Getting Started with FX2LP™

Author: Rama Sai Krishna. V

Associated Project: No
Associated Part Family: CY7C68013A/14A/15A/16A
Software Version: None
Related Documents: For a complete list of the documents, click here.

AN65209 gets you started with the EZ-USB® FX2LP™ USB 2.0 Device Controller. It provides background information on USB 2.0 and it details hardware, firmware, and software aspects of working with the FX2LP.

Contents

Introduction to USB

Universal serial bus (USB) was designed to standardize the connection of the computer peripherals, such as keyboards, mice, printers, pen drives, hard disks, and portable media players, both to communicate and supply power needed for them. USB is the most common connectivity solution for PCs and consumer devices today. Plug and play, easy to use and simple to implement, USB continues to gain traction in new applications and market segments. Coming to the history of the USB specifications, the first version of the specification, USB 1.0 was released in 1996. This version of specification defines two transfer speeds to address the different types of devices available until then. 1.5 Mbps (low-speed) is to address the low speed devices like keyboard and joysticks, 12 Mbps (full-speed) is to address the devices like disk drives. USB 2.0 specification was released in 2000 and it allows the maximum signaling rate of 480 Mbps (high-speed), which is 40 times to the signaling rate of full-speed. USB 3.0 specification was released in 2008 and it allows the maximum signaling rate of 5 Gbps (Superspeed), which is 10 times to the signaling rate of high-speed.

Cypress FX2LP is a USB 2.0 peripheral controller and the following sections talk about the overview of USB 2.0, features of FX2LP, hardware, software and firmware resource that are available for FX2LP.

Cypress offers a wide range of USB products including low-speed, full-speed, and Superspeed devices. Refer to the overview of the USB product portfolio on Cypress website.

USB 2.0 Overview

Signaling rates supported by USB 2.0:

The USB 2.0 transfers signal and power over a four wire cable and the supported data rates are:

- The USB high-speed signaling bit rate is 480 Mbps.
- The USB full-speed signaling bit rate is 12 Mbps.
- The USB low-speed signaling bit rate is 1.5 Mbps.

NOTE: FX2LP does not support the low-speed signaling mode of 1.5 Mbps.

If you are looking for a signaling rate more than 480 Mbps on the USB bus then you should go for USB 3.0.

Cables and Connectors:

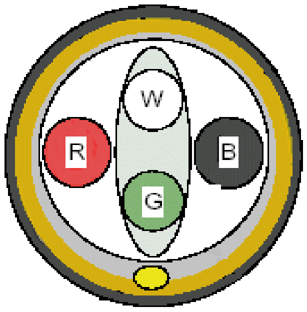
USB cable consists of four conductors, two power conductors, and two signal conductors. High-/full-speed cable consists of a signaling twisted pair, VBUS, GND, and an overall shield. Table 1 shows the description of these lines.

Table 1. USB 2.0 Pin Description

Pins Name	Description	Color
VBUS	Power	Red
D+	Differential pair	Green
D-		White
GND	Ground for power return	Black

Figure 1 shows the cross section of the USB 2.0 cable.

Figure 1. USB 2.0 Cable Cross Section



Series “A” connector is used to connect the upstream towards the host and Series “B” connector is used to connect the downstream towards the devices. Standard Series “B” connector is defined for relatively large and stationary devices like hard drives, printers. Micro A-type and B-type connectors are defined for handheld devices.

Protocol Layer:

USB 2.0 transaction consists of three packets: Token, Data, and Handshake. Transaction is initiated with the token packet and it is always from the host. Data packets deliver the payload data and it can be sourced by the host or device. Handshake packet acknowledges the error-free receipt of data and it is being sent by the receiver of data.

USB transfer may consist of one or more transactions and may spread across multiple frames. There are four types of transfers defined by the USB Specification:

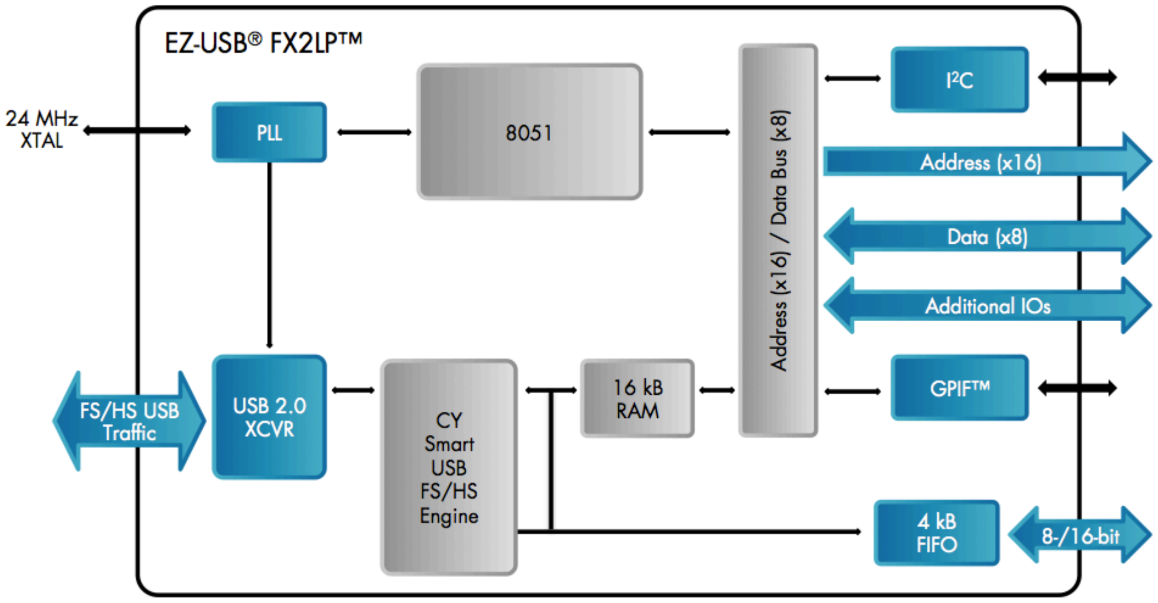
- Control
- Interrupt
- Bulk
- Isochronous

For more details on USB, visit www.usb.org/developers/docs/

Introduction to FX2LP

EZ-USB FX2LP (CY7C68013A/14A/15A/16A) is a programmable, low-power USB 2.0 USB peripheral controller. The FX2LP chipset integrates the USB 2.0 transceiver, serial interface engine (SIE), an enhanced 8051 microcontroller, and a programmable peripheral interface in a single chip. Using this, you can create a wide range of cost-effective solutions with superior time-to-market. The low-power consumption enables you to create both bus-powered and self-powered applications.

Figure 2. FX2LP Block Diagram



Features of FX2LP

USB:

- FX2LP has an integrated USB 2.0 transceiver, serial interface engine (SIE).
- 4 KB of endpoint memory.
- Endpoint type (BULK, INTERRUPT and ISOCHRONOUS) and Endpoint buffering (double, triple, quad) are programmable.
- Additional programmable (BULK/INTERRUPT) 64 byte endpoint.
- 7 physical endpoints including the control endpoint.

CPU and memory:

- FX2LP has an enhanced 8051 core with two USART, three counter/timers, and an expanded interrupt system. The core can work on 48 MHz, 24 MHz, or 12 MHz clock.
- 16 KB of on-chip code/data RAM.
- Program needs to be stored in an external nonvolatile memory as no flash memory exists in the FX2LP. You can also download the program from the USB host.

Serial interfaces:

- Integrated I^2C controller, which runs at 100 or 400 kHz.
- UART interface.
- Up to 40 GPIOs.

Parallel interface:

- Has a general-purpose programming interface (GPIF) with 8/16-bit external data interface. Using GPIF, FX2LP can directly connect to most parallel interfaces. GPIF has 9 address lines, 6 control lines and 6 ready signals. Typically used when FX2LP is a master on the interface.
- Slave FIFO interface – typically used when FX2LP needs to function as a slave device on the interface.

Different variants of FX2LP

FX2LP is available in three different Pb free pin/packages:

- 56-pin version available in SSOP, QFN, VFBGA packages.
- 100-pin version available only in TQFP package.
- 128-pin version available only in TQFP package.

Table 2 shows the comparison of these three different packages of FX2LP.

Table 2. Comparison of Features of Three Different Packages of FX2LP

Feature	56-pin package	100-pin package	128-pin package
8-bit I/O ports	3 ports (Port A, B, D)	5 ports (Port A, B, C, D, E)	5 ports (port A, B, C, D, E)
I2C bus	Available	Available	Available
UART	Not Available. So firmware debugging through UART is not possible with this package.	Available	Available
GPIF as master	■ 8- or 16-bit GPIF multiplexed onto Port B and D. ■ 5 non-multiplexed control signals	■ 8- or 16-bit GPIF multiplexed onto Port B and D. ■ 12 non-multiplexed control signals ■ Nine GPIF address lines, multiplexed onto PORTC (eight) and PORTE (one)	■ 8- or 16-bit GPIF multiplexed onto Port B and D. ■ 12 non-multiplexed control signals ■ Nine GPIF address lines, multiplexed onto PORTC (eight) and PORTE (one)
Slave FIFO	■ 8- or 16-bit Slave FIFO Interface multiplexed onto Port B and D. ■ 5 non-multiplexed control signals and 4 or 5 control signals multiplexed with Port A.	■ 8- or 16-bit Slave FIFO Interface multiplexed onto Port B and D. ■ 5 non-multiplexed control signals and 4 or 5 control signals multiplexed with Port A.	■ 8- or 16-bit Slave FIFO Interface multiplexed onto Port B and D. ■ 5 non-multiplexed control signals and 4 or 5 control signals multiplexed with Port A.
Other features	CY7C68015A and CY7C68016A have two additional GPIO signals to provide more flexibility when neither IFCLK or CLKOUT are needed in the 56-pin package. PE0 (Port E, pin 0) replaces the IFCLK and PE1 replaces the CLKOUT.	RD# and WR# signals which may be used as read and write strobes for PORTC	RD# and WR# signals which may be used as read and write strobes for PORTC 16-bit 8051 address bus to access off chip memory. 8-bit 8051 data bus. Address/data bus control signals

In the same USB High-speed peripherals family, we have the following additional devices:

- **AT2LP:** Cypress's EZ-USB® AT2LP™ (CY7C68300C/301C/320C) implements a fixed-function bridge between one USB port and one or two ATA- or ATAPI-based mass storage device ports. The PATA interface on AT2LP enables the use of hard disk drives (HDD), compact flash, and solid state drives (SSD) in your design. The AT2LP is perfect for mass storage type applications and enable quick time to market without the hassle of custom firmware. AT2LP supports all ATA/ATAPI-6 compliant mass storage devices.
- **NX2LP-Flex:** Cypress's EZ-USB NX2LP-Flex™ (CY7C68033/34) is a fixed-function, low-power programmable USB to SLC NAND controller. The flexibility of NX2LP-Flex makes it superior to other fixed function NAND controller. Its programmability allows for designers to include special features in the controller along with support multiple different NAND devices easily with one single controller. The hardware ECC engine present in NX2LP-Flex supports 1-bit error correction and 2-bit error detection.
- **SX2:** The EZ-USB SX2 (CY7C68001) is a programmable device 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. SX2 has a built-in USB transceiver and serial interface engine (SIE), along with a command decoder to send and receive USB data. The controller has four endpoints that share a 4-KB FIFO space for maximum flexibility and throughput. SX2 has three address pins and a selectable 8- or 16- bit data bus for command and data input or output.

- **FX2LP18:** MoBL-USB FX2LP18 (CY7C68053) operates at 1.8V and it is specially designed for handheld devices. AN6076 lists all the differences between FX2LP and FX2LP18.

Design Resources

For designing products based on FX2LP, the following design resources are available:

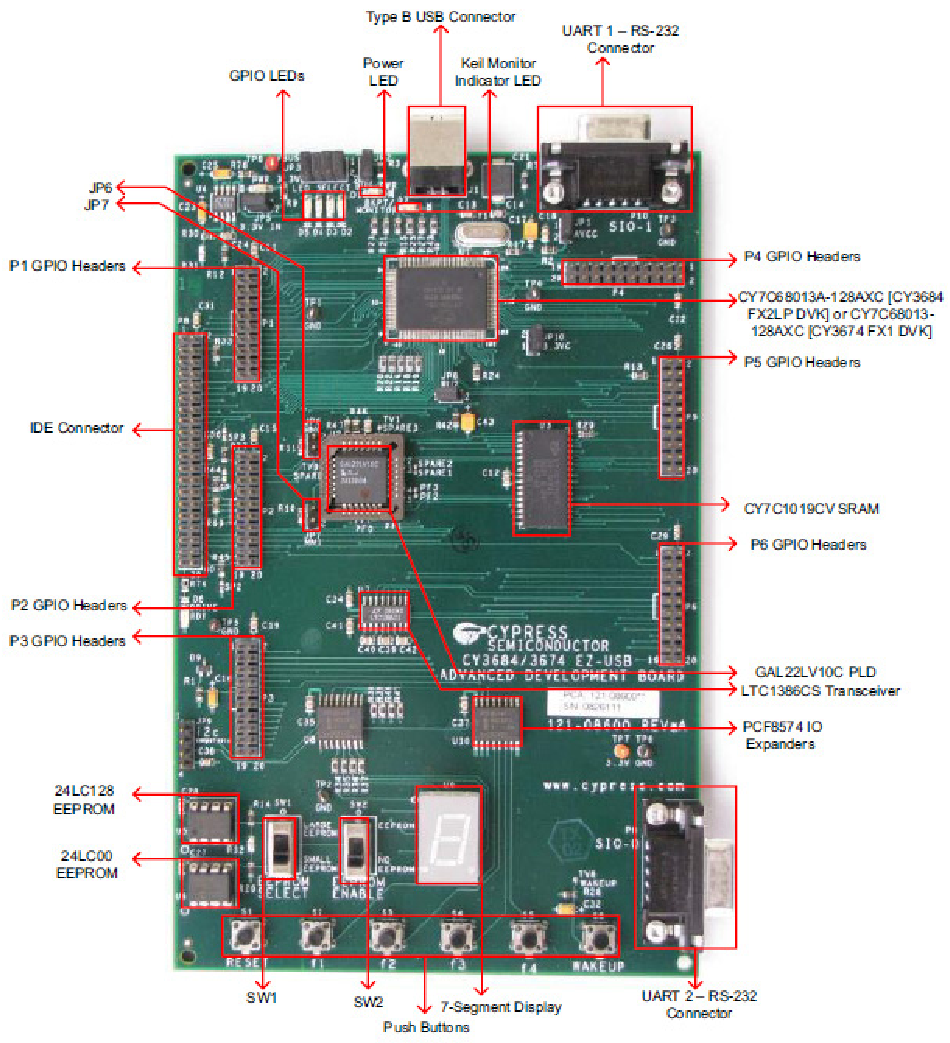
- FX2LP development kit (DVK)
- Software tools
- Firmware example projects
- Reference designs
- Documentation

Following sections present an overview of each design resource. This will help in the identification of the right collateral for a particular application.

FX2LP Development Kit (DVK)

The CY3684 EZ-USB FX2LP Development Kit is the most important resource for development. It provides a platform to develop and test custom projects. It is also a valuable source of support collateral, helping to speed up product design and, thereby, reducing the time-to-market. The development kit contains supporting collateral for the firmware, hardware, and software aspects of a design.

Figure 3. FX2LP Development Kit



This FX2LP-DVK can be used as the debugging platform for FX2LP. The on-board EEPROM can store different configuration options for FX2LP.

Note Since this development kit is for EZ-USB, which is a product family name that includes both FX2LP and FX1, the support collaterals in the DVK are divided into separate folders for FX1-based designs (FX1) and for FX2LP-based designs (FX2LP).

When creating a USB 2.0 high-speed device, board layout and design are critical to the success of the project. Oversights in the layout and design of the board can cause the device enumeration and data transfer to fail. It may also cause a board, which appears to function properly in a particular environment, to fail USB compliance testing. To help developers avoid common errors, resulting in hours of debugging and possible costly board spins, Cypress has several resources available to help design a new board.

The hardware directory of the FX2LP DVK contains the FX2LP development kit schematic, the BOM for the development board, and the development board layout and design files.

Software Tools

For aiding the firmware development and for achieving effective communication with the host, Cypress provides the necessary software tools, driver, and API library. These consist of the 'Cypress USB console', 'GPIF designer', 'Cypress generic USB driver', and 'Cypress USB class Library API'.

Note To incorporate new technologies, these 'software resources' are continuously improved. Therefore, Cypress recommends you to download the latest software from the Cypress website.

Cypress Control Center (CyControl)

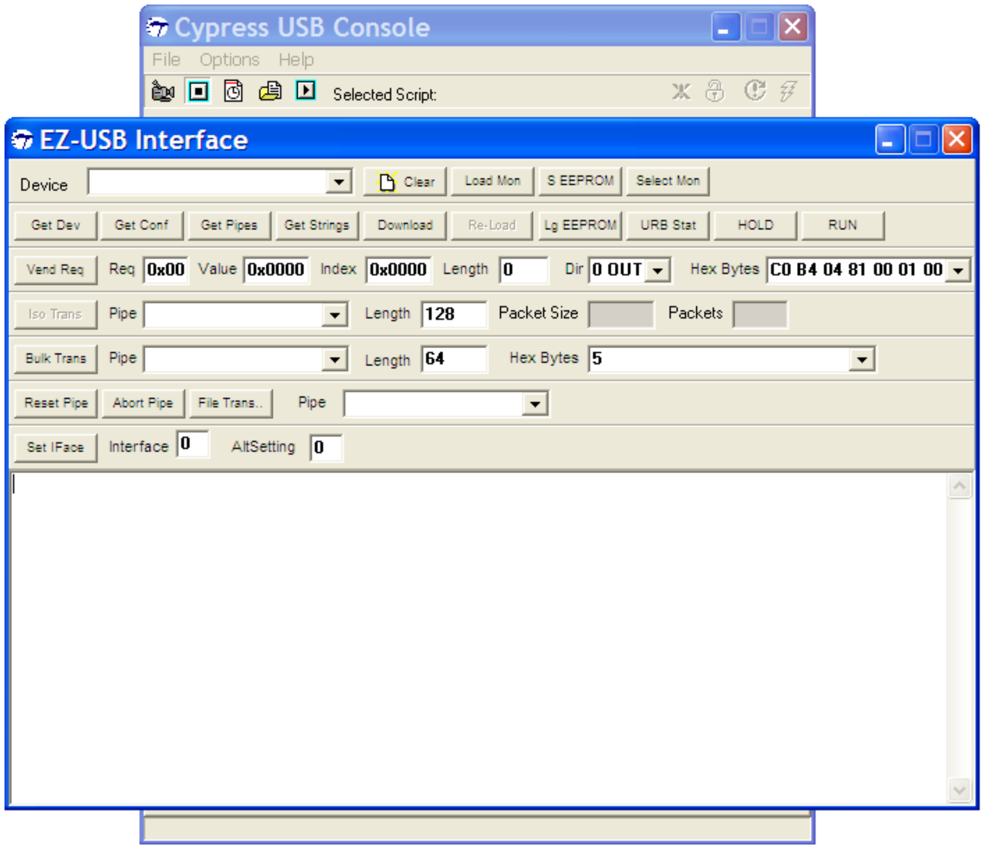
The CyControl is a host application, which can be used for communicating with your device through the Cypress generic USB driver (CyUSB.sys). This application does not come as a part of DVK install, but it is a part of the SuiteUSB3.4 package. The SuiteUSB package is a set of USB host application development tools for Visual studio. The CyControl application is placed in the 'CyConsole' folder after the SuiteUSB package installation. The SuiteUSB installer is a part of FX2LP DVK and also available at the Cypress website. Detailed documentation about the utility is available in the 'CyConsole.pdf' document, placed under the 'CyConsole' folder of the SuiteUSB3.4 installation. The application working details also can be accessed by clicking the 'Help' menu on the tool itself.

The SuiteUSB package is common for the host .NET windows application development for all Cypress USB2.0 chipset families.

Since it is an important application for exploring and exercising the USB targets, it is recommended that, after establishment of a working connection between the host and target, you should explore this tool for its generic uses like downloading the final hex file into FX2LP RAM, Programming the EEPROM connected to FX2LP, to transfer data to a particular endpoint, to send vendor commands and so on.

The snapshot of the CyControl is shown in the following figure:

Figure 4. CyControl



Req 0x00 Value 0x0000 Index 0x0000 Length 0 Dir 0 OUT Hex Bytes C0 B4 04 81 00 01 00

Pipe Length 128 Packet Size Packets

Pipe Length 64 Hex Bytes 5

Pipe

Interface 0 AltSetting 0

Cypress GPIF Designer

The general programmable interface (GPIF) is an extremely flexible parallel interface, which can be programmed to act as a glueless interface, for connecting many types of external peripherals to EZ-USB FX2LP. The programmed GPIF is powerful enough to directly implement protocols such as ATAPI, EPP parallel port, Utopia, and so on. In GPIF these protocols are stored as 'waveform descriptors' data.

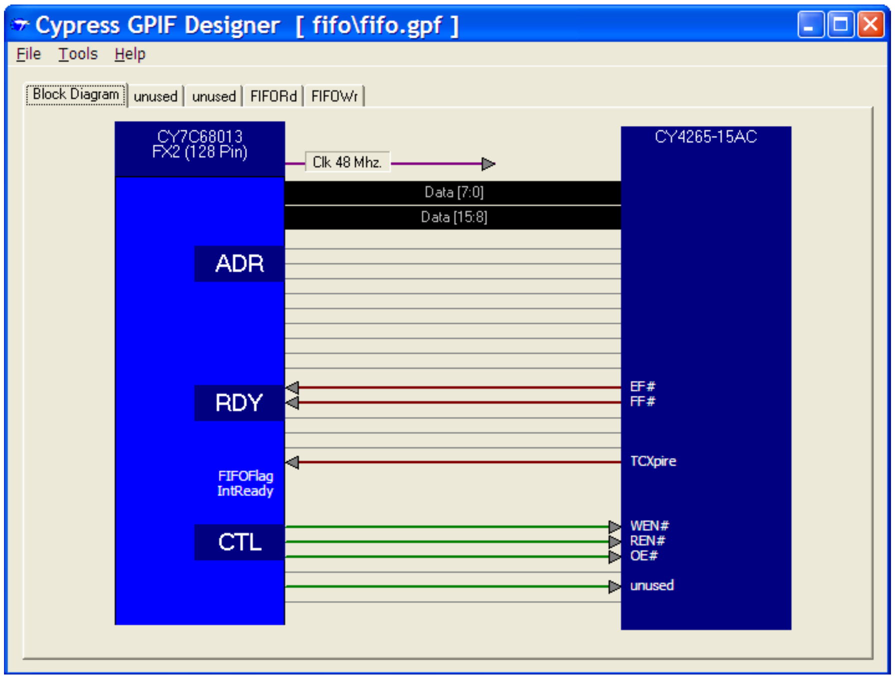
The GPIF designer is a graphical tool (GUI) that allows you to easily create and modify the waveform descriptors required to configure the GPIF. The user needs to export the *Gpif.c* after completion of the waveform development by going to **Tools → Export to *GPIF.c***. This will generate a *.c* file and this file needs to be attached in your project. New GPIF II designer should not be used for FX2 devices as it is not compatible with GPIF designer in any way.

Since this tool is only required for GPIF functionality, it is optional in the EZ-USB FX2LP DVK install and is included as a separate folder in the DVK installable CD. The tool also can be downloaded from the Cypress website.

More information on the importance of GPIF is included in the EZ-USB® FX2LP™ GPIF Design Guide - AN66806. This document also contains examples of how to use the GPIF designer tool. Detailed information on this tool can be found by clicking the 'Help' menu on the tool itself.

The snapshot of the GPIF Designer is shown in the following figure:

Figure 5. GPIF Designer



Cypress Generic USB Driver

The Cypress generic USB driver is a robust high-performance Windows driver used for host communication with a USB connected target. This driver gets installed on the host PC, while doing EZ-USB FX2LP DVK installation. The driver comes in binary form and can be distributed with FX2LP-based devices. Cypress recommends that before distributing the driver it should be 'WHQL certified'. This driver is included in DVK in the 'Drivers' folder. This driver is also included in the SuiteUSB installation and it can be found in the folder 'Driver\bin'. More information on the driver can be found in the document 'CyUSB.pdf' present in the 'Driver' directory of the SuiteUSB install.

Generic USB driver on Windows platform:

WinUSB:

Windows USB (WinUSB) is a generic driver for USB devices that was developed concurrently with the Windows Driver Frameworks (WDF) for Windows XP with SP2. The WinUSB architecture consists of a kernel-mode driver (*Winusb.sys*) and a user-mode dynamic link library (*Winusb.dll*) that exposes WinUSB functions. By using these functions, you can manage USB devices with user-mode software. Winusb.sys is supported on Windows XP/Vista/7/8. More details on WinUSB driver can be found from the msdn site.

Generic USB driver on Linux/Mac OS platform:

LIBUSB:

Libusb is a suite of user-mode routines for controlling data transfer to and from USB devices on Unix-like systems without the need for kernel-mode drivers. Libusb is an open source library that allows you to communicate with USB devices from user space. Libusb-1.0 API Reference provides a detailed documentation of the libusb driver.

Table 3. Comparison of Cypress Generic Driver with the Other Generic USB Drivers

Feature	Cypress USB Driver	WinUSB	LIBUSB
Bulk, control, and interrupt transfers	Supported	Supported	Supported
Isochronous transfers	Supported	Not supported	Supported
OS support	Windows	Windows	Linux, Mac OS X, BSDs.

Cypress USB Class Library API

CyAPI.lib

This is a C++ class library that simplifies the application accesses to custom FX2LP-based devices through the Cypress generic USB driver. The 'Cypress USB Class Library API' (CyApi) is available after the SuiteUSB package installation in the 'CyAPI' folder. This library is compatible with Microsoft tool chains. For more information on how to use these APIs, refer to the CyBulk, CyDesc and Streamer examples located in the 'examples' directory. The API documentation can be found in *CyApi.pdf* under the same directory.

CyUsb.dll

For easing the host application development using C#.NET, Cypress also provides a library named *CyUsb.dll*. You can find this library in the 'lib' folder after the SuiteUSB package installation. The SuiteUSB package is common for the host .NET windows application development on all Cypress USB 2.0 chipset families. The SuiteUSB details and executable file is available at the Cypress website.

Firmware Example Projects

The FX2LP DVK provides a framework that satisfies the Chapter 9 requirements of the USB2.0 specification for high-speed devices. The 'Target\Fw\LP' directory of DVK installation contains a 'firmware framework'. This framework simplifies and accelerates the custom firmware development by using a Cypress pre-developed code for common functionalities like FX2LP chip initialization, USB standard device request handling, USB suspend power management, and so on. The framework also provides function hooks and the firmware examples, making the firmware development process easier. The user needs to provide the USB descriptor table and the code for implementing the custom peripheral functionality.

The FX2LP firmware frameworks are written using Keil uVision2 IDE. Cypress includes an evaluation version of the 8051 Keil Software Tools in the CY3684 EZ-USB FX2LP Development Kit. The supplied Keil tools are fully functional, but are limited in object size to 4 kilobytes. If you need to build an object of size more than 4KB then you need to purchase the license from Keil.

Alternative tools for developing the firmware for FX2LP are the combination of Eclipse, CDT and SDCC plug in. The **CDT** is Eclipse's C/C++ Development Tooling project. Small Device C Compiler (SDCC) allows embedded 'C' applications for 8051 to be developed using the fully featured eclipse IDE. Eclipse is an open source platform, so you can find all these installation files on web for free.

The firmware examples are not stored along with the firmware framework itself, but stored separately in the 'Firmware' directory after installing CY3684 EZ-USB FX2LP Development Kit. These examples can be used as a reference or can be built upon, while developing firmware for custom FX2LP-based products. The following firmware examples are provided in the DVK.

Table 4. Description of FX2LP Firmware Examples

S.No	Firmware Example	Description
1	hid_kb	Example firmware that emulates a HID-class keyboard using the buttons and 7-segment display on the DVK board
2	Bulkloop	Contains a bulk loopback test that exercises the EZ-USB bulk endpoints. It loops back EP2OUT to EP6IN and EP4OUT to EP8IN.
3	Bulkext	Contains a bulk loopback test that exercises the EZ-USB bulk endpoints. The loopback is performed using the external auto pointer. Data is copied from the OUT endpoint buffer to external RAM and then to the IN endpoint buffer. It loops back EP2OUT to EP6IN and EP4OUT to EP8IN
4	Bulksrc	Contains bulk endpoint endless source/sink firmware. It can be driven using the CyConsole or CyBulk. EP2OUT always accepts a bulk OUT; EP4OUT always accept a bulk OUT; EP6IN always returns a 512-byte packet, 64 bytes at full-speed. Based on buffer availability in EP8IN, the most recent packet of EP4OUT is written to EP8IN.
5	dev_io	Contains the source files to build simple development board I/O sample. This software demonstrates how to use the buttons and LED on the EZ-USB development kit.
6	EP_Interrupts	Bulk loopback firmware that demonstrates use of endpoint interrupts using EZ-USB FX2LP.
7	extr_intr	Firmware that demonstrates external interrupt handling INT0, INT1, INT4, INT5, and INT6.
8	ibn	Contains firmware to perform bulk loopback of EP2OUT to EP6IN and EP4OUT to EP8IN using the IBN (In Bulk Nak) interrupt to initiate the transfer

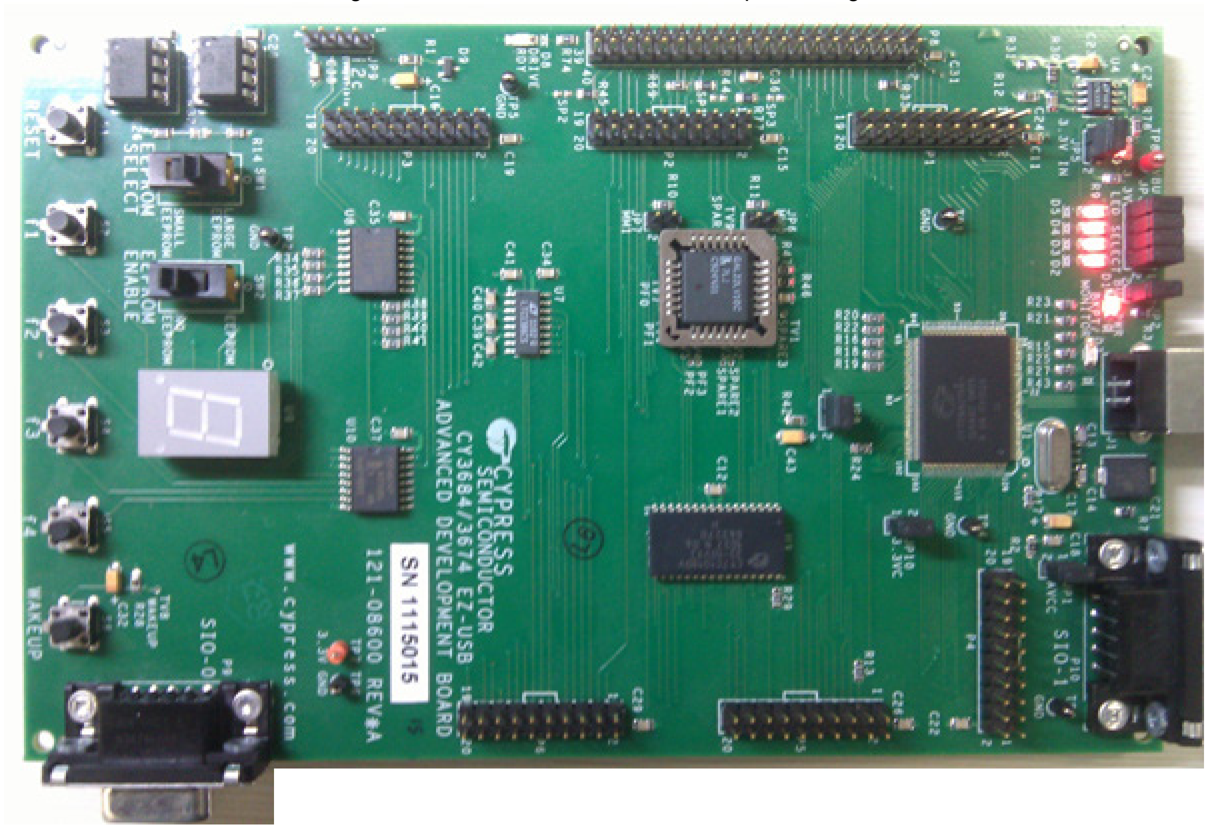
S.No	Firmware Example	Description
9	LEDCycle	Simple firmware example to demonstrate use of the general purpose indicator LEDs (D2, D3, D4, D5) on the Development Kit board.
10	Pingnak	Contains firmware to perform bulk loopback of EP2OUT to EP6IN and EP4OUT to EP8IN using the PING NAK interrupt to initiate the transfer.
11	iMemtest	Memory test firmware example. Tests on-chip RAM.
12	vend_ax	Contains the source files to build a vendor specific command sample. This example demonstrates how to implement different vendor commands.

The framework uses the EZ-USB library (EZUSB.LIB). The EZ-USB library is an 8051 .LIB file that implements functions that are common to many firmware projects. These functions need not be modified and are therefore provided in library form. However, the kit includes the source code for the library in the event that you need to modify a function or if you just want to know how something is done. Detailed information about the EZ-USB library (section 5.4) and firmware framework (chapter 5) is available in the 'CY3684 DVK Kit_Guide'. This library is included in DVK under the 'Target\Lib\LP' folder.

Steps to Bind cyusb.sys to the Connected FX2LP DVK

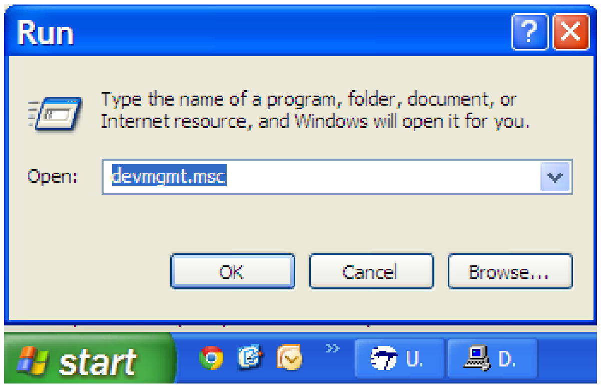
EZ-USB FX2LP DVK with default jumper settings is shown in Figure 6.

Figure 6. FX2LP DVK with Default Jumper Settings



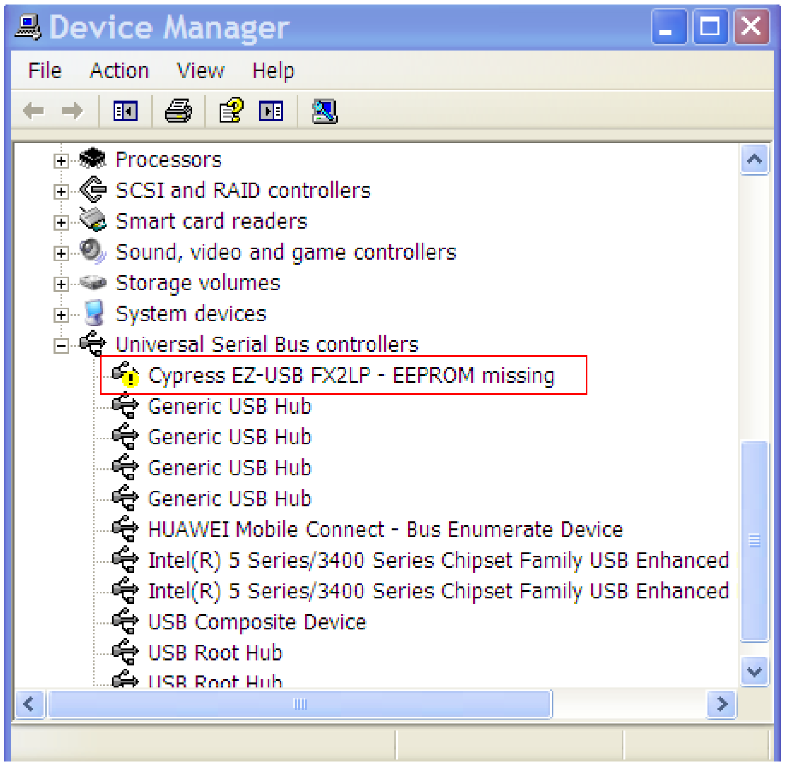
1. FX2LP DVK enumerates with the Vendor ID (VID) 0x04B4 and the Product ID (PID) 0x8613, when you connect it to PC using USB 2.0 cable. Open "**Device Manager**" and look for the new devices.

Figure 7. Opening Device Manager


Run
Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.
Open:
OK
Cancel
Browse...
start
U.
D.

Open: devmgmt.msc

Figure 8. Device Manager Before Binding the Driver to FX2LP

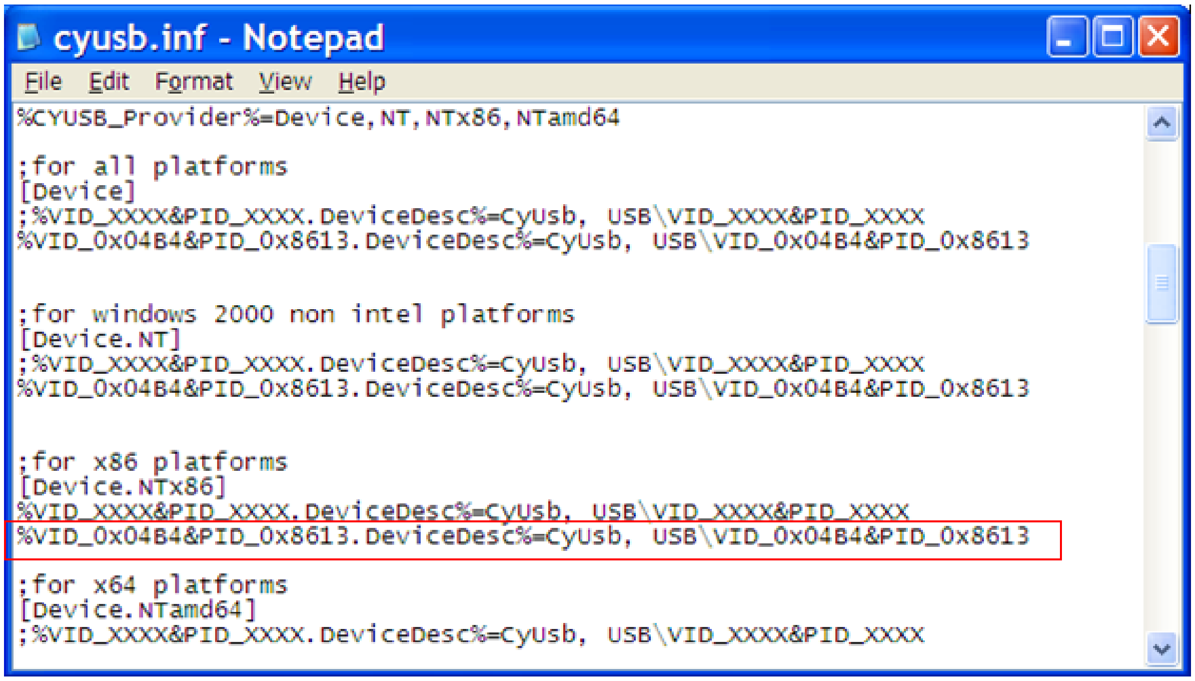


Processors
SCSI and RAID controllers
Smart card readers
Sound, video and game controllers
Storage volumes
System devices
Universal Serial Bus controllers
Cypress EZ-USB FX2LP - EEPROM missing
Generic USB Hub
Generic USB Hub
Generic USB Hub
Generic USB Hub
HUAWEI Mobile Connect - Bus Enumerate Device
Intel(R) 5 Series/3400 Series Chipset Family USB Enhanced
Intel(R) 5 Series/3400 Series Chipset Family USB Enhanced
USB Composite Device
USB Root Hub
USB Root Hub

If you are connecting the FX2LP DVK for the first time then most probably the FX2LP device will show up in the "**other devices**" list.

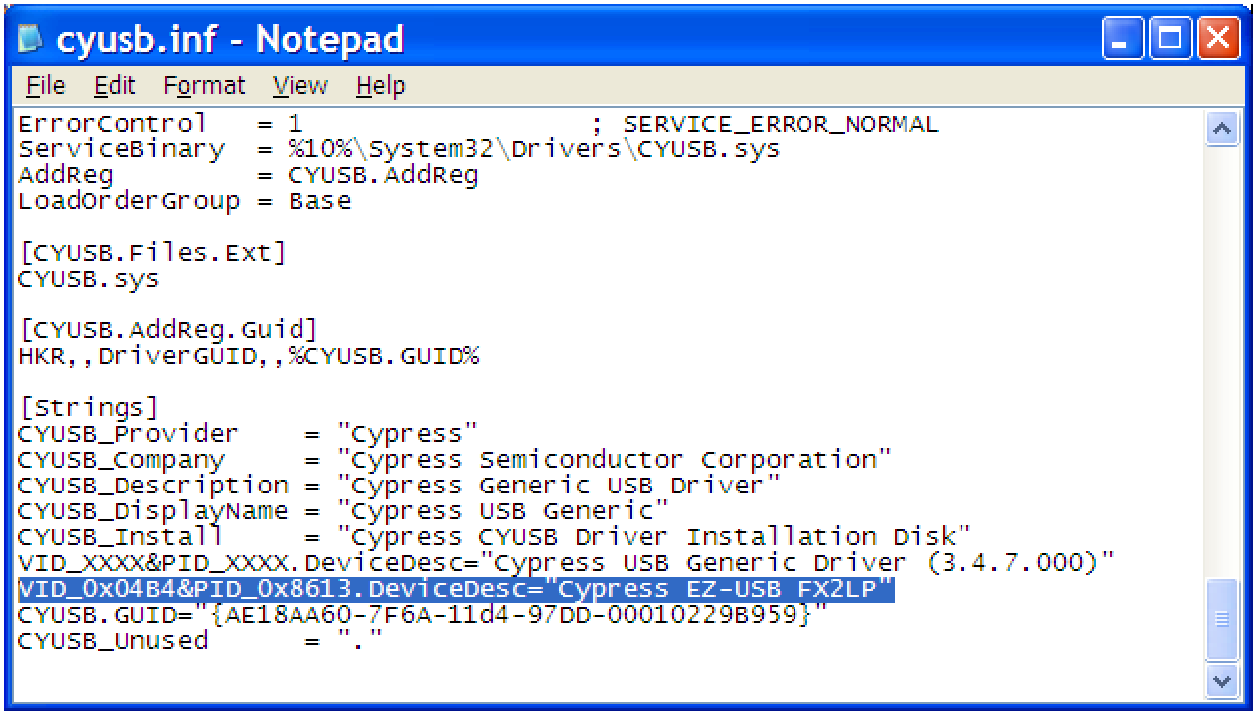
2. Please make sure that the VID and PID of FX2LP is present in the *CyUSB.inf* file. If it is not available please add the VID and PID information as shown in the following image.

Figure 9. Adding the Default VID and PID of FX2LP to cyusb.inf



```
%CYUSB_Provider%=Device,NT,NTx86,NTamd64

;for all platforms
[Device]
;%VID_XXXX&PID_XXXX.DeviceDesc%=CyUsb, USB\VID_XXXX&PID_XXXX
%VID_0x04B4&PID_0x8613.DeviceDesc%=CyUsb, USB\VID_0x04B4&PID_0x8613


;for windows 2000 non intel platforms
[Device.NT]
;%VID_XXXX&PID_XXXX.DeviceDesc%=CyUsb, USB\VID_XXXX&PID_XXXX
%VID_0x04B4&PID_0x8613.DeviceDesc%=CyUsb, USB\VID_0x04B4&PID_0x8613


;for x86 platforms
[Device.NTx86]
%VID_XXXX&PID_XXXX.DeviceDesc%=CyUsb, USB\VID_XXXX&PID_XXXX
%VID_0x04B4&PID_0x8613.DeviceDesc%=CyUsb, USB\VID_0x04B4&PID_0x8613

;for x64 platforms
[Device.NTamd64]
;%VID_XXXX&PID_XXXX.DeviceDesc%=CyUsb, USB\VID_XXXX&PID_XXXX
```

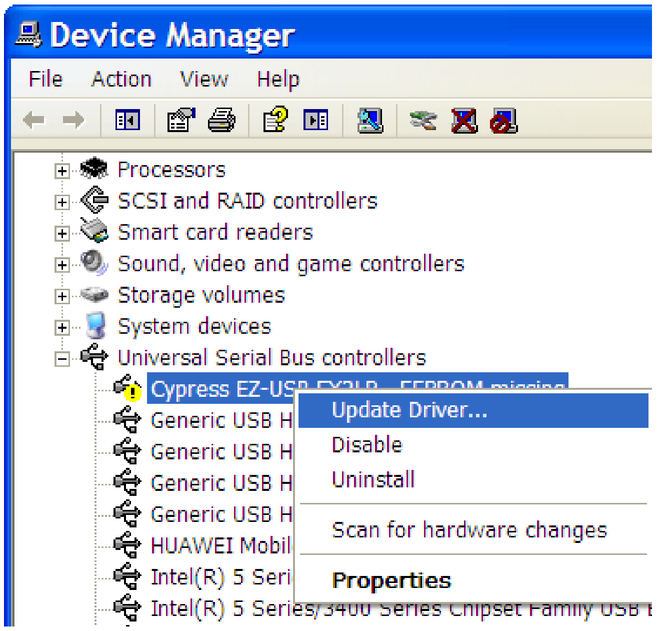
Figure 10. Adding a String to Default VID and PID of FX2LP in cyusb.inf



```
ErrorControl    = 1                    ; SERVICE_ERROR_NORMAL
ServiceBinary   = %10%\System32\Drivers\CYUSB.sys
AddReg          = CYUSB.AddReg
LoadOrderGroup = Base

[CYUSB.Files.Ext]
CYUSB.sys

[CYUSB.AddReg.Guid]
HKR,,DriverGUID,,%CYUSB.GUID%

[Strings]
CYUSB_Provider     = "Cypress"
CYUSB_Company      = "Cypress Semiconductor Corporation"
CYUSB_Description = "Cypress Generic USB Driver"
CYUSB_DisplayName = "Cypress USB Generic"
CYUSB_Install      = "Cypress CYUSB Driver Installation Disk"
VID_XXXX&PID_XXXX.DeviceDesc="Cypress USB Generic Driver (3.4.7.000)"
VID_0x04B4&PID_0x8613.DeviceDesc="Cypress EZ-USB FX2LP"
CYUSB.GUID="{AE18AA60-7F6A-11d4-97DD-00010229B959}"
CYUSB_Unused       = "."
```

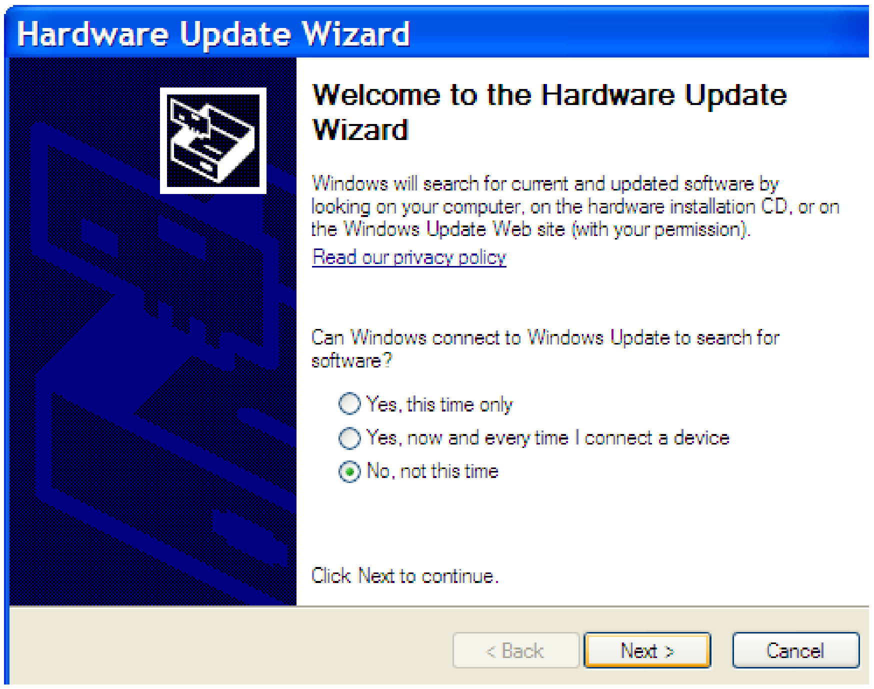
3. Right click on the FX2LP device and do “Update Driver”.

Figure 11. Updating the Driver for FX2LP



4. We need to point to the location of *cyusb.inf* file that comes with the SuiteUSB installation. Following snapshots guide you to bind the *cyusb.sys* driver file to the FX2LP device.

Figure 12. Updating the Driver for FX2LP



Welcome to the Hardware Update Wizard

Windows will search for current and updated software by looking on your computer, on the hardware installation CD, or on the Windows Update Web site (with your permission).

Read our privacy policy

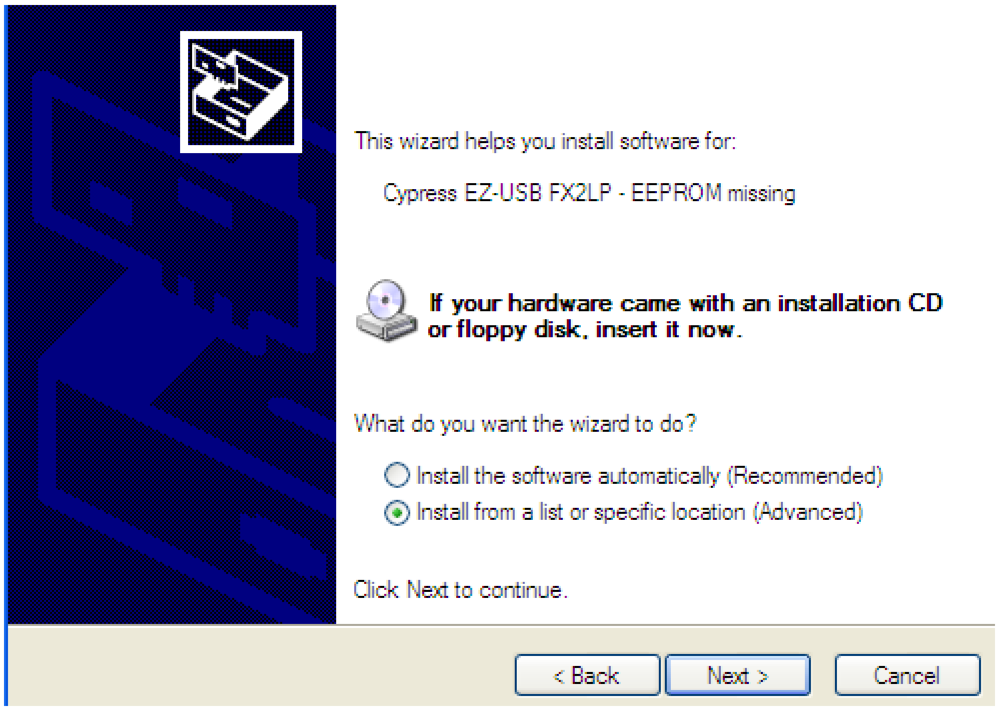
Can Windows connect to Windows Update to search for software?

- Yes, this time only
- Yes, now and every time I connect a device
- [✓ No, not this time]

Click Next to continue.

Check “**No, not this time**” and click “**Next**”.

Figure 13. Updating the Driver for FX2LP



This wizard helps you install software for:

Cypress EZ-USB FX2LP - EEPROM missing

If your hardware came with an installation CD or floppy disk, insert it now.

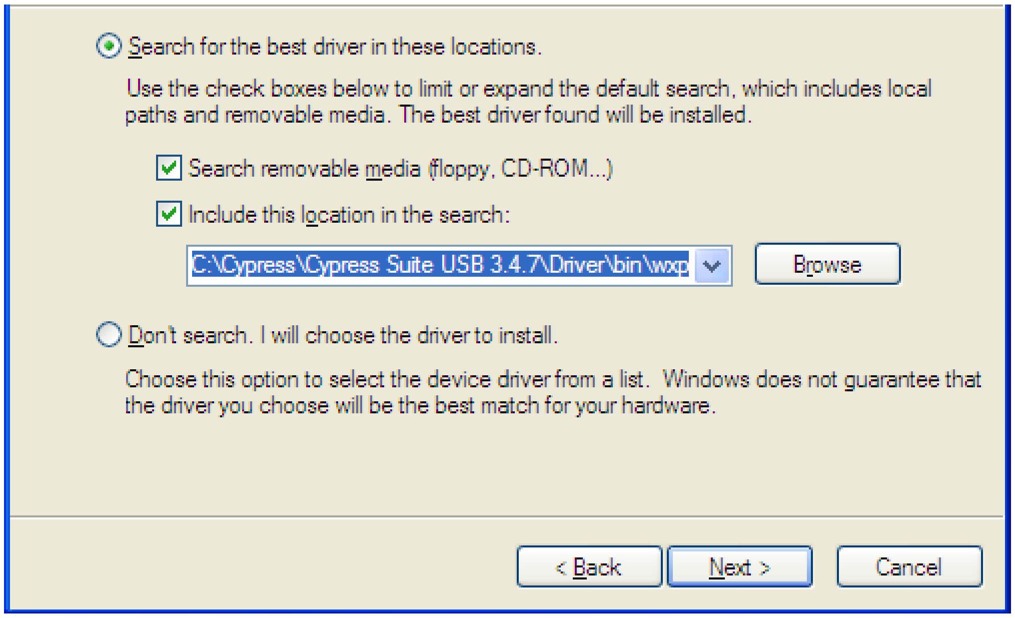
What do you want the wizard to do?

( ) Install the software automatically (Recommended)
[✓ Install from a list or specific location (Advanced)]

Click Next to continue.

Check "**Install from a list or specific location**" and click "**Next**".

Figure 14. Navigating to the Location of cyusb.inf File



[✓ Search for the best driver in these locations.]

Use the check boxes below to limit or expand the default search, which includes local paths and removable media. The best driver found will be installed.

[✓] Search removable media (floppy, CD-ROM...)

[✓] Include this location in the search:

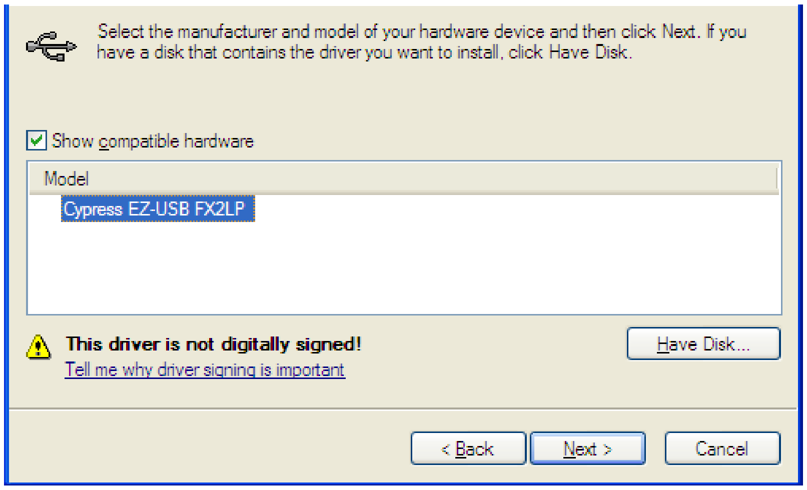
C:\Cypress\Cypress Suite USB 3.4.7\Driver\bin\wxp

( ) Don't search. I will choose the driver to install.

Choose this option to select the device driver from a list. Windows does not guarantee that the driver you choose will be the best match for your hardware.

Browse to the location where *cyusb.inf* exists (C:\Cypress\Cypress Suite USB 3.4.7\Driver\bin\wxp\x86) and click "**Next**".

Figure 15. FX2LP Device with the Name Given in the cyusb.inf

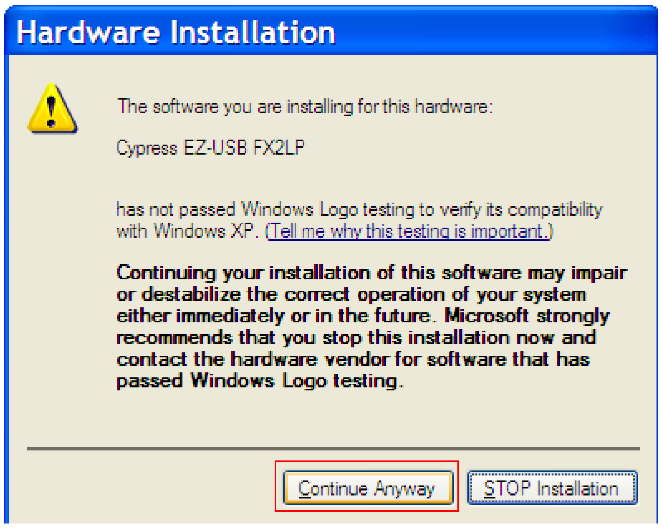


Select the manufacturer and model of your hardware device and then click Next. If you have a disk that contains the driver you want to install, click Have Disk.

✓ Show compatible hardware
Model
Cypress EZ-USB FX2LP
This driver is not digitally signed!
Tell me why driver signing is important
Have Disk...
< Back
Next >
Cancel

Here you can identify the name of FX2LP device that you have mentioned in the *cyusb.inf* file. Click "**Next**".

Figure 16. Pop-up Message Regarding Windows Logo Testing of cyusb.sys



Hardware Installation

The software you are installing for this hardware:

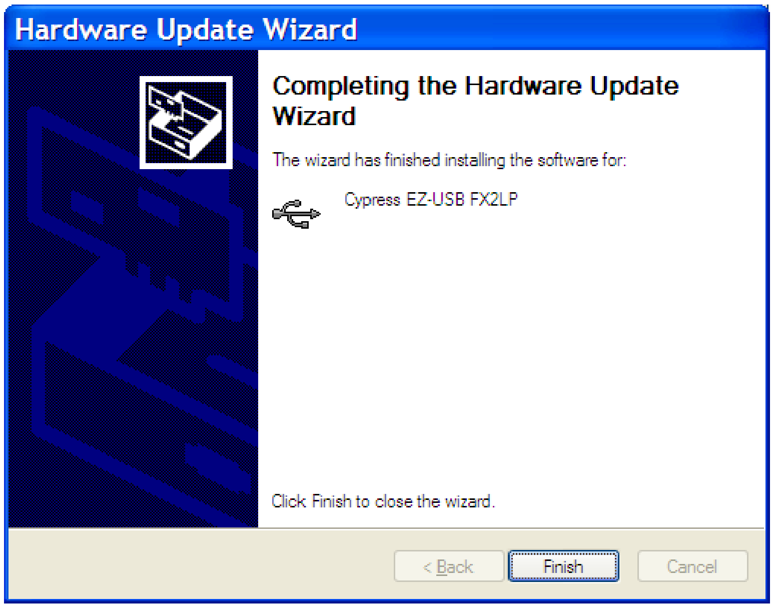
Cypress EZ-USB FX2LP

has not passed Windows Logo testing to verify its compatibility with Windows XP. (Tell me why this testing is important.)

Continuing your installation of this software may impair or destabilize the correct operation of your system either immediately or in the future. Microsoft strongly recommends that you stop this installation now and contact the hardware vendor for software that has passed Windows Logo testing.

Click "**Continue Anyway**"

Figure 17. Final Step of Binding cyusb.sys to FX2LP



Completing the Hardware Update Wizard

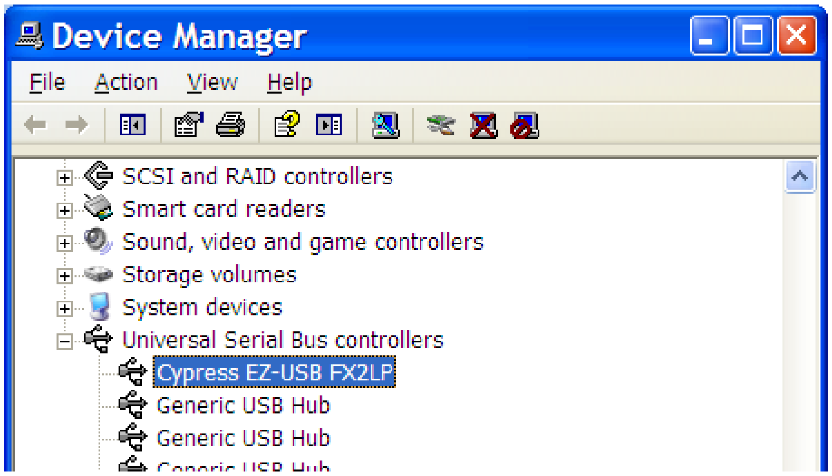
The wizard has finished installing the software for:

Cypress EZ-USB FX2LP

Click Finish to close the wizard.

Click "**Finish**". Now the connected FX2LP device is bound to the generic USB driver provided by the Cypress and it can be found in the "**Device Manager**" as shown in the following figure:

Figure 18. Device Manager After FX2LP is Bound to cyusb.sys

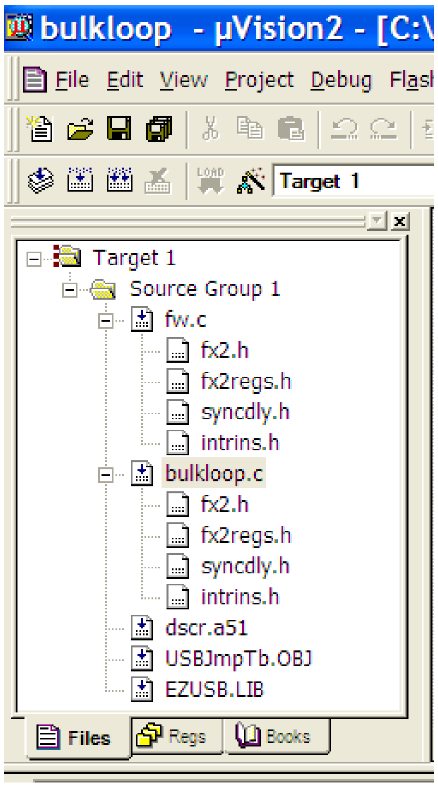


SCSI and RAID controllers
Smart card readers
Sound, video and game controllers
Storage volumes
System devices
Universal Serial Bus controllers
Cypress EZ-USB FX2LP
Generic USB Hub
Generic USB Hub

Steps to Build an Example Project and Downloading the *.hex* file into FX2LP's RAM

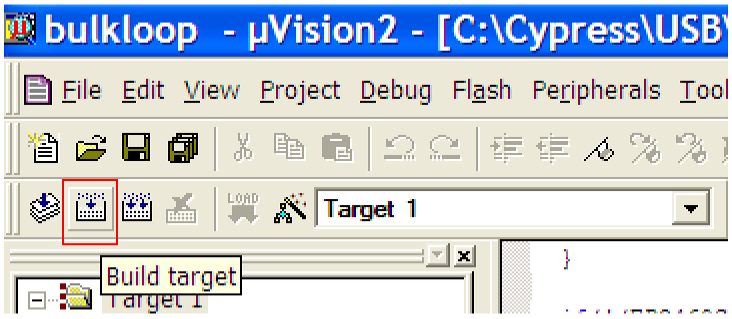
Open Bulkloop example project in the "firmware" folder of the FX2LP DVK installation, using Keil IDE. The source files exist in this project can be found in the following figure:

Figure 19. Bulkloop Project Files Opened Using Keil IDE



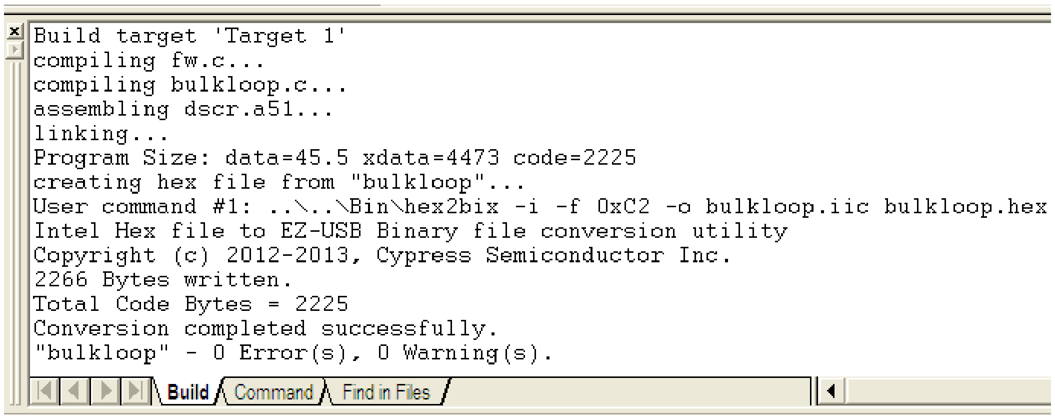
Click on "**Build Target**"

Figure 20. Build Target Button on Keil IDE


bulkloop - µVision2 - [C:\Cypress\USB
File Edit View Project Debug Flash Peripherals Tool
Target 1
Build target
Target 1

Bulklloop.hex will be generated as a result of this build and this *.hex* file can be found in the same folder. We are also running a user command from the Keil compiler. This command uses the hex2bix utility to convert the *.hex* file into *.iic* file. More information about the hex2bix utility used for this conversion is available in the AN45197 - Using the Hex2bix Conversion Utility document. Following snapshot shows the output of your build:

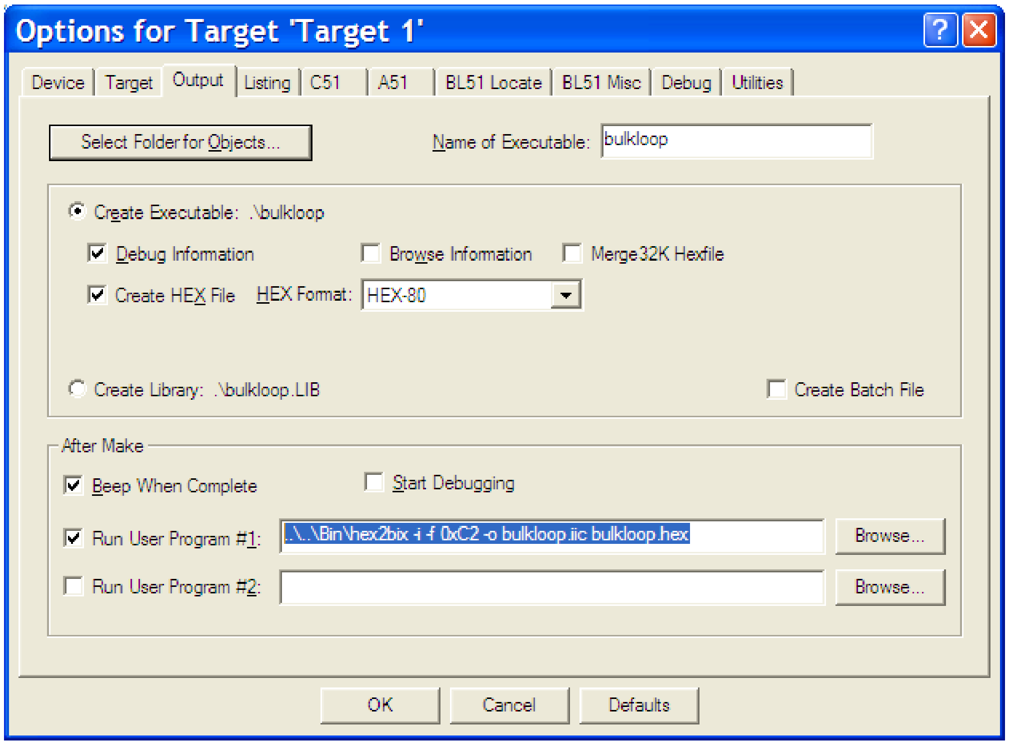
Figure 21. Build Output of Bulkloop Project Using Keil IDE



```
Build target 'Target 1'
compiling fw.c...
compiling bulkloop.c...
assembling dscr.a51...
linking...
Program Size: data=45.5 xdata=4473 code=2225
creating hex file from "bulkloop"...
User command #1: ..\..\Bin\hex2bix -i -f 0xC2 -o bulkloop.iic bulkloop.hex
Intel Hex file to EZ-USB Binary file conversion utility
Copyright (c) 2012-2013, Cypress Semiconductor Inc.
2266 Bytes written.
Total Code Bytes = 2225
Conversion completed successfully.
"bulkloop" - 0 Error(s), 0 Warning(s).
```

User command can be found in the “output” tab of “Options for Target”.

Figure 22. User Command to Convert a *.hex* File to *.iic* File



Options for Target 'Target 1'

Device | Target | Output | Listing | C51 | A51 | BL51 Locate | BL51 Misc | Debug | Utilities

Select Folder for Objects...   Name of Executable: bulkloop

(●) Create Executable: .\bulkloop

[✓] Debug Information   [ ] Browse Information   [ ] Merge32K Hexfile

[✓] Create HEX File   HEX Format: HEX-80

( ) Create Library: .\bulkloop.LIB   [ ] Create Batch File

After Make

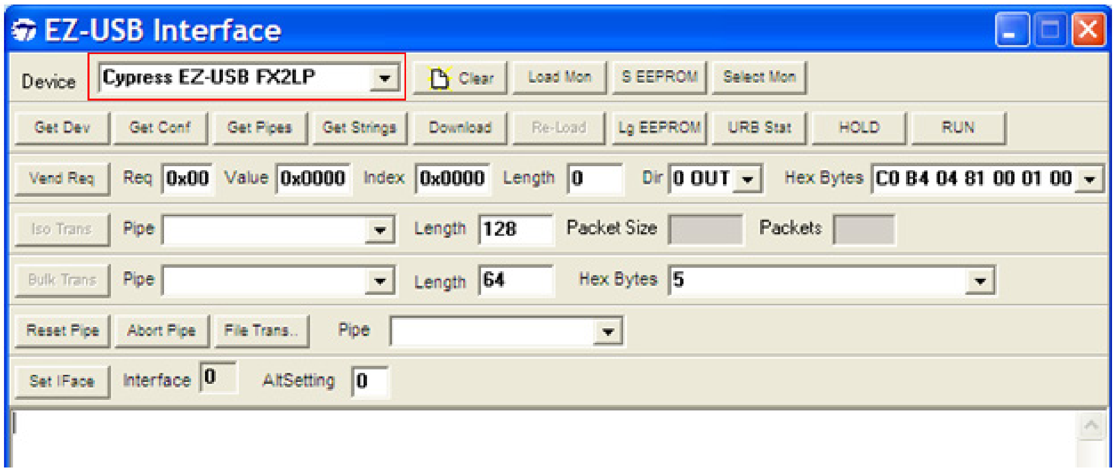
[✓] Beep When Complete   [ ] Start Debugging

[✓] Run User Program #1: ..\..\Bin\hex2bix -i -f 0xC2 -o bulkloop.iic bulkloop.hex   Browse...

[ ] Run User Program #2:   Browse...

Open **CyConsole**. You can observe the FX2LP device getting listed in “**Device**” tab.

Figure 23. CyConsole Listing the Connected FX2LP Device



Device Cypress EZ-USB FX2LP

Req 0x00 Value 0x0000 Index 0x0000 Length 0 Dir 0 OUT Hex Bytes C0 B4 04 81 00 01 00

Pipe Length 128 Packet Size Packets

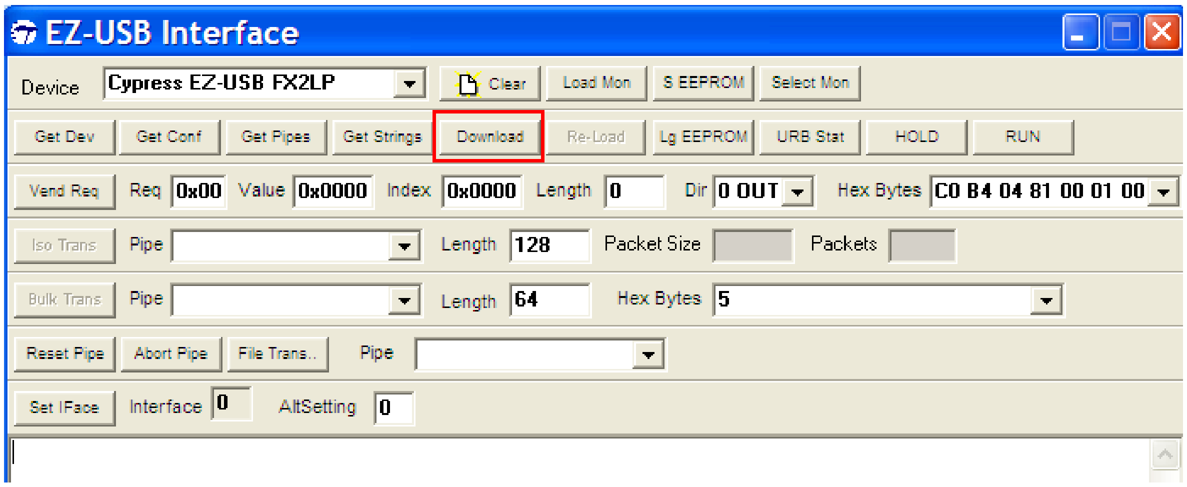
Pipe Length 64 Hex Bytes 5

Pipe

Interface 0 AltSetting 0

Using "**Download**" tab of CyConsole we can download the *.hex* file into FX2LP RAM.

Figure 24. Download Option on CyConsole



Device Cypress EZ-USB FX2LP

Req 0x00 Value 0x0000 Index 0x0000 Length 0 Dir 0 OUT Hex Bytes C0 B4 04 81 00 01 00

Pipe Length 128 Packet Size Packets

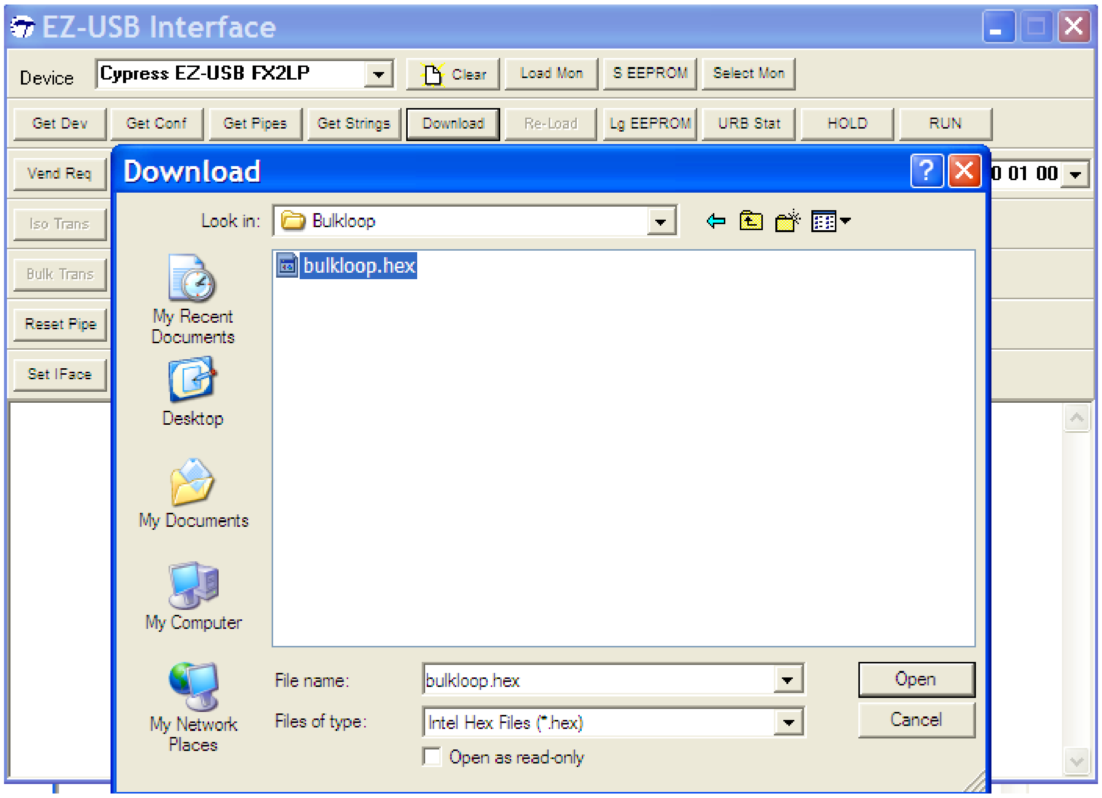
Pipe Length 64 Hex Bytes 5

Pipe

Interface 0 AltSetting 0

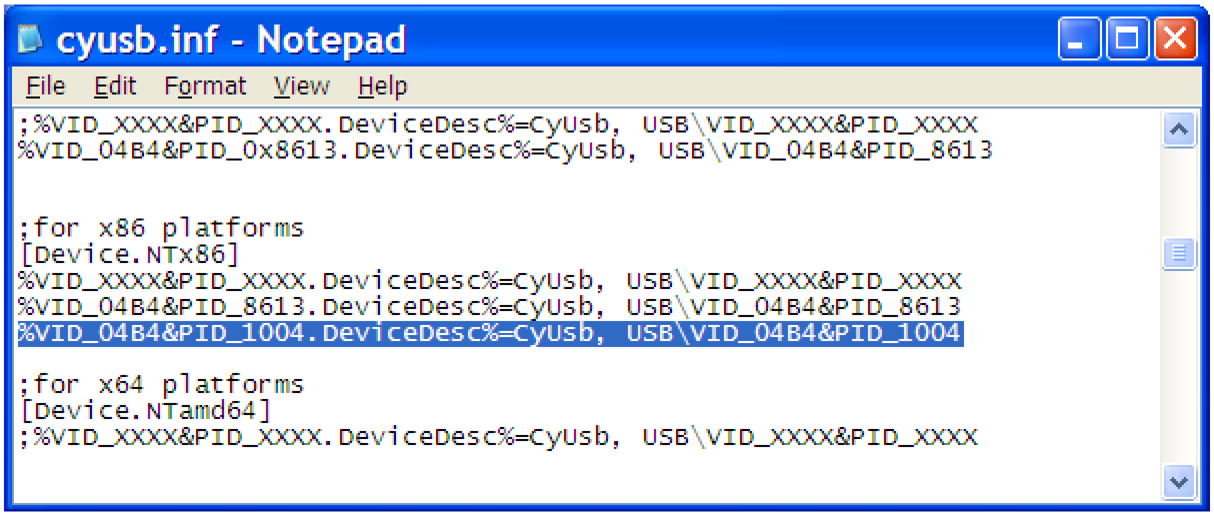
Click "**Download**" and browse the *Bulkloop.hex* file.

Figure 25. Navigate to the *bulkloop.hex*



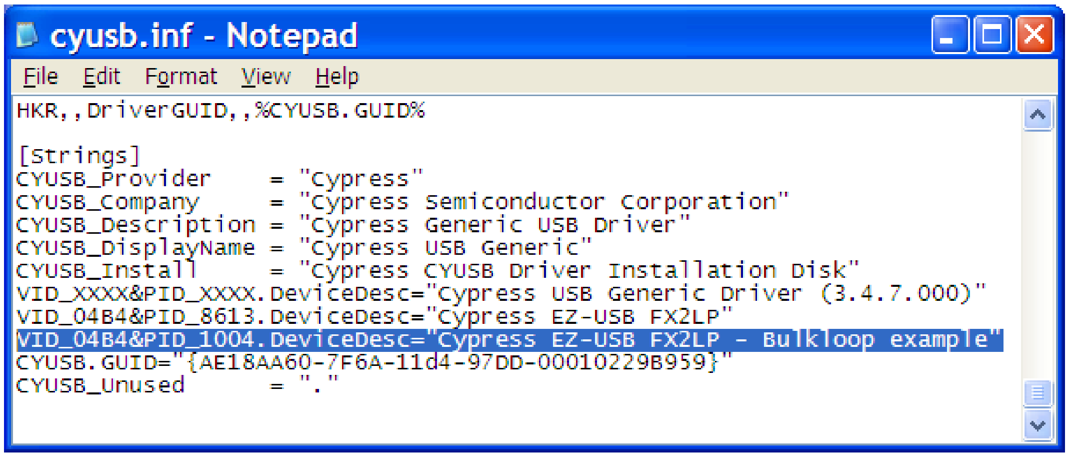
Please make sure that the *cyusb.inf* file has listed with the VID and PID values that are mentioned in the *dscr.a51* file of Bulkloop project. Here in my case, FX2LP enumerates with VID 0x04B4 and PID 0x1004.

Figure 26. Adding the VID and PID of Bulkloop Project to *cyusb.inf*



```
;%VID_XXXX&PID_XXXX.DeviceDesc%=CyUsb, USB\VID_XXXX&PID_XXXX
%VID_04B4&PID_0x8613.DeviceDesc%=CyUsb, USB\VID_04B4&PID_8613


;for x86 platforms
[Device.NTx86]
%VID_XXXX&PID_XXXX.DeviceDesc%=CyUsb, USB\VID_XXXX&PID_XXXX
%VID_04B4&PID_8613.DeviceDesc%=CyUsb, USB\VID_04B4&PID_8613
%VID_04B4&PID_1004.DeviceDesc%=CyUsb, USB\VID_04B4&PID_1004

;for x64 platforms
[Device.NTamd64]
;%VID_XXXX&PID_XXXX.DeviceDesc%=CyUsb, USB\VID_XXXX&PID_XXXX
```

Figure 27. Adding a String to VID and PID of Bulkloop Project in *cyusb.inf*

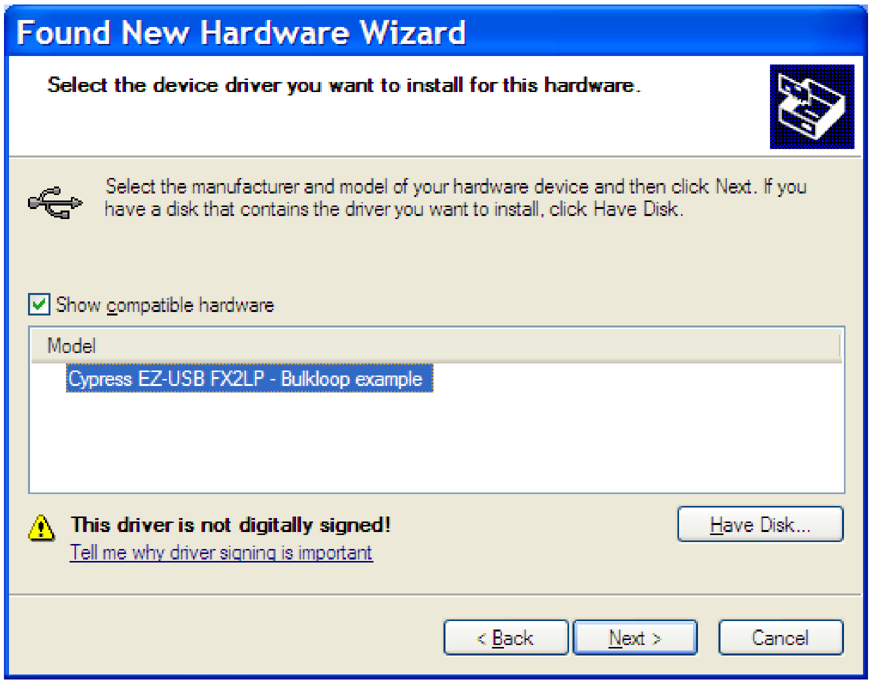


cyusb.inf - Notepad

```
HKR,,DriverGUID,,%CYUSB.GUID%

[Strings]
CYUSB_Provider     = "Cypress"
CYUSB_Company      = "Cypress Semiconductor Corporation"
CYUSB_Description  = "Cypress Generic USB Driver"
CYUSB_DisplayName  = "Cypress USB Generic"
CYUSB_Install      = "Cypress CYUSB Driver Installation Disk"
VID_XXXX&PID_XXXX.DeviceDesc="Cypress USB Generic Driver (3.4.7.000)"
VID_04B4&PID_8613.DeviceDesc="Cypress EZ-USB FX2LP"
VID_04B4&PID_1004.DeviceDesc="Cypress EZ-USB FX2LP - Bulkloop example"
CYUSB.GUID="{AE18AA60-7F6A-11d4-97DD-00010229B959}"
CYUSB_Unused       = "."
```

Point to the modified *cyusb.inf* file when FX2LP re-enumerates with VID and PID provided in the Bulkloop example project. Following is the snapshot of FX2LP DVK after binding it with the modified *cyusb.inf*.

Figure 28. FX2LP Device with the Name Given in the *cyusb.inf*



Found New Hardware Wizard

Select the device driver you want to install for this hardware.

Select the manufacturer and model of your hardware device and then click Next. If you have a disk that contains the driver you want to install, click Have Disk.

Show compatible hardware ✓

Model

Cypress EZ-USB FX2LP - Bulkloop example

This driver is not digitally signed!

Tell me why driver signing is important

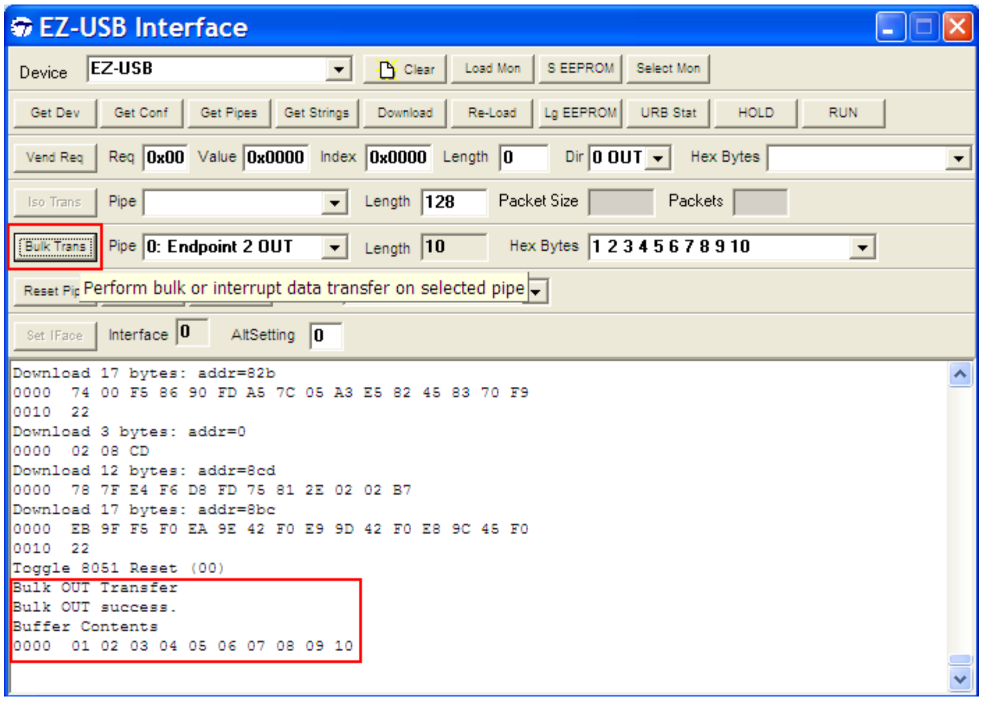
1. Bulkloop example uses total four endpoints.
   - Endpoints 2 and 4 are configured as BULK OUT endpoints.
   - Endpoints 6 and 8 are configured as BULK IN endpoints.
2. Data is looped back from endpoint 2 to 6 and 4 to 8.

Following snapshot shows you the transfer of 10 bytes data to Endpoint 2 using the CyConsole. “Bulk Trans” button available on CyConsole is used to perform bulk or interrupt data transfer on selected pipe.

The data transferred to EP2 is as follows: 1 2 3 4 5 6 7 8 9 10

You can observe the Bulk OUT tranfer getting sucessful in the message window of CyConsole.

Figure 29. Bulk OUT Transfer Using Bulk Trans Button Available on CyConsole



Req 0x00 Value 0x0000 Index 0x0000 Length 0 Dir 0 OUT

Iso Trans Pipe Length 128

Bulk Trans Pipe 0: Endpoint 2 OUT Length 10 Hex Bytes 1 2 3 4 5 6 7 8 9 10

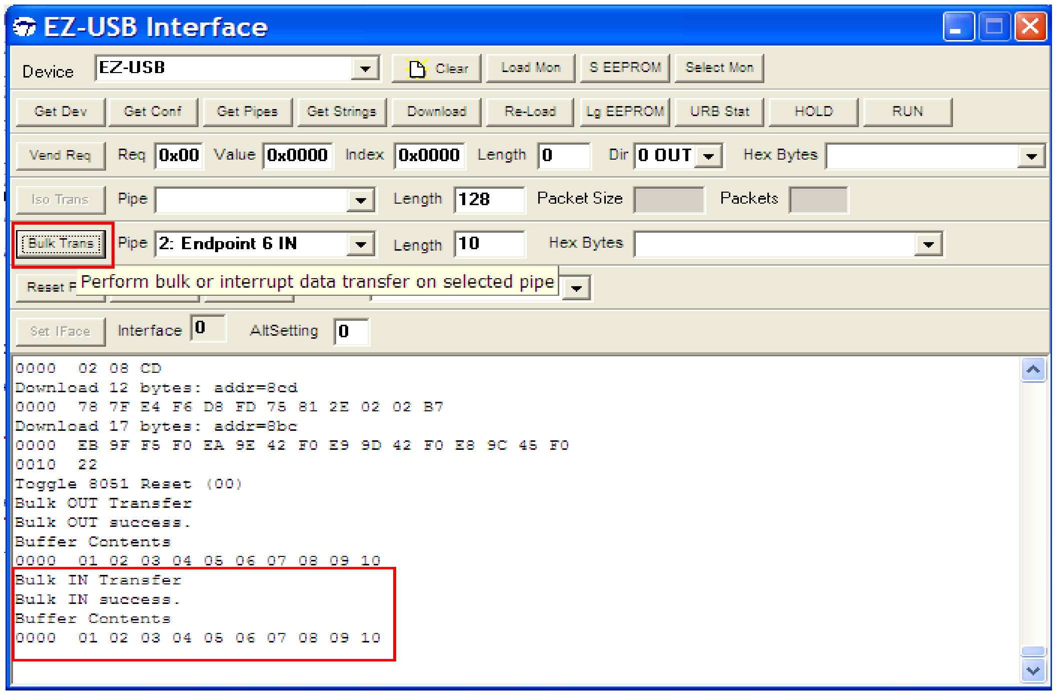
Perform bulk or interrupt data transfer on selected pipe

Interface 0 AltSetting 0

```
Download 17 bytes: addr=82b
0000  74 00 F5 86 90 FD A5 7C 05 A3 E5 82 45 83 70 F9
0010  22
Download 3 bytes: addr=0
0000  02 08 CD
Download 12 bytes: addr=8cd
0000  78 7F E4 F6 D8 FD 75 81 2E 02 02 B7
Download 17 bytes: addr=8bc
0000  EB 9F F5 F0 EA 9E 42 F0 E9 9D 42 F0 E8 9C 45 F0
0010  22
Toggle 8051 Reset (00)
Bulk OUT Transfer
Bulk OUT success.
Buffer Contents
0000  01 02 03 04 05 06 07 08 09 10
```

Perform BULK IN transfer on Endpoint 6 to read back the data that we sent earlier. Click “Bulk Trans” button on CyConsole by selecting the “Endpoint 6 IN” in the pipe tab. You can observe the same data getting received over endpoint 6 with the help of message window of CyConsole. Following snapshot shows you the received data over endpoint 6.

Figure 30. Bulk IN Transfer Using Bulk Trans Button Available on CyConsole



Req 0x00 Value 0x0000 Index 0x0000 Length 0 Dir 0 OUT

Iso Trans Pipe Length 128

Bulk Trans Pipe 2: Endpoint 6 IN Length 10 Hex Bytes

Perform bulk or interrupt data transfer on selected pipe

Interface 0 AltSetting 0

```
0000  02 08 CD
Download 12 bytes: addr=8cd
0000  78 7F E4 F6 D8 FD 75 81 2E 02 02 B7
Download 17 bytes: addr=8bc
0000  EB 9F F5 F0 EA 9E 42 F0 E9 9D 42 F0 E8 9C 45 F0
0010  22
Toggle 8051 Reset (00)
Bulk OUT Transfer
Bulk OUT success.
Buffer Contents
0000  01 02 03 04 05 06 07 08 09 10
Bulk IN Transfer
Bulk IN success.
Buffer Contents
0000  01 02 03 04 05 06 07 08 09 10
```

For more details on building your project and programming the FX2LP, refer to eighth chapter of CY3684 DVK Kit_Guide.

Debugging the Example Program

The Keil uVision2 IDE supplied with the kit enables user to debug the firmware example. Using Keil debug monitor program and UART ports (SIO-0 and SIO-1) on EZ-USB development board the firmware examples are debugged. For knowing the procedure to debug the firmware, refer to section 8.14 of the CY3684 DVK Kit_Guide document.

More information on debugging with the Keil, is available in the application note '*AN42499 - Setting up, using, and troubleshooting the Keil debugger environment*'.

In cases where debugging must happen along with USB enumeration, the USB responses from the device are not fast enough if the code is stepped through and, therefore, enumeration might fail. In such cases, the technique of debug messages can be used. These debug messages can be embedded in firmware and can be routed to the PC through its serial port. More information on this technique is available in the application note, '*AN58009 - Serial Port Debugging using FX2LP*'.

Documentation with the DVK

The CY3684 FX2LP Development Kit contains two Getting Started guides – CY3684 DVK Kit_Quickstart_Guide and CY3684 DVK Kit_Guide. These guides also come as a part of the "Documentation" directory of the kit. They illustrate the DVK contents and contain instructions on how to use the development kit.

CY3684 DVK Kit_Quickstart_Guide

This document discusses the development kit contents, how to install the kit, and start communication between the development board provided with the kit and the host PC. The document also explains the development board contents and the initial board configuration procedure.

Note It is recommended that you read this guide before installing or using the kit.

CY3684 DVK Kit_Guide

The EZ-USB Development Kit User Guide describes the different software components and tools contained in the DVK.

It describes, in detail, the "firmware framework" supplied by Cypress as a part of development kit.

Reference Designs

Several reference designs of FX2LP for popular applications are available. The reference designs include the demonstration source code, reference schematics, and a BOM, where appropriate, for the design.

The reference designs available on the Cypress website are:

- CY4661 - External USB Hard Disk Drives (HDD) with Fingerprint Authentication Security.

  The CY4661 reference design kit from Cypress and UPEK provides customers with a turnkey solution for external USB hard disk drive (HDD), with fingerprint authentication, security for conveniently protecting data and enabling authentication services. The reference design uses UPEK's TouchStrip Fingerprint Authentication Solution (TCS3 swipe fingerprint sensor and TCD42 security ASIC), the only match-on-chip fingerprint authentication solution available on the market.

- FX2LP DMB-T/H TV Dongle reference design.

  This reference design kit is based on Cypress FX2LP and Legend Silicon's chipset. In this captured and demodulated RF signal information in the form of MPEG2 TS stream is sent to the PC through FX2LP which acts as a high speed USB interface to the PC. On the PC these streams are played through media player. It is a complete design, with all the required files included.

Third Party Development Kits and SDKs

FPGA + FX2LP board from ZTEX:

More details of these boards can be found in the following location:

http://www.ztex.de/usb-fpga-1/

Features of one such board from ZTEX:

- Cypress CY7C68013A EZ-USB FX2LP Microcontroller
- High-Speed (480 MBit/s) USB interface
- Xilinx Spartan 3 XC3S400 FPGA
- 60 General Purpose I/O's (GPIO)
- 20 special I/O's (SIO)
- 128 KBit EEPROM (e.g,. for Firmware)
- Flash memory (optionally)

FPGA + FX2LP board from Opal Kelly:

More details of this board can be found in the following location:

http://www.opalkelly.com/products/xem6010/

Features:

- High-speed USB 2.0 interface (Cypress FX2LP - CY68013A) for downloading and control

- Xilinx Spartan-6 (XC6SLX45-2FGG or XC6SLX150-2FGG)
- 32-Mib serial flash (Numonyx M32P25)
- 128-MiByte DDR2 (Micron MT47H64M16HR)
- Small form-factor -- smaller than a credit card at 75 mm x 50 mm x 15.9 mm (2.95" x 1.97" x 0.63")
- Self-powered by external DC source
- Multi-PLL, multi-output clock generator (Cypress CY22393).

Third party SDKs

ZTEX provides a SDK which works with FX2LP based boards and also provides JAVA based APIs to help development of the host software.

For more details, please visit http://www.ztex.de/firmware-kit/

Related Documentation

Technical Reference Manual

EZ-USB Technical Reference Manual

This document explains the various blocks present in the FX2LP and also it has description and usage of all registers. This document is available as part of the DVK documents.

Application Notes

- AN65209 - Getting Started with FX2LP

This application note presents the features and resources available to speed up the EZ-USB FX2LP.-based design from concept to production. This document serves as a starting point for the new user to get familiar with FX2LP. It also gives an overview of the design resources available

- AN1168 - High-speed USB PCB Layout Recommendations

This application note details guidelines for designing, controlled-impedance; high-speed USB printed circuit boards to comply with the USB specification. This note is applicable to all Cypress high-speed USB solutions. Some Cypress high-speed USB chips have separate application notes that address chip-specific PCB design guidelines

- AN15456 - Guide to Successful EZ-USB® FX2LP™ and EZ-USB FX1™ Hardware Design and Debug

This application note outlines a process that isolates many of the most likely causes of EZ-USB FX2LP, and EZ-USB FX1 hardware problems. It also facilitates the process of catching potential problems before building a board and assists in the debugging when getting a board up and running.

- AN064 - EZ-USB FX2LP/AT2LP. Reset and Power Considerations

Many designers have had difficulty with the reset and power needs of the FX2LP and the USB specification. This Application Note addresses the main areas where USB and FX2LP designs have special needs. Both these chips (FX2LP, AT2LP) have similar power and reset needs. This application note refers to the FX2LP, but is also applicable to AT2LP.

- AN5078 - EZ-USB Hardware - Design considerations for EEPROM usage

EZ-USB downloads firmware automatically into the on-chip RAM from the EEPROM connected to it. The purpose of this application note is to present recommended design guidelines for assuring the data integrity of serial EEPROM devices when used in EZ-USB designs.

- AN50963 - Firmware Download Methods to FX1/FX2LP

This application note discusses the various methods to download firmware in to FX1/FX2LP.

- AN66806 - EZ-USB® FX2LP™ GPIF Design Guide

This application note helps you in teaching the steps to develop GPIF waveforms using the GPIF designer.

- AN61345 - Implementing an FX2LP™- FPGA Interface

This application note provides a sample project to interface an FX2LP™ with FPGA. An FX2LP™-FPGA interface is implemented to add High-Speed USB connectivity for FPGA based applications, such as data acquisition, industrial control and monitoring, and image processing. The FX2LP acts in Slave-FIFO mode and the FPGA acts as the master. This Application Note also gives a sample FX2LP firmware for Slave-FIFO implementation and a sample VHDL and Verilog project for FPGA implementation.

- AN58009 - Serial (UART) Port Debugging of FX1/FX2LP Firmware

This application note describes the code needed in the FX2LP firmware for serial debugging. This code enables the developer to print debug messages and real time values of the required variables in the HyperTerminal of the PC or capture it in a file using the UART engine in FX2LP.

- AN42499 - Setting Up, Using, and Troubleshooting the Keil Debugger Environment

This application note is a step-by-step beginner's guide to using the Keil Debugger. This guide covers the serial cable connection from PC to SIO-1/0, the monitor code download, and required project settings. Additionally, the guidelines to start and stop a debug session, set breakpoints, step through code, and solve potential problems are considered.

- AN15813 - Monitoring the EZ-USB FX2LP VBUS

This application note explains the purpose and methods of monitoring VBUS from the upstream connector using the EZ-USB FX2LP.

- AN4067 - Endpoint FIFO Architecture of EZ-USB FX1/FX2LP

The purpose of this application note is to help the user understand the very basics of the FX1/FX2LP and get familiar with the terminologies used while describing the data flow in FX1/FX2LP. The application note addresses three modes of operation of the FX1/FX2LP, Endpoint Configuration and Multiple Buffering, Three Domains that form the basic component of the FIFO architecture, Arming and committing endpoint buffers Endpoint operation in manual vs. auto mode.

- AN4053 - Streaming Data through Isochronous/Bulk Endpoints on EZ-USBR FX2 and EZUSB FX2LP

This application note provides brief background information on what is involved while designing for a streaming application using the EZ-USB FX2 or the EZ-USB FX2LP part. It provides information on streaming data through bulk endpoints, isochronous endpoints, and high bandwidth isochronous endpoints along with pitfalls to consider and avoid while using the FX2/FX2LP for designing high-bandwidth applications.

- AN67442 - SPI Implementation Using Serial Mode-0 of EZ-USB FX2LP

This application note describes the implementation of serial peripheral interface (SPI) protocol using the FX2LP UART port in serial mode 0. This demonstration uses FX2LP as the SPI master for transferring data to and from an AT25080A EEPROM device. The example code includes functions to the Write/Read byte to and from AT25080A EEPROM.

- AN58069 - Implementing an 8-Bit Parallel MPEG2-TS Interface Using Slave FIFO Mode in FX2LP

This application note explains how to implement an 8-bit parallel MPEG2-TS interface using the Slave FIFO mode. The example code uses the EZ-USB FX2LP at the receiver end and a data generator as the source for the data stream. The hardware connections and example code are included along with this application note.

- AN58170 - Code/Memory Banking Using EZ-USB

The EZ-USB family of chips has an 8051 core. The 8051 core has a 16-bit address line and is only able to access 64 KB of memory. However, the firmware size sometimes exceeds 64 KB. This application note describes methods of overcoming this 64 KB limitation and also demonstrates the implementation of one such method.

- AN57322 - Interfacing SRAM with FX2LP over GPIF

This application note discusses how to connect Cypress SRAM CY7C1399B to FX2LP over the General Programmable Interface (GPIF). It describes how to create read and write waveforms using the GPIF Designer. This application note is also useful as a reference to connect FX2LP to other SRAMs.

- AN14558 - Implementing a SPI Interface with EZ-USB FX2LP

This application note demonstrates how to implement a SPI interface. It uses the EZ-USB FX2LP as a SPI Master and a SPI Serial EEPROM (25AA256) as a SPI slave. This example comes with a host application with which the user can access the EEPROM. The EZ-USB FX2LP firmware uses the ports mode and bit-bangs the General Purpose IOs to create the SPI interface. The hardware connection diagram and code listing is included.

- AN1193 - Using Timer Interrupt in Cypress EZ-USBR FX2LP Based Applications

This application note is aimed at helping EZ-USBR FX2LP based firmware developers use timer interrupts in their applications, by providing a framework based timer interrupt program written in C.

- AN63787 - EZ-USB® FX2LP™ GPIF and Slave FIFO Configuration Examples using 8-bit Asynchronous Interface

This application note discusses how to configure the general programmable interface (GPIF) and slave FIFO's of EZ-USB FX2LP in both manual mode and auto mode, to implement an 8-bit asynchronous parallel interface. This Application Note is tested with two FX2LP development kits connected in back-to-back setup; the first one acting in master mode and the second in slave mode.

- AN61244 - Firmware Optimization in EZ-USB

This application note describes firmware optimization methods in EZ-USB. Some of these methods are common for any processor and some specific to the 8051 core of EZ-USB.

- AN74505 – EZ USB FX2LP - Developing USB Application on MAC OS X using LIBUSB

This application note describes a host application built on the MAC OS platform that uses libusb. The host application (Cocoa Application) communicates with the BULK IN and BULK OUT endpoints of FX2LP, using the interfaces provided by the APIs of libusb. This host application implements the transfer only with devices that pass the particular VID/PID(=0x04B4/0x1004) identification.

- AN6077 - Implementing an 8-Bit Asynchronous Interface with FX2LP.

This application note discusses how to configure the general programmable interface (GPIF) and slave FIFOs of the EZ-USB FX2LP to implement an 8-bit asynchronous interface. In this example, GPIF masters the slave FIFO interface of another EZ-USB FX2LP.

- AN58764 - Implementing a Virtual COM Port in FX2LP

This application note explains how to implement a virtual COM port device using the standard Windows driver in FX2LP. This information helps in easy migration from UART to USB.

- AN45471 - Vendor Command Design Guide for the FX2LP

This application note demonstrates how you can quickly design USB vendor commands to perform specific features of products. In addition, using the Cypress CyConsole utility to issue vendor commands is also explained.

- AN023 - USB Compliance Testing Overview

This application note discusses USB Compliance Testing. This program verifies that your USB devices meets the specification and works well with other USB devices.

Summary

The FX2LP is an outstanding choice to meet your USB 2.0 high speed design requirements. To help with each step of design cycle, Cypress has put together a formidable catalogue of support collateral. We look forward to assisting you in realizing your FX2LP project from concept to production.

About the Author

Name:	Rama Sai Krishna
Title:	Application Engineer Senior
Contact:	rskv@cypress.com

Document History

Document Title: AN65209 - Getting Started with FX2LP™

Document Number: 001-65209

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	3094213	SSJO	11/24/2010	New application note.
*A	3464470	HBM	12/4/2011	Updated template according to current Cypress standards. Removed references to obsolete documents.
*B	3754718	RSKV	10/10/2012	Re-written the application note.

Worldwide Sales and Design Support

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

Products

Automotive	cypress.com/go/automotive
Clocks & Buffers	cypress.com/go/clocks
Interface	cypress.com/go/interface
Lighting & Power Control	cypress.com/go/powerpsoc cypress.com/go/plc
Memory	cypress.com/go/memory
Optical Navigation Sensors	cypress.com/go/ons
PSoC	cypress.com/go/psoc
Touch Sensing	cypress.com/go/touch
USB Controllers	cypress.com/go/usb
Wireless/RF	cypress.com/go/wireless

PSoC® Solutions

psoc.cypress.com/solutions

PSoC 1 | PSoC 3 | PSoC 5

Cypress Developer Community

Community | Forums | Blogs | Video | Training

Technical Support

cypress.com/go/support

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone : 408-943-2600
Fax : 408-943-4730
Website : www.cypress.com