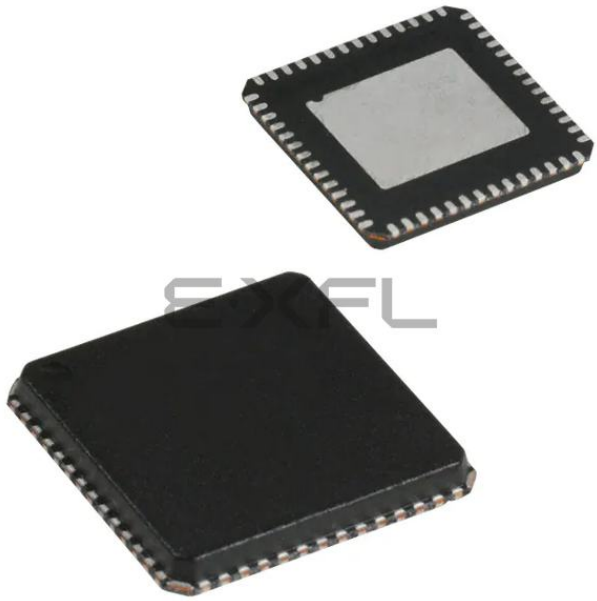


Welcome to E-XFL.COM

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

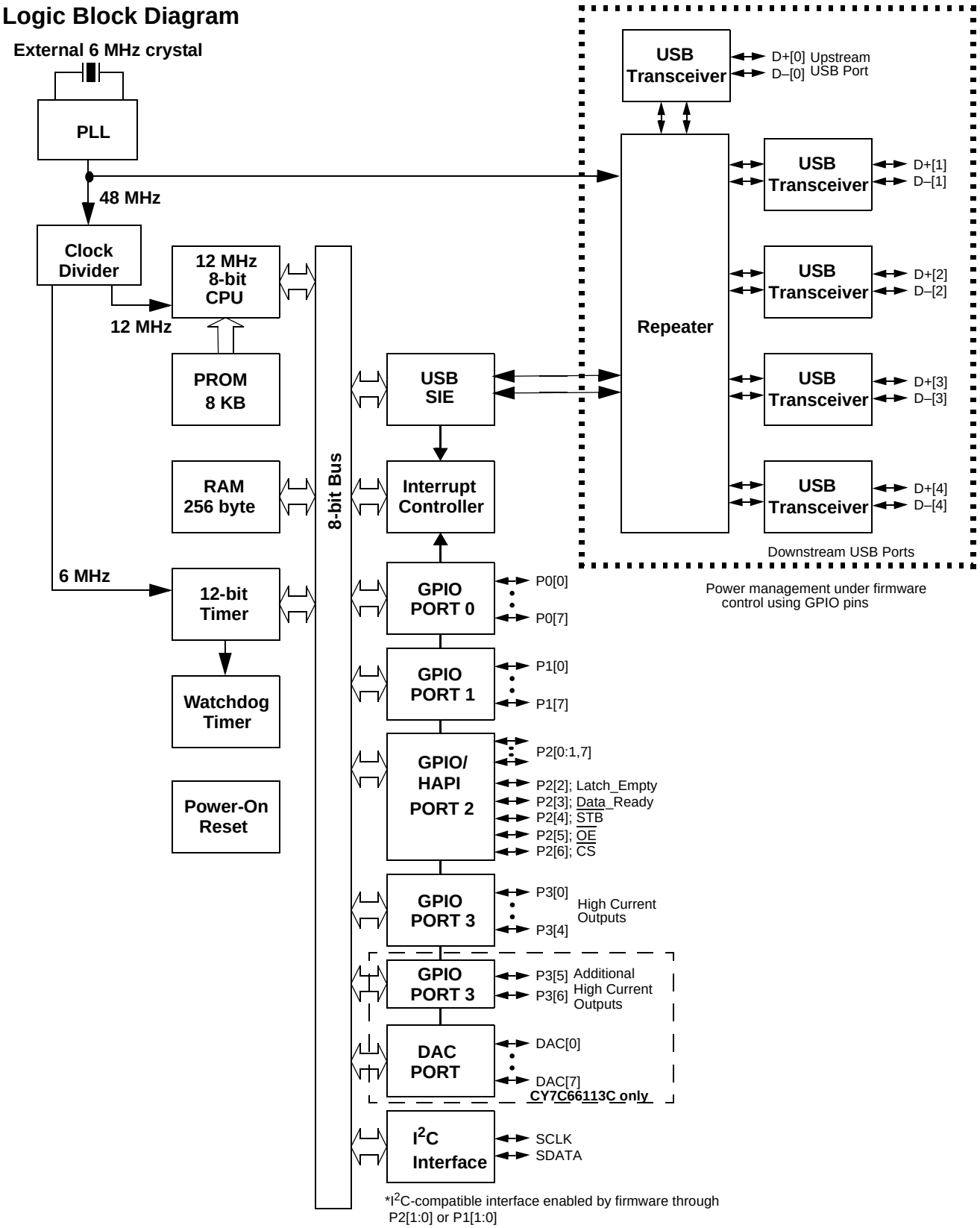
Embedded - Microcontrollers - Application Specific represents a category of microcontrollers designed with unique features and capabilities tailored to specific application needs. Unlike general-purpose microcontrollers, application-specific microcontrollers are optimized for particular tasks, offering 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-VFQFN Exposed Pad
Supplier Device Package	56-QFN (8x8)
Purchase URL	https://www.e-xfl.com/product-detail/infineon-technologies/cy7c66113c-ltxc

Logic Block Diagram

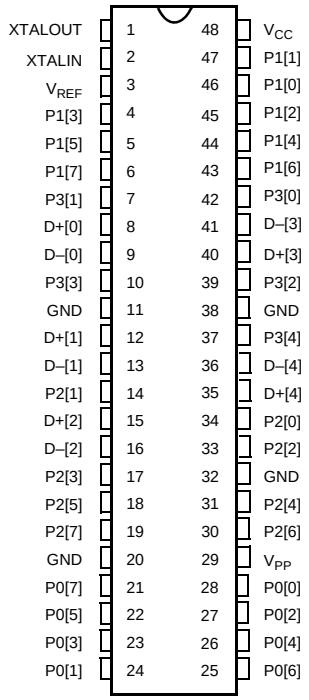


Pin Configurations

Figure 1. CY7C66013C 48-Pin SSOP and CY7C66113C 56-Pin SSOP

TOP VIEW

CY7C66013C



CY7C66113C

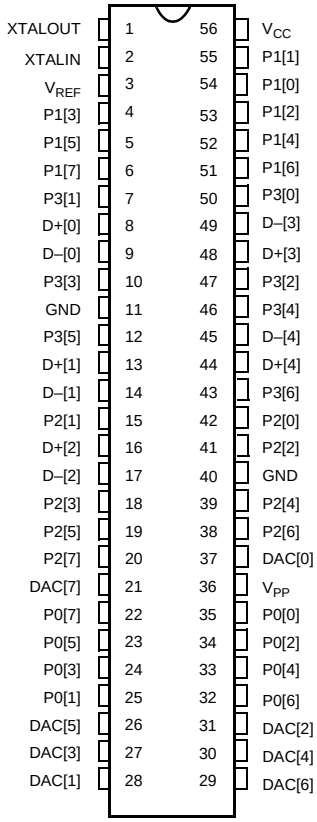


Figure 2. CY7C66113C 56-Pin QFN

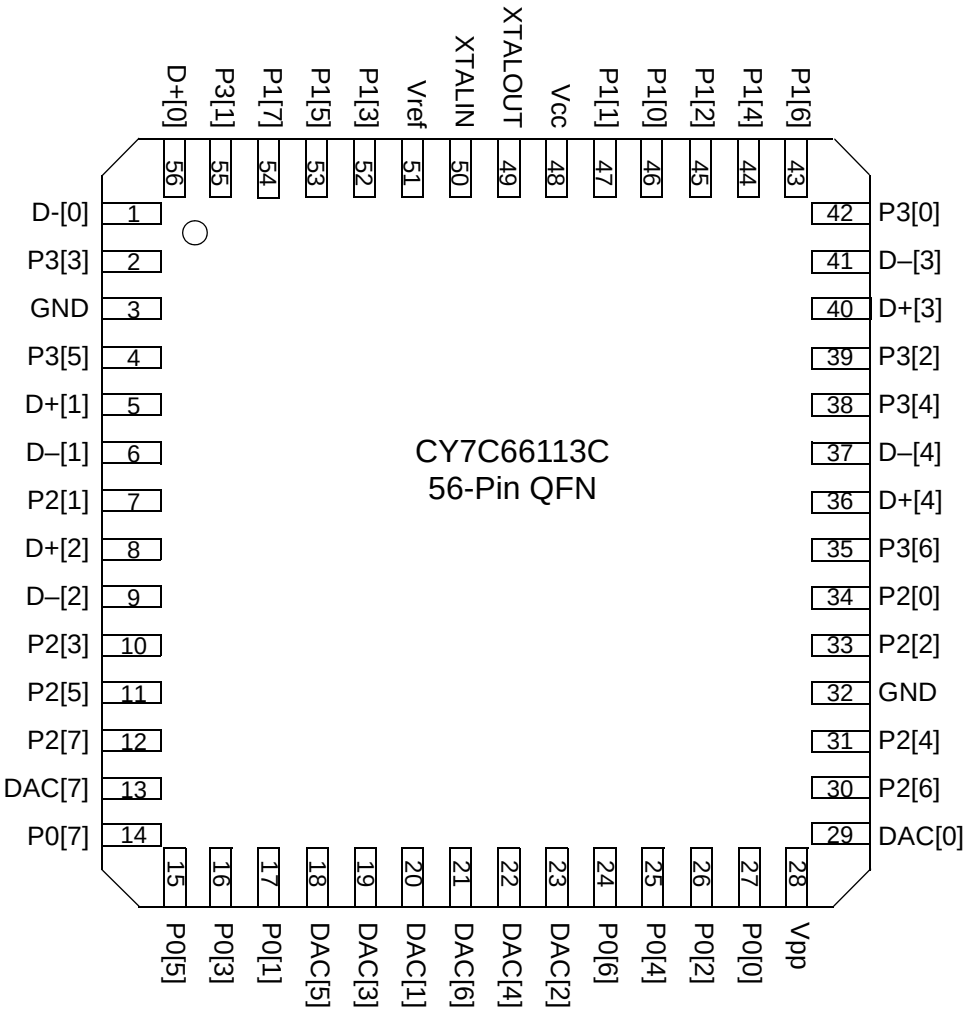


Table 3. I/O Register Summary (continued)

Register Name	I/O Address	Read/Write	Function	Page
HAPI and I^2C Configuration	0x09	R/W	HAPI Width and I^2C Position Configuration	22
USB Device Address A	0x10	R/W	USB Device Address A	38
EP A0 Counter Register	0x11	R/W	USB Address A, Endpoint 0 Counter	40
EP A0 Mode Register	0x12	R/W	USB Address A, Endpoint 0 Configuration	39
EP A1 Counter Register	0x13	R/W	USB Address A, Endpoint 1 Counter	40
EP A1 Mode Register	0x14	R/W	USB Address A, Endpoint 1 Configuration	40
EP A2 Counter Register	0x15	R/W	USB Address A, Endpoint 2 Counter	40
EP A2 Mode Register	0x16	R/W	USB Address A, Endpoint 2 Configuration	40
USB Status & Control	0x1F	R/W	USB Upstream Port Traffic Status and Control	37
Global Interrupt Enable	0x20	R/W	Global Interrupt Enable	27
Endpoint Interrupt Enable	0x21	R/W	USB Endpoint Interrupt Enables	27
Interrupt Vector	0x23	R	Pending Interrupt Vector Read/Clear	29
Timer (LSB)	0x24	R	Lower 8 Bits of Free-running Timer (1 MHz)	21
Timer (MSB)	0x25	R	Upper 4 Bits of Free-running Timer	21
WDT Clear	0x26	W	Watchdog Timer Clear	14
I^2C Control & Status	0x28	R/W	I^2C Status and Control	23
I^2C Data	0x29	R/W	I^2C Data	23
DAC Data	0x30	R/W	DAC Data	19
DAC Interrupt Enable	0x31	W	Interrupt Enable for each DAC Pin	20
DAC Interrupt Polarity	0x32	W	Interrupt Polarity for each DAC Pin	20
DAC Isink	0x38-0x3F	W	Input Sink Current Control for each DAC Pin	20
USB Device Address B	0x40	R/W	USB Device Address B (not used in 5-endpoint mode)	38
EP B0 Counter Register	0x41	R/W	USB Address B, Endpoint 0 Counter	40
EP B0 Mode Register	0x42	R/W	USB Address B, Endpoint 0 Configuration, or USB Address A, Endpoint 3 in 5-endpoint Mode	39
EP B1 Counter Register	0x43	R/W	USB Address B, Endpoint 1 Counter	40
EP B1 Mode Register	0x44	R/W	USB Address B, Endpoint 1 Configuration, or USB Address A, Endpoint 4 in 5-endpoint Mode	40
Hub Port Connect Status	0x48	R/W	Hub Downstream Port Connect Status	32
Hub Port Enable	0x49	R/W	Hub Downstream Ports Enable	33
Hub Port Speed	0x4A	R/W	Hub Downstream Ports Speed	33
Hub Port Control (Ports [4:1])	0x4B	R/W	Hub Downstream Ports Control	34
Hub Port Suspend	0x4D	R/W	Hub Downstream Port Suspend Control	35
Hub Port Resume Status	0x4E	R	Hub Downstream Ports Resume Status	36
Hub Ports SE0 Status	0x4F	R	Hub Downstream Ports SE0 Status	35
Hub Ports Data	0x50	R	Hub Downstream Ports Differential Data	35
Hub Downstream Force Low	0x51	R/W	Hub Downstream Ports Force LOW	34
Processor Status & Control	0xFF	R/W	Microprocessor Status and Control Register	26

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

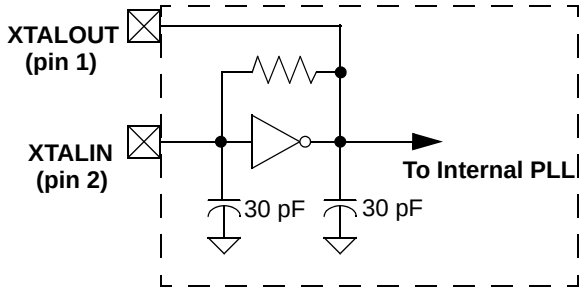
“Indexed” address mode allows the firmware to manipulate arrays of data stored in SRAM. The address of the data operand is the sum of a constant encoded in the instruction and the contents of the “X” register. Normally, the constant is the “base” address of an array of data and the X register contains 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 5. 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.

Suspend Mode

The CY7C66x13C is placed into a low power state by setting the Suspend bit of the Processor Status and Control register. All logic blocks in the device are turned off except the GPIO interrupt logic and the USB receiver. The clock oscillator, PLL, and the free-running and WDTs are shut down. Only the occurrence of an enabled GPIO interrupt or non idle bus activity at a USB upstream or downstream port wakes the part from suspend. The Run bit in the Processor Status and Control Register must be set to resume a part out of suspend.

The clock oscillator restarts immediately after exiting suspend mode. The microcontroller returns to a fully functional state 1 ms after the oscillator is stable. The microcontroller executes the instruction following the I/O write that placed the device into suspend mode before servicing any interrupt requests.

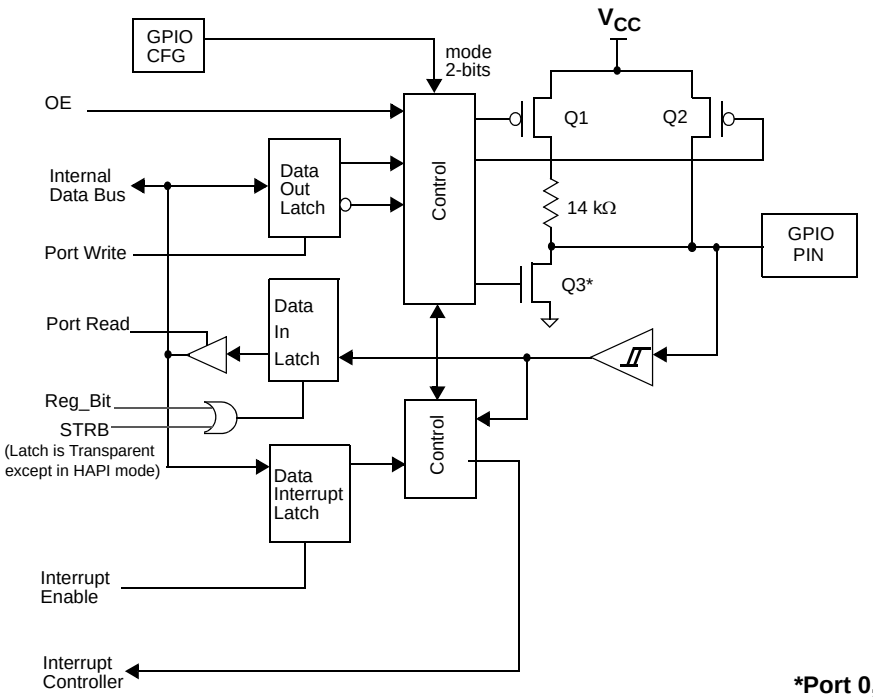
The GPIO interrupt allows the controller to wake up periodically and poll system components while maintaining a very low average power consumption. To achieve the lowest possible current during suspend mode, all I/O should be held at V_{CC} or Gnd. This also applies to internal port pins that may not be bonded in a particular package.

Typical code for entering suspend is given here:

```
     ...  ; All GPIO set to low power state (no
floating pins)
     ...  ; Enable GPIO interrupts if desired
for wakeup
     mov a, 09h; Set suspend and run bits
     iowr FFh; Write to Status and Control
Register – Enter suspend, wait for USB activity
(or GPIO Interrupt)
     nop  ; This executes before any ISR
```

General Purpose I/O (GPIO) Ports

Figure 7. Block Diagram of a GPIO Pin



***Port 0,1,2: Low I_{sink}**
Port 3: High I_{sink}

There are up to 31 GPIO pins (P0[7:0], P1[7:0], P2[7:0], and P3[6:0]) for the hardware interface. The number of GPIO pins changes based on the package type of the chip. Each port is configured as inputs with internal pull ups, open drain outputs, or traditional CMOS outputs. Port 3 offers a higher current drive, with typical current sink capability of 12 mA. The data for each GPIO port is accessible through the data registers. Port data registers are shown in Figure 8 through Figure 11, and are set to 1 on reset.

Figure 8. 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 9. Port1 Data

Port 1Data **ADDRESS 0x01**

Bit #	7	6	5	4	3	2	1	0
Bit Name	P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0
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 10. Port 2 Data

Port 2 Data **ADDRESS 0x02**

Bit #	7	6	5	4	3	2	1	0
Bit Name	P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0
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 11. Port 3 Data

Port 3 Data **ADDRESS 0x03**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Reserved	P3.6 CY7C66113C only	P3.5 CY7C66113C only	P3.4	P3.3	P3.2	P3.1	P3.0
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

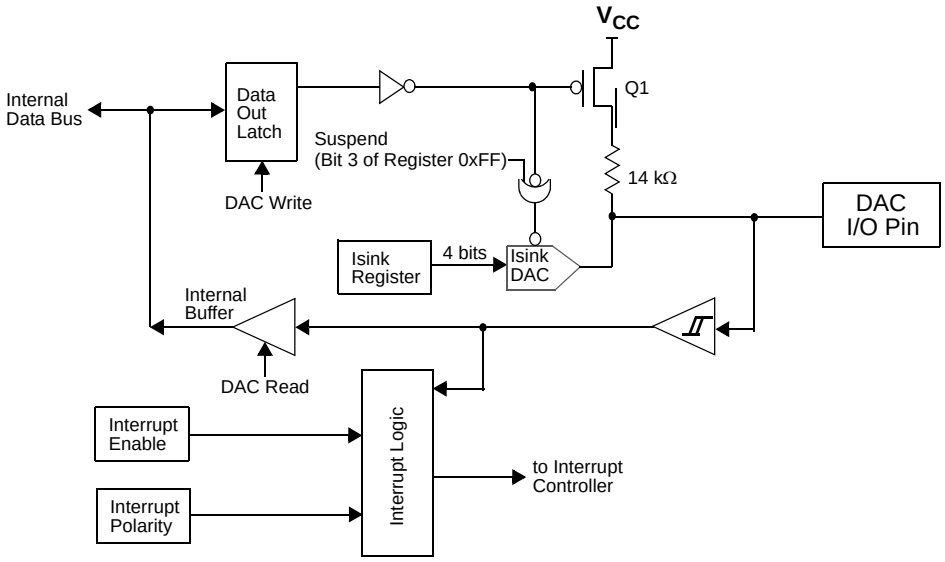
Special care should be taken with any unused GPIO data bits. An unused GPIO data bit, either a pin on the chip or a port bit that is not bonded on a particular package, must not be left floating when the device enters the suspend state. If a GPIO data bit is left floating, the leakage current caused 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.' Notice that the CY7C66013C always requires that P3[7:5] be written with a '0.' When the CY7C66113C is used the P3[7] should be written with a '0.'

In normal non HAPI mode, reads from a GPIO port always return the present state of the voltage at the pin, independent of the settings in the Port Data Registers. If HAPI mode is activated for a port, reads of that port return latched data as controlled by the HAPI signals (see Hardware Assisted Parallel Interface (HAPI)). During reset, all of the GPIO pins are set to a high impedance input state ('1' in open drain mode). 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.

DAC Port

The CY7C66113CC features a programmable sink current 8-bit port, which is also known as DAC port. Each of these port I/O pins have a programmable current sink. Writing a ‘1’ to a DAC I/O pin disables the output current sink (I_{sink} DAC) and drives the I/O pin HIGH through an integrated 14 kΩ resistor. When a ‘0’ is written to a DAC I/O pin, the I_{sink} DAC is enabled and the pull up resistor is disabled. This causes the I_{sink} DAC to sink current to drive the output LOW. Figure 17 shows a block diagram of the DAC port pin.

Figure 17. Block Diagram of a DAC Pin



The amount of sink current for the DAC I/O pin is programmable over 16 values based on the contents of the DAC Isink Register (Figure 19) for that output pin. DAC[1:0] are high current outputs that are programmable from 3.2 mA to 16 mA (typical). DAC[7:2] are low current outputs, programmable from 0.2 mA to 1.0 mA (typical).

When the suspend bit in Processor Status and Control Register (Figure 28) is set, the Isink DAC block of the DAC circuitry is disabled. Special care should be taken when the CY7C66113C device is placed in the suspend. The DAC Port Data Register (Figure 18) should normally be loaded with all ‘1’s (Figure 28) before setting the suspend bit. If any of the DAC bits are set to ‘0’ when the device is suspended, that DAC input floats. The floating pin could result in excessive current consumption by the device, unless an external load places the pin in a deterministic state.

Figure 18. DAC Port Data

DAC Port Data **ADDRESS 0x30**

Bit #	7	6	5	4	3	2	1	0
Bit Name	DAC[7]	DAC[6]	DAC[5]	DAC[4]	DAC[3]	DAC[2]	DAC[1]	DAC[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

Bit [1..0]: High Current Output 3.2 mA to 16 mA typical

1= I/O pin is an output pulled HGH through the 14 kΩ resistor.
0 = I/O pin is an input with an internal 14 kΩ pull up resistor.

Bit [7..2]: Low Current Output 0.2 mA to 1 mA typical

1= I/O pin is an output pulled HGH through the 14 kΩ resistor.
0 = I/O pin is an input with an internal 14 kΩ pull up resistor.

12-Bit Free-Running Timer

The 12-bit timer operates with a 1 µs tick, provides two interrupts (128 µs and 1.024 ms) and allows the firmware to directly time events that are up to 4 ms in duration. The lower eight bits of the timer is read directly by the firmware. Reading the lower 8 bits latches the upper four bits into a temporary register. When the firmware reads the upper four bits of the timer, it is actually reading the count stored in the temporary register. The effect of this is to ensure a stable 12-bit timer value is read, even when the two reads are separated in time.

Figure 22. Timer LSB Register

Timer LSB **ADDRESS 0x24**

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

Bit [7:0]: Timer lower eight bits

Figure 23. Timer MSB Register

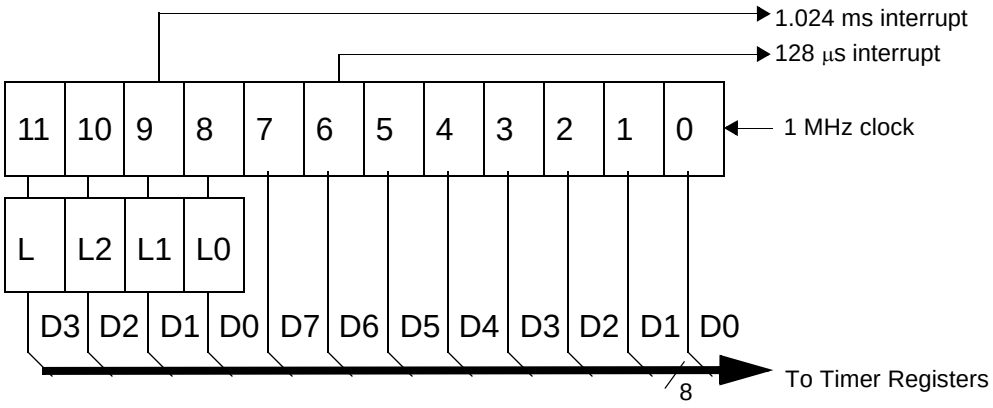
Timer MSB **ADDRESS 0x25**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Reserved	Reserved	Reserved	Reserved	Timer Bit 11	Timer Bit 10	Timer Bit 9	Timer Bit 8
Read/Write	-	-	-	-	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit [3:0]: Timer higher nibble

Bit [7:4]: Reserved

Figure 24. Timer Block Diagram



I²C and HAPI Configuration Register

Internal hardware supports communication with external devices through two interfaces: a two wire I²C compatible, and a HAPI for 1, 2, or 3 byte transfers. The I²C compatible and HAPI functions, share a common configuration register (see Figure 25)[3]. All bits of this register are cleared on reset.

Figure 25. HAPI/I²C Configuration Register

I²C Configuration **ADDRESS 0x09**

Bit #	7	6	5	4	3	2	1	0
Bit Name	I²C Position	Reserved	LEMPTY Polarity	DRDY Polarity	Latch Empty	Data Ready	HAPI Port Width Bit 1	HAPI Port Width Bit 0
Read/Write	R/W	-	R/W	R/W	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits [7,1:0] of the HAPI and I²C Configuration Register control the pin out configuration of the HAPI and I²C compatible interfaces. Bits [5:2] are used in HAPI mode only, and are described in Hardware Assisted Parallel Interface (HAPI). Table 7 shows the HAPI port configurations, and Table 8 shows I²C pin location configuration options. These I²C compatible options exist due to pin limitations in certain packages, and to allow simultaneous HAPI and I²C compatible operation.

HAPI operation is enabled whenever either HAPI Port Width Bit (Bit 1 or 0) is non zero. This affects GPIO operation as described in Hardware Assisted Parallel Interface (HAPI). The I²C compatible interface must be separately enabled.

Table 7. HAPI Port Configuration

Port Width (Bit 0 and 1, Figure 25)	HAPI Port Width
11	24 Bits: P3[7:0], P1[7:0], P0[7:0]
10	16 Bits: P1[7:0], P0[7:0]
01	8 Bits: P0[7:0]
00	No HAPI Interface

Table 8. I²C Port Configuration

I²C Position (Bit 7, Figure 25)	I²C Port Width (Bit 1, Figure 25)	I²C Position
Don't Care	1	I²C on P2[1:0], 0:SCL, 1:SDA
0	0	I²C on P1[1:0], 0:SCL, 1:SDA
1	0	I²C on P2[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 interface 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 Hardware Assisted Parallel Interface (HAPI).

The I²C compatible interface consists of two registers, an I²C Data Register (Figure 14) and an I²C Status and Control Register (Figure 27). The Data Register is implemented as separate read and write registers. Generally, the I²C Status and Control Register are only 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 SCL clock is connected to bit 0 of GPIO port 1 or GPIO port 2, and the I²C SDA data is connected to bit 1 of GPIO port 1 or GPIO port 2. Refer to I²C and HAPI Configuration Register for the bit definitions and functionality of the HAPI and ²C Configuration Register, which is used to set the locations of the configurable I²C pins. When the I²C compatible functionality is enabled by setting bit 0 of the I²C Status & Control Register, 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. The electrical characteristics of the I²C compatible interface is the same as that of GPIO ports 1 and 2. Note that the I_{OL} (max) is 2 mA at V_{OL} = 2.0V for ports 1 and 2.

All control of the I²C clock and data lines is performed by the I²C compatible block.

Note
3. I²C compatible function must be separately enabled.

Figure 34. 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, Figure 35, 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.

Figure 35. 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).

Hub Downstream Ports Status and Control

Data transfer on hub downstream ports is controlled according to the bit settings of the Hub Downstream Ports Control Register (Figure 36). Each downstream port is controlled by two bits, as defined in Table 12. The Hub Downstream Ports Control Register is cleared upon reset or bus reset, and the reset state is the state for normal USB traffic. Any downstream port being forced must be marked as disabled (Figure 35) for proper operation of the hub repeater.

Firmware uses this register for driving bus reset and resume signaling to downstream ports. Controlling the port pins through this register uses standard USB edge rate control according to the speed of the port, set in the Hub Port Speed Register.

The downstream USB ports are designed for connection of USB devices, but also serves as output ports under firmware control. This allows unused USB ports to be used for functions such as driving LEDs or providing additional input signals. Pulling up these pins to voltages above V_{REF} may cause current flow into the pin.

This register is not reset by bus reset. These bits must be cleared before going into suspend.

Figure 36. Hub Downstream Ports Control Register

Hub Downstream Ports Control Register **ADDRESS 0x4B**

Bit #	7	6	5	4	3	2	1	0
Bit Name	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
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 12. Control Bit Definition for Downstream Ports

Control Bits		Control Action
Bit1	Bit 0	
0	0	Not Forcing (Normal USB Function)
0	1	Force Differential ‘1’ (D+ HIGH, D– LOW)
1	0	Force Differential ‘0’ (D+ LOW, D– HIGH)
1	1	Force SE0 state

An alternate means of forcing the downstream ports is through the Hub Ports Force Low Register (Figure 37). With these registers the pins of the downstream ports are individually forced LOW, or left unforced. Unlike the Hub Downstream Ports Control Register, above, the Force Low Register does not produce standard USB edge rate control on the forced pins. However, this register allows downstream port pins to be held LOW in suspend. This register is used to drive SE0 on all downstream ports when unconfigured, as required in the USB 1.1 specification.

Figure 37. Hub Ports Force Low Register

Hub Ports Force Low **ADDRESS 0x51**

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

The data state of downstream ports are read through the HUB Ports SE0 Status Register (Figure 38) and the Hub Ports Data Register (Figure 39). The data read from the Hub Ports Data Register is the differential data only and is independent of the settings of the Hub Ports Speed Register (Figure 34). When the SE0 condition is sensed on a downstream port, the corresponding bits of the Hub Ports Data Register hold the last differential data state before the SE0. Hub Ports SE0 Status Register and Hub Ports Data Register are cleared upon reset or bus reset.

Endpoint Mode and Count Registers Update and Locking Mechanism

The contents of the endpoint mode and counter registers are updated, based on the packet flow diagram in Figure 47. Two time points, UPDATE and SETUP, are shown in the same figure. The following activities occur at each time point:

SETUP:

The SETUP bit of the endpoint 0 mode register is forced HIGH at this time. This bit is forced HIGH by the SIE until the end of the data phase of a control write transfer. The SETUP bit can not be cleared by firmware during this time.

The affected mode and counter registers of endpoint 0 are locked from any CPU writes when they are updated. These registers are unlocked by a CPU read, only if the read operation occurs after the UPDATE. The firmware needs to perform a register read as a part of the endpoint ISR processing to unlock the effected registers. The locking mechanism on mode and counter registers ensures that the firmware recognizes the changes that the SIE might have made since the previous I/O read of that register.

UPDATE:

1. Endpoint Mode Register – All the bits are updated (except the SETUP bit of the endpoint 0 mode register).
2. Counter Registers – All bits are updated.
3. Interrupt – If an interrupt is to be generated as a result of the transaction, the interrupt flag for the corresponding endpoint is set at this time. For details on what conditions are required to generate an endpoint interrupt, refer to Table 16.
4. The contents of the updated endpoint 0 mode and counter registers are locked, except the SETUP bit of the endpoint 0 mode register which was locked earlier.

USB Mode Tables

Table 15. USB Register Mode Encoding

Mode	Mode Bits	SETUP	IN	OUT	Comments
Disable	0000	ignore	ignore	ignore	Ignore all USB traffic to this endpoint
Nak In/Out	0001	accept	NAK	NAK	Forced from Setup on Control endpoint, from modes other than 0000
Status Out Only	0010	accept	stall	check	For Control endpoints
Stall In/Out	0011	accept	stall	stall	For Control endpoints
Ignore In/Out	0100	accept	ignore	ignore	For Control endpoints
Isochronous Out	0101	ignore	ignore	always	For Isochronous endpoints
Status In Only	0110	accept	TX 0 Byte	stall	For Control Endpoints
Isochronous In	0111	ignore	TX count	ignore	For Isochronous endpoints
Nak Out	1000	ignore	ignore	NAK	Is set by SIE on an ACK from mode 1001 (Ack Out)
Ack Out(STALL[4]=0) Ack Out(STALL[4]=1)	1001 1001	ignore ignore	ignore ignore	ACK stall	On issuance of an ACK this mode is changed by SIE to 1000 (NAK Out)
Nak Out-Status In	1010	accept	TX 0 Byte	NAK	Is set by SIE on an ACK from mode 1011 (Ack Out-Status In)
Ack Out-Status In	1011	accept	TX 0 Byte	ACK	On issuance of an ACK this mode is changed by SIE to 1010 (NAK Out – Status In)
Nak In	1100	ignore	NAK	ignore	Is set by SIE on an ACK from mode 1101 (Ack In)
Ack IN(STALL[4]=0) Ack IN(STALL[4]=1)	1101 1101	ignore ignore	TX count stall	ignore ignore	On issuance of an ACK this mode is changed by SIE to 1100 (NAK In)
Nak In – Status Out	1110	accept	NAK	check	Is set by SIE on an ACK from mode 1111 (Ack In – Status Out)
Ack In – Status Out	1111	accept	TX Count	check	On issuance of an ACK this mode is changed by SIE to 1110 (NAK In – Status Out)

Mode

This lists the mnemonic given to the different modes that are set in the Endpoint Mode Register by writing to the lower nibble (bits 0..3). The bit settings for different modes are covered in the column marked “Mode Bits.” The Status IN and Status OUT represent the Status stage in the IN or OUT transfer involving the control endpoint.

Mode Bits

These column lists the encoding for different modes by setting Bits[3..0] of the Endpoint Mode register. This modes represents how the SIE responds to different tokens sent by the host to an endpoint. For instance, if the mode bits are set to “0001” (NAK IN/OUT), the SIE responds with an

- ACK on receiving a SETUP token from the host
- NAK on receiving an OUT token from the host
- NAK on receiving an IN token from the host

Refer to I2C Compatible Controller on page 22 for more information on SIE functioning.

SETUP, IN, and OUT

These columns shows the SIE’s response to the host on receiving a SETUP, IN, and OUT token depending on the mode set in the Endpoint Mode Register.

A “Check” on the OUT token column, implies that on receiving an OUT token the SIE checks to see whether the OUT packet is of zero length and has a Data Toggle (DTOG) set to ‘1.’ If the DTOG bit is set and the received OUT Packet has zero length, the OUT is ACKed to complete the transaction. If either of this condition is not met the SIE responds with a STALLL or just ignore the transaction.

A “TX Count” entry in the IN column implies that the SIE transmit the number of bytes specified in the Byte Count (bits 3..0 of the Endpoint Count Register) to the host in response to the IN token received.

A “TX0 Byte” entry in the IN column implies that the SIE transmit a zero length byte packet in response to the IN token received from the host.

An “Ignore” in any of the columns means that the device does not send any handshake tokens (no ACK) to the host.

An “Accept” in any of the columns means that the device responds with an ACK to a valid SETUP transaction to the host.

Comments

Some Mode Bits are automatically changed by the SIE in response to certain USB transactions. For example, if the Mode Bits [3:0] are set to '1111' which is ACK IN-Status OUT mode as shown in Table 14, the SIE changes the endpoint Mode Bits [3:0] to NAK IN-Status OUT mode (1110) after ACK’ing a valid status

Note
4. STALL bit is bit 7 of the USB Non Control Device Endpoint Mode registers. For more information, refer to USB Non Control Endpoint Mode Registers on page 40.

Table 17. Details of Modes for Differing Traffic Conditions (see Table 16 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			

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

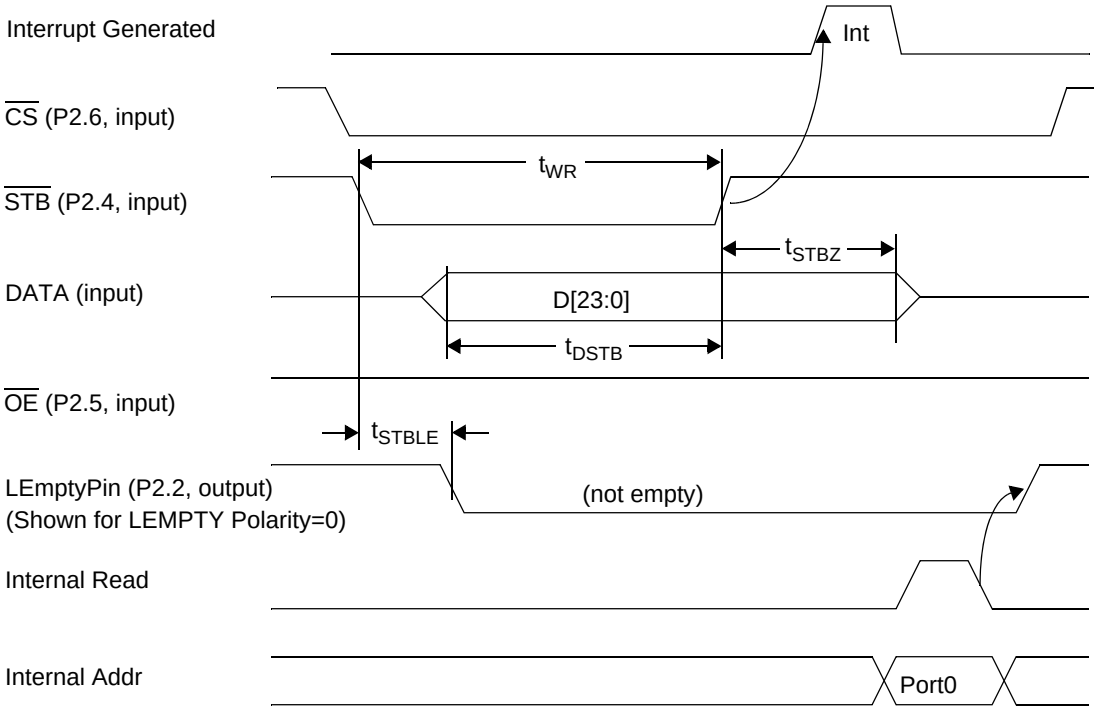
Parameter	Description	Conditions	Min	Max	Unit
V_{OH}	Output High Voltage	I_{OH} = 1.9 mA (all ports 0,1,2,3)	2.4		V
DAC Interface					
R_{up}	DAC Pull Up Resistance (typical 14 kΩ)		8.0	24.0	kΩ
$I_{sink0(0)}$	DAC[7:2] Sink Current (0)	V_{out} = 2.0V DC	0.1	0.3	mA
$I_{sink0(F)}$	DAC[7:2] Sink Current (F)	V_{out} = 2.0V DC	0.5	1.5	mA
$I_{sink1(0)}$	DAC[1:0] Sink Current (0)	V_{out} = 2.0V DC	1.6	4.8	mA
$I_{sink1(F)}$	DAC[1:0] Sink Current (F)	V_{out} = 2.0V DC	8	24	mA
I_{range}	Programmed Isink Ratio: max/min	V_{out} = 2.0V DC[11]	4	6	
T_{ratio}	Tracking Ratio DAC[1:0] to DAC[7:2]	V_{out} = 2.0V[12]	14	22	
$I_{sinkDAC}$	DAC Sink Current	V_{out} = 2.0V DC	1.6	4.8	mA
I_{lin}	Differential Nonlinearity	DAC Port[13]		0.6	LSB

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[14]				
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
DAC Interface				
t_{sink}	Current Sink Response Time		0.8	µs
HAPI Read Cycle Timing				
t_{RD}	Read Pulse Width	15		ns
t_{OED}	OE LOW to Data Valid[15, 16]		40	ns
t_{OEZ}	OE HIGH to Data High-Z[16]		20	ns
t_{OEDR}	OE LOW to Data_Ready Deasserted[15, 16]	0	60	ns

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.
14. Per Table 7-6 of revision 1.1 of USB specification.
15. For 25 pF load.
16. Assumes chip select $\overline{CS}$ is asserted (LOW).

Figure 52. HAPI Write by External Device to USB Microcontroller



Ordering Information

Ordering Code	PROM Size	Package Type	Operating Range
CY7C66013C-PVXC	8 KB	48-pin (300-Mil) SSOP	Commercial
CY7C66113C-PVXC	8 KB	56-pin (300-Mil) SSOP	Commercial
CY7C66113C-LFXC	8 KB	56-pin QFN	Commercial
CY7C66113C-PVXCT	8 KB	56-pin (300-Mil) SSOP	Commercial
CY7C66113C-XC	8 KB	Die	Commercial
CY7C66113C-LTXC	8 KB	56-pin QFN	Commercial
CY7C66113C-LTXCT	8 KB	56-pin QFN	Commercial

Package Diagrams

Figure 53. 48-Pin Shrunk Small Outline Package O48

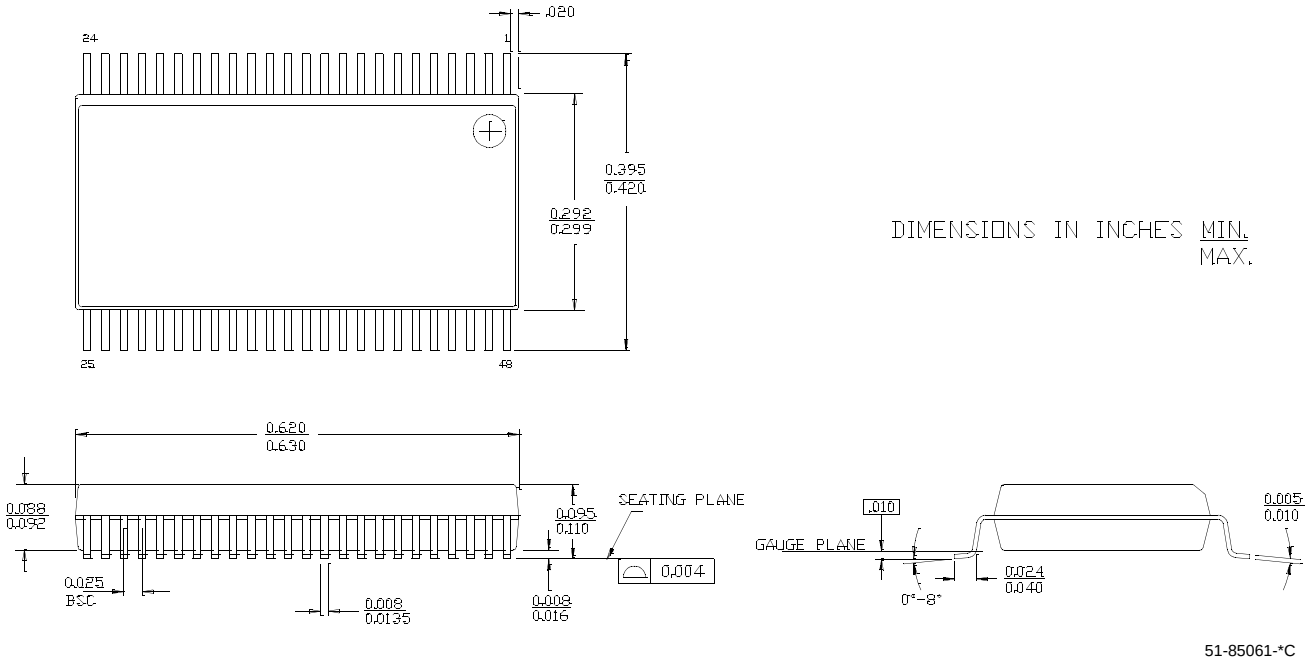


Figure 54. 56-Pin Shrunk Small Outline Package O56

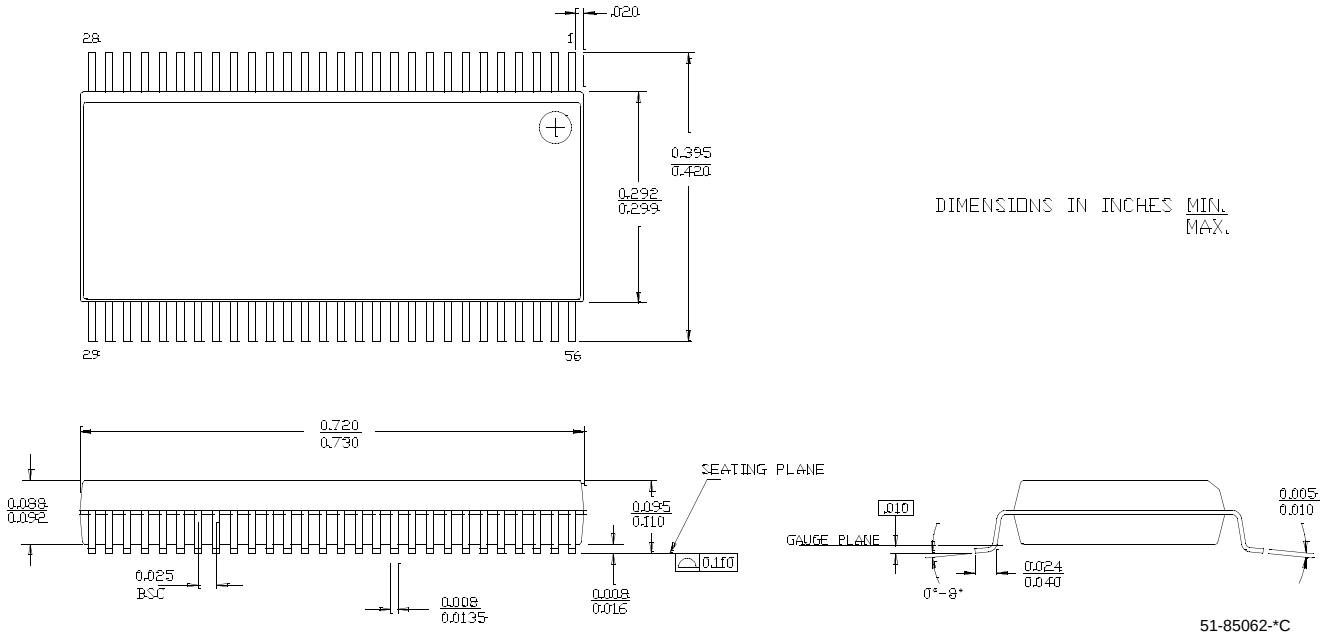
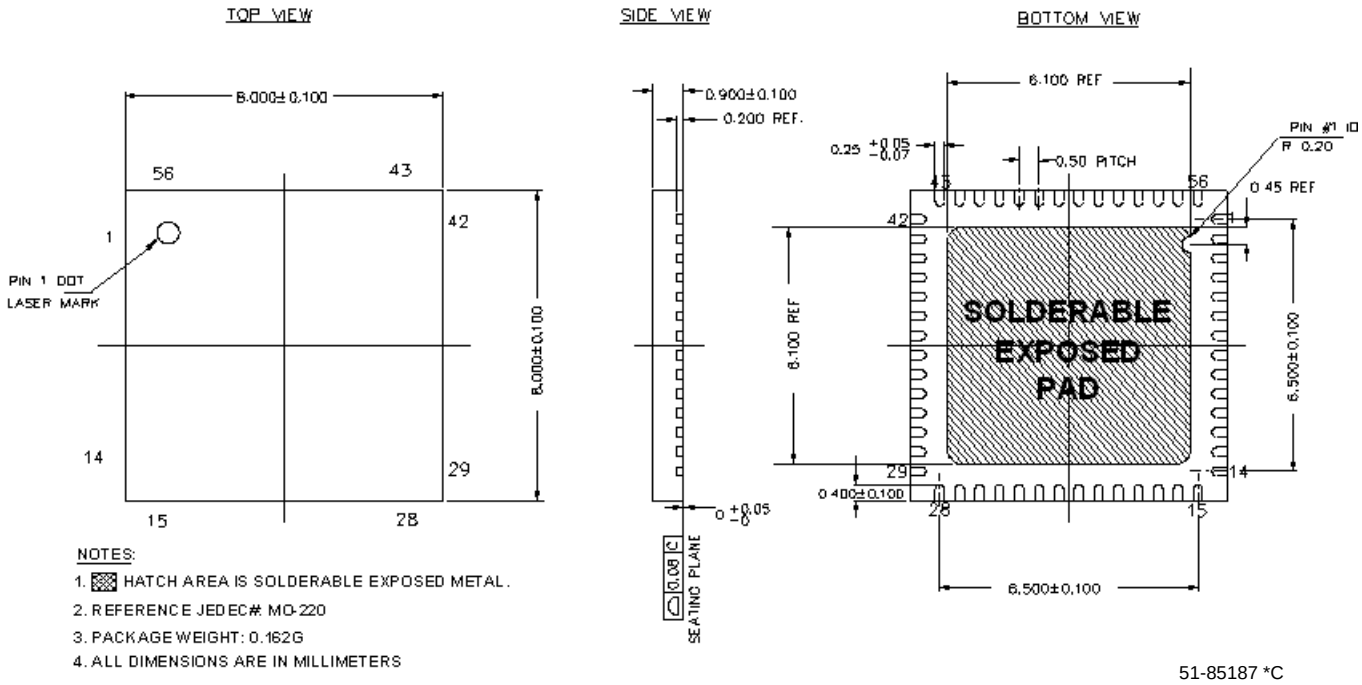


Figure 56. 56-QFN 8x8x0.9 mm EPad 6.1x6.1 mm



Document History Page

Document Title: CY7C66013C, CY7C66113C Full Speed USB (12 Mbps) Peripheral Controller with Integrated Hub Document Number: 38-08024				
Rev.	**ECN No.**	**Submission Date**	**Orig. of Change**	**Description of Change**
**	114525	3/27/02	DSG	Change from Spec number: 38-00591 to 38-08024
*A	124768	03/20/03	MON	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 15 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	See ECN	BHA	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	See ECN	TLY/PYRS	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	06/18/09	DPT/AESA	Added 56 QFN 8x8x1 mm package diagram and ordering information

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.com/sales.

Products			
PSoC	psoc.cypress.com	Wireless	wireless.cypress.com
Clocks & Buffers	clocks.cypress.com	Memories	memory.cypress.com
		Image Sensors	image.cypress.com