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, HAPI
Number of I/O	31
Voltage - Supply	4V ~ 5.25V
Operating Temperature	0°C ~ 70°C
Mounting Type	Surface Mount
Package / Case	56-BSSOP (0.295", 7.50mm Width)
Supplier Device Package	56-SSOP
Purchase URL	https://www.e-xfl.com/product-detail/infineon-technologies/cy7c66113c-pvxct

Clock

The microcontroller uses an external 6 MHz crystal and an internal oscillator to provide a reference to an internal PLL based clock generator. This technology allows the customer application to use an inexpensive 6 MHz fundamental crystal that reduces the clock related noise emissions (EMI). A PLL clock generator provides the 6, 12, and 48 MHz clock signals for distribution within the microcontroller.

Memory

The CY7C66013C and CY7C66113C have 8 kB of PROM.

Power on Reset, Watchdog, and Free Running Timer

These parts include POR logic, a WDT, and a 12-bit free-running timer. The POR logic detects when power is applied to the device, resets the logic to a known state, and begins executing instructions at PROM address 0x0000. The WDT is used to ensure that the microcontroller recovers after a period of inactivity. The firmware may become inactive for a variety of reasons, including errors in the code or a hardware failure such as waiting for an interrupt that never occurs.

I^2C and HAPI Interface

The microcontroller communicates with external electronics through the GPIO pins. An I^2C compatible interface accommodates a 100 kHz serial link with an external device. There is also a HAPI to transfer data to an external device.

Timer

The free-running 12-bit timer clocked at 1 MHz provides two interrupt sources, 128 µs and 1.024 ms. The timer is used to measure the duration of an event under firmware control by reading the timer at the start of the event and after the event is complete. The difference between the two readings indicates the duration of the event in microseconds. The upper four bits of the timer are latched into an internal register when the firmware reads the lower eight bits. A read from the upper four bits actually reads data from the internal register, instead of the timer. This feature eliminates the need for firmware to try to compensate if the upper four bits increment immediately after the lower eight bits are read.

Interrupts

The microcontroller supports eleven maskable interrupts in the vectored interrupt controller. Interrupt sources include the 128 µs (bit 6) and 1.024 ms (bit 9) outputs from the free-running timer, five USB endpoints, the USB hub, the DAC port, the GPIO ports, and the I^2C compatible master mode interface. The timer bits cause an interrupt (if enabled) when the bit toggles from LOW ‘0’ to HIGH ‘1.’ The USB endpoints interrupt after the USB host has written data to the endpoint FIFO or after the USB controller sends a packet to the USB host. The DAC ports have an additional level of masking that allows the user to select which DAC inputs causes a DAC interrupt. The GPIO ports also have a level of masking to select which GPIO inputs causes a GPIO interrupt. For additional flexibility, the input transition polarity that causes an interrupt is programmable for each pin of the DAC port. Input transition polarity is programmed for each GPIO port as part of the port configuration. The interrupt polarity can be rising edge (‘0’ to ‘1’) or falling edge (‘1’ to ‘0’).

USB

The CY7C66013C and CY7C66113C include an integrated USB Serial Interface Engine (SIE) that supports the integrated peripherals and the hub controller function. The hardware supports up to two USB device addresses with one device address for the hub (two endpoints) and a device address for a compound device (three endpoints). The SIE allows the USB host to communicate with the hub and functions integrated into the microcontroller. The part includes a 1:4 hub repeater with one upstream port and four downstream ports. The USB Hub allows power management control of the downstream ports by using GPIO pins assigned by the user firmware. The user has the option of ganging the downstream ports together with a single pair of power management pins, or providing power management for each port with four pairs of power management pins.

Contents

Instruction Set Summary

Refer to the *CYASM Assembler User's Guide* for more details.

Table 4. Instruction Set Summary

Mnemonic	Operand	Opcode	Cycles
HALT		00	7
ADD A,expr	data	01	4
ADD A,[expr]	direct	02	6
ADD A,[X+expr]	index	03	7
ADC A,expr	data	04	4
ADC A,[expr]	direct	05	6
ADC A,[X+expr]	index	06	7
SUB A,expr	data	07	4
SUB A,[expr]	direct	08	6
SUB A,[X+expr]	index	09	7
SBB A,expr	data	0A	4
SBB A,[expr]	direct	0B	6
SBB A,[X+expr]	index	0C	7
OR A,expr	data	0D	4
OR A,[expr]	direct	0E	6
OR A,[X+expr]	index	0F	7
AND A,expr	data	10	4
AND A,[expr]	direct	11	6
AND A,[X+expr]	index	12	7
XOR A,expr	data	13	4
XOR A,[expr]	direct	14	6
XOR A,[X+expr]	index	15	7
CMP A,expr	data	16	5
CMP A,[expr]	direct	17	7
CMP A,[X+expr]	index	18	8
MOV A,expr	data	19	4
MOV A,[expr]	direct	1A	5
MOV A,[X+expr]	index	1B	6
MOV X,expr	data	1C	4
MOV X,[expr]	direct	1D	5
reserved		1E	
XPAGE		1F	4
MOV A,X		40	4
MOV X,A		41	4
MOV PSP,A		60	4
CALL	addr	50-5F	10
JMP	addr	80-8F	5
CALL	addr	90-9F	10
JZ	addr	A0-AF	5
JNZ	addr	B0-BF	5

Mnemonic	Operand	Opcode	Cycles
NOP		20	4
INC A	acc	21	4
INC X	x	22	4
INC [expr]	direct	23	7
INC [X+expr]	index	24	8
DEC A	acc	25	4
DEC X	x	26	4
DEC [expr]	direct	27	7
DEC [X+expr]	index	28	8
IORD expr	address	29	5
IOWR expr	address	2A	5
POP A		2B	4
POP X		2C	4
PUSH A		2D	5
PUSH X		2E	5
SWAP A,X		2F	5
SWAP A,DSP		30	5
MOV [expr],A	direct	31	5
MOV [X+expr],A	index	32	6
OR [expr],A	direct	33	7
OR [X+expr],A	index	34	8
AND [expr],A	direct	35	7
AND [X+expr],A	index	36	8
XOR [expr],A	direct	37	7
XOR [X+expr],A	index	38	8
IOWX [X+expr]	index	39	6
CPL		3A	4
ASL		3B	4
ASR		3C	4
RLC		3D	4
RRC		3E	4
RET		3F	8
DI		70	4
EI		72	4
RETI		73	8
JC	addr	C0-CF	5
JNC	addr	D0-DF	5
JACC	addr	E0-EF	7
INDEX	addr	F0-FF	14

Address Modes

The CY7C66013C and CY7C66113C microcontrollers support three addressing modes for instructions that require data operands: data, direct, and indexed.

Data (Immediate)

"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 0xD8:

```
MOV A, 0D8h.
```

This instruction requires 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 is the constant "0xD8". 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 described earlier:

```
DSPINIT: EQU 0D8h
MOV A, DSPINIT.
```

Direct

"Direct" address mode is used when the data operand is a variable stored in SRAM. In that case, the one byte address of the variable is encoded in the instruction. As an example, consider an instruction that loads A with the contents of memory address location 0x10:

```
MOV A, [10h].
```

Normally, variable names are assigned to variable addresses using "EQU" statements to improve the readability of the assembler source code. As an example, the following code is equivalent to the example described earlier:

```
buttons: EQU 10h
MOV A, [buttons].
```

Indexed

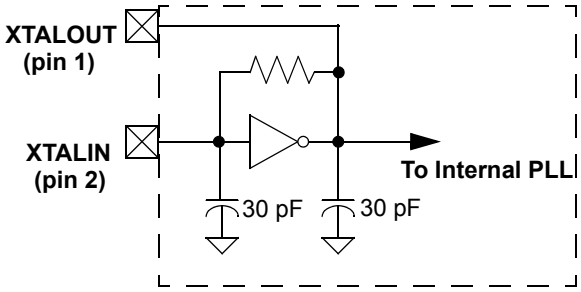
"Indexed" address mode allows the firmware to manipulate arrays of data stored in SRAM. The address of the data operand is the sum of a constant encoded in the instruction and the contents of the "X" register. Normally, the constant is the "base" address of an array of data and the X register contains an index that indicates which element of the array is actually addressed:

```
array: EQU 10h
MOV X, 3
MOV A, [X+array].
```

This has the effect of loading A with the fourth element of the SRAM "array" that begins at address 0x10. The fourth element would be at address 0x13.

Clocking

Figure 4. Clock Oscillator On-Chip Circuit



The XTALIN and XTALOUT are the clock pins to the microcontroller. The user connects 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 is 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 so a ceramic resonator is not recommended with these parts.

An external 6 MHz clock is 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.

GPIO Configuration Port

Every GPIO port is 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 is programmed. The Port Configuration bits (Table 11) and the Interrupt Enable bit (Table 9 on page 17 through Table 16 on page 19) determine the interrupt polarity of the port pins.

Table 11. GPIO Configuration Register

GPIO Configuration **ADDRESS 0x08**

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

As shown in Table 12, 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 (Table 31 on page 28) is enabled, the Interrupt Enable Sense (bit 2, Table 30 on page 27) 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 (Table 7 on page 17 through Table 10 on page 17) and by its associated Port Configuration bits as shown in the GPIO Configuration Register (Table 9 on page 17). 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 12. As shown in this table, when a GPIO port is configured with CMOS outputs, interrupts from that port are disabled.

During reset, all 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 12. 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 here are the transistors referenced in Figure 6 on page 16. 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 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 14 kΩ 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 is individually enabled or disabled as an interrupt source. The Port 0–3 Interrupt Enable registers provide this feature with an interrupt enable bit for each GPIO pin. When HAPI mode is enabled the GPIO interrupts are blocked, including ports not used by HAPI, so GPIO pins are not used as interrupt sources.

During a reset, GPIO interrupts are disabled by clearing all of the GPIO interrupt enable ports. 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 GPIO and HAPI Interrupt on page 31.

Table 13. 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

Table 14. Port 1 Interrupt Enable

Port 1 Interrupt Enable **ADDRESS 0x05**

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

Table 15. Port 2 Interrupt Enable

Port 2 Interrupt Enable **ADDRESS 0x06**

Bit #	7	6	5	4	3	2	1	0
Bit Name	P2.7 Intr Enable	P2.6 Intr Enable	P2.5 Intr Enable	P2.4 Intr Enable	P2.3 Intr Enable	P2.2 Intr Enable	P2.1 Intr Enable	P2.0 Intr Enable
Read/Write	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Table 16. Port 3 Interrupt Enable

Port 3 Interrupt Enable **ADDRESS 0x07**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Reserved	P3.6 Intr Enable CY7C66113C only	P3.5 Intr Enable CY7C66113C only	P3.4 Intr Enable	P3.3 Intr Enable	P3.2 Intr Enable	P3.1 Intr Enable	P3.0 Intr Enable
Read/Write	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Processor Status and Control Register

Table 30. 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 (Table 31 on page 28) and USB End Point Interrupt Enable Register (Table 32 on page 28). 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 Suspend Mode on page 16 for more details on suspend mode operation.

Bit 4: Power on Reset

The POR is set to '1' during a power on reset. The firmware checks 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 WDR before firmware begins executing, as explained here.

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: WDR

The WDR is set during a reset initiated by the WDT. This indicates the WDT went for more than t_{WATCH} (8 ms minimum) between Watchdog clears. This occurs with a POR event.

Bit 7: IRQ Pending

The IRQ pending, when set, indicates that one or more of the interrupts is recognized as active. An interrupt remains pending until its interrupt enable bit is set (Table 31 on page 28, Table 32 on page 28) 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 Power on Reset on page 15), a WDR 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 is clearly identified. If an upstream bus reset is received before firmware examines this register, the Bus Reset bit may also be set.

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

Interrupts

Interrupts are generated by the GPIO and DAC pins, the internal timers, I²C compatible or HAPI operation, the internal USB hub, or on various 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.

Table 31. Global Interrupt Enable Register

Global Interrupt Enable Register **ADDRESS 0X20**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Reserved	I²C Interrupt Enable	GPIO Interrupt Enable	DAC Interrupt Enable	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	R/W
Reset	-	0	0	0	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 USB Bus Reset Interrupt on page 30).

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 USB Hub Interrupt on page 30.)

Bit 4: DAC Interrupt Enable

1 = Enable DAC Interrupt; 0 = Disable DAC interrupt.

Bit 5: GPIO Interrupt Enable

1 = Enable Interrupt on falling and rising edge on any GPIO; 0 = Disable Interrupt on falling and rising edge on any GPIO. (Refer to sections GPIO and HAPI Interrupt on page 31, GPIO Configuration Port on page 18, and GPIO Interrupt Enable Ports on page 19.)

Bit 6: I²C Interrupt Enable

1 = Enable Interrupt on I2C related activity;
0 = Disable I2C related activity interrupt.
(Refer to I²C Interrupt on page 32.)

Bit 7: Reserved.

Table 32. 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

Bit 0: EPA0 Interrupt Enable

1 = Enable Interrupt on data activity through endpoint A0;
0 = Disable Interrupt on data activity through endpoint A0.

Bit 1: EPA1 Interrupt Enable

1 = Enable Interrupt on data activity through endpoint A1;
0 = Disable Interrupt on data activity through endpoint A1.

Bit 2: EPA2 Interrupt Enable

1 = Enable Interrupt on data activity through endpoint A2;
0 = Disable Interrupt on data activity through endpoint A2.

Bit 3: EPB0 Interrupt Enable

1 = Enable Interrupt on data activity through endpoint B0;
0 = Disable Interrupt on data activity through endpoint B0.

Bit 4: EPB1 Interrupt Enable

1 = Enable Interrupt on data activity through endpoint B1;
0 = Disable Interrupt on data activity through endpoint B1.

Bit [7..5]: Reserved

During a reset, the contents the Global Interrupt Enable Register and USB End Point Interrupt Enable Register are cleared, effectively, disabling all interrupts.

The interrupt controller contains a separate flip flop for each interrupt. See Figure 9 on page 29 for the logic block diagram of the interrupt controller. When an interrupt is generated, it is first registered as a pending interrupt. It stays pending until it is serviced or a reset occurs. A pending interrupt only generates an interrupt request if it is enabled by the corresponding bit in the interrupt enable registers. The highest priority interrupt request

is serviced following the completion of the currently executing instruction.

When servicing an interrupt, the hardware does the following:

1. Disables all interrupts by clearing the Global Interrupt Enable bit in the CPU (the state of this bit is read at Bit 2 of the Processor Status and Control Register, Table 30 on page 27).
2. Clears the flip flop of the current interrupt.
3. Generates an automatic CALL instruction to the ROM address associated with the interrupt being serviced (that is, the Interrupt Vector).

The instruction in the interrupt table is typically a JMP instruction to the address of the Interrupt Service Routine (ISR). The user re-enables interrupts in the interrupt service routine by executing an EI instruction. Interrupts are nested to a level limited only by the available stack space.

The Program Counter value and the Carry and Zero flags (CF, ZF) are stored onto the Program Stack by the automatic CALL instruction generated as part of the interrupt acknowledge process. The user firmware is responsible for ensuring that the processor state is preserved and restored during an interrupt.

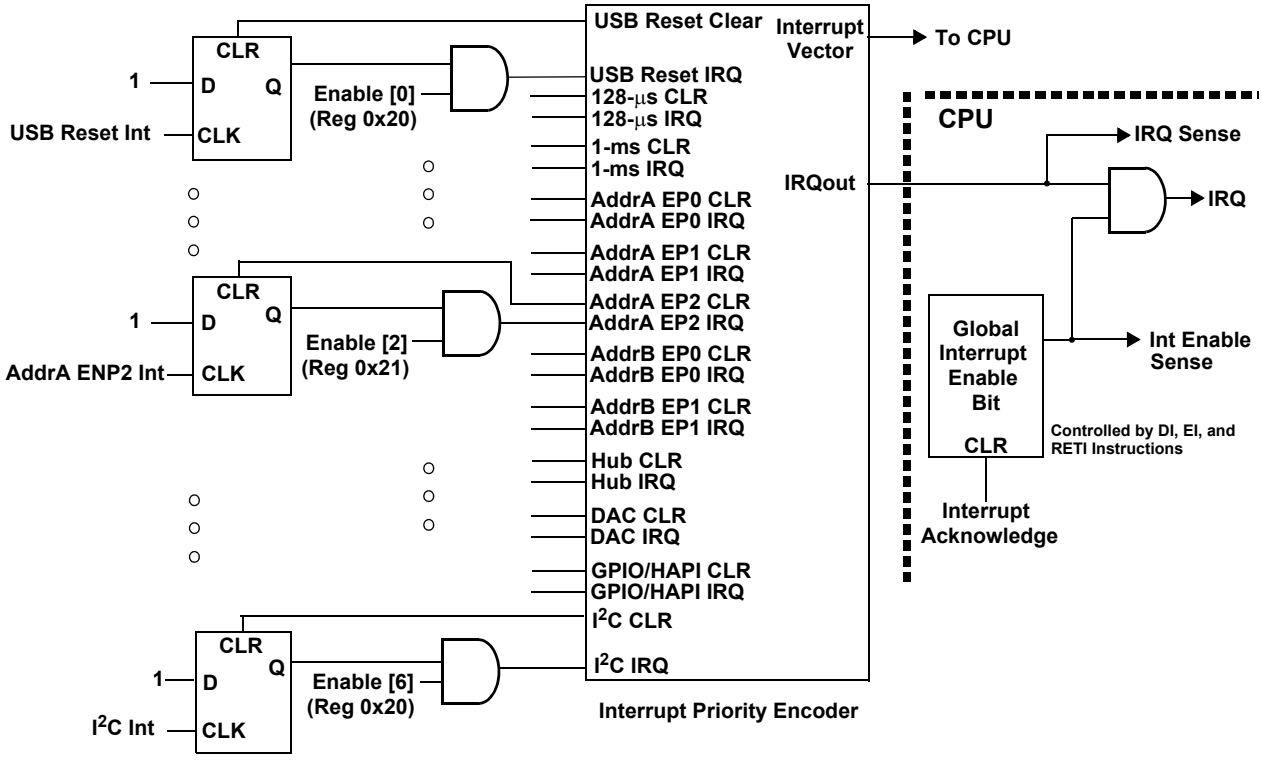
The PUSH A instruction should typically be used as the first command in the ISR to save the accumulator value and the POP A instruction should be used to restore the accumulator value just before the RETI instruction. The program counter CF and ZF are restored and interrupts are enabled when the RETI instruction is executed.

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

Interrupt Vectors

The Interrupt Vectors supported by the USB Controller are listed in Table 33 on page 30. The lowest numbered interrupt (USB Bus Reset interrupt) has the highest priority, and the highest numbered interrupt (I^2C interrupt) has the lowest priority.

Figure 9. Interrupt Controller Function Diagram



5. Firmware stores the new address in its USB Device Address Register (for example, as Address B) after the no data control sequence completes.
6. The host sends a request for the Device descriptor using the new USB address.
7. Firmware decodes the request and retrieves the Device descriptor from program memory tables.
8. The host performs a control read sequence and Firmware responds by sending its Device descriptor over the USB bus.
9. The host generates control reads from the device to request the Configuration and Report descriptors.
10. When the device receives a Set Configuration request, its functions may now be used.
11. Following enumeration as a hub, Firmware optionally indicates to the host that a compound device exists (for example, the keyboard in a keyboard/hub device).
12. The host carries out the enumeration process with this additional function as though it were attached downstream from the hub.
13. When the host assigns an address to this device, it is stored as the other USB address (for example, Address A).

USB Hub

A USB hub is required to support:

- Connectivity behavior: service connect and disconnect detection
- Bus fault detection and recovery
- Full and low speed device support.

These features are mapped onto a hub repeater and a hub controller. The hub controller is supported by the processor integrated into the CY7C66013C and CY7C66113C microcontrollers. The hardware in the hub repeater detects whether a USB device is connected to a downstream port and the interface speed of the downstream device. The connection to a downstream port is through a differential signal pair (D+ and D–). Each downstream port provided by the hub requires external R_{UDN} resistors from each signal line to ground, so that when a downstream port has no device connected, the hub reads a LOW (zero) on both D+ and D–. This condition is used to identify the “no connect” state.

The hub must have a resistor R_{UUP} connected between its upstream D+ line and V_{REG} to indicate it is a full speed USB device.

The hub generates an EOP at EOF1, in accordance with the USB 1.1 Specification, Section 11.2.2.

Connecting and Disconnecting a USB Device

A low speed (1.5 Mbps) USB device has a pull up resistor on the D– pin. At connect time, the bias resistors set the signal levels on the D+ and D– lines. When a low speed device is connected to a hub port, the hub sees a LOW on D+ and a HIGH on D–. This causes the hub repeater to set a connect bit in the Hub Ports Connect Status register for the downstream port. Then the hub repeater generates a Hub Interrupt to notify the microcontroller that there is a change in the Hub downstream status.

A full speed (12 Mbps) USB device has a pull up resistor from the D+ pin, so the hub sees a HIGH on D+ and a LOW on D–. In this case, the hub repeater sets a connect bit in the Hub Ports Connect Status register, clears a bit in the Hub Ports Speed register (for full speed), and generates a Hub Interrupt to notify the microcontroller of the change in Hub status. The firmware sets the speed of this port in the Hub Ports Speed Register (see Table 35 on page 34).

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 when 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.

Table 34. 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 [7..4]: Reserved.

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 [7..4] should always read as ‘0’ to indicate no connection.

Table 35. 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 [7..4]: Reserved.

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 [7..4] should always read as '0'.

Enabling and Disabling a USB Device

After a USB device connection is 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, Table 36, for the downstream port. The hub repeater hardware responds to an enable bit in the Hub Ports Enable register by enabling the downstream port, so that USB traffic flows 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 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 is detected, firmware must update status bits in the hub change status data structure that is polled periodically by the USB host. In suspend, a connect or disconnect event generates an interrupt (if the hub interrupt is enabled) even if the port is disabled.

Table 36. 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

Bit [0..3]: Port x Enable (where x = 1..4)

Set to 1 if Port x is enabled; Set to 0 if Port x is disabled.

Bit [7..4]: Reserved.

The Hub Ports Enable register is cleared to zero by reset or bus reset to disable all downstream ports as the default condition. A port is also disabled by internal hub hardware (enable bit cleared) if babble is detected on that downstream port. Babble is defined as:

- Any non idle downstream traffic on an enabled downstream port at EOF2
- Any downstream port with upstream connectivity established at EOF2 (that is, no EOP received by EOF2).

Table 40. Hub Ports SE0 Status Register

Hub Ports SE0 Status **ADDRESS 0x4F**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Reserved	Reserved	Reserved	Reserved	Port 4 SE0 Status	Port 3 SE0 Status	Port 2 SE0 Status	Port 1 SE0 Status
Read/Write	-	-	-	-	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit [0..3]: Port x SE0 Status (where x = 1..4)

Set to 1 if a SE0 is output on the Port x bus; Set to 0 if a Non-SE0 is output on the Port x bus.

Bit [7..4]: Reserved.

Table 41. Hub Ports Data Register

Hub Ports Data **ADDRESS 0x50**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Reserved	Reserved	Reserved	Reserved	Port 4 Diff. Data	Port 3 Diff. Data	Port 2 Diff. Data	Port 1 Diff. Data
Read/Write	-	-	-	-	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit [0..3]: Port x Diff Data (where x = 1..4)

Set to 1 if D+ > D– (forced differential 1, if signal is differential, i.e. not a SE0 or SE1). Set to 0 if D– > D+ (forced differential 0, if signal is differential, i.e., not a SE0 or SE1);

Bit [7..4]: Reserved.

Downstream Port Suspend and Resume

The Hub Ports Suspend Register (Table 42) and Hub Ports Resume Status Register (Table 49 on page 41) indicate the suspend and resume conditions on downstream ports. The suspend register must be set by firmware for any ports that are selectively suspended. Also, this register is only valid for ports that are selectively suspended.

If a port is marked as selectively suspended, normal USB traffic is not sent to that port. Resume traffic is also prevented from going to that port, unless the Resume comes from the selectively suspended port. If a resume condition is detected on the port, hardware reflects a Resume back to the port, sets the Resume bit in the Hub Ports Resume Register, and generates a hub interrupt. If a disconnect occurs on a port marked as selectively suspended, the suspend bit is cleared.

The Device Remote Wakeup bit (bit 7) of the Hub Ports Suspend Register controls whether or not the resume signal is propagated by the hub after a connect or a disconnect event. If the Device Remote Wakeup bit is set, the hub automatically propagates the resume signal after a connect or a disconnect event. If the Device Remote Wakeup bit is cleared, the hub does not propagate the resume signal. The setting of the Device Remote Wakeup flag has no impact on the propagation of the resume signal after a downstream remote wakeup event. The hub automatically propagates the resume signal after a remote wakeup event, regardless of the state of the Device Remote wakeup bit. The state of this bit has no impact on the generation of the hub interrupt. These registers are cleared on reset or USB bus reset.

Table 42. Hub Ports Suspend Register

Hub Ports Suspend **ADDRESS 0x4D**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Device Remote Wakeup	Reserved	Reserved	Reserved	Port 4 Selective Suspend	Port 3 Selective Suspend	Port 2 Selective Suspend	Port 1 Selective Suspend
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 Selective Suspend (where x = 1..4)

Set to 1 if Port x is Selectively Suspended; Set to 0 if Port x Do not suspend.

Bit 7: Device Remote Wakeup.

When set to 1, Enable hardware upstream resume signaling for connect and disconnect events during global resume.

When set to 0, Disable hardware upstream resume signaling for connect and disconnect events during global resume.

USB Upstream Port Status and Control

USB status and control is regulated by the USB Status and Control Register, as shown in Table 44. All bits in the register are cleared during reset.

Table 44. USB Status and Control Register

USB Status and Control **ADDRESS 0x1F**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Endpoint Size	Endpoint Mode	D+ Upstream	D– Upstream	Bus Activity	Control Action Bit 2	Control Action Bit 1	Control Action Bit 0
Read/Write	R/W	R/W	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits[2..0]: Control Action

Set to control action as per Table 45.The three control bits allow the upstream port to be driven manually by firmware. For normal USB operation, all of these bits must be cleared. Table 45 shows how the control bits affect the upstream port.

Table 45. Control Bit Definition for Upstream Port

Control Bits	Control Action
000	Not Forcing (SIE Controls Driver)
001	Force D+[0] HIGH, D–[0] LOW
010	Force D+[0] LOW, D–[0] HIGH
011	Force SE0; D+[0] LOW, D–[0] LOW
100	Force D+[0] LOW, D–[0] LOW
101	Force D+[0] HiZ, D–[0] LOW
110	Force D+[0] LOW, D–[0] HiZ
111	Force D+[0] HiZ, D–[0] HiZ

Bit 3: Bus Activity

This is a “sticky” bit that indicates if any non idle USB event has occurred on the upstream USB port. Firmware should check and clear this bit periodically to detect any loss of bus activity. Writing a ‘0’ to the Bus Activity bit clears it, while writing a ‘1’ preserves the current value. In other words, the firmware clears the Bus Activity bit, but only the SIE can set it.

Bits 4 and 5: D– Upstream and D+ Upstream

These bits give the state of each upstream port pin individually: 1 = HIGH, 0 = LOW.

Bit 6: Endpoint Mode

This bit used to configure the number of USB endpoints. See USB Device Endpoints on page 39 for a detailed description.

Bit 7: Endpoint Size

This bit used to configure the number of USB endpoints. See USB Device Endpoints on page 39 for a detailed description.

The hub generates an EOP at EOF1 in accordance with the USB 1.1 Specification.

USB SIE Operation

The CY7C66x13C SIE supports operation as a single device or a compound device. This section describes the two device addresses, the configurable endpoints, and the endpoint function.

USB Device Addresses

The USB Controller provides two USB Device Address Registers: A (addressed at 0x10)and B (addressed at 0x40). Upon reset and under default conditions, Device A has three endpoints and Device B has two endpoints. The USB Device Address Register contents are cleared during a reset, setting the USB device addresses to zero and disabling these addresses. Table 46 shows the format of the USB Address Registers.

Table 46. USB Device Address Registers

USB Device Address (Device A, B) **ADDRESSES 0x10(A) and 0x40(B)**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Device Address Enable	Device Address Bit 6	Device Address Bit 5	Device Address Bit 4	Device Address Bit 3	Device Address Bit 2	Device Address Bit 1	Device Address 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[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 responds to USB traffic to the Device Address.

USB Device Endpoints

The CY7C66x13C 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 (see Table 44 on page 38). Bit 7 controls the size of the endpoints and bit 6 controls the number of addresses. These configuration options are detailed in Table 47. Endpoint FIFOs are part of user RAM (as shown in Data Memory Organization on page 13).

Table 47. 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 End-points) & B (2 Endpoints)			Two USB Addresses: A (3 End-points) &B (2 Endpoints)			One USB Address: A (5 Endpoints)			One USB Address: A (5 Endpoints)		
Label	**Start Ad-dress**	**Size**	**Label**	**Start Ad-dress**	**Size**	**Label**	**Start Ad-dress**	**Size**	**Label**	**Start Ad-dress**	**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 0 EPA0 and EPB0 mode registers use the format shown in Table 48 on page 40.

Table 53. Details of Modes for Differing Traffic Conditions (see Table 52 on page 45 for the decode legend) *(continued)*

Nak In/premature status Out																							
1	1	1	0	Out	2	UC	valid	1	1	updates	UC	UC	1	1	No Change				ACK	yes
1	1	1	0	Out	2	UC	valid	0	1	updates	UC	UC	1	UC	0	0	1	1	Stall	yes
1	1	1	0	Out	!=2	UC	valid	updates	1	updates	UC	UC	1	UC	0	0	1	1	Stall	yes
1	1	1	0	Out	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	No Change				ignore	no
1	1	1	0	Out	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	No Change				ignore	no
1	1	1	0	In	x	UC	x	UC	UC	UC	UC	1	UC	UC	No Change				NAK	yes
Status Out/extra In																							
0	0	1	0	Out	2	UC	valid	1	1	updates	UC	UC	1	1	No Change				ACK	yes
0	0	1	0	Out	2	UC	valid	0	1	updates	UC	UC	1	UC	0	0	1	1	Stall	yes
0	0	1	0	Out	!=2	UC	valid	updates	1	updates	UC	UC	1	UC	0	0	1	1	Stall	yes
0	0	1	0	Out	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	No Change				ignore	no
0	0	1	0	Out	x	UC	invalid	UC	UC	UC	UC	1	UC	UC	No Change				ignore	no
0	0	1	0	In	x	UC	x	UC	UC	UC	UC	1	UC	UC	0	0	1	1	Stall	yes
OUT ENDPOINT																							
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/erroneous In																							
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	No Change				ignore	yes			
1	0	0	1	Out	x	junk	invalid	updates	0	updates	UC	UC	1	UC	No Change				ignore	yes			
1	0	0	1	In	x	UC	x	UC	UC	UC	UC	UC	UC	UC	No Change				ignore	no			
																			(STALL[4] = 0)				
1	0	0	1	In	x	UC	x	UC	UC	UC	UC	UC	UC	UC	No Change				Stall	no			
																			(STALL[4] = 1)				
NAK Out/erroneous In																							
1	0	0	0	Out	<= 10	UC	valid	UC	UC	UC	UC	UC	1	UC	No Change				NAK	yes			
1	0	0	0	Out	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	No Change				ignore	no			
1	0	0	0	Out	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	No Change				ignore	no			
1	0	0	0	In	x	UC	x	UC	UC	UC	UC	UC	UC	UC	No Change				ignore	no			
Isochronous endpoint (Out)																							
0	1	0	1	Out	x	updates	updates	updates	updates	updates	UC	UC	1	1	No Change				RX	yes			
0	1	0	1	In	x	UC	x	UC	UC	UC	UC	UC	UC	UC	No Change				ignore	no			
IN ENDPOINT																							
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/erroneous Out																							
1	1	0	1	Out	x	UC	x	UC	UC	UC	UC	UC	UC	UC	No Change				ignore	no			
																			(STALL[4] = 0)				
1	1	0	1	Out	x	UC	x	UC	UC	UC	UC	UC	UC	UC	No Change				stall	no			
																			(STALL[4] = 1)				
1	1	0	1	In	x	UC	x	UC	UC	UC	UC	1	UC	1	1	1	0	0	ACK (back)	yes			
NAK In/erroneous Out																							
1	1	0	0	Out	x	UC	x	UC	UC	UC	UC	UC	UC	UC	No Change				ignore	no			

Register Summary *(continued)*

	Addr	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Read/Write/Both[5, 6, 7]	Default/Reset[8]
INTERRUPT	0x20	Global Interrupt Enable	Reserved	I²C Interrupt Enable	GPIO Interrupt Enable	DAC Interrupt Enable	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	I²C Control and Status	MSTR Mode	Continue/ Busy	Xmit Mode	ACK	Addr	ARB Lost/ Restart	Received Stop	I²C Enable	bbbbbbbb	00000000
	0x29	I²C Data	I²C Data 7	I²C Data 6	I²C Data 5	I²C Data 4	I²C Data 3	I²C Data 2	I²C Data 1	I²C 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	b--bbbbb	00000000
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	----bbbb	00000000
	0x49	Hub Port Enable	Reserved	Reserved	Reserved	Reserved	Port 4 Enable	Port 3 Enable	Port 2 Enable	Port 1 Enable	----bbbb	00000000
	0x4A	Hub Port Speed	Reserved	Reserved	Reserved	Reserved	Port 4 Speed	Port 3 Speed	Port 2 Speed	Port 1 Speed	----bbbb	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	b---bbbb	00000000
	0x4E	Hub Port Resume Status	Reserved	Reserved	Reserved	Reserved	Resume 4	Resume 3	Resume 2	Resume 1	----rrrr	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	----rrrr	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	----rrrr	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	WDR	USB Bus Reset Interrupt	Power-on Reset	Suspend	Interrupt Enable Sense	Reserved	Run	rbbbbrbb	00010001

Electrical Characteristics *(continued)*

(Fosc = 6 MHz; Operating Temperature = 0 to 70 °C, V_{CC} = 4.0 V to 5.25 V)

Parameter	Description	Conditions	Min	Max	Unit
General Purpose I/O (GPIO)					
R_{up}	pull up Resistance (typical 14 kΩ)		8.0	24.0	kΩ
V_{ITH}	Input Threshold Voltage	All ports, LOW to HIGH edge	20%	40%	V_{CC}
V_H	Input Hysteresis Voltage	All ports, HIGH to LOW edge	2%	8%	V_{CC}
V_{OL}	Port 0,1,2,3 Output Low Voltage	I_{OL} = 3 mA	–	0.4	V
		I_{OL} = 8 mA	–	2.0	V
V_{OH}	Output High Voltage	I_{OH} = 1.9 mA (all ports 0,1,2,3)	2.4	–	V
DAC Interface					
R_{up}	DAC Pull Up Resistance (typical 14 kΩ)		8.0	24.0	kΩ
$I_{sink0(0)}$	DAC[7:2] Sink Current (0)	V_{out} = 2.0 V DC	0.1	0.3	mA
$I_{sink0(F)}$	DAC[7:2] Sink Current (F)	V_{out} = 2.0 V DC	0.5	1.5	mA
$I_{sink1(0)}$	DAC[1:0] Sink Current (0)	V_{out} = 2.0 V DC	1.6	4.8	mA
$I_{sink1(F)}$	DAC[1:0] Sink Current (F)	V_{out} = 2.0 V DC	8	24	mA
I_{range}	Programmed Isink Ratio: max/min	V_{out} = 2.0 V DC[11]	4	6	
T_{ratio}	Tracking Ratio DAC[1:0] to DAC[7:2]	V_{out} = 2.0 V[12]	14	22	
$I_{sinkDAC}$	DAC Sink Current	V_{out} = 2.0 V DC	1.6	4.8	mA
I_{lin}	Differential Nonlinearity	DAC Port[13]		0.6	LSB

Notes
11. Irange: $I_{sinkn}(15)/I_{sinkn}(0)$ for the same pin.
12. $T_{ratio} = I_{sink1}[1:0](n)/I_{sink}0[7:2](n)$ for the same n, programmed.
13. I_{lin} measured as largest step size vs. nominal according to measured full scale and zero programmed values.

Quad Flat Package No Leads (QFN) Package Design Notes

Electrical contact of the part to the Printed Circuit Board (PCB) is made by soldering the leads on the bottom surface of the package to the PCB. Hence, special attention is required to the heat transfer area below the package to provide a good thermal bond to the circuit board. A Copper (Cu) fill is to be designed into the PCB as a thermal pad under the package. Heat is transferred from the FX1 through the device's metal paddle on the bottom side of the package. Heat from here, is conducted to the PCB at the thermal pad. It is then conducted from the thermal pad to the PCB inner ground plane by a 5 x 5 array of via. A via is a plated through hole in the PCB with a finished diameter of 13 mil. The QFN's metal die paddle must be soldered to the PCB's thermal pad. Solder mask is placed on the board top side over each via to resist solder flow into the via. The mask on the top side also minimizes outgassing during the solder reflow process.

For further information on this package design please refer to the application note *Surface Mount Assembly of AMKOR's MicroLeadFrame (MLF) Technology*. This application note can be downloaded from AMKOR's website from the following URL http://www.amkor.com/products/notes_papers/MLF_AppNote_0902.pdf. The application note provides detailed information on board mounting guidelines, soldering flow, rework process, etc.

Figure 20 displays a cross sectional area underneath the package. The cross section is of only one via. The thickness of the solder paste template should be 5 mil. It is recommended that "No Clean" type 3 solder paste is used for mounting the part. Nitrogen purge is recommended during reflow.

Figure 21 is a plot of the solder mask pattern. This pad is thermally connected and is not electrically connected inside the chip. To minimize EMI, this pad should be connected to the ground plane of the circuit board.

Figure 20. Cross Section of the Area Underneath the QFN Package

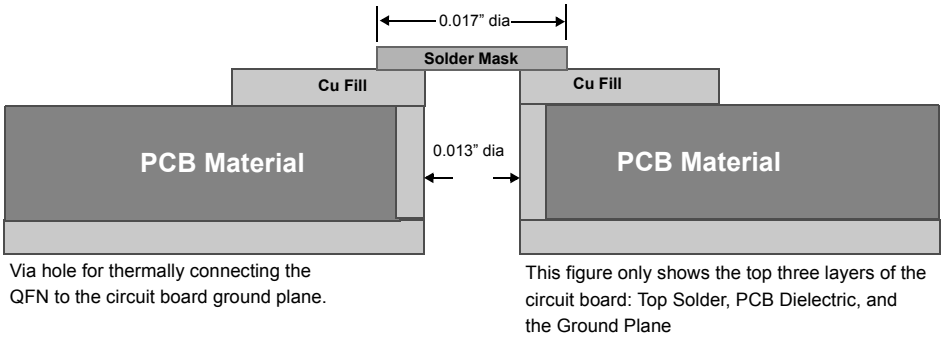
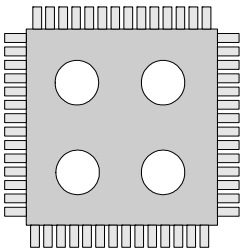


Figure 21. Plot of the Solder Mask (White Area)



Document History Page

Document Title: CY7C66013C/CY7C66113C, Full-Speed USB (12 Mbps) Peripheral Controller with Integrated Hub Document Number: 38-08024				
Rev.	**ECN No.**	**Orig. of Change**	**Submission Date**	**Description of Change**
**	114525	DSG	3/27/02	Change from Spec number: 38-00591 to 38-08024
*A	124768	MON	03/20/03	Added register bit definitions; Added default bit state of each register. Corrected the Schematic (location of the pull-up on D+). Added register summary. Removed information on the availability of the part in PDIP package. Modified Table 51 and provided more explanation regarding locking/unlocking mechanism of the mode register. Removed any information regarding the speed detect bit in Hub Port Speed register being set by hardware.
*B	417632	BHA	See ECN	Updated part number and ordering information. Added QFN Package Drawing and Design Notes. Corrected bit names in Figures 9-3, 9-4, 9-5, 9-8, 9-9, 9-10, 10-5, 16-1, 18-1, 18-2, 18-3, 18-6, 18-7, 18-9, 18-10. Removed Hub Ports Force Low register address 0x52. Added HAPI to Interrupt Vector Number 11 in Table 16-1. Corrected bit names in Section 21.0. Corrected Units in Table 24.0 for R_{UUP}, R_{UDN}, R_{EXT}, and Z_O. Added DIE diagram and related information. Added HAPI to GPIO interrupt vector in Table 5-1 and figure 16-3
*C	1825466	TLY / PYRS	See ECN	Changed Title from “CY7C66013, CY7C66113 Full Speed USB (12 Mbps) Peripheral Controller with Integrated Hub” to “CY7C66013C, CY7C66113C Full Speed USB (12 Mbps) Peripheral Controller with Integrated Hub”. Changed package description for CY7C66013C and CY7C66113C from -PVC to -PVXC
*D	2720540	DPT / AESA	06/18/09	Added 56 QFN 8x8x1 mm package diagram and ordering information
*E	2896318	AESA	03/18/10	Removed Part CY7C66113C-LFXC. Updated all package diagrams.
*F	3057657	AJHA	10/13/10	Added “Not recommended for new designs” watermark in the PDF. No technical or content updates.
*G	3177081	NXZ	02/18/2011	Added Ordering Code Definitions under Ordering Information. Added Acronyms and Units of Measure. Updated to new template.
*H	4313900	AKSL	03/21/2014	Removed “Not recommended for new designs” watermark. Updated Package Diagrams.
*I	5693560	HBM	04/12/2017	Updated to new template. Completing Sunset Review.

Sales, Solutions, and Legal Information

Worldwide Sales and Design Support

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

Products

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

PSoC® Solutions

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

Cypress Developer Community

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

Technical Support

cypress.com/support