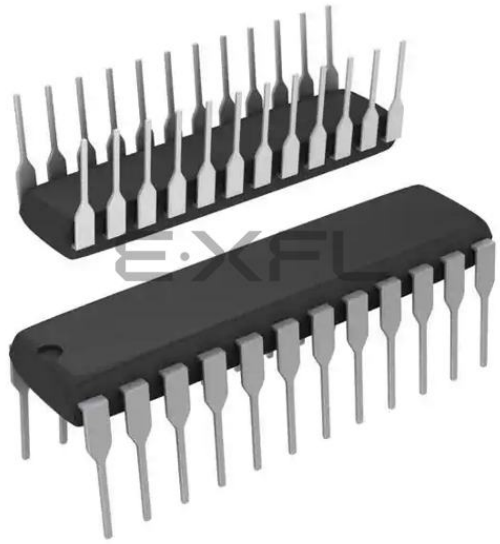


Welcome to E-XFL.COM

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

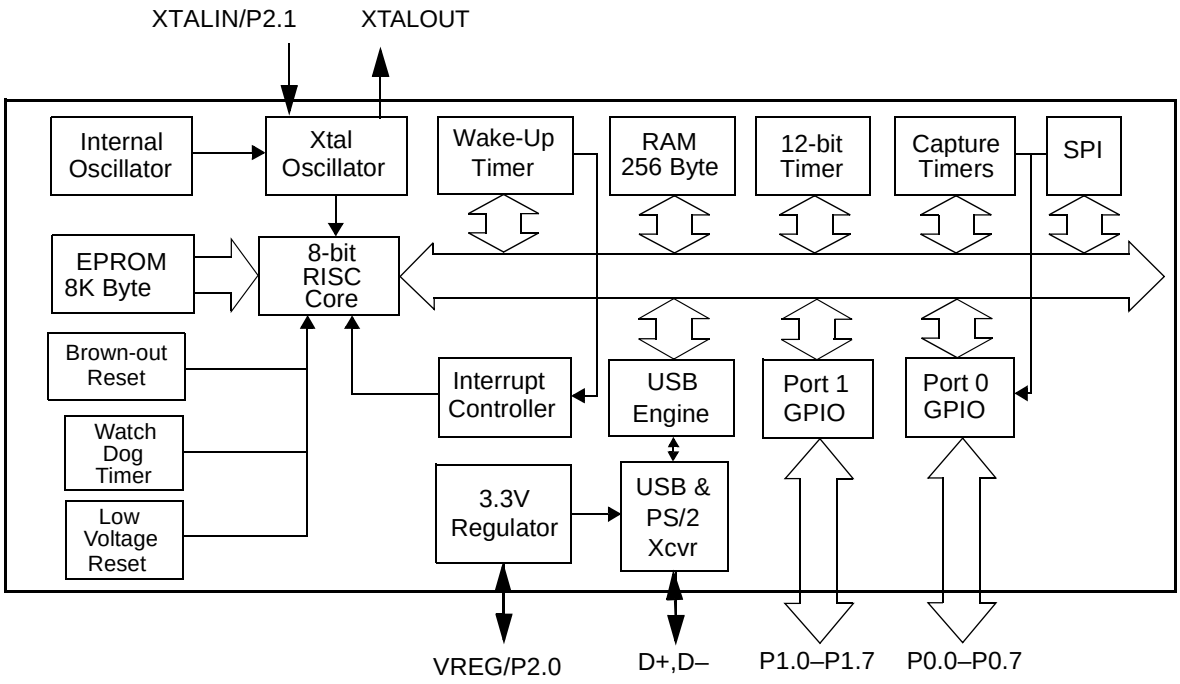
Embedded - Microcontrollers - Application Specific represents a category of microcontrollers designed with unique features and capabilities tailored to specific application needs. Unlike general-purpose microcontrollers, application-specific microcontrollers are optimized for particular tasks, offering 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	16
Voltage - Supply	4V ~ 5.5V
Operating Temperature	0°C ~ 70°C
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/cy7c63743-pxc

2.0 Logic Block Diagram



3.0 Functional Overview

3.1 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 3.3V regulator. All of this adds up to a lower system cost.

The CY7C637xx 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 CY7C637xx 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 CY7C637xx 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 CY7C637xx 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.

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

6.0 Programming Model

Refer to the *CYASM Assembler User's Guide* for more details on firmware operation with the CY7C637xx microcontrollers.

6.1 Program Counter (PC)

The 14-bit program counter (PC) allows access for up to 8 Kbytes of EPROM using the CY7C637xx architecture. The program counter is cleared during reset, such that the first instruction executed after a reset is at address 0x0000. This instruction is typically a jump instruction to a reset handler that initializes the application.

The lower 8 bits of the program counter are incremented as instructions are loaded and executed. The upper six bits of the program counter are incremented by executing an XPAGE instruction. As a result, the last instruction executed within a 256-byte "page" of sequential code should be an XPAGE instruction. The assembler directive "XPAGEON" will cause the assembler to insert XPAGE instructions automatically. As instructions can be either one or two bytes long, the assembler may occasionally need to insert a NOP followed by an XPAGE for correct execution.

The program counter of the next instruction to be executed, carry flag, and zero flag are saved as two bytes on the program stack during an interrupt acknowledge or a CALL instruction. The program counter, carry flag, and zero flag are restored from the program stack only during a RETI instruction.

Please note the program counter cannot be accessed directly by the firmware. The program stack can be examined by reading SRAM from location 0x00 and up.

6.2 8-bit Accumulator (A)

The accumulator is the general-purpose, do everything register in the architecture where results are usually calculated.

6.3 8-bit Index Register (X)

The index register "X" is available to the firmware as an auxiliary accumulator. The X register also allows the processor to perform indexed operations by loading an index value into X.

6.4 8-bit Program Stack Pointer (PSP)

During a reset, the program stack pointer (PSP) is set to zero. This means the program "stack" starts at RAM address 0x00 and "grows" upward from there. Note that the program stack pointer is directly addressable under firmware control, using the MOV PSP,A instruction. The PSP supports interrupt service under hardware control and CALL, RET, and RETI instructions under firmware control.

During an interrupt acknowledge, interrupts are disabled and the program counter, carry flag, and zero flag are written as two bytes of data memory. The first byte is stored in the memory addressed by the program stack pointer, then the PSP is incremented. The second byte is stored in memory addressed by the program stack pointer and the PSP is incremented again. The net effect is to store the program counter and flags on the program "stack" and increment the program stack pointer by two.

The return from interrupt (RETI) instruction decrements the program stack pointer, then restores the second byte from memory addressed by the PSP. The program stack pointer is decremented again and the first byte is restored from memory addressed by the PSP. After the program counter and flags have been restored from stack, the interrupts are enabled. The effect is to restore the program counter and flags from the program stack, decrement the program stack pointer by two, and reenable interrupts.

The call subroutine (CALL) instruction stores the program counter and flags on the program stack and increments the PSP by two.

The return from subroutine (RET) instruction restores the program counter, but not the flags, from program stack and decrements the PSP by two.

Note that there are restrictions in using the JMP, CALL, and INDEX instructions across the 4-KByte boundary of the program memory. Refer to the *CYASM Assembler User's Guide* for a detailed description.

6.5 8-bit Data Stack Pointer (DSP)

The data stack pointer (DSP) supports PUSH and POP instructions that use the data stack for temporary storage. A PUSH instruction will pre-decrement the DSP, then write data to the memory location addressed by the DSP. A POP instruction will read data from the memory location addressed by the DSP, then post-increment the DSP.

During a reset, the Data Stack Pointer will be set to zero. A PUSH instruction when DSP equals zero will write data at the top of the data RAM (address 0xFF). This would write data to the memory area reserved for a FIFO for USB endpoint 0. In non-USB applications, this works fine and is not a problem.

For USB applications, the firmware should set the DSP to an appropriate location to avoid a memory conflict with RAM dedicated to USB FIFOs. The memory requirements for the USB endpoints are shown in Section 8.2. For example, assembly instructions to set the DSP to 20h (giving 32 bytes for program and data stack combined) are shown below.

```
MOV A,20h    ; Move 20 hex into Accumulator (must be D8h or less to avoid USB FIFOs)
SWAP A,DSP   ; swap accumulator value into DSP register
```

6.6 Address Modes

The CY7C637xx microcontrollers support three addressing modes for instructions that require data operands: data, direct, and indexed.

6.6.1 Data

The "Data" address mode refers to a data operand that is actually a constant encoded in the instruction. As an example, consider the instruction that loads A with the constant 0x30:

- MOV A, 30h

This instruction will require two bytes of code where the first byte identifies the "MOV A" instruction with a data operand as the second byte. The second byte of the instruction will be the constant "0xE8h". A constant may be referred to by name if a prior "EQU" statement assigns the constant value to the name. For example, the following code is equivalent to the example shown above.

8.2 Data Memory Organization

The CY7C637xx microcontrollers provide 256 bytes of data RAM. In normal usage, the SRAM is partitioned into four areas: program stack, data stack, user variables and USB endpoint FIFOs as shown below.

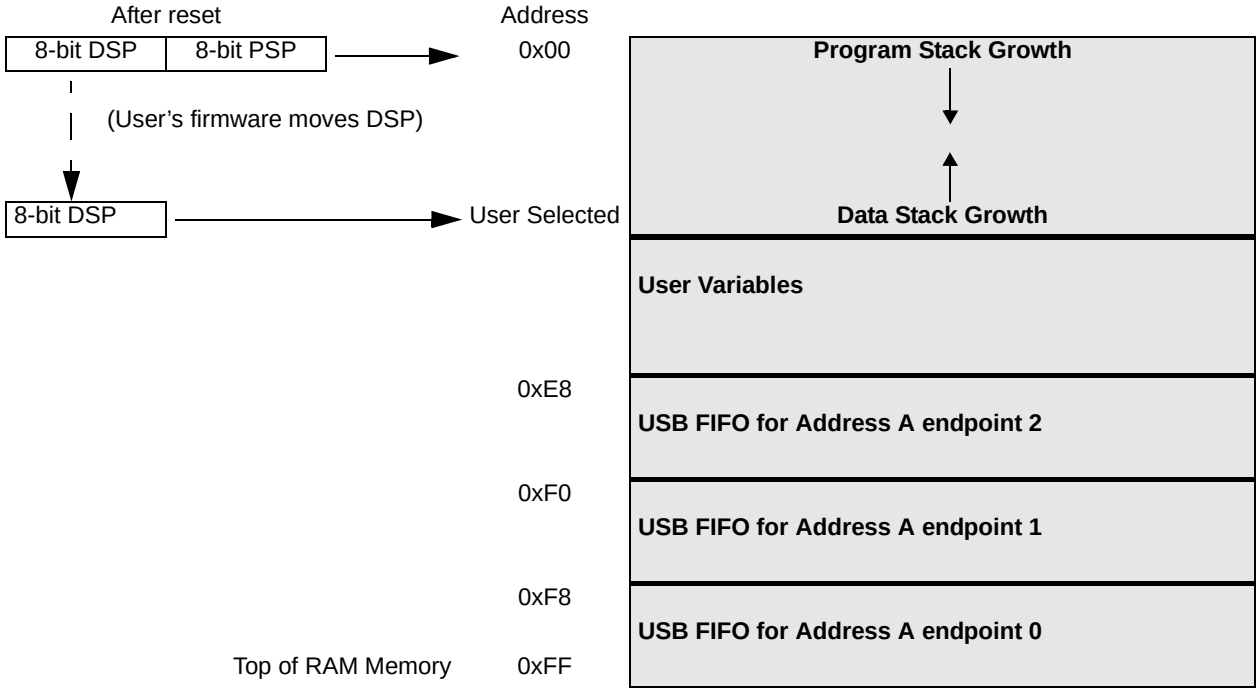


Figure 8-2. Data Memory Organization

8.3 I/O Register Summary

I/O registers are accessed via the I/O Read (IORD) and I/O Write (IOWR, IOWX) instructions. IORD reads the selected port into the accumulator. IOWR writes data from the accumulator to the selected port. Indexed I/O Write (IOWX) adds the contents of X to the address in the instruction to form the port address and writes data from the accumulator to the specified port. Note that specifying address 0 with IOWX (e.g., IOWX 0h) means the I/O port is selected solely by the contents of X.

Note: All bits of all registers are cleared to all zeros on reset, except the Processor Status and Control Register (*Figure 20-1*). *All registers not listed are reserved, and should never be written by firmware. All bits marked as reserved should always be written as 0 and be treated as undefined by reads.*

Table 8-1. I/O Register Summary

Register Name	I/O Address	Read/Write	Function	Fig.
Port 0 Data	0x00	R/W	GPIO Port 0	12-2
Port 1 Data	0x01	R/W	GPIO Port 1	12-3
Port 2 Data	0x02	R	Auxiliary input register for D+, D–, VREG, XTALIN	12-8
Port 0 Interrupt Enable	0x04	W	Interrupt enable for pins in Port 0	21-4
Port 1 Interrupt Enable	0x05	W	Interrupt enable for pins in Port 1	21-5
Port 0 Interrupt Polarity	0x06	W	Interrupt polarity for pins in Port 0	21-6
Port 1 Interrupt Polarity	0x07	W	Interrupt polarity for pins in Port 1	21-7
Port 0 Mode0	0x0A	W	Controls output configuration for Port 0	12-4
Port 0 Mode1	0x0B	W		12-5
Port 1 Mode0	0x0C	W	Controls output configuration for Port 1	12-6
Port 1 Mode1	0x0D	W		12-7

before the part executes code. See Section 10.1 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 Section 9.1.

0 = Enable the internal oscillator.

9.1 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 Section 11.0 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 12-8* 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.

9.2 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 9-1*. 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.

10.0 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 20-1*). Bits 4 (Low-voltage or Brown-out Reset bit) and 6 (Watchdog Reset bit) are used to record the occurrence of LVR/BOR and WDR respectively. The firmware can interrogate these bits to determine the cause of a reset.

The microcontroller begins execution from ROM address 0x0000 after a LVR, BOR, or WDR reset. Although this looks like interrupt vector 0, there is an important difference. Reset processing does NOT push the program counter, carry flag, and zero flag onto program stack. Attempting to execute either a RET or RETI in the reset handler will cause unpredictable execution results.

The following events take place on reset. More details on the various resets are given in the following sections.

1. All registers are reset to their default states (all bits cleared, except in Processor Status and Control Register).
2. GPIO and USB pins are set to high-impedance state.
3. The VREG pin is set to high-impedance state.
4. Interrupts are disabled.
5. USB operation is disabled and must be enabled by firmware if desired, as explained in Section 14.1.
6. For a BOR or LVR, the external oscillator is disabled and Internal Clock mode is activated, followed by a time-out period t_{START} for V_{CC} to stabilize. A WDR does not change the clock mode, and there is no delay for V_{CC} stabilization on a WDR. Note that the External Oscillator Enable (Bit 0, *Figure 9-2*) will be cleared by a WDR, but it does not take effect until suspend mode is entered.
7. The Program Stack Pointer (PSP) and Data Stack Pointer (DSP) reset to address 0x00. Firmware should move the DSP for USB applications, as explained in Section 6.5.
8. Program execution begins at address 0x0000 after the appropriate time-out period.

10.1 Low-voltage Reset (LVR)

When V_{CC} is first applied to the chip, the internal oscillator is started and the Low-voltage Reset is initially enabled by default. At the point where V_{CC} has risen above V_{LVR} (see Section 25.0 for the value of V_{LVR}), an internal counter starts counting for a period of t_{START} (see Section 26.0 for the value of t_{START}). During this t_{START} time, the microcontroller enters a partial suspend state to wait for V_{CC} to stabilize before it begins executing code from address 0x0000.

As long as the LVR circuit is enabled, this reset sequence repeats whenever the V_{CC} pin voltage drops below V_{LVR}. The LVR can be disabled by firmware by setting the Low-voltage Reset Disable bit in the Clock Configuration Register (*Figure 9-2*). In addition, the LVR is automatically disabled in suspend mode to save power. If the LVR was enabled before entering suspend mode, it becomes active again once the suspend mode ends.

When LVR is disabled during normal operation (i.e., by writing '0' to the Low-voltage Reset Disable bit in the Clock Configuration Register), the chip may enter an unknown state if V_{CC} drops below V_{LVR}. Therefore, LVR should be enabled at all times during normal operation. If LVR is disabled (i.e., by firmware or during suspend mode), a secondary low-voltage monitor, BOR, becomes active, as described in the next section. The LVR/BOR Reset bit of the Processor Status and Control Register (*Figure 20-1*), is set to '1' if either a LVR or BOR has occurred.

10.2 Brown Out Reset (BOR)

The Brown Out Reset (BOR) circuit is always active and behaves like the POR. BOR is asserted whenever the V_{CC} voltage to the device is below an internally defined trip voltage of approximately 2.5V. The BOR re-enables LVR. That is, once V_{CC} drops and trips BOR, the part remains in reset until V_{CC} rises above V_{LVR}. At that point, the t_{START} delay occurs before normal operation resumes, and the microcontroller starts executing code from address 0x00 after the t_{START} delay.

In suspend mode, only the BOR detection is active, giving a reset if V_{CC} drops below approximately 2.5V. Since the device is suspended and code is not executing, this lower reset voltage is safe for retaining the state of all registers and memory. Note that in suspend mode, LVR is disabled as discussed in Section 10.1.

10.3 Watchdog Reset (WDR)

The Watchdog Timer Reset (WDR) occurs when the internal Watchdog timer rolls over. Writing any value to the write-only Watchdog Reset Register at address 0x26 will clear the timer. The timer will roll over and WDR will occur if it is not cleared within t_{WATCH} (see *Figure 10-1*) of the last clear. Bit 6 (Watchdog Reset bit) of the Processor Status and Control Register is set to record this event (see Section 20.0 for more details). A Watchdog Timer Reset typically lasts for 2–4 ms, after which the microcontroller begins execution at ROM address 0x0000.

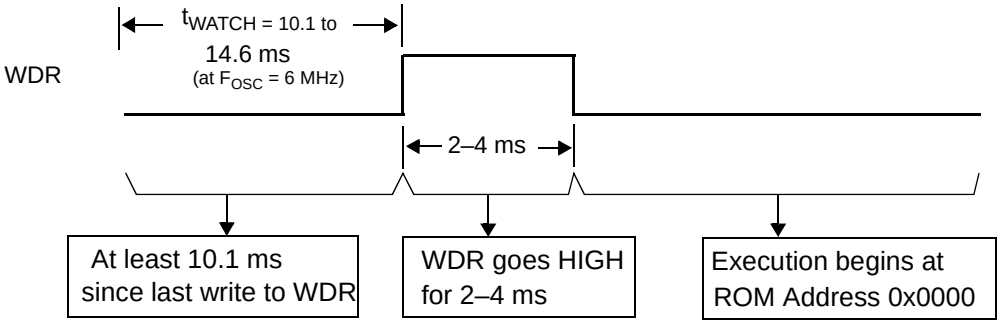


Figure 10-1. Watchdog Reset (WDR, Address 0x26)

Bit #	7	6	5	4	3	2	1	0
Bit Name	P0							
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

Figure 12-2. Port 0 Data (Address 0x00)

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

1 = Port Pin is logic HIGH

0 = Port Pin is logic LOW

Bit #	7	6	5	4	3	2	1	0
Bit Name	P1							
Notes	Pins 7:2 only in CY7C63743						Pins 1:0 in all parts	
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

Figure 12-3. Port 1 Data (Address 0x01)

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

1 = Port Pin is logic HIGH

0 = Port Pin is logic LOW

Bit #	7	6	5	4	3	2	1	0
Bit Name	P0[7:0] Mode0							
Read/Write	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Figure 12-4. GPIO Port 0 Mode0 Register (Address 0x0A)

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

1 = Port 0 Mode 0 is logic HIGH

0 = Port 0 Mode 0 is logic LOW

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

Figure 12-5. GPIO Port 0 Mode1 Register (Address 0x0B)

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

1 = Port Pin Mode 1 is logic HIGH

0 = Port Pin Mode 1 is logic LOW

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

Figure 12-6. GPIO Port 1 Mode0 Register (Address 0x0C)

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

1 = Port Pin Mode 0 is logic HIGH

0 = Port Pin Mode 0 is logic LOW

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

Figure 12-7. GPIO Port 1 Mode1 Register (Address 0x0D)

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 12-1* 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 12-1*) 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 Section 25.0 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.

13.2 USB Port Status and Control

USB status and control is regulated by the USB Status and Control Register as shown in *Figure 13-1*.

Bit #	7	6	5	4	3	2:0		
Bit Name	PS/2 Pull-up Enable	VREG Enable	USB Reset-PS/2 Activity Interrupt Mode	Reserved	USB Bus Activity	D+/D– Forcing Bit		
Read/ Write	R/W	R/W	R/W	-	R/W	R/W		
Reset	0	0	0	0	0	0	0	0

Figure 13-1. USB Status and Control Register (Address 0x1F)

Bit 7: PS/2 Pull-up Enable

This bit is used to enable the internal PS/2 pull-up resistors on the SDATA and SCLK pins. Normally the output high level on these pins is V_{CC}, but note that the output will be clamped to approximately 1 Volt above V_{REG} if the VREG Enable bit is set, or if the Device Address is enabled (bit 7 of the USB Device Address Register, *Figure 14-1*).

1 = Enable PS/2 Pull-up resistors. The SDATA and SCLK pins are pulled up internally to V_{CC} with two resistors of approximately 5 kΩ (see Section 25.0 for the value of R_{PS2}).

0 = Disable PS/2 Pull-up resistors.

Bit 6: V_{REG} Enable

A 3.3V voltage regulator is integrated on chip to provide a voltage source for a 1.5-kΩ pull-up resistor connected to the D– pin as required by the USB Specification. Note that the VREG output has an internal series resistance of approximately 200Ω, the external pull-up resistor required is approximately 1.3-kΩ (see *Figure 16-1*).

1 = Enable the 3.3V output voltage on the VREG pin.

0 = Disable. The VREG pin can be configured as an input.

Bit 5: USB-PS/2 Interrupt Select

This bit allows the user to select whether an USB bus reset interrupt or a PS/2 activity interrupt will be generated when the interrupt conditions are detected.

1 = PS/2 interrupt mode. A PS/2 activity interrupt will occur if the SDATA pin is continuously LOW for 128 to 256 µs.

0 = USB interrupt mode (default state). In this mode, a USB bus reset interrupt will occur if the single ended zero (SE0, D– and D+ are LOW) exists for 128 to 256 µs.

See Section 21.3 for more details.

Bit 4: Reserved. Must be written as a '0'.

Bit 3: USB Bus Activity

The Bus Activity bit is a "sticky" bit that detects any non-idle USB event has occurred on the USB bus. Once set to HIGH by the SIE to indicate the bus activity, this bit retains its logical HIGH value until firmware clears it. Writing a '0' to this bit clears it; writing a '1' preserves its value. The user firmware should check and clear this bit periodically to detect any loss of bus activity. Firmware can clear the Bus Activity bit, but only the SIE can set it. The 1.024-ms timer interrupt service routine is normally used to check and clear the Bus Activity bit.

1 = There has been bus activity since the last time this bit was cleared. This bit is set by the SIE.

0 = No bus activity since last time this bit was cleared (by firmware).

Bit [2:0]: D+/D– Forcing Bit [2:0]

Forcing bits allow firmware to directly drive the D+ and D– pins, as shown in *Table 13-1*. Outputs are driven with controlled edge rates in these modes for low EMI. For forcing the D+ and D– pins in USB mode, D+/D– Forcing Bit 2 should be 0. Setting D+/D– Forcing Bit 2 to '1' puts both pins in an open-drain mode, preferred for applications such as PS/2 or LED driving.

Table 13-1. Control Modes to Force D+/D– Outputs

D+/D– Forcing Bit [2:0]	Control Action	Application
000	Not forcing (SIE controls driver)	Any Mode
001	Force K (D+ HIGH, D– LOW)	USB Mode
010	Force J (D+ LOW, D– HIGH)	
011	Force SE0 (D– LOW, D+ LOW)	
100	Force D– LOW, D+ LOW	PS/2 Mode[2]
101	Force D– LOW, D+ HiZ	
110	Force D– HiZ, D+ LOW	
111	Force D– HiZ, D+ HiZ	

Note:
2. For PS/2 operation, the D+/D– Forcing Bit [2:0] = 111b mode must be set initially (one time only) before using the other PS/2 force modes.

15.0 USB Regulator Output

The VREG pin provides a regulated output for connecting the pull-up resistor required for USB operation. For USB, a 1.5-kΩ resistor is connected between the D– pin and the V_{REG} voltage, to indicate low-speed USB operation. Since the VREG output has an internal series resistance of approximately 200Ω, the external pull-up resistor required is R_{PU} (see Section 25.0).

The regulator output is placed in a high-impedance state at reset, and must be enabled by firmware by setting the VREG Enable bit in the USB Status and Control Register (*Figure 13-1*). This simplifies the design of a combination PS/2-USB device, since the USB pull-up resistor can be left in place during PS/2 operation without loading the PS/2 line. In this mode, the V_{REG} pin can be used as an input and its state can be read at port P2.0. Refer to *Figure 12-8* for the Port 2 data register. This input has a TTL threshold.

In suspend mode, the regulator is automatically disabled. If VREG Enable bit is set (*Figure 13-1*), the VREG pin is pulled up to V_{CC} with an internal 6.2-kΩ resistor. This holds the proper V_{OH} state in suspend mode

Note that enabling the device for USB (by setting the Device Address Enable bit, *Figure 14-1*) activates the internal regulator, even if the VREG Enable bit is cleared to 0. This insures proper 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 CY7C637xx 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.

16.0 PS/2 Operation

The CY7C637xx 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 14-1*).
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 Section 21.3 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 13-1*).
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 16-1*.

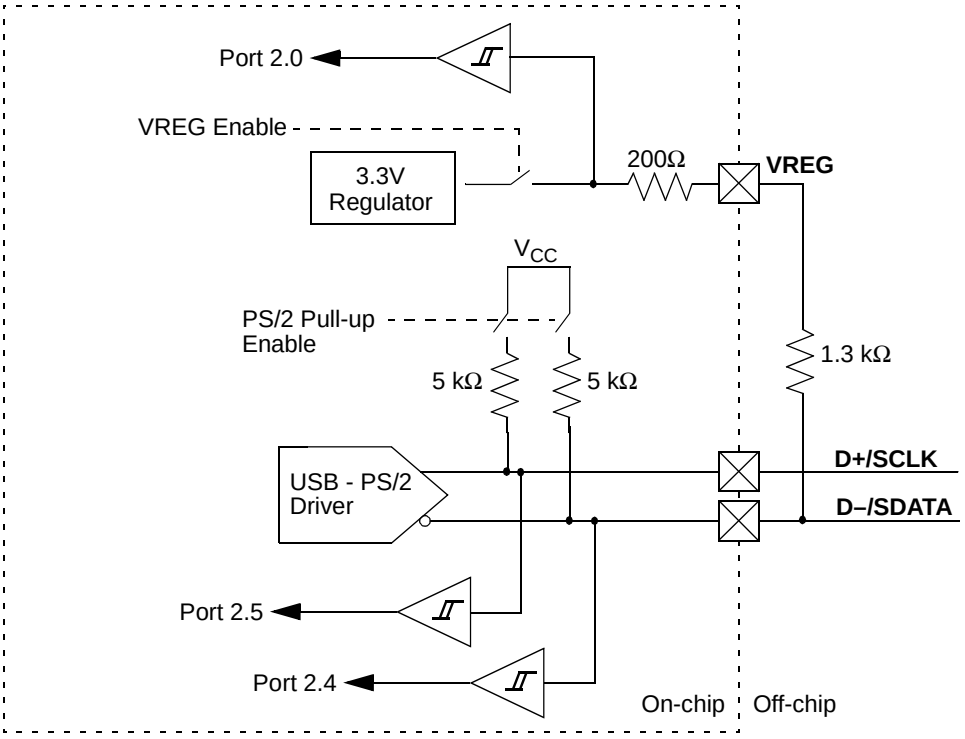


Figure 16-1. Diagram of USB-PS/2 System Connections

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 17-1* and *17-4*).

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

17.3 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 17-3* and *Figure 17-4*).

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.

17.4 SPI Status and Control

The SPI Control Register is shown in *Figure 17-3*. The timing diagram in *Figure 17-4* shows the clock and data states for the various SPI modes.

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

Figure 17-3. SPI Control Register (Address 0x61)

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 17-4*)

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

18.0 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.

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

Figure 18-1. Timer LSB Register (Address 0x24)

Bit [7:0]: Timer lower eight bits

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

Figure 18-2. Timer MSB Register (Address 0x25)

Bit [7:4]: Reserved

Bit [3:0]: Timer upper four bits

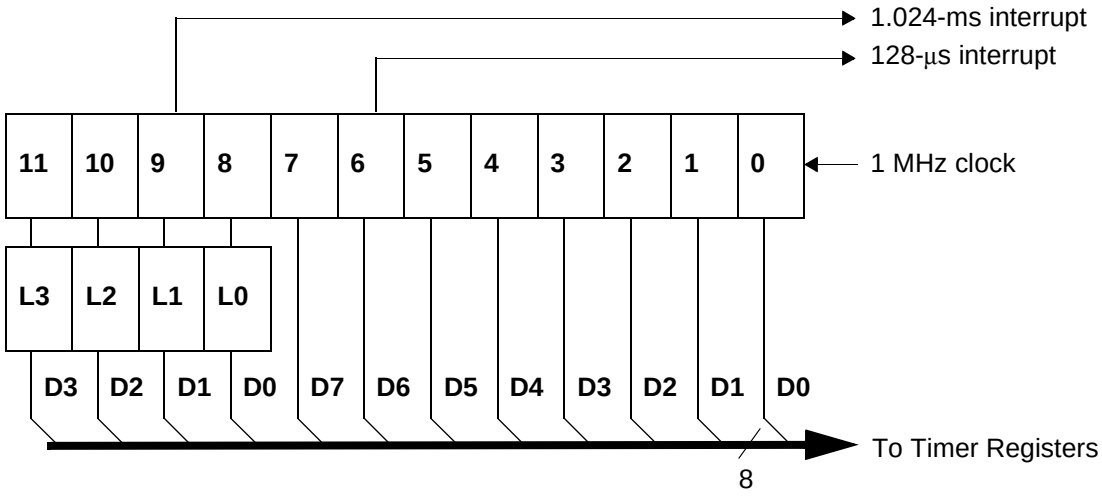


Figure 18-3. Timer Block Diagram

20.0 Processor Status and Control Register

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

Figure 20-1. Processor Status and Control Register (Address 0xFF)

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 21-1* and *Figure 21-2*) 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 Section 26.0 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 13-1*). The details on the event conditions that set this bit are given in Section 21.3. 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 Section 11.0 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 21-1*) and USB Endpoint Interrupt Enable Register (*Figure 21-2*). 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 Section 10.1), 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.*

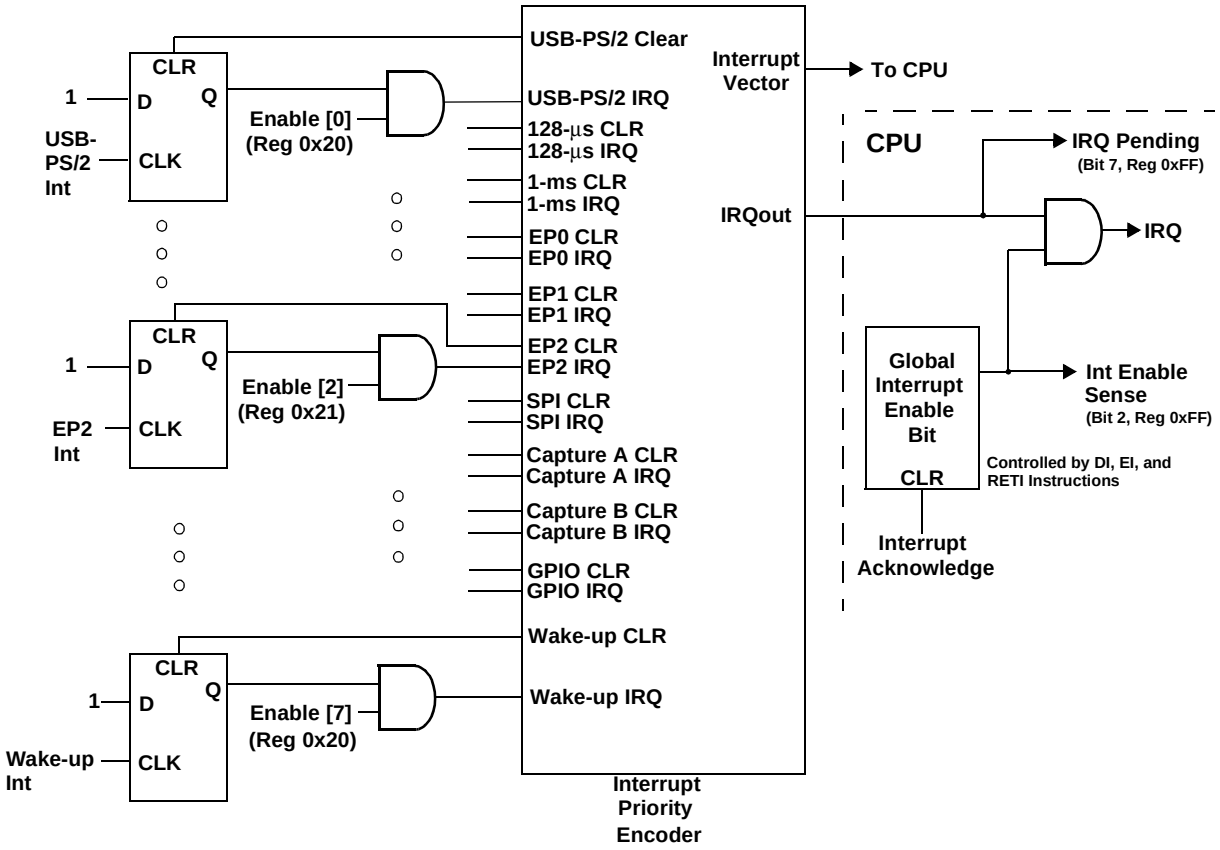


Figure 21-3. Interrupt Controller Logic Block Diagram

Bit #	7	6	5	4	3	2	1	0
Bit Name	P0 Interrupt Enable							
Read/Write	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Figure 21-4. Port 0 Interrupt Enable Register (Address 0x04)

Bit [7:0]: P0 [7:0] Interrupt Enable

1 = Enables GPIO interrupts from the corresponding input pin.

0 = Disables GPIO interrupts from the corresponding input pin.

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

Figure 21-5. Port 1 Interrupt Enable Register (Address 0x05)

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

1 = Enables GPIO interrupts from the corresponding input pin.

0 = Disables GPIO interrupts from the corresponding input pin.

The polarity that triggers an interrupt is controlled independently for each GPIO pin by the GPIO Interrupt Polarity Registers. *Figure 21-6* and *Figure 21-7* control the interrupt polarity of each GPIO pin.

Bit #	7	6	5	4	3	2	1	0
Bit Name	P0 Interrupt Polarity							
Read/Write	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Figure 21-6. Port 0 Interrupt Polarity Register (Address 0x06)

Bit [7:0]: P0[7:0] Interrupt Polarity

1 = Rising GPIO edge

0 = Falling GPIO edge

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

Figure 21-7. Port 1 Interrupt Polarity Register (Address 0x07)

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

1 = Rising GPIO edge

0 = Falling GPIO edge

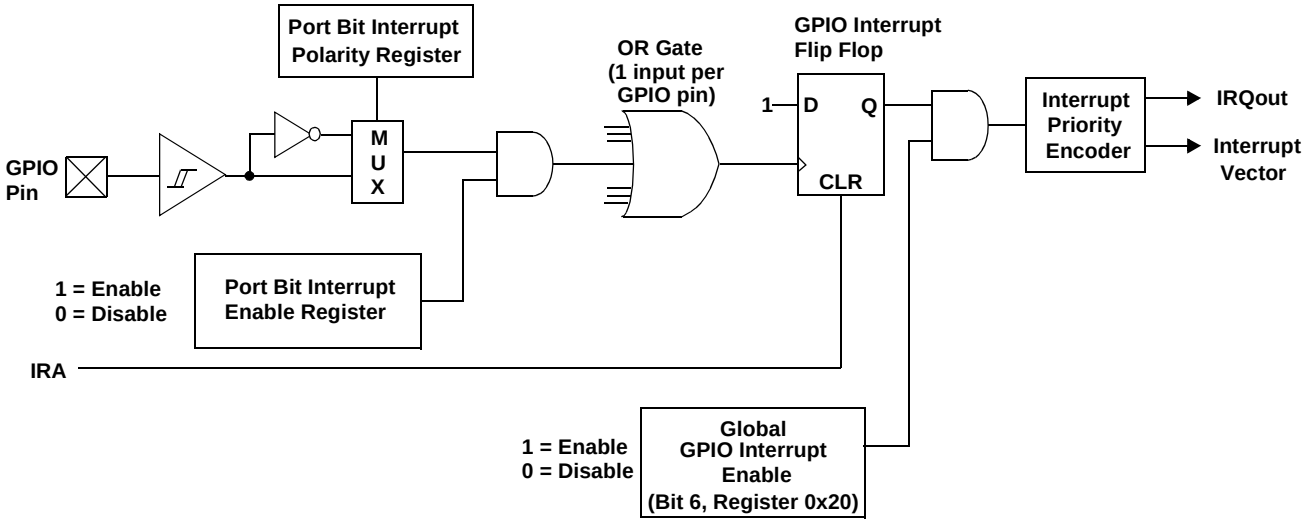


Figure 21-8. GPIO Interrupt Diagram

22.0 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 22-1. 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[3]=0) ACK OUT(STALL[3]=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[3]=0) ACK IN(STALL[3]=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:

3. STALL bit is the bit 7 of the USB Non-Control Device Endpoint Mode registers. Refer to Section 14.3 for more explanation.

The response of the SIE can be summarized as follows:

1. The SIE will only respond to valid transactions, and will ignore non-valid ones.
2. The SIE will generate an interrupt when a valid transaction is completed or when the FIFO is corrupted. FIFO corruption occurs during an OUT or SETUP transaction to a valid internal address, that ends with a non-valid CRC.
3. An incoming Data packet is valid if the count is ≤ Endpoint Size + 2 (includes CRC) and passes all error checking;
4. An IN will be ignored by an OUT configured endpoint and visa versa.
5. The IN and OUT PID status is updated at the end of a transaction.
6. The SETUP PID status is updated at the beginning of the Data packet phase.
7. The entire Endpoint 0 mode register and the Count register are locked to CPU writes at the end of any transaction to that endpoint in which an ACK is transferred. These registers are only unlocked by a CPU read of these registers, and only if that read happens after the transaction completes. This represents about a 1-µs window in which the CPU is locked from register writes to these USB registers. Normally the firmware should perform a register read at the beginning of the Endpoint ISRs to unlock and get the mode register information. The interlock on the Mode and Count registers ensures that the firmware recognizes the changes that the SIE might have made during the previous transaction.

Table 22-3. Details of Modes for Differing Traffic Conditions

End Point Mode											**PID**				Set End Point Mode					
3	2	1	0	Rcved Token	Count	Buffer	Dval	DTOG	DVAL	COUNT	SETUP	IN	OUT	ACK	3	2	1	0	Response	Int
SETUP Packet (if accepting)																				
See22-1				SETUP	<= 10	data	valid	updates	1	updates	1	UC	UC	1	0	0	0	1	ACK	yes
See22-1				SETUP	> 10	junk	x	updates	updates	updates	1	UC	UC	UC	NoChange				Ignore	yes
See 22-1				SETUP	x	junk	invalid	updates	0	updates	1	UC	UC	UC	NoChange				Ignore	yes
Disabled																				
0	0	0	0	x	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
NAK IN/OUT																				
0	0	0	1	OUT	x	UC	x	UC	UC	UC	UC	UC	1	UC	NoChange				NAK	yes
0	0	0	1	OUT	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
0	0	0	1	OUT	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
0	0	0	1	IN	x	UC	x	UC	UC	UC	UC	1	UC	UC	NoChange				NAK	yes
Ignore IN/OUT																				
0	1	0	0	OUT	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
0	1	0	0	IN	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
STALL IN/OUT																				
0	0	1	1	OUT	x	UC	x	UC	UC	UC	UC	UC	1	UC	NoChange				STALL	yes
0	0	1	1	OUT	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
0	0	1	1	OUT	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
0	0	1	1	IN	x	UC	x	UC	UC	UC	UC	1	UC	UC	NoChange				STALL	yes
Control Write																				
ACK OUT/NAK IN																				
1	0	1	1	OUT	<= 10	data	valid	updates	1	updates	UC	UC	1	1	0	0	0	1	ACK	yes
1	0	1	1	OUT	> 10	junk	x	updates	updates	updates	UC	UC	1	UC	NoChange				Ignore	yes
1	0	1	1	OUT	x	junk	invalid	updates	0	updates	UC	UC	1	UC	NoChange				Ignore	yes
1	0	1	1	IN	x	UC	x	UC	UC	UC	UC	1	UC	UC	NoChange				NAK	yes
NAK OUT/Status IN																				
1	0	1	0	OUT	<= 10	UC	valid	UC	UC	UC	UC	UC	1	UC	NoChange				NAK	yes
1	0	1	0	OUT	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
1	0	1	0	OUT	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
1	0	1	0	IN	x	UC	x	UC	UC	UC	UC	1	UC	1	NoChange				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	NoChange				Ignore	no
0	1	1	0	OUT	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
0	1	1	0	IN	x	UC	x	UC	UC	UC	UC	1	UC	1	NoChange				TX 0 Byte	yes

Table 22-3. Details of Modes for Differing Traffic Conditions (continued)

Control Read																		
ACK IN/Status OUT																		
1	1	1	1	OUT	2	UC	valid	1	1	updates	UC	UC	1	1	NoChange				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	NoChange				Ignore	no
1	1	1	1	OUT	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	NoChange				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	NoChange				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	NoChange				Ignore	no
1	1	1	0	OUT	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
1	1	1	0	IN	x	UC	x	UC	UC	UC	UC	1	UC	UC	NoChange				NAK	yes
Status OUT Only																		
0	0	1	0	OUT	2	UC	valid	1	1	updates	UC	UC	1	1	NoChange				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	NoChange				Ignore	no
0	0	1	0	OUT	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
0	0	1	0	IN	x	UC	x	UC	UC	UC	UC	1	UC	UC	0	0	1	1	STALL	yes
OUT Endpoint																		
ACK OUT, STALL Bit = 0 (*Figure 14-3*)																		
1	0	0	1	OUT	<= 10	data	valid	updates	1	updates	UC	UC	1	1	1	0	0	0	ACK	yes
1	0	0	1	OUT	> 10	junk	x	updates	updates	updates	UC	UC	1	UC	NoChange				Ignore	yes
1	0	0	1	OUT	x	junk	invalid	updates	0	updates	UC	UC	1	UC	NoChange				Ignore	yes
1	0	0	1	IN	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
ACK OUT, STALL Bit = 1 (*Figure 14-3*)																		
1	0	0	1	OUT	<= 10	UC	valid	UC	UC	UC	UC	UC	1	UC	NoChange				STALL	yes
1	0	0	1	OUT	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
1	0	0	1	OUT	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
1	0	0	1	IN	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
NAK OUT																		
1	0	0	0	OUT	<= 10	UC	valid	UC	UC	UC	UC	UC	1	UC	NoChange				NAK	yes
1	0	0	0	OUT	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
1	0	0	0	OUT	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
1	0	0	0	IN	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
Reserved																		
0	1	0	1	OUT	x	updates	updates	updates	updates	updates	UC	UC	1	1	NoChange				RX	yes
0	1	0	1	IN	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
IN Endpoint																		
ACK IN, STALL Bit = 0 (*Figure 14-3*)																		
1	1	0	1	OUT	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
1	1	0	1	IN	x	UC	x	UC	UC	UC	UC	1	UC	1	1	1	0	0	ACK (back)	yes
ACK IN, STALL Bit = 1 (*Figure 14-3*)																		
1	1	0	1	OUT	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				Ignore	no
1	1	0	1	IN	x	UC	x	UC	UC	UC	UC	1	UC	UC	NoChange				STALL	yes
NAK IN																		

Table 22-3. Details of Modes for Differing Traffic Conditions (continued)

1	1	0	0	OUT	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange	Ignore	no
1	1	0	0	IN	x	UC	x	UC	UC	UC	UC	1	UC	UC	NoChange	NAK	yes
Reserved																	
0	1	1	1	Out	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange	Ignore	no
0	1	1	1	IN	x	UC	x	UC	UC	UC	UC	1	UC	UC	NoChange	TX	yes

26.0 Switching Characteristics (continued)

Parameter	Description	Conditions	Min.	Max.	Unit
T_{SCKH}	SPI Clock High Time	High for CPOL = 0, Low for CPOL = 1	125		ns
T_{SCKL}	SPI Clock Low Time	Low for CPOL = 0, High for CPOL = 1	125		ns
T_{MDO}	Master Data Output Time	SCK to data valid	–25	50	ns
T_{MDO1}	Master Data Output Time, First bit with CPHA = 1	Time before leading SCK edge	100		ns
T_{MSU}	Master Input Data Set-up time		50		ns
T_{MHD}	Master Input Data Hold time		50		ns
T_{SSU}	Slave Input Data Set-up Time		50		ns
T_{SHD}	Slave Input Data Hold Time		50		ns
T_{SDO}	Slave Data Output Time	SCK to data valid		100	ns
T_{SDO1}	Slave Data Output Time, First bit with CPHA = 1	Time after $\overline{SS}$ LOW to data valid		100	ns
T_{SSS}	Slave Select Set-up Time	Before first SCK edge	150		ns
T_{SSH}	Slave Select Hold Time	After last SCK edge	150		ns

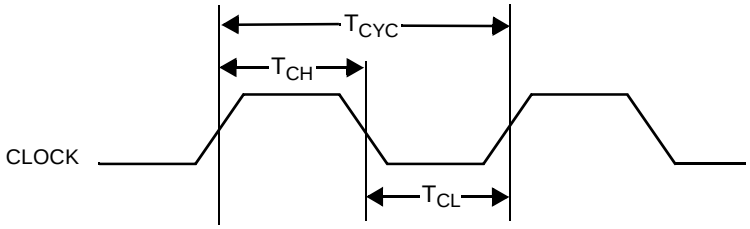


Figure 26-1. Clock Timing

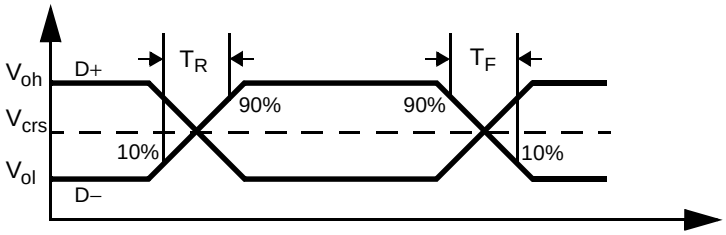


Figure 26-2. USB Data Signal Timing

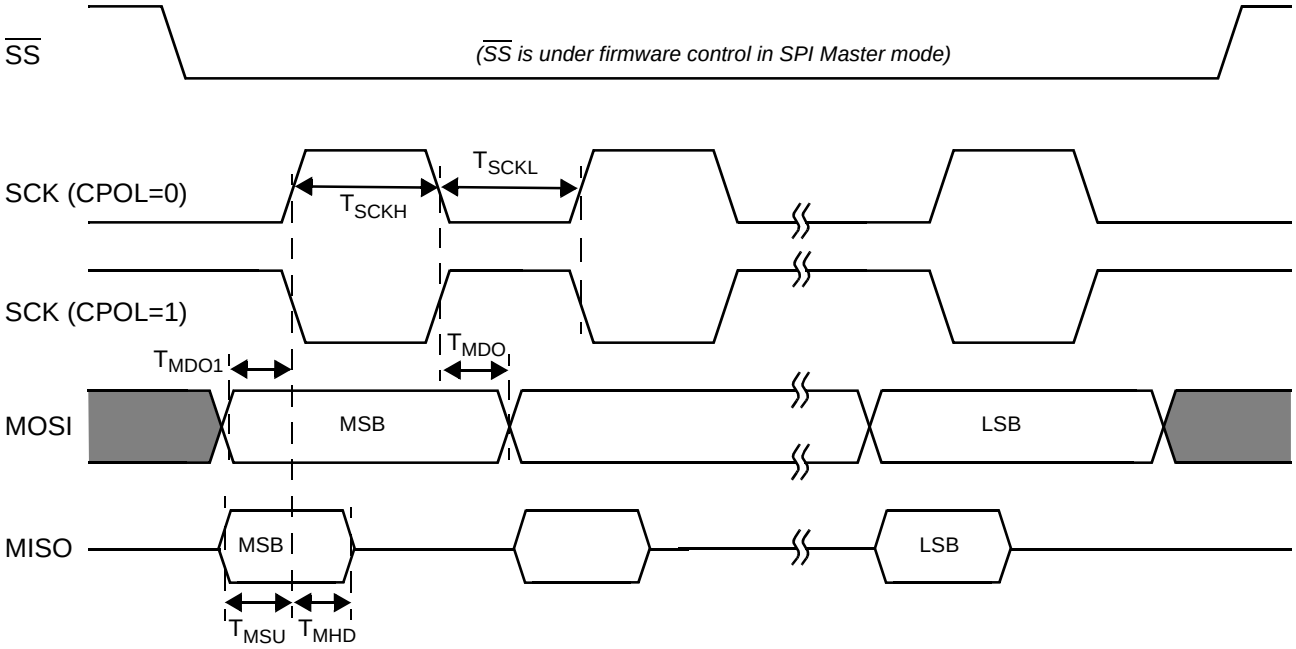


Figure 26-8. SPI Master Timing, CPHA = 1

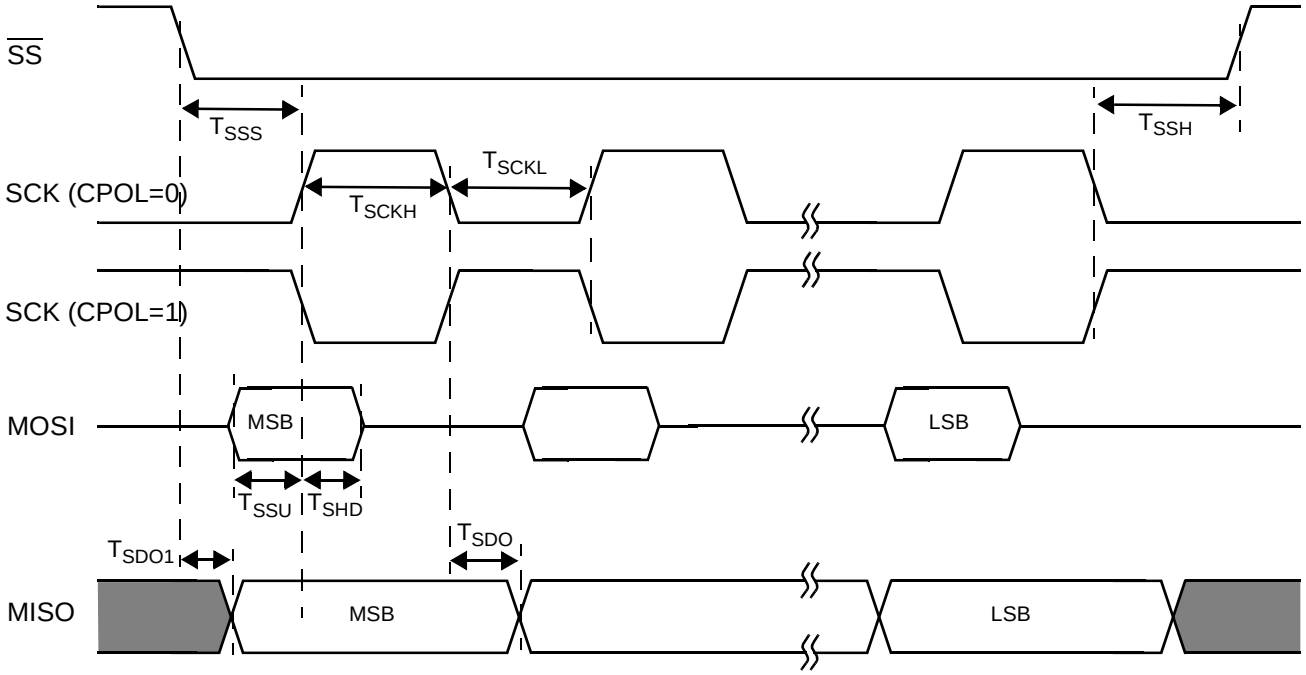


Figure 26-9. SPI Slave Timing, CPHA = 1

28.0 Package Diagrams (continued)

DIE FORM

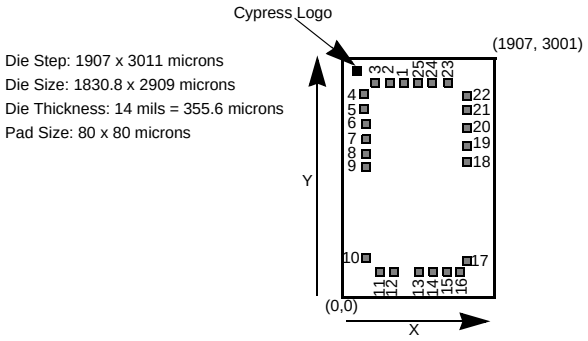


Table 28-1 below shows the die pad coordinates for the CY7C63722-XC. The center location of each bond pad is relative to the bottom left corner of the die which has coordinate (0,0).

Table 28-1. CY7C63722-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

Document History Page

Document Title: CY7C63722, CY7C63723, CY7C63743 enCoRe™ USB Combination Low-Speed USB and PS/2 Peripheral Controller Document Number: 38-08022				
REV.	**ECN NO.**	**Issue Date**	**Orig. of Change**	**Description of Change**
**	118643	10/22/02	BON	Converted from Spec 38-00944 to Spec 38-08022. Added notes 17, 18 to section 26 Removed obsolete parts (63722-PC and 63742) Added die sale Added section 23 (Register Summary)
*A	243308	SEE ECN	KKU	Added 24 QSOP package Added Lead-free packages to section 27 Reformatted to update format
*B	267229	See ECN	ARI	Corrected part number in the Ordering Information section