpp	363.90	184.85
12	VREG	531.70	184.85
13	XTALIN	1066.55	184.85
14	XTALOUT	1210.75	184.85
15	Vcc	1449.75	184.85
16	D–	1662.35	184.85
17	D+	1735.35	289.85
18	P1.7	1752.05	1832.75
19	P1.5	1752.05	2024.30
20	P1.3	1752.05	2215.75
21	P1.1	1752.05	2407.15
22	P0.7	1752.05	2598.65
23	P0.6	1393.25	2843.15
24	P0.5	1171.80	2843.15
25	P0.4	980.35	2843.15

Errata

This section describes the errata for the enCoRe™ USB Combination Low-speed USB & PS/2 Periheral Controller / CY7C637xx. The details include errata trigger conditions, available workaround, and silicon revision applicability.

Please contact your local Cypress Sales Representative if you have further questions.

Part Numbers Affected

Part Number	Device Characteristics
CY7C63722	All packages
CY7C63723	All packages
CY7C63743	All packages

enCoRe™ USB Combination Low-speed USB & PS/2 Peripheral Controller Qualification Status

Product status: In Production - Qual report: 001406

enCoRe™ USB Combination Low-speed USB & PS/2 Peripheral Controller Errata Summary

The following table defines the errata applicability to available enCoRe™ USB Combination Low-speed USB & PS/2 Peripheral Controller family devices. An "X" indicates that the errata pertains to the selected device.

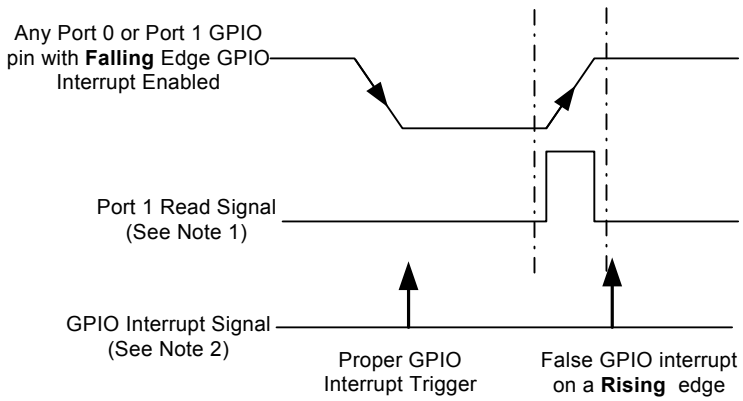
Note: Errata titles are hyperlinked. Click on table entry to jump to description.

Items	CY7C637xx	Rev Letter	Fix Status
1. Faulty GPIO Interrupt	X	A	No silicon fix planned.

1. Faulty GPIO Interrupt

■ **Problem Definition**

When a falling edge interrupt is enabled for a GPIO pin, reading the GPIO Port 1 coincident to a rising edge of that GPIO signal may generate a false GPIO interrupt.



When a rising edge interrupt is enabled for a GPIO pin, reading the GPIO Port 1 coincident to a falling edge of that GPIO signal may generated a false GPIO intterupt.

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

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6

Cypress Developer Community

Forums | WICED IOT Forums | Projects | Video | Blogs | Training | Components

Technical Support

cypress.com/support