CY4625

MoBL-USB™ Bridge Firmware Guide

Document # 001-15686 Version **

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone (USA): 800.858.1810
Phone (Intnl): 408.943.2600
http://www.cypress.com

Copyrights

© Cypress Semiconductor Corporation, 2007. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Cypress, the Cypress Logo, MoBL-USB™ and EZ-USB FX2LP™ are trademarks or registered trademarks of Cypress Semiconductor. I^2C is a registered trademark of Philips. Windows is a registered trademark of Microsoft Corporation. All product and company names mentioned in this document are the trademarks of their respective holders.

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer

CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.

Contents

1. Overview **7**

1.1 Purpose 7
1.2 Chapter Overviews 7
1.3 Support 7
1.3.1 Acronyms 8
1.4 Document History 8
1.5 Document Conventions 8

2. Introduction **9**

2.1 Introducing the MoBL-USB™ Bridge Firmware 9
2.1.1 Features 9
2.2 Applications 10
2.2.1 System Diagram 11
2.3 Ordering Information 13

3. Functional Overview **15**

3.1 USB Signaling Speed 15
3.2 Boot Method 15
3.3 Resets and Wakeup 15
3.3.1 Reset 15
3.3.2 USB Reset 16
3.3.3 Wakeup 16
3.4 Endpoint RAM Organization 16
3.5 External Interface 16
3.5.1 Control Signals 17
3.5.1.1 FIFOADDR Lines 17
3.5.1.2 Read: SLOE and SLRD 18
3.5.1.3 Write: SLWR 18
3.5.1.4 PKTEND 18
3.5.2 IFCLK 18
3.5.3 FIFO Access 18
3.5.4 FIFO Flag Pins Configuration 19
3.5.5 Default FIFO Programmable Flag Setup 19
3.5.6 FIFO Programmable Flag Setup 19
3.5.7 Command Protocol 20
3.5.7.1 Write Register Example 20
3.5.7.2 Write Register with Response Example 20
3.5.7.3 Read Register Example 21
3.5.8 Enabling OUT Transfers on EP2 and EP6 21
3.5.9 Commands to Initialize Extended Endpoints EP1 OUT, EP4OUT, and EP8IN 21
3.5.10 Getting EP1OUT Data 22

3.5.11 Getting EP2OUT Data 22
3.5.12 Getting EP4OUT Data 24
3.5.13 Putting EP1IN Data 24
3.5.13.1 EP1IN and EP8IN Data Commit Bit Settings 25
3.5.14 Putting EP8 IN Data 25
3.5.15 Putting EP6IN Data - Full Packet 26
3.5.16 Putting EP6IN Data - Short Packet 26
3.5.17 Bridge Firmware Images and Standby Sequences 27
3.5.17.1 Sleep on Boot 27
3.5.17.2 Host Initiated Suspend 27
3.5.17.3 Cable Unplug Initiated Suspend 28
3.5.18 Command List 30

4. Enumeration 31

4.1 Manual Enumeration 31
4.2 Default Enumeration 32
4.3 External Master Passthrough Enumeration 32
4.4 Interrupt System 34
4.4.1 Architecture 34
4.4.2 INTENABLE Register Bit Definition 34
4.4.3 Standard Interrupt Bit Positions and Extended Interrupts 35
4.4.3.1 INTMODE - Interrupt Mode 35
4.4.3.2 RDREGRSP - Read Register Response 35
4.4.3.3 INTCODE[3:0] - Interrupt Code 35
4.4.3.4 PL[9:0] - Packet Length for OUT Packets 36
4.4.3.5 EP[3:0] - Endpoint Number for INNF, OUTNE, and INSHORTPKTACK 36

5. Endpoint 0 37

5.1 SETUP Commands 37

6. Register Summary 39

6.1 Register Summary 39
6.2 IFCONFIG Register 0x01 40
6.3 FLAGSAB/FLAGSCD Registers 0x02/0x03 41
6.4 POLAR Register 0x04 43
6.5 REVID Register 0x05 44
6.6 EPxCFG Register 0x06–0x09 44
6.7 EPxPKTLENH/L Registers 0x0A–0x0F 45
6.8 EPxPFH/L Registers 0x12–0x17 46
6.8.1 DECIS: EPxPFH.7 47
6.8.2 PKSTAT: EPxPFH.6 47
6.8.3 IN: PKTS(2:0)/OUT: PFC[12:10]: EPxPFH[5:3] 47
6.8.3.1 IN Endpoints 47
6.8.3.2 OUT Endpoints 48
6.9 EPxFLAGS Registers 0x1E–0x1F 48
6.10 INPKTEND/FLUSH Register 0x20 48
6.11 USBFRAMEH/L Registers 0x2A, 0x2B 49
6.12 MICROFRAME Registers 0x2C 49
6.13 FNADDR Register 0x2D 49
6.14 INTENABLE Register 0x2E 50
6.15 Extended Interrupt INTENABLE1 Register 0x2F 51

6.16 EP0STALL Register 0x32 52
6.17 CPUCS Unindexed Register 0xE600 52
6.17.1 Example Unindexed Register Read 53
6.17.2 Example Un-indexed Register Write 53

7. Default Descriptor 55
7.1 Introduction to the Default Descriptor 55

8. Pin Assignments 59
8.1 56-Pin SSOP 59
8.2 56-Pin QFN 60
8.3 56-Pin VFBGA 61
8.4 MoBL-USB™ FX2LP18 Device Pin Descriptions 62

9. External Interface Description 65
9.1 External Interface Pin Description 65

10. DMA Interface 67
10.1 Fast Slave Interface to DMA Interface on External Processor 67
10.2 DREQ Waveform Generation with FLAGB 68
10.2.1 OUT Packet Read Bursts 68
10.2.2 IN Packet Write Bursts 69

11. Package Diagrams 71
11.1 56-pin SSOP Package 71
11.2 56-pin QFN Package 72
11.3 56-pin VFBGA Package 73

Index 75

1. Overview

1.1 Purpose

The purpose of this manual is to help developers understand how the MoBL-USB Bridge Firmware works.

1.2 Chapter Overviews

Table 1-1. Overview of the MoBL-USB Bridge Firmware Guide Chapters

Chapter	Description
Overview (on page 7)	Describes the purpose of this guide, overviews each chapter, supplies product support and upgrade information, and acronyms. This chapter lists the document history and the document conventions.
Introduction (on page 9)	Introduces the firmware features, applications, and ordering information.
Functional Overview (on page 15)	Lists USB signaling speed, boot method, resets and wakeup, endpoint RAM organization, external interface, and command lists.
Enumeration (on page 31)	Discusses manual enumeration, default enumeration, external master passthrough enumeration, and the interrupt system.
Endpoint 0 (on page 37)	Presents the setup commands.
Register Summary (on page 39)	Presents a table that summarizes all the commands.
Default Descriptor (on page 55)	Lists example code.
Pin Assignments (on page 59)	Includes figures of the 56-Pin SSOP, 56-Pin QFN, and 56-Pin VFBGA. A table lists the MoBL-USB FX2LP18 device pin descriptions.
External Interface Description (on page 65)	Presents a table that lists the FX2LP pin names and the cross reference pins to the Bridge Slave FIFO.
DMA Interface (on page 67)	Describes the fast slave interface to DMA interface on an external processor, and the DREQ waveform generation with FLAGB which includes the OUT packet read burst and the In packet write burst.
Package Diagrams (on page 71)	Presents the packages for the 56-pin SSOP, 56-pin QFN, and the 56-pin VFBGA.

1.3 Support

Technical Support can be reached at http://www.cypress.com/support or can be contacted by phone at: 1-800-541-4736.

The following related documentation can be found on the Cypress web site.

Document Number	Document Type	Document Name
38-08032	Data Sheet	EZ-USB FX2LP™ USB Microcontroller
001-06120	Data Sheet	MoBL-USB™ FX2LP18 USB Microcontroller
001-11981	TRM	MoBL-USB™ FX2LP18 Technical Reference Manual
001-13670	TRM	EZ-USB® Technical Reference Manual

1.3.1 Acronyms

The following are acronyms used throughout this user guide.

Table 1-2. Acronyms

Acronym	Description
ADC	analog-to-digital converter
API	application programming interface
C	(refers to C programming language)
DAC	digital-to-analog converter
EE	empty flag
EEPROM	electrical erasable programmable read only memory
FF	full flag
IO	input/output
ISR	interrupt service routine
MCU	microcontroller unit
MHz	megahertz
MSC	mass storage class
MTP	media transfer protocol
PF	programmable flag
PLL	phase lock loop
PWM	pulse width modulator
RAM	random access memory
ROM	read only memory
USB	universal serial bus

1.4 Document History

This section serves as a chronicle of the *CY4626 MoBL-USB™ Bridge Firmware Guide*.

CY4626 MoBL-USB™ Bridge Firmware Guide History

Release Date	Firmware Revision	Manual Version	Originator	Description of Change
05/24/07	5.3	**	ARI	This manual is a new document to the Cypress Document Control (Revision **) system; this manual has gone through several versions as a data sheet.

1.5 Document Conventions

This manual uses the `Courier New` font to distinguish code examples from regular text. File names are presented in *italics* text.

2. Introduction

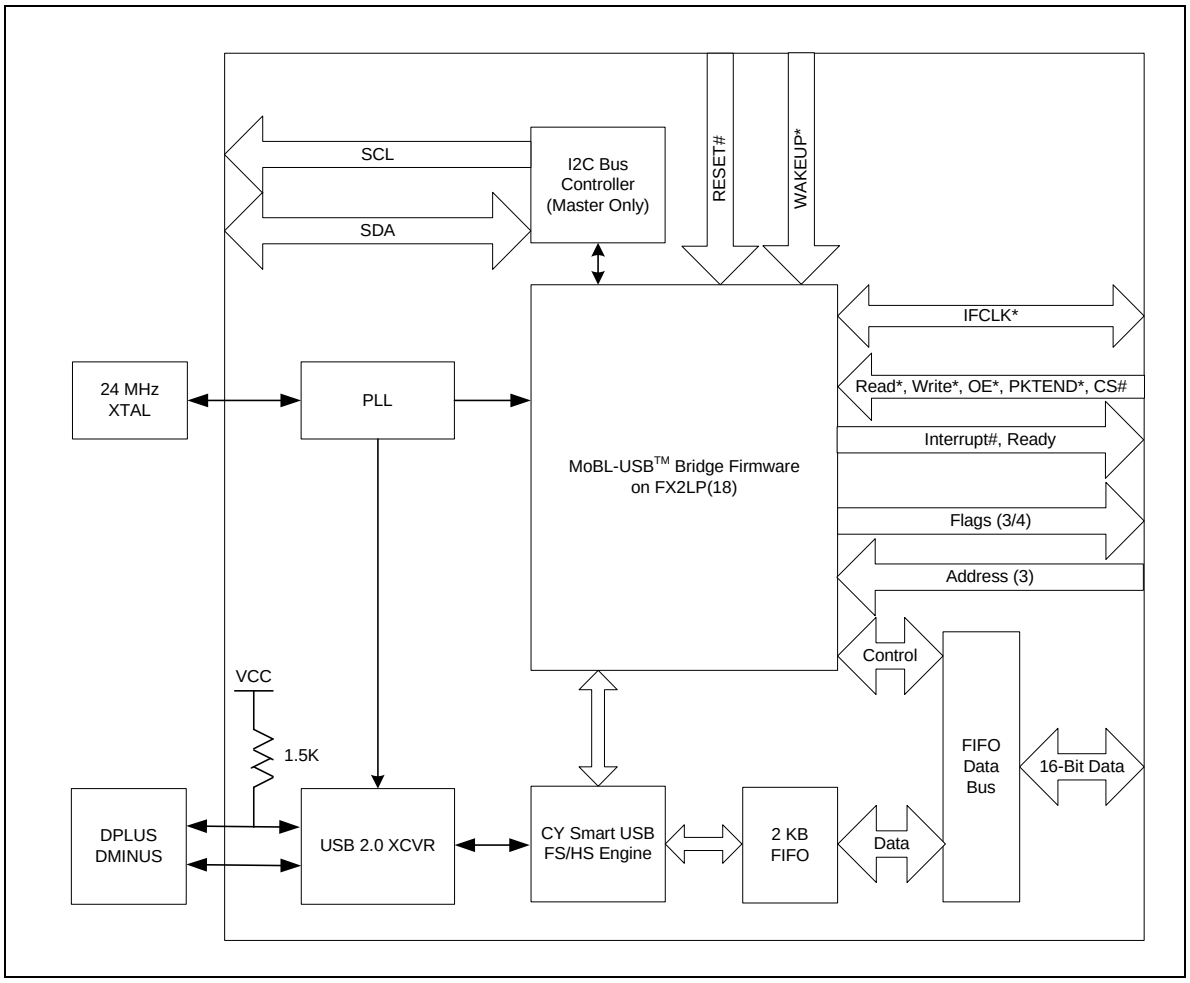
2.1 Introducing the MoBL-USB™ Bridge Firmware

The MoBL-USB Bridge firmware is designed to work with any external master, such as standard microprocessors, DSPs, ASICs, and FPGAs to enable USB 2.0 support for any peripheral design. The Bridge firmware operates on both the EZ-USB FX2LP™ and MoBL-USB™ FX2LP18 chips, which have a built-in USB transceiver and Serial Interface Engine (SIE), along with a command decoder for sending and receiving USB data. The controller has two double buffered high-speed capable endpoints that share a 2 KB FIFO space for maximum flexibility and throughput, as well as Control Endpoint 0. The Bridge firmware exposes three FIFO address pins and a 16 bit data bus for both command and data input or output.

2.1.1 Features

- USB 2.0 compliant
- Operates at high (480 Mbps) or full (12 Mbps) speed
- Supports Control Endpoint 0
  - Used for handling USB device requests
- Supports two double buffered high-speed endpoints that share a 2 KB FIFO space
  - Endpoints 2 and 6 for application specific control and data
- Supports two additional single buffered high-speed endpoints that share buffer space with the command interface
  - Endpoints 4 and 8 for application specific control and data
- Bidirectional Endpoint 1 to support interrupt transfers
- Standard 16 bit Command and Data FIFO interface
  - Glueless interface to most standard microprocessors DSPs, ASICs, and FPGAs
  - Synchronous or Asynchronous interface
- Targeted to CY7C68013/14/53A FX2LP in 56 pin SSOP, QFN, and VFBGA packages
- Complies with most device class specifications
- Supports Microsoft's PMP (Portable Media Player) specification

Figure 2-1. Block Diagram



2.2 Applications

- Cell phones
- DSL modems
- ATA interface
- Memory card readers
- Legacy conversion devices
- Cameras
- Scanners
- Home PNA
- Wireless LAN
- MP3 players
- Portable media players
- Networking
- Printers

The Reference Designs section of the Cypress web site provides additional tools for typical USB applications. Each reference design kit comes complete with firmware source code and object code, schematics, and documentation. See the Cypress web site at http://www.cypress.com.

2.2.1 System Diagram

Figure 2-2. Example USB System Diagram

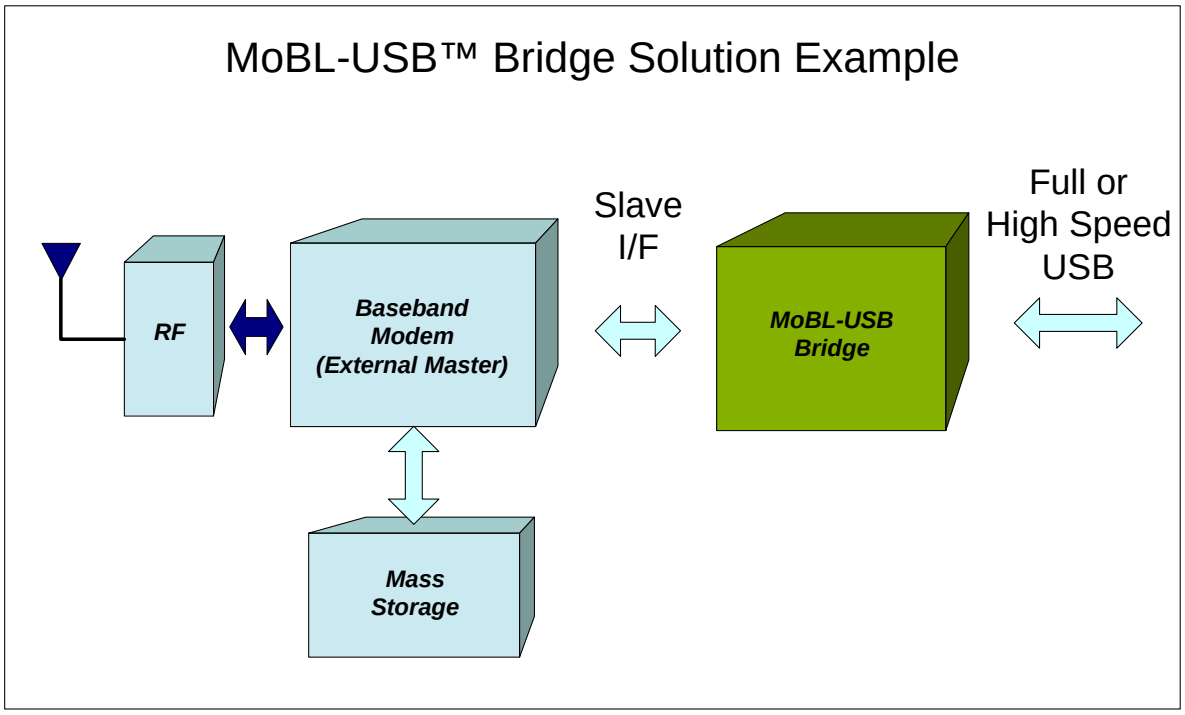
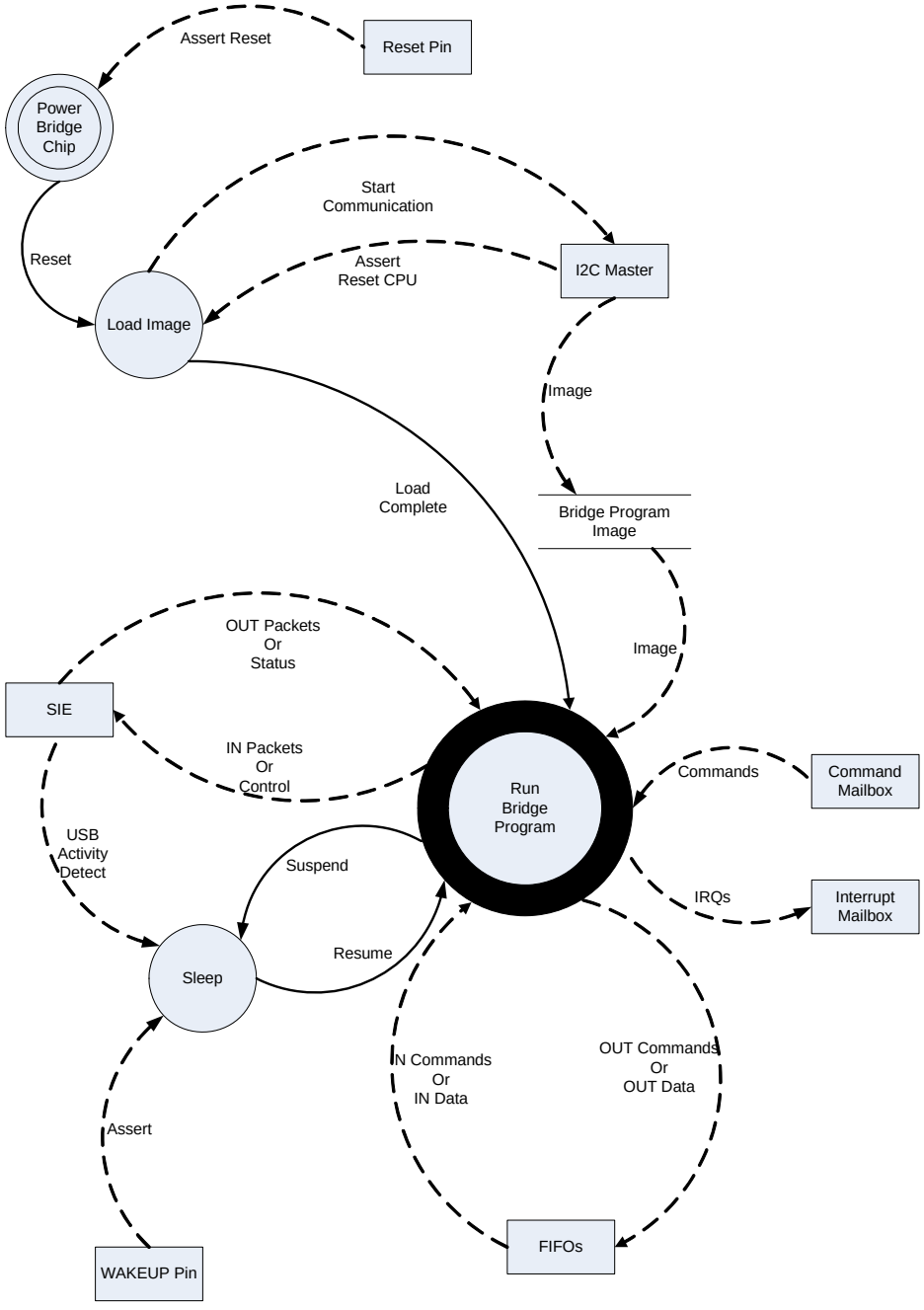


Figure 2-3. Bridge System States



2.3 Ordering Information

Use the following table to select the appropriate FX2LP part for the application. Refer to the data sheet for the selected part for pinout information.

Table 2-1. Ordering Information

Ordering Code	Package Type	RAM Size	# Prog I/Os
Ideal for battery powered applications			
CY7C68014A-56PVXC	FX2LP 56 SSOP – Lead-free	16K	24
CY7C68014A-56LFXC	FX2LP 56 QFN – Lead-free	16K	24
CY7C68014A-56BAXC	FX2LP 56 VFBGA – Lead-free	16K	24
CY7C68053-56BAXI	FX2LP18 56 VFBGA – Lead-free	16K	24
Ideal for non-battery powered applications			
CY7C68013A-56PVXC	FX2LP 56 SSOP – Lead-free	16K	24
CY7C68013A-56PVXI	FX2LP 56 SSOP – Lead-free (Industrial)	16K	24
CY7C68013A-56LFXC	FX2LP 56 QFN – Lead-free	16K	24
CY7C68013A-56LFXI	FX2LP 56 QFN – Lead-free (Industrial)	16K	24
CY7C68013A-56BAXC	FX2LP 56 VFBGA – Lead-free	16K	24
Development Tool Kit			
CY3684	EZ-USB FX2LP Development Kit		
CY3687	MoBL-USB FX2LP18 Development Kit		
Reference Design Kit			
CY4624	Media Transfer Protocol for Microsoft Portable Media Center		

3. Functional Overview

The intended use of this Bridge firmware is with class drivers such as Mass Storage Class (MSC) or Media Transfer Protocol (MTP). Therefore, for OUT transfers with data bus equal to 16 bits (word-wide), the master processor must ignore the extra byte for an odd packet size. For IN transfers with an odd number of bytes, the last odd byte is sent in the IN Short Packet command (see Command List on page 30).

3.1 USB Signaling Speed

The Bridge firmware operates at two of the three rates defined in the *Universal Serial Bus Specification Revision 2.0*, dated April 27, 2000.

- Full-speed, with a signaling bit rate of 12M bits per second
- High-speed, with a signaling bit rate of 480M bits per second
- The Bridge firmware does not support the low-speed signaling rate of 1.5M bits per second.

3.2 Boot Method

Upon reset, MoBL-USB chips automatically look for and load from an EEPROM attached to the SDA and SCL signals. If an image is found and the first byte matches the proper signature value (0xC2), then the chip loader loads the image contained in the EEPROM. The eighth byte (byte 7) is the configuration byte and is set to 0x40 for the supplied Bridge images. This value sets the device in disconnect and sets the I^2C™ interface speed to approximately 100 KHz or the default speed. If this value is modified in the image to 0x41, then the loader changes the EEPROM access speed to approximately 400 KHz.

Either an EEPROM can be used with one of the images identified in Bridge Firmware Images and Standby Sequences on page 27 loaded on the EEPROM, or the external master can be connected to the MoBL-USB SDA and SCL lines and emulate a slave EEPROM device, providing one of the images to the loader. The MoBL-USB chip is an I2C master-only device therefore, if the external master does not have a slave-only I2C interface, then bit-banging general purpose IO pins may be used to provide the slave I2C interface.

3.3 Resets and Wakeup

The Bridge firmware replies to a reset, USB reset, and wakeup call.

3.3.1 Reset

An input pin (RESET#) resets the chip. The internal PLL stabilizes after V_{CC} has reached 3.3V. Typically, an external RC network (R = 100K ohms, C = 0.1 µf) is used to provide the RESET# signal. The clock must be in a stable state for at least 200 µs before the RESET is released.

3.3.2 USB Reset

When the Bridge firmware detects a USB reset condition on the USB bus, Bridge firmware clears the data toggle bits for all endpoints, flushes all FIFOs, asserts the USB reset interrupt, then handles it the same as any other enumeration sequence. This means that the Bridge firmware enumerates again and asserts the ENUMOK interrupt to let the external master know that it has enumerated. The external master is then responsible for configuring the Bridge firmware for the application. The external master must also check whether the Bridge enumerated at High or Full speed in order to adjust the EPxPKTLENH/L register values accordingly. The last initialization task is for the external master to flush all of the Bridge firmware FIFOs.

3.3.3 Wakeup

The Bridge firmware exits its low power state when one of the following events occur:

- USB bus signals a resume. The Bridge firmware asserts a HOST_RESUME extended interrupt.
- The external master asserts the WAKEUP pin. The Bridge firmware asserts a READY interrupt.

3.4 Endpoint RAM Organization

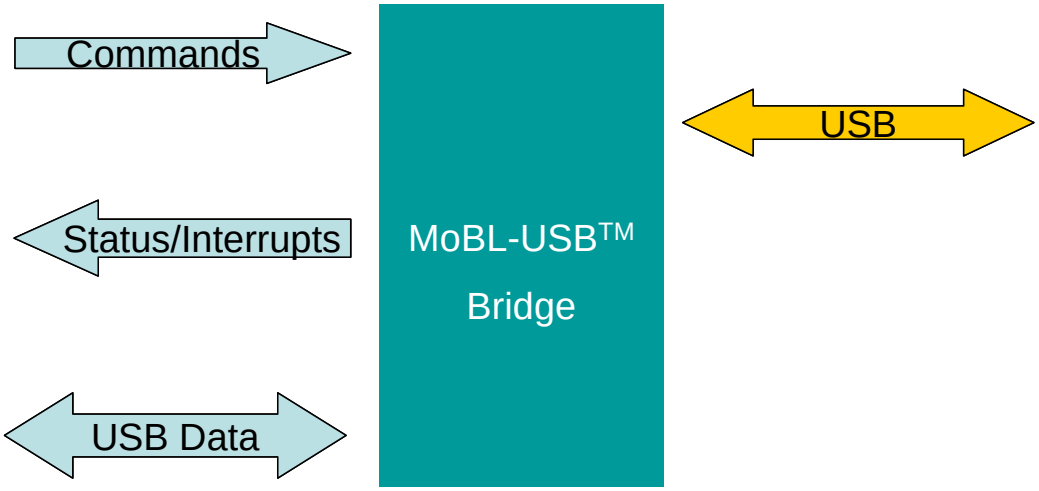
- EP0–Bidirectional Endpoint 0, 64-byte buffer.
- EP2 and EP6 – Four 512-byte buffers, bulk or interrupt. EP2 and EP6 are high-speed double-buffered endpoints. Applications configure EP2 for OUT and EP6 for IN packets.
- EP4OUT and EP8IN are available through shifted mode of the command interface. They are high-speed endpoints that are bulk or interrupt and single buffered.
- EP1IN is a 64-byte endpoint. It is intended for use as an interrupt endpoint and is single-buffered. It is accessed via the command interface.
- EP1OUT is a 64-byte endpoint. It is single-buffered. It is accessed via the command interface.

3.5 External Interface

The Bridge firmware presents two interfaces to the external master.

1. A double-buffered FIFO interface through which EP2 and EP6 data flows. This interface buffers up to two packets at the current transfer speed. At high-speed, each FIFO holds two 512-byte packets. At full-speed, each FIFO holds two 64-byte packets.
2. A single-buffered command interface, which is used to set up the Bridge firmware, read the status, load the descriptors, and access Endpoints 0 and 1. The extended endpoints 4 and 8 are also accessed through the command interface.

Figure 3-1. Bridge Interfaces



3.5.1 Control Signals

Bridge firmware has FIFOADDR control lines, the read controls: SLOE and SLRD, the write control: SLWR, and the PKTEND control line.

3.5.1.1 FIFOADDR Lines

The Bridge firmware has three address pins that are used to select either the FIFOs or the command interface. How the address lines correspond are listed in the following table.

Table 3-1. FIFO Address Lines Setting

Address/Selection	FIFOADR2	FIFOADR1	FIFOADR0
FIFO2	0	0	0
Command OUT	0	0	1
FIFO6	0	1	0
Command IN	0	1	1
Interrupt (read) Command done (write)	1	0	0
RESERVED	1	0	1
RESERVED	1	1	0
RESERVED	1	1	1

The Bridge firmware accepts either an internally derived clock (30 or 48 MHz) or externally supplied clock (IFCLK, 5 – 48 MHz), and SLRD, SLWR, SLOE, CS#, FIFOADR[2:0] signals from an external master. The interface is 16 bits wide; an Output Enable signal, SLOE, enables the data bus driver for read operations. The external master must ensure that the output enable signal is inactive when writing data to the Bridge firmware. The interface can operate either asynchronously where the SLRD and SLWR signals act directly as strobes, or synchronously where the SLRD and SLWR act as clock qualifiers. The optional CS# signal tri-states the data bus and ignores SLRD and SLWR. Since the Bridge only supports 16 bit wide data, the PKTEND is typically not used since it does not allow odd

byte size transfers. The Bridge firmware provides a command for committing short packets that also includes handling odd packet size transfers.

The external master reads from OUT endpoints and writes to IN endpoints, and it reads from or writes to the command interface.

3.5.1.2 *Read: SLOE and SLRD*

In synchronous mode, the FIFO pointer is incremented on each rising edge of IFCLK while SLRD is asserted. In asynchronous mode, the FIFO pointer is incremented on each asserted-to-deasserted transition of SLRD.

SLOE is a data bus driver enable. When SLOE is asserted, the data bus is driven by the Bridge firmware.

3.5.1.3 *Write: SLWR*

In synchronous mode, data on the FD bus is written to the FIFO (and the FIFO pointer is incremented) on each rising edge of IFCLK while SLWR is asserted. In asynchronous mode, data on the FD bus is written to the FIFO (and the FIFO pointer is incremented) on each asserted-to-deasserted transition of SLWR.

3.5.1.4 *PKTEND*

PKTEND commits the current buffer to USB. To send a short IN packet (one that has not been filled to max packet size determined by the value of PL[X:0] in EPxPKTLENH/L), the external master strobes the PKTEND pin.

Packet end can also be signaled using the IN Short Packet command shown in Command List on page 30. This is the preferred method in order to be able to write the last byte in the command for an odd packet length. If the PKTEND pin is not used, then it must be pulled high assuming that the default active low polarity is used.

The default configuration for this interface is active low. In order to change the polarity of the PKTEND pin, the master may write to the POLAR register at any time. In order to switch the polarity of the SLWR/SLRD/SLOE, the master must set bits 2, 3, and 4 respectively in the FIFOPINPOLAR register located at XDATA space 0xE609. Note that the Bridge powers up with the polarities set to low. POLAR Register 0x04 on page 43 provides further information on how to access this register located at XDATA space.

3.5.2 IFCLK

The IFCLK pin can be configured to be either an input (default) or an output interface clock. Bits IFCONFIG[7:4] define the behavior of the interface clock. To use the internally derived 30- or 48-MHz clock, set IFCONFIG.7 to ‘1’ and set IFCONFIG.6 to ‘0’ (30 MHz) or to ‘1’ (48 MHz). To use an externally supplied clock, set IFCONFIG.7=0 and drive the IFCLK pin (5 MHz – 48 MHz). The input or output IFCLK signal can be inverted by setting IFCONFIG.4=1.

3.5.3 FIFO Access

An external master can access the slave FIFOs either asynchronously or synchronously:

- Asynchronous – SLRD, SLWR, and PKTEND pins are strobes.
- Synchronous – SLRD, SLWR, and PKTEND pins are enables for the IFCLK clock pin.

An external master accesses the FIFOs through the data bus, FD [15:0]. This bus must be set for 16 bits wide; the width is selected via the WORDWIDE bit in the EPxPKTLENH/L registers. The data

bus is bidirectional, with its output drivers controlled by the SLOE pin. The FIFOADR[2:0] pins select which of the two FIFOs is connected to the FD [15:0] bus, or if the command interface is selected.

3.5.4 FIFO Flag Pins Configuration

The FIFO flags are FLAGA, FLAGB, FLAGC, and FLAGD. These FLAGx pins report the status of the FIFO selected by the FIFOADR[2:0] pins. At reset, these pins are configured to report the status of the following:

- FLAGA reports the status of the programmable flag.
- FLAGB reports the status of the full flag.
- FLAGC reports the status of the empty flag.
- FLAGD defaults to the CS# function.

The FIFO flags can either be indexed or fixed. Fixed flags report the status of a particular FIFO regardless of the value on the FIFOADR [2:0] pins. Indexed flags report the status of the FIFO selected by the FIFOADR [2:0] pins.[1] Applications configure EP2 as an OUT endpoint and typically set FLAGA to fixed signalling of EP2 empty/not-empty status; they configure EP6 as an IN endpoint and typically set FLAGB to fixed signalling of EP6 full/not full status. For reduced pin interfacing, a single FLAG pin can be used by setting it for the current direction of data flow and using the INNF and OUTNE interrupts to signal when the direction should be reversed.

3.5.5 Default FIFO Programmable Flag Setup

By default, FLAGA is the Programmable Flag (PF) for the endpoint being pointed to by the FIFOADR[2:0] pins. For EP2, the default endpoint configuration is BULK, OUT, 512, 2x, and the PF pin asserts when the entire FIFO has greater than/equal to 512 bytes. For EP6, the default endpoint configuration is BULK, IN, 512, 2x, and the PF pin asserts when the entire FIFO is less than or equal to 512 bytes. In other words, EP6 reports a half-empty state, and EP2 reports a half-full state. The polarity of the programmable flags are set in the EEPROM configuration bytes.

3.5.6 FIFO Programmable Flag Setup

Each FIFO's programmable-level flag asserts when the FIFO reaches a user-defined fullness threshold. That threshold is configured as follows:

1. For OUT packets: The threshold is stored in PFC12:0. The PF is asserted when the number of bytes in the entire FIFO is less than or equal to (DECIS = 0) or greater than/equal to (DECIS = 1) the threshold.
2. For IN packets, with PKTSTAT = 1: The threshold is stored in PFC9:0. The PF is asserted when the number of bytes written into the current packet in the FIFO is less than or equal to (DECIS = 0) or greater than/equal to (DECIS = 1) the threshold.
3. For IN packets, with PKTSTAT = 0: The threshold is stored in two parts: PKTS2:0 holds the number of committed packets, and PFC9:0 holds the number of bytes in the current packet. The PF is asserted when the FIFO is at or less full than (DECIS = 0), or at or more full than (DECIS = 1), the threshold.

Note that the Programmable Flags tests the buffer levels on word bounds, since the Bridge is a 16-bit interface only. This introduces ambiguity for odd packet lengths. Use a FLAG pin configured as EF for the OUT endpoint EP2 and a FLAG pin configured as FF for the IN endpoint EP6.

1. In indexed mode, the value of the FLAGx pins is indeterminate except when addressing a FIFO (FIFOADR[2:0]={000,001,010,011}).

3.5.7 Command Protocol

Commands are passed through by the same FIFO access method as data. To send a command, wait for the READY line to be high (or wait until the CMDRDY interrupt is received), select address [0 1 1] and write the command in the same manner as data. When all of the command data has been placed in the FIFO, complete the command with a write of 0x0000 to address [1 0 0]. For the command response signaling Setup Out Phase data, when the CMDSTATRDY interrupt is received, the response may be read from the command OUT FIFO. If an EP1OUTPKTAVAIL or EP4OUTPKTAVAIL interrupt occurs, then either the EP4OUT or EP1OUT packet data is available on the command OUT FIFO. The respective interrupts indicate the packet data size in bytes.

3.5.7.1 Write Register Example

Prior to writing to a register, two conditions must be met: FIFOADR[2:0] must hold [0 1 1], and READY must be high (the CMDRDY interrupt is received). The external master must not initiate a command if READY is low.

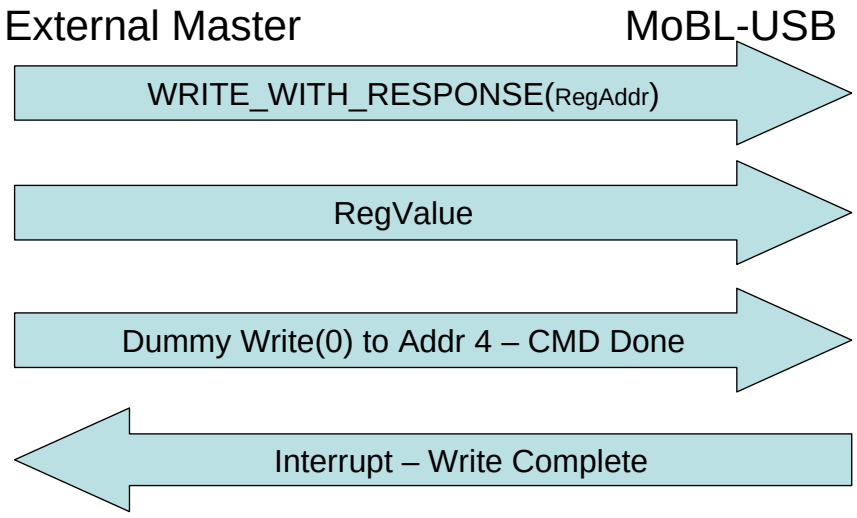
Example Write the byte <10110000> into the IFCONFIG register (0x01).

1. Wait for READY (pin signal or CMDRDY interrupt).
2. Select FIFOADR [0 1 1].
3. Write 0xB041 = 0x40+0x01 (Write Register command plus IFCONFIG register address in the low byte and the value 0xB0 in the high byte on the FIFO Data bus).
4. Write 0x0000 to FIFOADR [1 0 0] to commit the command.

3.5.7.2 Write Register with Response Example

Prior to writing to a register, two conditions must be met: FIFOADR[2:0] must hold [0 1 1] and READY must be high (or CMDRDY interrupt must be received). The external master must not initiate a command if READY is low.

Figure 3-2. Write Register With Response



Example Write the byte <10110000> into the IFCONFIG register (0x01).

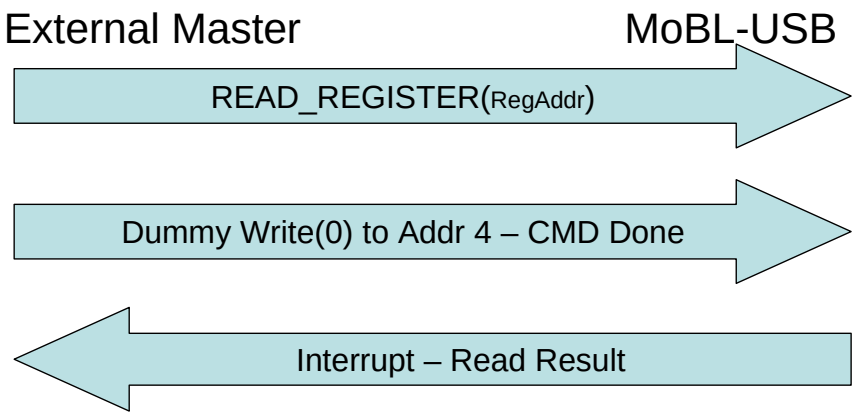
1. Wait for READY (pin signal or CMDRDY interrupt).
2. Select FIFOADR [0 1 1].

3. Write 0x0105 = Write register with response command 0x05 on the low byte and 0x01 (IFCONFIG register ID) on the high byte.
4. Write 0x00B0 = register data value 0xB0 on the low byte and 0x00 on the high byte.
5. Select FIFOADR [1 0 0].
6. Write 0x0000 to FIFOADR [1 0 0] to commit the command.
7. Wait for the Write Register Response (WRREGRSP) interrupt and verify the register ID in the interrupt matches the written register ID.

3.5.7.3 Read Register Example

Reading a register uses a similar protocol.

Figure 3-3. Read Register



Example Read from the IFCONFIG register (0x01).

1. Wait for READY (pin signal or CMDRDY interrupt).
2. Select FIFOADR [0 1 1].
3. Write 0x0104 = Read register command 0x04 on the low byte and 0x01 (IFCONFIG register ID) on the high byte.
4. Write 0x0000 to FIFOADR [1 0 0] to commit the command.
5. Select FIFOADR [0 0 1].
6. Wait for the Read Register Response (RDREGRSP) interrupt and verify the register ID in the interrupt matches the written register ID.
7. Read the register value from the interrupt response.

3.5.8 Enabling OUT Transfers on EP2 and EP6

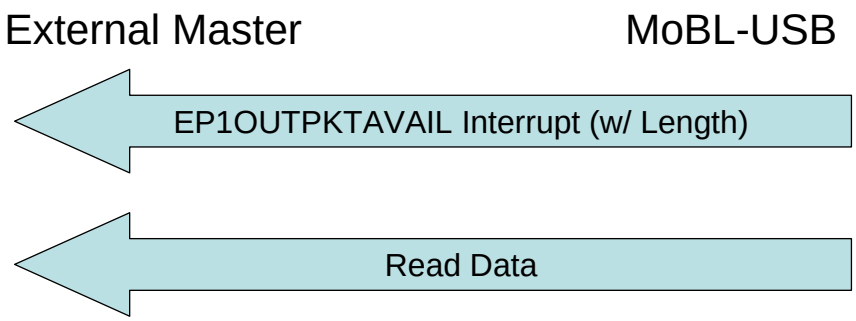
The external master must write the OUT Data Commit command with the endpoint specified before data is committed to the OUT FIFO for the endpoint.

3.5.9 Commands to Initialize Extended Endpoints EP1 OUT, EP4OUT, and EP8IN

Prior to using the endpoints EP1 OUT, EP4 OUT, and EP8 IN each of these extended endpoints must be initialized using one of the commands: EP1OUTEN, EP4OUTEN, EP8INEN. Each of these commands takes two parameters: enable and mode. The enable parameter enables or disables the endpoint. The mode command specifies whether the endpoint is an Interrupt type endpoint or a bulk type endpoint.

3.5.10 Getting EP1OUT Data

Figure 3-4. Read EP1OUT Data



1. Wait for READY (pin signal or CMDRDY interrupt).
2. Write the EP1OUTEN command with mode at FIFOADR[0 1 1].
3. Write 0x0000 at FIFOADR[1 0 0].
4. Wait for EP1OUTPKTAVAIL with length interrupt.
5. Read length bytes at FIFOADR [0 0 1].
6. Repeat at step 4.

3.5.11 Getting EP2OUT Data

Figure 3-5. Read OUT Data

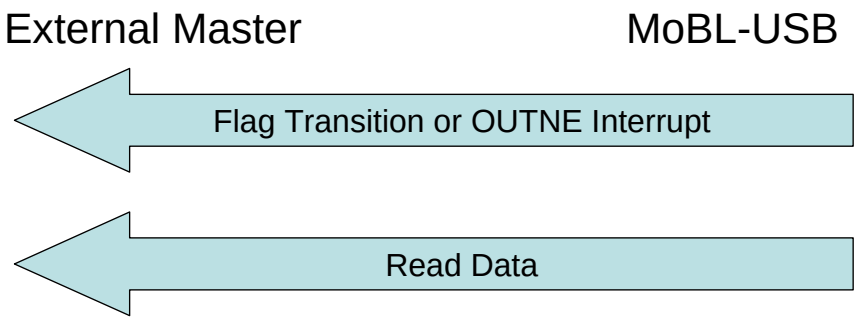
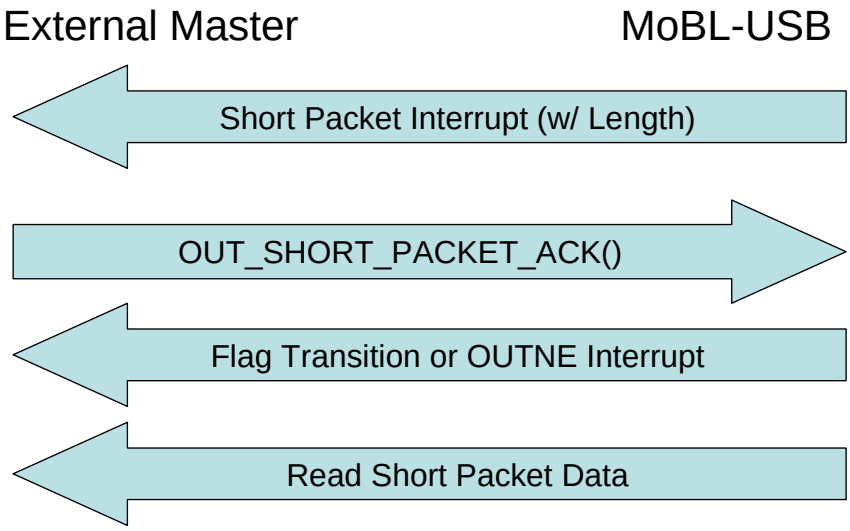


Figure 3-6. Read Short OUT Data

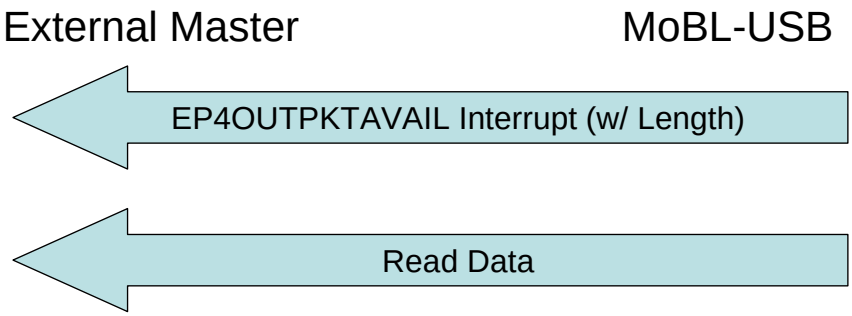


1. Wait for READY (pin signal or CMDRDY interrupt).
2. Write OUT Data Commit command at FIFOADR[0 1 1].
3. Write 0x0000 at FIFOADR [1 0 0].
4. Wait for Endpoint Empty Pin Flag to deassert (or get OUTNE interrupt) or get OUT Short Packet Received message interrupt.
5. If a get OUT Short Packet Received message interrupts with a short packet length, go to step 7.
6. Else Empty Pin Flag deasserts (or get OUTNE interrupt) then:
   a. Read full packet data from FIFOADR [0 0 0].
   b. Repeat at step 4.
7. Write OUT_SHORT_PKT_MSG_ACK command at FIFOADR [0 1 1].
8. Write 0x0000 at FIFOADR [1 0 0].
9. If short packet is not zero length then:
   a. Wait for Endpoint Empty Pin Flag to de-assert (or get OUTNE interrupt).
   b. Read short packet length data from FIFOADR [0 0 0].
10. Go to step 4.

Note that steps 7 and 8 are not required if the BYPASS_OUT_SHORT_PKT_ACK command was sent prior to starting transfers.

3.5.12 Getting EP4OUT Data

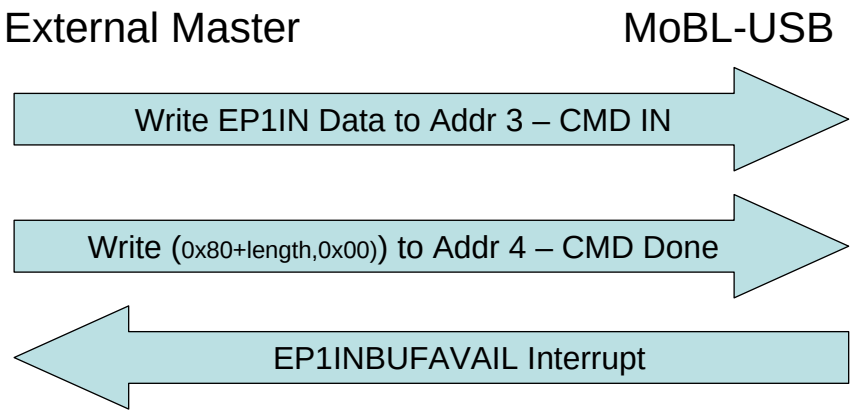
Figure 3-7. Read EP4OUT Data



1. Wait for READY (pin signal or CMDRDY interrupt).
2. Write EP4OUTEN command with mode at FIFOADR [0 1 1].
3. Write 0x0000 at FIFOADR [1 0 0].
4. Wait for EP4OUTPKTAVAIL with length interrupt.
5. Read length bytes from FIFOADR [0 0 1].
6. Go to step 4.

3.5.13 Putting EP1IN Data

Figure 3-8. Write EP1IN Data



1. Wait for READY (pin signal or CMDRDY interrupt).
2. Write 0 to 64 bytes of packet data at FIFOADR [0 1 1].
3. Write 0x80+packet length on the high byte and 0x00 on the low byte at FIFOADR [1 0 0] to commit the packet.
4. Wait for EP1INBUFAVAIL interrupt.
5. Go to step 1.

3.5.13.1 EP1IN and EP8IN Data Commit Bit Settings

Upper Data Commit Byte:

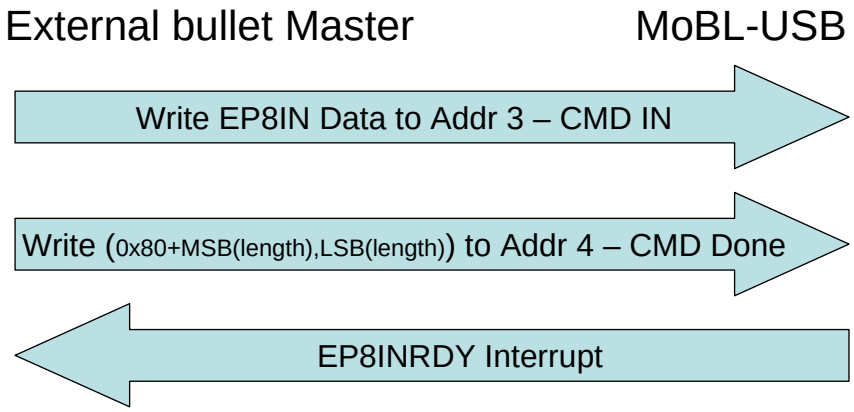
Bit #	7	6	5	4	3	2	1	0
Bit Name	EP1IN DATA ID	EP8IN DATA ID / EP1IN PL6	EP1IN PL5	EP1IIN PL4	EP1IN PL3	EP1IN PL2	EP1IN PL1 / EP8IN PL9	EP1IN PL0 / EP8IN PL8

Lower Data Commit Byte:

Bit #	7	6	5	4	3	2	1	0
Bit Name	EP8IN PL7	EP8IN PL6	EP8IN PL5	EP8IN PL4	EP8IN PL3	EP8IN PL2	EP8IN PL1	EP8IN PL0

3.5.14 Putting EP8 IN Data

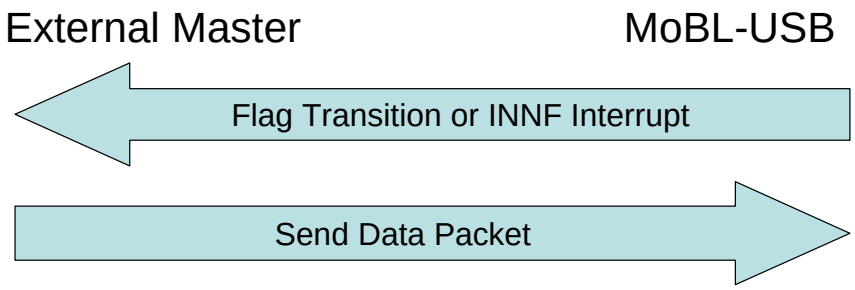
Figure 3-9. Write EP8IN Data



1. Wait for READY (pin signal or CMDRDY interrupt).
2. Write EP8INEN command with mode at FIFOADR[0 1 1].
3. Write 0x0000 at FIFOADR[1 0 0].
4. Wait for READY (pin signal or CMDRDY interrupt).
5. Wait for EP8INRDY interrupt.
6. Write 0 to 64 bytes (Full Speed) or 0 to 512 bytes (High Speed) of packet data at FIFOADR [0 1 1].
7. Write 0x40+upper packet length bits on the high byte and lower packet length bits on the low byte at address [1 0 0] to commit the packet.
8. Go to step 4.

3.5.15 Putting EP6IN Data - Full Packet

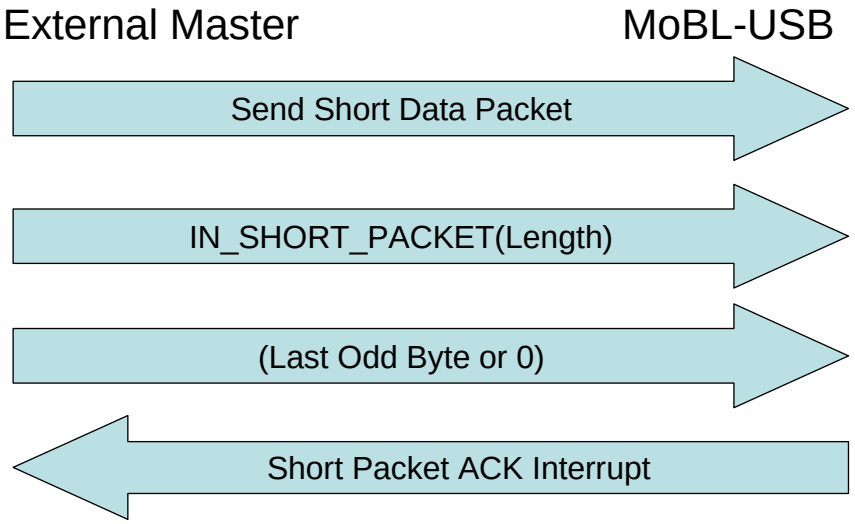
Figure 3-10. Write IN Data



1. Wait until a pin flag or INNF interrupt indicates a buffer is available in the Bridge.
2. Write 64 bytes (Full Speed) or 512 bytes (High Speed) of packet data at FIFOADR[0 1 0] (EP6).
3. Repeat for each packet in transfer.

3.5.16 Putting EP6IN Data - Short Packet

Figure 3-11. Write Short IN Data



1. Wait until a pin flag or INNF interrupt indicates a buffer is available in the Bridge.
2. Write 0 to 62 even number of bytes for Full-Speed or 0 to 510 even number of bytes for High-Speed at FIFOADR[0 1 0] (EP6).
3. Wait for READY (pin signal or CMDRDY interrupt).
4. Write IN Short Packet command with length at FIFOADR[0 1 1].
5. Write last odd byte value or 0 if even packet length at FIFOADR[0 1 1].
6. Write 0x0000 at FIFOADR[1 0 0].
7. Wait for INSHORTPKTACK interrupt. This signal controls further access to the endpoint buffer before the Bridge has committed the short packet. A short packet does not auto commit to the USB interface the way that full packets do, therefore it is required to wait until the Bridge acknowl-

edges that it has committed the short packet. Not waiting can result in next packet data getting written into the short packet buffer.

3.5.17 Bridge Firmware Images and Standby Sequences

There are three standby sequences. The first applies to Bridge images that sleep on boot. The second is host initiated sleep. The third sequence is for cable unplug handling.

The Bridge is available in five images.

- No Sleep on Boot with Wakeup Active Low and Handle all Standard Requests in the Bridge - *MoBLNSSR.iic*.
- No Sleep on Boot with Wakeup Active High - *MoBLNSpH.iic*.
- No Sleep on Boot with Wakeup Active Low - *MoBLNSpL.iic*.
- Sleep on Boot with Wakeup Active High - *MoBLWUHi.iic*.
- Sleep on Boot with Wakeup Active Low - *MoBLWULo.iic*.

3.5.17.1 Sleep on Boot

There are two Bridge images that perform some initialization and then enter suspend mode where the Bridge processor (FX2LP(18)) is in its low power state with its oscillator off. There are two versions of the sleep on boot images: one for Wakeup active low and the second for Wakeup active high. The appropriate image is required based on how the WAKEUP pin is wired in the design. There is a third image that does not sleep on boot and is set to the default Wakeup active low. This polarity can be changed by writing to the POLAR (0x04) register.

When the Sleep on Boot images are loaded, they go into suspend. The external master can strobe the WAKEUP pin when it is ready to initiate USB activities, such as detecting that the USB cable is plugged in. This is detected by monitoring the USB VBus line.

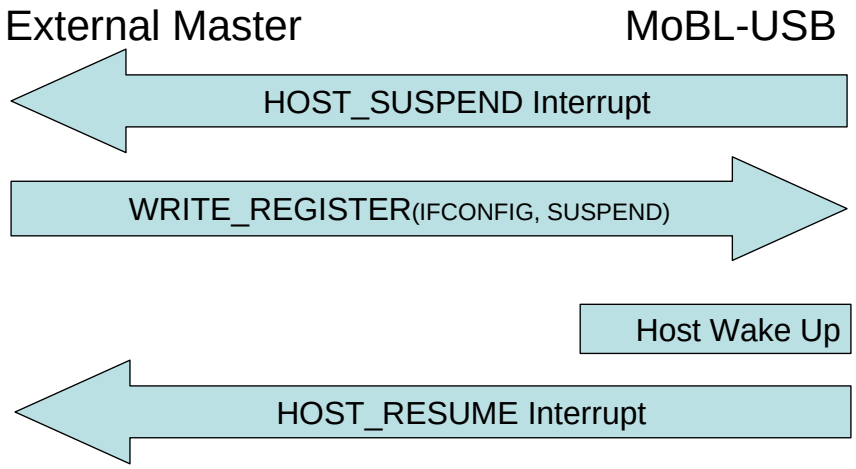
When WAKEUP is strobed, or the image is the No Sleep on Boot image, the Bridge completes its initialization sequence and enumeration is performed using one of the methods described in the End-point 0 chapter on page 37.

3.5.17.2 Host Initiated Suspend

When the host initiates a suspend, the Bridge firmware signals the external master with a HOST_SUSPEND extended interrupt. The external master examines the VBus line to see if VBus is present. If VBus is present then the external master writes to the Bridge IFCONFIG register (0x01) to set the STANDBY bit. When the host wakes up, USB activity begins and the Bridge automatically wakes up and signals the external master with a HOST_RESUME extended interrupt.

On some hosts, or with standby settings adjusted to turn off VBus power during suspend, the external master should follow the Cable Unplug Initiated Suspend.

Figure 3-12. Host Suspend/Resume

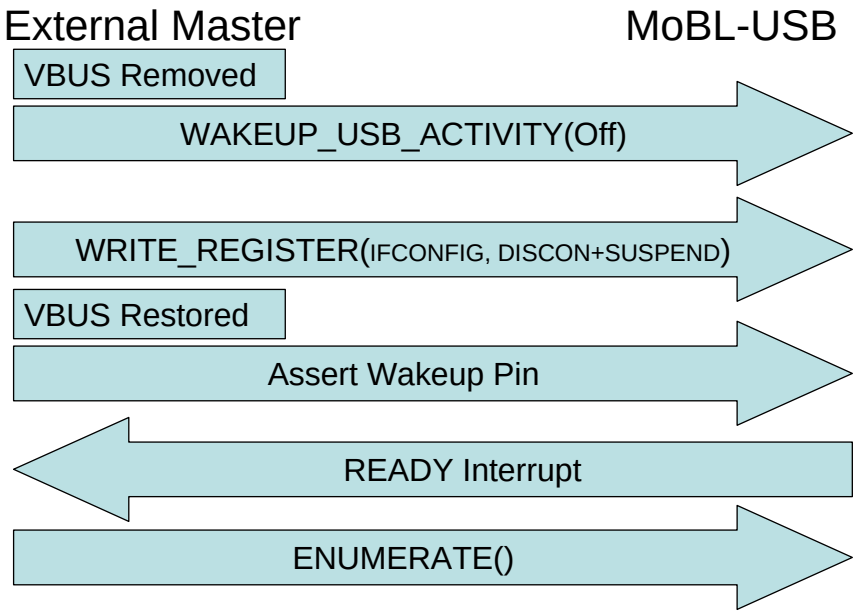


3.5.17.3 Cable Unplug Initiated Suspend

When the USB cable is unplugged from the host, the Bridge firmware signals the external master with a HOST_SUSPEND extended interrupt. The external master also senses that the VBus power is gone. For these conditions, the external master must send the WAKEUP_USB_ACTIVITY command to disable USB wakeups so that noise on the unplugged cable does not cause an unwanted wakeup condition. Then the external master writes to the IFCONFIG (0x01) register to set the DISCON bit and the STANDBY bit. The DISCON bit is set, so that the Bridge hardware does not drive the bus during this unplugged condition. The Bridge disconnects from the bus and then goes into the suspend mode

When the cable is plugged back in, the external master senses the VBus power and then strobes the wakeup pin on the Bridge chip. Then the external master waits for a READY interrupt from the Bridge and then writes the IFCONFIG register with the DISCON bit turned off. This causes the Bridge to reconnect to the USB bus. Normal USB resets occur followed by an enumeration sequence. For applications using default or manual enumeration, the Bridge gives an ENUMOK interrupt signal. For applications using the passthrough enumeration, the normal enumeration setup request sequence occurs. The external master must then send the WAKEUP_USB_ACTIVITY command to enable USB wakeups to prepare for any host initiated suspend.

Figure 3-13. USB Cable Unplug



3.5.18 Command List

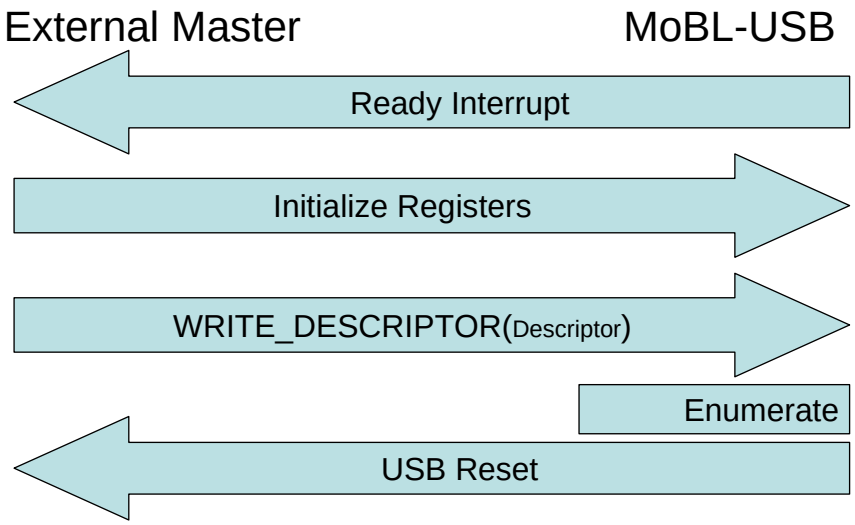
Command Name	In or Out	Length	Byte 0 ID	Byte 1 Data	Byte 2 Data	Notes
SETUP commands						
SETUP packet	OUT	10	1	Speed	Data[0..7]	Contains all SETUP data. Speed is the connection speed: 1=HS; 0=FS.
OUT phase	OUT	variable	2	Length	Data[0..n]	Contains up to 32 bytes of OUT data. Short packet terminates.
IN phase	IN	variable	3	Length	Data[0..n]	Contains up to 32 bytes of IN data. Short packet terminates.
SETUP complete	IN	1	6			Must be sent to complete every SETUP transaction.
Register Access						
Read register	IN	2	4	RegID		
Write register with response	IN	3	5	RegID	Data	Triggers response 'Write register response'.
Write register without response	IN	2	0x4x+ Reg ID	Data		Does not trigger response. Use command 5 if deterministic delay is needed.
Descriptors						
Descriptor Write	IN	variable	8	Length	Data[0..n]	Use multiple 32-byte packets, terminate with a zero length or short packet.
Data Transfer						
IN short packet	IN	3	0xc0 (EP6) + Length MSB	LSB	Last byte	Used to send all short packets. First, place the data in the IN buffer, then send this command. The last byte is sent here to allow packet_size-1 sized packets.
OUT Data Commit	IN	2	0x14	EP		Sent to start get EP OUT data; master checks EF flag is deasserted prior to reading each packet. EP=0: Endpoint 2
Set SETUP Pass Through On	IN	1	0x19			Enables Pass Through of all Standard Requests. Command must be sent after receive READY interrupt.
Enumerate	IN	1	0x1A			Master controlled enumeration. Connects to USB. This command is sent after sending Set SEUTP Pass Through On command when the external master is ready to handle enumeration standard requests.
EP1OUTEN	IN	3	0x1B	Enable	Mode	Enable Endpoint 1 OUT Enable: 1=Enable 0=Disable Mode: 1=Interrupt 0=Bulk
EP4OUTEN	IN	3	0x1C	Enable	Mode	Enable Endpoint 4 OUT Enable: 1=Enable 0=Disable Mode: 1=Interrupt 0=Bulk
EP8INEN	IN	3	0x1D	Enable	Mode	Enable Endpoint 8 IN Enable: 1=Enable 0=Disable Mode: 1=Interrupt 0=Bulk
CLEAR_DATA_TOGGLE_AND_STALL	IN	2	0x1E	EP		EP=0: Endpoint 1IN EP=1: Endpoint 1OUT EP=2: Endpoint 2 EP=4: Endpoint4 EP=6: Endpoint 6 EP=8: Endpoint8
WAKEUP_USB_ACTIVITY	IN	2	0x1F	Enable		Enable: 1=Enable 0=Disable, Default=Enable
OUT_SHORT_PKT_MSG_ACK	IN	2	0x20	EP		Sent in response to OUT short packet received message. EP=0: Endpoint 2
SET_STALL	IN	2	0x21	EP		EP=0: Endpoint 1IN EP=1: Endpoint 1OUT EP=2: Endpoint 2 EP=4: Endpoint4 EP=6: Endpoint 6 EP=8: Endpoint8
OUT_PKTCOUNT_NOTIFY	IN	3	0x22	EP	Enable	Sent to enable full OUT Packet Count for the specified EP, EP=1: Endpoint1OUT EP=2: Endpoint2OUT EP=4: Endpoint4OUT Enable: 1=Enable 0=Disable
OUT_PKTCOUNT	IN	4	0x23	EP	Count MSB, Count LSB	Sent at the beginning of each full packet OUT sequence.
BYPASS_OUT_SHORT_PKT_ACK	IN	3	0x24	EP	Enable	Sent to cause Bridge to bypass the wait for OUT_SHORT_PKT_ACK commands. EP=0 Endpoint2 Enable Bypass 1=Enable 0=Disable
USBRESET_ACK	IN	1	0x25			Sent in response to USBReset Interrupt to unblock EP1IN and EP8IN packets.

4. Enumeration

The Bridge firmware has three modes of enumeration. The first mode is automatic through EEPROM boot load, as described in Resets and Wakeup on page 15. The second method is a manual load of the descriptor or VID, PID, and DID from the external master as described below. In the third mode, the external master manages the entire enumeration process.

4.1 Manual Enumeration

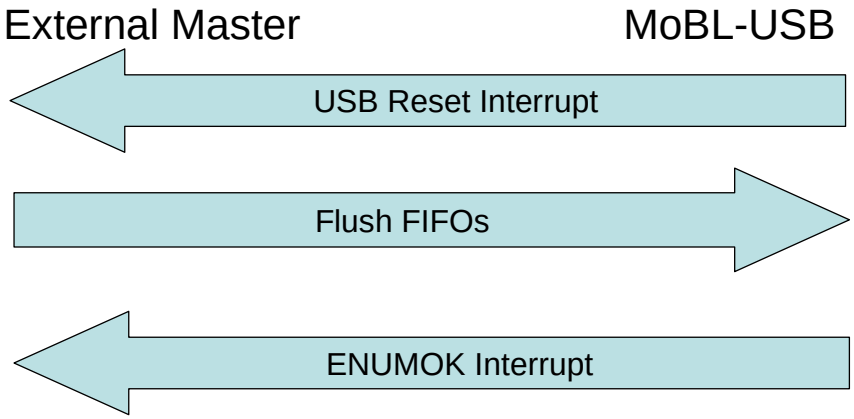
Figure 4-1. Manual Enumeration



The Bridge firmware has 500 bytes of descriptor RAM into which the external master may write its descriptor. The descriptor RAM is accessed through command 0x08. To load a descriptor, the external master does as follows:

1. Wait for READY (pin signal or CMDRDY interrupt).
2. Select FIFOADR [0 1 1] (Command FIFO).
3. Write 0x2008 = Write descriptor data (where 0x08 is the command and 0x20 is the length).
4. Write the first 0x20 bytes of the descriptor to the Command FIFO.
5. Write 0x0000 to FIFOADR [1 0 0] to send the command packet to the Bridge firmware.
6. Repeat steps 1-5 until all of the data has been written to the descriptor RAM. If the descriptor ends with a packet of less than 32 bytes, set the length field to the remainder length and write the remainder bytes. If the descriptor ends on a 32-byte boundary, write one more descriptor with a zero length by following steps one through five.

Figure 4-2. USB Reset (Passthrough Mode Off)



After the entire descriptor has been transferred, the Bridge firmware floats the pull up resistor connected to D+, and parses through the descriptor to locate the individual descriptors. After the Bridge firmware has parsed the entire descriptor, the Bridge firmware connects the pull up resistor and enumerates automatically. When enumeration is complete, the Bridge firmware notifies the external master with an ENUMOK interrupt.

The format and order of the descriptor must be as follows (see the Default Descriptor chapter on page 55 for an example):

- Device
- Device qualifier
- High-speed configuration, high-speed interface, high-speed endpoints
- Full-speed configuration, full-speed interface, full-speed endpoints
- String

4.2 Default Enumeration

The external master may simply load a VID, PID, and DID and use the default descriptor built into the Bridge firmware. To use the default descriptor, the descriptor length described previously must equal six. After the external master has written the length, the VID, PID, and DID must be written LSB, then MSB. For example, if the VID, PID, and DID are 0x04B4, 0x1002, and 0x0001 respectively, then the external master does the following:

1. Wait for READY (pin signal or CMDRDY interrupt).
2. Select FIFOADR [0 1 1] (Command FIFO).
3. Write 0x0608 = Write descriptor data (where 0x08 is the command and 0x06 is the special use defaults length).
4. Write the VID, PID, DID (0xB4, 0x04, 0x02, 0x10, 0x01, 0x00) to the Command FIFO.
5. Write 0x0000 to FIFOADR [1 0 0] to send the command.

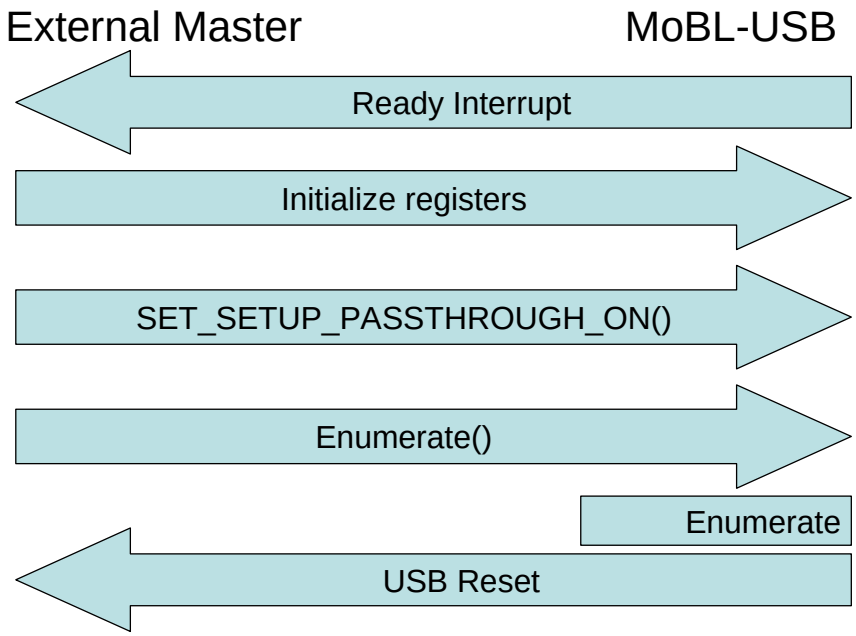
The default descriptor is listed in Default Descriptor chapter on page 55. The default descriptor can be used as a starting point for a custom descriptor.

4.3 External Master Passthrough Enumeration

The external master can control the enumeration process by setting all standard SETUP requests to pass through to the external master by sending the Set SETUP Passthrough On command. When

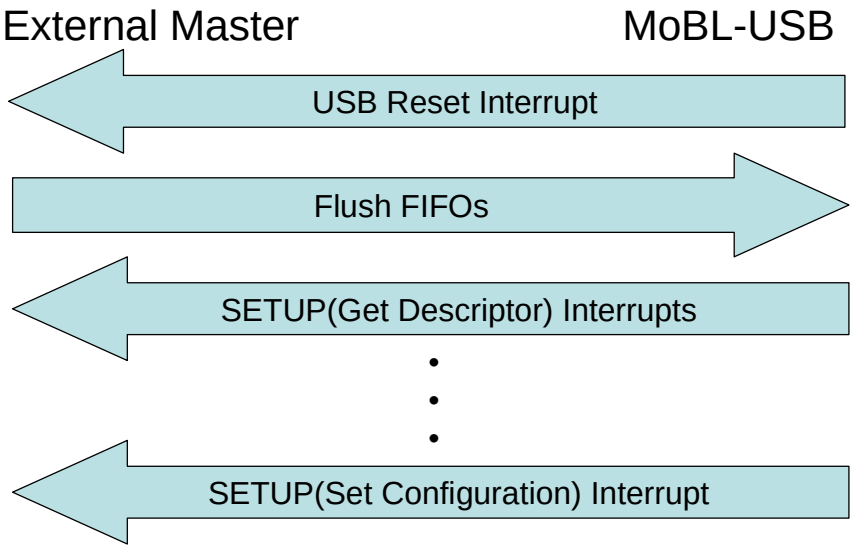
this is done, the external master sends the Enumerate command. The Bridge firmware connects to the USB and then passes through all enumeration SETUP requests to the external master. The external master is responsible for supplying all descriptor information as part of a SETUP IN sequence. Enumeration is typically considered complete when the Set Configuration standard request is received. When the passthrough mode is set, all SETUP requests are sent to the external master with the exception of the special Test Set Feature requests which the Bridge automatically responds to with the handshake.

Figure 4-3. Passthrough Enumeration



If the passthrough mode is not set, then any SETUP request not handled by the Bridge is sent to the external master. The external master must Stall EP0 for any unrecognized commands. Again, the Bridge automatically responds to the special Test Set Feature request.

Figure 4-4. USB Reset (Passthrough Mode On)



4.4 Interrupt System

4.4.1 Architecture

The Bridge firmware provides an output signal that indicates to the external master that the Bridge firmware has an interrupt condition, or that the data from a register read request is available. The Bridge firmware has six standard interrupt sources: SETUP, EP0BUF, USBRESET, CMDSTATRDY, ENUMOK, and READY. Each standard interrupt can be enabled or disabled by setting or clearing the corresponding bit in the INTENABLE register. An extended set of interrupts provides control for using endpoints 4, and 8, and endpoint 1 OUT. The upper most bit of the high byte of the interrupt status signals whether the interrupt is a standard interrupt or one of the extended interrupts. The extended interrupts, INNF and OUTNE, are enabled or disabled by setting or clearing the corresponding bit in the extended INTENABLE1 register.

When an interrupt occurs, the INT# pin is asserted and the corresponding bit or bits are set in the Interrupt Status upper and lower bytes. The external master reads the Interrupt Status bytes by strobing SLRD/SLOE. This presents the Interrupt Status bytes on the data bus (FD[15:0]). Reading the Interrupt Status bytes automatically clears the interrupt. Only one interrupt request occurs at a time; the Bridge firmware buffers multiple pending interrupts.

4.4.2 INTENABLE Register Bit Definition

Bit 7: SETUP

If this interrupt is enabled, and the Bridge firmware receives a setup packet from the USB host, the Bridge firmware asserts the INT# pin and sets bit 7 in the lower byte of the Interrupt Status. This interrupt only occurs if the setup request is not one that the Bridge firmware automatically handles. For complete details on how to handle the SETUP interrupt, refer to the Endpoint 0 chapter on page 37.

Bit 6: EP0BUF

If this interrupt is enabled, and the Endpoint 0 buffer becomes available to the external master for read or write operations, the Bridge firmware asserts the INT# pin and sets bit 6 in the lower byte of the Interrupt Status. This interrupt is used for handling the data phase of a setup request. For complete details on how to handle the EP0BUF interrupt, refer to Endpoint 0 chapter on page 37.

Bit 5: UNUSED

This was formerly the FLAGS interrupt. This interrupt has been replaced by the INNF (IN not full) and OUTNE (OUT not empty) extended interrupts as they provide exact signalling.

Bit 4: USB RESET

If this interrupt is enabled, and the Bridge firmware detects a USB bus reset, the Bridge firmware asserts the INT# pin and sets bit 4 in the lower byte of the Interrupt Status. The external master uses the command interface to reset the STALL and data toggle bits for all endpoints.

Bit 3: CMDSTATRDY

The Bridge firmware provides a Command Status Ready interrupt by asserting the INT# pin and setting bit 3 in the lower byte of the Interrupt Status. This interrupt signals data availability on the Command OUT interface. This interrupt is always enabled for normal operations. The corresponding bit in the INTENABLE register enables the CMDRDY extended interrupt.

Bit 2: ENUMOK

If this interrupt is enabled and the Bridge firmware receives a SET_CONFIGURATION request from the USB host, the Bridge firmware asserts the INT# pin and sets bit 2 in the lower byte of the Interrupt Status. This event signals the completion of the Bridge firmware enumeration process.

Bit 1: UNUSED

This was formerly the BUSACTIVITY interrupt. This interrupt has been replaced by the HOST_SUSPEND and HOST_RESUME extended interrupts.

Bit 0: READY

If this interrupt is enabled, bit 0 in the lower byte of the Interrupt Status is set when the Bridge firmware has powered up and performed a self-test. The external master must always wait for this interrupt before trying to read or write to the Bridge firmware, unless an external EEPROM with a valid descriptor is present. If an external EEPROM with a valid descriptor is present, the ENUMOK interrupt occurs instead of the READY interrupt after power up. A READY interrupt also occurs if the Bridge firmware is awakened from a low-power mode via the WAKEUP pin.

4.4.3 Standard Interrupt Bit Positions and Extended Interrupts

Upper Interrupt Status Byte:
Extended Interrupts

Bit #	7	6	5	4	3	2	1	0
Bit Name	INTMODE	RDREGRSP	INTCODE3 / RDREGID5	INTCODE2 / RDREGID4	INTCODE1 / RDREGID3	INTCODE0 / RDREGID2	PL9 / RDREGID1	PL8 / RDREGID0

Lower Interrupt Status Byte:
Standard and Extended Interrupts

Bit #	7	6	5	4	3	2	1	0
Bit Name	SETUP / PL7 / RDREGVAL7	EP0BUF / PL6 / RDREGVAL6	FLAGS / PL5 / RDREGVAL5 / WRREGID5	USB RESET / PL4 / RDREGVAL4 / WRREGID4	CMDSTATRDY/ PL3 / EP3 / RDREGVAL3 / WRREGID3	ENUMOK / PL2 / EP2 / RDREGVAL2 / WRREGID2	UNUSED / PL1 / EP1 / RDREGVAL1 / WRREGID1	READY / PL0 / EP0 / RDREGVAL0 / WRREGID0

4.4.3.1 INTMODE - Interrupt Mode

- 0: Use Standard Interrupt definitions for each bit in the Lower Interrupt Status byte. The Upper Interrupt Status byte is not used.
- 1: Use Extended Interrupt definitions for the RDREGRSP or the INTCODE[3:0] in the Upper Interrupt Status byte. Use PL[9:0] (Packet Length), EP[3:0] (Endpoint ID), RDREGID[5:0] (Read Register Response Register ID), RDREGVAL[7:0] (Read Register Response Register Value), and WRREGID[5:0] (Write Register with Response Register ID) in the upper and lower bytes.

4.4.3.2 RDREGRSP - Read Register Response

When INTMODE is set and this bit is SET, then the extended interrupt is the response to the Read Register command. When this bit is set then RDREGID[5:0] contains the ID of the read register and RDREGVAL[7:0] contains the returned read register value. This interrupt is always enabled.

4.4.3.3 INTCODE[3:0] - Interrupt Code

- 0000: EP1OUTPKTAVAIL - Indicates EP1OUT data packet is available at FIFO address [001] and the packet length is in PL[7:0]. This interrupt is always enabled.
- 0001: EP2OUTSHORTPKTAVAIL - Indicates EP2OUT short data packet is available at FIFO address [001] and the short packet length is in PL[9:0]. This interrupt is always enabled.

- 0010: UNUSED
- 0011: EP4OUTPKTAVAIL - Indicates EP4OUT data packet is available at FIFO address [001] and the packet length is in PL[9:0]. This interrupt is always enabled.
- 0100: EP8INRDY - Indicates that EP8 is ready for an IN data packet.
- 0101: CMDRDY - Indicates that the Bridge firmware is ready for a new command. When the CMDRDY is enabled in the INTENABLE register, then a Command Ready interrupt is generated when the command interface is ready to receive a new command. If this interrupt is not enabled, the READY pin is used to determine when the Bridge firmware is ready for a new command.
- 0110: OUTNE - Indicates EP2 OUT is not empty. EP[3:0]=2 for EP2OUTNE. This interrupt is enabled in the INTENABLE1 register. External master applications must use a counting flag that is initialized to zero when the endpoint is initialized or flushed or when a USB RESET occurs. Whenever a packet is read from the Bridge, decrement the count by one. Whenever this interrupt is received, increment the count by one.
- 0111: INNF - Indicates EP6 IN is not full. EP[3:0]=6 for EP6INNF. This interrupt is enabled in the INTENABLE1 register. External master applications must use a counting flag that is initialized to 2 when the endpoint is initialized or flushed, or when a USB RESET occurs. Whenever a packet is written to the Bridge, decrement the count by one. Whenever this interrupt is received, increment the count by one.
- 1000: WRREGRSP - Indicates response to Write Register with Response command. WRREGID[5:0] contains the ID of the written register. This interrupt is always enabled.
- 1001: EP1INBUFAVAIL - Indicates that the EP1IN buffer is ready for an IN data packet. This interrupt is always enabled.
- 1010: INSHORTPKTACK - Indicates short data packet has been committed. EP[3:0]=6 for EP6IN. This interrupt is always enabled.
- 1011: HOST_SUSPEND - Indicates that host has suspended bus activity. The external master can put the Bridge in standby in response to this interrupt. This interrupt is always enabled.
- 1100: HOST_RESUME - Indicates that the host has resumed bus activity. The Bridge firmware generates this interrupt in response to the host generated wakeup event. This interrupt is always enabled.
- 1101: OUT_PKTCOUNT_NOTIFY - Indicates that the Packet Count has been reached for the specified endpoint. EP[3:0]=1 for EP1OUT. EP[3:0]=2 for EP2OUT. EP[3:0]=4 for EP4OUT. EP[3:0]=6 for EP6OUT. Packet Counts are enabled for an endpoint by sending the OUT_PKTCOUNT_NOTIFY command. This command is sent once to enable counts and once to disable counts. An OUT_PKTCOUNT_COUNT command is sent to start the counter for an endpoint. This command is sent once for each sequence of out transfers. The count is incremented once per full OUT packet received on the endpoint and cleared when a short OUT packet is received on the endpoint.
- 1110: HS - Indicates that a high-speed connection has been made to the host PC.

4.4.3.4 PL[9:0] - Packet Length for OUT Packets

These bits indicate the size of an OUT packet. They are used with the EP2OUTSHORTPKTAVAIL, the EP1OUTPKTAVAIL, and the EP4OUTPKTAVAIL extended interrupts.

4.4.3.5 EP[3:0] - Endpoint Number for INNF, OUTNE, and INSHORTPKTACK

- 0010: Endpoint 2
- 0110: Endpoint 6

5. Endpoint 0

5.1 SETUP Commands

The Bridge firmware automatically responds to USB chapter 9 SETUP requests without any external master intervention if passthrough mode is not enabled. If the Bridge firmware receives a request to which it cannot respond automatically, the Bridge firmware notifies the external master. The external master then has the choice of responding to the request or stalling it.

After the Bridge firmware receives a setup packet to which it cannot respond automatically, the Bridge firmware sends the packet to the external master via the Command FIFO with an ID byte of 0x01.

The external master can stall the request at this or any other time. To stall a request, the external master initiates a write request for the SETUP register, 0x32, and writes any non-zero value to the register.

If this setup request has OUT data associated with it, the Bridge firmware sends the OUT data to the command FIFO with an ID byte of 0x02. If a zero length OUT packet is received, the Bridge firmware forwards a zero length command FIFO transaction to the external master. After the OUT data phase of the SETUP, the external master sends the IN data through the command FIFO with an ID byte of 0x03.

For an IN setup transaction, the external master can write up to 32 bytes at a time for the data phase. The steps to write a packet are as follows:

1. Wait for READY (pin signal or CMDRDY interrupt).
2. Select FIFOADR [0 1 1].
3. Write 0x03.
4. Write the length of this portion of the data (up to 32 bytes).
5. Write the data.
6. Select FIFOADR [1 0 0].
7. Write 0x0000 to FIFOADR [1 0 0] to terminate the 32 byte buffer.
8. If the SETUP IN data length is greater than 64 bytes, wait for the EP0BUF interrupt signaling buffer availability.
9. Repeat steps 1 through 8 until all of the data has been written. If the data ends on an even 32 byte boundary it must be followed by a 0 length packet to terminate the IN phase of the transaction.
10. Wait for READY (pin signal or CMDRDY interrupt).
11. Select FIFOADR [0 1 1].
12. Write 0x06 to complete the SETUP transaction.
13. Select FIFOADR[1 0 0].
14. Write 0x0000 to FIFOADR[1 0 0] to commit the Setup Done command.

For an OUT setup transaction, the external master can read each packet received from the USB host during the data phase. The steps to read a packet are as follows:

1. Select FIFOADR [0 0 1] and wait for CMDSTATRDY interrupt to indicate not EMPTY.
2. Read first byte of the data from the FIFO. An ID byte of 0x02 indicates OUT data.
3. Read the second byte of data from the FIFO. This byte indicates the length of THIS PORTION of the data.
4. Read the remainder of the OUT data from the FIFO.
5. If the SETUP length is greater than 64, then wait for the EP0BUF interrupt to indicate the next up to 64 bytes are available. Then repeat steps 2 though 4 for the next block of data. Repeat this step until all SETUP OUT data is received.
6. Wait for READY (pin signal or CMDRDY interrupt).
7. Select FIFOADR [0 1 1].
8. Write 0x06 to complete the SETUP transaction.
9. Select FIFOADR[1 0 0].
10. Write 0x0000 to FIFOADR[1 0 0] to commit the Setup Done command.

The Bridge firmware automatically responds to all USB standard requests covered in chapter 9 of the USB 2.0 specification except the Clear Feature Endpoint requests. The Bridge firmware clears the Stall bit for the endpoint specified in the request and then sends the request to the external master via the SETUP interrupt. The external master must send the SETUP Complete command to the Bridge to allow it to terminate the SETUP request.

6. Register Summary

6.1 Register Summary

Table 6-1. Bridge Firmware Register Summary

Hex	Size	Name	Description	D7	D6	D5	D4	D3	D2	D1	D0	Default	Access
		General Configuration											
01	1	IFCONFIG	Interface Configuration	IFCLKSRC	3048MHZ	IFCLKOE	IFCLKPOL	ASYNC	STANDBY	FLAGD/CS#	DISCON	11001001	bbbbbbbb
02	1	FLAGSAB	FIFO FLAGA and FLAGB Assignments	FLAGB3	FLAGB2	FLAGB1	FLAGB0	FLAGA3	FLAGA2	FLAGA1	FLAGA0	00000000	bbbbbbbb
03	1	FLAGSCD	FIFO FLAGC and FLAGD Assignments	FLAGD3	FLAGD2	FLAGD1	FLAGD0	FLAGC3	FLAGC2	FLAGC1	FLAGC0	00000000	bbbbbbbb
04	1	POLAR	FIFO polarities	WUPOL	0	PKTEND	SLOE	SLRD	SLWR	EF	FF	00000000	bbbrrrbb
05	1	REVID	Chip Revision	Major	Major	Major	Major	minor	minor	minor	minor	xxxxxxxx	rrrrrrrr
		Endpoint Configuration*											
06	1	EP2CFG	Endpoint 2 Configuration	VALID	dir	TYPE1	TYPE0	SIZE	STALL	BUF1	BUF0	10100010	bbbbbbbb
07	1	Reserved											
08	1	EP6CFG	Endpoint 6 Configuration	VALID	dir	TYPE1	TYPE0	SIZE	STALL	BUF1	BUF0	11100010	bbbbbbbb
09	1	EP1INCFG	Endpoint 1 IN Configuration	VALID	0	TYPE1	TYPE0	0	0	0	0	10100000	bbbbbbbb
0A	1	EP2PKTLENH	Endpoint 2 Packet Length H	INFM1	OEP1	ZEROLEN	WORD-WIDE	0	PL10	PL9	PL8	00110010	bbbbbbbb
0B	1	EP2PKTLENL	Endpoint 2 Packet Length L (IN only)	PL7	PL6	PL5	PL4	PL3	PL2	PL1	PL0	00000000	bbbbbbbb
0C	1	Reserved											
0D	1	Reserved											
0E	1	EP6PKTLENH	Endpoint 6 Packet Length H	INFM1	OEP1	ZEROLEN	WORD-WIDE	0	PL10	PL9	PL8	00110010	bbbbbbbb
0F	1	EP6PKTLENL	Endpoint 6 Packet Length L (IN only)	PL7	PL6	PL5	PL4	PL3	PL2	PL1	PL0	00000000	bbbbbbbb
10	1	Reserved											
11	1	Reserved											
12	1	EP2PFH	EP2 Programmable Flag H	DECIS	PKTSTAT	IN: PKTS[2] OUT:PFC12	IN: PKTS[1] OUT:PFC11	IN: PKTS[0] OUT:PFC10	0	PFC9	PFC8	10001000	bbbbbbbb
13	1	EP2PFL	EP2 Programmable Flag L	PFC7	PFC6	PFC5	PFC4	PFC3	PFC2	PFC1	PFC0	00000000	bbbbbbbb
14	1	Reserved											
15	1	Reserved											
16	1	EP6PFH	EP6 Programmable Flag H	DECIS	PKTSTAT	IN: PKTS[2] OUT:PFC12	IN: PKTS[1] OUT:PFC11	IN: PKTS[0] OUT:PFC10	0	PFC9	PFC8	00001000	bbbbbbbb
17	1	EP6PFL	EP6 Programmable Flag L	PFC7	PFC6	PFC5	PFC4	PFC3	PFC2	PFC1	PFC0	00000000	bbbbbbbb
18	1	Reserved											
19	1	Reserved											
1A	1	Reserved											
1B	1	Reserved											
1C	1	Reserved											
1D	1	Reserved											
		FLAGS											
1E	1	EP2FLAGS	Endpoint 2 FIFO Flags	0	0	1	0	0	EP2PF	EP2EF	EP2FF	00100010	rrrrrrrr
1F	1	EP6FLAGS	Endpoint 6 FIFO Flags	0	0	1	0	0	EP6PF	EP6EF	EP6FF	01100110	rrrrrrrr
		INPKTEND/FLUSH											
20	1	INPKTEND/ FLUSH	Force Packet End / Flush FIFOs	0	FIFO6	0	FIFO2	EP3	EP2	EP1	EP0	00000000	wwwwww-ww
		USB Configuration											
2A	1	USBFRAMEH	USB Frame count H	0	0	0	0	0	FC10	FC9	FC8	xxxxxxxx	rrrrrrrr
2B	1	USBFRAMEL	USB Frame count L	FC7	FC6	FC5	FC4	FC3	FC2	FC1	FC0	xxxxxxxx	rrrrrrrr
2C	1	MICROFRAME	Microframe count, 0-7	0	0	0	0	0	MF2	MF1	MF0	xxxxxxxx	rrrrrrrr
2D	1	FNADDR	USB Function address	HSGRANT	FA6	FA5	FA4	FA3	FA2	FA1	FA0	00000000	rrrrrrrr

Table 6-1. Bridge Firmware Register Summary *(continued)*

Hex	Size	Name	Description	D7	D6	D5	D4	D3	D2	D1	D0	Default	Access
		Interrupts											
2E	1	INTENABLE	Interrupt Enable	SETUP	EP0BUF	UNUSED	USBRESET	CMDRDY	ENUMOK	UNUSED	READY	11011101	bbbbbbbb
2F	1	INTENABLE1	Extended Interrupt Enable	reserved	reserved	reserved	HS_NOTIFY	EP6INNF	reserved	reserved	EP2OUTNE	00000000	bbbbbbbb
30	1	Reserved											
		Endpoint 0											
31	1	Reserved											
32	1	EP0STALL	Any non-zero write STALLS EP0	d7	d6	d5	d4	d3	d2	d1	d0	0000000	wwwwww-ww
33	1	Reserved											
		Un-Indexed Register Control											
3A	1		Un-Indexed Register Low Byte pointer	a7	a6	a5	a4	a3	a2	a1	a0		
3B	1		Un-Indexed Register High Byte pointer	a7	a6	a5	a4	a3	a2	a1	a0		
3C	1		Un-Indexed Register Data	d7	d6	d5	d4	d3	d2	d1	d0		
Address		**Un-Indexed Registers in XDATA Space**											
0xE600		CPUCS	CPU Control and Status	reserved	reserved	reserved	CLKSPD1	CLKSPD0	CLKINV	CLKOE	reserved	00010010	rrrbbbbr

*__Note__ that the Bridge firmware was not designed to support dynamic modification of these endpoint configuration registers. If your applications need the ability to change endpoint configurations after the device has already enumerated with a specific configuration, expect some delay in being able to access the FIFOs after changing the configuration. This delay time varies for different registers so Cypress recommends using the “write register with response” command when modifying these registers.

6.2 IFCONFIG Register 0x01

IFCONFIG **0x01**

Bit #	7	6	5	4	3	2	1	0
Bit Name	IFCLKSRC	3048MHZ	IFCLKOE	IFCLKPOL	ASYNC	STANDBY	FLAGD/CS#	DISCON
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	1	1	0	0	1	0	0	1

Bit 7: IFCLKSRC

This bit selects the clock source for the FIFOs. If IFCLKSRC = 0, the external clock on the IFCLK pin is selected. If IFCLKSRC = 1 (default), an internal 30 or 48 MHz clock is used.

Bit 6: 3048MHZ

This bit selects the internal FIFO clock frequency. If 3048MHZ = 0, the internal clock frequency is 30 MHz. If 3048MHZ = 1 (default), the internal clock frequency is 48 MHz.

Bit 5: IFCLKOE

This bit selects if the IFCLK pin is driven. If IFCLKOE = 0 (default), the IFCLK pin is floated. If IFCLKOE = 1, the IFCLK pin is driven.

Bit 4: IFCLKPOL

This bit controls the polarity of the IFCLK signal.

- When IFCLKPOL=0, the clock has the polarity shown in all the timing diagrams in this guide (the rising edge is the activating edge).
- When IFCLKPOL=1, the clock is inverted (in some cases may help with satisfying data setup times).

Bit 3: ASYNC

This bit controls whether the FIFO interface is synchronous or asynchronous. When ASYNC = 0, the FIFOs operate synchronously. In synchronous mode, a clock is supplied either internally or externally on the IFCLK pin, and the FIFO control signals function as read and write enable signals for the clock signal.

When ASYNC = 1 (default), the FIFOs operate asynchronously. No clock signal input to IFCLK is required, and the FIFO control signals function directly as read and write strobes.

Bit 2: STANDBY

This bit instructs the Bridge firmware to enter a low-power mode. When STANDBY=1, the Bridge firmware enters a low-power mode by turning off its oscillator. The external master must write this bit after it receives a BUSACTIVITY interrupt (indicating that the host has signaled a USB suspend condition). If the Bridge is disconnected from the USB bus, the external master can write this bit at any time to save power. Once suspended, the Bridge firmware is awakened either by resumption of USB bus activity or by assertion of its WAKEUP pin.

Bit 1: FLAGD/CS#

This bit controls the function of the FLAGD/CS# pin. When FLAGD/CS# = 0 (default), the pin operates as a slave chip select. If FLAGD/CS# = 1, the pin operates as FLAGD.

Bit 0: DISCON

This bit controls whether the internal pull up resistor connected to D+ is pulled high or floating. When DISCON = 1 (default), the pull up resistor is floating simulating a USB unplug. When DISCON=0, the pull up resistor is pulled high signaling a USB connection.

6.3 FLAGSAB/FLAGSCD Registers 0x02/0x03

The Bridge firmware has four FIFO flags output pins: FLAGA, FLAGB, FLAGC, and FLAGD.

FLAGSAB **0x02**

Bit #	7	6	5	4	3	2	1	0
Bit Name	FLAGB3	FLAGB2	FLAGB1	FLAGB0	FLAGA3	FLAGA2	FLAGA1	FLAGA0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

FLAGSCD **0x03**

Bit #	7	6	5	4	3	2	1	0
Bit Name	FLAGD3	FLAGD2	FLAGD1	FLAGD0	FLAGC3	FLAGC2	FLAGC1	FLAGC0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

These flags can be programmed to represent various FIFO flags using four select bits for each FIFO. The 4 bit coding for all four flags is the same, as shown in the following table.

Table 6-2. FIFO Flag 4 Bit Coding

FLAGx3	FLAGx2	FLAGx1	FLAGx0	Pin Function
0	0	0	0	FLAGA = PF, FLAGB = FF, FLAGC = EF, FLAGD = CS# (actual FIFO is selected by FIFOADR[2:0] pins)
0	0	0	1	Reserved
0	0	1	0	Reserved
0	0	1	1	Reserved
0	1	0	0	EP2 PF
0	1	0	1	Reserved
0	1	1	0	EP6 PF
0	1	1	1	Reserved
1	0	0	0	EP2 EF
1	0	0	1	Reserved
1	0	1	0	EP6 EF
1	0	1	1	Reserved
1	1	0	0	EP2 FF
1	1	0	1	Reserved
1	1	1	0	EP6 FF
1	1	1	1	Reserved

For the default (0000) selection, the four FIFO flags are fixed-function as shown in the first table entry; the input pins FIFOADR[2:0] select to which of the four FIFOs the flags correspond. These pins are decoded as shown in Table 3-1 on page 17.

The other (non-zero) values of FLAGx[3:0] allow the designer to independently configure the four flag outputs FLAGA-FLAGD to correspond to any flag-Programmable, Full, or Empty-from any of the four endpoint FIFOs. This allows each flag to be assigned to any of the four FIFOs, including those not currently selected by the FIFOADR [2:0] pins. For example, the external master could be filling the EP2IN FIFO with data while also checking the empty flag for the EP6OUT FIFO.

6.4 POLAR Register 0x04

This register controls the polarities of FIFO pin signals and the WAKEUP pin.

POLAR **0x04**

Bit #	7	6	5	4	3	2	1	0
Bit Name	WUPOL	0	PKTEND	SLOE	SLRD	SLWR	EF	FF
Read/Write	R/W	R/W	R/W	R	R	R	R/W	R/W
Default	0	0	0	0	0	0	0	0

Bit 7: WUPOL

This flag sets the polarity of the WAKEUP pin. If WUPOL = 0 (default), the polarity is active LOW. If WUPOL=1, the polarity is active HIGH.

Bit 5: PKTEND

This flag selects the polarity of the PKTEND pin. If PKTEND = 0 (default), the polarity is active LOW. If PKTEND = 1, the polarity is active HIGH.

Bit 4: SLOE

This flag selects the polarity of the SLOE pin. If SLOE = 0 (default), the polarity is active LOW. If SLOE = 1, the polarity is active HIGH. This bit can only be changed by using the EEPROM configuration load.

Bit 3: SLRD

This flag selects the polarity of the SLRD pin. If SLRD = 0 (default), the polarity is active LOW. If SLRD = 1, the polarity is active HIGH. This bit can only be changed by using the EEPROM configuration load.

Bit 2: SLWR

This flag selects the polarity of the SLWR pin. If SLWR = 0 (default), the polarity is active LOW. If SLWR = 1, the polarity is active HIGH. This bit can only be changed by using the EEPROM configuration load.

Bit 1: EF

This flag selects the polarity of the EF pin (FLAGA/B/C/D). If EF = 0 (default), the EF pin is pulled low when the FIFO is empty. If EF = 1, the EF pin is pulled HIGH when the FIFO is empty.

Bit 0: FF

This flag selects the polarity of the FF pin (FLAGA/B/C/D). If FF = 0 (default), the FF pin is pulled low when the FIFO is full. If FF = 1, the FF pin is pulled HIGH when the FIFO is full.

6.5 REVID Register 0x05

These register bits define the firmware revision.

REVID **0x05**

Bit #	7	6	5	4	3	2	1	0
Bit Name	Major	Major	Major	Major	Minor	Minor	Minor	Minor
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	X	X	X	X	X	X	X	X

The upper nibble is the major revision. The lower nibble is the minor revision. For example: if REVID = 0x11, then the firmware revision is 1.1.

6.6 EPxCFG Register 0x06–0x09

These registers configure the large, data-handling Bridge endpoints, EP2 and EP6 and the small interrupt endpoint, EP1IN. It is recommended to use the default settings for EP2 as an OUT endpoint and EP6 as an IN endpoint (see Table 6-1 on page 39). Only the TYPE bits should be changed as required for Interrupt endpoints instead of Bulk endpoints. EP1IN must be programmed as an Interrupt endpoint.

EPxCFG **0x06, 0x08**

Bit #	7	6	5	4	3	2	1	0
Bit Name	VALID	DIR	TYPE1	TYPE0	SIZE	STALL	BUF1	BUF0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	1	0	1	0	0	0	1	0

EP1INCFG **0x09**

Bit #	7	6	5	4	3	2	1	0
Bit Name	VALID	0	TYPE1	TYPE0	0	STALL	0	0
Read/Write	R/W	R	R/W	R/W	R	R/W	R	R
Default	1	0	1	0	0	0	1	0

Bit 7: VALID

The external master sets VALID = 1 to activate an endpoint, and VALID = 0 to deactivate it. All Bridge endpoints default to valid. An endpoint whose VALID bit is 0 does not respond to any USB traffic.

Bit 6: DIR

0 = OUT, 1 = IN. Default for EP2 is DIR = 0, OUT, and for EP6 is DIR = 1, IN. This bit is not used in the EP1 config registers.

Bit [5,4]: TYPE1, TYPE0

These bits define the endpoint type, as shown in Table 6-3. The TYPE bits apply to all of the endpoint configuration registers. All Bridge endpoints except EP0 default to BULK.

Table 6-3. Endpoint Type

TYPE1	TYPE0	Endpoint Type
0	0	Invalid
0	1	Invalid
1	0	Bulk (Default)
1	1	Interrupt

Bit 3: SIZE

Must be set to 0 = 512 bytes (64 bytes at full-speed).

Bit 2: STALL

Each bulk endpoint (IN or OUT) has a STALL bit (bit 2). If the external master sets this bit, any requests to the endpoint return a STALL handshake rather than ACK or NAK. The Get Status-Endpoint Request returns the STALL state for the endpoint indicated in byte 4 of the request. Note that bit 7 of the endpoint number EP (byte 4) specifies direction.

Bit [1,0]: BUF1, BUF0

These bits must be set to 10 to select double buffering. Any other setting is invalid.

6.7 EPxPKTLENH/L Registers 0x0A–0x0F

The external master can use these registers to set smaller packet sizes than the physical buffer size (512 bytes at high-speed, 64 bytes at full-speed). In addition, the EPxPKTLENH register has four other endpoint configuration bits.

EP2PKTLENL, EP6PKTLENL — **0x0B, 0x0F**

Bit #	7	6	5	4	3	2	1	0
Bit Name	PL7	PL6	PL5	PL4	PL3	PL2	PL1	PL0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

EP2PKTLENH, EP6PKTLENH — **0x0A, 0x0E**

Bit #	7	6	5	4	3	2	1	0
Bit Name	INFM1	OEP1	ZEROLEN	WORDWIDE	0	0	PL9	PL8
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	1	1	0	0	1	0

Bit 7: INFM1 EPxPKTLENH.7

When the external master sets INFM = 1 in an endpoint configuration register, the FIFO flags for that endpoint become valid one sample earlier than when the full condition occurs. These bits take effect only when the FIFOs are operating synchronously according to an internally or externally supplied clock. Having the FIFO flag indications one sample early simplifies some synchronous interfaces. This applies only to IN endpoints. Default is INFM1 = 0.

Bit 6: OEP1 EPxPKTLENH.6

When the external master sets an OEP = 1 in an endpoint configuration register, the FIFO flags for that endpoint become valid one sample earlier than when the empty condition occurs. These bits take effect only when the FIFOs are operating synchronously according to an internally or externally supplied clock. Having the FIFO flag indications one sample early simplifies some synchronous interfaces. This applies only to OUT endpoints. Default is OEP1 = 0.

Bit 5: ZEROLEN EPxPKTLENH.5

When ZEROLEN = 1 (default), a zero length packet is sent when the PKTEND pin is asserted and there are no bytes in the current packet. If ZEROLEN = 0, then a zero length packet is not sent under these conditions.

Bit 4: WORDWIDE EPxPKTLENH.4

This bit controls whether the data interface is 8 or 16 bits wide. If WORDWIDE = 0, the data interface is eight bits wide, and FD[15:8] have no function. If WORDWIDE = 1 (default), the data interface is 16 bits wide.

Bit [2:0]: PL[X:0] Packet Length Bits

The Packet Length Bits apply to IN endpoint EP6 only. The default packet size is 512 bytes and indicates the automatic fill point where the endpoint FIFO automatically commits a buffer to the USB interface. If the device enumerates in Full Speed, then the Packet Length Bits must be set to 64 bytes. Whenever the device is USB reset and re-enumerates, then these register bits must be set based on the connection speed.

6.8 EPxPFH/L Registers 0x12–0x17

The Programmable Flag registers control when the PF goes active for each of the endpoint FIFOs: EP2 and EP6. The EPxPFH/L fields are interpreted differently for the high speed operation and full speed operation and for OUT and IN endpoints.

The following is the register bit definition for high-speed operation:

High Speed Mode: EP2PFL, EP6PFL **0x13, 0x17**

Bit #	7	6	5	4	3	2	1	0
Bit Name	PFC7	PFC6	PFC5	PFC4	PFC3	PFC2	PFC1	PFC0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

High Speed Mode: EP2PFH, EP6PFH **0x12, 0x16**

Bit #	7	6	5	4	3	2	1	0
Bit Name	DECIS	PKTSTAT	IN: PKTS[2] OUT: PFC12	IN: PKTS[1] OUT: PFC11	IN: PKTS[0] OUT: PFC10	0	PFC9	PFC8
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	1	0	0	0	1	0	0	0

The following is the bit definition for the same register when the device is operating at full speed.

Full Speed Mode: EP2PFL, EP6PFL **0x13, 0x17**

Bit #	7	6	5	4	3	2	1	0
Bit Name	IN: PKTS[1] OUT: PFC7	IN: PKTS[0] OUT: PFC6	PFC5	PFC4	PFC3	PFC2	PFC1	PFC0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

Full Speed Mode: EP2PFH, EP6PFH **0x12, 0x16**

Bit #	7	6	5	4	3	2	1	0
Bit Name	DECIS	PKTSTAT	OUT: PFC12	OUT: PFC11	OUT: PFC10	0	PFC9	IN: PKTS[2] OUT: PFC8
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	1	0	0	0	1	0	0	0

6.8.1 DECIS: EPxPFH.7

If DECIS = 0, then PF goes high when the byte count is equal to or less than what is defined in the PF registers. If DECIS = 1 (default), then PF goes high when the byte count equal to or greater than what is set in the PF register. For OUT endpoints, the byte count is the total number of bytes in the FIFO that are available to the external master. For IN endpoints, the byte count is determined by the PKSTAT bit.

6.8.2 PKSTAT: EPxPFH.6

For IN endpoints, the PF can apply to either the entire FIFO, comprising multiple packets, or only to the current packet being filled. If PKTSTAT = 0 (default), the PF refers to the entire IN endpoint FIFO. If PKTSTAT = 1, the PF refers to the number of bytes in the current packet.

PKTSTAT	PF Applies To	EPnPFH:L Format
0	Number of committed packets + current packet bytes	PKTS[...] and PFC[...]
1	Current packet bytes only	PFC[...]

6.8.3 IN: PKTS(2:0)/OUT: PFC[12:10]: EPxPFH[5:3]

These three bits have a different meaning, depending on whether this is an IN or OUT endpoint.

6.8.3.1 IN Endpoints

If IN endpoint, the meaning of these *EPxPFH[5:3]* bits depend on the PKTSTAT bit setting. When PKTSTAT = 0 (default), the PF considers when there are PKTS packets plus PFC bytes in the FIFO. PKTS[2:0] determines how many packets are considered, according to the following table.

Table 6-4. PKTS Bits

PKTS2	PKTS1	PKTS0	Number of Packets
0	0	0	0
0	0	1	1
0	1	0	2

When PKTSTAT = 1, the PF considers when there are PFC bytes in the FIFO, no matter how many packets are in the FIFO. The PKTS[2:0] bits are ignored.

6.8.3.2 OUT Endpoints

The PF considers when there are PFC bytes in the FIFO regardless of the PKTSTAT bit setting.

6.9 EPxFLAGS Registers 0x1E–0x1F

The EPxFLAGS provide an alternate way of checking the status of the endpoint FIFO flags. If enabled, the Bridge firmware can interrupt the external master when a flag is asserted, and the external master can read these two registers to determine the state of the FIFO flags. If the INFM1 and/or OEP1 bits are set, then the EPxEF and EPxFF bits are actually empty +1 and full –1.

EP2FLAGS **0x1E**

Bit #	7	6	5	4	3	2	1	0
Bit Name	0	0	1	0	0	EP2PF	EP2EF	EP2FF
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	1	0	0	0	1	0

EP6FLAGS **0x1F**

Bit #	7	6	5	4	3	2	1	0
Bit Name	0	0	1	0	0	EP6PF	EP6EF	EP6FF
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	1	0	0	0	1	0

Bit 2: EPxPF

This bit is the current state of endpoint x's programmable flag.

Bit 1: EPxEF

This bit is the current state of endpoint x's empty flag. EPxEF = 1 if the endpoint is empty.

Bit 0: EPxFF

This bit is the current state of endpoint x's full flag. EPxFF = 1 if the endpoint is full.

6.10 INPKTEND/FLUSH Register 0x20

This register allows the external master to duplicate the function of the PKTEND pin. The register also allows the external master to selectively flush endpoint FIFO buffers.

INPKTEND/FLUSH **0x20**

Bit #	7	6	5	4	3	2	1	0
Bit Name	0	FIFO6	0	FIFO2	EP3	EP2	EP1	EP0
Read/Write	W	W	W	W	W	W	W	W
Default	0	0	0	0	0	0	0	0

Bit [4:7]: FIFOx

These bits allows the external master to selectively flush any or all of the endpoint FIFOs. By writing the desired endpoint FIFO bit, the Bridge firmware logic flushes the selected FIFO. For example setting bit 6 flushes endpoint 6 FIFO.

Bit [3:0]: EPx

These bits are is used only for IN transfers. By writing the desired endpoint number (2 or 6), the Bridge firmware logic automatically commits an IN buffer to the USB host. For example, for committing a packet through endpoint 6 set the lower nibble to 6, set bits 1 and 2 high.

6.11 USBFRAMEH/L Registers 0x2A, 0x2B

Every millisecond, the USB host sends an SOF token indicating 'Start Of Frame,' along with an 11-bit incrementing frame count. The Bridge firmware copies the frame count into these registers at every SOF.

USBFRAMEH **0x2A**

Bit #	7	6	5	4	3	2	1	0
Bit Name	0	0	0	0	0	FC10	FC9	FC8
Read/Write	R	R	R	R	R	R	R	R
Default	X	X	X	X	X	X	X	x

USBFRAMEL **0x2B**

Bit #	7	6	5	4	3	2	1	0
Bit Name	FC7	FC6	FC5	FC4	FC3	FC2	FC1	FC0
Read/Write	R	R	R	R	R	R	R	R
Default	X	X	X	X	X	X	X	X

One use of the frame count is to respond to the USB SYNC_FRAME Request. If the Bridge firmware detects a missing or garbled SOF, the Bridge firmware generates an internal SOF and increments USBFRAMEL–USBRAMEH.

6.12 MICROFRAME Registers 0x2C

MICROFRAME **0x2C**

Bit #	7	6	5	4	3	2	1	0
Bit Name	0	0	0	0	0	MF2	MF1	MF0
Read/Write	R	R	R	R	R	R	R	R
Default	X	X	X	X	X	X	X	x

MICROFRAME contains a count 0–7 that indicates which of the 125 microsecond microframes last occurred. This register is active only when the Bridge is operating in high-speed mode (480 Mbits/sec).

6.13 FNADDR Register 0x2D

During the USB enumeration process, the host sends a device a unique 7-bit address that the Bridge firmware copies into this register. There is normally no reason for the external master to know its USB device address because the Bridge firmware automatically responds only to its assigned address.

FNADDR **0x2D**

Bit #	7	6	5	4	3	2	1	0
Bit Name	HSGRANT	FA6	FA5	FA4	FA3	FA2	FA1	FA0
Read/Write	R	R	R	R	R	R	R	R
Default	0	0	0	0	0	0	0	0

Bit 7: HSGRANT

Set to ‘1’ if the Bridge firmware enumerated at high speed. Set to ‘0’ if the Bridge firmware enumerated at full speed.

Bit[6:0]: FA[6:0]

Address set by the host.

6.14 INTENABLE Register 0x2E

This register is used to enable/disable the various interrupt sources, and by default all interrupts are enabled.

INTENABLE **0x2E**

Bit #	7	6	5	4	3	2	1	0
Bit Name	SETUP	EP0BUF	UNUSED	USB RESET	CMDRDY	ENUMOK	UNUSED	READY
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	1	1	0	1	1	1	0	1

Bit 7: SETUP

Setting this bit to ‘1’ enables an interrupt when a setup packet is received from the USB host.

Bit 6: EP0BUF

Setting this bit to a ’1’ enables an interrupt when the Endpoint 0 buffer becomes available.

Bit 5: Unused

This was formerly the FLAGS interrupt enable. This interrupt has been replaced by the INNF and OUTNE extended interrupts.

Bit 4: USB RESET

Setting this bit to a ‘1’ enables an interrupt when a USB Reset condition is detected.

Bit 3: CMDRDY

Setting this bit to a ‘1’ enables an interrupt when the Bridge firmware is ready for another command. With the exception of this interrupt and the interrupts covered in INTENABLE1, the extended interrupts and the CMDSTATRDY interrupts are always enabled.

Bit 2: ENUMOK

Setting this bit to a ‘1’ enables an interrupt when the Bridge firmware enumeration is complete.

Bit 1: Unused

This was formerly the BUSACTIVITY interrupt enable. This interrupt has been replaced by the HOST_SUSPEND and HOST_RESUME extended interrupts.

Bit 0: READY

Setting this bit to a ’1’ enables an interrupt when the Bridge firmware has powered on and performed an internal self-test.

6.15 Extended Interrupt INTENABLE1 Register 0x2F

This register is used to enable/disable a subset of the extended interrupt sources, and by default all the subset of extended interrupts are disabled. The extended interrupts that are not covered in this register are automatically enabled/disabled based on the CMDSTATRDY bit setting in the INTENABLE register 0x2E,

INTENABLE1 **0x2F**

Bit #	7	6	5	4	3	2	1	0
Bit Name	reserved	reserved	reserved	HS_NOTIFY	EP6INNF	reserved	reserved	EP2OUTNE
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

Bit 4: HS_NOTIFY

Setting this bit to ’1’ enables an interrupt whenever the Bridge gets a High-Speed USB connection signal (the HS extended interrupt). This event occurs whenever the Bridge is connected to the USB and successfully connects at High Speed. Set this bit before commanding the Bridge to enumerate.

Bit 3: EP6INNF

Setting this bit to ’1’ enables an interrupt whenever Endpoint 6, configured as an IN endpoint FIFO, becomes ‘not full’. External master applications must use a counting flag that is initialized to two when the endpoint is initialized or flushed, or when a USB RESET occurs. Whenever a packet is written to the Bridge, decrement the count by ‘1’. Whenever this interrupt is received, increment the count by ‘1’. This interrupt occurs whenever a packet is transferred from this endpoint to the host PC on the USB. The interrupting endpoint is identified in the lower byte of the INNF interrupt.

Bit 2: Reserved

This bit must always be set to ‘0’.

Bit 1: Reserved

This bit must always be set to ‘0’.

Bit 0: EP2OUTNE

Setting this bit to ’1’ enables an interrupt whenever Endpoint 2, configured as an OUT endpoint FIFO, becomes ‘not empty’. External master applications must use a counting flag initialized to ‘0’ when the endpoint is initialized or flushed, or when a USB RESET occurs and incremented by one when this interrupt is received. Whenever a packet is read from the Bridge, then decrement the count by one.This interrupt occurs whenever the Bridge commits a packet from the USB to the FIFO interface signaling to the external master that a packet is available and can be read from the Bridge by the master. The interrupting endpoint is identified in the lower byte of the OUTNE interrupt.

6.16 EP0STALL Register 0x32

This register is used to cause a stall on the control endpoint EP0. Control endpoint EP0 stalls are automatically cleared after the stall events.

EP0STALL **0x32**

Bit #	7	6	5	4	3	2	1	0
Bit Name	d7	d6	d5	d4	d3	d2	d1	d0
Read/Write	W	W	W	W	W	W	W	W
Default	0	0	0	0	0	0	0	0

Bits[7:0]: DATA[7:0]

Writing any non-zero value to DATA[7:0] to set EP0 Stall.

6.17 CPUCS Unindexed Register 0xE600

The CPU Control and Status register is used to enable or disable the CLKOUT pin. The CLKOUT pin is enabled by default and is typically used for debugging. If the CLKOUT pin is not otherwise used in the design, then to reduce noise, the master processor must write to this register to clear the CLKOE bit.

CPUCS **0x32**

Bit #	7	6	5	4	3	2	1	0
Bit Name	reserved	reserved	reserved	CLKSPD1	CLKSPD0	CLKINV	CLKOE	reserved
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R
Default	0	0	0	1	0	0	1	0

Bits[4:3]: CLKSPD[1:0]

These bits are used to set the CPU clock speed. The default setting for these bits is [1 0] which selects 48 MHz clock speed. Other clock speed settings have not been tested for the Bridge firmware.

Speed settings:

- 00: 12 MHz
- 01: 24 MHz
- 10: 48 MHz (default)
- 11: reserved

Bit 2: CLKINV

This bit is used to invert the CLKOUT signal. The default for this bit is [1] which selects normal output.

- 0: CLKOUT normal
- 1: CLKOUT inverted

Bit 1: CLKOE

This bit is used to enable the CPU clock output on the CLKOUT pin. The default for this bit is [1] which selects clock output enabled.

- 0: CLKOUT pin is driven
- 1: CLKOUT pin floats

To turn off the CLKOUT pin, the master must perform an unindexed write to the CPUCS register with the value 0x10. This setting preserves the default clock speed settings and turns off the CLKOE.

To avoid altering any other bits of the CPUCS register (0xE6009), the external master can do an unindexed read from CPUCS register, modify the value to set/clear the CLKOE bit and perform an unindexed write of the modified value back to the CPUCS register.

6.17.1 Example Unindexed Register Read

(CPUCS register at 0xE600)

1. Send Low Byte of the Register Address (0x00).
   Write Register (Register ID = 0x3A) with value = 0x00.
2. Send High Byte of the Register Address (0xE6).
   Write Register (Register ID - 0x3B) with value = 0xE6.
3. Send the actual value to write to the Register Address (in this case 0x1C).
   Read Register (Register ID = 0x3C) returns value (default = 0x12).

6.17.2 Example Un-indexed Register Write

(CPUCS register at 0xE600 with value = 0x10)

1. Send Low Byte of the Register Address (0x00).
   Write Register (Register ID = 0x3A) with value = 0x00.
2. Send High Byte of the Register Address (0xE6).
   Write Register (Register ID - 0x3B) with value = 0xE6.
3. Send the actual value to write to the Register Address (in this case 0x10).
   Write Register (Register ID = 0x3C) with value = 0x10.

7. Default Descriptor

7.1 Introduction to the Default Descriptor

```
//Device Descriptor
18,               //Descriptor length
1,                //Descriptor type
00,02,            //Specification Version (BCD)
00,               //Device class
00,               //Device sub-class
00,               //Device sub-sub-class
64,               //Maximum packet size
LSB(VID),MSB(VID),//Vendor ID
LSB(PID),MSB(PID),//Product ID
LSB(DID),MSB(DID),//Device ID
1,                //Manufacturer string index
2,                //Product string index
0,                //Serial number string index
1,                //Number of configurations


//DeviceQualDscr
10,               //Descriptor length
6,                //Descriptor type
0x00,0x02,        //Specification Version (BCD)
00,               //Device class
00,               //Device sub-class
00,               //Device sub-sub-class
64,               //Maximum packet size
1,                //Number of configurations
0,                //Reserved

//HighSpeedConfigDscr
9,                //Descriptor length
2,                //Descriptor type
39,               //Total Length (LSB)
0,                //Total Length (MSB)
1,                //Number of interfaces
1,                //Configuration number
0,                //Configuration string
0xA0,             //Attributes (b7 - buspwr, b6 - s
50,               //Power requirement (div 2 ma)
```

```
//Interface Descriptor
9,                  //Descriptor length
4,                  //Descriptor type
0,                  //Zero-based index of this interface
0,                  //Alternate setting
3,                  //Number of end points
0xFF,               //Interface class
0x00,               //Interface sub class
0x00,               //Interface sub sub class
0,                  //Interface descriptor string index

//Endpoint Descriptor -- Bulk Endpoint 2 OUT
7,                  //Descriptor length
5,                  //Descriptor type
0x02,               //Endpoint number, and direction
2,                  //Endpoint type
0x00,               //Maximum packet size (LSB)
0x02,               //Max packet size (MSB)
0x00,               //Polling interval

//Endpoint Descriptor -- Bulk Endpoint 6 IN
7,                  //Descriptor length
5,                  //Descriptor type
0x86,               //Endpoint number, and direction
2,                  //Endpoint type
0x00,               //Maximum packet size (LSB)
0x02,               //Max packet size (MSB)
0x00,               //Polling interval

//Endpoint Descriptor -- Interrupt Endpoint 1 IN
7,                  //Descriptor length
5,                  //Descriptor type
0x81,               //Endpoint number, and direction
3,                  //Endpoint type
0x40,               //Maximum packet size (LSB)
0x00,               //Max packet size (MSB)
0x00,               //Polling interval


//FullSpeedConfigDscr
9,                  //Descriptor length
2,                  //Descriptor type
39,                 //Total Length (LSB)
0,                  //Total Length (MSB)
1,                  //Number of interfaces
1,                  //Configuration number
0,                  //Configuration string
0xA0,               //Attributes (b7 - buspwr, b6 - selfpwr, b5 - rwu)
50,                 //Power requirement (div 2 ma)
```

```
//Interface Descriptor
9,                  //Descriptor length
4,                  //Descriptor type
0,                  //Zero-based index of this interface
0,                  //Alternate setting
3,                  //Number of end points
0xFF,               //Interface class
0x00,               //Interface sub class
0x00,               //Interface sub sub class
0,                  //Interface descriptor string index


//Endpoint Descriptor
7,                  //Descriptor length
5,                  //Descriptor type
0x02,               //Endpoint number, and direction
2,                  //Endpoint type
0x40,               //Maximum packet size (LSB)
0x00,               //Max packet size (MSB)
0x00,               //Polling interval


//Endpoint Descriptor
7,                  //Descriptor length
5,                  //Descriptor type
0x86,               //Endpoint number, and direction
2,                  //Endpoint type
0x40,               //Maximum packet size (LSB)
0x00,               //Max packet size (MSB)
0x00,               //Polling interval


//Endpoint Descriptor -- Interrupt Endpoint 1 IN
7,                  //Descriptor length
5,                  //Descriptor type
0x81,               //Endpoint number, and direction
3,                  //Endpoint type
0x40,               //Maximum packet size (LSB)
0x00,               //Max packet size (MSB)
0x00,               //Polling interval


//StringDscr

//StringDscr0
4,                  //String descriptor length
3,                  //String Descriptor
0x09,0x04,          //US LANGID Code
```

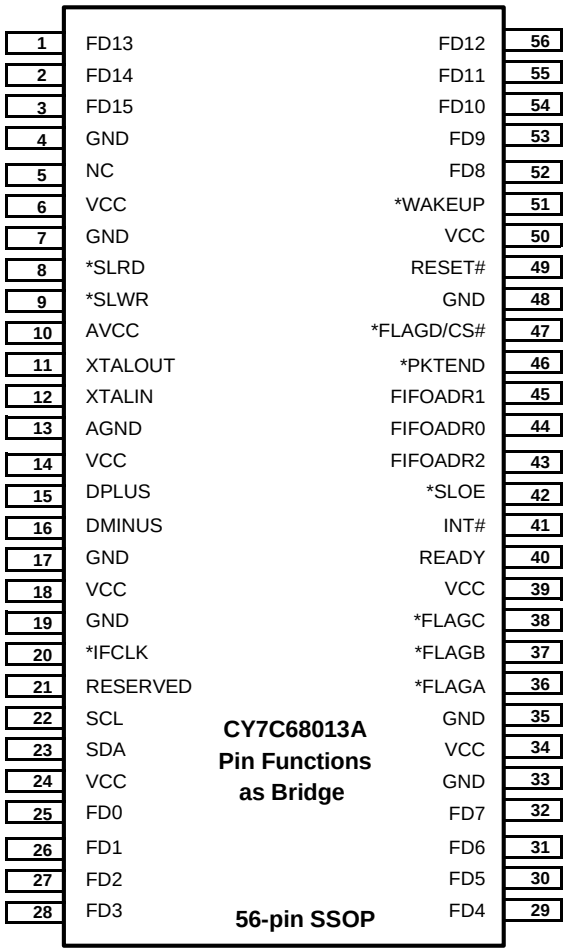
```
//StringDscr1
16,               //String descriptor length
3,                //String Descriptor
'C',00,
'y',00,
'p',00,
'r',00,
'e',00,
's',00,
's',00,

//StringDscr2
20,               //String descriptor length
3,                //String Descriptor
'C',00,
'Y',00,
'7',00,
'C',00,
'6',00,
'8',00,
'0',00,
'0',00,
'2',00,
```

8. Pin Assignments

8.1 56-Pin SSOP

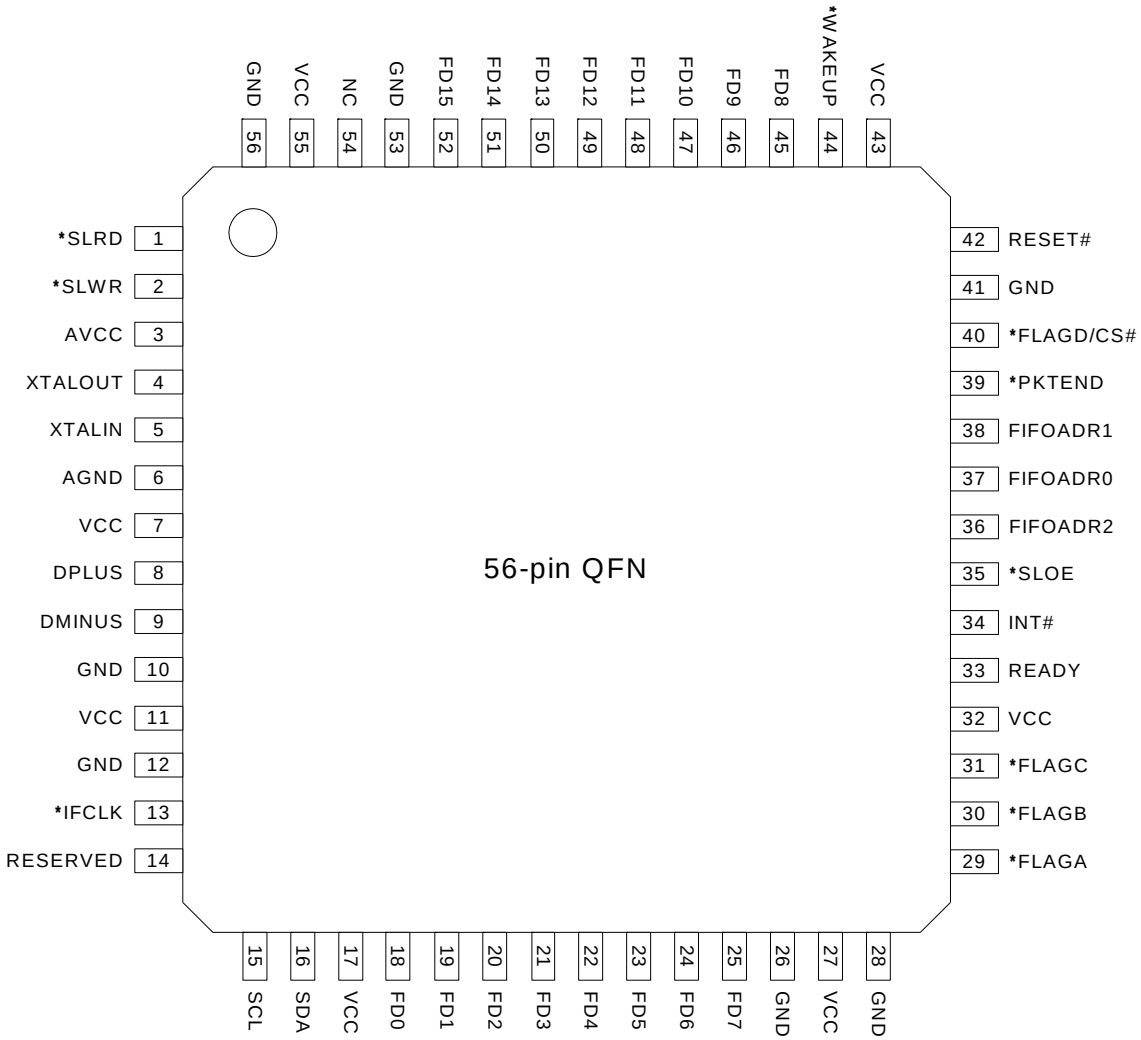
Figure 8-1. 56-pin SSOP Pin Assignment[1]



1. * Denotes programmable polarity.

8.2 56-Pin QFN

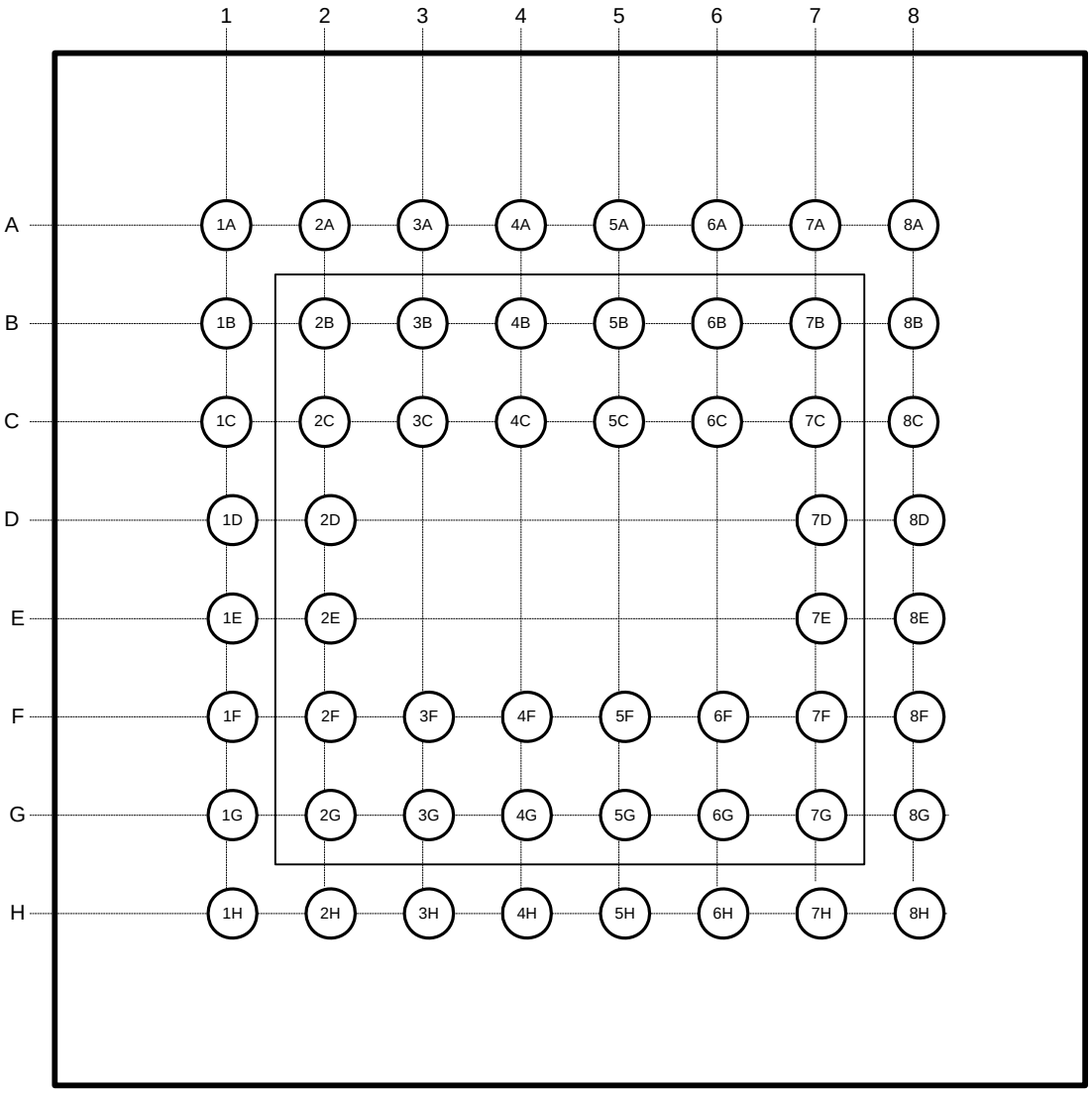
Figure 8-2. 56-pin QFN Pin Assignment[2]



2. * Denotes programmable polarity.

8.3 56-Pin VFBGA

Figure 8-3. 56-pin VFBGA Pin Assignment - Top View



8.4 MoBL-USB™ FX2LP18 Device Pin Descriptions

Table 8-1. Pin Descriptions

QFN Pin	SSOP Pin	VFBGA Pin	Name	Type	Default	Description
3	10	2D	AVCC	Power	N/A	**Analog V_{CC}**. Connect this pin to 3.3V power source. This signal provides power to the analog section of the chip.
6	13	2F	AGND	Power	N/A	**Analog Ground**. Connect to ground with as short a path as possible.
9	16	1E	DMINUS	I/O/Z	Z	**USB D– Signal**. Connect to the USB D– signal.
8	15	2E	DPLUS	I/O/Z	Z	**USB D+ Signal**. Connect to the USB D+ signal.
42	49	8B	RESET#	Input	N/A	**Active LOW Reset**. Resets the entire chip. This pin is normally tied to V_{CC} through a 100K resistor, and to GND through a 0.1-µF capacitor.
5	12	1C	XTALIN	Input	N/A	**Crystal Input**. Connect this signal to a 24 MHz parallel-resonant, fundamental mode crystal and load capacitor to GND. It is also correct to drive XTALIN with an external 24 MHz square wave derived from another clock source.
4	11	2C	XTALOUT	Output	N/A	**Crystal Output**. Connect this signal to a 24 MHz parallel-resonant, fundamental mode crystal and load capacitor to GND. If an external clock is used to drive XTALIN, leave this pin open.
54	5	2B	NC	Output	O	**No Connect**. This pin must be left unconnected.
33	40	8G	READY	Output	N/A	**READY**. Indicates that the Bridge is ready for commands (the command buffer has space for additional commands). This pin is NOT the same as the READY interrupt. This pin is high-impedance for a short time at startup.
34	41	6G	INT#	Output	H	**INT#**. This is an output-only external interrupt signal. Active Low.
35	42	8F	SLOE	Input	I	**SLOE**. This is an input-only output enable with programmable polarity (POLAR.4) for the slave FIFOs connected to FD[7:0] or FD[15:0].
36	43	7F	FIFOADR2	Input	I	**FIFOADR2**. This is an input-only address select for the slave FIFOs connected to FD[7:0] or FD[15:0].
37	44	6F	FIFOADR0	Input	I	**FIFOADR0**. This is an input-only address select for the slave FIFOs connected to FD[7:0] or FD[15:0].
38	45	8C	FIFOADR1	Input	I	**FIFOADR1**. This is an input-only address select for the slave FIFOs connected to FD[7:0] or FD[15:0].
39	46	7C	PKTEND	Input	I	**PKTEND**. This is an input-only packet end with programmable polarity (POLAR.5) for the slave FIFOs connected to FD[7:0] or FD[15:0].
40	47	6C	FLAGD/ CS#	CS#:I FLAGD:O	I	**FLAGD**. This is a programmable slave-FIFO output status flag signal. CS# is a master chip select (default).
18	25	3H	FD[0]	I/O/Z	I	**FD[0]**. This is the bidirectional FIFO/Command data bus.
19	26	4F	FD[1]	I/O/Z	I	**FD[1]**. This is the bidirectional FIFO/Command data bus.
20	27	4H	FD[2]	I/O/Z	I	**FD[2]**. This is the bidirectional FIFO/Command data bus.
21	28	4G	FD[3]	I/O/Z	I	**FD[3]**. This is the bidirectional FIFO/Command data bus.
22	29	5H	FD[4]	I/O/Z	I	**FD[4]**. This is the bidirectional FIFO/Command data bus.
23	30	5G	FD[5]	I/O/Z	I	**FD[5]**. This is the bidirectional FIFO/Command data bus.
24	31	5F	FD[6]	I/O/Z	I	**FD[6]**. This is the bidirectional FIFO/Command data bus.

Table 8-1. Pin Descriptions *(continued)*

QFN Pin	SSOP Pin	VFBGA Pin	Name	Type	Default	Description
25	32	6H	FD[7]	I/O/Z	I	**FD[7]**. This is the bidirectional FIFO/Command data bus.
45	52	8A	FD[8]	I/O/Z	I	**FD[8]**. This is the bidirectional FIFO data bus.
46	53	7A	FD[9]	I/O/Z	I	**FD[9]**. This is the bidirectional FIFO data bus.
47	54	6B	FD[10]	I/O/Z	I	**FD[10]**. This is the bidirectional FIFO data bus.
48	55	6A	FD[11]	I/O/Z	I	**FD[11]**. This is the bidirectional FIFO data bus.
49	56	3B	FD[12]	I/O/Z	I	**FD[12]**. This is the bidirectional FIFO data bus.
50	1	3A	FD[13]	I/O/Z	I	**FD[13]**. This is the bidirectional FIFO data bus.
51	2	3C	FD[14]	I/O/Z	I	**FD[14]**. This is the bidirectional FIFO data bus.
52	3	2A	FD[15]	I/O/Z	I	**FD[15]**. This is the bidirectional FIFO data bus.
1	8	1A	SLRD	Input	N/A	**SLRD**. This is the input-only read strobe with programmable polarity (POLAR.3) for the slave FIFOs connected to FD[7:0] or FD[15:0].
2	9	1B	SLWR	Input	N/A	**SLWR**. This is the input-only write strobe with programmable polarity (POLAR.2) for the slave FIFOs connected to FD[7:0] or FD[15:0].
29	36	7H	FLAGA	Output	H	**FLAGA**. This is a programmable slave-FIFO output status flag signal. Defaults to PF for the FIFO selected by the FIFOADR[2:0] pins.
30	37	7G	FLAGB	Output	H	**FLAGB**. This is a programmable slave-FIFO output status flag signal. Defaults to FULL for the FIFO selected by the FIFOADR[2:0] pins.
31	38	8H	FLAGC	Output	H	**FLAGC**. This is a programmable slave-FIFO output status flag signal. Defaults to EMPTY for the FIFO selected by the FIFOADR[2:0] pins.
13	20	2G	IFCLK	I/O/Z	Z	**Interface Clock**. This is used for synchronously clocking data into or out of the slave FIFOs. IFCLK also serves as a timing reference for all slave FIFO control signals. When using the internal clock reference (IFCONFIG.7=1), the IFCLK pin can be configured to output 30/48 MHz by setting bits IFCONFIG.5 and IFCONFIG.6. IFCLK may be inverted by setting the bit IFCONFIG.4=1. Programmable polarity.
14	21	2H	Reserved	Input	N/A	**Reserved**. This pin must be connected to ground.
44	51	7B	WAKEUP	Input	N/A	**USB Wakeup**. If the Bridge is in suspend, asserting this pin starts the oscillator and interrupts the Bridge firmware to allow it to exit the suspend mode. During normal operation, holding WAKEUP asserted inhibits the Bridge chip from suspending. This pin has programmable polarity (POLAR.7).
15	22	3F	SCL	OD	Z	**I^2C Clock**. Connect to V_{CC} with a 2.2K -10 K ohms resistor, even if no I^2C EEPROM is attached. (For CY7C68053: Connect to V_{CC_IO} with the resistor.)
16	23	3G	SDA	OD	Z	**I^2C Data**. Connect to V_{CC} with a 2.2K -10 K Ohms resistor, even if no I^2C EEPROM is attached. (For CY7C68053: Connect to V_{CC_IO} with the resistor.)

Table 8-1. Pin Descriptions *(continued)*

QFN Pin	SSOP Pin	VFBGA Pin	Name	Type	Default	Description
55	6	5A	V_{CC}	Power	N/A	**V_{CC}**. Connect to 3.3V power source. (For CY7C68053: V_{CC_IO} - connect to 1.8V - 3.3V power source.)
7	14	1D	AV_{CC}	Power	N/A	**Analog V_{CC}**. Connect this pin to 3.3V power source. Connect to 3.3V power source.
11	18	1G	V_{CC}	Power	N/A	**V_{CC}**. Connect to 3.3V power source. (For CY7C68053: $V_{CC,}$ connect to 1.8V power source.)
17	24	7E	V_{CC}	Power	N/A	**V_{CC}**. Connect to 3.3V power source. (For CY7C68053: $V_{CC_IO,}$ connect to 1.8V - 3.3V power source.)
27	34	8E	V_{CC}	Power	N/A	**V_{CC}**. Connect to 3.3V power source. (For CY7C68053: $V_{CC_IO,}$ connect to 1.8V - 3.3V power source.)
32	39	5C	V_{CC}	Power	N/A	**V_{CC}**. Connect to 3.3V power source. (For CY7C68053: $V_{CC,}$ connect to 1.8V power source.)
43	50	5B	V_{CC}	Power	N/A	**V_{CC}**. Connect to 3.3V power source. (For CY7C68053: $V_{CC_IO,}$ connect to 1.8V - 3.3V power source.)
53	4	4A	GND	Ground	N/A	**Connect to ground**.
56	7	4B	GND	Ground	N/A	**Connect to ground**.
10	17	1F	AGND	Ground	N/A	**Analog Ground**. Connect to ground with as short a path as possible
12	19	1H	GND	Ground	N/A	**Connect to ground**.
26	33	7D	GND	Ground	N/A	**Connect to ground**.
28	35	8D	GND	Ground	N/A	**Connect to ground**.
41	48	4C	GND	Ground	N/A	**Connect to ground**.

9. External Interface Description

9.1 External Interface Pin Description

Table 9-1. Pin Cross Reference

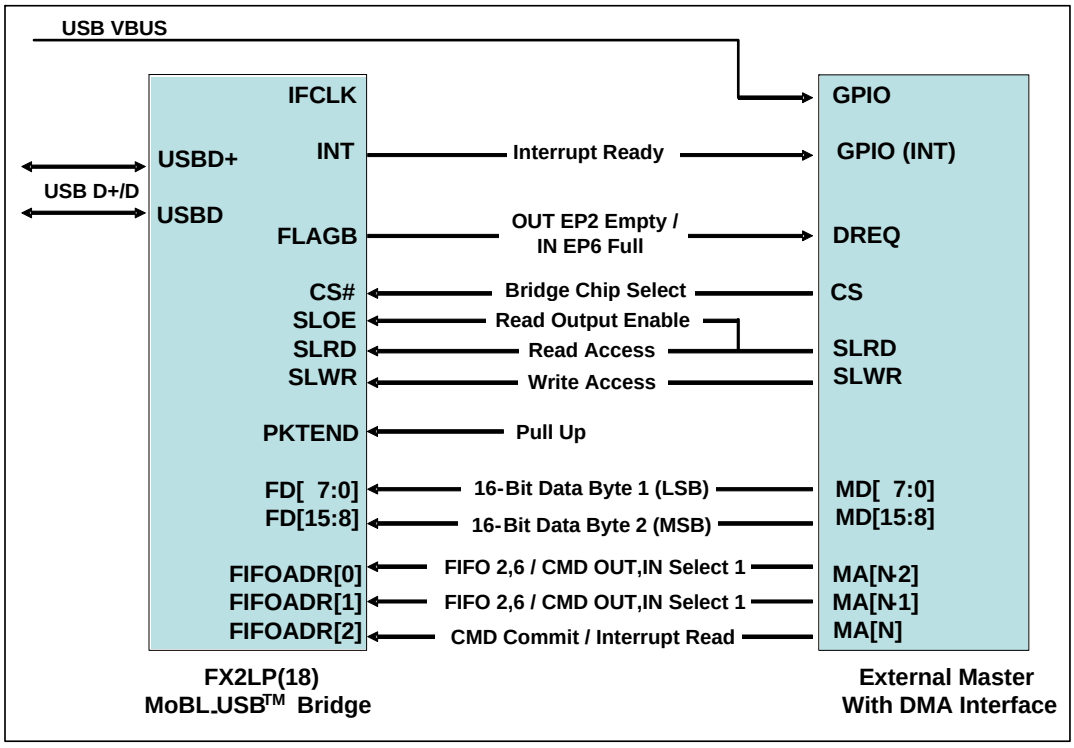
FX2LP Pin Name	Bridge Slave FIFO	Description
PA0/INT0#	READY	**READY** indicates that the Bridge firmware is ready for commands (the command buffer has space for additional commands). This pin is NOT the same as the READY interrupt. This pin is high-impedance for a short time at startup.
PA1/INT1#	INT#	**INT#** is an output-only external interrupt signal. Active Low.
PA2/SLOE	SLOE	**SLOE** is an input-only output enable with programmable polarity (POLAR.4) for the slave FIFOs connected to FD[7:0] or FD[15:0].
PA3/WU2	FIFOADR2	**FIFOADR2** is an input-only address select for the slave FIFOs connected to FD[7:0] or FD[15:0].
PA4/FIFOADR0	FIFOADR0	**FIFOADR0** is an input-only address select for the slave FIFOs connected to FD[7:0] or FD[15:0].
PA5/FIFOADR1	FIFOADR1	**FIFOADR1** is an input-only address select for the slave FIFOs connected to FD[7:0] or FD[15:0].
PA6/PKTEND	PKTEND	**PKTEND** is an input-only packet end with programmable polarity (POLAR.5) for the slave FIFOs connected to FD[7:0] or FD[15:0].
PA7/FLAGD/SLCS#	FLAGD/SLCS#	**FLAGD** is a programmable slave-FIFO output status flag signal. **SLCS#** is a master chip select (default).
CTL2/FLAGC	FLAGC	**FLAGC** is a programmable slave-FIFO output status flag signal. Defaults to EMPTY for the FIFO selected by the FIFOADR[2:0] pins.
CTL1/FLAGB	FLAGB	**FLAGB** is a programmable slave-FIFO output status flag signal. Defaults to FULL for the FIFO selected by the FIFOADR[2:0] pins.
CTL0/FLAGA	FLAGA	**FLAGA** is a programmable slave-FIFO output status flag signal. Defaults to PF for the FIFO selected by the FIFOADR[2:0] pins.
RDY0/SLRD	SLRD	**SLRD** is the input-only read strobe with programmable polarity (POLAR.3) for the slave FIFOs connected to FD[7:0] or FD[15:0].
RDY1/SLWR	SLWR	**SLWR** is the input-only write strobe with programmable polarity (POLAR.2) for the slave FIFOs connected to FD[7:0] or FD[15:0].
IFCLK	IFCLK	**Interface Clock** is used for synchronously clocking data into or out of the slave FIFOs. IFCLK also serves as a timing reference for all slave FIFO control signals. When using the internal clock reference (IFCONFIG.7=1) the IFCLK pin can be configured to output 30/48 MHz by setting bits IFCONFIG.5 and IFCONFIG.6. IFCLK may be inverted by setting the bit IFCONFIG.4=1. Programmable polarity.
WAKEUP	WAKEUP	**USB Wakeup** If the Bridge firmware is in suspend, asserting this pin starts up the oscillator and interrupts the Bridge firmware to allow it to exit the suspend mode. During normal operation, holding WAKEUP asserted inhibits the Bridge firmware chip from suspending. This pin has programmable polarity (POLAR.7).
PB0-PB7/FD0-FD7	FD0-FD7	Lower byte of FIFO/Command data bus.
PD0-PD7/FD8-FD15	FD8-FD15	Upper byte of FIFO/Command data bus.

10. DMA Interface

10.1 Fast Slave Interface to DMA Interface on External Processor

External master processors with DMA capabilities readily interface with the Bridge firmware slave FIFOs to perform fast data and command transfers. Figure 10-1 shows how a typical DMA interface capable microprocessor connection to an FX2LP running the Bridge firmware might appear.

Figure 10-1. Typical DMA Interface



The example illustrated interface uses a single DMA Channel for all Master/Slave transactions–data read, data write, and command/status operations–between the Bridge firmware and external master. If a two DMA channel master is used, then FLAGA can be used for one of the channels instead of sharing FLAGB. The DMA interface must be capable of burst mode transfers using DREQ as a burst toggle control and it must also be capable of self acknowledging each data access. The following is a brief description of the Bridge firmware interface connections to the master processor.

- INT -> GPIOx (INT). Ensures timely reading of the interrupt status value. GPIOx must be programmed to trigger an interrupt on the falling edge.
- FLAG B -> DREQ. Allows throttling of the DMA request flow.
- CS# -> CS. Chip select (CS# function of Bridge FLAG D) localizes addressing of the Bridge on a common address bus.

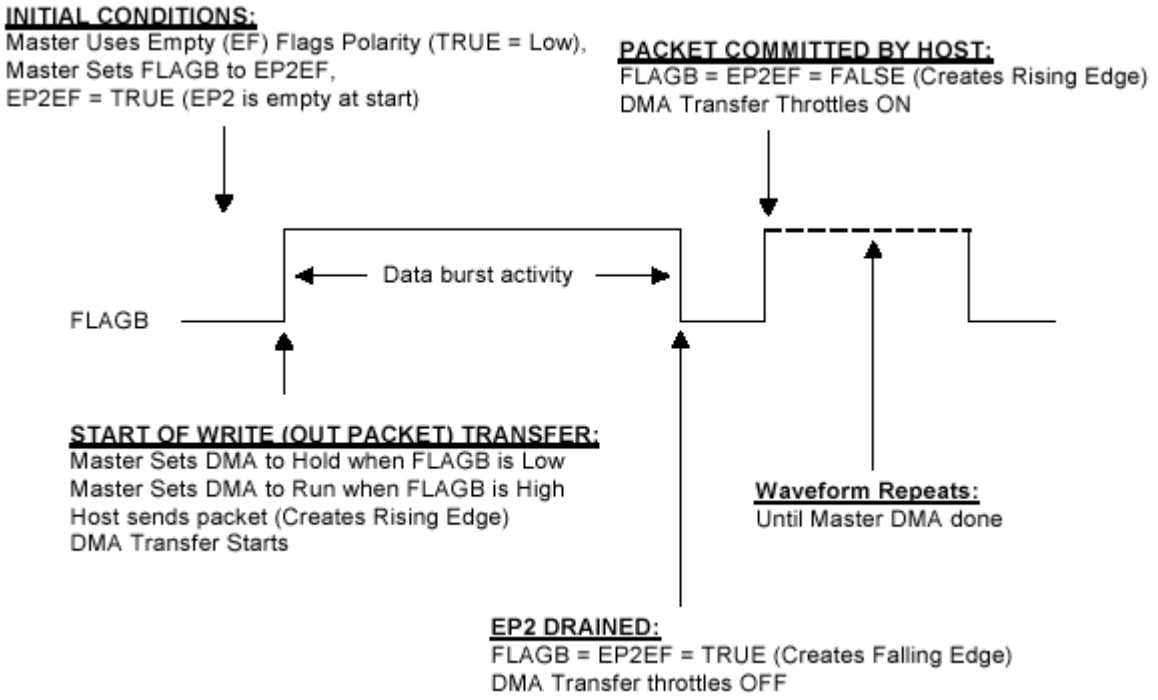
- SLOE/SLRD -> DRD. Single read signal on master during DMA read burst operations drives Bridge SLOE and SLRD signals for asynchronous signalling.
- SLWR <- DWR. Single write signal on master during DMA write burst operations drives Bridge SLWR signal.
- FD[7:0] <- MD[7:0]. The External master reads and writes data in 16-bit operations from or to the Bridge. Command, status, and data operations use pins FD[7:0] to carry the first byte (LSB) of a 16-bit data word.
- FD[15:8] <- MD[15:8]. The External master reads and writes data in 16-bit operations from or to the Bridge. Command, status, and data operations use pins FD[15:8] to carry the second byte (MSB) of a 16-bit data word.
- FIFOADR[2:0] <- MA[N:N-2]. DMA memory addressing lines to source or sink Bridge endpoint and command data.

10.2 DREQ Waveform Generation with FLAGB

10.2.1 OUT Packet Read Bursts

The FLAGB line is used to create a waveform to throttle a DMA read burst of data from the Bridge to the external master when reading OUT endpoint packet contents. Programming FLAGB as an Endpoint 2 FIFO empty flag with default, active-low polarity creates a low to high transition when the Endpoint 2 FIFO buffer goes not-empty and creates a DMA request on the external master DREQ line. When the FIFO goes empty, a high to low transition occurs, creating a DMA request termination. This waveform throttles the DMA request line indefinitely or until the DMA transfers a preset amount of data.

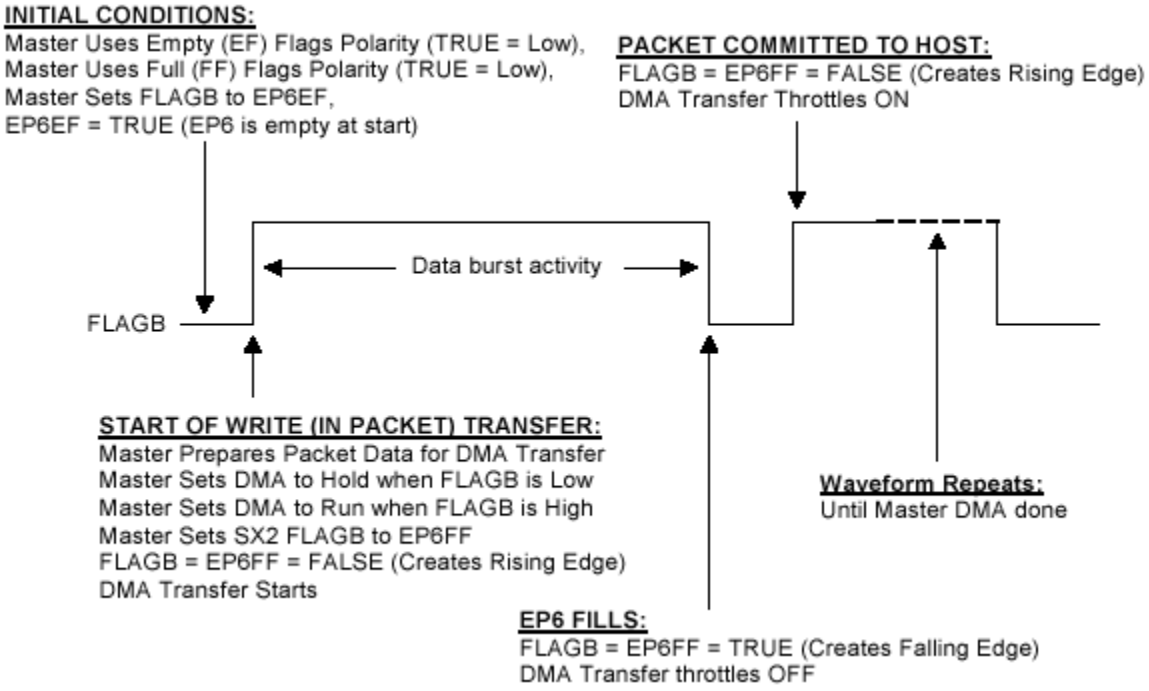
Figure 10-2. OUT Packet Read Bursts



10.2.2 IN Packet Write Bursts

The FLAGB line is again used to create a waveform to throttle a DMA burst write from the external master to the Bridge when writing IN endpoint packet contents. Programming FLAGB as an Endpoint 6 FIFO empty flag with active-low polarity ensures that FLAGB is low at the start of the transfer. Reprogramming FLAGB as an Endpoint 6 FIFO full flag creates a low to high transition for the DREQ signal which remains high until the full flag condition occurs. When the Endpoint 6 FIFO buffer goes full, FLAGB asserts and throttles a DMA burst transfer.

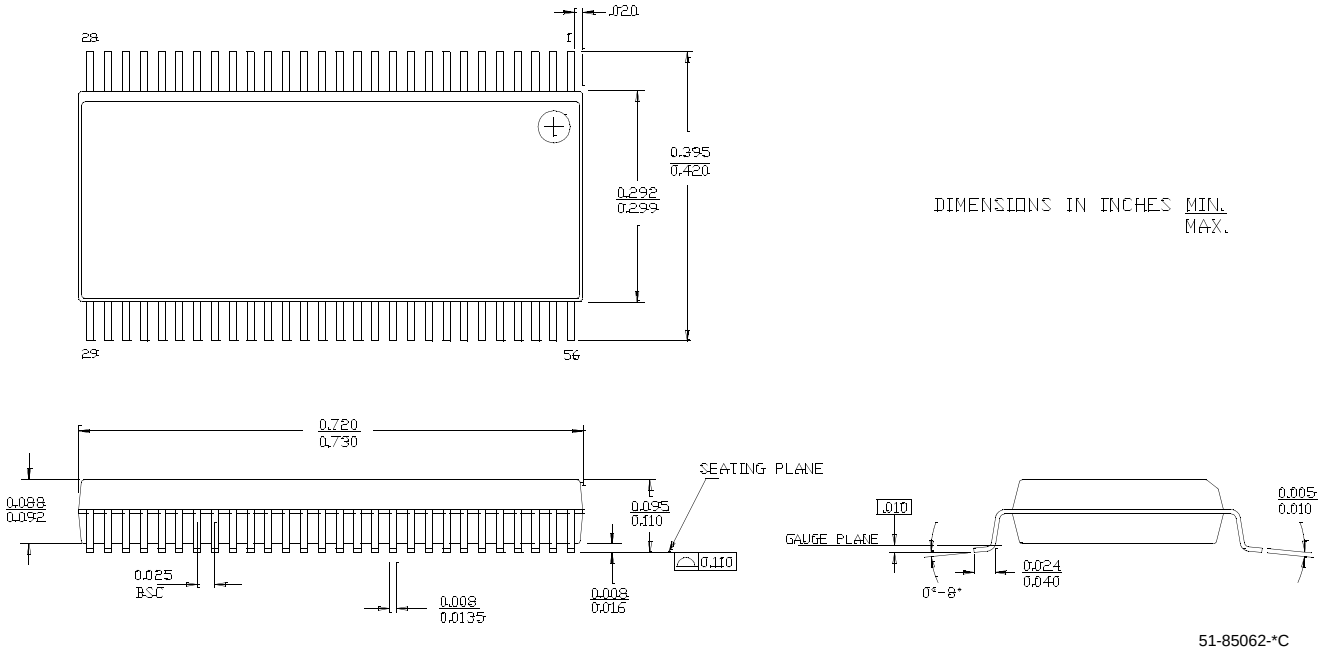
Figure 10-3. IN Packet Write Bursts



11. Package Diagrams

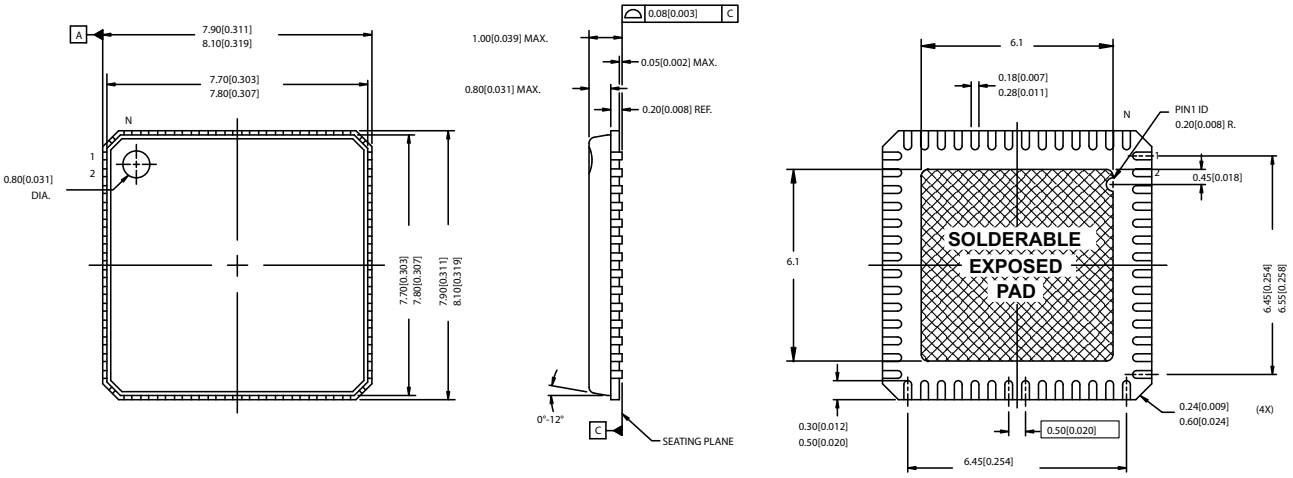
11.1 56-pin SSOP Package

Figure 11-1. 56-lead Shrunk Small Outline Package O56



11.2 56-pin QFN Package

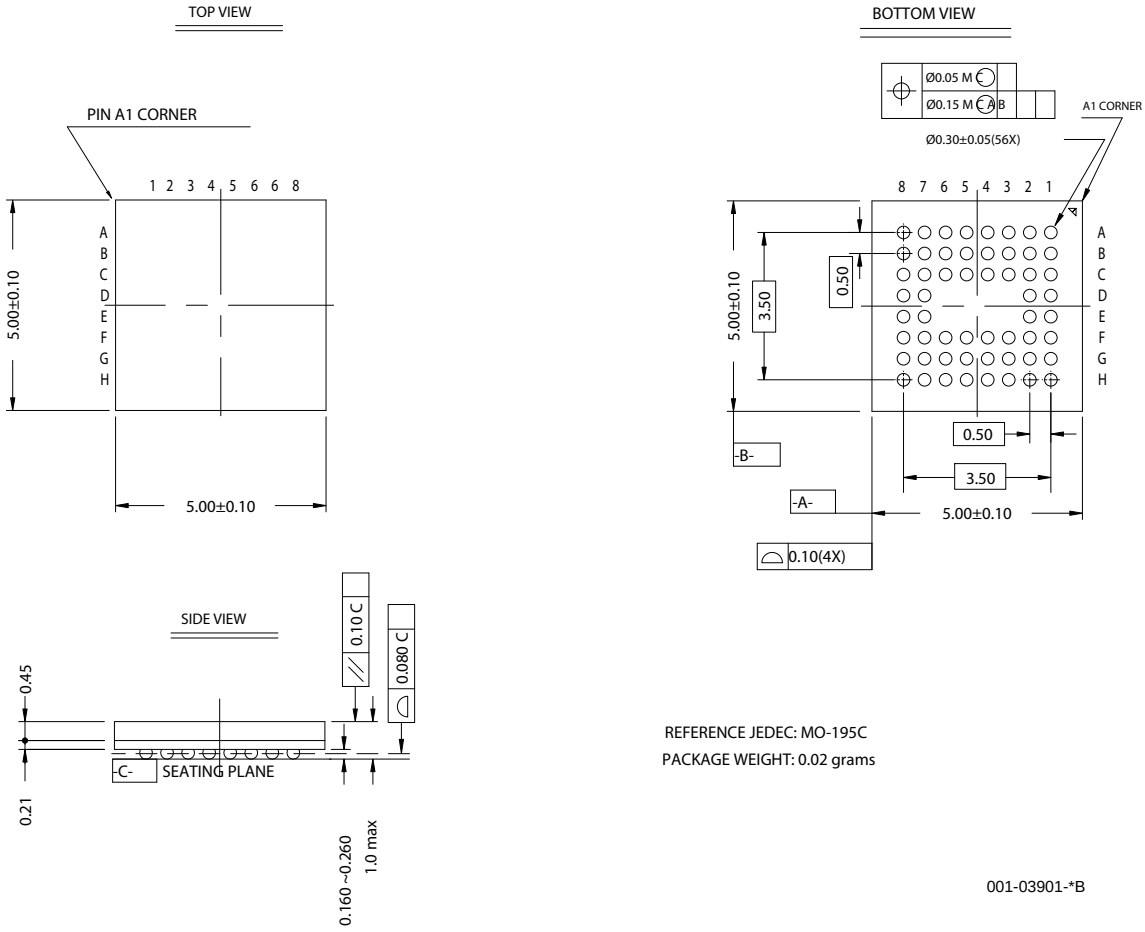
Figure 11-2. 56-Lead QFN 8 x 8 mm LF56A



51-85144-*G

11.3 56-pin VFBGA Package

Figure 11-3. 56 VFBGA (5 x 5 x 1.0 mm) 0.50 Pitch, 0.30 Ball BZ56



Index

#

56-pin QFN pin assignment 60
56-pin SSOP pin assignment 59
56-pin VFBGA pin assignment 61

A

acronyms 8
architecture
    interrupt system 34

B

bit definition
    INTENABLE register 34
bridge interfaces 17
bridge solution example 11
bridge system states 12

C

command protocol 20
command ready interrupt 36

D

default descriptor 32, 55
default enumeration 32
DMA interface 67
document history 8
documentation
    acronyms 8
    overview 7

E

enabling OUT transfers 21
endpoint RAM organization 16
EP2PKTLENL register 45
EP6PKTLENL register 45
EPxCFG register 44
example code 55
external interface 16
external master 32
external master processors 67

F

features 9
FIFO
    configuration, of flag pins 19
    default programmable flag setup 19
    programmable flag setup 19
FIFO access 18
FIFOADDR lines 17
firmware images 27
FLAGSAB register 41
FLAGSCD register 41

H

host suspend/resume 28

I

IFCLK 18
IFCONFIG register 40
IN setup transaction 37
initialize extended endpoints 21
INTENABLE1 register 51
interrupt status byte 35
interrupt system
    architecture 34

L

list of commands 30

M

manual enumeration 31
MoBL-USB FX2LP18 device pin descriptions 62

O

ordering information 13
OUT setup transaction 38
overview 7

P

passthrough enumeration 33
pin assignment
    56-pin QFN 60
    56-pin SSOP 59
    56-pin VFBGA 61
pin cross reference 65
POLAR register 43

R

read EP1OUT data 22
read EP4OUT data 24
register
    EP2PKTLENL 45
    EP6PKTLENL 45
    EPxCFG 44
    FLAGSAB 41
    FLAGSCD 41
    INTENABLE1 51
    POLAR 43
    REVID 44
REVID register 44

S

setup commands 37
standby sequences 27
system diagram 11

T

typical applications 10

U

USB cable unplug 29
USB reset 16

W

wakeup 16
write EP1IN data 24
write EP8IN data 25
write IN data 26
write short IN data 26

Revision History (All Docs)

Document Revision History

Document Title: CY4625 MoBL-USB Bridge Firmware Guide Document Number: 001-15686				
Revision	**ECN#**	**Issue Date**	**Origin of Change**	**Description of Change**
**	1094923	See ECN	ARI	New user guide.
Distribution: Internal **Posting**: None				