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	CY7C636xx
RAM Size	256 x 8
Interface	PS/2, USB
Number of I/O	16
Voltage - Supply	4V ~ 5.5V
Operating Temperature	0°C ~ 70°C
Mounting Type	Surface Mount
Package / Case	24-SOIC (0.295", 7.50mm Width)
Supplier Device Package	24-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/infineon-technologies/cy7c63613-sc

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

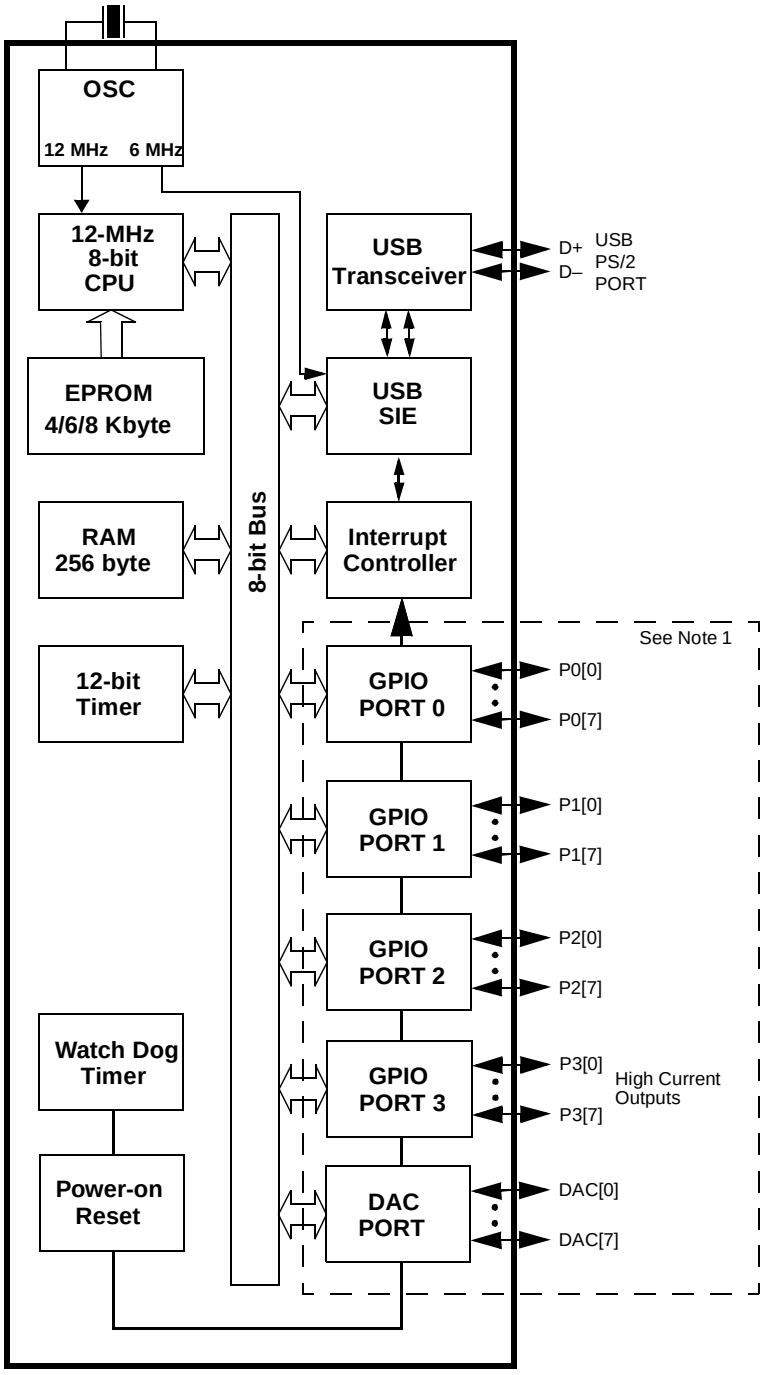
LIST OF FIGURES

LIST OF TABLES

1.0 Features

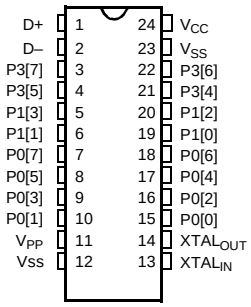
- Low-cost solution for low-speed applications with low I/O requirements such as mice, gamepads, and joystick applications
- USB Specification Compliance
  - Conforms to USB Specification, Version 1.1
  - Conforms to USB HID Specification, Version 1.1
  - Supports 1 device address and 3 data endpoints
  - Integrated USB transceiver
- 8-bit RISC microcontroller
  - Harvard architecture
  - 6-MHz external ceramic resonator
  - 12-MHz internal CPU clock
- Internal memory
  - 256 bytes of RAM
  - 6 Kbytes of EPROM (CY7C63612)
  - 8 Kbytes of EPROM (CY7C63613)
- Interface can auto-configure to operate as PS2 or USB
- I/O port
  - 12 General-Purpose I/O (GPIO) pins (Port 0 to 2) capable of sinking 7 mA per pin (typical)
  - Four GPIO pins (Port 3) capable of sinking 12 mA per pin (typical) which can drive LEDs
  - Higher current drive is available by connecting multiple GPIO pins together to drive an common output
  - Each GPIO port can be configured as inputs with internal pull-ups or open drain outputs or traditional CMOS outputs
  - Maskable interrupts on all I/O pins
- 12-bit free-running timer with one microsecond clock ticks
- Watch Dog Timer (WDT)
- Internal Power-On Reset (POR)
- Improved output drivers to reduce EMI
- Operating voltage from 4.0V to 5.5V DC
- Operating temperature from 0 to 70 degrees Celsius
- CY7C63612/13 available in 24-pin SOIC packages for production
- Industry-standard programmer support

Logic Block Diagram



Pin Configuration

CY7C63612/13
24-pin SOIC



TOP VIEW

Note:
1. CY7C63612/13 is not bonded out for all GPIO pins shown in Logic Block Diagram. Refer to pin configuration diagram for bonded out pins. See note on page 16 for firmware code needed for unused GPIO pins.

3.0 Pin Assignments

Name	I/O	CY7C63613 24-Pin	Description
D+, D–	I/O	1,2	USB differential data; PS/2 clock and data signals
P0[7:0]	I/O	7,18,8,17, 9,16,10,15	GPIO port 0 capable of sinking 7 mA (typical)
P1[3:0]	I/O	5,20,6,19	GPIO Port 1 capable of sinking 7 mA (typical). P1[7:4] not bonded out on CY7C63612/13. See note on page 16 for firmware code needed for unused pins.
P2	I/O	n/a	GPIO Port 2 not bonded out on CY7C63612/13. See note on page 16 for firmware code needed for unused pins.
P3[7:4]	I/O	3,22,4,21	GPIO Port 3 capable of sinking 12 mA (typical). P3[3:0] not bonded out on CY7C63612/13. See note on page 16 for firmware code needed for unused pins.
DAC	I/O	n/a	DAC I/O Port not bonded out on CY7C63612/13. See note on page 16 for firmware code needed for unused pins.
$XTAL_{IN}$	IN	13	6-MHz ceramic resonator or external clock input
$XTAL_{OUT}$	OUT	14	6-MHz ceramic resonator
V_{PP}		11	Programming voltage supply, ground for normal operation
V_{CC}		24	Voltage supply
Vss		12,23	Ground

4.0 Programming Model

4.1 14-bit Program Counter (PC)

The 14-bit Program Counter (PC) allows access for up to 8 kilobytes of EPROM using the CY7C636xx architecture. The program counter is cleared during reset, such that the first instruction executed after a reset is at address 0x0000h. This is typically a jump instruction to a reset handler that initializes the application.

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

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

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

4.2 8-bit Accumulator (A)

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

4.3 8-bit Index Register (X)

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

4.4 8-bit Program Stack Pointer (PSP)

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

During an interrupt acknowledge, interrupts are disabled and the 14-bit program counter, carry flag, and zero flag are written as two bytes of data memory. The first byte is stored in the memory addressed by the program stack pointer, then the PSP is

incremented. The second byte is stored in memory addressed by the program stack pointer and the PSP is incremented again. The net effect is to store the program counter and flags on the program “stack” and increment the program stack pointer by two.

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

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

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

4.5 8-bit Data Stack Pointer (DSP)

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

During a reset, the Data Stack Pointer will be set to zero. A PUSH instruction when DSP equal zero will write data at the top of the data RAM (address 0xFF). This would write data to the memory area reserved for a FIFO for USB endpoint 0. In non-USB applications, this works fine and is not a problem. For USB applications, it is strongly recommended that the DSP is loaded after reset just below the USB DMA buffers.

4.6 Address Modes

The CY7C63612/13 microcontrollers support three addressing modes for instructions that require data operands: data, direct, and indexed.

4.6.1 Data

The “Data” address mode refers to a data operand that is actually a constant encoded in the instruction. As an example, consider the instruction that loads A with the constant 0xE8h:

- MOV A,0E8h

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

- DSPINIT: EQU 0E8h
- MOV A,DSPINIT

4.6.2 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 0x10h:

- MOV A, [10h]

In normal usage, 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 shown above:

- buttons: EQU 10h
- MOV A,[buttons]

4.6.3 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. In normal usage, the constant will be the “base” address of an array of data and the X register will contain an index that indicates which element of the array is actually addressed:

- array: EQU 10h
- MOV X,3
- MOV A,[x+array]

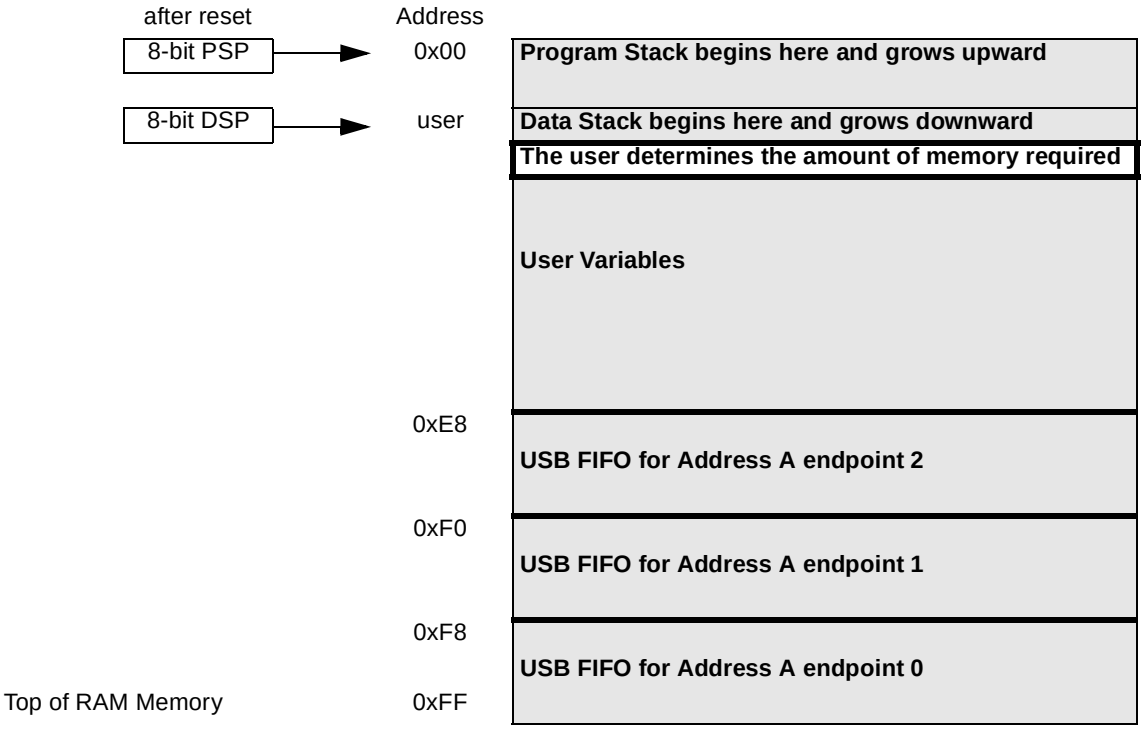
This would have the effect of loading A with the fourth element of the SRAM “array” that begins at address 0x10h. The fourth element would be at address 0x13h.

5.0 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

6.2 Data Memory Organization

The CY7C63612/13 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:



7.0 Clocking

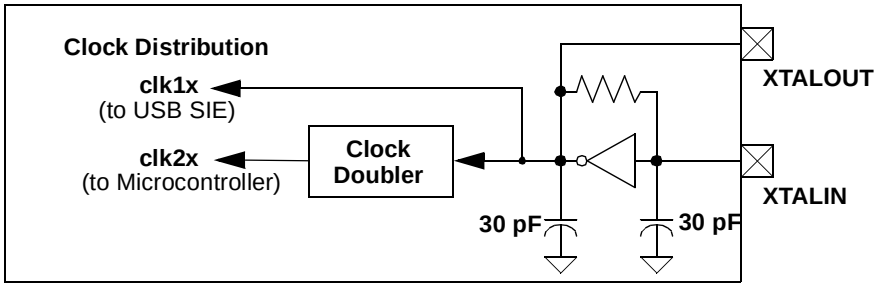


Figure 7-1. Clock Oscillator On-chip Circuit

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.

8.0 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 0xE8h 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 0x0000h 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.

8.1 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 IO 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 IO 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 128-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 FFh before going into suspend as this status bit selects the 128-µs or 128-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 128-ms is used; ELSE 128-µs is used.

provides “HIGH” source current when the GPIO port is configured for CMOS outputs and the output data bit is written as a “1”. Q2 and Q3 are sized to sink and source, respectively, roughly the same amount of current to support traditional CMOS outputs with symmetric drive.

P0[7]	P0[6]	P0[5]	P0[4]	P0[3]	P0[2]	P0[1]	P0[0]

Figure 9-2. Port 0 Data 0x00h (read/write)

P1[7]	P1[6]	P1[5]	P1[4]	P1[3]	P1[2]	P1[1]	P1[0]

Figure 9-3. Port 1 Data 0x01h (read/write)

P2[7]	P2[6]	P2[5]	P2[4]	P2[3]	P2[2]	P2[1]	P2[0]

Figure 9-4. Port 2 Data 0x02h (read/write)

P3[7]	P3[6]	P3[5]	P3[4]	P3[3]	P3[2]	P3[1]	P3[0]

Figure 9-5. Port 3 Data 0x03h (read/write)

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]

Figure 9-6. DAC Port Data 0x30h (read/write)

Port 3 has eight GPIO pins. Port 3 (8 bits) can be configured as inputs with internal pull-ups, open drain outputs, or traditional CMOS outputs. An open drain output is also a high-impedance input. Port 3 offers high current drive with a typical current sink capability of 12 mA. The internal pull-up resistors are typically 7 kΩ.

NOTE: Special care should be exercised 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 Specification. If a ‘1’ is written to the unused data bit and the port is configured with open drain outputs, the unused data bit will be 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 CY7C63612/13 will always require that data bits P1[7:4], P2[7:0], P3[3:0] and DAC[7:0] be written with a ‘0.’

During reset, all of the GPIO pins are set to output “1” (input) with the internal pull-up enabled. In this state, a “1” will always be read on that GPIO pin unless an external current sink drives the output to a “0” state. Writing a “0” to a GPIO pin enables the output current sink to ground (LOW) and disables the internal pull-up for that pin.

9.1 GPIO Interrupt Enable Ports

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.

P0[7]	P0[6]	P0[5]	P0[4]	P0[3]	P0[2]	P0[1]	P0[0]

Figure 9-7. Port 0 Interrupt Enable 0x04h (write only)

P1[7]	P1[6]	P1[5]	P1[4]	P1[3]	P1[2]	P1[1]	P1[0]

Figure 9-8. Port 1 Interrupt Enable 0x05h (write only)

P2[7]	P2[6]	P2[5]	P2[4]	P2[3]	P2[2]	P2[1]	P2[0]

Figure 9-9. Port 2 Interrupt Enable 0x06h (write only)

P3[7]	P3[6]	P3[5]	P3[4]	P3[3]	P3[2]	P3[1]	P3[0]

Figure 9-10. Port 3 Interrupt Enable 0x07h (write only)

9.2 GPIO Configuration Port

Every GPIO port can be programmed as inputs with internal pull-ups, open drain outputs, and traditional CMOS outputs. In addition, the interrupt polarity for each port can be programmed. With positive interrupt polarity, a rising edge (“0” to “1”) on an input pin causes an interrupt. With negative polarity, a falling edge (“1” to “0”) on an input pin causes an interrupt. As shown in the table below, when a GPIO port is configured with CMOS outputs, interrupts from that port are disabled. The GPIO Configuration Port register provides two bits per port to program these features. The possible port configurations are:

Port Configuration bits	Pin Interrupt Bit	Driver Mode	Interrupt Polarity
11	X	Resistive	-
10	0	CMOS Output	disabled
10	1	Open Drain	disabled
01	X	Open Drain	-
00	X	Open Drain	+ (default)

In “Resistive” mode, a 7-kΩ pull-up resistor is conditionally enabled for all pins of a GPIO port. The resistor is enabled for any pin that has been written as a “1.” The resistor is disabled on any pin that has been written as a “0”. An I/O pin will be driven high through a 7-kΩ pull-up resistor when a “1” has been written to the pin. Or the output pin will be driven LOW, with the pull-up disabled, when a “0” has been written to the pin. An I/O pin that has been written as a “1” can be used as an input pin with an integrated 7-kΩ pull-up resistor. Resistive mode selects a negative (falling edge) interrupt polarity on all pins that have the GPIO interrupt enabled.

In “CMOS” mode, all pins of the GPIO port are outputs that are actively driven. The current source and sink capacity are roughly the same (symmetric output drive). A CMOS port is not a possible source for interrupts.

A port configured in CMOS mode has interrupt generation disabled, yet the interrupt mask bits serve to control port direction. If a port’s associated Interrupt Mask bits are cleared, those port bits are strictly outputs. If the Interrupt Mask bits are set then those bits will be open drain inputs. As open drain inputs, if their data output values are ‘1’ those port pins will be CMOS inputs (HIGH Z output).

In “Open Drain” mode the internal pull-up resistor and CMOS driver (HIGH) are both disabled. An I/O pin that has been written as a “1” can be used as either a high-impedance input or a three-state output. An I/O pin that has been written as a “0” will drive the output LOW. The interrupt polarity for an open drain GPIO port can be selected as either positive (rising edge) or negative (falling edge).

During reset, all of the bits in the GPIO Configuration Register are written with “0”. This selects the default configuration: Open Drain output, positive interrupt polarity for all GPIO ports.

7	6	5	4	3	2	1	0
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

Figure 9-11. GPIO Configuration Register 0x08h (write only)

10.0 USB Serial Interface Engine (SIE)

The SIE allows the microcontroller to communicate with the USB host. The SIE simplifies the interface between the microcontroller and USB by incorporating hardware that handles the following USB bus activity independently of the microcontroller:

- Bit stuffing/unstuffing
- Checksum generation/checking
- ACK/NAK
- Token type identification
- Address checking

Firmware is required to handle the rest of the USB interface with the following tasks:

- Coordinate enumeration by responding to set-up packets
- Fill and empty the FIFOs
- Suspend/Resume coordination
- Verify and select Data toggle values

10.1 USB Enumeration

The enumeration sequence is shown below:

1. The host computer sends a **Setup** packet followed by a **Data** packet to USB address 0 requesting the Device descriptor.
2. The USB Controller decodes the request and retrieves its Device descriptor from the program memory space.
3. The host computer performs a control read sequence and the USB Controller responds by sending the Device descriptor over the USB bus.
4. After receiving the descriptor, the host computer sends a **Setup** packet followed by a **Data** packet to address 0 assigning a new USB address to the device.
5. The USB Controller stores the new address in its USB Device Address Register after the no-data control sequence is complete.
6. The host sends a request for the Device descriptor using the new USB address.
7. The USB Controller decodes the request and retrieves the Device descriptor from the program memory.
8. The host performs a control read sequence and the USB Controller responds by sending its Device descriptor over the USB bus.
9. The host generates control reads to the USB Controller to request the Configuration and Report descriptors.
10. The USB Controller retrieves the descriptors from its program space and returns the data to the host over the USB.

10.2 PS/2 Operation

PS/2 operation is possible with the CY7C636xx series through the use of firmware and several operating modes. The first enabling feature:

1. USB Bus reset on D+ and D– is an interrupt that can be disabled;
2. USB traffic can be disabled via bit 7 of the USB register;
3. D+ and D– can be monitored and driven via firmware as independent port bits.

Bits 5 and 4 of the Upstream Status and Control register are directly connected to the D+ and D– USB pins of the CY7C636xx. These pins constantly monitor the levels of these signals with CMOS input thresholds. Firmware can poll and decode these signals as PS/2 clock and data.

Bits [2:0] defaults to '000' at reset which allows the USB SIE to control output on D+ and D–. Firmware can override the SIE and directly control the state of these pins via these 3 control bits. Since PS/2 is an open drain signaling protocol, these modes allow all 4 PS/2 states to be generated on the D+ and D– pins

10.3 USB Port Status and Control

USB status and control is regulated by the USB Status and Control Register located at I/O address 0x1Fh as shown in *Figure 10-1*. This is a read/write register. All reserved bits must be written to zero. All bits in the register are cleared during reset.

7	6	5	4	3	2	1	0
		R	R	R/W	R/W	R/W	R/W
Reserved	Reserved	D+	D–	Bus Activity	Control Bit 2	Control Bit 1	Control Bit 0

Figure 10-1. USB Status and Control Register 0x1Fh

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.

14.2.2 Timer Interrupt

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

14.2.3 USB Endpoint Interrupts

There are three USB endpoint interrupts, one per endpoint. The USB endpoints interrupt after the either the USB host or the USB controller sends a packet to the USB.

14.2.4 DAC Interrupt

Each DAC I/O pin can generate an interrupt, if enabled.The interrupt polarity for each DAC I/O pin is programmable. A positive polarity is a rising edge input while a negative polarity is a falling edge input. All of the DAC pins share a single interrupt vector, which means the firmware will need to read the DAC port to determine which pin or pins caused an interrupt.

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

14.2.5 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.

15.0 Truth Tables

Table 15-1. 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 --> 111 Ack In - Status Out
Ack In - Status Out	**1111**	accept	TX cnt	Check	This mode is changed by SIE on issuance of ACK -->1110

Table 15-3. 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 15-1*				Setup	<= 10	data	valid	updates	1	updates	1	UC	UC	1	0	0	0	1	ACK	yes
See *Table 15-1*				Setup	> 10	junk	x	updates	updates	updates	1	UC	UC	UC	NoChange				ignore	yes
See *Table 15-1*				Setup	x	junk	invalid	updates	0	updates	1	UC	UC	UC	NoChange				ignore	yes
Disabled																				
0	0	0	0	x	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
Nak In/Out																				
0	0	0	1	Out	x	UC	x	UC	UC	UC	UC	UC	1	UC	NoChange				NAK	yes
0	0	0	1	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
3	2	1	0	token	count	buffer	dval	DTOG	DVAL	COUNT	Setup	In	Out	ACK	3	2	1	0	response	int
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

Table 15-3. Details of Modes for Differing Traffic Conditions (continued)

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	UC	UC	UC	UC	ignore	no
0	0	1	0	Out	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	UC	UC	UC	UC	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																				
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	NoChange				ignore	yes
1	0	0	1	Out	x	junk	invalid	updates	0	updates	UC	UC	1	UC	NoChange				ignore	yes
1	0	0	1	In	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
NAK Out/erroneous In																				
1	0	0	0	Out	<= 10	UC	valid	UC	UC	UC	UC	UC	1	UC	NoChange				NAK	yes
1	0	0	0	Out	> 10	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
1	0	0	0	Out	x	UC	invalid	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
1	0	0	0	In	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
Isochronous endpoint (Out)																				
0	1	0	1	Out	x	updates	updates	updates	updates	updates	UC	UC	1	1	NoChange				RX	yes
0	1	0	1	In	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
In endpoint																				
Normal In/erroneous Out																				
1	1	0	1	Out	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
1	1	0	1	In	x	UC	x	UC	UC	UC	UC	1	UC	1	1	1	0	0	ACK (back)	yes
NAK In/erroneous Out																				
1	1	0	0	Out	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
1	1	0	0	In	x	UC	x	UC	UC	UC	UC	1	UC	UC	NoChange				NAK	yes
Isochronous endpoint (In)																				
0	1	1	1	Out	x	UC	x	UC	UC	UC	UC	UC	UC	UC	NoChange				ignore	no
0	1	1	1	In	x	UC	x	UC	UC	UC	UC	1	UC	UC	NoChange				TX	yes

16.0 Absolute Maximum Ratings

Storage Temperature .....–65°C to +150°C
Ambient Temperature with Power Applied .....–0°C to +70°C
Supply Voltage on V_{CC} relative to V_{SS}.....–0.5V to +7.0V
DC Input Voltage..... –0.5V to +V_{CC}+0.5V
DC Voltage Applied to Outputs in High Z State ..... –0.5V to + V_{CC}+0.5V
Max. Output Current into Port 0,1,2,3 and DAC[1:0] Pins..... 60 mA
Max. Output Current into DAC[7:2] Pins ..... 10 mA
Power Dissipation .....300 mW
Static Discharge Voltage ..... >2000V
Latch-up Current ..... >200 mA

17.0 DC Characteristics

Fosc = 6 MHz; Operating Temperature = 0 to 70°C

	Parameter	Min	Max	Units	Conditions
	General				
$V_{CC\ (1)}$	Operating Voltage	4.0	5.5	V	Non USB activity (note 3)
$V_{CC\ (2)}$	Operating Voltage	4.35	5.25	V	USB activity (note 4)
I_{CC1}	V_{CC} Operating Supply Current		40	mA	V_{CC}=5.5V
I_{CC2}	V_{CC} = 4.35V		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	Vcc = 5.0V, 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} IO Sink Current		60	mA	Cumulative across all ports (note 10)
	Power On Reset				
t_{vccs}	V_{CC} Reset Slew	0.001	200	ms	Linear ramp: 0 to 4.35V (notes 6,7)
	USB Interface				
V_{oh}	Static Output HIGH	2.8	3.6	V	15k ±5% ohms to Gnd (note 4)
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} (note 14)
R_{pu}	Bus Pull-up Resistance (Ext. 3.3V option)	1.425	1.575	kΩ	1.5 kΩ ± 5% to 3.0–3.6V
R_{pd}	Bus Pull-down Resistance	14.25	15.75	kΩ	15 kΩ ± 5%
	General Purpose I/O Interface				
R_{up}	Pull-up Resistance	4.9K	9.1K	Ohms	
V_{ith}	Input Threshold Voltage	45%	65%	V_{CC}	All ports, LOW to HIGH edge
V_H	Input Hysteresis Voltage	6%	12%	V_{CC}	All ports, HIGH to LOW edge
I_{ol}	Sink Current	7.2	16.5	mA	Port 3, Vout = 1.0V (note 3)
I_{ol}	Sink Current	3.5	10.6	mA	Port 0,1,2, Vout = 2.0V (note 3)
I_{oh}	Source Current	1.4	7.5	mA	Voh = 2.4V (all ports 0,1,2,3) (note 3)
	DAC Interface				
R_{up}	Pull-up Resistance	8.0K	20.0K	Ohms	(note 15)
$I_{sink0(0)}$	DAC[7:2] Sink Current (0)	0.1	0.3	mA	Vout = 2.0 V DC (note 4,15)
$I_{sink0(F)}$	DAC[7:2] Sink Current (F)	0.5	1.5	mA	Vout = 2.0 V DC (note 4,15)
$I_{sink1(0)}$	DAC[1:0] Sink Current (0)	1.6	4.8	mA	Vout = 2.0 V DC (note 4,15)
$I_{sink1(F)}$	DAC[1:0] Sink Current (F)	8	24	mA	Vout = 2.0 V DC (note 4,15)
I_{range}	Programmed Isink Ratio: max/min	4	6		Vout = 2.0 V DC (notes 4,11,15)
I_{lin}	Differential Nonlinearity		0.5	lsb	Any pin (note 8,15)
t_{sink}	Current Sink Response Time		0.8	µs	Full scale transition (note 15)
T_{ratio}	Tracking Ratio DAC[1:0] to DAC[7:2]	14	21		Vout = 2.0V (note 9,15)

18.0 Switching Characteristics

Parameter	Description	Min.	Max.	Unit	Conditions
	Clock				
t_{CYC}	Input Clock Cycle Time	165.0	168.3	ns	
t_{CH}	Clock HIGH Time	0.45 t_{CYC}		ns	
t_{CL}	Clock LOW Time	0.45 t_{CYC}		ns	
	USB Driver Characteristics				
t_r	Transition Rise Time	75		ns	CLoad = 50 pF (notes 4,5)
t_r	Transition Rise Time		300	ns	CLoad = 600 pF (notes 4,5)
t_f	Transition Fall Time	75		ns	CLoad = 50 pF (notes 4,5)
t_f	Transition Fall Time		300	ns	CLoad = 600 pF (notes 4,5)
t_{rfm}	Rise/Fall Time Matching	80	125	%	t_r/t_f (note 13)
V_{crs}	Output Signal Crossover Voltage	1.3	2.0	V	
	USB Data Timing				
t_{drate}	Low Speed Data Rate	1.4775	1.5225	Mbs	Ave. Bit Rate (1.5Mb/s ± 1.5%)
t_{djr1}	Receiver Data Jitter Tolerance	–75	75	ns	To Next Transition (note 12)
t_{djr2}	Receiver Data Jitter Tolerance	–45	45	ns	For Paired Transitions (note 12)
t_{deop}	Differential to EOP Transition Skew	–40	100	ns	(note 10)
t_{eopr1}	EOP Width at Receiver	330		ns	Rejects as EOP (note 12)
t_{eopr2}	EOP Width at Receiver	675		ns	Accepts as EOP (note 12)
t_{eopt}	Source EOP Width	1.25	1.50	µs	
t_{udj1}	Differential Driver Jitter	–95	95	ns	To next transition, *Figure 18-5*
t_{udj2}	Differential Driver Jitter	–150	150	ns	To paired transition, *Figure 18-5*

Notes:

3. Functionality is guaranteed of the $V_{CC\ (1)}$ range, except USB transmitter and DACs.
4. USB transmitter functionality is guaranteed over the $V_{CC\ (2)}$ range, as well as DAC outputs.
5. Per Table 7-7 of revision 1.1 of USB specification, for C_{LOAD} of 50–600 pF.
6. Port 3 bit 7 controls whether the parts goes into suspend after a POR event or waits 128 ms to begin running.
7. POR will re-occur whenever V_{CC} drops to approximately 2.5V.
8. Measured as largest step size vs. nominal according to measured full scale and zero programmed values.
9. T_{ratio} = Isink1[1:0](n)/Isink0[7:2](n) for the same n, programmed.
10. Total current cumulative across all Port pins flowing to V_{SS} is limited to minimize Ground-Drop noise effects.
11. Irange: Isinkn(15)/ Isinkn(0) for the same pin.
12. Measured at crossover point of differential data signals.
13. Tested at 200 pF.
14. Limits total bus capacitance loading (C_{LOAD}) to 400 pF per section 7.1.5 of revision 1.1 of USB specification.
15. DAC I/O Port not bonded out on CY7C63612/13. See note on page 16 for firmware code needed for unused pins.

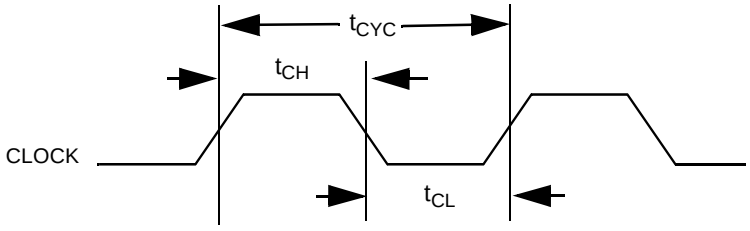


Figure 18-1. Clock Timing

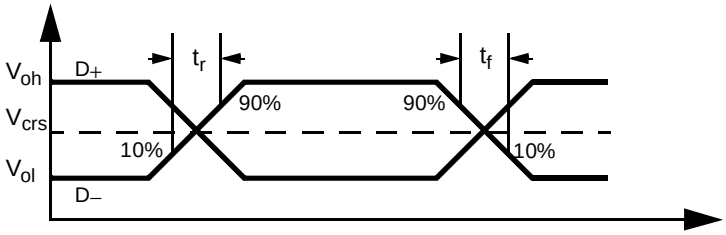


Figure 18-2. USB Data Signal Timing

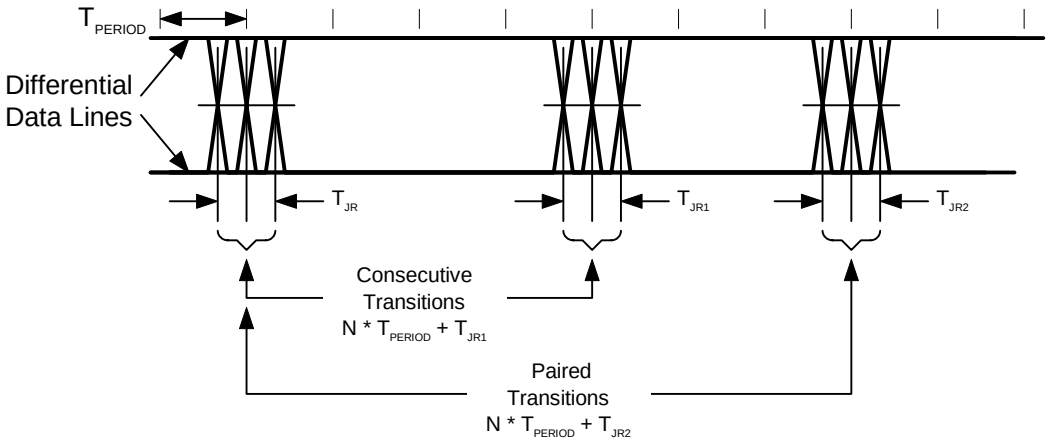


Figure 18-3. Receiver Jitter Tolerance

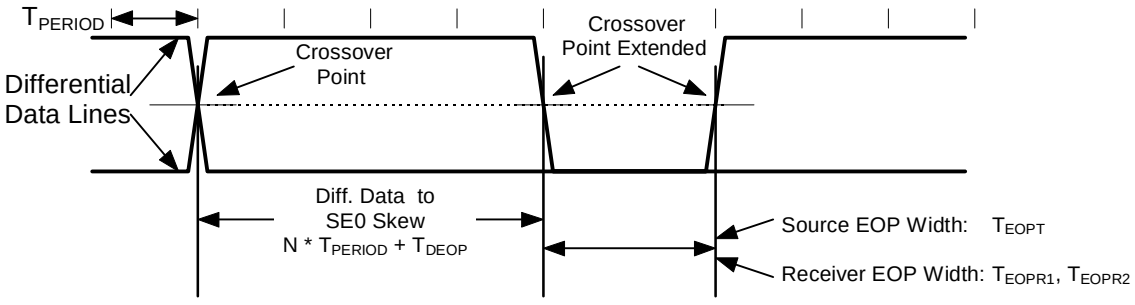


Figure 18-4. Differential to EOP Transition Skew and EOP Width

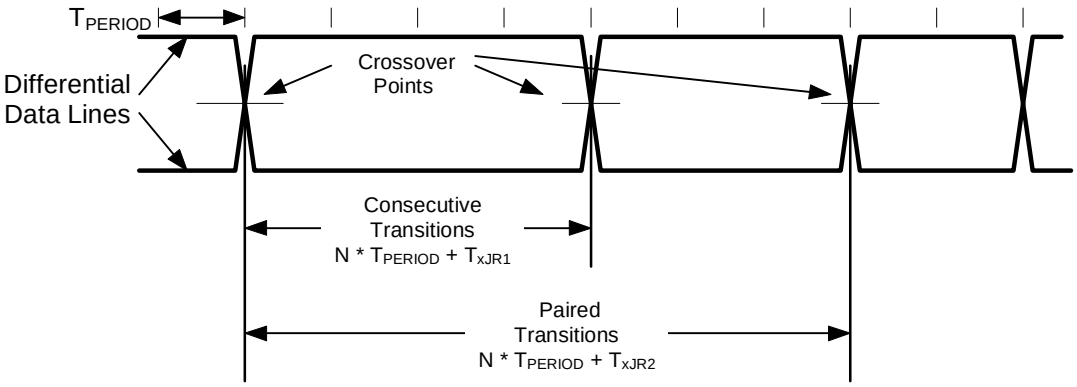


Figure 18-5. Differential Data Jitter

19.0 Ordering Information

Ordering Code	EPROM Size	Package Name	Package Type	Operating Range
CY7C63612-SC	6 KB	S13	24-Pin (300-Mil) SOIC	Commercial
CY7C63613-SC	8 KB	S13	24-Pin (300-Mil) SOIC	Commercial

Document #: 38-00754

20.0 Package Diagram

24-Lead (300-Mil) Molded SOIC S13