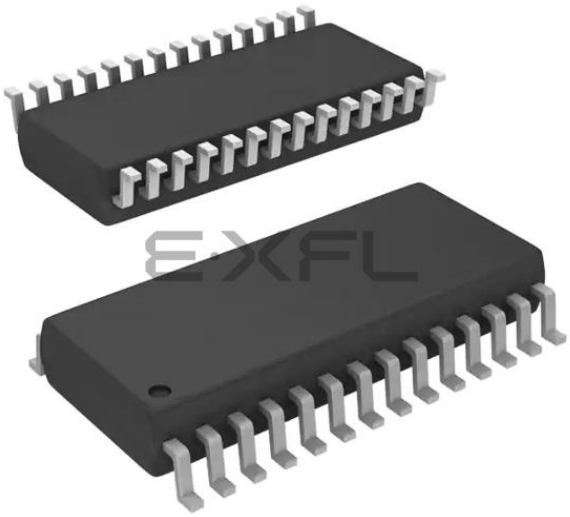


Welcome to E-XFL.COM

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

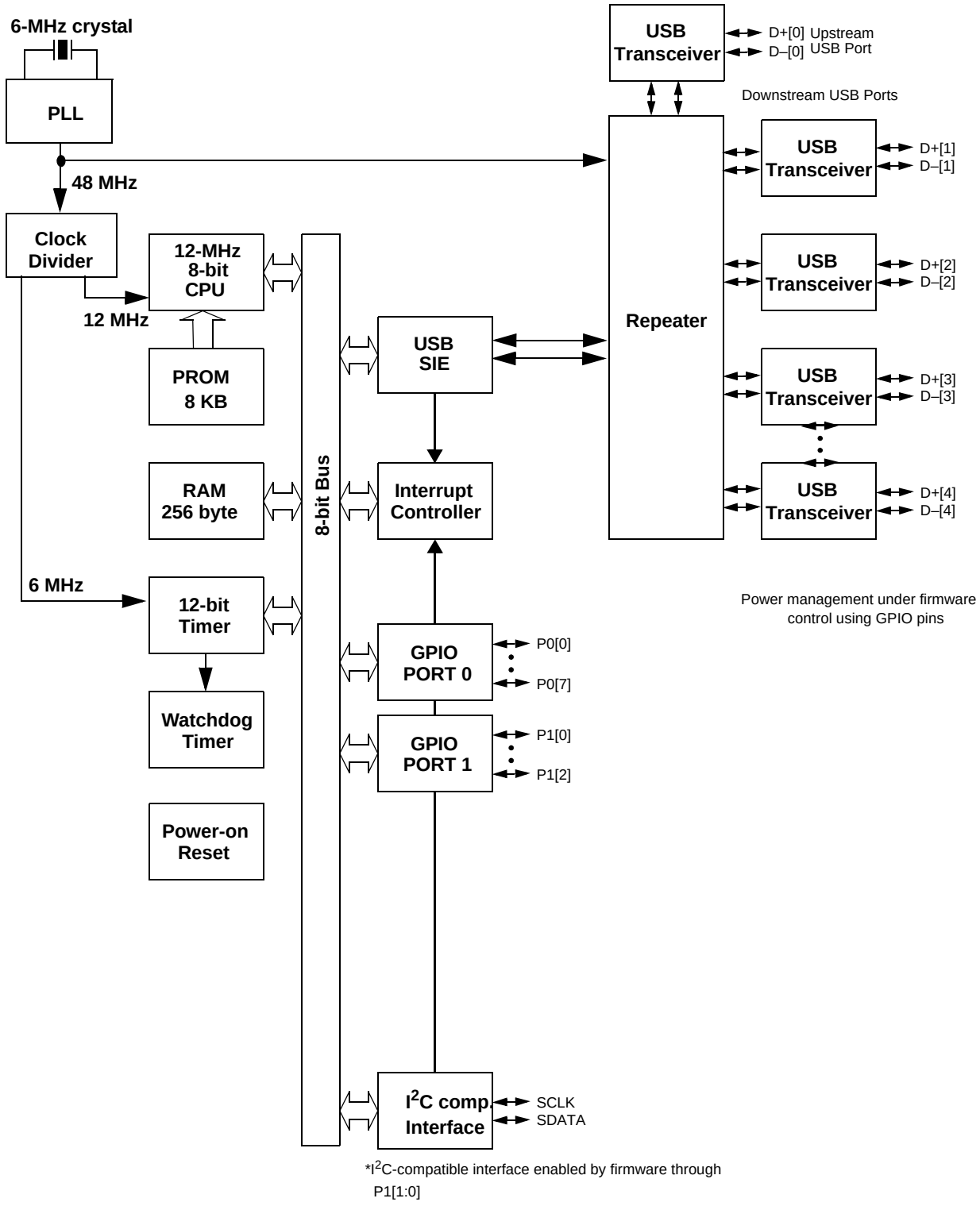
Embedded - Microcontrollers - Application Specific represents a category of microcontrollers designed with unique features and capabilities tailored to specific application needs. Unlike general-purpose microcontrollers, application-specific microcontrollers are optimized for particular tasks, offering 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 Hub/Microcontroller
Core Processor	M8
Program Memory Type	OTP (8kB)
Controller Series	USB Hub
RAM Size	256 x 8
Interface	I²C, USB
Number of I/O	11
Voltage - Supply	4V ~ 5.5V
Operating Temperature	0°C ~ 70°C
Mounting Type	Surface Mount
Package / Case	28-SOIC (0.295", 7.50mm Width)
Supplier Device Package	28-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/infineon-technologies/cy7c65113c-sxc

Logic Block Diagram



I/O Register Summary

I/O registers are accessed via the I/O Read (IORD) and I/O Write (IOWR, IOWX) instructions. IORD reads data from the selected port into the accumulator. IOWR performs the reverse; it 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. Specifying address 0 (e.g., IOWX 0h) means the I/O register is selected solely by the contents of X.

All undefined registers are reserved. Do not write to reserved registers as this may cause an undefined operation or increased current consumption during operation. When writing to registers with reserved bits, the reserved bits must be written with '0.'

Table 2. I/O Register Summary

Register Name	I/O Address	Read/Write	Function	Page
Port 0 Data	0x00	R/W	GPIO Port 0 Data	14
Port 1 Data	0x01	R/W	GPIO Port 1 Data	17
Port 0 Interrupt Enable	0x04	W	Interrupt Enable for Pins in Port 0	19
Port 1 Interrupt Enable	0x05	W	Interrupt Enable for Pins in Port 1	19
GPIO Configuration	0x08	R/W	GPIO Port Configurations	18
I^2C Configuration	0x09	R/W	I^2C Position Configuration	20
USB Device Address A	0x10	R/W	USB Device Address A	31
EP A0 Counter Register	0x11	R/W	USB Address A, Endpoint 0 Counter	33
EP A0 Mode Register	0x12	R/W	USB Address A, Endpoint 0 Configuration	32
EP A1 Counter Register	0x13	R/W	USB Address A, Endpoint 1 Counter	33
EP A1 Mode Register	0x14	R/W	USB Address A, Endpoint 1 Configuration	33
EP A2 Counter Register	0x15	R/W	USB Address A, Endpoint 2 Counter	33
EP A2 Mode Register	0x16	R/W	USB Address A, Endpoint 2 Configuration	33
USB Status & Control	0x1F	R/W	USB Upstream Port Traffic Status and Control	31
Global Interrupt Enable	0x20	R/W	Global Interrupt Enable	21
Endpoint Interrupt Enable	0x21	R/W	USB Endpoint Interrupt Enables	21
Interrupt Vector	0x23	R	Pending Interrupt Vector Read/Clear	23
Timer (LSB)	0x24	R	Lower Eight Bits of Free-running Timer (1 MHz)	20
Timer (MSB)	0x25	R	Upper Four Bits of Free-running Timer	20
WDR Clear	0x26	W	Watchdog Reset Clear	13
I^2C Control & Status	0x28	R/W	I^2C Status and Control	21
I^2C Data	0x29	R/W	I^2C Data	18
Reserved	0x30		Reserved	
Reserved	0x31		Reserved	
Reserved	0x32		Reserved	
Reserved	0x38-0x3F		Reserved	
USB Device Address B	0x40	R/W	USB Device Address B (not used in 5-endpoint mode)	31
EP B0 Counter Register	0x41	R/W	USB Address B, Endpoint 0 Counter	33
EP B0 Mode Register	0x42	R/W	USB Address B, Endpoint 0 Configuration, or USB Address A, Endpoint 3 in 5-endpoint mode	32
EP B1 Counter Register	0x43	R/W	USB Address B, Endpoint 1 Counter	33
EP B1 Mode Register	0x44	R/W	USB Address B, Endpoint 1 Configuration, or USB Address A, Endpoint 4 in 5-endpoint mode	33
Hub Port Connect Status	0x48	R/W	Hub Downstream Port Connect Status	27
Hub Port Enable	0x49	R/W	Hub Downstream Ports Enable	27

Table 2. I/O Register Summary (continued)

Register Name	I/O Address	Read/Write	Function	Page
Hub Port Speed	0x4A	R/W	Hub Downstream Ports Speed	27
Hub Port Control (Ports [4:1])	0x4B	R/W	Hub Downstream Ports Control (Ports [4:1])	28
Hub Port Suspend	0x4D	R/W	Hub Downstream Port Suspend Control	30
Hub Port Resume Status	0x4E	R	Hub Downstream Ports Resume Status	30
Hub Ports SE0 Status	0x4F	R	Hub Downstream Ports SE0 Status	29
Hub Ports Data	0x50	R	Hub Downstream Ports Differential Data	29
Hub Downstream Force Low	0x51	R/W	Hub Downstream Ports Force LOW (Ports [1:4])	28
Processor Status & Control	0xFF	R/W	Microprocessor Status and Control Register	20

Instruction Set Summary

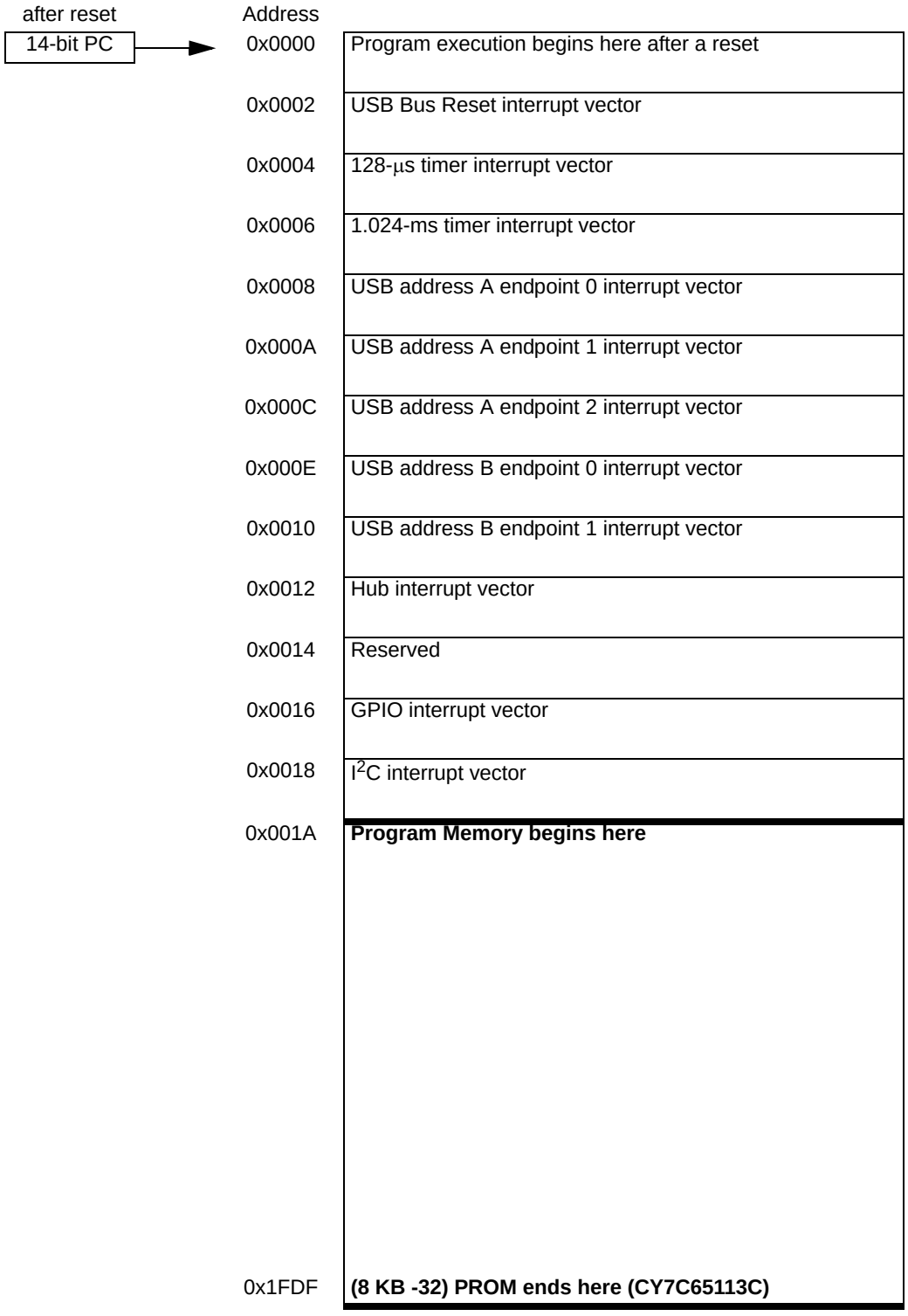
Refer to the *CYASM Assembler User's Guide* for more details. Note that conditional jump instructions (i.e., JC, JNC, JZ, JNZ) take five cycles if jump is taken, four cycles if no jump.

Table 3. Instruction Set Summary

MNEMONIC	operand	opcode	cycles	MNEMONIC	operand	opcode	cycles
HALT		00	7	NOP		20	4
ADD A,expr	data	01	4	INC A	acc	21	4
ADD A,[expr]	direct	02	6	INC X	x	22	4
ADD A,[X+expr]	index	03	7	INC [expr]	direct	23	7
ADC A,expr	data	04	4	INC [X+expr]	index	24	8
ADC A,[expr]	direct	05	6	DEC A	acc	25	4
ADC A,[X+expr]	index	06	7	DEC X	x	26	4
SUB A,expr	data	07	4	DEC [expr]	direct	27	7
SUB A,[expr]	direct	08	6	DEC [X+expr]	index	28	8
SUB A,[X+expr]	index	09	7	IORD expr	address	29	5
SBB A,expr	data	0A	4	IOWR expr	address	2A	5
SBB A,[expr]	direct	0B	6	POP A		2B	4
SBB A,[X+expr]	index	0C	7	POP X		2C	4
OR A,expr	data	0D	4	PUSH A		2D	5
OR A,[expr]	direct	0E	6	PUSH X		2E	5
OR A,[X+expr]	index	0F	7	SWAP A,X		2F	5
AND A,expr	data	10	4	SWAP A,DSP		30	5
AND A,[expr]	direct	11	6	MOV [expr],A	direct	31	5
AND A,[X+expr]	index	12	7	MOV [X+expr],A	index	32	6
XOR A,expr	data	13	4	OR [expr],A	direct	33	7
XOR A,[expr]	direct	14	6	OR [X+expr],A	index	34	8
XOR A,[X+expr]	index	15	7	AND [expr],A	direct	35	7
CMP A,expr	data	16	5	AND [X+expr],A	index	36	8
CMP A,[expr]	direct	17	7	XOR [expr],A	direct	37	7
CMP A,[X+expr]	index	18	8	XOR [X+expr],A	index	38	8
MOV A,expr	data	19	4	IOWX [X+expr]	index	39	6
MOV A,[expr]	direct	1A	5	CPL		3A	4
MOV A,[X+expr]	index	1B	6	ASL		3B	4

Program Memory Organization

Figure 2. Program Memory Space with Interrupt Vector Table



Note that the upper 32 bytes of the 8K PROM are reserved. Therefore, user's program must not overwrite this space.

The XTALIN and XTALOUT are the clock pins to the microcontroller. The user can connect an external oscillator or a crystal to these pins. When using an external crystal, keep PCB traces between the chip leads and crystal as short as possible (less than 2 cm). A 6-MHz fundamental frequency parallel resonant crystal can be connected to these pins to provide a reference frequency for the internal PLL. The two internal 30-pF load caps appear in series to the external crystal and would be equivalent to a 15-pF load. Therefore, the crystal must have a required load capacitance of about 15–18 pF. A ceramic resonator does not allow the microcontroller to meet the timing specifications of full speed USB and therefore a ceramic resonator is not recommended with these parts.

An external 6-MHz clock can be applied to the XTALIN pin if the XTALOUT pin is left open. Grounding the XTALOUT pin when driving XTALIN with an oscillator does not work because the internal clock is effectively shorted to ground.

Reset

The CY7C65113C supports two resets: POR and WDR. Each of these resets causes:

- all registers to be restored to their default states
- the USB device addresses to be set to 0
- all interrupts to be disabled
- the PSP and DSP to be set to memory address 0x00.

The occurrence of a reset is recorded in the Processor Status and Control Register, as described in Section. Bits 4 and 6 are used to record the occurrence of POR and WDR respectively. Firmware can interrogate these bits to determine the cause of a reset.

Program execution starts at ROM address 0x0000 after a 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. The firmware reset handler should configure the hardware before the “main” loop of code. Attempting to execute a RET or RETI in the firmware reset handler causes unpredictable execution results.

Power-on Reset

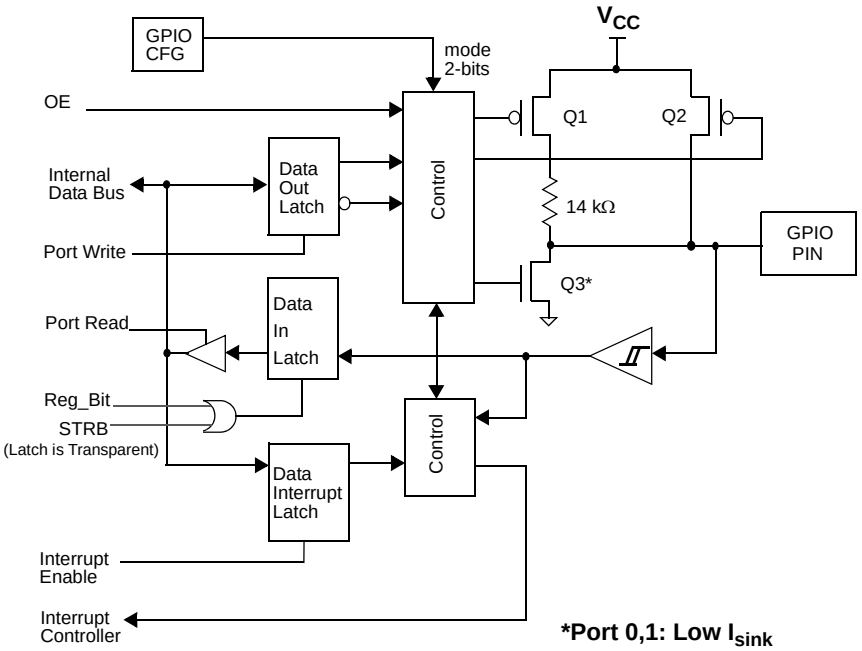
When V_{CC} is first applied to the chip, the POR signal is asserted and the CY7C65113C enters a “semi-suspend” state. During the semi-suspend state, which is different from the suspend state defined in the USB specification, the oscillator and all other blocks of the part are functional, except for the CPU. This semi-suspend time ensures that both a valid V_{CC} level is reached and that the internal PLL has time to stabilize before full operation begins. When the V_{CC} has risen above approximately 2.5V, and the oscillator is stable, the POR is deasserted and the on-chip timer starts counting. The first 1 ms of suspend time is not interruptible, and the semi-suspend state continues for an additional 95 ms unless the count is bypassed by a USB Bus Reset on the upstream port. The 95 ms provides time for V_{CC} to stabilize at a valid operating voltage before the chip executes code.

If a USB Bus Reset occurs on the upstream port during the 95 ms semi-suspend time, the semi-suspend state is aborted and program execution begins immediately from address 0x0000. In this case, the Bus Reset interrupt is pending but not serviced until firmware sets the USB Bus Reset Interrupt Enable bit (Bit 0, *Figure 18*) and enables interrupts with the EI command.

The POR signal is asserted whenever V_{CC} drops below approximately 2.5V, and remains asserted until V_{CC} rises above this level again. Behavior is the same as described above.

General-purpose I/O Ports

Figure 5. Block Diagram of a GPIO Pin



There are 11 GPIO pins (P0[7:0] and P1[2:0]) for the hardware interface. Each port can be configured as inputs with internal pull-ups, open drain outputs, or traditional CMOS outputs. The data for each GPIO port is accessible through the data registers. Port data registers are shown in *Figure 6* through Figure , and are set to 1 on reset.

Figure 6. Port 0 Data.

Port 0 Data **Address 0x00**

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

Figure 7. Port1 Data

Port 1 Data **Address 0x01**

Bit #	-	-	-	-	-	2	1	0
Bit Name	-	-	-	-	-	P1.2	P1.1	P1.0
Read/Write	-	-	-	-	-	R/W	R/W	R/W
Reset	-	-	-	-	-	1	1	1

Special care should be taken with any unused GPIO data bits. An unused GPIO data bit, either a pin on the chip or a port bit that is not bonded on a particular package, must not be left floating when the device enters the suspend state. If a GPIO data bit is left floating, the leakage current caused by the floating bit may violate the suspend current limitation specified by the USB Specifications. If a '1' is written to the unused data bit and the port is configured with open drain outputs, the unused data bit remains in an indeterminate state. Therefore, if an unused port bit is programmed in open-drain mode, it must be written with a '0.'

A read from a GPIO port always returns the present state of the voltage at the pin, independent of the settings in the Port Data Registers. During reset, all of the GPIO pins are set to a high-impedance input state. Writing a ‘0’ to a GPIO pin drives the pin LOW. In this state, a ‘0’ is always read on that GPIO pin unless an external source overdrives the internal pull-down device.

GPIO Configuration Port

Every GPIO port can be programmed as inputs with internal pull-ups, outputs LOW or HIGH, or Hi-Z (floating, the pin is not driven internally). In addition, the interrupt polarity for each port can be programmed. The Port Configuration bits (Figure) and the Interrupt Enable bit (Figure 10 through Figure 10) determine the interrupt polarity of the port pins

Figure 8. GPIO Configuration Register.

GPIO Configuration **Address 0x08**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Reserved	Reserved	Reserved	Reserved	Port 1 Config Bit 1	Port 1 Config Bit 0	Port 0 Config Bit 1	Port 0 Config Bit 0
Read/Write	-	-	-	-	R/W	R/W	R/W	R/W
Reset	-	-	-	-	0	0	0	0

As shown in Table 4 below, a positive polarity on an input pin represents a rising edge interrupt (LOW to HIGH), and a negative polarity on an input pin represents a falling edge interrupt (HIGH to LOW).

The GPIO interrupt is generated when all of the following conditions are met: the Interrupt Enable bit of the associated Port Interrupt Enable Register is enabled, the GPIO Interrupt Enable bit of the Global Interrupt Enable Register (*Figure 18*) is enabled, the Interrupt Enable Sense (bit 2, *Figure 17*) is set, and the GPIO pin of the port sees an event matching the interrupt polarity.

The driving state of each GPIO pin is determined by the value written to the pin’s Data Register (*Figure 6* through Figure) and by its associated Port Configuration bits as shown in the GPIO Configuration Register (*Figure*). These ports are configured on a per-port basis, so all pins in a given port are configured together. The possible port configurations are detailed in Table 4. As shown in this table below, when a GPIO port is configured with CMOS outputs, interrupts from that port are disabled.

During reset, all of the bits in the GPIO Configuration Register are written with ‘0’ to select Hi-Z mode for all GPIO ports as the default configuration.

Table 4. GPIO Port Output Control Truth Table and Interrupt Polarity

Port Config Bit 1	Port Config Bit 0	Data Register	Output Drive Strength	Interrupt Enable Bit	Interrupt Polarity
1	1	0	Output LOW	0	Disabled
		1	Resistive	1	– (Falling Edge)
1	0	0	Output LOW	0	Disabled
		1	Output HIGH	1	Disabled
0	1	0	Output LOW	0	Disabled
		1	Hi-Z	1	– (Falling Edge)
0	0	0	Output LOW	0	Disabled
		1	Hi-Z	1	+ (Rising Edge)

Q1, Q2, and Q3 discussed below are the transistors referenced in Figure . The available GPIO drive strength are:

- **Output LOW Mode**: The pin's Data Register is set to '0.'

  Writing '0' to the pin's Data Register puts the pin in output LOW mode, regardless of the contents of the Port Configuration Bits[1:0]. In this mode, Q1 and Q2 are OFF. Q3 is ON. The GPIO pin is driven LOW through Q3.

- **Output HIGH Mode**: The pin's Data Register is set to 1 and the Port Configuration Bits[1:0] is set to '10.'

  In this mode, Q1 and Q3 are OFF. Q2 is ON. The GPIO is pulled up through Q2. The GPIO pin is capable of sourcing... of current.

- **Resistive Mode**: The pin's Data Register is set to 1 and the Port Configuration Bits[1:0] is set to '11.'

  Q2 and Q3 are OFF. Q1 is ON. The GPIO pin is pulled up with an internal 14kΩ resistor. In resistive mode, the pin may serve as an input. Reading the pin's Data Register returns a logic HIGH if the pin is not driven LOW by an external source.

- **Hi-Z Mode**: The pin's Data Register is set to1 and Port Configuration Bits[1:0] is set either '00' or '01.'

  Q1, Q2, and Q3 are all OFF. The GPIO pin is not driven internally. In this mode, the pin may serve as an input. Reading the Port Data Register returns the actual logic value on the port pins.

GPIO Interrupt Enable Ports

Each GPIO pin can be individually enabled or disabled as an interrupt source. The Port 0–1 Interrupt Enable Registers provide this feature with an Interrupt Enable bit for each GPIO pin.

During a reset, GPIO interrupts are disabled by clearing all of the GPIO Interrupt Enable bits. Writing a '1' to a GPIO Interrupt Enable bit enables GPIO interrupts from the corresponding input pin. All GPIO pins share a common interrupt, as discussed in Section .

Figure 9. Port 0 Interrupt Enable

Port 0 Interrupt Enable **Address 0x04**

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

Figure 10. Port 1 Interrupt Enable

Port 1 Interrupt Enable **Address 0x05**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Reserved	Reserved	Reserved	Reserved	Reserved	P0.2 Intr Enable	P1.1 Intr Enable	P1.0 Intr Enable
Read/Write	-	-	-	-	-	W	W	W
Reset	-	-	-	-	-	0	0	0

I²C Configuration Register

Internal hardware supports communication with external devices through an I²C-compatible interface. I²C-compatible function is discussed in detail in Section .[3] The I²C Position bit (Bit 7, *Figure 14*) and I²C Port Width bit (Bit 1, *Figure 14*) select the locations of the SCL (clock) and SDA (data) pins on Port 1 as shown in Table 5. These bits are cleared on reset. When the GPIO is configured for I²C function, the internal pull ups on the pins are disabled. Addition of an external weak pull-up resistors on SCL and SDA is recommended.

Figure 14. I²C Configuration Register

I²C Configuration **Address 0x09**

Bit #	7	6	5	4	3	2	1	0
Bit Name	I²C Position	Reserved	Reserved	Reserved	Reserved	Reserved	I²C Port Width	Reserved
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

Table 5. I²C Port Configuration

I²C Position (Bit7, *Figure 14*)	I²C Port Width (Bit1, *Figure 14*)	I²C Position
0	0	I²C on P1[1:0], 0:SCL, 1:SDA

I2C-compatible Controller

The I2C-compatible block provides a versatile two-wire communication with external devices, supporting master, slave, and multi-master modes of operation. The I2C-compatible block functions by handling the low-level signaling in hardware, and issuing interrupts as needed to allow firmware to take appropriate action during transactions. While waiting for firmware response, the hardware keeps the I2C-compatible bus idle if necessary.

The I2C-compatible block generates an interrupt to the microcontroller at the end of each received or transmitted byte, when a stop bit is detected by the slave when in receive mode, or when arbitration is lost. Details of the interrupt responses are given in Section .

The I2C-compatible interface consists of two registers, an I²C Data Register (*Figure 15*) and an I²C Status and Control Register (Figure 16). The I²C Data Register is implemented as separate read and write registers. Generally, the I²C Status and Control Register should only be monitored after the I²C interrupt, as all bits are valid at that time. Polling this register at other times could read misleading bit status if a transaction is underway.

The I²C clock (SCL) is connected to bit 0 of GPIO port 1, and the I²C SDA data is connected to bit 1 GPIO port 1. The port selection is determined by settings in the I²C Port Configuration Register (Section). Once the I²C-compatible functionality is enabled by setting the I²C Enable bit of the I²C Status and Control Register (bit 0, Figure 16), the two LSB ([1:0]) of the corresponding GPIO port is placed in Open Drain mode, regardless of the settings of the GPIO Configuration Register. In Open Drain mode, the GPIO pin outputs LOW if the pin's Data Register is '0', and the pin is in Hi-Z mode if the pin's Data Register is '1'. The electrical characteristics of the I²C-compatible interface is the same as that of GPIO port 1. Note that the I_{OL} (max) is 2 mA @ V_{OL} = 2.0V for port 1.

All control of the I²C clock (SCL) and data (SDA) lines is performed by the I²C-compatible block.

Figure 15. I²C Data Register

I2C Data **Address 0x29**

Bit #	7	6	5	4	3	2	1	0
Bit Name	I²C Data 7	I²C Data 6	I²C Data 5	I²C Data 4	I²C Data 3	I²C Data 2	I²C Data 1	I²C Data 0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	X	X	X	X	X	X	X	X

Note
3. I²C-compatible function must be separately enabled, as described in Section .

start bits, as these cases always cause transmit mode for the first byte.

Bit 4: ACK

This bit is set or cleared by firmware during receive operation to indicate if the hardware should generate an ACK signal on the I^2C-compatible bus. Writing a 1 to this bit generates an ACK (SDA LOW) on the I2C-compatible bus at the ACK bit time. During transmits (Xmit Mode = 1), this bit should be cleared.

Bit 3: Addr

This bit is set by the I^2C-compatible block during the first byte of a slave receive transaction, after an I^2C start or restart. The Addr bit is cleared when the firmware sets the Continue bit. This bit allows the firmware to recognize when the master has lost arbitration, and in slave mode it allows the firmware to recognize that a start or restart has occurred.

Bit 2: ARB Lost/Restart

This bit is valid as a status bit (ARB Lost) after master mode transactions. In master mode, set this bit (along with the Continue and MSTR Mode bits) to perform an I^2C restart sequence. The I^2C target address for the restart must be written to the data register before setting the Continue bit. To prevent false ARB Lost signals, the Restart bit is cleared by hardware during the restart sequence.

Bit 1: Receive Stop

This bit is set when the slave is in receive mode and detects a stop bit on the bus. The Receive Stop bit is not set if the firmware terminates the I^2C transaction by not acknowledging the previous byte transmitted on the I^2C-compatible bus, e.g., in receive mode if firmware sets the Continue bit and clears the ACK bit.

Bit 0: I^2C Enable

Set this bit to override GPIO definition with I^2C-compatible function on the two I^2C-compatible pins. When this bit is cleared, these pins are free to function as GPIOs. In I^2C-compatible mode, the two pins operate in open drain mode, independent of the GPIO configuration setting.

Processor Status and Control Register

Figure 17. Processor Status and Control Register

Processor Status and Control **Address 0xFF**

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

Bit 0: Run

This bit is manipulated by the HALT instruction. When Halt is executed, all the bits of the Processor Status and Control Register are cleared to 0. Since the run bit is cleared, the processor stops at the end of the current instruction. The processor remains halted until an appropriate reset occurs (power-on or Watchdog). This bit should normally be written as a '1.'

Bit 1: Reserved

Bit 1 is reserved and must be written as a zero.

Bit 2: Interrupt Enable Sense

This bit indicates whether interrupts are enabled or disabled. Firmware has no direct control over this bit as writing a zero or one to this bit position has no effect on interrupts. A '0' indicates that interrupts are masked off and a '1' indicates that the interrupts are enabled. This bit is further gated with the bit settings of the Global Interrupt Enable Register (*Figure 18*) and USB End Point Interrupt Enable Register (*Figure 19*). Instructions DI, EI, and RETI manipulate the state of this bit.

Bit 3: Suspend

Writing a '1' to the Suspend bit halts the processor and cause the microcontroller to enter the suspend mode that significantly reduces power consumption. A pending, enabled interrupt or USB bus activity causes the device to come out of suspend. After coming out of suspend, the device resumes firmware execution at the instruction following the IOWR which put the part into suspend. An IOWR attempting to put the part into suspend is ignored if USB bus activity is present. See Section for more details on suspend mode operation.

Bit 4: Power-on Reset

The Power-on Reset is set to '1' during a power-on reset. The firmware can check bits 4 and 6 in the reset handler to determine whether a reset was caused by a power-on condition or a Watchdog timeout. A POR event may be followed by a Watchdog reset before firmware begins executing, as explained below.

Bit 5: USB Bus Reset Interrupt

The USB Bus Reset Interrupt bit is set when the USB Bus Reset is detected on receiving a USB Bus Reset signal on the upstream port. The USB Bus Reset signal is a single-ended zero (SE0) that lasts from 12 to 16 μs. An SE0

is defined as the condition in which both the D+ line and the D– line are LOW at the same time.

Bit 6: Watchdog Reset

The Watchdog Reset is set during a reset initiated by the Watchdog Timer. This indicates the Watchdog Timer went for more than t_{WATCH} (8 ms minimum) between Watchdog clears. This can occur with a POR event, as noted below.

Bit 7: IRQ Pending

The IRQ pending, when set, indicates that one or more of the interrupts has been recognized as active. An interrupt remains pending until its interrupt enable bit is set (*Figure 18*, *Figure 19*) and interrupts are globally enabled. At that point, the internal interrupt handling sequence clears this bit until another interrupt is detected as pending.

During power-up, the Processor Status and Control Register is set to 00010001, which indicates a POR (bit 4 set) has occurred and no interrupts are pending (bit 7 clear). During the 96-ms suspend at start-up (explained in Section), a Watchdog Reset also occurs unless this suspend is aborted by an upstream SE0 before 8 ms. If a WDR occurs during the power-up suspend interval, firmware reads 01010001 from the Status and Control Register after power-up. Normally, the POR bit should be cleared so a subsequent WDR can be clearly identified. If an upstream bus reset is received before firmware examines this register, the Bus Reset bit may also be set.

During a Watchdog Reset, the Processor Status and Control Register is set to 01XX0001, which indicates a Watchdog Reset (bit 6 set) has occurred and no interrupts are pending (bit 7 clear). The Watchdog Reset does not effect the state of the POR and the Bus Reset Interrupt bits.

Interrupts

Interrupts are generated by GPIO pins, internal timers, I^2C-compatible operation, internal USB hub and USB traffic conditions. 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.

Figure 18. Global Interrupt Enable Register

Global Interrupt Enable Register **Address 0X20**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Reserved	I^2C Interrupt Enable	GPIO Interrupt Enable	Reserved	USB Hub Interrupt Enable	1.024-ms Interrupt Enable	128-µs Interrupt Enable	USB Bus RST Interrupt Enable
Read/Write	–	R/W	R/W	-	R/W	R/W	R/W	R/W
Reset	–	0	0	X	0	0	0	0

Bit 0: USB Bus RST Interrupt Enable

1 = Enable Interrupt on a USB Bus Reset; 0 = Disable interrupt on a USB Bus Reset (Refer to section).

Bit 1:128-µs Interrupt Enable

1 = Enable Timer interrupt every 128 µs; 0 = Disable Timer Interrupt for every 128 µs.

Bit 2: 1.024-ms Interrupt Enable

1 = Enable Timer interrupt every 1.024 ms; 0 = Disable Timer Interrupt every 1.024 ms.

Bit 3: USB Hub Interrupt Enable

1 = Enable Interrupt on a Hub status change; 0 = Disable interrupt due to hub status change. (Refer to section .)

Bit 4: Reserved.

Bit 5: GPIO Interrupt Enable

1 = Enable Interrupt on falling/rising edge on any GPIO; 0 = Disable Interrupt on falling/rising edge on any GPIO (Refer to section , and .).

Bit 6: I^2C Interrupt Enable

1 = Enable Interrupt on I2C related activity; 0 = Disable I2C related activity interrupt. (Refer to section .)

Bit 7: Reserved

Figure 19. USB Endpoint Interrupt Enable Register.

USB Endpoint Interrupt Enable **Address 0X21**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Reserved	Reserved	Reserved	EPB1 Interrupt Enable	EPB0 Interrupt Enable	EPA2 Interrupt Enable	EPA1 Interrupt Enable	EPA0 Interrupt Enable
Read/Write	–	–	–	R/W	R/W	R/W	R/W	R/W
Reset	–	–	–	0	0	0	0	0

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 executes a minimum of 16 clocks (1+10+5) or a maximum of 20 clocks (5+10+5) after the interrupt is issued. For a 12-MHz internal clock (6-MHz crystal), 20 clock periods is 20/12 MHz = 1.667 µs.

USB Bus Reset Interrupt

The USB Controller recognizes a USB Reset when a Single Ended Zero (SE0) condition persists on the upstream USB port for 12–16 µs. SE0 is defined as the condition in which both the D+ line and the D– line are LOW. A USB Bus Reset may be recognized for an SE0 as short as 12 µs, but is always recognized for an SE0 longer than 16 µs. When a USB Bus Reset is detected, bit 5 of the Processor Status and Control Register *(Figure 17*) is set to record this event. In addition, the controller clears the following registers:

SIE Section:.....USB Device Address Registers (0x10, 0x40)

Hub Section: ..... Hub Ports Connect Status (0x48)

..... Hub Ports Enable (0x49)

.....Hub Ports Speed (0x4A)

.....Hub Ports Suspend (0x4D)

.....Hub Ports Resume Status (0x4E)

..... Hub Ports SE0 Status (0x4F)

.....Hub Ports Data (0x50)

..... Hub Downstream Force (0x51).

A USB Bus Reset Interrupt is generated at the end of the USB Bus Reset condition when the SE0 state is deasserted. If the USB reset occurs during the start-up delay following a POR, the delay is aborted as described in Section .

Timer Interrupt

There are two periodic timer interrupts: the 128-µs interrupt and the 1.024-ms interrupt. The user should disable both timer interrupts before going into the suspend mode to avoid possible conflicts between servicing the timer interrupts first or the suspend request first.

USB Endpoint Interrupts

There are five USB endpoint interrupts, one per endpoint. A USB endpoint interrupt is generated after the USB host writes to a USB endpoint FIFO or after the USB controller sends a packet to the USB host. The interrupt is generated on the last packet of the transaction (e.g., on the host's ACK on an IN transfer, or on the device ACK on an OUT transfer). If no ACK is received during an IN transaction, no interrupt is generated.

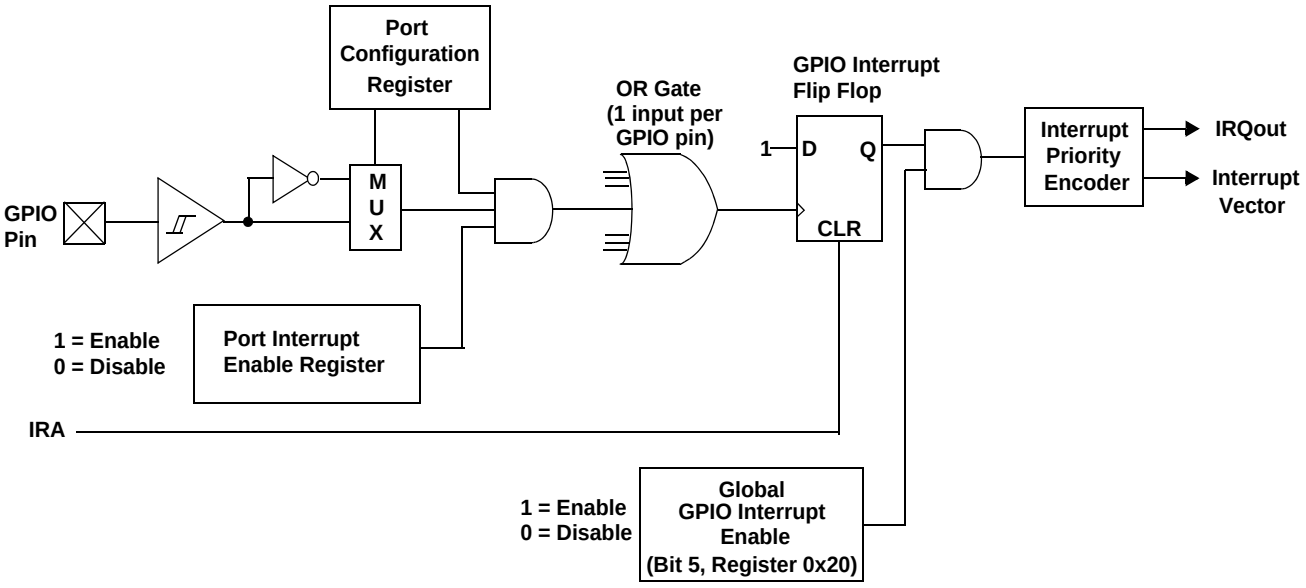
USB Hub Interrupt

A USB hub interrupt is generated by the hardware after a connect/disconnect change, babble, or a resume event is detected by the USB repeater hardware. The babble and resume events are additionally gated by the corresponding bits of the Hub Port Enable Register (*Figure 24*). The connect/disconnect event on a port does not generate an interrupt if the SIE does not drive the port (i.e., the port is being forced).

GPIO Interrupt

Each of the GPIO pins can generate an interrupt, if enabled. The interrupt polarity can be programmed for each GPIO port as part of the GPIO configuration. All of the GPIO pins share a single interrupt vector, which means the firmware needs to read the GPIO ports with enabled interrupts to determine which pin or pins caused an interrupt. A block diagram of the GPIO interrupt logic is shown in Figure .

Figure 21. GPIO Interrupt Structure



Refer to Sections and for more information of setting GPIO interrupt polarity and enabling individual GPIO interrupts. If one port pin has 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 USB Controller does not assign interrupt priority to different port pins and the Port Interrupt Enable Registers are not cleared during the interrupt acknowledge process.

I²C Interrupt

The I²C interrupt occurs after various events on the I²C-compatible bus to signal the need for firmware interaction. This generally involves reading the I²C Status and Control Register (Figure 16) to determine the cause of the interrupt, loading/reading the I²C Data Register as appropriate, and finally writing the Processor Status and Control Register *(Figure 17*) to initiate the subsequent transaction. The interrupt indicates that status bits are stable and it is safe to read and write the I²C registers. Refer to Section for details on the I²C registers.

When enabled, the I²C-compatible state machines generate interrupts on completion of the following conditions. The referenced bits are in the I²C Status and Control Register.

1. In **slave receive** mode, after the slave receives a byte of data: The *Addr* bit is set, if this is the first byte since a start or restart signal was sent by the external master. Firmware must read or write the data register as necessary, then set the *ACK, Xmit MODE,* and *Continue/Busy* bits appropriately for the next byte.
2. In **slave receive** mode, after a stop bit is detected: The *Received Stop* bit is set, if the stop bit follows a slave receive transaction where the *ACK* bit was cleared to 0, no stop bit detection occurs.
3. In **slave transmit** mode, after the slave transmits a byte of data: The *ACK* bit indicates if the master that requested the byte acknowledged the byte. If more bytes are to be sent, firmware writes the next byte into the Data Register and then sets the *Xmit MODE* and *Continue/Busy* bits as required.
4. In **master transmit** mode, after the master sends a byte of data. Firmware should load the Data Register if necessary, and set the *Xmit MODE, MSTR MODE*, and *Continue/Busy* bits appropriately. Clearing the *MSTR MODE* bit issues a stop signal to the I²C-compatible bus and return to the idle state.
5. In **master receive** mode, after the master receives a byte of data: Firmware should read the data and set the *ACK* and *Continue/Busy* bits appropriately for the next byte. Clearing the *MSTR MODE* bit at the same time causes the master state machine to issue a stop signal to the I²C-compatible bus and leave the I2C-compatible hardware in the idle state.
6. When the master loses arbitration: This condition clears the *MSTR MODE* bit and sets the *ARB Lost/Restart* bit immediately and then waits for a stop signal on the I²C-compatible bus to generate the interrupt.

The *Continue/Busy* bit is cleared by hardware prior to interrupt conditions 1 to 4. Once the Data Register has been read or written, firmware should configure the other control bits and set the *Continue/Busy* bit for subsequent transactions. Following an interrupt from master mode, firmware should perform only one write to the Status and Control Register that sets the *Continue/Busy* bit, without checking the value of the *Continue/Busy* bit. The Busy bit may otherwise be active and I²C register contents may be changed by the hardware during the transaction, until the I²C interrupt occurs.

Connects are recorded by the time a non-SE0 state lasts for more than 2.5 μs on a downstream port.

When a USB device is disconnected from the Hub, the downstream signal pair eventually floats to a single-ended zero state. The hub repeater recognizes a disconnect once the SE0 state on a downstream port lasts from 2.0 to 2.5 μs. On a disconnect, the corresponding bit in the Hub Ports Connect Status register is cleared, and the Hub Interrupt is generated.

Figure 22. Hub Ports Connect Status

Hub Ports Connect Status **Address 0x48**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Reserved	Reserved	Reserved	Reserved	Port 4 Connect Status	Port 3 Connect Status	Port 2 Connect Status	Port 1 Connect Status
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 [0..3]: Port x Connect Status (where x = 1..4).

When set to 1, Port x is connected; When set to 0, Port x is disconnected.

Bit [4..7]: Reserved.

Set to 0.

The Hub Ports Connect Status register is cleared to zero by reset or USB bus reset, then set to match the hardware configuration by the hub repeater hardware. The Reserved bits [4..7] should always read as '0' to indicate no connection.

Figure 23. Hub Ports Speed

Hub Ports Speed **Address 0x4A**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Reserved	Reserved	Reserved	Reserved	Port 4 Speed	Port 3 Speed	Port 2 Speed	Port 1 Speed
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 [0..3]: Port x Speed (where x = 1..4).

Set to 1 if the device plugged in to Port x is Low Speed; Set to 0 if the device plugged in to Port x is Full Speed.

Bit [4..7]: Reserved.

Set to 0.

The Hub Ports Speed register is cleared to zero by reset or bus reset. This must be set by the firmware on issuing a port reset. The Reserved bits [4..7] should always read as '0.'

Enabling/Disabling a USB Device

After a USB device connection has been detected, firmware must update status change bits in the hub status change data structure that is polled periodically by the USB host. The host responds by sending a packet that instructs the hub to reset and enable the downstream port. Firmware then sets the bit in the Hub Ports Enable register (*Figure 24*), for the downstream port. The hub repeater hardware responds to an enable bit in the Hub Ports Enable register (*Figure 24*) by enabling the downstream port, so that USB traffic can flow to and from that port.

If a port is marked enabled and is not suspended, it receives all USB traffic from the upstream port, and USB traffic from the downstream port is passed to the upstream port (unless babble is detected). Low-speed ports do not receive full-speed traffic from the upstream port.

When firmware writes to the Hub Ports Enable register (*Figure 24*) to enable a port, the port is not enabled until the end of any packet currently being transmitted. If there is no USB traffic, the port is enabled immediately.

When a USB device disconnection has been detected, firmware must update status bits in the hub change status data structure that is polled periodically by the USB host. In suspended mode, a connect or disconnect event generates an interrupt (if the hub interrupt is enabled) even if the port is disabled.

Figure 24. Hub Ports Enable Register

Hub Ports Enable Register **Address 0x49**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Reserved	Reserved	Reserved	Reserved	Port 4 Enable	Port 3 Enable	Port 2 Enable	Port 1 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

Bits[6..0]: Device Address.

Firmware writes this bits during the USB enumeration process to the non-zero address assigned by the USB host.

Bit 7: Device Address Enable.

Must be set by firmware before the SIE can respond to USB traffic to the Device Address.

USB Device Endpoints

The CY7C65113C controller supports up to two addresses and five endpoints for communication with the host. The configuration of these endpoints, and associated FIFOs, is controlled by bits [7,6] of the USB Status and Control Register (Figure). Bit 7 controls the size of the endpoints and bit 6 controls the number of addresses. These configuration options are detailed in Table 10. Endpoint FIFOs are part of user RAM (as shown in *Section*).

Table 10. Memory Allocation for Endpoints

USB Status And Control Register (0x1F) Bits [7, 6]											
[0,0]			[1,0]			[0,1]			[1,1]		
Two USB Addresses: A (3 Endpoints) and B (2 Endpoints)			Two USB Addresses: A (3 Endpoints) and B (2 Endpoints)			One USB Address: A (5 Endpoints)			One USB Address: A (5 Endpoints)		
Label	**Start Address**	**Size**	**Label**	**Start Address**	**Size**	**Label**	**Start Address**	**Size**	**Label**	**Start Address**	**Size**
EPB1	0xD8	8	EPB0	0xA8	8	EPA4	0xD8	8	EPA3	0xA8	8
EPB0	0xE0	8	EPB1	0xB0	8	EPA3	0xE0	8	EPA4	0xB0	8
EPA2	0xE8	8	EPA0	0xB8	8	EPA2	0xE8	8	EPA0	0xB8	8
EPA1	0xF0	8	EPA1	0xC0	32	EPA1	0xF0	8	EPA1	0xC0	32
EPA0	0xF8	8	EPA2	0xE0	32	EPA0	0xF8	8	EPA2	0xE0	32

When the SIE writes data to a FIFO, the internal data bus is driven by the SIE; not the CPU. This causes a short delay in the CPU operation. The delay is three clock cycles per byte. For example, an 8-byte data write by the SIE to the FIFO generates a delay of 2 μs (3 cycles/byte * 83.33 ns/cycle * 8 bytes).

USB Control Endpoint Mode Registers

All USB devices are required to have a control endpoint 0 (EPA0 and EPB0) that is used to initialize and control each USB address. Endpoint 0 provides access to the device configuration information and allows generic USB status and control accesses. Endpoint 0 is bidirectional to both receive and transmit data. The other endpoints are unidirectional, but selectable by the user as IN or OUT endpoints.

The endpoint mode registers are cleared during reset. When USB Status And Control Register Bits [6,7] are set to [0,0] or [1,0], the endpoint zero EPA0 and EPB0 mode registers use the format shown in Figure 33.

Figure 33. USB Device Endpoint Zero Mode Registers

USB Device Endpoint Zero Mode (A0, B0)						Addresses 0x12(A0) and 0x42(B0)	
Bit #	7	6	5	4	3	2	1	0
Bit Name	Endpoint 0 SETUP Received	Endpoint 0 IN Received	Endpoint 0 OUT Received	ACK	Mode Bit 3	Mode Bit 2	Mode Bit 1	Mode Bit 0
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

Bits[3..0]: Mode.

These sets the mode which control how the control endpoint responds to traffic.

Bit 4: ACK.

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

Bit 5: Endpoint 0 OUT Received.

1 = Token received is an OUT token. 0 = Token received is not an OUT token. This bit is set by the SIE to report the type of token received by the corresponding device address is an OUT token. The bit must be cleared by firmware as part of the USB processing.

Bit 6: Endpoint 0 IN Received.

1 = Token received is an IN token. 0 = Token received is not an IN token. This bit is set by the SIE to report the type of token received by the corresponding device address is an IN token. The bit must be cleared by firmware as part of the USB processing.

Bit 7: Endpoint 0 SETUP Received.

1 = Token received is a SETUP token. 0 = Token received is not a SETUP token. This bit is set ONLY by the SIE to

Table 13. Details of Modes for Differing Traffic Conditions (see Table 12 for the decode legend)

SETUP (if accepting SETUPs)																		
Properties of Incoming Packet								**Changes made by SIE to Internal Registers and Mode Bits**										
Mode Bits				**token**	**count**	**buffer**	**dval**	**DTOG**	**DVAL**	**COUNT**	**Setup**	**In**	**Out**	**ACK**	**Mode Bits**				**Response**	**Intr**
See Table 11				Setup	<= 10	data	valid	updates	1	updates	1	UC	UC	1	0	0	0	1	ACK	yes
See Table 11				Setup	> 10	junk	x	updates	updates	updates	1	UC	UC	UC	NoChange				ignore	yes
See Table 11				Setup	x	junk	invalid	updates	0	updates	1	UC	UC	UC	NoChange				ignore	yes
Properties of Incoming Packet								**Changes made by SIE to Internal Registers and Mode Bits**										
Mode Bits				**token**	**count**	**buffer**	**dval**	**DTOG**	**DVAL**	**COUNT**	**Setup**	**In**	**Out**	**ACK**	**Mode Bits**				**Response**	**Intr**
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	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	In	x	UC	x	UC	UC	UC	UC	1	UC	UC	NoChange				Stall	yes
CONTROL WRITE																				
Properties of Incoming Packet								**Changes made by SIE to Internal Registers and Mode Bits**										
Mode Bits				**token**	**count**	**buffer**	**dval**	**DTOG**	**DVAL**	**COUNT**	**Setup**	**In**	**Out**	**ACK**	**Mode Bits**				**Response**	**Intr**
Normal Out/premature status In																				
1	0	1	1	Out	<= 10	data	valid	updates	1	updates	UC	UC	1	1	1	0	1	0	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	1	NoChange				TX 0	yes
NAK Out/premature 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	yes
Status In/extra Out																				
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	yes
CONTROL READ																				
Properties of Incoming Packet								**Changes made by SIE to Internal Registers and Mode Bits**										
Mode Bits				**token**	**count**	**buffer**	**dval**	**DTOG**	**DVAL**	**COUNT**	**Setup**	**In**	**Out**	**ACK**	**Mode Bits**				**Response**	**Intr**
Normal In/premature 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/premature 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
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/extra In																				

Register Summary

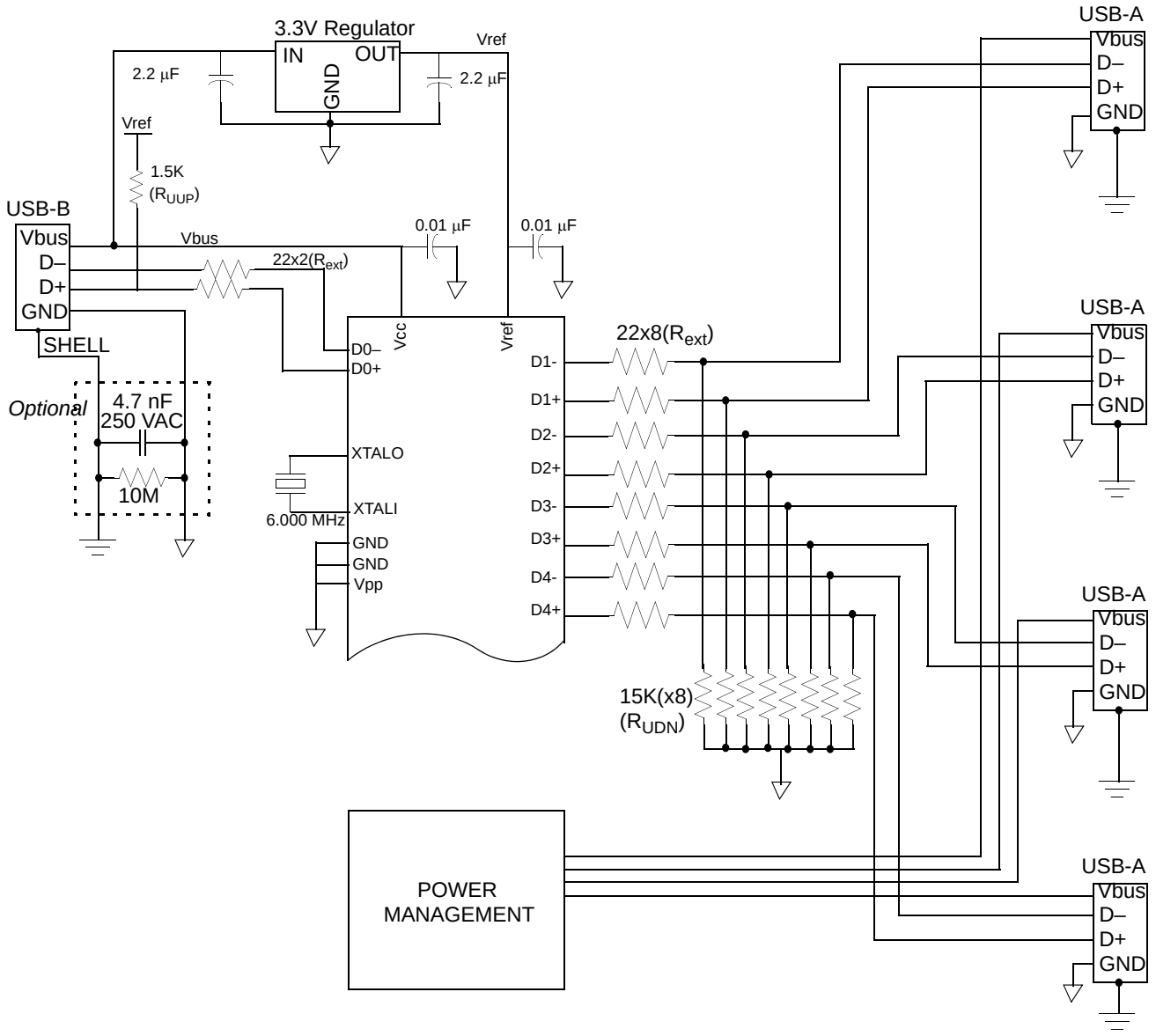
	Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Read/Write/Both/–[7]	Default/Reset
GPIO CONFIGURATION PORTS 0 AND 1	0x00	Port 0 Data	P0.7	P0.6	P0.5	P0.4	P0.3	P0.2	P0.1	P0.0	BBBBBBBB	11111111
	0x01	Port 1 Data	P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0	BBBBBBBB	11111111
	0x02	Port 2 Data	P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0	BBBBBBBB	11111111
	0x03	Port 3 Data	P3.7	P3.6	P3.5	P3.4	P3.3	P3.2	P3.1	P3.0	BBBBBBBB	11111111
	0x04	Port 0 Interrupt Enable	P0.7 Intr Enable	P0.6 Intr Enable	P0.5 Intr Enable	P0.4 Intr Enable	P0.3 Intr Enable	P0.2 Intr Enable	P0.1 Intr Enable	P0.0 Intr Enable	WWWWWWWW	00000000
	0x05	Port 1 Interrupt Enable	P1.7 Intr Enable	P1.6 Intr Enable	P1.5 Intr Enable	P1.4 Intr Enable	Reserved	P1.2 Intr Enable	P1.1 Intr Enable	P1.0 Intr Enable	WWWWWWWW	00000000
	0x08	GPIO Configuration	Reserved	Reserved	Reserved	Reserved	Port 1 Config Bit 1	Port 1 Config Bit 0	Port 0 Config Bit 1	Port 0 Config Bit 0	BBBBBBBB	00000000
HAPI I2C	0x09	HAPI/I^2C Configuration	I^2C Position	Reserved	Reserved	Reserved	Reserved	Reserved	I^2C Port Width	Reserved	BBBBBBBB	00000000
ENDPOINT A0, AI AND A2 CONFIGURATION	0x10	USB Device Address A	Device Address A Enable	Device Address A Bit 6	Device Address A Bit 5	Device Address A Bit 4	Device Address A Bit 3	Device Address A Bit 2	Device Address A Bit 1	Device Address A Bit 0	BBBBBBBB	00000000
	0x11	EP A0 Counter Register	Data 0/1 Toggle	Data Valid	Byte Count Bit 5	Byte Count Bit 4	Byte Count Bit 3	Byte Count Bit 2	Byte Count Bit 1	Byte Count Bit 0	BBBBBBBB	00000000
	0x12	EP A0 Mode Register	Endpoint0 SETUP Received	Endpoint0 IN Received	Endpoint0 OUT Received	ACK	Mode Bit 3	Mode Bit 2	Mode Bit 1	Mode Bit 0	BBBBBBBB	00000000
	0x13	EP A1 Counter Register	Data 0/1 Toggle	Data Valid	Byte Count Bit 5	Byte Count Bit 4	Byte Count Bit 3	Byte Count Bit 2	Byte Count Bit 1	Byte Count Bit 0	BBBBBBBB	00000000
	0x14	EP A1 Mode Register	STALL	-	-	ACK	Mode Bit 3	Mode Bit 2	Mode Bit 1	Mode Bit 0	BBBBBBBB	00000000
	0x15	EP A2 Counter Register	Data 0/1 Toggle	Data Valid	Byte Count Bit 5	Byte Count Bit 4	Byte Count Bit 3	Byte Count Bit 2	Byte Count Bit 1	Byte Count Bit 0	BBBBBBBB	00000000
	0x16	EP A2 Mode Register	STALL	-	-	ACK	Mode Bit 3	Mode Bit 2	Mode Bit 1	Mode Bit 0	BBBBBBBB	00000000
USB-CS	0x1F	USB Status and Control	Endpoint Size	Endpoint Mode	D+ Upstream	D– Upstream	Bus Activity	Control Bit 2	Control Bit 1	Control Bit 0	BBRRBBBB	-0xx0000
INTERRUPT	0x20	Global Interrupt Enable	Reserved	I^2C Interrupt Enable	GPIO Interrupt Enable	Reserved	USB Hub Interrupt Enable	1.024-ms Interrupt Enable	128-µs Interrupt Enable	USB Bus RESET Interrupt Enable	-BBBBBBB	-0000000
	0x21	Endpoint Interrupt Enable	Reserved	Reserved	Reserved	EPB1 Interrupt Enable	EPB0 Interrupt Enable	EPA2 Interrupt Enable	EPA1 Interrupt Enable	EPA0 Interrupt Enable	---BBBBB	---00000
TIMER	0x24	Timer (LSB)	Timer Bit 7	Timer Bit 6	Timer Bit 5	Timer Bit 4	Timer Bit 3	Timer Bit 2	Timer Bit 1	Timer Bit 0	RRRRRRRR	00000000
	0x25	Timer (MSB)	Reserved	Reserved	Reserved	Reserved	Timer Bit 11	Timer Bit 10	Time Bit 9	Timer Bit 8	----rrrr	----0000
I2C	0x28	I^2C Control and Status	MSTR Mode	Continue/ Busy	Xmit Mode	ACK	Addr	ARB Lost/ Restart	Received Stop	I^2C Enable	BBBBBBBB	00000000
	0x29	I^2C Data	I^2C Data 7	I^2C Data 6	I^2C Data 5	I^2C Data 4	I^2C Data 3	I^2C Data 2	I^2C Data 1	I^2C Data 0	BBBBBBBB	XXXXXXXX
ENDPOINT B0, B1 CONFIGURATION	0x40	USB Device Address B	Device Address B Enable	Device Address B Bit 6	Device Address B Bit 5	Device Address B Bit 4	Device Address B Bit 3	Device Address B Bit 2	Device Address B Bit 1	Device Address B Bit 0	BBBBBBBB	00000000
	0x41	EP B0 Counter Register	Data 0/1 Toggle	Data Valid	Byte Count Bit 5	Byte Count Bit 4	Byte Count Bit 3	Byte Count Bit 2	Byte Count Bit 1	Byte Count Bit 0	BBBBBBBB	00000000
	0x42	EP B0 Mode Register	Endpoint 0 SETUP Received	Endpoint 0 IN Received	Endpoint 0 OUT Received	ACK	Mode Bit 3	Mode Bit 2	Mode Bit 1	Mode Bit 0	BBBBBBBB	00000000
	0x43	EP B1 Counter Register	Data 0/1 Toggle	Data Valid	Byte Count Bit 5	Byte Count Bit 4	Byte Count Bit 3	Byte Count Bit 2	Byte Count Bit 1	Byte Count Bit 0	BBBBBBBB	00000000
	0x44	EP B1 Mode Register	STALL	-	-	ACK	Mode Bit 3	Mode Bit 2	Mode Bit 1	Mode Bit 0	BBBBBBBB	00000000

Note
7. B: Read and Write; W: Write; R: Read.

Register Summary (continued)

	Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Read/Write/Both/–[7]	Default/Reset
HUB PORT CONTROL, STATUS, SUSPEND RESUME, SE0, FORCE LOW	0x48	Hub Port Connect Status	Reserved	Reserved	Reserved	Reserved	Port 4 Connect Status	Port 3 Connect Status	Port 2 Connect Status	Port 1 Connect Status	BBBBBBBB	00000000
HUB PORT CONTROL, STATUS, SUSPEND RESUME, SE0, FORCE LOW	0x49	Hub Port Enable	Reserved	Reserved	Reserved	Reserved	Port 4 Enable	Port 3 Enable	Port 2 Enable	Port 1 Enable	BBBBBBBB	00000000
	0x4A	Hub Port Speed	Reserved	Reserved	Reserved	Reserved	Port 4 Speed	Port 3 Speed	Port 2 Speed	Port 1 Speed	BBBBBBBB	00000000
	0x4B	Hub Port Control (Ports 4:1)	Port 4 Control Bit 1	Port 4 Control Bit 0	Port 3 Control Bit 1	Port 3 Control Bit 0	Port 2 Control Bit 1	Port 2 Control Bit 0	Port 1 Control Bit 1	Port 1 Control Bit 0	BBBBBBBB	00000000
	0x4D	Hub Port Suspend	Device Remote Wakeup	Reserved	Reserved	Reserved	Port 4 Selective Suspend	Port 3 Selective Suspend	Port 2 Selective Suspend	Port 1 Selective Suspend	BBBBBBBB	00000000
	0x4E	Hub Port Resume Status	Reserved	Reserved	Reserved	Reserved	Resume 4	Resume 3	Resume 2	Resume 1	-RRRRRRR	00000000
	0x4F	Hub Port SE0 Status	Reserved	Reserved	Reserved	Reserved	Port 4 SE0 Status	Port 3 SE0 Status	Port 2 SE0 Status	Port 1 SE0 Status	RRRRRRRR	00000000
	0x50	Hub Ports Data	Reserved	Reserved	Reserved	Reserved	Port 4 Diff. Data	Port 3 Diff. Data	Port 2 Diff. Data	Port 1 Diff. Data	RRRRRRRR	00000000
	0x51	Hub Port Force Low (Ports 4:1)	Force Low D+[4]	Force Low D–[4]	Force Low D+[3]	Force Low D–[3]	Force Low D+[2]	Force Low D–[2]	Force Low D+[1]	Force Low D–[1]	BBBBBBBB	00000000
	0xFF	Process Status & Control	IRQ Pending	Watchdog Reset	USB Bus Reset Interrupt	Power-on Reset	Suspend	Interrupt Enable Sense	Reserved	Run	RBBBBRBB	00010001

Sample Schematic



Absolute Maximum Ratings

Storage Temperature ..... –65°C to +150°C

Ambient Temperature with Power Applied..... 0°C to +70°C

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

DC Input Voltage ..... –0.5V to +V_{CC} + 0.5V

DC Voltage applied to Outputs in High Z State –0.5V to +V_{CC} + 0.5V

Power Dissipation..... 500 mW

Static Discharge Voltage ..... > 2000V

Latch-up Current ..... > 200 mA

Max Output Sink Current into Port 0, 1 ..... 60 mA

Max Output Sink Current into DAC[7:2] Pins..... 10 mA

Max Output Source Current from Port 1, 2, 3, 4, 5, 6, 7 30 mA

Switching Characteristics (f_{OSC} = 6.0 MHz)

Parameter	Description	Min.	Max.	Unit
Clock Source				
f_{OSC}	Clock Rate	6 ±0.25%		MHz
t_{cyc}	Clock Period	166.25	167.08	ns
t_{CH}	Clock HIGH time	0.45 t_{CYC}		ns
t_{CL}	Clock LOW time	0.45 t_{CYC}		ns
USB Full-speed Signaling[10]				
t_{rfs}	Transition Rise Time	4	20	ns
t_{ffs}	Transition Fall Time	4	20	ns
t_{rfmfs}	Rise/Fall Time Matching; (t_r/t_f)	90	111	%
$t_{dratefs}$	Full Speed Date Rate	12 ±0.25%		Mb/s
Timer Signals				
t_{watch}	Watchdog Timer Period	8.192	14.336	ms

Note
10. Per Table 7-6 of revision 1.1 of USB specification.