Welcome to E-XFL.COM

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

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

What Are Embedded - Microcontrollers - Application Specific?

Application-specific microcontrollers are engineered to

Details	
Product Status	Obsolete
Applications	USB Microcontroller
Core Processor	M8B
Program Memory Type	OTP (8kB)
Controller Series	CY7C635xx
RAM Size	256 x 8
Interface	PS/2, USB
Number of I/O	40
Voltage - Supply	4V ~ 5.5V
Operating Temperature	0°C ~ 70°C
Mounting Type	Surface Mount
Package / Case	48-BSSOP (0.295", 7.50mm Width)
Supplier Device Package	48-SSOP
Purchase URL	https://www.e-xfl.com/product-detail/infineon-technologies/cy7c63513c-pvxct

The Cypress microcontrollers use an external 6-MHz ceramic resonator to provide a reference to an internal clock generator. This clock generator reduces the clock-related noise emissions (EMI). The clock generator provides the 6 and 12-MHz clocks that remain internal to the microcontroller.

The CY7C63413C/513C/613C are offered with single EPROM options. The CY7C63413C, CY7C63513C and the CY7C63613C have 8 Kbytes of EPROM.

These parts include Power-on Reset logic, a Watch Dog Timer, a vectored interrupt controller, and a 12-bit free-running timer. The Power-On Reset (POR) logic detects when power is applied to the device, resets the logic to a known state, and begins executing instructions at EPROM address 0x0000. The Watch Dog Timer can be used to ensure the firmware never gets stalled for more than approximately 8 ms. The firmware can get stalled for a variety of reasons, including errors in the code or a hardware failure such as waiting for an interrupt that never occurs. The firmware should clear the Watch Dog Timer periodically. If the Watch Dog Timer is not cleared for approximately 8 ms, the microcontroller will generate a hardware watch dog reset.

The microcontroller supports eight maskable interrupts in the vectored interrupt controller. Interrupt sources include the USB Bus-Reset, the 128-µs and 1.024-ms outputs from the free-running timer, three USB endpoints, the DAC port, and the GPIO ports. The timer bits cause an interrupt (if enabled) when the bit toggles from LOW “0” to HIGH “1.” The USB endpoints interrupt after either the USB host or the USB controller sends a packet to the USB. The DAC ports have an additional level of masking that allows the user to select which DAC inputs can cause a DAC interrupt. The GPIO ports also have a level of masking to select which GPIO inputs can cause a GPIO interrupt. For additional flexibility, the input transition polarity that causes an interrupt is programmable for each pin of the DAC port. Input transition polarity can be programmed for each GPIO port as part of the port configuration. The interrupt polarity can be either rising edge (“0” to “1”) or falling edge (“1” to “0”).

The free-running 12-bit timer clocked at 1 MHz provides two interrupt sources as noted above (128-µs and 1.024-ms). The timer can be used to measure the duration of an event under firmware control by reading the timer twice: once at the start of the event, and once after the event is complete. The difference between the two readings indicates the duration of the event measured 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 attempt to compensate if the upper four bits happened to increment right after the lower 8 bits are read.

The CY7C63413C/513C/613C include an integrated USB serial interface engine (SIE) that supports the integrated peripherals. The hardware supports one USB device address with three endpoints. The SIE allows the USB host to communicate with the function integrated into the microcontroller.

Finally, the CY7C63413C/513C/613C support PS/2 operation. With appropriate firmware the D+ and D– USB pins can also be used as PS/2 clock and data signals. Products utilizing these devices can be used for USB and/or PS/2 operation with appropriate firmware.

Contents

.

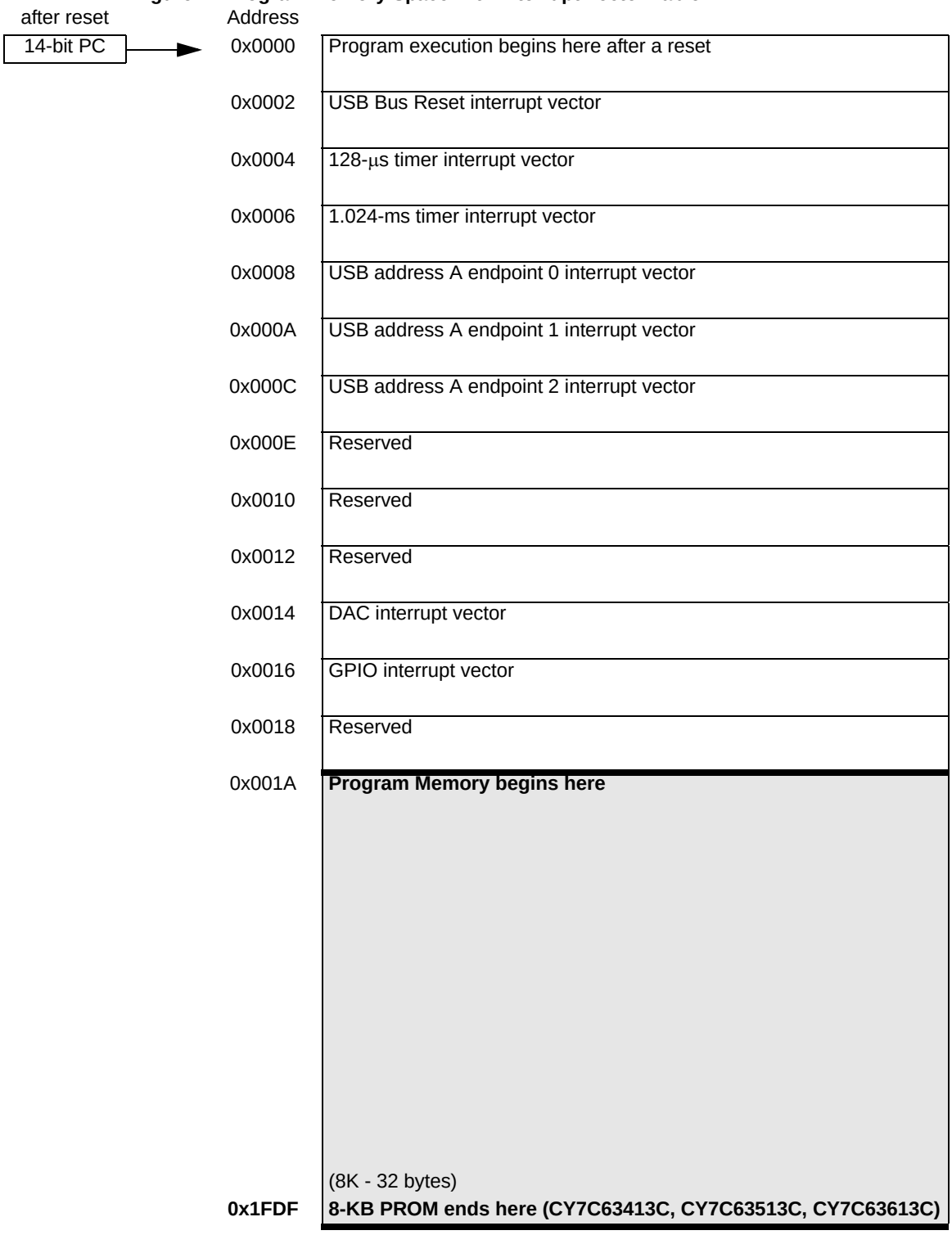
Instruction Set Summary

MNEMONIC	operand	opcode	cycles		MNEMONIC	operand	opcode	cycles
HALT		00	7		NOP		20	4
ADD A,expr	data	01	4		INC A	acc	21	4
ADD A,[expr]	direct	02	6		INC X	x	22	4
ADD A,[X+expr]	index	03	7		INC [expr]	direct	23	7
ADC A,expr	data	04	4		INC [X+expr]	index	24	8
ADC A,[expr]	direct	05	6		DEC A	acc	25	4
ADC A,[X+expr]	index	06	7		DEC X	x	26	4
SUB A,expr	data	07	4		DEC [expr]	direct	27	7
SUB A,[expr]	direct	08	6		DEC [X+expr]	index	28	8
SUB A,[X+expr]	index	09	7		IORD expr	address	29	5
SBB A,expr	data	0A	4		IOWR expr	address	2A	5
SBB A,[expr]	direct	0B	6		POP A		2B	4
SBB A,[X+expr]	index	0C	7		POP X		2C	4
OR A,expr	data	0D	4		PUSH A		2D	5
OR A,[expr]	direct	0E	6		PUSH X		2E	5
OR A,[X+expr]	index	0F	7		SWAP A,X		2F	5
AND A,expr	data	10	4		SWAP A,DSP		30	5
AND A,[expr]	direct	11	6		MOV [expr],A	direct	31	5
AND A,[X+expr]	index	12	7		MOV [X+expr],A	index	32	6
XOR A,expr	data	13	4		OR [expr],A	direct	33	7
XOR A,[expr]	direct	14	6		OR [X+expr],A	index	34	8
XOR A,[X+expr]	index	15	7		AND [expr],A	direct	35	7
CMP A,expr	data	16	5		AND [X+expr],A	index	36	8
CMP A,[expr]	direct	17	7		XOR [expr],A	direct	37	7
CMP A,[X+expr]	index	18	8		XOR [X+expr],A	index	38	8
MOV A,expr	data	19	4		IOWX [X+expr]	index	39	6
MOV A,[expr]	direct	1A	5		CPL		3A	4
MOV A,[X+expr]	index	1B	6		ASL		3B	4
MOV X,expr	data	1C	4		ASR		3C	4
MOV X,[expr]	direct	1D	5		RLC		3D	4
reserved		1E			RRC		3E	4
XPAGE		1F	4		RET		3F	8
MOV A,X		40	4		DI		70	4
MOV X,A		41	4		EI		72	4
MOV PSP,A		60	4		RETI		73	8
CALL	addr	50-5F	10					
JMP	addr	80-8F	5		JC	addr	C0-CF	5
CALL	addr	90-9F	10		JNC	addr	D0-DF	5
JZ	addr	A0-AF	5		JACC	addr	E0-EF	7
JNZ	addr	B0-BF	5		INDEX	addr	F0-FF	14

Memory Organization

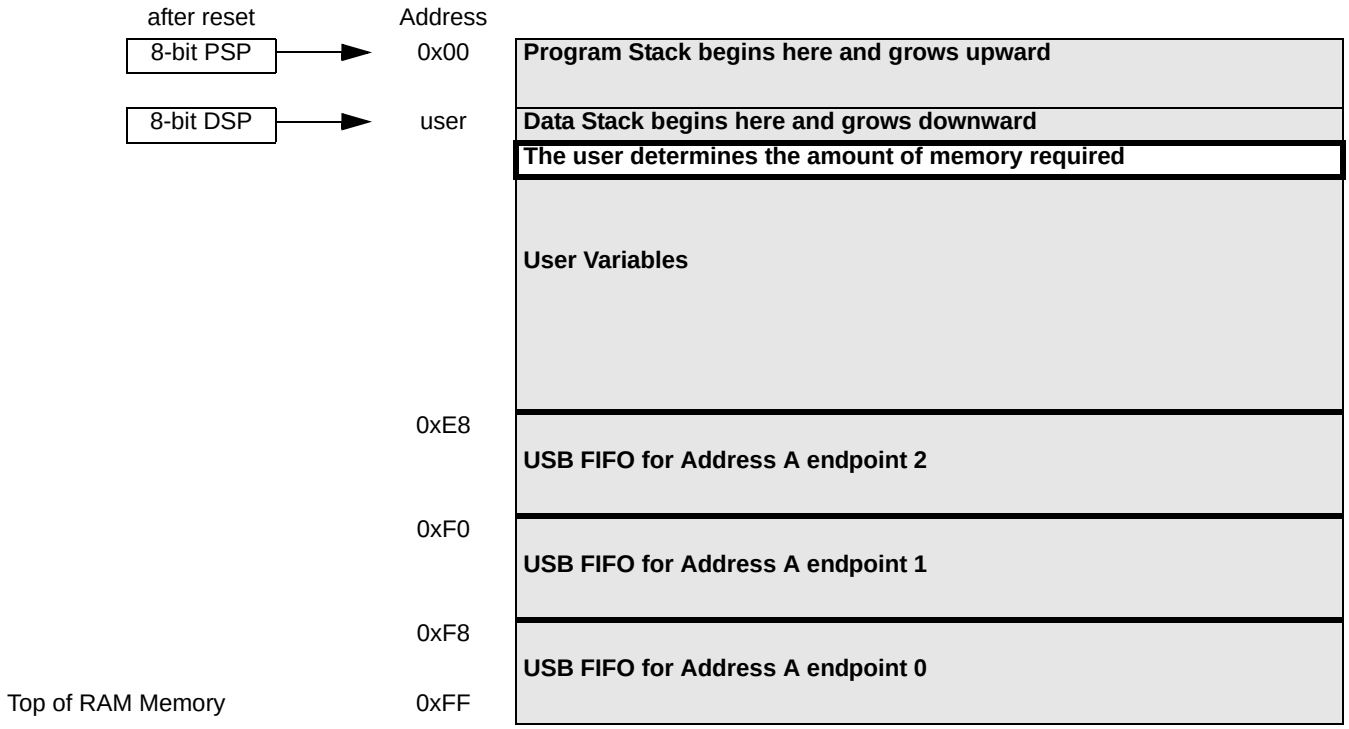
Program Memory Organization

Figure 1. Program Memory Space with Interrupt Vector Table



Data Memory Organization

The CY7C63413C/513C/613C microcontrollers provide 256 bytes of data RAM. In normal usage, the SRAM is partitioned into four areas: program stack, data stack, user variables and USB endpoint FIFOs as shown below:



I/O Register Summary

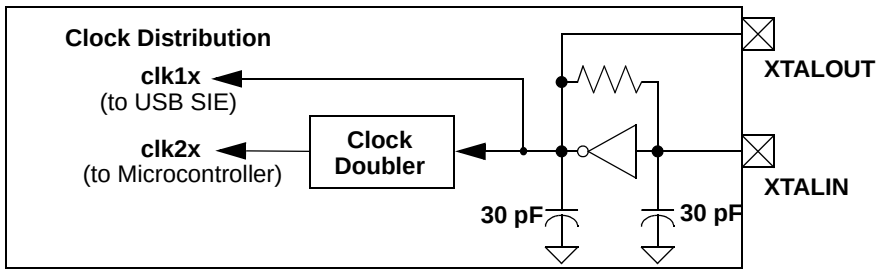
I/O registers are accessed via the I/O Read (IORD) and I/O Write (IOWR, IOWX) instructions. IORD reads the selected port into the accumulator. IOWR writes data from the accumulator to the selected port. Indexed I/O Write (IOWX) adds the contents of X to the address in the instruction to form the port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

Table 1. I/O Register Summary

Register Name	I/O Address	Read/Write	Function
Port 0 Data	0x00	R/W	GPIO Port 0
Port 1 Data	0x01	R/W	GPIO Port 1
Port 2 Data	0x02	R/W	GPIO Port 2
Port 3 Data	0x03	R/W	GPIO Port 3
Port 0 Interrupt Enable	0x04	W	Interrupt enable for pins in Port 0
Port 1 Interrupt Enable	0x05	W	Interrupt enable for pins in Port 1
Port 2 Interrupt Enable	0x06	W	Interrupt enable for pins in Port 2
Port 3 Interrupt Enable	0x07	W	Interrupt enable for pins in Port 3
GPIO Configuration	0x08	R/W	GPIO Ports Configurations
USB Device Address A	0x10	R/W	USB Device Address A
EP A0 Counter Register	0x11	R/W	USB Address A, Endpoint 0 counter register
EP A0 Mode Register	0x12	R/W	USB Address A, Endpoint 0 configuration register
EP A1 Counter Register	0x13	R/W	USB Address A, Endpoint 1 counter register
EP A1 Mode Register	0x14	R/C	USB Address A, Endpoint 1 configuration register
EP A2 Counter Register	0x15	R/W	USB Address A, Endpoint 2 counter register
EP A2 Mode Register	0x16	R/C	USB Address A, Endpoint 2 configuration register
USB Status & Control	0x1F	R/W	USB upstream port traffic status and control register
Global Interrupt Enable	0x20	R/W	Global interrupt enable register
Endpoint Interrupt Enable	0x21	R/W	USB endpoint interrupt enables
Timer (LSB)	0x24	R	Lower eight bits of free-running timer (1 MHz)
Timer (MSB)	0x25	R	Upper four bits of free-running timer that are latched when the lower eight bits are read.
WDR Clear	0x26	W	Watch Dog Reset clear
DAC Data	0x30	R/W	DAC I/O[2]
DAC Interrupt Enable	0x31	W	Interrupt enable for each DAC pin
DAC Interrupt Polarity	0x32	W	Interrupt polarity for each DAC pin
DAC Isink	0x38-0x3F	W	One four bit sink current register for each DAC pin
Processor Status & Control	0xFF	R/W	Microprocessor status and control

Note
2. DAC I/O Port not bonded out on CY7C63613C. See note on page 12 for firmware code needed for unused GPIO pins.

Figure 2. Clock Oscillator On-chip Circuit



Clocking

The $XTAL_{IN}$ and $XTAL_{OUT}$ are the clock pins to the microcontroller. The user can connect a low-cost ceramic resonator or an external oscillator can be connected to these pins to provide a reference frequency for the internal clock distribution and clock doubler.

An external 6-MHz clock can be applied to the $XTAL_{IN}$ pin if the $XTAL_{OUT}$ pin is left open. Please note that grounding the $XTAL_{OUT}$ pin is not permissible as the internal clock is effectively shorted to ground.

Reset

The USB Controller supports three types of resets. All registers are restored to their default states during a reset. The USB Device Addresses are set to 0 and all interrupts are disabled. In addition, the program stack pointer (PSP) and data stack pointer (DSP) are set to 0x00. For USB applications, the firmware should set the DSP below 0xE8 to avoid a memory conflict with RAM dedicated to USB FIFOs. The assembly instructions to do this are shown below:

```
Mov A, E8h     ; Move 0xE8 hex into Accumulator
Swap A,dsp     ; Swap accumulator value into dsp register
```

The three reset types are:

1. Power-On Reset (POR)
2. Watch Dog Reset (WDR)
3. USB Bus Reset (non hardware reset)

The occurrence of a reset is recorded in the Processor Status and Control Register located at I/O address 0xFF. Bits 4, 5, and 6 are used to record the occurrence of POR, USB Reset, and WDR respectively. The firmware can interrogate these bits to determine the cause of a reset.

The microcontroller begins execution from ROM address 0x0000 after a POR or WDR reset. Although this looks like interrupt vector 0, there is an important difference. Reset processing does NOT push the program counter, carry flag, and zero flag onto program stack. That means the reset handler in firmware should initialize the hardware and begin executing the “main” loop of code. Attempting to execute either a RET or RETI in the reset handler will cause unpredictable execution results.

Power-On Reset (POR)

Power-On Reset (POR) occurs every time the V_{CC} voltage to the device ramps from 0V to an internally defined trip voltage (Vrst) of approximately 1/2 full supply voltage. In addition to the normal reset initialization noted under “Reset,” bit 4 (PORS) of the Processor Status and Control Register is set to “1” to indicate to the firmware that a Power-On Reset occurred. The POR event forces the GPIO ports into input mode (high impedance), and the state of Port 3 bit 7 is used to control how the part will respond after the POR releases.

If Port 3 bit 7 is HIGH (pulled to V_{CC}) and the USB I/O are at the idle state (DM HIGH and DP LOW) the part will go into a semi-permanent power down/suspend mode, waiting for the USB I/O to go to one of Bus Reset, K (resume) or SE0. If Port 3 bit 7 is still HIGH when the part comes out of suspend, then a 128-µs timer starts, delaying CPU operation until the ceramic resonator has stabilized.

If Port 3 bit 7 was LOW (pulled to V_{SS}) the part will start a 96-ms timer, delaying CPU operation until V_{CC} has stabilized, then continuing to run as reset.

Firmware should clear the POR Status (PORS) bit in register 0xFF before going into suspend as this status bit selects the 128-µs or 96-ms start-up timer value as follows: IF Port 3 bit 7 is HIGH then 128-µs is always used; ELSE if PORS is HIGH then 96-ms is used; ELSE 128-µs is used.

Watch Dog Reset (WDR)

The Watch Dog Timer Reset (WDR) occurs when the Most Significant Bit (MSB) of the 2-bit Watch Dog Timer Register transitions from LOW to HIGH. In addition to the normal reset initialization noted under “Reset,” bit 6 of the Processor Status and Control Register is set to “1” to indicate to the firmware that a Watch Dog Reset occurred.

The Watch Dog Timer is a 2-bit timer clocked by a 4.096-ms clock (bit 11) from the free-running timer. Writing any value to the write-only Watch Dog Clear I/O port (0x26) will clear the Watch Dog Timer.

In some applications, the Watch Dog Timer may be cleared in the 1.024-ms timer interrupt service routine. If the 1.024-ms timer interrupt service routine does not get executed for 8.192 ms or more, a Watch Dog Timer Reset will occur. A Watch Dog Timer Reset lasts for 2.048 ms after which the microcontroller begins execution at ROM address 0x0000. The USB transmitter is disabled by a Watch Dog Reset because the USB Device Address Register is cleared. Otherwise, the USB Controller would respond to all address 0 transactions. The USB transmitter remains disabled until the MSB of the USB address register is set.

DAC Port

Figure 5. Block Diagram of DAC Port

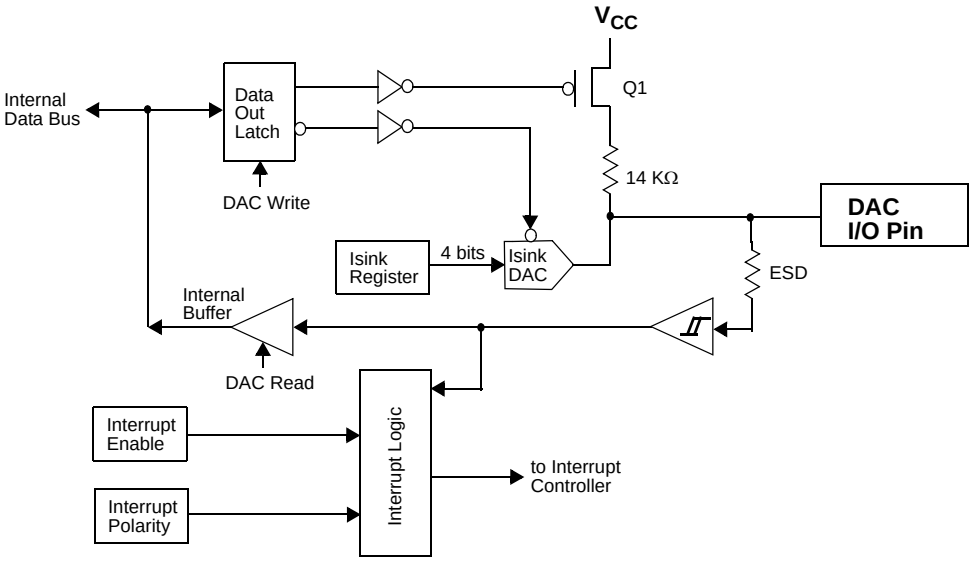


Table 13. DAC Port Data

Addr: 0x30		DAC Port Data					
Low current outputs 0.2 mA to 1.0 mA typical						High current outputs 3.2 mA to 16 mA typical	
DAC[7]	DAC[6]	DAC[5]	DAC[4]	DAC[3]	DAC[2]	DAC[1]	DAC[0]
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The DAC port provides the CY7C63513C with 8 programmable current sink I/O pins. Writing a “1” to a DAC I/O pin disables the output current sink (Isink DAC) and drives the I/O pin HIGH through an integrated 14 Kohm resistor. When a “0” is written to a DAC I/O pin, the Isink DAC is enabled and the pull-up resistor is disabled. A “0” output will cause the Isink DAC to sink current to drive the output LOW. 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 for that output pin. DAC[1:0] are the two high current outputs that are programmable from a minimum of 3.2 mA to a maximum of 16 mA (typical). DAC[7:2] are low current outputs that are programmable from a minimum of 0.2 mA to a maximum of 1.0 mA (typical).

When a DAC I/O bit is written as a “1,” the I/O pin is either an output pulled high through the 14 Kohm resistor or an input with an internal 14 Kohm pull-up resistor. All DAC port data bits are set to “1” during reset.

DAC Port Interrupts

A DAC port interrupt can be enabled/disabled for each pin individually. The DAC Port Interrupt Enable register provides this feature with an interrupt mask bit for each DAC I/O pin. Writing a “1” to a bit in this register enables interrupts from the corresponding bit position. Writing a “0” to a bit in the DAC Port Interrupt Enable register disables interrupts from the corresponding bit position. All of the DAC Port Interrupt Enable register bits are cleared to “0” during a reset.

As an additional benefit, the interrupt polarity for each DAC pin is programmable with the DAC Port Interrupt Polarity register. Writing a “0” to a bit selects negative polarity (falling edge) that will cause an interrupt (if enabled) if a falling edge transition occurs on the corresponding input pin. Writing a “1” to a bit in this register selects positive polarity (rising edge) that will cause an interrupt (if enabled) if a rising edge transition occurs on the corresponding input pin. All of the DAC Port Interrupt Polarity register bits are cleared during a reset.

DAC Isink Registers

Each DAC I/O pin has an associated DAC Isink register to program the output sink current when the output is driven LOW. The first Isink register (0x38) controls the current for DAC[0], the second (0x39) for DAC[1], and so on until the Isink register at 0x3F controls the current to DAC[7].

The Bus Activity bit is a “sticky” bit that indicates if any non-idle USB event has occurred on the USB bus. The user 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 can clear the Bus Activity bit, but only the SIE can set it. The 1.024-ms timer interrupt service routine is normally used to check and clear the Bus Activity bit. The following table shows how the control bits are encoded for this register.

Control Bits	Control Action
000	Not forcing (SIE controls driver)
001	Force K (D+ HIGH, D– LOW)
010	Force J (D+ LOW, D– HIGH)
011	Force SE0 (D+ LOW, D– LOW)
100	Force SE0 (D– LOW, D+ LOW)
101	Force D– LOW, D+ HiZ
110	Force D– HiZ, D+ LOW
111	Force D– HiZ, D+ HiZ

USB Device

USB Device Address A includes three endpoints: EPA0, EPA1, and EPA2. End Point 0 (EPA0) allows the USB host to recognize, set up, and control the device. In particular, EPA0 is used to receive and transmit control (including set-up) packets.

USB Ports

The USB Controller provides one USB device address with three endpoints. The USB Device Address Register contents are cleared during a reset, setting the USB device address to zero and marking this address as disabled. Figure 18 shows the format of the USB Address Register.

Bit 7 (Device Address Enable) in the USB Device Address Register must be set by firmware before the serial interface engine (SIE) will respond to USB traffic to this address. The Device Address in bits [6:0] must be set by firmware during the USB enumeration process to an address assigned by the USB host that does not equal zero. This register is cleared by a hardware reset or the USB bus reset.

Device Endpoints (3)

The USB controller communicates with the host using dedicated FIFOs, one per endpoint. Each endpoint FIFO is implemented as 8 bytes of dedicated SRAM. There are three endpoints defined for Device “A” that are labeled “EPA0,” “EPA1,” and EPA2.”

All USB devices are required to have an endpoint number 0 (EPA0) that is used to initialize and control the USB device. End Point 0 provides access to the device configuration information and allows generic USB status and control accesses. End Point 0 is bidirectional as the USB controller can both receive and transmit data.

The endpoint mode registers are cleared during reset. The EPA0 endpoint mode register uses the format shown in Table 19.

Bits[7:5] in the endpoint 0 mode registers (EPA0) are “sticky” status bits that are set by the SIE to report the type of token that was most recently received. The sticky bits must be cleared by firmware as part of the USB processing.

The endpoint mode registers for EPA1 and EPA2 do not use bits [7:5] as shown in Table 20.

Table 18. USB Device Address Register

Addr:0x10		USB Device Address Register					
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
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 19. USB Device EPA0, Mode Register

Addr:0x12		USB Device EPA0, Mode Register					
Endpoint 0 Set-up Received	Endpoint 0 In Received	Endpoint 0 Out Received	Acknowledge	Mode Bit 3	Mode Bit 2	Mode Bit 1	Mode Bit 0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20. USB Device Endpoint Mode Register

Addr: 0x14, 0x16		USB Device Endpoint Mode Register					
Reserved	Reserved	Reserved	Acknowledge	Mode Bit 3	Mode Bit 2	Mode Bit 1	Mode Bit 0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

12-bit Free-running Timer

The 12-bit timer 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 8 bits of the timer can be read directly by the firmware. Reading the lower 8 bits latches the upper 4 bits into a temporary register. When the firmware reads the upper 4 bits of the timer, it is actually reading the count stored in the temporary register. The effect of this logic is to ensure a stable 12-bit timer value can be read, even when the two reads are separated in time.

Timer (LSB)

Table 22. Timer Register

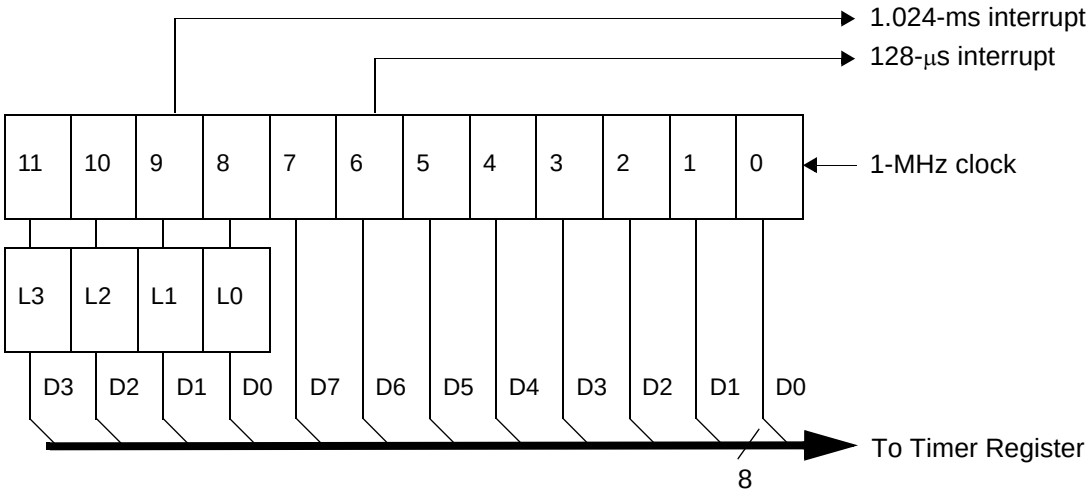
Addr: 0x24		Timer Register (LSB)					
Timer Bit 7	Timer Bit 6	Timer Bit 5	Timer Bit 4	Timer Bit 3	Timer Bit 2	Timer Bit 1	Timer Bit 0
R	R	R	R	R	R	R	R

Timer (MSB)

Table 23. Timer Register

Addr: 0x25		Timer Register (MSB)					
Reserved	Reserved	Reserved	Reserved	Timer Bit 11	Timer Bit 10	Timer Bit 9	Timer Bit 8
				R	R	R	R

Figure 6. Timer Block Diagram



Processor Status and Control Register

Table 24. Processor Status and Control Register

Addr: 0xFF		Processor Status and Control Register				POR Default: 0x0101 WDC Reset: 0x41	
7	6	5	4	3	2	1	0
IRQ Pending	Watch Dog Reset	USB Bus Reset	Power-on Reset	Suspend, Wait for Interrupt	Interrupt Mask	Single Step	Run
R	R/W	R/W	R/W	R/W	R	R/W	R/W

The “Run” (bit 0) is manipulated by the HALT instruction. When Halt is executed, the processor clears the run bit and halts at the end of the current instruction. The processor remains halted until a reset (Power On or Watch Dog). Notice, when writing to the processor status and control register, the run bit should always be written as a “1.”

The “Single Step” (bit 1) is provided to support a hardware debugger. When single step is set, the processor will execute one instruction and halt (clear the run bit). This bit must be cleared for normal operation.

The “Interrupt Mask” (bit 2) shows whether interrupts are enabled or disabled. The firmware has no direct control over this bit as writing a zero or one to this bit position will have no effect on interrupts. Instructions DI, EI, and RETI manipulate the internal hardware that controls the state of the interrupt mask bit in the Processor Status and Control Register.

Writing a “1” to “Suspend, Wait for Interrupts” (bit 3) will halt the processor and cause the microcontroller to enter the “suspend” mode that significantly reduces power consumption. A pending interrupt or bus activity will cause the device to come out of suspend. After coming out of suspend, the device will resume firmware execution at the instruction following the IOWR which put the part into suspend. An IOWR that attempts to put the part into suspend will be ignored if either bus activity or an interrupt is pending.

The “Power-on Reset” (bit 4) is only set to “1” during a power on reset. The firmware can check bits 4 and 6 in the reset handler to determine whether a reset was caused by a Power On condition or a Watch Dog Timeout. PORS is used to determine suspend start-up timer value of 128 μs or 96 ms.

The “USB Bus Reset” (bit 5) will occur when a USB bus reset is received. The USB Bus Reset is a singled-ended zero (SE0) that lasts more than 8 microseconds. An SE0 is defined as the condition in which both the D+ line and the D– line are LOW at the same time. When the SIE detects this condition, the USB Bus Reset bit is set in the Processor Status and Control register and an USB Bus Reset interrupt is generated. Please note this is an interrupt to the microcontroller and does not actually reset the processor.

The “Watch Dog Reset” (bit 6) is set during a reset initiated by the Watch Dog Timer. This indicates the Watch Dog Timer went for more than 8 ms between watch dog clears.

The “IRQ Pending” (bit 7) indicates one or more of the interrupts has been recognized as active. The interrupt acknowledge sequence should clear this bit until the next interrupt is detected.

During Power-on Reset, the Processor Status and Control Register is set to 00010001, which indicates a Power-on Reset (bit 4 set) has occurred and no interrupts are pending (bit 7 clear) yet.

During a Watch Dog Reset, the Processor Status and Control Register is set to 01000001, which indicates a Watch Dog Reset (bit 6 set) has occurred and no interrupts are pending (bit 7 clear) yet.

Interrupts

All interrupts are maskable by the Global Interrupt Enable Register and the USB End Point Interrupt Enable Register. Writing a “1” to a bit position enables the interrupt associated with that bit position. During a reset, the contents the Global Interrupt Enable Register and USB End Point Interrupt Enable Register are cleared, effectively disabling all interrupts.

Pending interrupt requests are recognized during the last clock cycle of the current instruction. When servicing an interrupt, the hardware will first disable all interrupts by clearing the Interrupt Enable bit in the Processor Status and Control Register. Next, the interrupt latch of the current interrupt is cleared. This is followed by a CALL instruction to the ROM address associated with the interrupt being serviced (i.e., 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 can re-enable interrupts in the interrupt service routine by executing an EI instruction. Interrupts can be nested to a level limited only by the available stack space.

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

interrupt priority to different DAC pins and the DAC Interrupt Enable Register is not cleared during the interrupt acknowledge process.

GPIO Interrupt

Each of the 32 GPIO pins can generate an interrupt, if enabled. The interrupt polarity can be programmed for each GPIO port as part of the GPIO configuration. All of the GPIO pins share a single interrupt vector, which means the firmware will need to read the GPIO ports with enabled interrupts to determine which pin or pins caused an interrupt.

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

Truth Tables

Table 27. USB Register Mode Encoding

Mode	Encoding	Setup	In	Out	Comments
Disable	**0000**	ignore	ignore	ignore	Ignore all USB traffic to this endpoint
Nak In/Out	**0001**	accept	NAK	NAK	Forced from Set-up 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	Available to low speed devices, future USB spec enhancements
Status In Only	**0110**	accept	TX 0	stall	For Control Endpoints
Isochronous In	**0111**	ignore	TX cnt	ignore	Available to low speed devices, future USB spec enhancements
Nak Out	**1000**	ignore	ignore	NAK	An ACK from mode 1001 --> 1000
Ack Out	**1001**	ignore	ignore	ACK	This mode is changed by SIE on issuance of ACK --> 1000
Nak Out - Status In	**1010**	accept	TX 0	NAK	An ACK from mode 1011 --> 1010
Ack Out - Status In	**1011**	accept	TX 0	ACK	This mode is changed by SIE on issuance of ACK --> 1010
Nak In	**1100**	ignore	NAK	ignore	An ACK from mode 1101 --> 1100
Ack In	**1101**	ignore	TX cnt	ignore	This mode is changed by SIE on issuance of ACK --> 1100
Nak In - Status Out	**1110**	accept	NAK	check	An ACK from mode 1111 --> 1110 NAck In - Status Out
Ack In - Status Out	**1111**	accept	TX cnt	Check	This mode is changed by SIE on issuance of ACK -->1110

The ‘In’ column represents the SIE’s response to the token type.

A disabled endpoint will remain such until firmware changes it, and all endpoints reset to disabled.

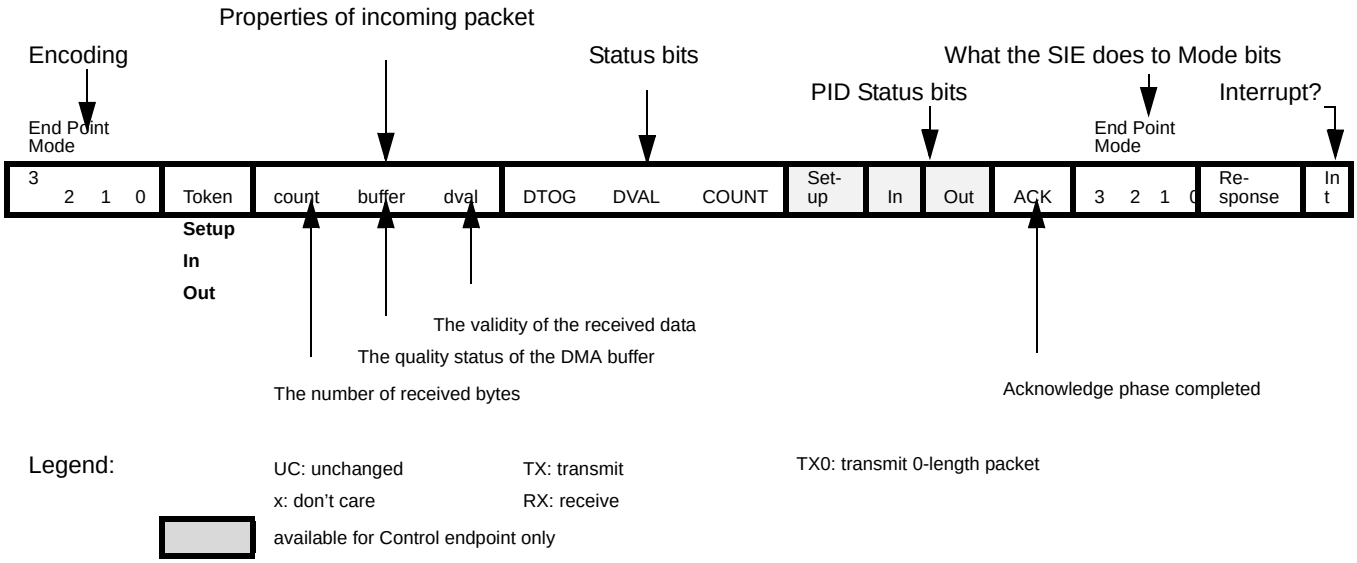
Any Setup packet to an enabled and accepting endpoint will be changed by the SIE to 0001 (NAKing). Any mode which indicates the acceptance of a Setup will acknowledge it.

Most modes that control transactions involving an ending ACK will be changed by the SIE to a corresponding mode which NAKs follow on packets.

A Control endpoint has three extra status bits for PID (Setup, In and Out), but must be placed in the correct mode to function as such. Also a non-Control endpoint can be made to act as a Control endpoint if it is placed in a non appropriate mode.

A ‘check’ on an Out token during a Status transaction checks to see that the Out is of zero length and has a Data Toggle (DTOG) of 1.

Figure 7. Decode table for Table 28: "Details of Modes for Differing Traffic Conditions"



The response of the SIE can be summarized as follows:

1. the SIE will only respond to valid transactions, and will ignore non-valid ones;
2. the SIE will generate IRQ when a valid transaction is completed or when the DMA buffer is corrupted
3. an incoming Data packet is valid if the count is <= 10 (CRC inclusive) and passes all error checking;
4. a Setup will be ignored by all non-Control endpoints (in appropriate modes);
5. an In will be ignored by an Out configured endpoint and vice versa.

The In and Out PID status is updated at the end of a transaction.

The Setup PID status is updated at the beginning of the Data packet phase.

The entire EndPoint 0 mode and the Count register are locked to CPU writes at the end of any transaction in which an ACK is transferred. These registers are only unlocked upon a CPU read of these registers, and only if that read happens after the transaction completes. This represents about a 1-µs window to which to the CPU is locked from register writes to these USB registers. Normally the firmware does a register read at the beginning of the ISR to unlock and get the mode register information. The interlock on the Mode and Count registers ensures that the firmware recognizes the changes that the SIE might have made during the previous transaction.

Table 28. Details of Modes for Differing Traffic Conditions

End Point Mode											PID				Set End Point Mode					
3	2	1	0	token	count	buffer	dval	DTOG	DVAL	COUNT	Setup	In	Out	ACK	3	2	1	0	response	int
Setup Packet (if accepting)																				
See Table 27				Setup	<= 10	data	valid	updates	1	updates	1	UC	UC	1	0	0	0	1	ACK	yes
See Table 27				Setup	> 10	junk	x	updates	updates	updates	1	UC	UC	UC	NoChange				ignore	yes
See Table 27				Setup	x	junk	invalid	updates	0	updates	1	UC	UC	UC	NoChange				ignore	yes
Disabled																				
0	0	0	0	x	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
Nak In/Out																				
0	0	0	1	Out	x	UC	x	UC	UC	UC	UC	UC	1	UC	NoChange				NAK	yes
0	0	0	1	In	x	UC	x	UC	UC	UC	UC	1	UC	UC	NoChange				NAK	yes
Ignore In/Out																				
0	1	0	0	Out	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
0	1	0	0	In	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
Stall In/Out																				
0	0	1	1	Out	x	UC	x	UC	UC	UC	UC	UC	1	UC	NoChange				Stall	yes
0	0	1	1	In	x	UC	x	UC	UC	UC	UC	1	UC	UC	NoChange				Stall	yes
Control Write																				
Normal Out/premature status In																				
1	0	1	1	Out	<= 10	data	valid	updates	1	updates	UC	UC	1	1	1	0	1	0	ACK	yes
1	0	1	1	Out	> 10	junk	x	updates	updates	updates	UC	UC	1	UC	NoChange				ignore	yes
1	0	1	1	Out	x	junk	invalid	updates	0	updates	UC	UC	1	UC	NoChange				ignore	yes
1	0	1	1	In	x	UC	x	UC	UC	UC	UC	1	UC	1	NoChange				TX 0	yes
NAK Out/premature status In																				
1	0	1	0	Out	<= 10	UC	valid	UC	UC	UC	UC	UC	1	UC	NoChange				NAK	yes
1	0	1	0	Out	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
1	0	1	0	Out	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
1	0	1	0	In	x	UC	x	UC	UC	UC	UC	1	UC	1	NoChange				TX 0	yes
Status In/extra Out																				
0	1	1	0	Out	<= 10	UC	valid	UC	UC	UC	UC	UC	1	UC	0	0	1	1	Stall	yes
0	1	1	0	Out	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
0	1	1	0	Out	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
0	1	1	0	In	x	UC	x	UC	UC	UC	UC	1	UC	1	NoChange				TX 0	yes
Control Read																				
Normal In/premature status Out																				
1	1	1	1	Out	2	UC	valid	1	1	updates	UC	UC	1	1	NoChange				ACK	yes
1	1	1	1	Out	2	UC	valid	0	1	updates	UC	UC	1	UC	0	0	1	1	Stall	yes
1	1	1	1	Out	!=2	UC	valid	updates	1	updates	UC	UC	1	UC	0	0	1	1	Stall	yes
1	1	1	1	Out	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
1	1	1	1	Out	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
1	1	1	1	In	x	UC	x	UC	UC	UC	UC	1	UC	1	1	1	1	0	ACK (back)	yes
Nak In/premature status Out																				
1	1	1	0	Out	2	UC	valid	1	1	updates	UC	UC	1	1	NoChange				ACK	yes
1	1	1	0	Out	2	UC	valid	0	1	updates	UC	UC	1	UC	0	0	1	1	Stall	yes
1	1	1	0	Out	!=2	UC	valid	updates	1	updates	UC	UC	1	UC	0	0	1	1	Stall	yes
1	1	1	0	Out	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
1	1	1	0	Out	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
1	1	1	0	In	x	UC	x	UC	UC	UC	UC	1	UC	UC	NoChange				NAK	yes
Status Out/extra In																				
0	0	1	0	Out	2	UC	valid	1	1	updates	UC	UC	1	1	NoChange				ACK	yes

Absolute Maximum Ratings

Storage temperature .....–65 °C to +150 °C
Ambient temperature with power applied ..... –0 °C to +70 °C
Supply voltage on V_{CC} relative to V_{SS}.....–0.5 V to +7.0 V
DC input voltage ..... –0.5 V to +V_{CC}+0.5 V
DC voltage applied to outputs in High Z state –0.5 V to + V_{CC}+0.5 V
Maximum output current into Port 0,1,2,3 and DAC[1:0] Pins60 mA
Maximum output current into DAC[7:2] Pins ..... 10 mA
Power dissipation ..... 300 mW
Static discharge voltage .....> 2000V[3]
Latch-up current ..... > 200 mA

DC Characteristics Fosc = 6 MHz; operating temperature = 0 to 70 °C

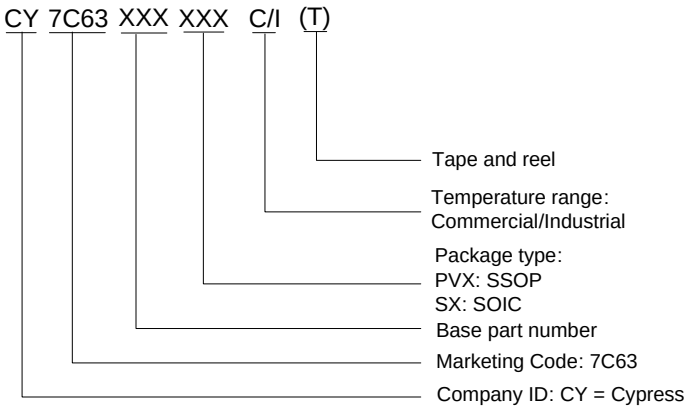
	Parameter	Min	Max	Unit	Conditions
General					
$V_{CC\ (1)}$	Operating voltage	4.0	5.5	V	Non USB activity[4]
$V_{CC\ (2)}$	Operating voltage	4.35	5.25	V	USB activity[5]
I_{CC1}	V_{CC} operating supply current	–	40	mA	V_{CC} = 5.5 V
I_{CC2}	V_{CC} = 4.35 V	–	15	mA	–
I_{SB1}	Supply current - suspend mode	–	30	µA	Oscillator off, D– > Voh min
V_{PP}	Programming voltage (disabled)	–0.4	0.4	V	–
T_{start}	Resonator start-up interval	–	256	µs	V_{CC} = 5.0 V, ceramic resonator
t_{int1}	Internal timer #1 interrupt period	128	128	µs	–
t_{int2}	Internal timer #2 interrupt period	1.024	1.024	ms	–
t_{watch}	Watch dog timer period	8.192	14.33	ms	–
I_{il}	Input leakage current	–	1	µA	Any pin
I_{sm}	Max I_{SS} I/O sink current	–	60	mA	Cumulative across all ports[6]
Power-On Reset					
t_{vccs}	V_{CC} reset slew	0.001	200	ms	Linear ramp: 0 to 4.35 V[7, 8]
USB Interface					
V_{oh}	Static output HIGH	2.8	3.6	V	15 k ± 5% Ω to Gnd [5]
V_{ol}	Static output LOW	–	0.3	V	–
V_{di}	Differential input sensitivity	0.2		V	|(D+)–(D–)|
V_{cm}	Differential input common mode range	0.8	2.5	V	9-1
V_{se}	Single-ended receiver threshold	0.8	2.0	V	–
C_{in}	Transceiver capacitance	–	20	pF	–
I_{lo}	Hi-Z state data line leakage	–10	10	µA	0 V < V_{in}<3.3 V
R_{pu}	Bus pull-up resistance (V_{CC} option)	7.35K	7.65	kΩ	7.5 kΩ ± 2% to V_{CC}
R_{pu}	Bus pull-up resistance (Ext. 3.3 V option)	1.425	1.575	kΩ	1.5 kΩ ± 5% to 3.0–3.6 V
R_{pd}	Bus pull-down resistance	14.25	15.75	kΩ	15 kΩ ± 5%
General Purpose I/O Interface					
R_{up}	Pull-up resistance	4.9 K	9.1 K	Ohms	–
V_{ith}	Input threshold voltage	45%	65%	V_{CC}	All ports, LOW to HIGH edge

Notes
3. Qualified with JEDEC EIA/JESD22-A114-B test method.
4. Functionality is guaranteed of the $V_{CC\ (1)}$ range, except USB transmitter and DACs.
5. USB transmitter functionality is guaranteed over the $V_{CC\ (2)}$ range, as well as DAC outputs.
6. Total current cumulative across all Port pins flowing to V_{SS} is limited to minimize Ground-Drop noise effects.
7. Port 3 bit 7 controls whether the parts goes into suspend after a POR event or waits 128 ms to begin running.
8. POR will re-occur whenever V_{CC} drops to approximately 2.5 V.

Ordering Information

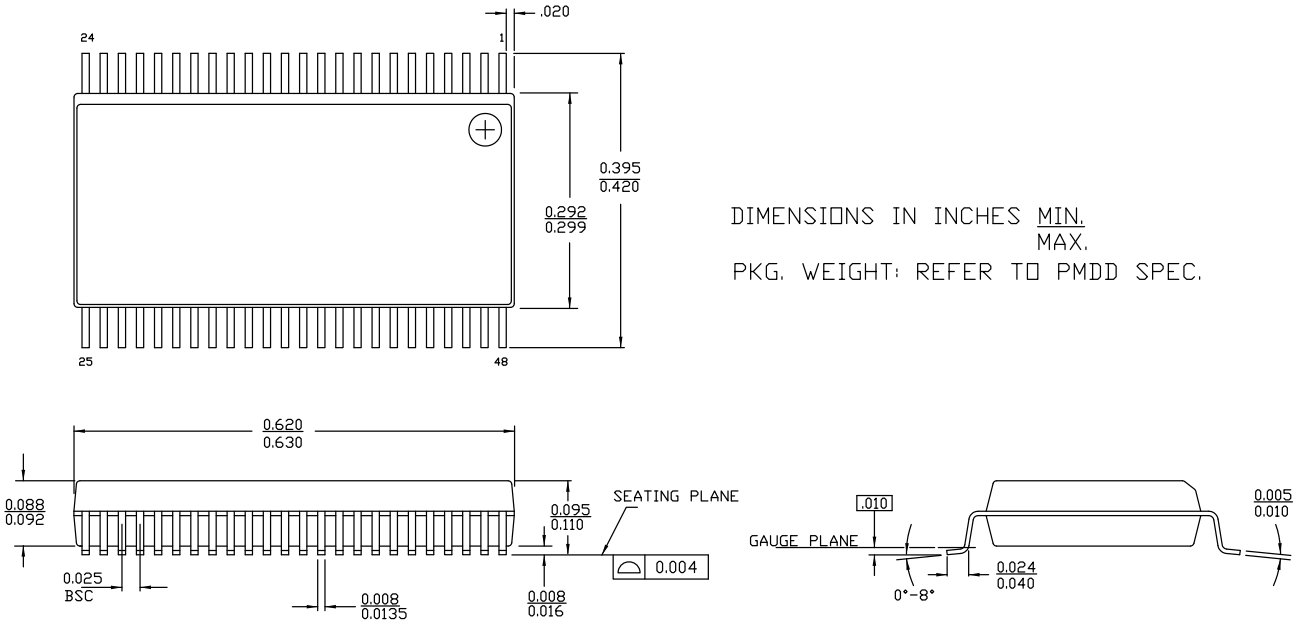
Ordering Code	EPROM Size	Package Name	Package Type	Operating Range
CY7C63413C-PVXC	8 KB	SP48	48-pin Shrunk Small Outline Package	Commercial
CY7C63413C-PVXCT	8 KB	SP48	48-pin SSOP Pb-free Tape-reel	Commercial
CY7C63513C-PVXC	8 KB	SP48	48-pin SSOP Pb-free	Commercial
CY7C63613C-SXC	8 KB	SZ24.3	24-pin (300 mil) SOIC Pb-free	Commercial
CY7C63613C-SXCT	8 KB	SZ24.3	24-pin (300 mil) SOIC Pb-free Tape-reel	Commercial
CY7C63513C-PVXCT	8 KB	SP48	48-pin SSOP Pb-free Tape-reel	Commercial

Ordering Code Definition



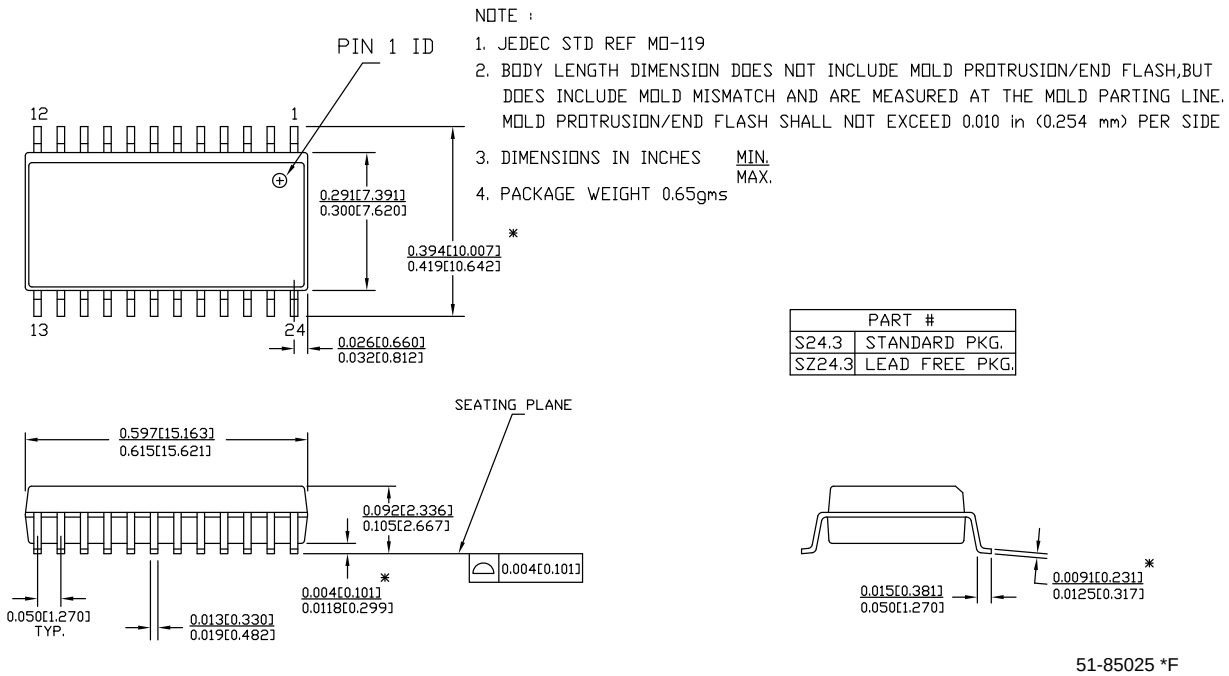
Package Diagrams

Figure 13. 48-pin SSOP (300 Mils) Package Outline, 51-85061



51-85061 *F

Figure 14. 24-pin SOIC (0.615 × 0.300 × 0.0932 Inches) Package Outline, 51-85025



Document History Page

Document Title: CY7C63413C/CY7C63513C/CY7C63613C, Low-Speed High Input/Output 1.5-Mbps USB Controller Document Number: 38-08027				
REV.	**ECN NO.**	**Issue Date**	**Orig. of Change**	**Description of Change**
**	116224	06/12/02	DSG	Change from Spec number: 38-00754 to 38-08027
*A	237148	SEE ECN	KKU	Removed 24 pin package, CY7C63411/12, CY7C63511/12 and CY7C636XX parts Added Pb-free part numbers to section 20.0 Added USB Logo.
*B	418699	See ECN	TYJ	Part numbers updated with MagnaChip offerings
*C	2896245	03/19/10	XUT	Removed inactive parts (CY7C63413C-PXC and CY7C63413C-XC) from the Ordering information. Updated package diagrams.
*D	3057657	10/13/10	AJHA	Added “Not recommended for new designs” watermark in the PDF. Updated template.
*E	3130046	01/18/2011	NXZ	Removed 40-pin PDIP package. Added Ordering Code Definitions, Acronyms, and Document Conventions.
*F	4272762	02/05/2014	DEJO	Updated Package Diagrams: spec 51-85061 – Changed revision from *D to *F. spec 51-85025 – Changed revision from *D to *F. Updated in new template. Completing Sunset Review.
*G	4313900	03/21/2014	AKSL	Removed “Not recommended for new designs” watermark. Added CY7C63613C-SXCT and CY7C63513C-PVXCT in Ordering Information.
*H	5647775	03/02/2017	HPPC	Updated the template.

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