CYUSB3KIT-001

EZ-USB® FX3™ Development Kit Guide

Doc. #: 001-70237 Rev. *C

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone (USA): +1.800.858.1810
Phone (Intnl): +1.408.943.2600
http://www.cypress.com

Copyrights

Flash Code Protection

Cypress products meet the specifications contained in their particular Cypress PSoC Data Sheets. Cypress believes that its family of PSoC products is one of the most secure families of its kind on the market today, regardless of how they are used. There may be methods, unknown to Cypress, that can breach the code protection features. Any of these methods, to our knowledge, would be dishonest and possibly illegal. Neither Cypress nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as 'unbreakable'.

Cypress is willing to work with the customer who is concerned about the integrity of their code. Code protection is constantly evolving. We at Cypress are committed to continuously improving the code protection features of our products.

Contents

1. Introduction **5**

1.1 Kit Contents 5
1.1.1 FX3 Software 5
1.1.2 Tools Not Included....5
1.1.3 Other Suggested Tools....6
1.2 Additional Learning Resources....6
1.3 Document History 6
1.4 Documentation Conventions....6
1.5 Abbreviations 7

2. Getting Started **8**

2.1 FX3 DVK Software Installation 8
2.2 FX3 SDK Installation....12
2.3 Rolling Back to a Previous Version of FX3 SDK....13
2.4 FX3 SDK Installation on Linux OS....13
2.5 Eclipse IDE Quick Tour....14
2.5.1 Start Eclipse 14
2.5.2 Import a Project....15
2.5.3 Tweak Eclipse IDE Settings 17
2.6 Example 1: Blinking LED and push button 18
2.6.1 FX3 Firmware Download and Debug Methods 20
2.6.2 Code Structure 33
2.7 Importing All of the Cypress Examples....34

3. Kit Operation **35**

3.1 Default Jumper Settings on DVK Board 35
3.2 Bus-Power Mode 36
3.3 Self-Power Mode 36
3.4 First Time USB Enumeration 38
3.4.1 Manual Installation of Cypress Driver 39
3.5 Boot Options 41
3.6 USB Boot 43
3.6.1 Download Firmware Image to FX3 RAM....43
3.7 I2C Boot....45
3.7.1 Download Firmware Image to I2C EEPROM 45
3.7.2 Booting from I2C EEPROM....48
3.7.3 Choosing the Correct I2C EEPROM Configuration....48
3.8 SPI Boot....50
3.8.1 Download Firmware Image to SPI Flash....50
3.8.2 Booting from SPI Flash 53
3.9 ADMUX Boot 54
3.10 USB 3.0 Enumeration 54

3.11 Example 2: Loopback of Data over Bulk Endpoints....55
3.12 Modify the Bulkloop Firmware Example....57
3.13 Example 3: Measurement of Throughput Using USB 3.0 Bulk Transfers....62

4. Hardware 65

4.1 Power Supply....65
4.2 USB Receptacle....66
4.3 Clocking Mechanism for FX3....67
4.4 GPIF II Connector....69
4.4.1 Interconnecting to the GPIF II Connector....73
4.4.2 FX3 DVK Interconnect Boards....76
4.5 Reset Circuit....76
4.6 Serial Interfaces....77
4.6.1 I2C Interface....77
4.6.2 I2S....77
4.6.3 SPI and UART....77
4.6.4 JTAG....80

A. Appendix - Troubleshooting 81

A.1 Self-Power Mode....81
A.2 Bus-Power Mode....81
A.3 USB 2.0 Enumeration....82
A.4 USB Driver Installation....82
A.5 USB Boot....83
A.6 I2C Boot....84
A.7 SPI Boot....84
A.8 USB3.0 Enumeration....85

B. Appendix - PCB Layout 86

B.1 Assembly Top....86
B.2 Assembly Bottom....87

1. Introduction

The Cypress EZ-USB® FX3™ Development Kit (DVK) is a combination of hardware, software, and documentation that enables customers to evaluate the FX3 device. This document describes how to install the software related to the FX3 DVK and operate the DVK board. A simple project to blink an LED helps verify correct installation and operation of the integrated development environment (IDE). The guide introduces different types of firmware download and debug methods with detailed instructions on how to use them. Two more example projects (USBBulkLoopAuto on page 55 and USBBulkSourceSink on page 62) serve to explain the use of software host applications provided for FX3. This document also briefly explains different hardware interfaces available on the DVK board. In addition, Appendix A provides a troubleshooting guide, which helps to isolate the root cause of errors while operating the DVK board along with a corresponding solution.

1.1 Kit Contents

The Cypress EZ-USB FX3 DVK includes the following:

- Development kit board
- USB 3.0 A to Micro B cable
- Quick Start Guide
- 5-V AC-DC adapter

Visit http://www.cypress.com/shop for more information. Inspect the contents of the kit. If any parts are missing, contact your nearest Cypress sales office for further assistance.

1.1.1 FX3 Software

- FX3 Development Kit (DVK) Installer: It installs documentation, such as user guide and release notes, and DVK hardware files, such as schematic, PCB Layout, and Gerber.
- FX3 Software Development Kit (SDK) Installer: It installs the Eclipse IDE and GCC tool chain, a firmware library with code samples, and a Cypress USB suite including a Windows driver and sample Visual Studio applications. After installation, a Cypress Update Manager insures that all modules are up to date.

1.1.2 Tools Not Included

- Microsoft Visual C++ and C# software required for editing and building USB PC application source code. Free Visual Studio Express editions are available on the Microsoft web site.
- USB 3.0 capable PC host: The FX3 DVK firmware examples can work in either USB 2.0 or USB 3.0 speeds.To achieve maximum performance with FX3 hardware, use a PC with USB 3.0 host controller ports.

1.1.3 Other Suggested Tools

You can use the following USB protocol analyzers to analyze the traffic between the PC host and the FX3 device.

- Hardware analyzers
  - Ellisys USB Explorer 280
  - Lecroy USB Voyager M3i
  - Beagle USB 5000 SuperSpeed Protocol Analyzer
- Software protocol analyzers
  - SourceQuest SourceUSB
  - SysNucleus USBTrace

1.2 Additional Learning Resources

Visit http://www.cypress.com/fx3 for additional learning resources in the form of datasheets, a technical reference manual, and application notes.

1.3 Document History

Revision	PDF Creation Date	Origin of Change	Description of Change
**	06/23/2011	MRKA	Initial version of kit guide
*A	08/11/2011	MRKA	Updated figures and table in section 2.1.4; added section 2.1.4.1. Updated Table 2-3 and Figure 2-11. Added Figure 2-12.
*B	12/22/2012	NMMA	Updated Getting Started chapter. Added the Hardware chapter. Added the Appendix chapter.
*C	03/30/2014	RSKV	Explained three firmware example projects. Added information on debugging using UART and JTAG. Added information on the GPIF II interconnect boards.

1.4 Documentation Conventions

Table 1-1. Document Conventions for Guides

Convention	Usage
`Courier New`	Displays file locations, user entered text, and source code: `C:\ ...cd\icc\`
Italics	Displays file names and reference documentation: Read about the *sourcefile.hex* file in the *PSoC Designer User Guide*.
[**Bracketed, Bold**]	Displays keyboard commands in procedures: [**Enter**] or [**Ctrl**] [**C**]
File > Open	Represents menu paths: File > Open > New Project
Bold	Displays commands, menu paths, and icon names in procedures: Click the **File** icon and then click **Open**.
Times New Roman	Displays an equation: 2 + 2 = 4
Text in gray boxes	Describes cautions or unique functionality of the product.

1.5 Abbreviations

Table 1-2. List of Abbreviations

Abbreviation	Meaning
AC	Alternating Current
ADMUX	Address Data Multiplexing
ASIC	Application-Specific Integrated Circuit
COM port	Communication Port
CTS	UART Clear To Send
DC	Direct Current
DSP	Digital Signal Processor
DVK	Development Kit
EEPROM	Electrically Erasable Programmable Read-only memory
EP	Endpoint
ESD	Electrostatic discharge
FMC	FPGA Mezzanine Card
FPGA	Field-Programmable Gate Array
GCC	GNU Compiler Collection
GDB	GNU Debugger
GPIF	General Programmable Interface
GPIO	General Purpose Input/Output
HSMC	High Speed Mezzanine Card
I^2C	Inter-integrated circuit
I^2S	Inter-IC Sound
IDE	Integrated Development Environment
KB	Kilobyte
LED	Light-emitting diode
MB	Mega Byte
OS	Operating System
OTG	USB On-The-Go
PC	Personal computer
PHY	Physical layer
PID	Product ID
RTS	UART Ready To Send
SCL	I2C Serial Clock
SDA	I2C Data
SDK	Software Development Kit
SPI	Serial Peripheral Interface
USB	Universal Serial Bus
USB-IF	Universal Serial Bus Implementers Forum
VID	Vendor ID

2. Getting Started

This chapter guides you through the installation of the Windows versions of the FX3 DVK and FX3 Software Development Kit (SDK). After installing the Eclipse-based tools, a simple example confirms that the installation is correct and introduces the Eclipse system.

2.1 FX3 DVK Software Installation

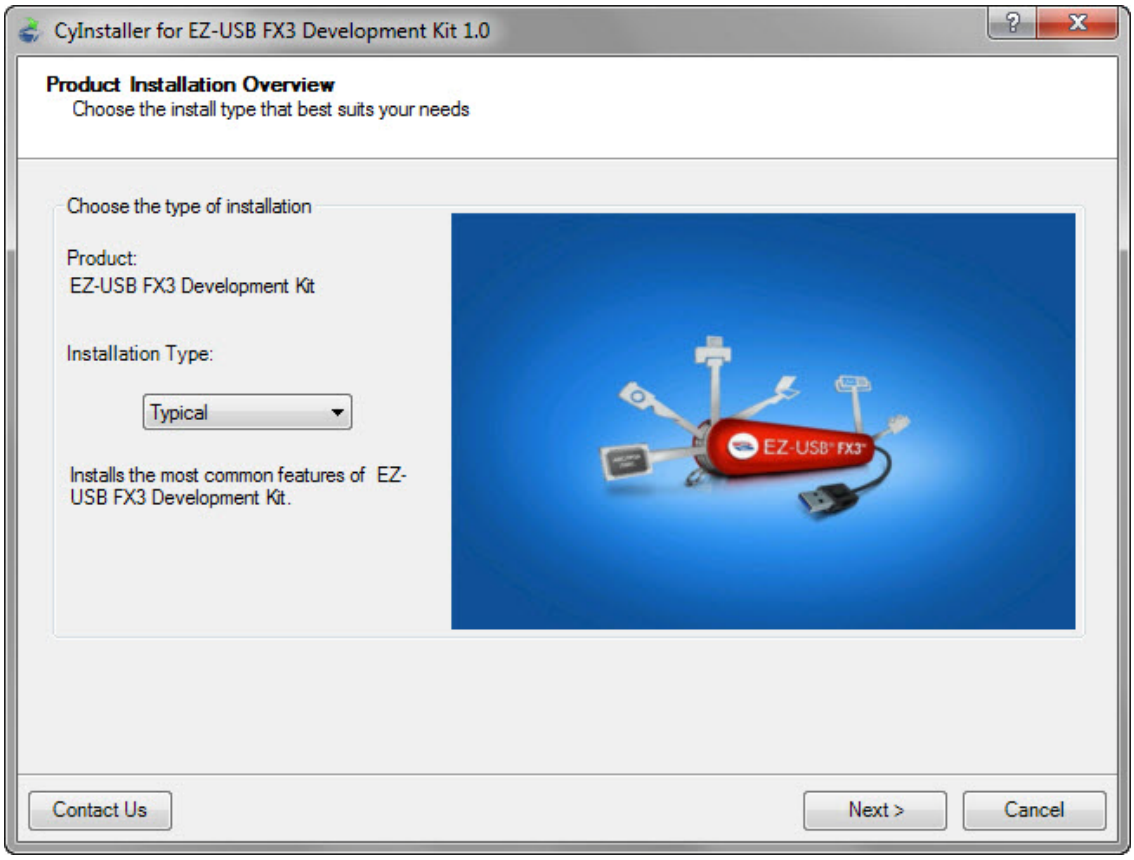
To install the kit software, follow these steps:

1. Download the latest FX3 DVK software from EZ-USB FX3 Development Kit page, which contains the kit hardware files, Kit documents, FX3 manuals, guides and SDK for Windows and Linux. If your browser does not ask to run the downloaded file, locate the *FX3DVKSetup.exe* file in your download folder and double-click it to start the Installer (Figure 2-1).

Figure 2-1. FX3 DVK Installer Screen

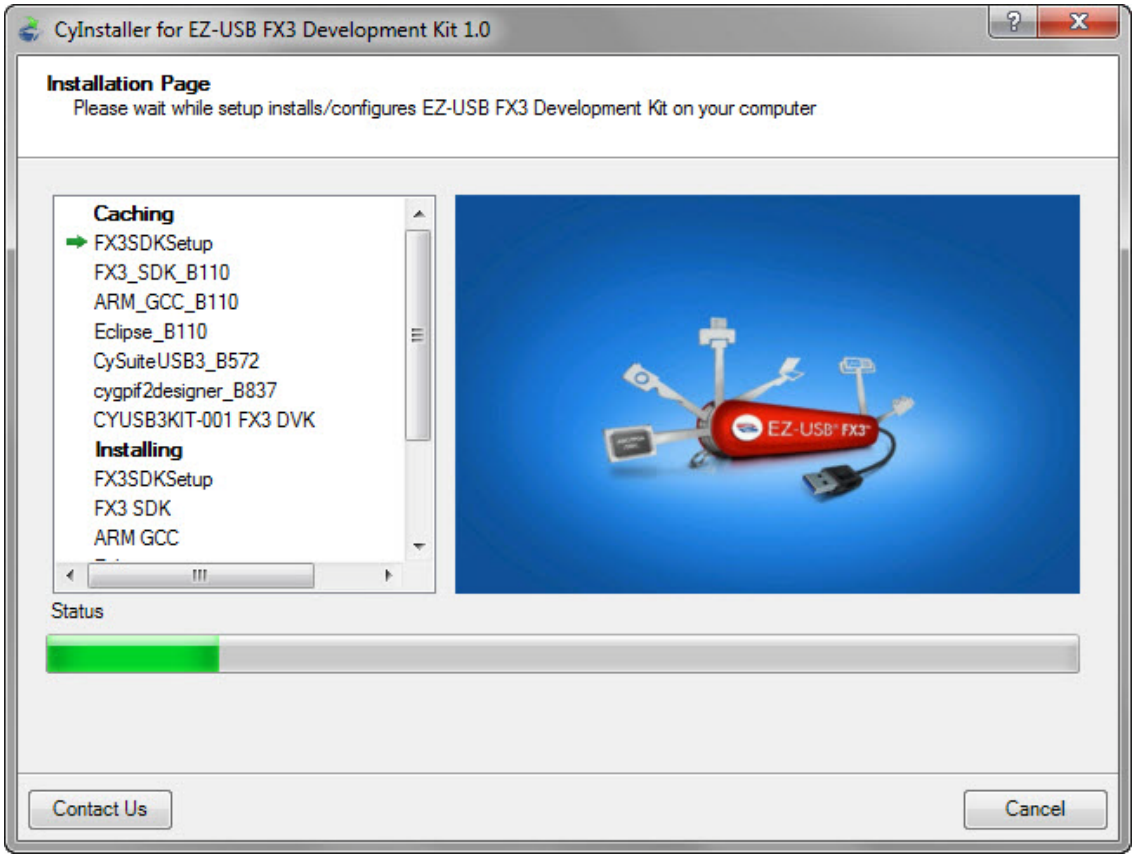
2. Select the required **Installation Type** and click the **Next** button to start the Installation Wizard (Figure 2-2). For the first time installation it is recommended to **Installation Type** as **Typical.**

Figure 2-2. Installation Wizard



3. Accept the license agreements for the various software components and click **Next** and wait till the installation is complete.

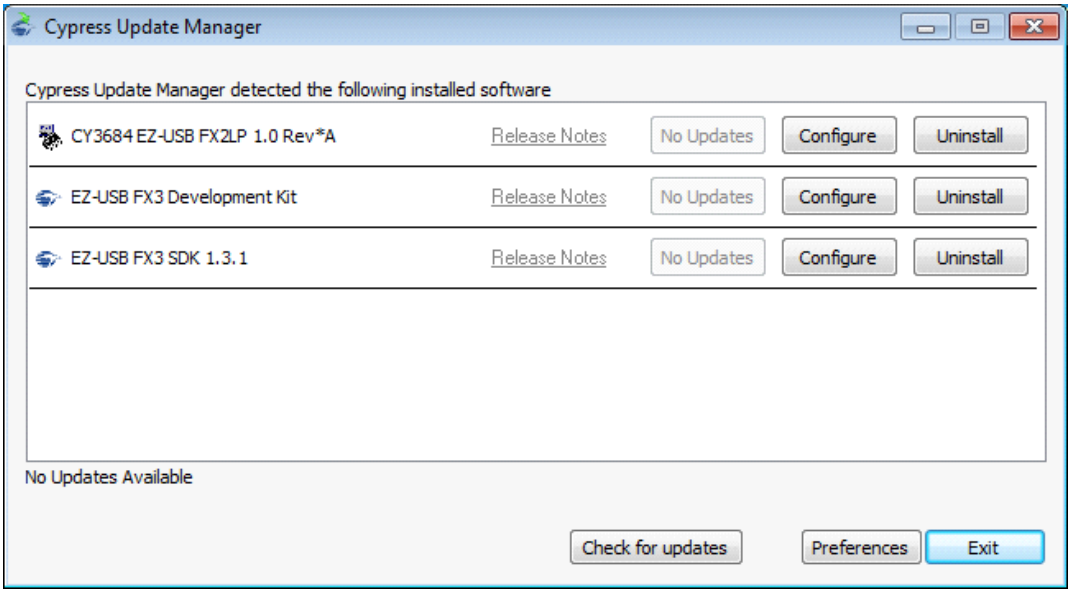
Figure 2-3. Installation Progress Showing Individual Modules



4. When installation completes you have the option to run the Cypress Update Manager (Figure 2-4), to insure you have the latest SDK. Click the **Check for Updates** button at the bottom of the **Cypress Update Manager** window. If **No Updates** appears against **EZ-USB FX3 SDK** you can click **Exit** button. If there are updates, click the **Update** button to download and install the latest SDK.

   Note: You can launch the Update Manager from the **Windows > Start > All Programs > Cypress** menu.

Figure 2-4. Cypress Update Manager



5. The installation creates two directories in the Program Files directory (Program Files (x86) for Windows 64-bit Operating System):
   a. *Cypress\EZ-USB FX3 Development Kit* contains documentation such as release notes, hardware files, schematics and PC layout files.
   b. *Cypress\EZ-USB FX3 SDK* contains the Cypress USBSuite tools, including Eclipse, and several utilities such as the USB Control Center in the bin directory (**CyControl.exe**). It also contains many Cypress firmware examples. The next section shows how to import these examples into Eclipse projects.

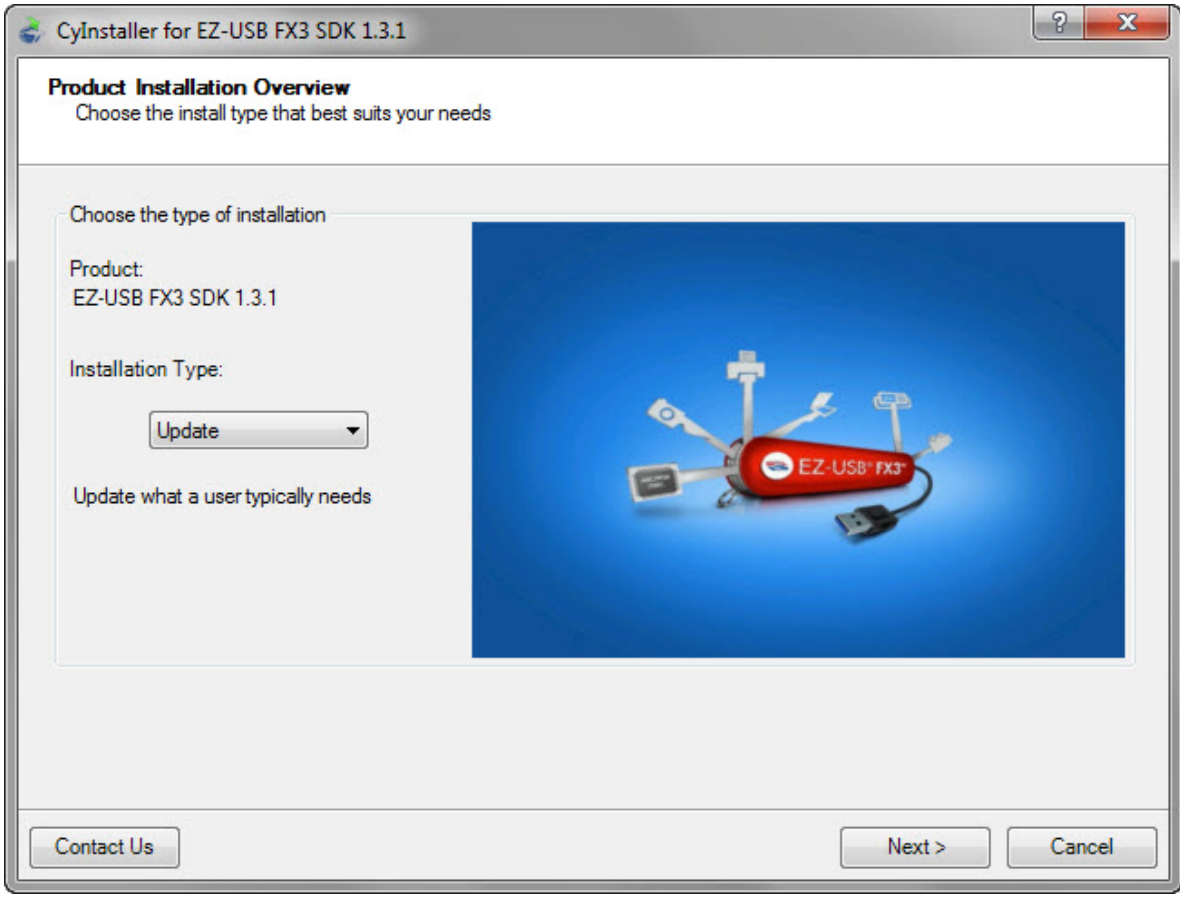
2.2 FX3 SDK Installation

Note: If you ran the **Cypress Update Manager** after installing the DVK and downloaded the latest version of the SDK you can skip the steps listed in this section.

To install the latest version of FX3 SDK, follow these steps:

1. Download the latest FX3 SDK from EZ-USB FX3 Software Development Kit page, which contains Windows and Linux downloads. If your browser does not ask to run the downloaded file, locate the *FX3SDKSetup.exe* file in your download folder and double-click it to start the Installer (Figure 2-5).

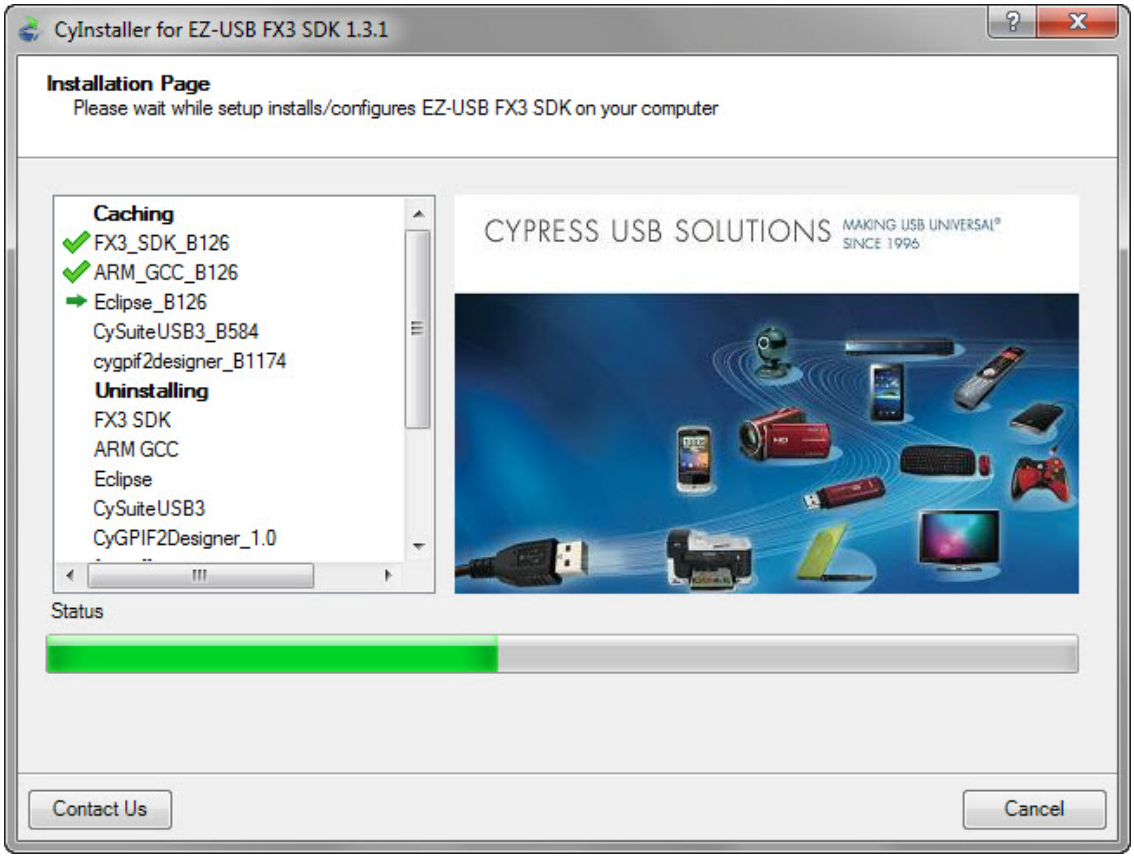
Figure 2-5. FX3 SDK Installer Screen



2. Click the **Next** button to start the Installation Wizard.

3. Accept the license agreements for the various software components and click **Next** and wait till the installation is complete.

Figure 2-6. Installation Progress Showing Individual Modules



2.3 Rolling Back to a Previous Version of FX3 SDK

You can roll back to any previous version of FX3 SDK using Cypress Update Manager. Use the steps given in the Knowledge Base article to install any previous version.

2.4 FX3 SDK Installation on Linux OS

FX3 SDK version 1.2 and later revisions support firmware development with the Eclipse IDE and debugging using the J-Link JTAG debugger probe on a Linux platform. The EZ-USB FX3 SDK for Linux is released in the form of a gzipped tar archive called "*FX3_SDK.tar.gz*". On extraction, this tar archive contains the following gzipped tar archives:

- FX3_Firmware.tar.gz: The FX3 firmware library and examples.
- ARM_GCC.tgz: Sourcery ARM GNU toolchain to compile firmware examples.
- eclipse_x86.tgz: Eclipse IDE for 32-bit Linux OS.
- eclipse_x64.tgz: Eclipse IDE for 64-bit Linux OS.
- cyusb_linux_<Build_no>.tar.gz: The CyUSB Suite provides QT based USB applications to communicate with FX3 device.

The installation procedure involves extracting these archives and setting of environment variables. Refer to the *FX3_SDK_Linux_Support.pdf* available in the following extracted folder of the FX3 SDK for Linux installation: *fx3_sdk_v1.3_linux\FX3_Firmware\cyfx3sdk\doc*.

2.5 Eclipse IDE Quick Tour

2.5.1 Start Eclipse

Navigate to **Windows > Start > All Programs**, and then click on the Cypress folder to expand it (Figure 2-7). Installed Cypress software packages appear, including **GPIF II Designer**, a utility to create design files for the FX3 General Programmable Interface (GPIF II) using state machine entry.

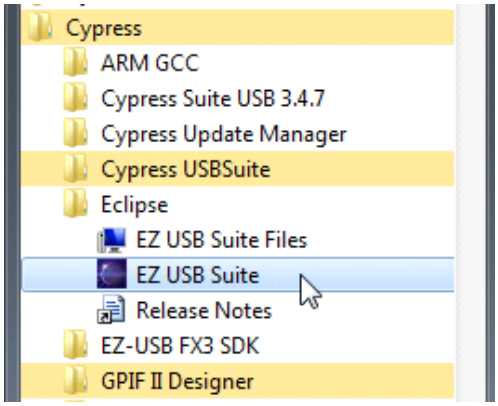
Click on the Eclipse entry to expand it, then double-click the **EZ USB Suite** entry.

Note: You can make a desktop shortcut by right-clicking **EZ USB Suite**, dragging it to the desktop, releasing the mouse button, and selecting **Create shortcuts here**.

This launches a first-time startup screen that sets up a workspace folder for all your Cypress Eclipse work. Accept the default path of *C:\Users\(username)\Cypress\Workspace*. You may want to select **Use this as the default and do not ask again** to skip this message every time you start the EZ-USB Suite.

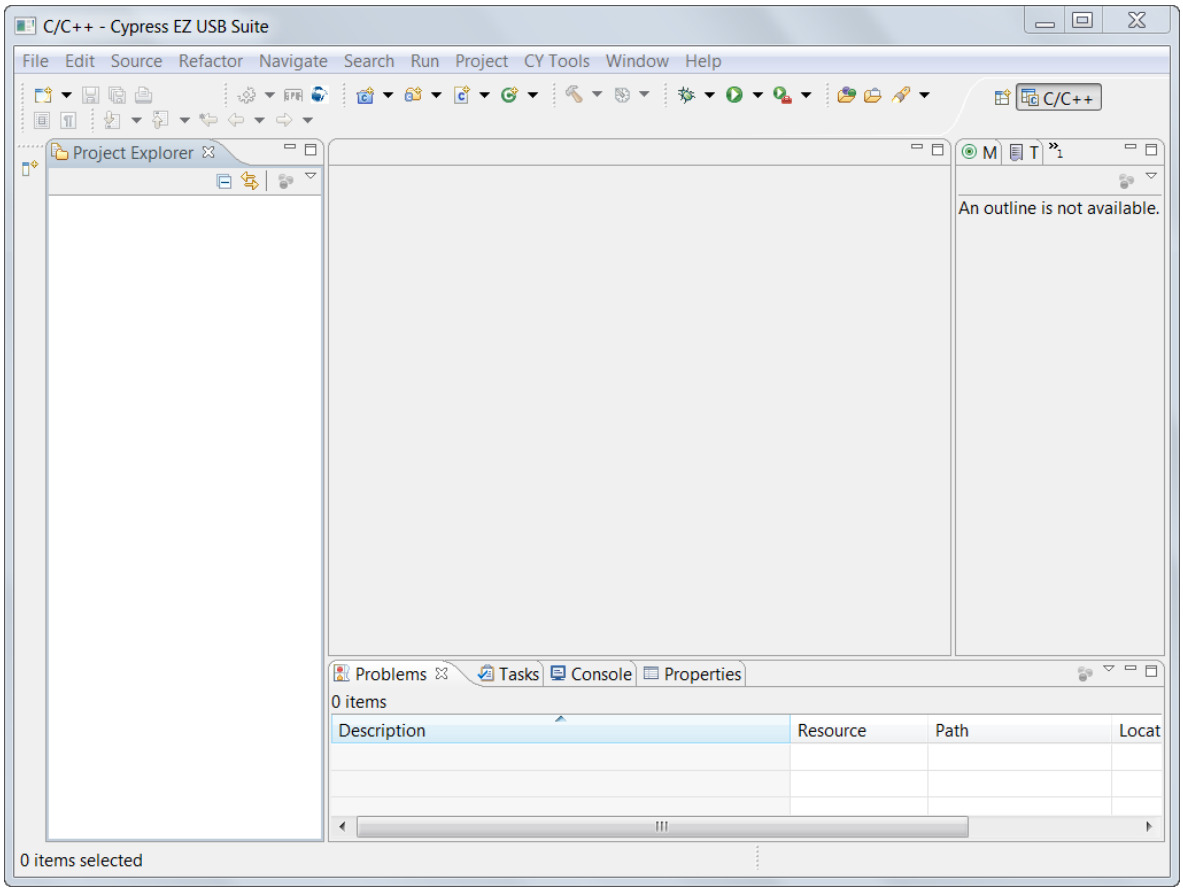
Note: You can create more workspaces by clicking **File > Switch Workspace > Other**. This can be used to define different workspaces for different projects.

Figure 2-7. Launch the EZ-USB Suite



An empty Eclipse Workbench appears as in Figure 2-8.

Figure 2-8. An Empty Eclipse Workbench



2.5.2 Import a Project

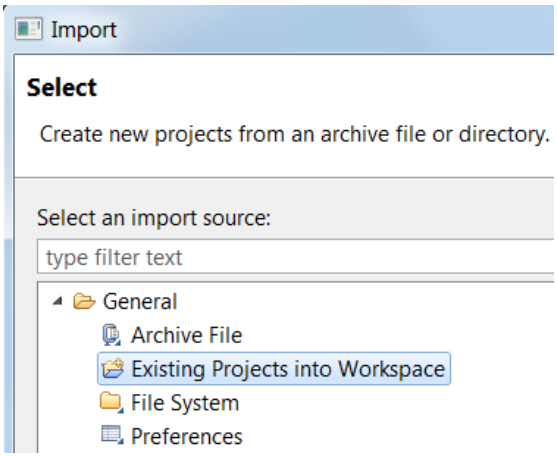
Although you can create projects from scratch, a better way to write FX3 code is to import an example project that has functionality similar to your requirements. You can use this project as a fully operational starting point for your edits. The SDK installation creates a folder full of example projects at *C:\Program Files\Cypress\EZ-USB FX3 SDK\firmware*. For 64-bit systems the first folder in the path is Program Files(x86).

Later in this chapter you will import all of the Cypress examples into your Workbench, but first it is helpful to learn how an Eclipse project is imported from anywhere, for example from another computer, an email or a colleague.

1. For this example the firmware project available on the FX3 DVK web page, First_FX3_App, is imported and tested. Download the First_FX3_App.zip and extract to a folder.

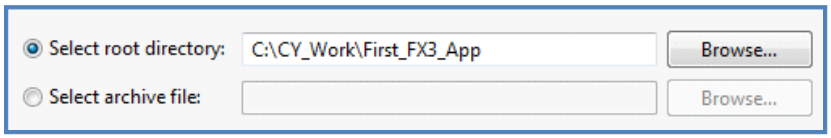
2. In the Eclipse IDE, select **File > Import > General > Existing Projects into Workspace** (Figure 2-9). Click **Next** button.

Figure 2-9. Import an Eclipse Project



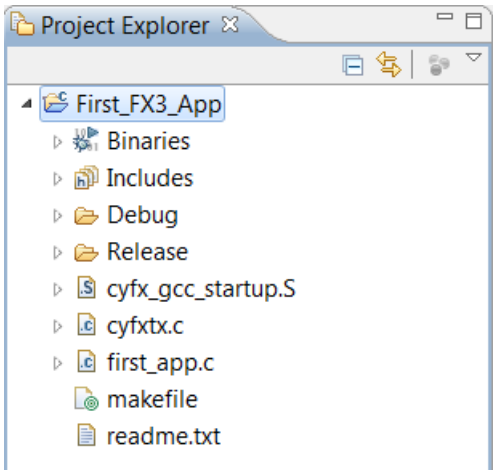
3. Select the **Browse…** button next to **Select root directory:** and browse to your **First_FX3_App** folder (Figure 2-10). If you want your own copy of the project, check the **Copy projects into workspace** and the folder will be copied to your Workspace folder. Check the firmware application in the **Projects:** box, and click the **Finish** button.

Figure 2-10. Browse to the Project Folder



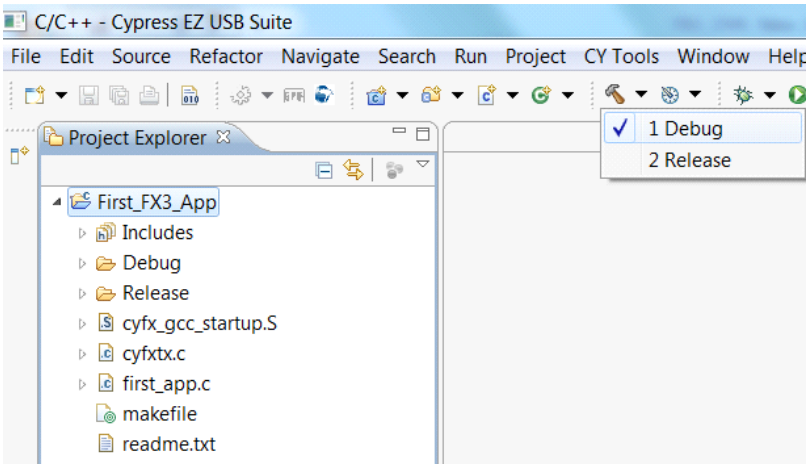
4. The added project now appears in the Eclipse **Project Explorer**. Double-click it to see the underlying components (Figure 2-11). Double-click the **first_app.c** entry and the C source file opens in the workspace editing window.

Figure 2-11. "First_FX3_App" Added to the Project Explorer



5. To build a FX3 project, select the project in the **Project Explorer** window and click the **Build** button (hammer) in the Eclipse IDE. You can choose whether you want to build a **Debug** version or a **Release** version of the firmware. The size of the generated image file in the Release version is smaller than that in the debug version.

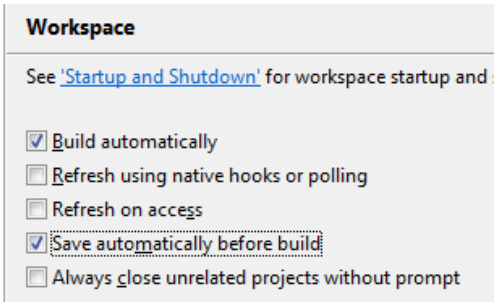
Figure 2-12. Build an FX3 project



2.5.3 Tweak Eclipse IDE Settings

Most IDE's default to automatically saving edited source files before compiling, but the Eclipse default is not to do so. If you forget to save a file after editing it before a re-compile and wonder why your edits did not "take", you will want to have Eclipse automatically save your edits before building the project. Navigate to **Windows > Preferences > General > Workspace** and select **Save automatically before build** (Figure 2-13).

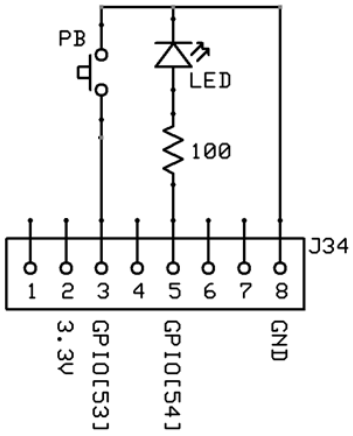
Figure 2-13. Tell Eclipse to Automatically Save



2.6 Example 1: Blinking LED and push button

The **first_app.c** code blinks an LED attached to a GPIO (General-Purpose IO) pin and reads the state of a GPIO input pin, printing its state on a Windows terminal whenever the state changes. To see the results of this application, prepare the small circuit shown in Figure 2-14. A small perf-board with a 0.1 inch header is a convenient way to build this circuit as shown in Figure 2-15. J34 is at the top center of the PCB (red highlight).

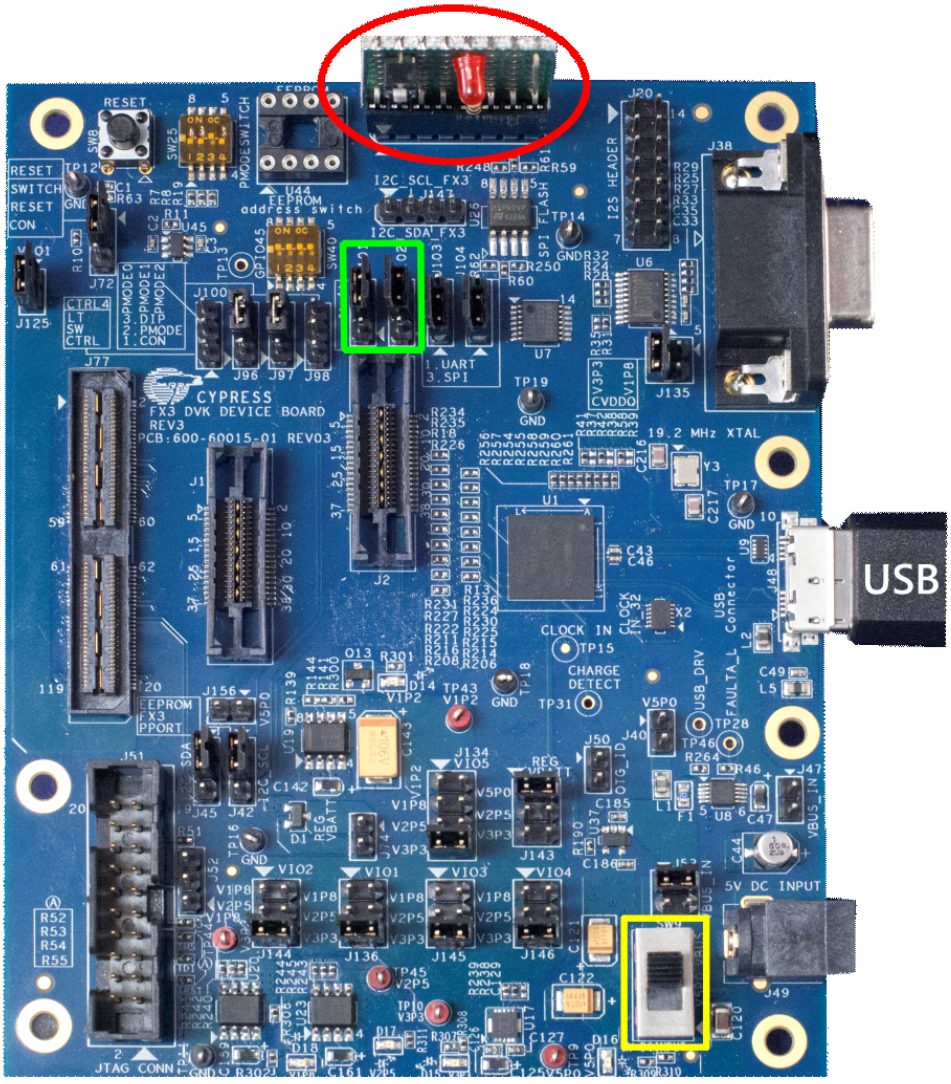
Figure 2-14. Circuit to test the FX3_First_App code



Most FX3 GPIO pins can be programmed to attach an internal pull-up or pull-down resistor to prevent floating inputs. This example uses two GPIO pins that are programmed as a group to serve as UART signals. Because the serial debugger does not use the UART CTS and RTS signals, these pins can be used for the LED and push button. There is no need for an external pull-up resistor on the GPIO[53] (UART RTS) signal since an internal pull-up resistor is enabled in the FX3 firmware.

Two jumpers need to be in place to connect the GPIO[53] and GPIO[54] signals to the J34 header: Jumpers J101 and J102 should short pins 2–3 (green rectangle in Figure 2-15). Adjacent jumpers J103 and J104 should short pins 1–2 to connect the RXD and TXD UART signals to the DB-9 connector. Jumper J53, just above the power switch (yellow rectangle) must also be in place to enable USB power to the board. Table 3-1 on page 35 gives all the FX3 jumper settings and their purposes.

Figure 2-15. FX3 SDK board with Added Circuit on J34, Jumpers, Power Switch



2.6.1 FX3 Firmware Download and Debug Methods

You can download and debug FX3 code using following methods:

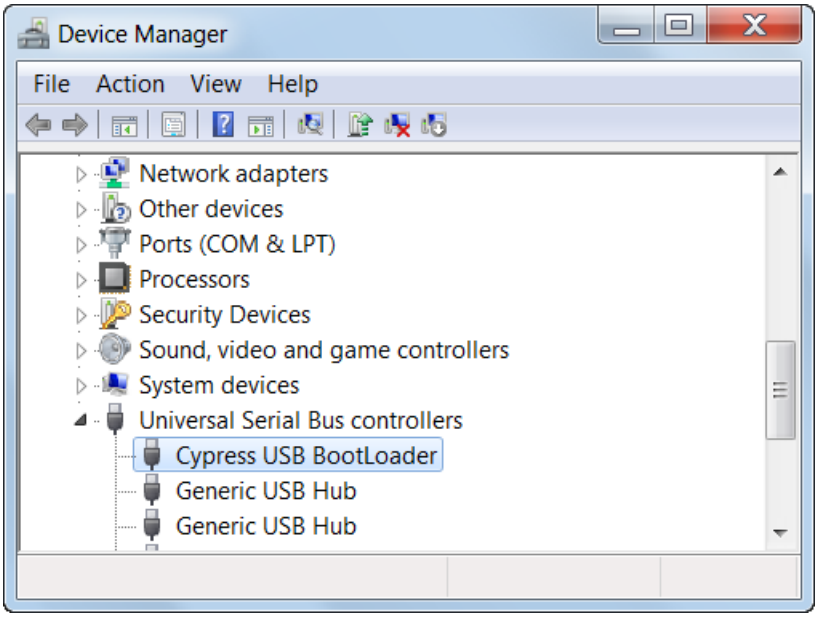
1. **Run Code:** Download a binary over USB and run it.
2. **Run Code with Serial Debug:** Download a binary over USB and debug it using print statements and a PC Terminal program.
3. **Download and debug over JTAG:** Plug in an in-circuit emulator and use it to download and debug code with full debug facility such as breakpoints and single-stepping.

2.6.1.1 Run Code

Follow these steps to verify correct operation of the example code:

1. Plug the FX3 board into a USB host port using the provided cable. You can use any speed USB port, but eventually you will want to plug into a USB 3.0 port to study and test FX3 SuperSpeed applications.
2. Move the slide switch (**SW9**) in the lower right corner of the board to the UP position. This supplies board power from USB, eliminating the need for an external power unit.
3. The board enumerates as a **Cypress BootLoader**, as seen in the **Device Manager** tree (Figure 2-16).

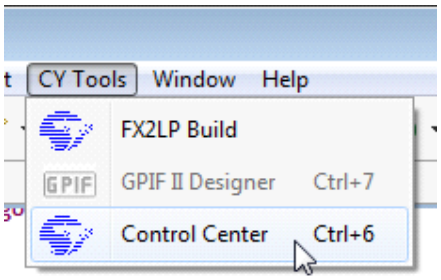
Figure 2-16. FX3 Board appears as BootLoader



If the board appears as **WestBridge** or as an **unknown device** in the Device Manager, right-click the entry, navigate to the bin folder in your EZ-USB FX3 SDK installation, and select your operating system folder. These steps are shown in Manual Installation of Cypress Driver on page 39 of this document.

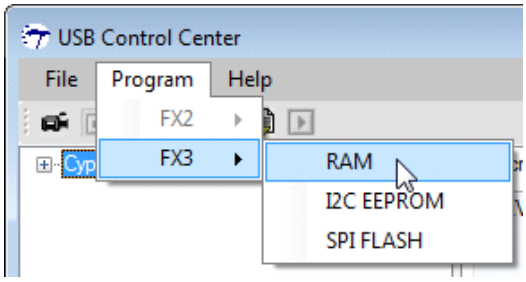
4. In the Eclipse IDE, select **CY Tools > Control Center** (Figure 2-17).

Figure 2-17. Open the Cypress USB Control Center



This tool downloads code into FX3 RAM or to a flash memory connected to the FX3 and also allows you to schedule USB transfers to a Cypress device. Highlight the **Cypress USB Boot-Loader** item, and select **Program > FX3 > RAM** (Figure 2-18). This opens a dialog to locate the download file.

Figure 2-18. Download Code into FX3



5. Navigate to the Debug folder in your workspace folder and double-click the *First_FX3_App.img* (img = code image) file. If your PC has sound turned on, you will hear the USB disconnect sound. The bootloader has loaded the code image and restarted as the First_FX3_App program. This application does not implement a USB device, so the **Cypress USB BootLoader** item disappears from the **USB Control Center** device window. Most Cypress examples implement USB devices, in which case you will see the new device name appear in place of the **Cypress USB BootLoader** device.

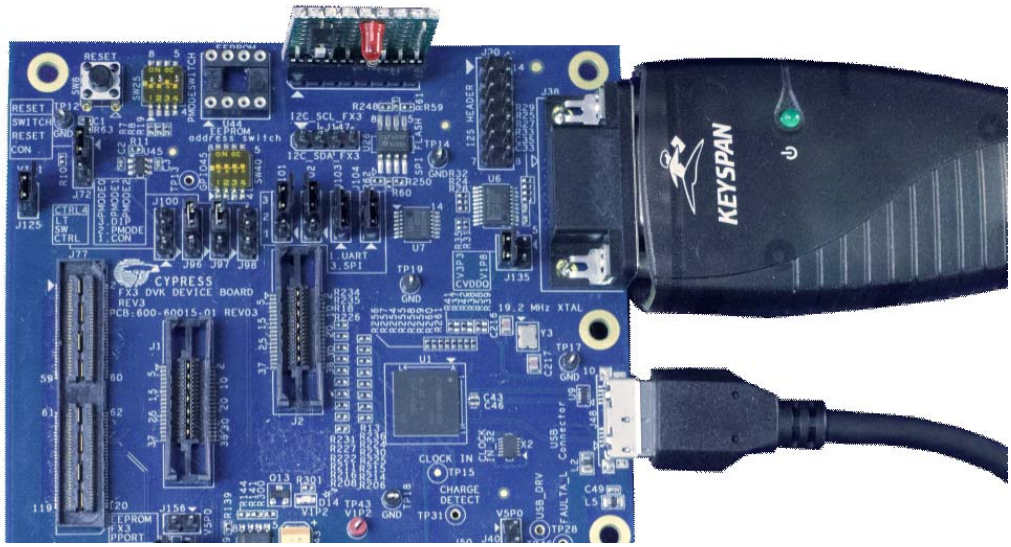
   You should see the LED blinking with a 2 second period. If an LED is not connected to GPIO[54] then you can see the output of this project by probing the GPIO pin using an oscilloscope.

   To reload any code, you first must press the RESET button in the upper left corner of the FX3 board. Try it now. You will hear the USB “connect” sound as the **Cypress USB BootLoader** device re-appears. If you highlight it and again choose **Program > FX3 RAM**, the USB Control Center code remembers your last download path. This is convenient when making and testing many code edits.

2.6.1.2 *Run Code with Serial Debug*

This debug method gives feedback from a running program via the FX3 UART port. Since modern PC's do not include serial ports, you will need a serial-to-USB converter such as the Keyspan (Tripp Lite) USA-19-HS (Figure 2-19). If you have a serial port, you can connect a DB-9 cable instead.

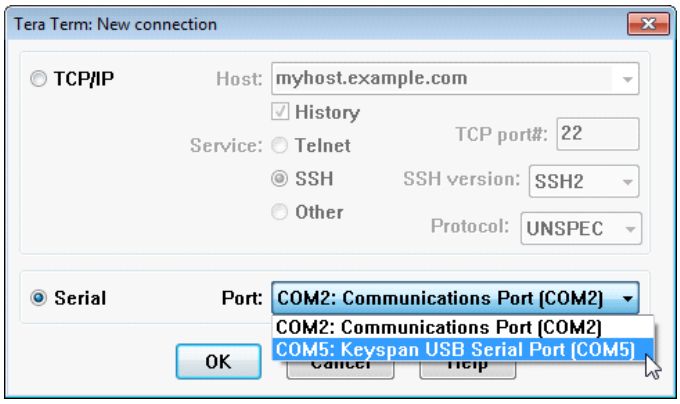
Figure 2-19. Connect a Serial-to-USB Adapter to the DB-9 Connector



To see the debug messages you need a terminal program such as the free Tera Term used in the following steps.

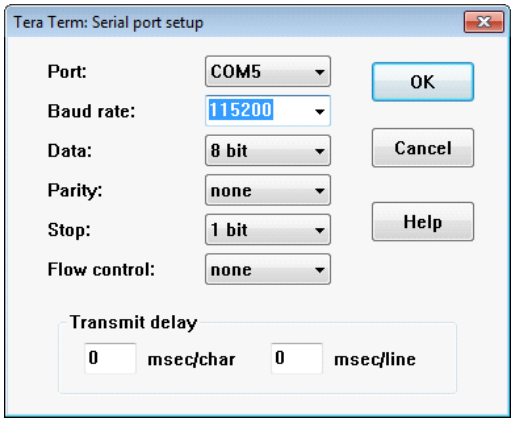
1. Plug the serial adapter USB cable into the PC, and then start Tera Term. A serial device must be plugged into the PC before starting Tera Term to be recognized.
2. Select the **Serial** option and the adapter COM port as shown in Figure 2-20.

Figure 2-20. Select the USB Converter COM port



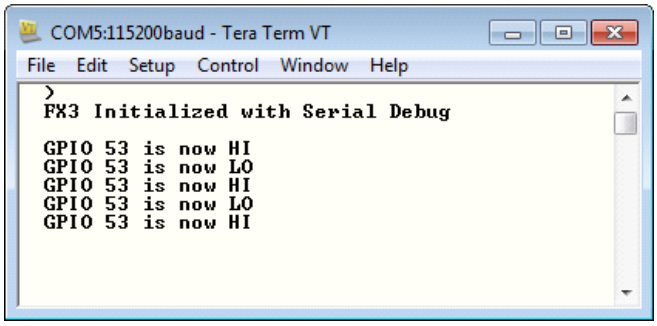
3. Choose **Setup > Serial Port** and adjust the baud rate to 115,200. Then press **OK** (Figure 2-21).

Figure 2-21. Select 115,200 baud rate



4. Load and run the code as in the first example. To recap: If the USB Control Panel is not already running, choose **CY Tools>Control Center**, press the board RESET button, choose **Program>FX3>RAM**, navigate to the .img file in your Workspace and load the program. You should see a screen similar to that shown in Figure 2-22. Every time you press the push button an interrupt-driven function displays the GPIO[53] pin state. This simple application does not de-bounce the button, so you may see several messages per button push. If you have not connected a push button to GPIO[53] then these debug messages can be observed by connecting this GPIO to ground using a jumper wire.

Figure 2-22. FX3 Sending Debug Messages



FX3 firmware sends serial debug messages using the **CyU3PDebugPrint** function, for example for the sign-on message, **CyU3PDebugPrint (4, "\r\n FX3 Initialized with Serial Debug\r\n\r\n")**;.

The first parameter sets a debug message priority. Your program can set a priority threshold to enable some messages and disable others.

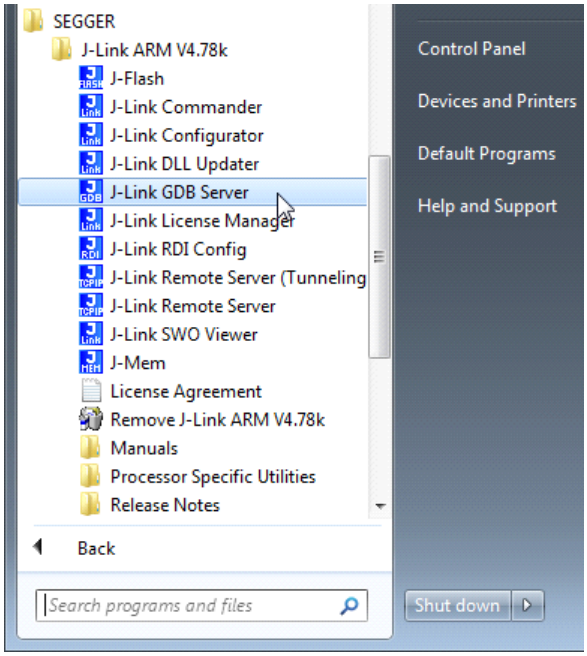
2.6.1.3 *Download and Debug over JTAG*

This debug method uses FX3 on-chip debug logic which is accessed over a JTAG (Joint Test Action Group) port. JTAG debug may coexist with serial port debug since any code can access the debug print function. To use this method, purchase a USB-to-JTAG adapter and connect it to the JTAG connector (J51) at the lower-left corner of the FX3 board (Figure 2-23).

Figure 2-23. J-Link Attaches to the FX3 Board JTAG Connector

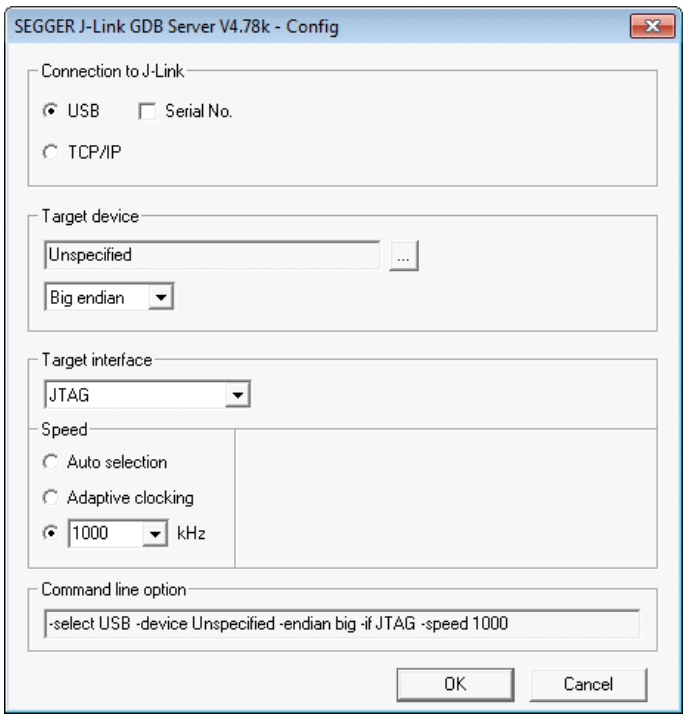
This example uses a Segger J-Link adapter along with the J-Link GDB Server program available on the Segger web site. The J-Link is available from Digi-Key, Mouser, and others. A J-Link educational version costs about US$60. Segger provides a download that includes J-Link GDB Server, accessible at **Start>All Programs>SEGGER>J-Link ARM Vxxxx** (Figure 2-24). If you want to create a desktop shortcut, right-click the item, drag it to your desktop, release the mouse button and click **Create shortcuts here**.

Figure 2-24. Link to the Segger Debugger



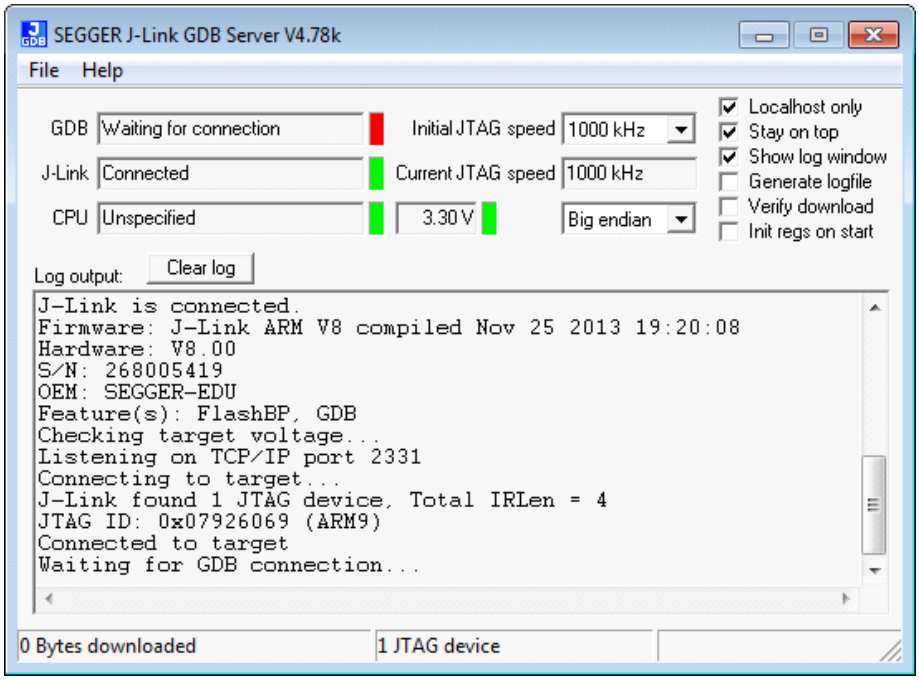
Start the Segger J-Link GDB Server (Figure 2-25). Click **OK** to accept the default settings.

Figure 2-25. Segger J-Link Startup Screen



The J-Link GDB Server screen appears as in Figure 2-26. This screen runs concurrently with your Eclipse sessions, so you can drag this window off to a side of your desktop. For now it is waiting for an Eclipse debug session to begin (**Waiting for GDB connection…**).

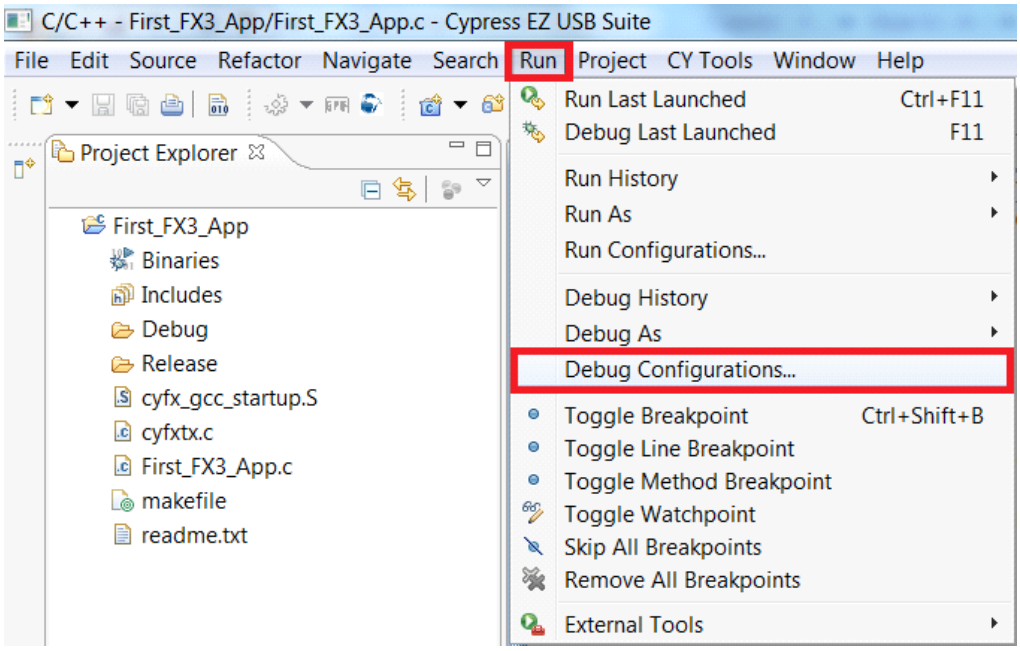
Figure 2-26. J-Link GDB Server



The following steps configure the Eclipse project to be compatible with the J-Link debugger. Some Cypress projects are already configured, so you only need to confirm the settings. Once a project is configured the settings are in force for all subsequent debug sessions.

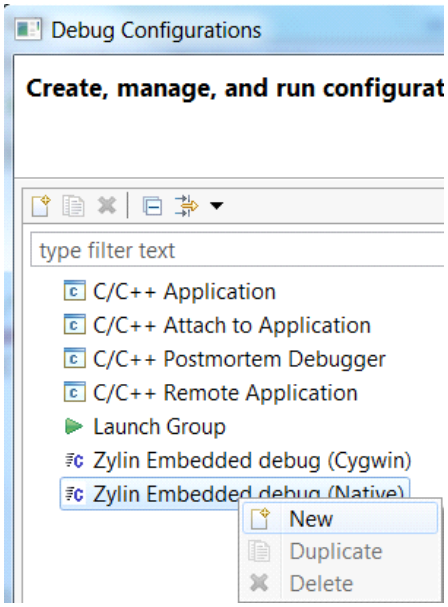
1. Select your project in the Project Explorer, then select **Run > Debug Configurations** (Figure 2-27).

Figure 2-27. Select Project and Choose Run>Debug Configurations...



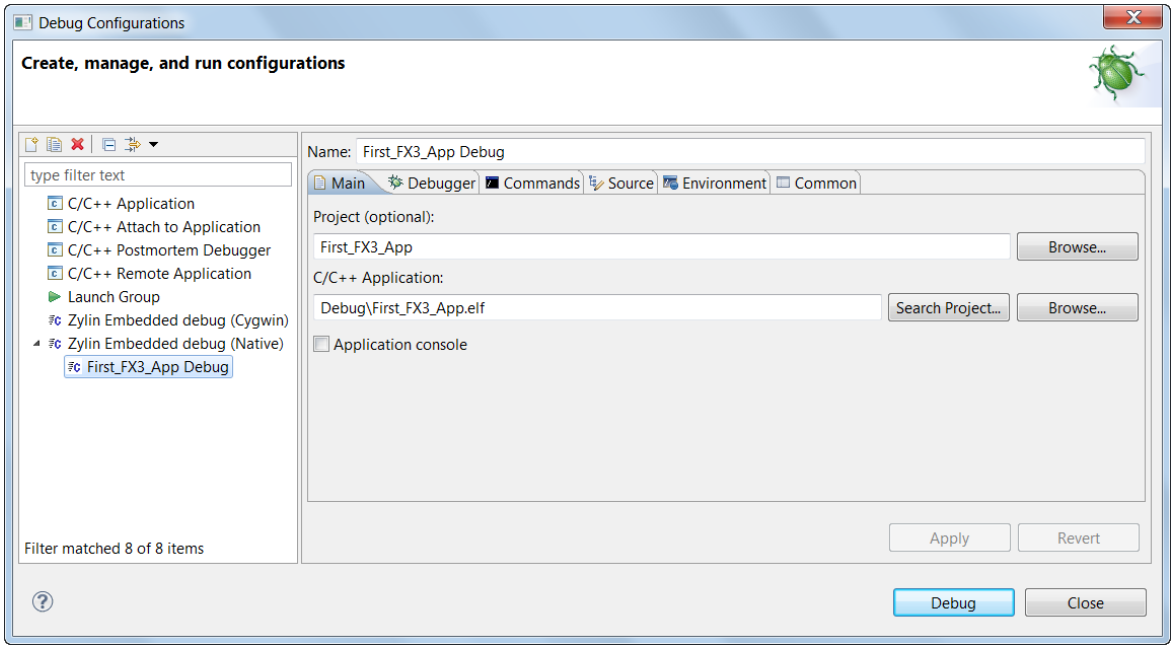
2. Right-click the **Zylin Embedded debug (Native)** item and select **New** to define a debug configuration (Figure 2-28).

Figure 2-28. Create a new Debug Configuration



3. In the **Main** tab, click the **Search Project…** button and select the *.elf* (executable and linkable) file shown there (Figure 2-29). You can name the debug profile anything you want.

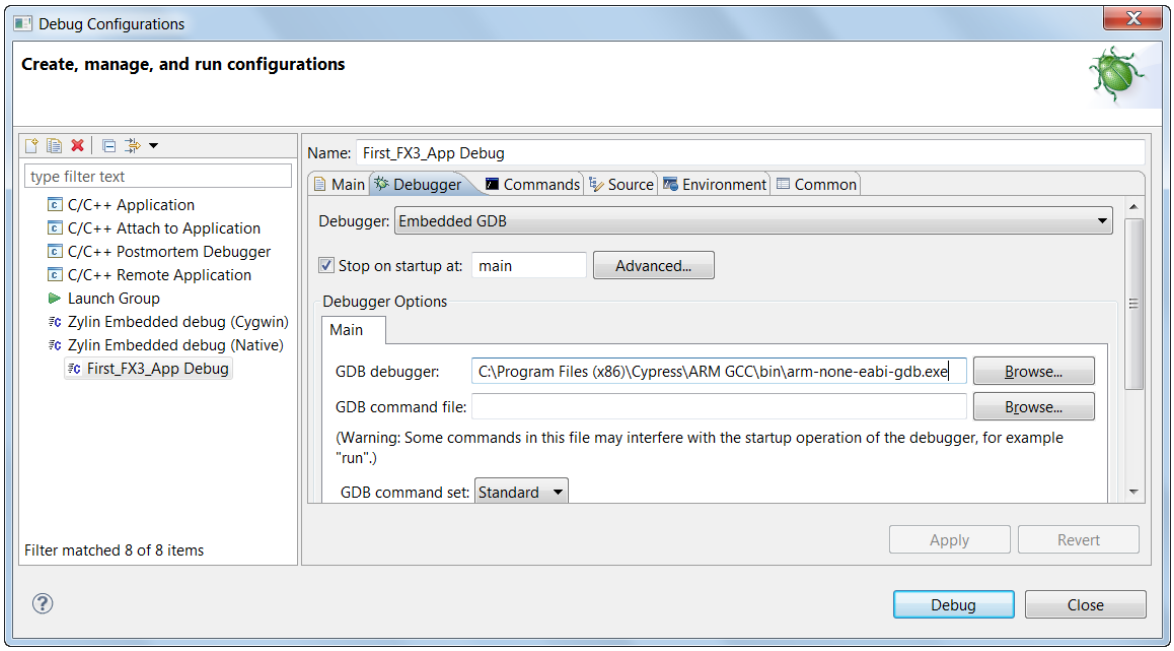
Figure 2-29. Search Project for Executable (elf) File



4. Select the **Debugger** tab. Confirm that the Embedded GDB is selected, and then **Browse** to the GDB Debugger at the path shown in Figure 2-30. (GDB stands for Gnu debugger.)

   Program Files
   (or Program Files (x86))\Cypress\EZ-USB FX3 SDK\version\bin\arm-none-eabi-gdb.exe.

Figure 2-30. Browse to Path to GDB Debugger



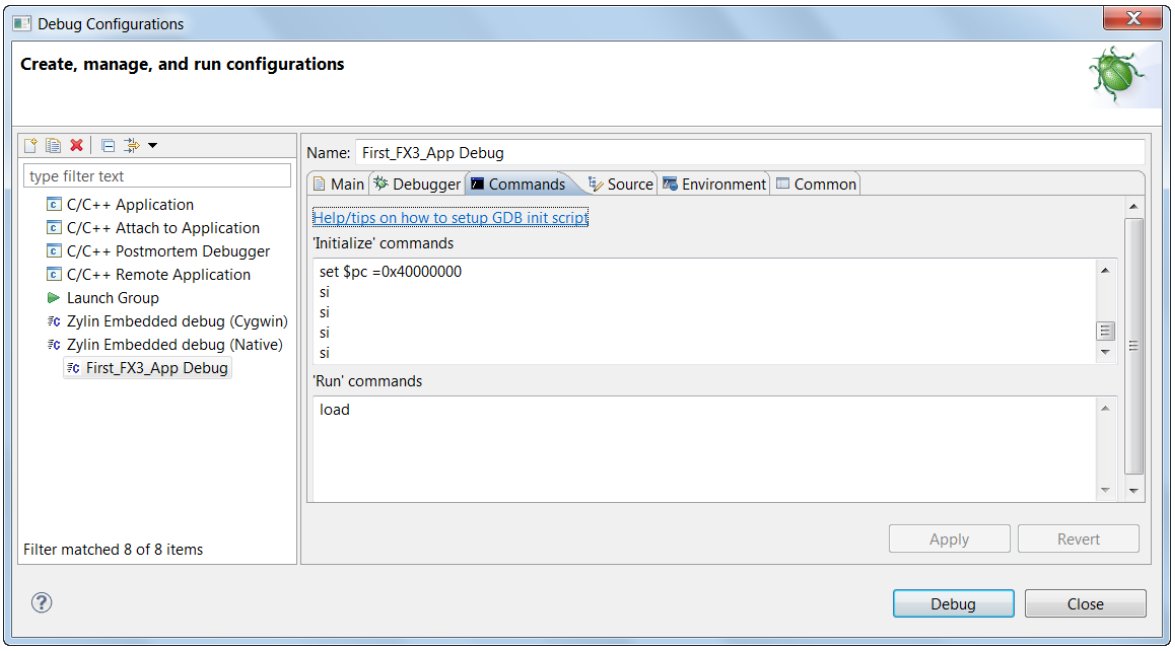
5. Select the **Commands** tab, and copy and paste the text shown in Figure 2-31 into the **'Initialize' commands** text box.

Figure 2-31. Copy/Paste Text into 'Initialize' commands box

```
set prompt (arm-gdb)
# This connects to a target via netsiliconLibRemote
# listening for commands on a TCP port on the local machine.
# 2331 if the Segger J-Link GDB Server is being used
# 3333 if OpenOCD is being used
# If OpenOCD is being used, the CPU should be halted
# using the "monitor halt" command.
# Uncomment the appropriate line below:
target remote localhost:2331
# target remote localhost:3333
# monitor halt
monitor speed 1000
monitor endian little
set endian little
monitor reset
# Set the processor to SVC mode
monitor reg cpsr =0xd3
# Disable all interrupts
monitor memU32 0xFFFFF014 =0xFFFFFFFF
# Enable the TCMs
monitor memU32 0x40000000 =0xE3A00015
monitor memU32 0x40000004 =0xEE090F31
monitor memU32 0x40000008 =0xE240024F
monitor memU32 0x4000000C =0xEE090F11
# Change the FX3 SYSCLK setting based on
# input clock frequency. Update with
# correct value from list below.
# Clock input is 19.2 MHz: Value = 0x00080015
# Clock input is 26.0 MHz: Value = 0x00080010
# Clock input is 38.4 MHz: Value = 0x00080115
# Clock input is 52.0 MHz: Value = 0x00080110
monitor memU32 0xE0052000 = 0x00080015
# Add a delay to let the clock stabilize.
monitor sleep 1000
set $pc =0x40000000
si
si
si
si
```

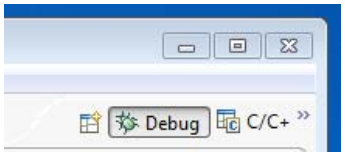
6. In the **'Run' commands** box, type "load". The **Commands** tab should now look like that shown in Figure 2-32.

Figure 2-32. Filled-in Commands Screen



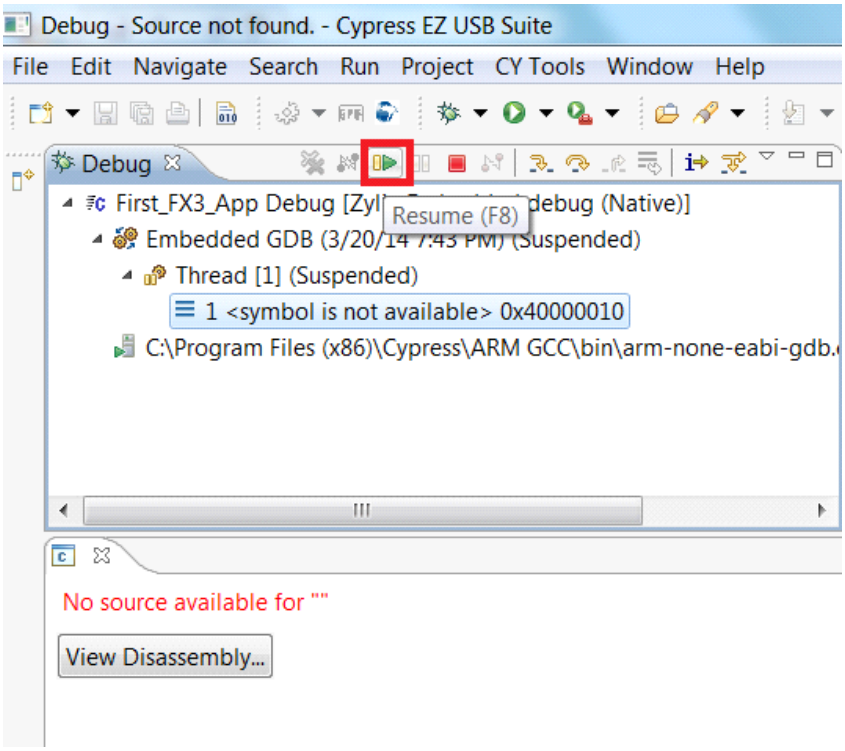
7. Click the **Debug** button to start a debug session. You are given the option of switching to the **Debug** Perspective. An Eclipse Perspective is a screen layout tailored to the task at hand. The Debug perspective contains windows that help you debug. To change perspectives at any time, select between **Debug** and **C/C++** in the upper-right Eclipse window (Figure 2-33).

Figure 2-33. Change the Eclipse Perspective



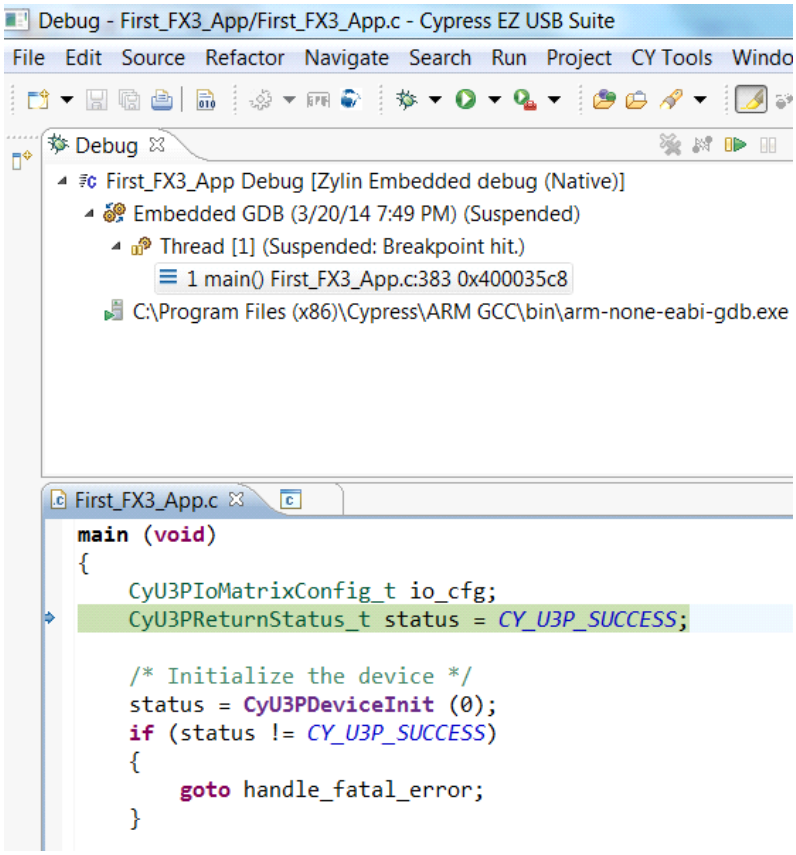
8. You should see a burst of activity in the J-Link Server window as the FX3 code loads over JTAG. When the activity stops the FX3 code execution stops inside an internal code module that is not visible to the debugger. To get past this, click the **Resume** button (Figure 2-34) or press **[F8]**.

Figure 2-34. Resume, Suspend and Terminate Buttons



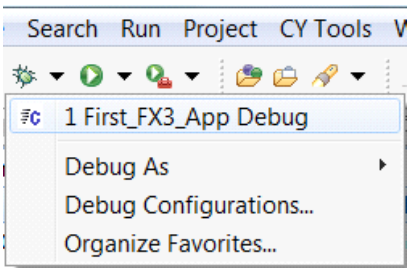
9. The first code line in the **main** function is highlighted. Pressing **[F6]** single-steps through the code. You can set and clear breakpoints by double-clicking lines of code.

Figure 2-35. Set and clear breakpoints



10. Click the **Terminate** button to stop a debug session. To return to the code development perspective, select the **C/C++** perspective as shown in Figure 2-33.
11. To restart the debugger, select the bug button. Once you run one debug session, the debugger remembers the name and places it as the first debug choice (Figure 2-36).

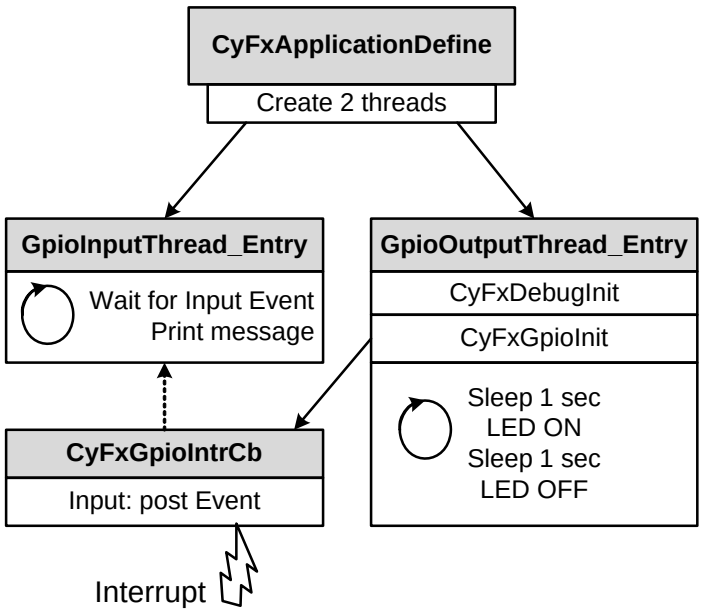
Figure 2-36. Start the Debugger



2.6.2 Code Structure

The simple GPIO application (First_FX3_App) illustrates the basic structure of an FX3 application. FX3 code development uses an RTOS (real-time operating system) called ThreadX to manage high performance applications. Refer to the *FX3 Programmers Manual.pdf* file available in the documentation folder of the FX3 SDK (*C:\Program Files (x86)\Cypress\EZ-USB FX3 SDK\1.3\doc*) to get a detailed explanation of example FX3 application source code. Figure 2-37 illustrates the structure of a typical FX3 ThreadX application.

Figure 2-37. Structure of the FX3 First Application



The **CyFxApplicationDefine** function creates two threads. Threads run concurrently on a time-shared basis as the application executes. This example creates an output thread for the LED and an input thread for the push button.

The output thread initializes the serial debugger and creates an interrupt callback link to the function **CyFxGpioIntrCb**. Then it runs a continuous loop that blinks the LED. Timing is accomplished by telling the thread to sleep for a programmed number of milliseconds, 1000 ms in this example. Putting threads to sleep until work is necessary is a good way to maximize the CPU utilization.

Part of the GPIO initialization is to cause an interrupt on any change of the push button state. A state change calls the **CyFxGpioIntrCb** ISR (Interrupt Service Routine). Serial debug messages cannot be printed from the GPIO callback as it runs in the interrupt context. All interrupts compete for CPU attention, and there may be more important tasks at hand. ThreadX handles this by providing a messaging system, whereby the ISR posts a message for another non-interrupt function to execute.

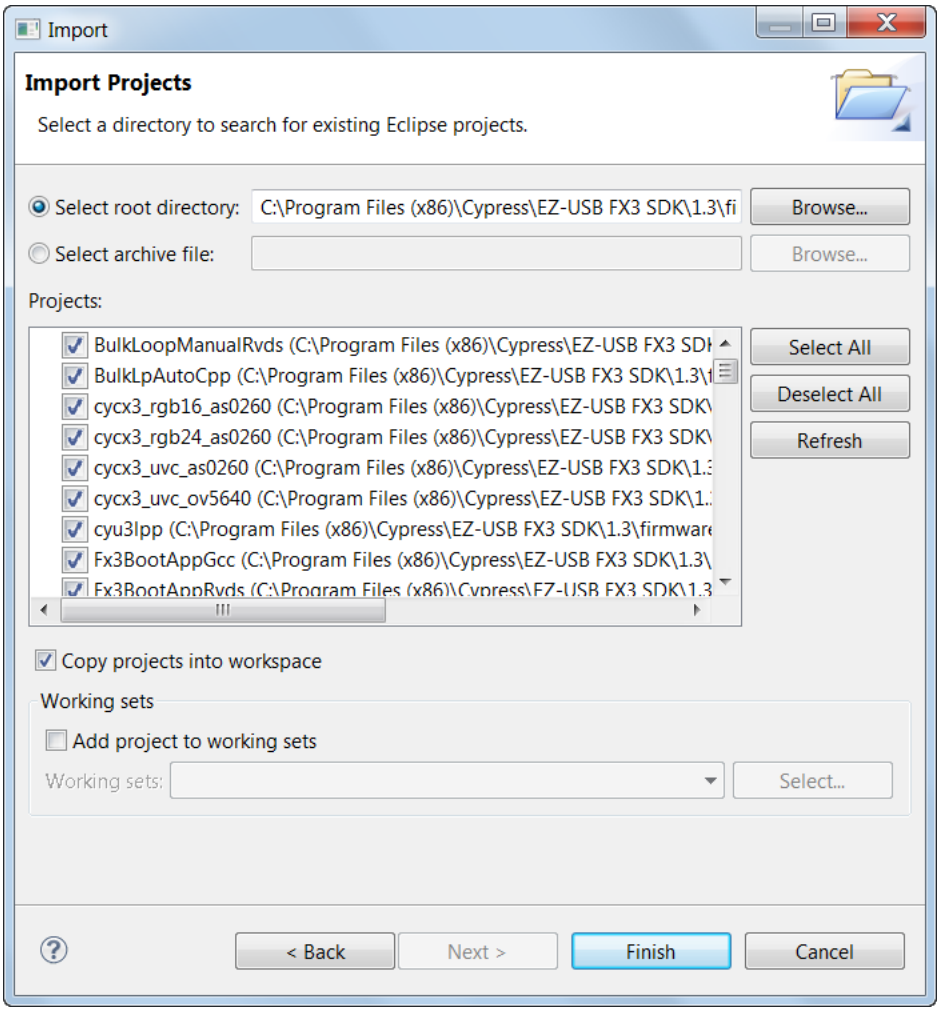
The Input thread is the recipient of the message to print a debug message. It runs a continuous loop that checks for the message posted by the ISR and prints the debug message over the serial port.

2.7 Importing All of the Cypress Examples

The GPIO example (First_FX3_App) was added to Eclipse as a single project for simplicity. The Eclipse Project Manager can contain multiple projects for easy reference. Follow these steps to import all of the Cypress example projects:

1. Choose **File>Import>General>Existing Projects into Workspace**.
2. Navigate to *Program Files\Cypress\EZ-USB FX3 SDK\(version)* and highlight the **firmware** folder then click **OK**. All the projects in the **firmware** folder appear in the **Projects**: window (Figure 2-38).
3. Select **Copy projects into workspace** to make your own local copies, and then click **Finish**.

Figure 2-38. Import All Cypress Example Projects



The Eclipse **Project Explorer** window now contains all the Cypress example projects. To work on an individual project, select it in the **Project Explorer** window. The project name is highlighted so you can tell at a glance which project you are editing, building and debugging.

3. Kit Operation

The FX3 DVK board provides convenient access to FX3 interfaces such as I^2C, SPI, UART and I^2S. The DVK board also provides a high-speed (Samtec) connector for interface to external devices through the GPIF II (General Programmable Interface, Gen 2). FX3 SDK examples help you evaluate and program each of these interfaces.

3.1 Default Jumper Settings on DVK Board

The jumpers and dip switches on the FX3 DVK board are factory set to power the board over USB. The factory-set jumper and switch settings are shown in Table 3-1.

Table 3-1. Default Jumper Settings

S.No	Jumper/Switch	Pins to be shorted using jumpers	Function	
1	J53	1 and 2	Bus powered	
2	J101	1 and 2	GPIO_46=UART_RTS	
3	J102	1 and 2	GPIO_47=UART_CTS	
4	J103	1 and 2	GPIO_48=UART_TX	
5	J104	1 and 2	GPIO_49=UART_RX	
6	J136	3 and 4	VIO1(3.3 V)	
7	J144	3 and 4	VIO2(3.3 V)	
8	J145	3 and 4	VIO3(3.3 V)	
9	J146	3 and 4	VIO4(3.3 V)	
10	J134	4 and 5	VIO5(3.3 V)	
11	J135	2 and 3	CVDDQ(3.3 V)	
12	J143	1 and 6	VBATT(2.5 V)	
13	J96 and SW25	2 and 3	a. PMODE0 Pin state (ON/OFF) selection using SW25. b. SW25.1 should be OFF	USB boot
14	J97 and SW25	2 and 3	a. PMODE0 Pin state (ON/OFF) selection using SW25. b. SW25.2 should be OFF	
13	J98	1 and 2	PMODE2 Pin Floating	
14	J72	1 and 2	RESET	
15	J42	Not Installed	GPIO_58=I2C_SCL	
16	J45	Not Installed	GPIO_59=I2C_SDA	
17	J100	1 and 2	GPIO_21=CTL4	

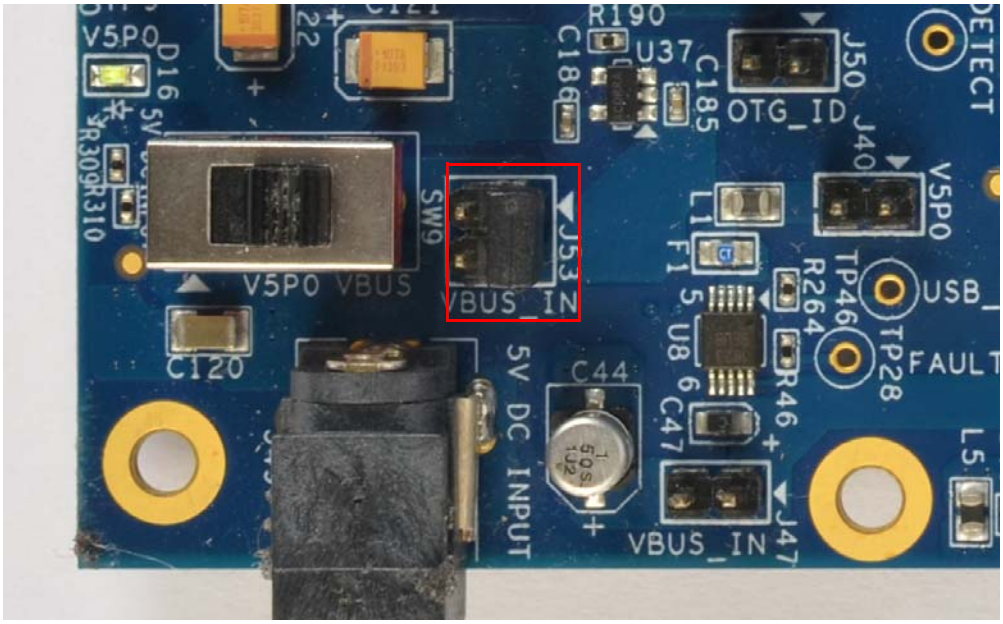
Note For more details on each I/O pin refer to Chapter Hardware on page 65.

3.2 Bus-Power Mode

The FX3 DVK board can operate in bus-power mode. Follow these steps:

1. Verify that pins 1–3 or 2–4 of jumper J53 are shorted.
2. Use the toggle switch SW9 to power ON the board. The switch should point to the direction labeled VBUS_IN instead of V5P0, as shown in Figure 3-1.

Figure 3-1. Bus-Power Mode Setting on FX3



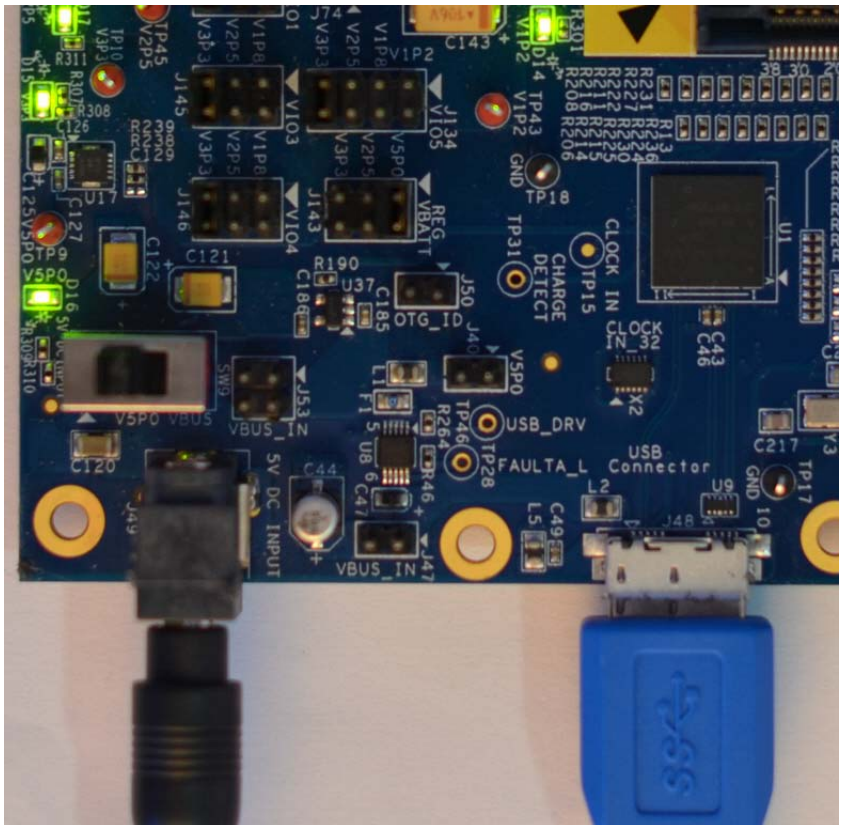
3.3 Self-Power Mode

The FX3 DVK board also can operate without a USB connection by utilizing the DC input jack. Follow these steps to connect the board in self-power mode.

1. Remove all jumpers from J53.

2. Plug the 5-V power supply adapter supplied with the kit into the J49 power jack. Use the toggle switch SW9 to power the board. The switch should point to the direction labeled V5P0 instead of VBUS_IN as shown in Figure 3-2.

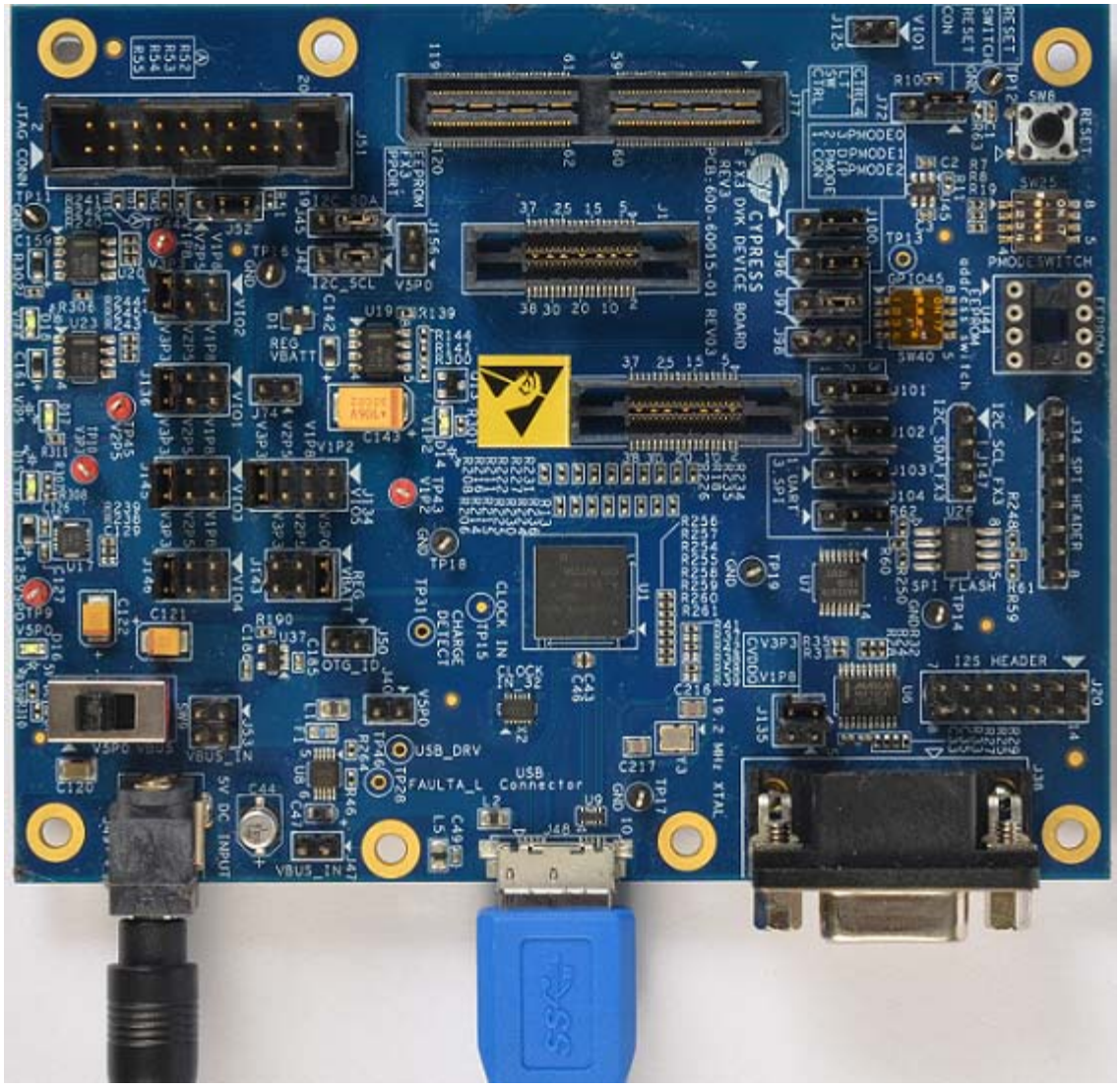
Figure 3-2. Self-Power Mode



3.4 First Time USB Enumeration

When the FX3 board is powered for the first time, connect the USB 3.0 A to Micro-B cable supplied with the kit to the J48 connector, as shown in Figure 3-3.

Figure 3-3. USB 3.0 Cable to J48 Connector



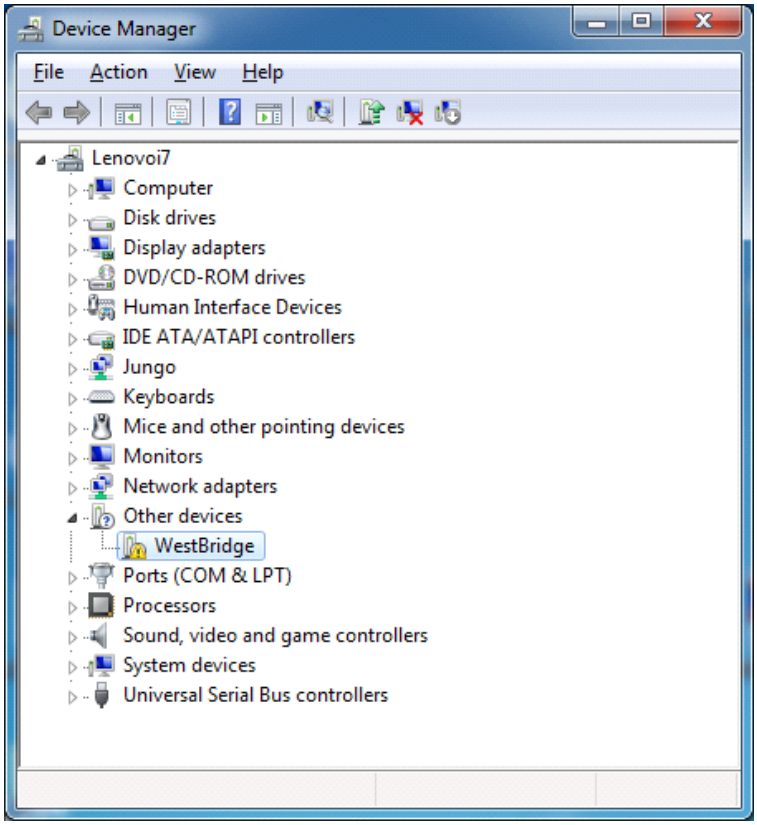
Note The bootloader in FX3 ROM by default enumerates as USB 2.0 device. After downloading, the firmware to the FX3 DVK board will enumerate as either a USB 3.0 or USB 2.0 peripheral depending on the configuration defined in firmware and the host PC capability.

3.4.1 Manual Installation of Cypress Driver

1. In Windows, invoke **Start > Computer** and right-click **Properties > Select Device Manager**. Locate the FX3 device entry with a yellow symbol in the Other Devices list. Sometimes, the FX3 DVK enumerates as **Westbridge**, as shown in Figure 3-4.

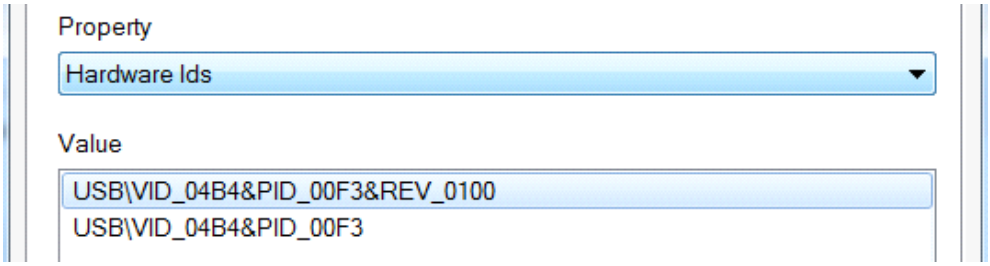
   Note: In Windows XP invoke **Start > My Computer** and right-click **Properties > Hardware Tab > Select Device Manager**.

Figure 3-4. FX3 Device Entry in Windows Device Manager



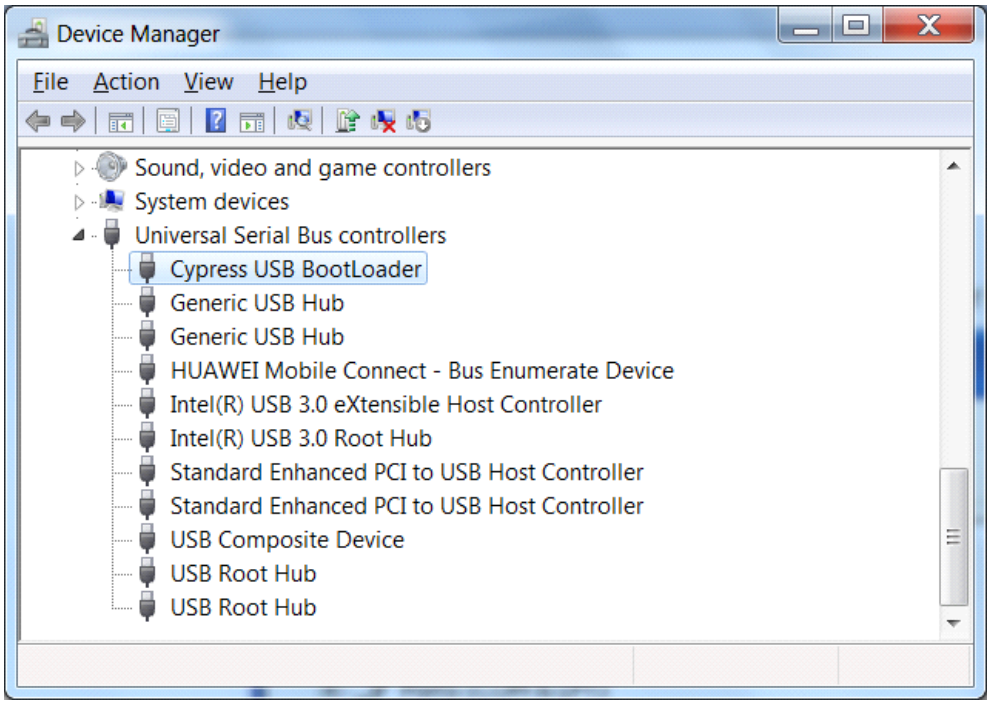
2. Right-click on the yellow device entry to verify the VID/PID of the device. Choose **Properties > Details**. Select **Hardware Ids** and observe if the default VID/PID is 0x04B4/0x00F3, as shown in Figure 3-5.

Figure 3-5. Default FX3 Bootloader VID/PID



3. Right-click on **WestBridge** and select **Browse my computer for driver software**. On a 64-bit Windows 7 machine, the driver is located at *C:\Program Files (x86)\Cypress\EZ-USB FX3 SDK\1.3\driver\win7\x64*. Your SDK version number may be higher than 1.3. The **Device Manager** window should remove the **WestBridge** entry and identify the FX3 DVK board as the **Cypress USB BootLoader** (top entry) as shown in Figure 3-6.

Figure 3-6. FX3 Entry in Device Manager after Driver Binding



3.5 Boot Options

FX3 provides several booting options. The boot option is determined by the FX3 PMODE[2:0] input pins. These PMODE[2:0] pins are configured using combination of Jumpers (J98, J97, and J96) and dip switch SW25. For I^2C and SPI boot options additional jumpers (J42, J45, J101, J102, J103, and J104) and a dip switch (SW40) needs to be configured.

Figure 3-7 summarizes the PCB areas that implement different boot options.

Figure 3-7. Boot Dip Switch and Jumper Settings

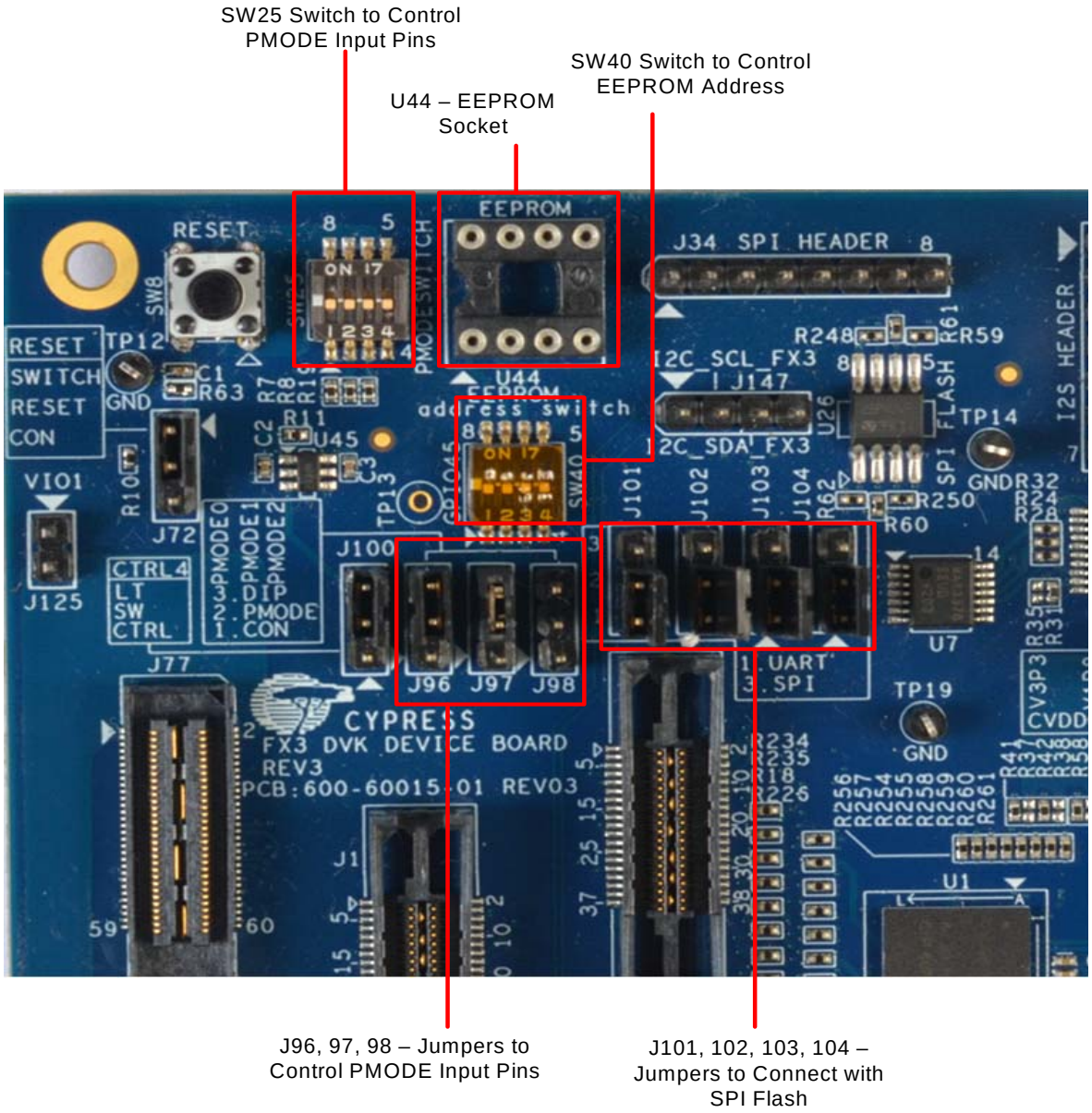


Table 3-2 shows the FX3 boot interfaces according to the PMODE[2:0] settings. “USB Fallback” means that USB is used to boot code if the indicated interface is not present.

Table 3-2. Boot Options with PMODE Pins

PMODE2 Pin state	PMODE1 Pin state	PMODE0 Pin state	Boot Option	USB Fallback
Z	0	0	Sync ADMUX(16-bit)	No
Z	0	1	Async ADMUX(16-bit)	No
Z	1	1	USB Boot	Yes
Z	0	Z	SRAM16-bit)	No
1	Z	Z	I2C	No
Z	1	Z	I2C=> USB	Yes
0	Z	1	SPI=> USB	Yes

Note “Z” = floating I/O pin state.

On the DVK board, the PMODE [2:0] pins are available through jumpers J98, J97, and J96, and switch SW25. Table 3-3 shows the details of the jumper and switch combination to select a specific boot option.

Table 3-3. PMODE Pin Options on DVK Board

PMODE Pin	PMODE Pin State	Jumper and Dip Switch Combination for PMODE Pin State
PMODE2	Z	1. No jumper installed on J98 2. SW25.3 – Don't care (the switch can be either turned ON or OFF)
	1	1. Short pins 2–3 of jumper J98 2. SW25.3 set to OFF
	0	1. Short pins 2–3 of jumper J98 2. SW25.3 set to ON
PMODE1	Z	1. No jumper installed on J97 2. SW25.2 – Don't care (the switch can be either turned ON or OFF)
	1	1. Short pins 2–3 of jumper J97 2. SW25.2 set to OFF
	0	1. Short pins 2–3 of jumper J97 2. SW25.2 set to ON
PMODE0	Z	1. No jumper installed on J96 2. SW25.1 – Don't care (the switch can be either turned ON or OFF)
	1	1. Short pins 2–3 of jumper J96 2. SW25.1 set to OFF
	0	1. Short pins 2–3 of jumper J96 2. SW25.1 set to ON

Note The “OFF” setting on dipswitch SW25 means logical 1 on the corresponding PMODE pin.

See AN76405 - EZ-USB FX3 Boot Options for an explanation of each of these boot options.

3.6 USB Boot

The FX3 DVK board boots in USB mode if the PMODE[2:0] pins are set to Z11. The board is by default set to USB boot. The board enumerates with bootloader vendor ID/product ID (VID/PID-0x04B4/0x00F3) when the USB cable is connected to the USB host PC.

3.6.1 Download Firmware Image to FX3 RAM

Follow the procedure outlined here to download the firmware image to FX3 RAM.

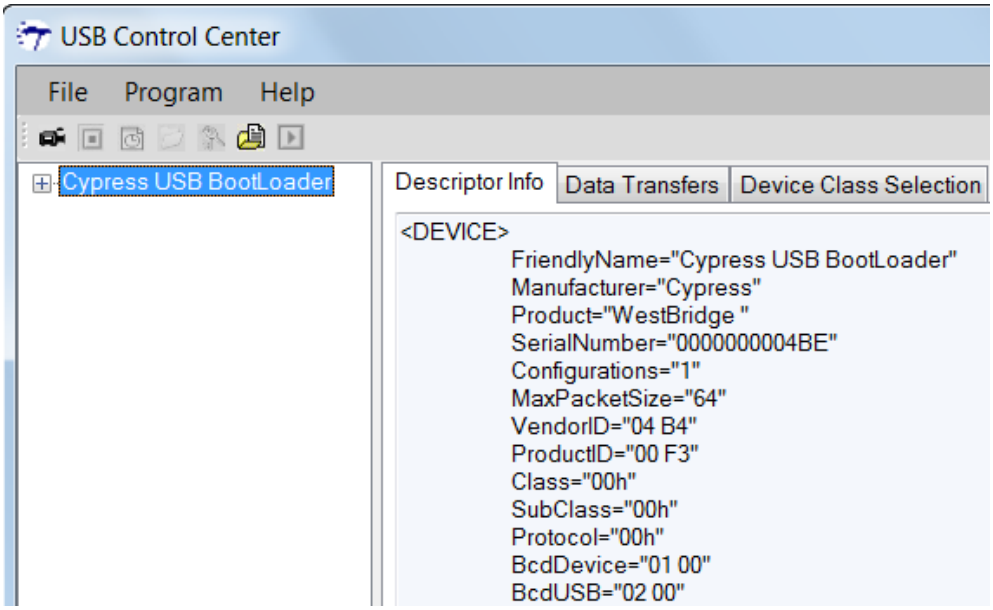
1. Enable USB boot by setting the PMODE[2:0] pins to Z11. On the DVK board, this is done by configuring the jumpers and switches, as shown in Table 3-4.

Table 3-4. Jumper Settings for USB Boot

PMODE Pin	Required PMODE Pin State	Jumper and Dip Switch Combination
PMODE2	Z	1. No jumper installed on J98 2. SW25.3 – Don't care (the switch can be either turned ON or OFF)
PMODE1	1	1. Short pins 2–3 of jumper J97 2. SW25.2 set to OFF
PMODE0	1	1. Short pins 2–3 of jumper J96 2. SW25.1 set to OFF

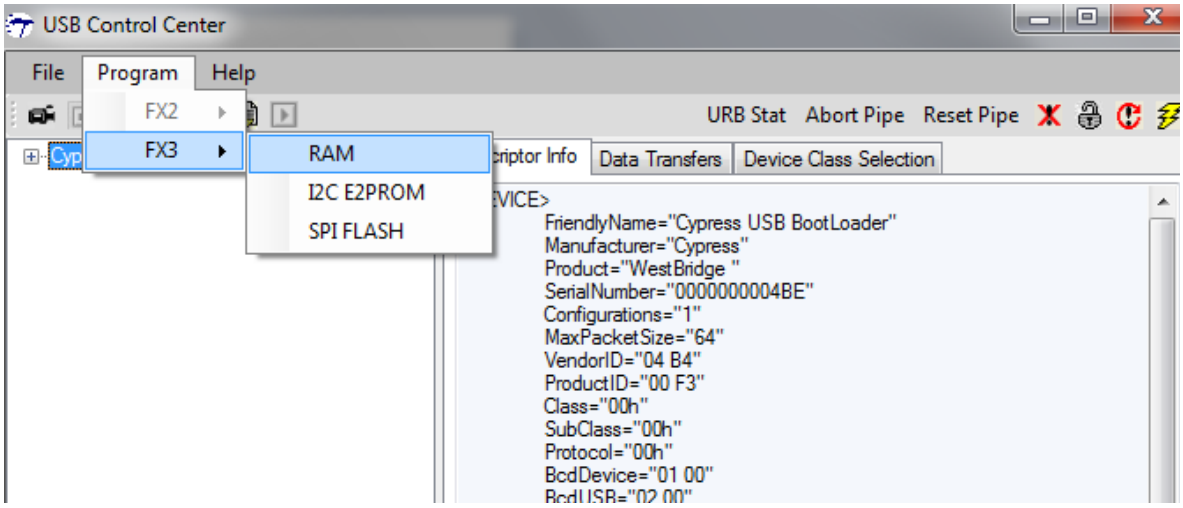
2. Open the USB Control Center application by clicking **Start > All Programs > Cypress > Cypress USBSuite > Control Center**. When connected to a USB host, the FX3 device enumerates in the USB Control Center as **Cypress USB BootLoader**, as shown in Figure 3-8.

Figure 3-8. FX3 Default BootLoader Entry in USB Control Center



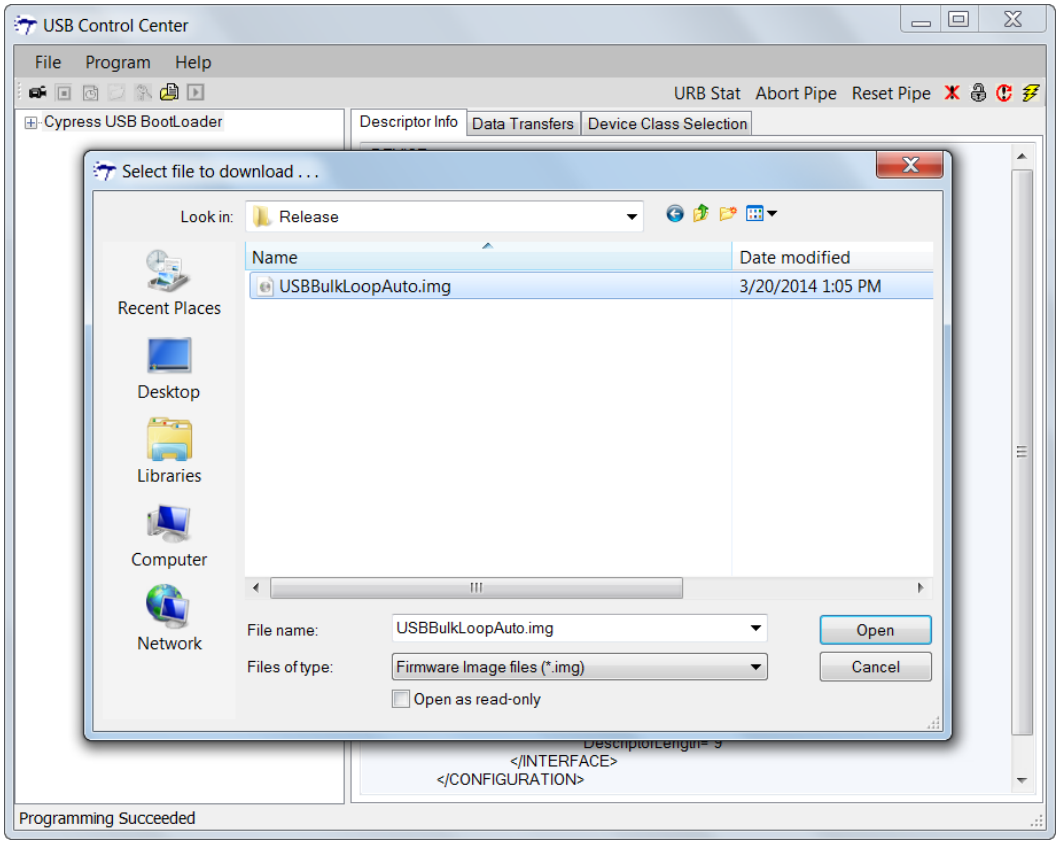
3. In the USB Control Center, choose **Program > FX3 > RAM**, as shown in Figure 3-9.

Figure 3-9. Choose FX3 RAM from the USB Control Center



4. Browse to the firmware image (.img) file to be programmed into the FX3 RAM. Double-click on the .img file, as shown in Figure 3-10.

Figure 3-10. Firmware Image Selection Before Download



5. A **Programming Succeeded** message is displayed on the bottom left pane of the Control Center and the FX3 device re-enumerates with the programmed firmware.

3.7 I²C Boot

The FX3 DVK board provides an I²C interface as one of the boot options. Mount an I²C EEPROM (for e.g., 24LC1025 or 24LC1026) into the U44 socket and verify this feature by programming the firmware image. The size of the selected EEPROM should be of at least 128 KB capacity.

3.7.1 Download Firmware Image to I²C EEPROM

Follow the procedure outlined here to download the firmware image to I2C EEPROM:

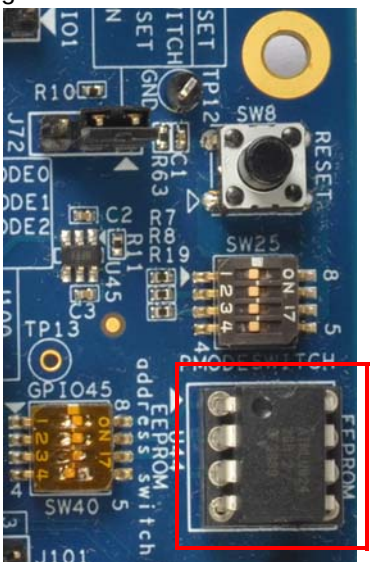
1. Disconnect the USB cable between the USB host PC and FX3 DVK board.
2. Enable USB boot by setting the PMODE[2:0] pins to Z11. On the DVK board, this is done by configuring the jumpers and switches, as shown in Table 3-5.

Table 3-5. USB Boot Jumper and Dipswitch Settings

PMODE Pin	Required PMODE Pin State	Jumper and Dip Switch Combination
PMODE2	Z	1. No jumper installed on J98 2. SW25.3 – Don't care (the switch can be either turned ON or OFF)
PMODE1	1	1. Short pins 2–3 of jumper J97 2. SW25.2 set to OFF
PMODE0	1	1. Short pins 2–3 of jumper J96 2. SW25.1 set to OFF

3. Verify the firmware image (.img) size before downloading to EEPROM. To select the EEPROM of correct configuration, see Choosing the Correct I2C EEPROM Configuration on page 48.
4. Mount the selected I²C EEPROM onto the FX3 board. Verify that the EEPROM pin 1 indicator lines up with the board pin 1 triangle and is firmly fixed in the U44 socket. Figure 3-11 shows an I²C EEPROM mounted into the U44 socket.

Figure 3-11. I²C EEPROM Mounted into U44 Socket



5. Before attempting to program the EEPROM, ensure that the address signals of the EEPROM are configured correctly using dip switch SW40 (for example, Microchip part 24AA1025, 1–8 ON, 2–7 ON, 3–6 OFF). Also, the I^2C clock (SCL) and data line (SDA) jumpers J42 and J45 pins 1–2 should be shorted on the DVK board.
6. When connected to a USB host, the FX3 device enumerates in USB Control Center as **Cypress USB BootLoader**.
7. In Control Center, select the FX3 device and then choose **Program > FX3 > I2C EEPROM**. This causes a special I^2C boot firmware to be programmed into the FX3 device, which then enables programming of the I^2C device connected to FX3. Hence, now the FX3 device re-enumerates as **Cypress USB BootProgrammer**.

Figure 3-12. FX3 USB Bootloader Entry in Control Center

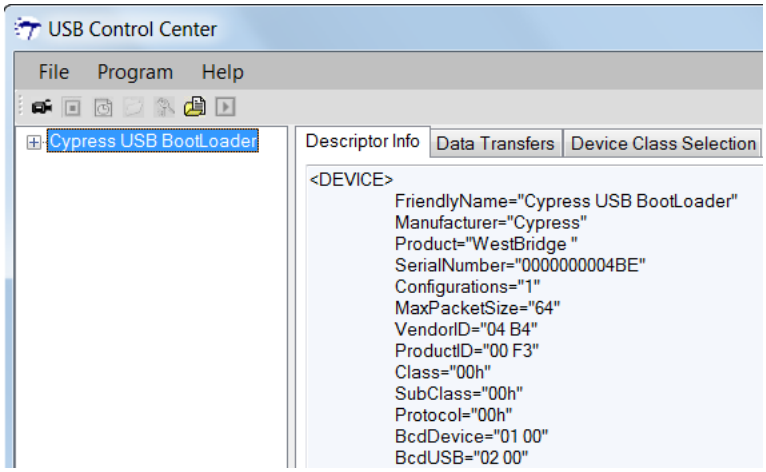


Figure 3-13. I^2C EEPROM Download Option in Control Center

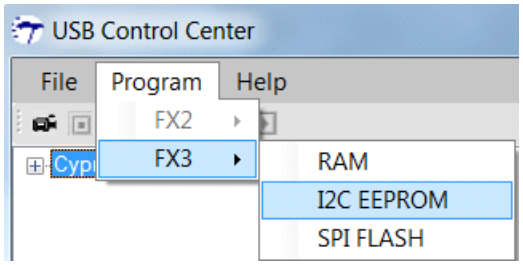
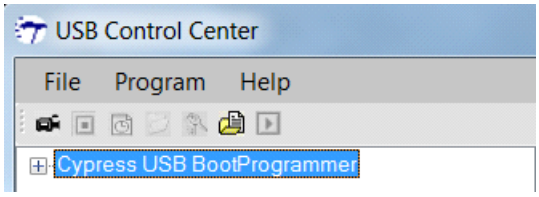


Figure 3-14. FX3 Re-Enumerates as Cypress USB Boot Programmer



8. After the FX3 DVK board enumerates as **Cypress USB BootProgrammer**, the Control Center application prompts the user to select the firmware binary to download. Browse to the relevant release mode firmware binary, as shown in Figure 3-15.
9. The bottom left corner of the window displays **Programming of I2C E2PROM in Progress---**, as shown in Figure 3-16.

Figure 3-15. Select Firmware Image to Download

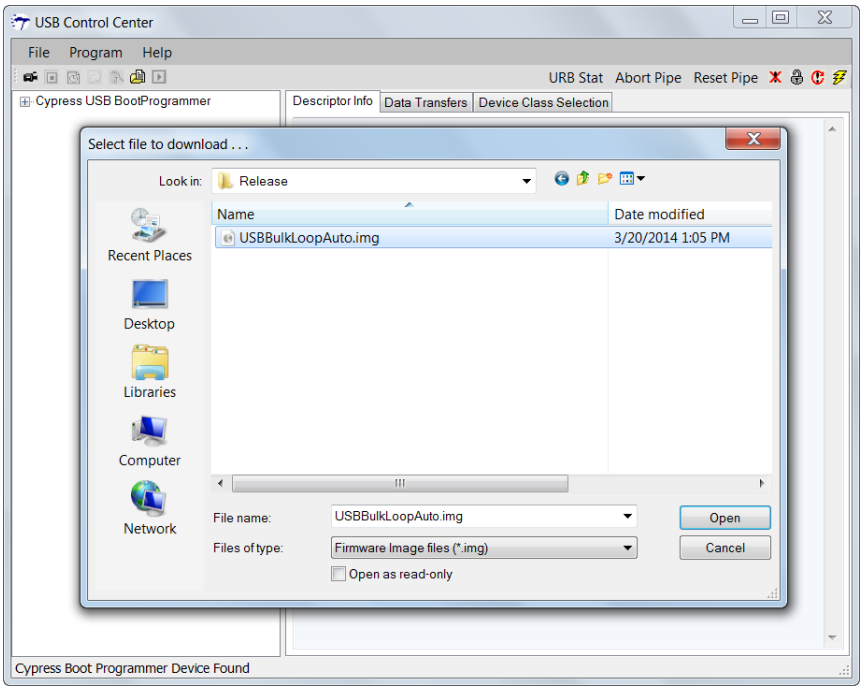
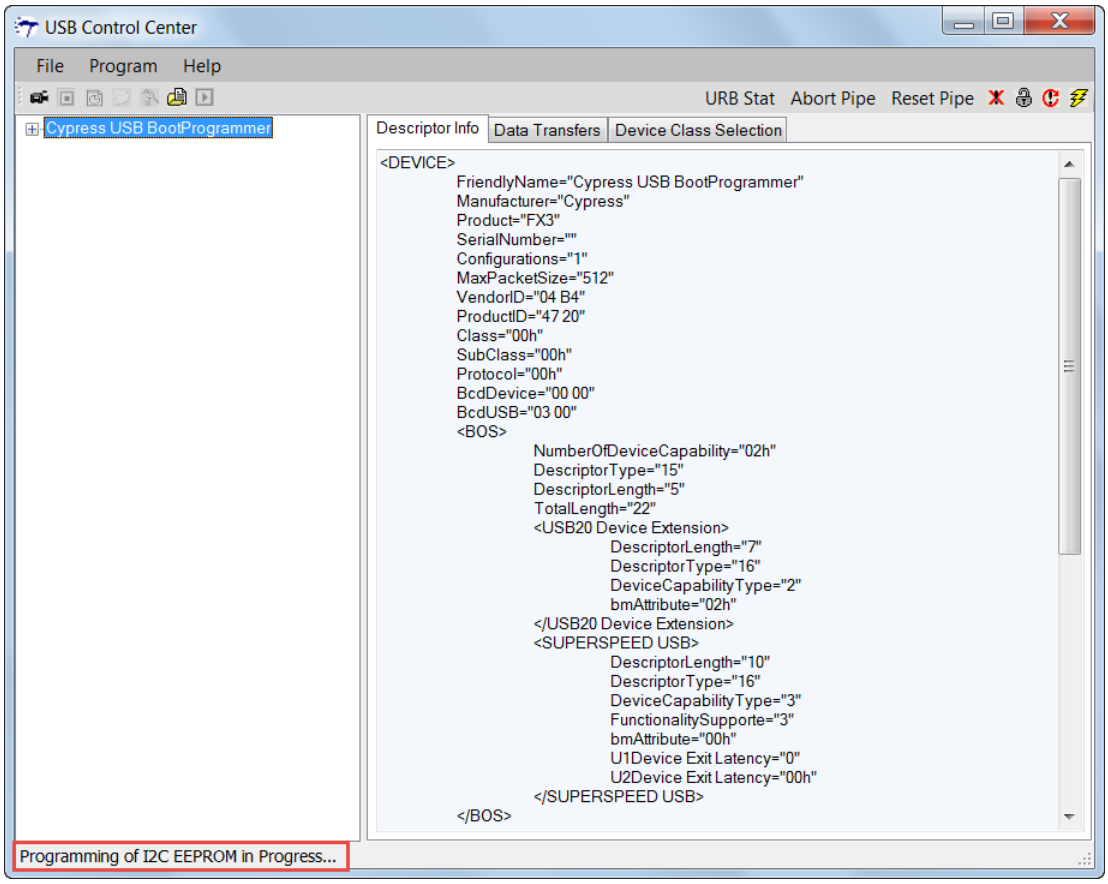


Figure 3-16. I2C EEPROM Programming Update in Control Center



3.7.2 Booting from I²C EEPROM

Change the PMODE pins on the DVK board to Z1Z to enable I2C boot. On the DVK board, this is done by configuring the jumpers and switches, as shown in Table 3-6. Press RESET button SW8. The FX3 DVK board re-enumerates with the boot image in the I²C EEPROM.

Table 3-6. I2C Boot Jumper and Dip Switch Settings

PMODE)in	Required PMODE Pin State	Jumper and Dip Switch Combination
PMODE2	Z	1. No jumper installed on J98 2. SW25.3 – Don't care (the switch can be either turned ON or OFF
PMODE1	1	1. Short pins 2–3 of jumper J97 2. SW25.2 set to OFF
PMODE0	Z	1. No jumper installed on J96 2. SW25.1 – Don't care (the switch can be either turned ON or OFF)

3.7.3 Choosing the Correct I²C EEPROM Configuration

Before mounting the I²C EEPROM into the U44 socket, you need to consider the following factors to select the correctly configured EEPROM for the respective size of the firmware image.

- Verify that the I²C EEPROM can accommodate the firmware image built in Release mode. Check the addressing (A0, A1, and A2) mechanism for the EEPROM part and select these pins on board using SW40. Follow the steps to decide on the addressing mechanism for the different EEPROM configurations.
  - The typical EEPROM pins consists of A0, A1, and A2 address lines. FX3 can address eight EEPROM (000–111). The address selection can be done using dip switch SW40.

Figure 3-17. EEPROM Block Diagram

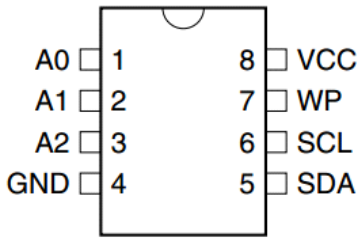
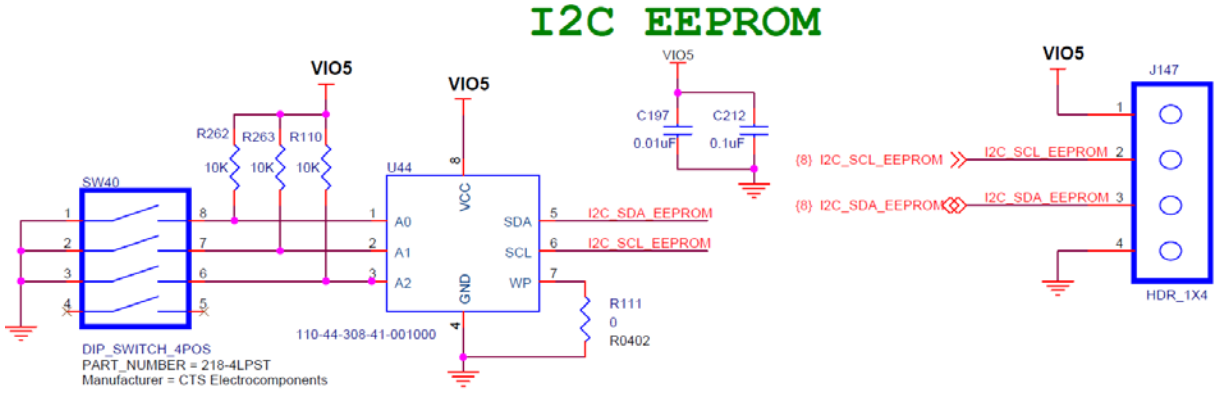


Figure 3-18. SW40 and EEPROM Interconnect



- The firmware image has a fixed header format to boot from the I^2C EEPROM. After the 2-byte “CY” signature, the third byte bits[3:1] can be used to specify the EEPROM size format. The parameter "bImageCTL" should be modified to reflect the correct EEPROM size. For 128-KB/ 64-KB Atmel EEPROMs, bit[3:1] should be 6; for the 128-KB Microchip EEPROM (24LC1024), it should be 7.
- As shown in Figure 3-18, if the A[2:0] pins are set to 'ON' (closed position), the corresponding address line is connected to ground. If the switch position is set to OFF (open position), the corresponding address line is pulled high to the VIO5 voltage level.

Note For the 24LC1026 Microchip EEPROM, follow the same addressing mechanism as for the Atmel EEPROM See AN76405 – EZ-USB FX3 Boot Options for more details on I^2C booting mechanism.

3.8 SPI Boot

The FX3 DVK board can boot from a Serial Flash Memory using its SPI interface if PMODE[2:0] pins are set to 0Z1. Similarly to the I^2C boot method, the FX3 board first loads SPI programmer code which can load code images into the onboard flash memory (M25P40) using the USB Control Center. Then the DVK board jumper and switch settings are readjusted to boot the image from the flash memory.

3.8.1 Download Firmware Image to SPI Flash

Follow this procedure to download the firmware image to SPI flash:

1. Disconnect the USB cable between the USB host PC and FX3 DVK board.
2. Enable USB boot, by setting the PMODE[2:0] pins to Z11. On the DVK board, this is done by configuring the jumpers and switches as shown in Table 3-7.

Table 3-7. USB Boot Jumper and Dip Switch Settings

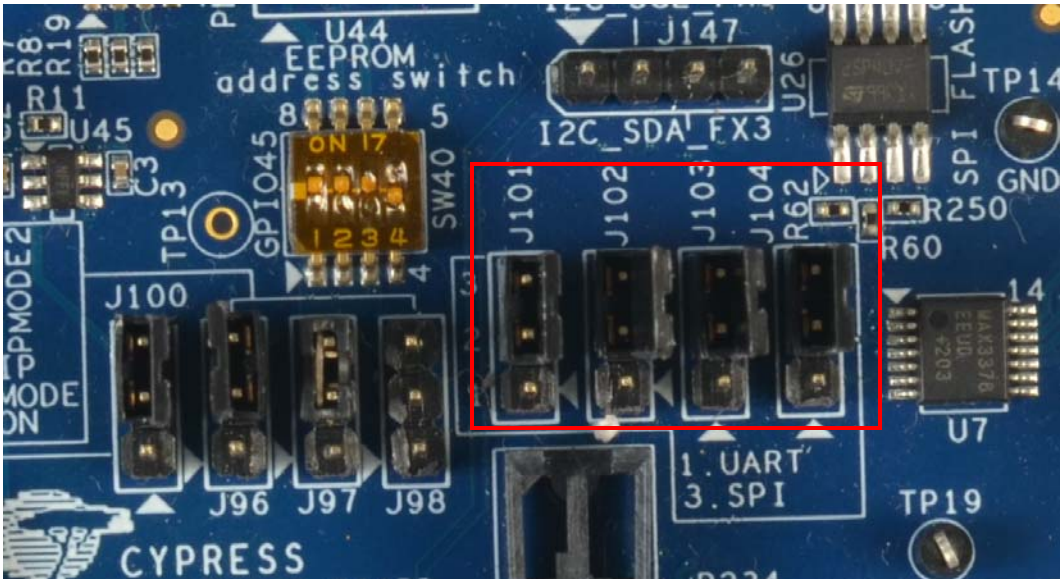
PMODE Pin	Required PMODE Pin State	Jumper and Dip Switch Combination
PMODE2	Z	1. No jumper installed on J98 2. SW25.3 – Don't care (the switch can be either turned ON or OFF)
PMODE1	1	1. Short pins 2–3 of jumper J97 2. SW25.2 set to OFF
PMODE0	1	1. Short pins 2–3 of jumper J96 2. SW25.1 set to OFF

3. When the FX3 DVK board is connected to a USB PC host using the USB 3.0 cable, the FX3 device enumerates in Control Center as **Cypress USB Bootloader**.
4. Verify the SPI mode related jumper settings, as shown in Table 3-8. These jumpers are used to select either the UART or SPI I/O pins.

Table 3-8. SPI Mode Jumper Settings

Sl. No.	Pin Description	Jumper Position
1	J101-SPI_CLK	2–3
2	J102-SPI_SSN	2–3
3	J103-SPI_MISO	2–3
4	J104-SPI_MOSI	2–3

Figure 3-19. Select SPI Interface Using Jumpers (J101–J104)



Note The white arrows on the DVK board indicate pin1 of the corresponding jumper.

5. In the USB Control Center, select **Program > FX3 > SPI FLASH**. The USB Control Center application downloads the Serial Flash programming code, and the FX3 DVK board reconnects to USB as **Cypress USB BootProgrammer**, as shown in Figure 3-21.

Figure 3-20. Select SPI Flash Download in Control Center

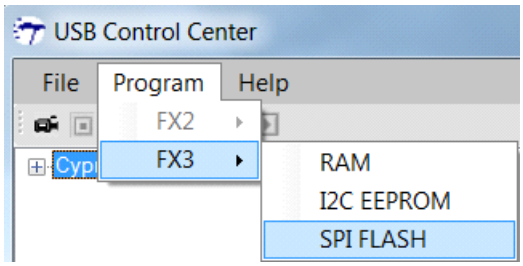
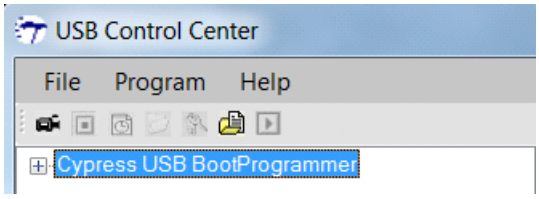
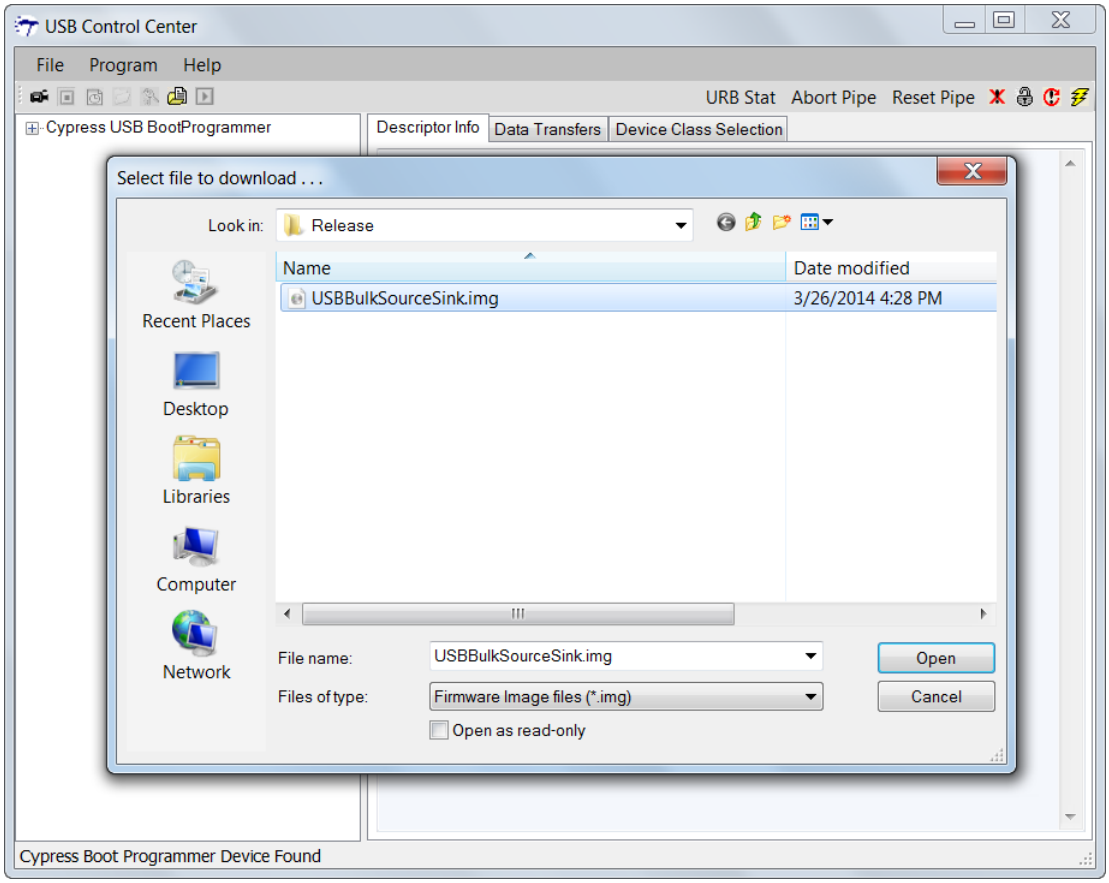


Figure 3-21. Device after Programmer Firmware Update



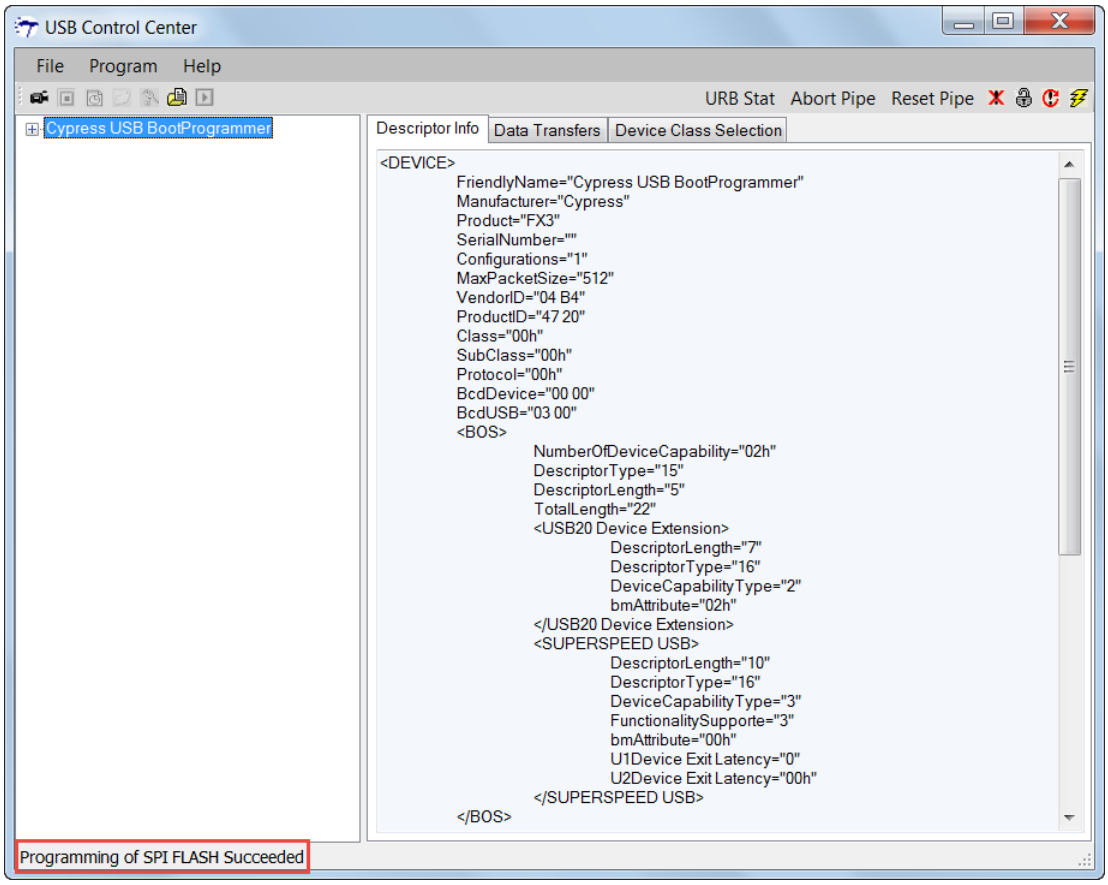
6. After the FX3 DVK board enumerates as **Cypress USB BootProgrammer**, the USB Control Center application prompts the user to select the firmware binary to download. Browse to the relevant img file and select the code image, as shown in Figure 3-22.

Figure 3-22. Select Firmware Image to Download to SPI Flash



7. The application downloads the firmware into the flash memory. The download status appears at the bottom left corner of the USB Control Center application, as shown in Figure 3-23.

Figure 3-23. SPI Firmware Download Update in USB Control Center



3.8.2 Booting from SPI Flash

After programming the firmware to SPI flash, set the PMODE[2:0] pins to 0Z1 to enable SPI boot. On the DVK board, this is done by configuring the jumpers and switches, as shown in Table 3-9.

Table 3-9. SPI Boot Jumper and Dip Switch Settings on FX3 DVK Board

PMODE Pin	Required PMODE Pin State	Jumper and Dip Switch Combination
PMODE2	0	1. Short pins 2-3 of jumper J98 2. SW25.3 set to ON
PMODE1	Z	1. No jumper installed on J97 2. SW25.2 – Don't care (the switch can be either turned ON or OFF)
PMODE0	1	1. Short pins 2–3 of jumper J96 2. SW25.1 set to OFF

Press RESET button SW8. This prompts the FX3 device to monitor the PMODE I/O pins again and to boot from the SPI interface.

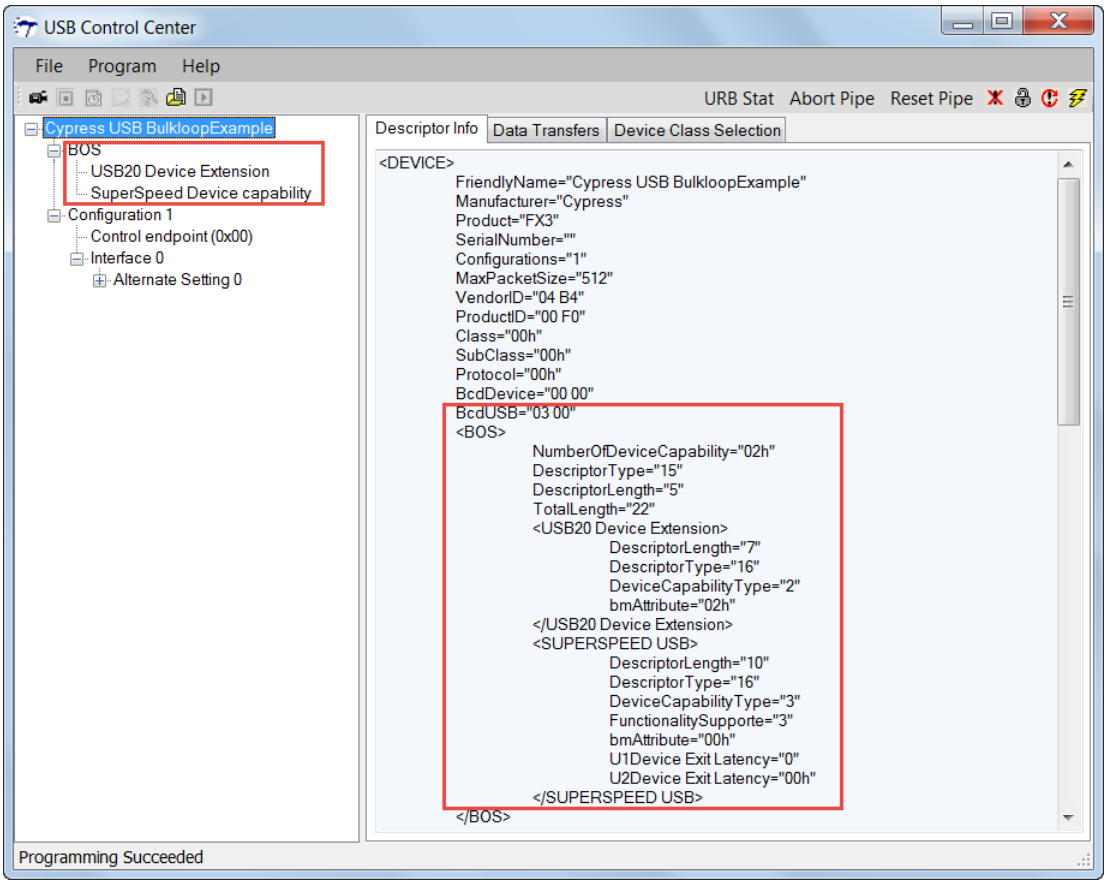
3.9 ADMUX Boot

The FX3 device can boot from the synchronous and asynchronous ADMux interface if the PMODE[2:0] pins are set to Z00 and Z01, respectively. For more details, see application note AN76405 - EZ-USB FX3 Boot Options.

3.10 USB 3.0 Enumeration

The FX3 DVK board bootloader operates as a USB 2.0 device. To enumerate as USB 3.0 device, a firmware image that implements a USB Superspeed device is downloaded to the FX3 board. The PC host requires a USB 3.0 host controller to run this code at Superspeed rate. The firmware examples illustrate how to write dual-speed applications, enabling the device automatically to operate at the maximum speed for USB 2.0 or USB 3.0 PC connection. You can confirm Superspeed operation of a downloaded application using the USB Control Center. After firmware download, the USB Control Center shows the list of descriptors (Figure 3-24) for a USB 3.0 device.

Figure 3-24. FX3 USB 3.0 Device Descriptor Example in Control Center



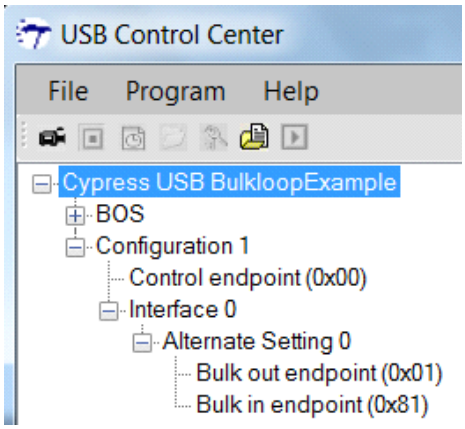
3.11 Example 2: Loopback of Data over Bulk Endpoints

USBBulkLoopAuto is the FX3 firmware example project located at `(C:\Program Files (x86)\Cypress\EZ-USB FX3 SDK\1.3\firmware\dma_examples)` that loops data over the FX3's bulk endpoints. A host application called Bulkloop is provided with the SuiteUSB to verify the loopback operation over bulk endpoints.

The SuiteUSB applications are available at `C:\Program Files (x86)\Cypress\EZ-USB FX3 SDK\1.3\bin`.
The Bulkloop host application is designed in both the C# and C++ frameworks. Follow these steps to test the Bulkloop firmware (USBBulkLoopAuto):

1. Import the USBBulkLoopAuto example project into the workspace and build. The steps to import a project are described in the section Import a Project on page 15. You can skip this step if the project is already imported to the Workspace.
2. Download the firmware binary image (USBBulkLoopAuto.img) by following the steps outlined in the section Download Firmware Image to FX3 RAM on page 43.
3. In the Windows Device Manager, the board appears as **Cypress USB BulkloopExample** under Universal Bus Controllers. If the board is not listed, bind the driver manually using the steps outlined in the Manual Installation of Cypress Driver on page 39.

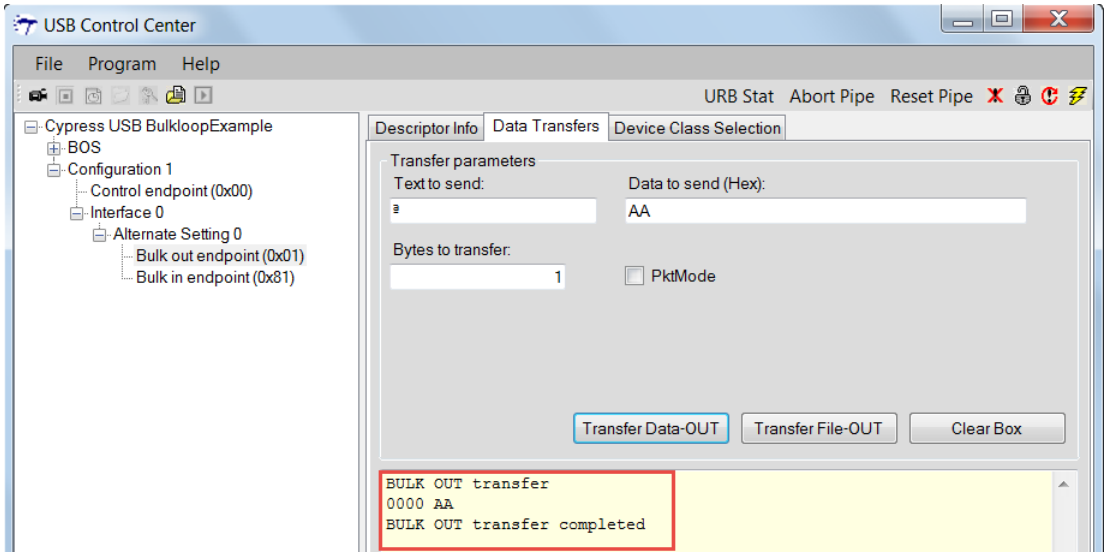
Figure 3-25. FX3 device enumeration



4. Take the following steps to verify the Bulkloop operation using the USB Control Center application.

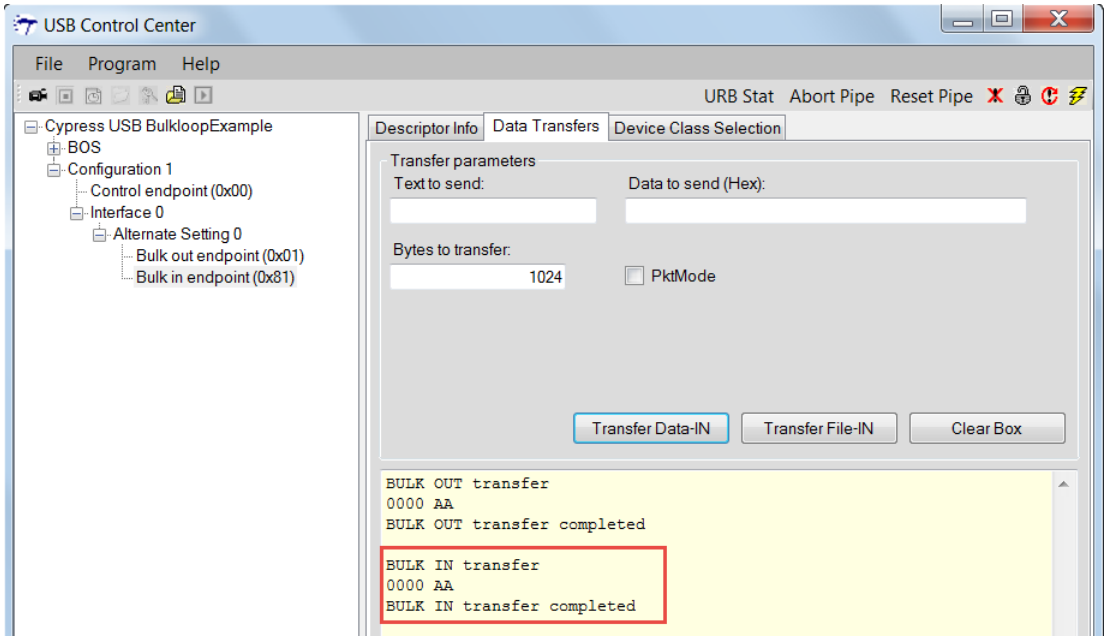
5. Select **Bulk out endpoint (0x1)** in the left pane of Control Center. Enter **AA** in **Data to send (Hex)**, and **1** in **Bytes to Transfer**, and click the **Transfer Data-OUT** button. Figure 3-26 illustrates these steps.

Figure 3-26. Sequence of steps for transferring data



6. To receive the same data on EP2 IN, select **Bulk in endpoint (0x81)**, enter **1024** (max packet size) in **Bytes to Transfer Length = 1024**, and click the **Transfer Data-IN** button to receive the same data, as shown in Figure 3-27.

Figure 3-27. Sequence of steps for receiving data

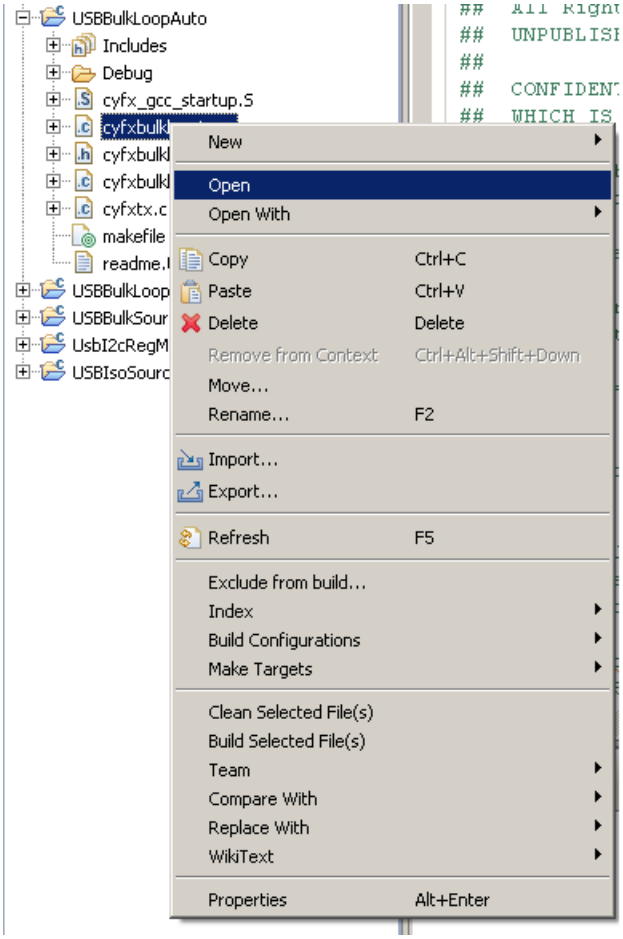


3.12 Modify the Bulkloop Firmware Example

The firmware examples can be modified and recompiled using the Eclipse IDE. The modified firmware binary can then be downloaded to FX3 RAM and tested using a SuiteUSB application. To demonstrate the modification, the USBBulkLoopAuto firmware example of FX3 SDK is used. The bulk endpoint number and its associated socket are modified here to simply demonstrate how to modify, recompile the firmware example, and test the functionality using a PC application.

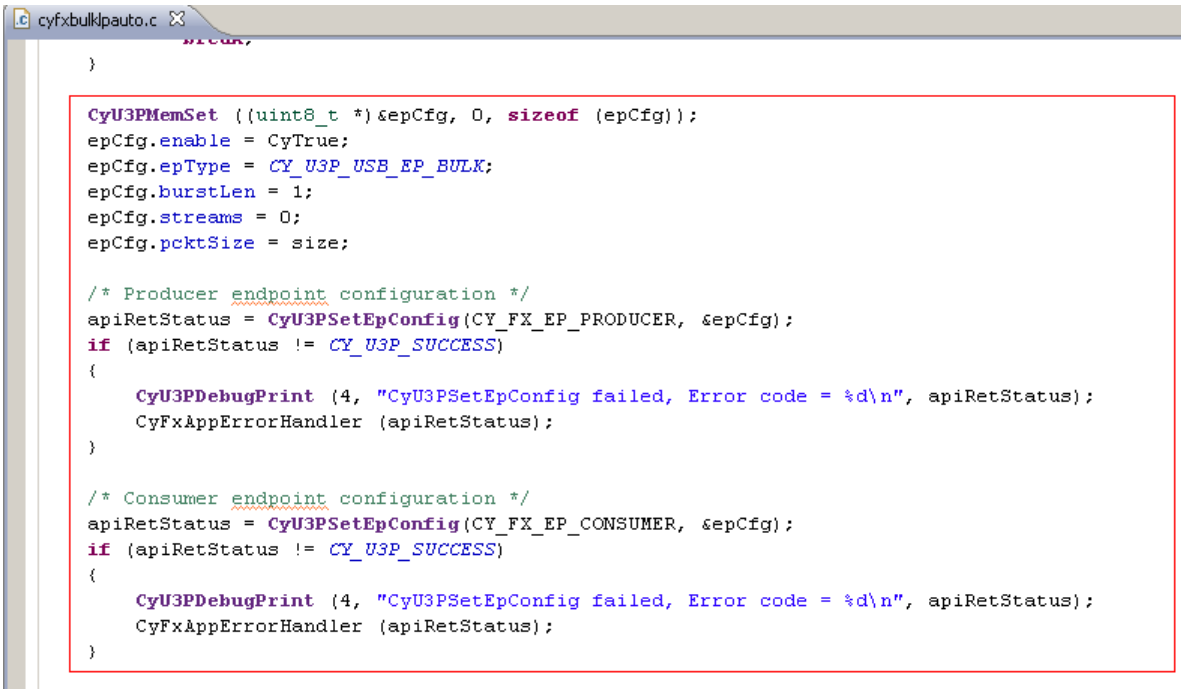
Select the **USBBulkLoopAuto** example in the Project Explorer window of the Eclipse IDE. In the list of .c files, select cyfxbulklpauto.c, as shown in Figure 3-28.

Figure 3-28. Selecting cyfxbulklpauto.c in Eclipse IDE



In this example, the bulk endpoint EP1 handles both IN and OUT transfers. The endpoint configuration is performed in the CyFxBulkAppInStart() function. The code snippet from Cyfxbulklpauto.c, shown in Figure 3-29, displays how EP1 is configured as a bulk endpoint type using API CyU3PSetEpConfig().

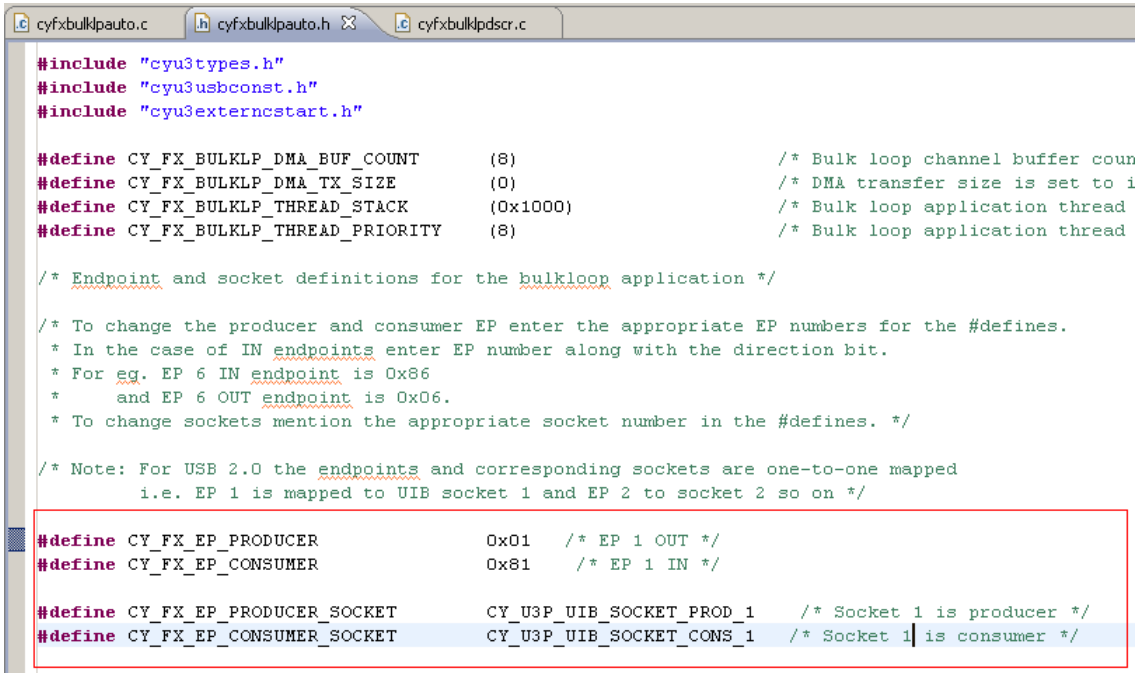
Figure 3-29. cyfxbulklpauto.c Code Snippet



```
cyfxbulklpauto.c
    }

    CyU3PMemSet ((uint8_t *)&epCfg, 0, sizeof (epCfg));
    epCfg.enable = CyTrue;
    epCfg.epType = CY_U3P_USB_EP_BULK;
    epCfg.burstLen = 1;
    epCfg.streams = 0;
    epCfg.pcktSize = size;

    /* Producer endpoint configuration */
    apiRetStatus = CyU3PSetEpConfig(CY_FX_EP_PRODUCER, &epCfg);
    if (apiRetStatus != CY_U3P_SUCCESS)
    {
        CyU3PDebugPrint (4, "CyU3PSetEpConfig failed, Error code = %d\n", apiRetStatus);
        CyFxAppErrorHandler (apiRetStatus);
    }

    /* Consumer endpoint configuration */
    apiRetStatus = CyU3PSetEpConfig(CY_FX_EP_CONSUMER, &epCfg);
    if (apiRetStatus != CY_U3P_SUCCESS)
    {
        CyU3PDebugPrint (4, "CyU3PSetEpConfig failed, Error code = %d\n", apiRetStatus);
        CyFxAppErrorHandler (apiRetStatus);
    }
```

The endpoint number and the associated socket are defined in .h using the macros CY_FX_EP_PRODUCER, CY_FX_EP_CONSUMER, CY_FX_EP_PRODUCER_SOCKET, and CY_FX_EP_CONSUMER_SOCKET. Figure 3-30 shows the code snippet defined in this file. To understand sockets, refer to the “FX3 Terminology” section in the application note AN75705 – Getting Started with EZ-USB FX3.

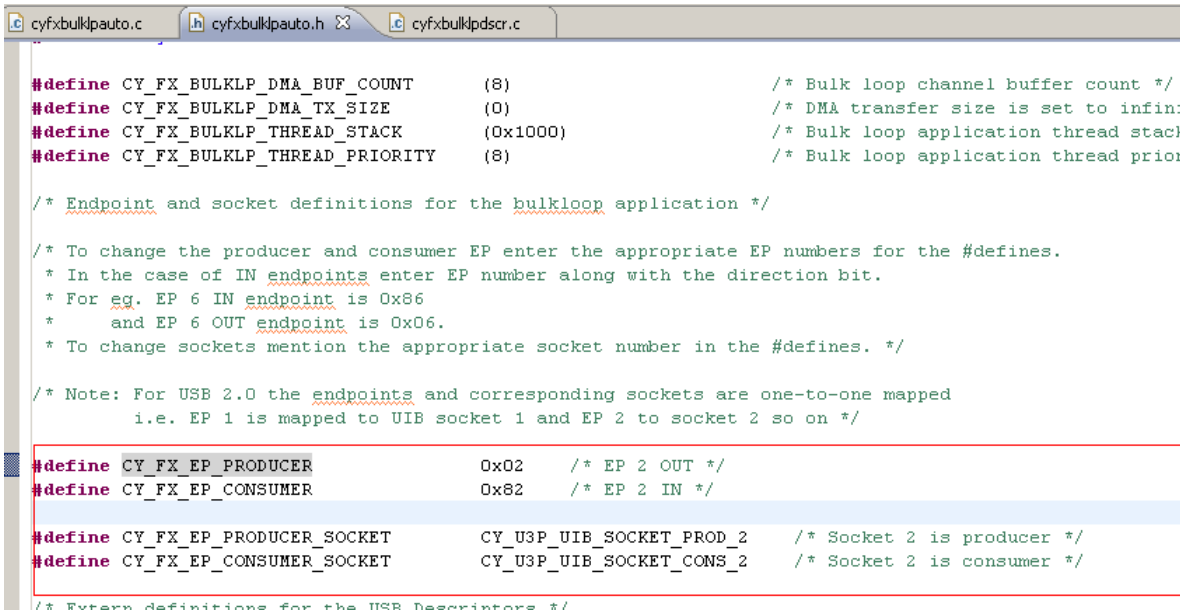
Figure 3-30. Endpoint and Socket Definitions in cyfxbulklpauto.h



```
cyfxbulklpauto.c | cyfxbulklpauto.h | cyfxbulklpdscr.c

#include "cyu3types.h"
#include "cyu3usbconst.h"
#include "cyu3externcstart.h"

#define CY_FX_BULKLP_DMA_BUF_COUNT      (8)                     /* Bulk loop channel buffer coun
#define CY_FX_BULKLP_DMA_TX_SIZE        (0)                     /* DMA transfer size is set to i
#define CY_FX_BULKLP_THREAD_STACK       (0x1000)                /* Bulk loop application thread
#define CY_FX_BULKLP_THREAD_PRIORITY    (8)                     /* Bulk loop application thread

/* Endpoint and socket definitions for the bulkloop application */

/* To change the producer and consumer EP enter the appropriate EP numbers for the #defines.
 * In the case of IN endpoints enter EP number along with the direction bit.
 * For eg. EP 6 IN endpoint is 0x86
 *     and EP 6 OUT endpoint is 0x06.
 * To change sockets mention the appropriate socket number in the #defines. */

/* Note: For USB 2.0 the endpoints and corresponding sockets are one-to-one mapped
         i.e. EP 1 is mapped to UIB socket 1 and EP 2 to socket 2 so on */

#define CY_FX_EP_PRODUCER               0x01    /* EP 1 OUT */
#define CY_FX_EP_CONSUMER               0x81    /* EP 1 IN */

#define CY_FX_EP_PRODUCER_SOCKET        CY_U3P_UIB_SOCKET_PROD_1    /* Socket 1 is producer */
#define CY_FX_EP_CONSUMER_SOCKET        CY_U3P_UIB_SOCKET_CONS_1    /* Socket 1 is consumer */
```

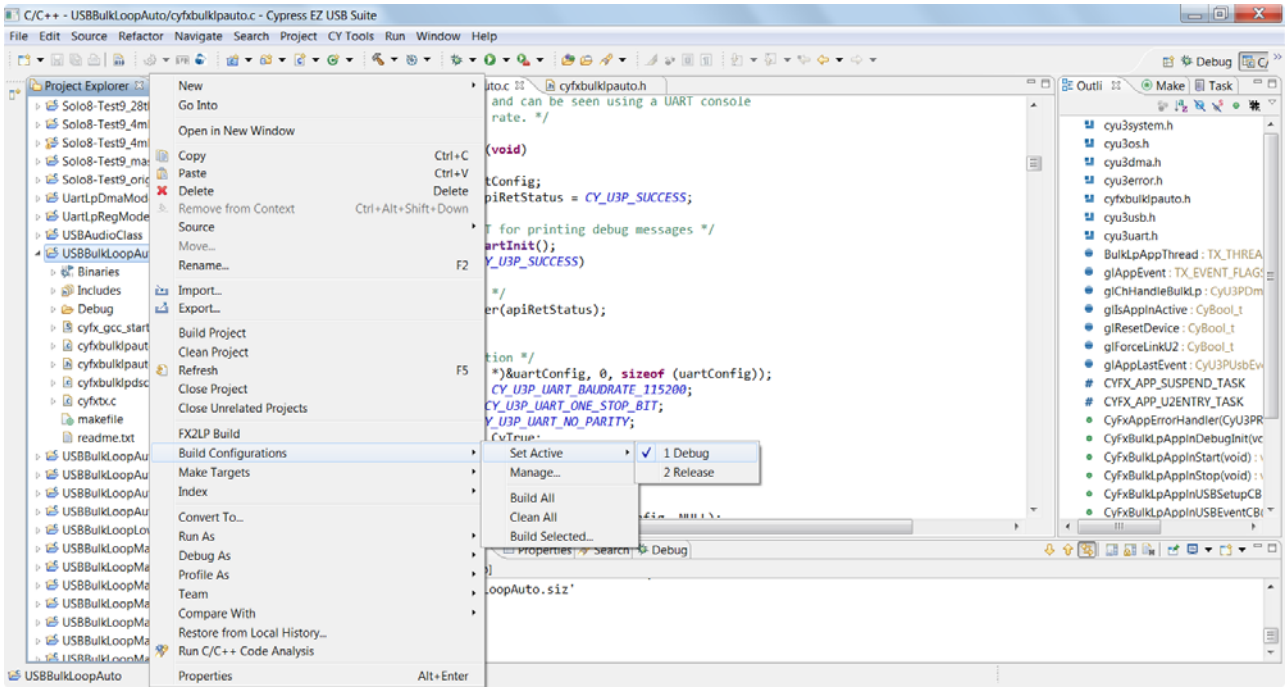
Using these macros, the endpoint number and associated socket are changed from EP1 to EP2 (Figure 3-31). Now EP2 acts as the Bulk IN and OUT endpoints in the modified firmware.

Figure 3-31. Endpoint Number and Associated Socket Modified from EP1 to EP2



Build the modified firmware example. You can also build a project as shown in Figure 3-32. Build the firmware example in both debug and release modes by choosing **Build Configuration > Set Active > Debug/Release**.

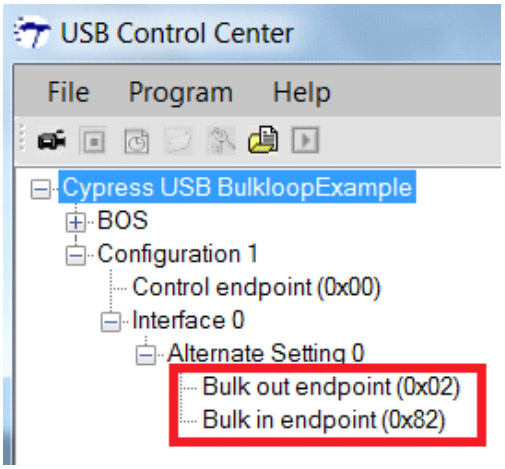
Figure 3-32. Selecting Build Configuration in Eclipse IDE



Note Release code images are smaller than Debug code images. The debug mode image contains additional debug symbols to allow the user to perform step-by-step debugging using JTAG hardware.

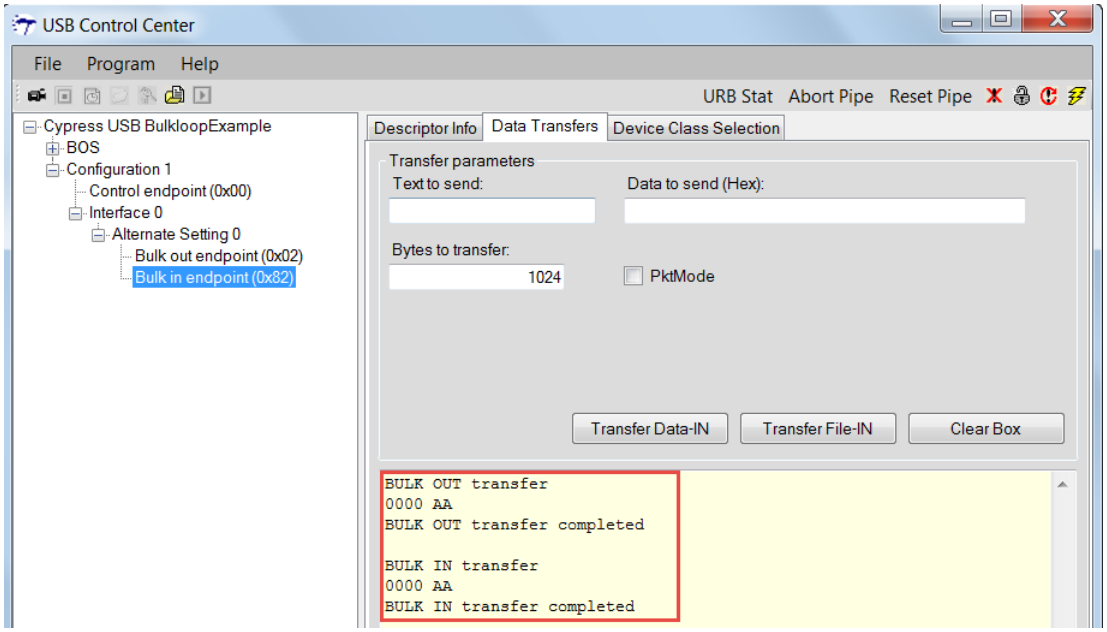
Follow the procedure outlined in USB Boot on page 43 to download the firmware image to FX3 RAM using Control Center. The FX3 DVK board enumerates as Cypress USB BulkloopExample in Control Center. Observe that the modified firmware example shows EP2 as both Bulk IN and OUT in Control Center (Figure 3-33).

Figure 3-33. Bulkloop Device with EP2 in Control Center



Select **Bulk out endpoint (0x2)** in the left pane of Control Center. Enter **Data to send**, and **1** in **Bytes to transfer**, and click the **Transfer Data-OUT** button. To receive the same data on EP2 IN, select **Bulk in endpoint (0x82)**, **Bytes to transfer**, and click the **Transfer Data-IN** button to receive the same data. Figure 3-34 summarizes the sequence.

Figure 3-34. Bulkloop OUT and IN transfer on EP2



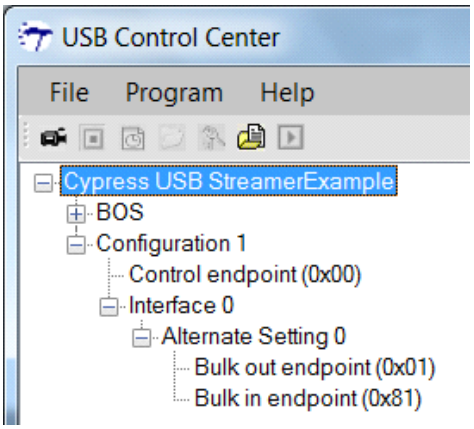
3.13 Example 3: Measurement of Throughput Using USB 3.0 Bulk Transfers

USBBulkSourceSink is the FX3 firmware example project (`C:\Program Files (x86)\Cypress\EZ-USB FX3 SDK\1.3\firmware\basic_examples`) that measures the throughput of USB 3.0 bulk transfers using FX3. USBBulkSourceSink firmware generates data internal to FX3. A host application called Streamer is provided with the SuiteUSB to measure the throughput over bulk/ISO/interrupt endpoints.

The SuiteUSB applications are available at `C:\Program Files (x86)\Cypress\EZ-USB FX3 SDK\1.3\bin`. The Streamer host application is designed in both the C# and C++ framework. The C++ version of the Streamer application is recommended to achieve higher throughput numbers. Follow these steps to test the Bulk source sink firmware (USBBulkSourceSink):

1. Import the USBBulkSourceSink example project into the workspace and build. The steps to import a project are described in the section Import a Project on page 15.You can skip this step if this project is already imported to the Workspace.
2. Download the firmware binary image (USBBulkSourceSink.img) by following the steps outlined in the section Download Firmware Image to FX3 RAM on page 43.
3. In the Windows Device Manager, the board appears as Cypress USB StreamerExample under Universal Bus Controllers. If the board is not listed, bind the driver manually using the steps outlined in the section Manual Installation of Cypress Driver on page 39.

Figure 3-35. FX3 device enumeration

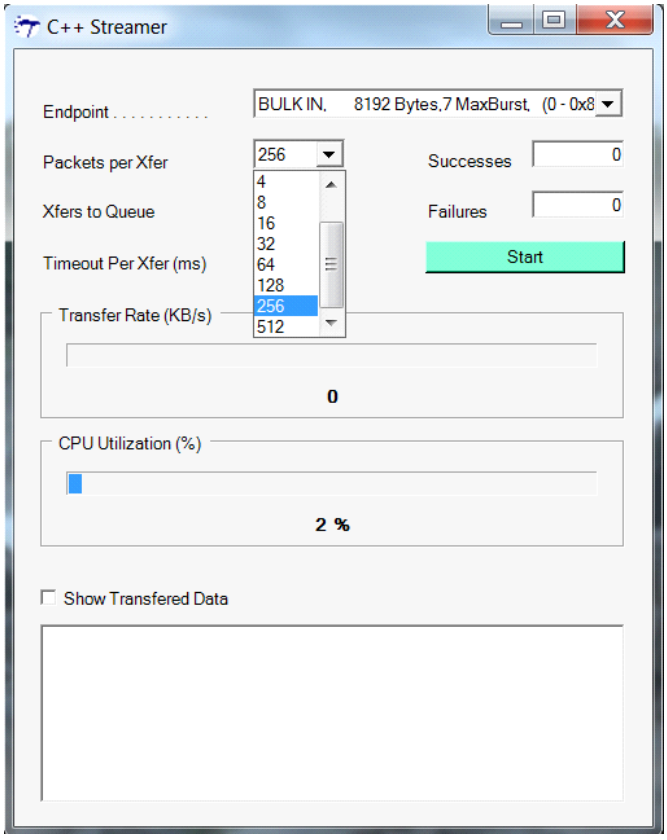


4. Run Streamer.exe from the location `C:\Program Files (x86)\Cypress\EZ-USB FX3 SDK\1.3\application\cpp\streamer\x86\Release`. As shown in Figure 3-36, choose 256 as **Packets per Xfer** and 32 as **Xfers to Queue** from the dropdown list available on the Streamer application.

   Packets per Xfer: A transfer is a collection of packets for one data set. A greater number of packets per each transfer reduces the USB overhead and achieves higher data throughput.

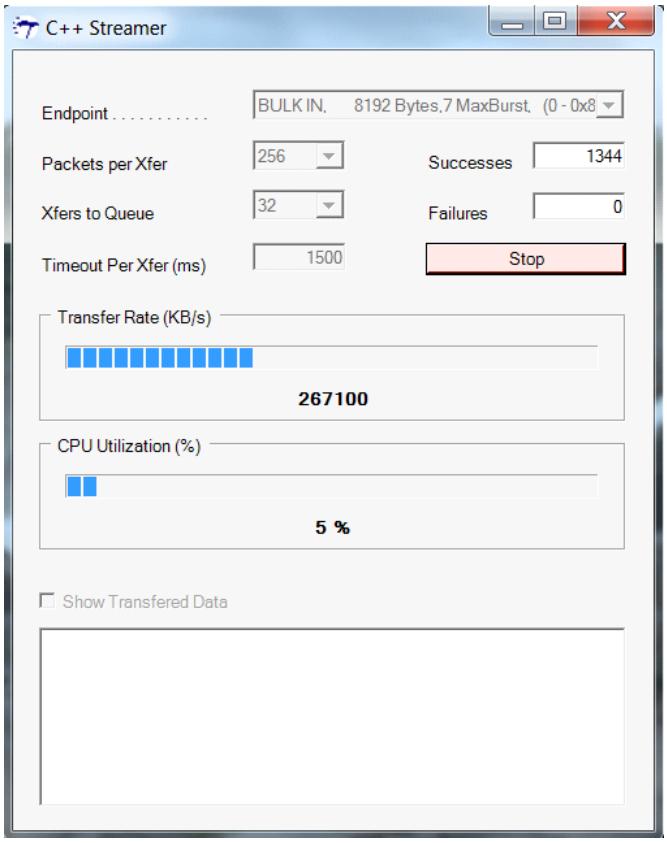
   Xfers to Queue: This setting initiates multiple transfers and adds them to the task queue. This reduces the latency between successive transfers on the host application side. Therefore, queuing more transfers gives a higher data rate.

Figure 3-36. Streamer.exe



5. Click the **Start** button to see the throughput of USB 3.0 bulk IN transfers. The throughput shown with the default bulk source sink firmware image is about 267 MBps (Figure 3-37).

Figure 3-37. Throughput of USB 3.0 Bulk IN transfers



The SuperSpeed bulk throughput depends on the burst size, buffer size, and the number of buffers. The bulk source sink firmware can be optimized further to achieve throughput at about 450 MBps. Refer to AN86947 - Optimizing USB 3.0 Throughput with EZ-USB® FX3™ for more details about the parameters to be optimized in the bulk source sink firmware.

This application note (AN86947) lists the USB 3.0 throughput numbers only with the data generated internal to FX3. The USB 3.0 throughput numbers will be slightly lower when data is coming from an external device into FX3 through GPIF II. Refer to AN65974 - Designing with the EZ-USB® FX3™ Slave FIFO Interface for the throughput numbers when an FPGA supplies data to FX3 via the GPIFII interface.

4. Hardware

This chapter describes the different hardware interfaces available on the FX3 DVK. It explains the following DVK board components:

- Power supply
- USB 3.0 Micro-B receptacle connector
- Clocking mechanism using on-board crystal
- GPIFII connector
- Reset circuit
- Serial interfaces – I^2S, I^2C, SPI, UART, and JTAG.

4.1 Power Supply

The FX3 DVK board can be powered in two ways:

- Self-power: Use the external 5-V power adapter supplied with the kit to power the DVK board.
- Bus-power: The board can be powered using the USB 3.0 cable. Connect the USB cable supplied with the kit between the PC host and the J48 USB connector on board. The PC host supplies power to the DVK board through the USB cable. Short pins 1–3 or 2–4 of jumper J53 in bus-power mode.

FX3 supports several I/O interfaces including I^2S, I^2C, SPI, UART and GPIF II. These interfaces are capable of operating at different voltage levels. The voltage levels for FX3 I/O power domains can be selected using jumpers as shown in Table 4-1.

Table 4-1. Power Supply Domain

Power Domain	Jumper	Jumper positions	Voltage levels
VIO1	J136	1–6 (labeled V1P8 on board)	1.8 V
		2–5 (labeled V2P5 on board)	2.5 V
		3–4 (labeled V3P3 on board)	3.3 V
VIO2	J144	1–6 (labeled V1P8 on board)	1.8 V
		2–5 (labeled V2P5 on board)	2.5 V
		3–4 (labeled V3P3 on board)	3.3 V
VIO3	J145	1–6 (labeled V1P8 on board)	1.8 V
		2–5 (labeled V2P5 on board)	2.5 V
		3–4 (labeled V3P3 on board)	3.3 V
VIO4	J146	1–6 (labeled V1P8 on board)	1.8 V
		2–5 (labeled V2P5 on board)	2.5 V
		3–4 (labeled V3P3 on board)	3.3 V

Table 4-1. Power Supply Domain

Power Domain	Jumper	Jumper positions	Voltage levels
VIO5	J134	1–8 (labeled V1P2 on board)	1.2 V
		2–7 (labeled V1P8 on board)	1.8 V
		3–6 (labeled V2P5 on board)	2.5 V
		4–5 (labeled V3P3 on board)	3.3 V
REG_VBATT	J143	1–6 (labeled V2P5 on board)	2.5 V
		2–5 (labeled V3P3 on board)	3.3 V
		3–4 (labeled V5P0 on board)	5 V
CVDDQ	J135	1–2 (labeled V1P8 on board	1.8 V
		3–4 (labeled V3P3 on board)	3.3 V

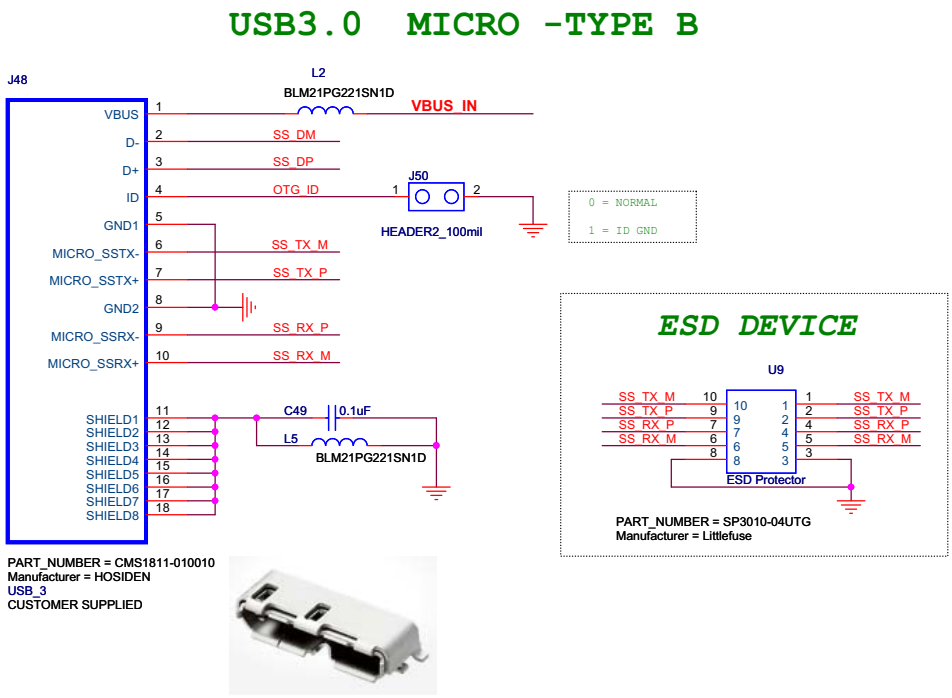
Notes

- The FX3 device also has additional power domains (U3TXVDDQ, U3RXVDDQ, AVDD, and VDD). These are fixed at 1.2 V on the FX3 DVK.
- Refer to the “Pin Description” section of the EZ-USB FX3 SuperSpeed USB Controller datasheet for details on how the I/O in each power domain can be configured.

4.2 USB Receptacle

A standard Micro-B receptacle is used on the FX3 DVK board. Figure 4-1 shows the USB 3.0 pins (SS_TX_M, SS_TX_P, SS_RX_P, and SS_RX_M) and USB 2.0 pins (OTG_ID, D+, and D–) available on the J48 USB connector. The FX3 DVK board can be bus-powered using the VBUS pin on the connector. The USB3.0 and USB2.0 lines go through an ESD protection device.

Figure 4-1. USB3.0 Micro B Connector and ESD Device



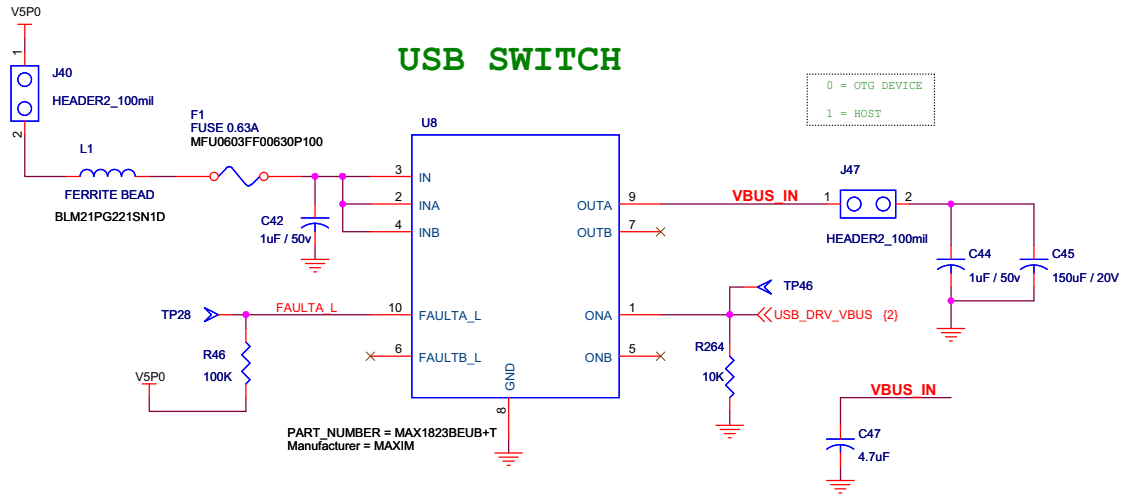
The DVK board supports USB 3.0 peripheral and Hi-Speed OTG functionality using the J48 connector. Table 4-2 shows the jumper settings required to operate in USB 2.0 OTG device or Host mode.

Table 4-2. OTG/Host Mode Jumper Settings

Jumper	Pin Position	Mode: Description
J40	Short	OTG Host: To supply VBUS to the connected peripheral as OTG host from the power adapter (V5P0).
J50	Short	OTG Host: When connecting a USB device to the FX3 OTG host with a USB 2.0 Micro-B to Micro-A receptacle cable the OTG_ID pin can be grounded using this jumper. FX3 then acts as OTG Host by default.
	Open	OTG Device: The OTG_ID pin state depends upon the pin state at J48. The role of FX3 (as the OTG A or B device) depends on the OTG cable (Micro-A to Micro-B type) orientation.
J47	Short	OTG Host: To select proper capacitance on VBUS between the OTG host manually short this VBUS_IN jumper.
	Open	OTG device: To select proper capacitance on VBUS as the OTG B device.
J100	1–2	Routes the CTL_4 signal to the Samtec connector (GPIF II interface)
	2–3	OTG Host/OTG Device: USB_DRV_VBUS line is connected to the CTL_4 signal. The I/O line can turn the power (VBUS_IN) ON or OFF using firmware.

To operate the DVK in USB 3.0 peripheral mode, refer to the default jumper settings mentioned in Table 3-1 on page 35 and follow the procedure outlined in USB 3.0 Enumeration on page 54.

Figure 4-2. OTG and Host Mode Power Configuration Jumpers



4.3 Clocking Mechanism for FX3

FX3 allows either a crystal to be connected between the XTALIN and XTALOUT pins or an external clock to be connected at the CLKIN pin. The XTALIN, XTALOUT, and CLKIN I/O pins can be left unconnected if they are not used.

FX3 supports a 19.2-MHz crystal, while the supported external clock frequencies are 19.2, 26, 38.4, and 52 MHz.

The FSLC[2:0] pins are configured to select the crystal- or clock-frequency option. The configuration options are shown in Table 4-3.

Table 4-3. Crystal/Clock Frequency Selection

FSLC[2]	FSLC[1]	FSLC[0]	Crystal/Clock Frequency
0	0	0	19.2-MHz crystal
1	0	0	19.2-MHz input CLK
1	0	1	26-MHz input CLK
1	1	0	38.4-MHz input CLK
1	1	1	52-MHz input CLK

On the DVK board, the clock for the FX3 is provided through an on-board 19.2-MHz crystal connected to the XTALIN and XTALOUT pins of FX3. The FSLC[2:0] lines of FX3 are tied to ground to select the 19.2-MHz crystal for clocking.

Figure 4-3. Crystal Circuit

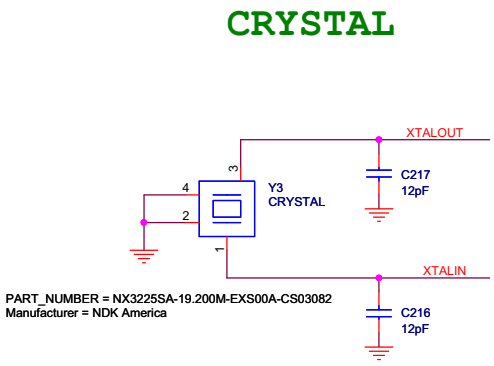
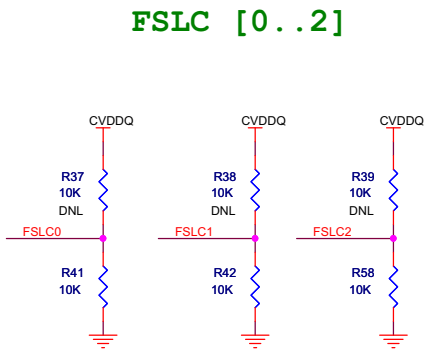


Figure 4-4. FSCL[0:2] Lines Pulled to Ground



Note For the 19.2-MHz crystal option, FSLC[0:2] pins are tied to ground and hence R37, R38, and R39 are not populated on the board.

4.4 GPIF II Connector

The GPIF II interface enables functionality similar to but more advanced than FX2LP's GPIF. It is a programmable state machine that enables a flexible interface, which functions either as a master or a slave using industry standard or proprietary interfaces. You can implement both parallel and serial interfaces with GPIF II. The DVK board provides a Samtec expansion connector to interface with external processors, ASICs, DSPs, or FPGAs. The GPIF II lines going to the Samtec connector also come out on Mictor debug connectors, J1 and J2. You can connect logic analyzer pods to J1 and J2 for debugging.

Figure 4-5 shows the Samtec expansion connector on the FX3 DVK board.

Figure 4-5. Samtec and Mictor Debug Connectors on DVK

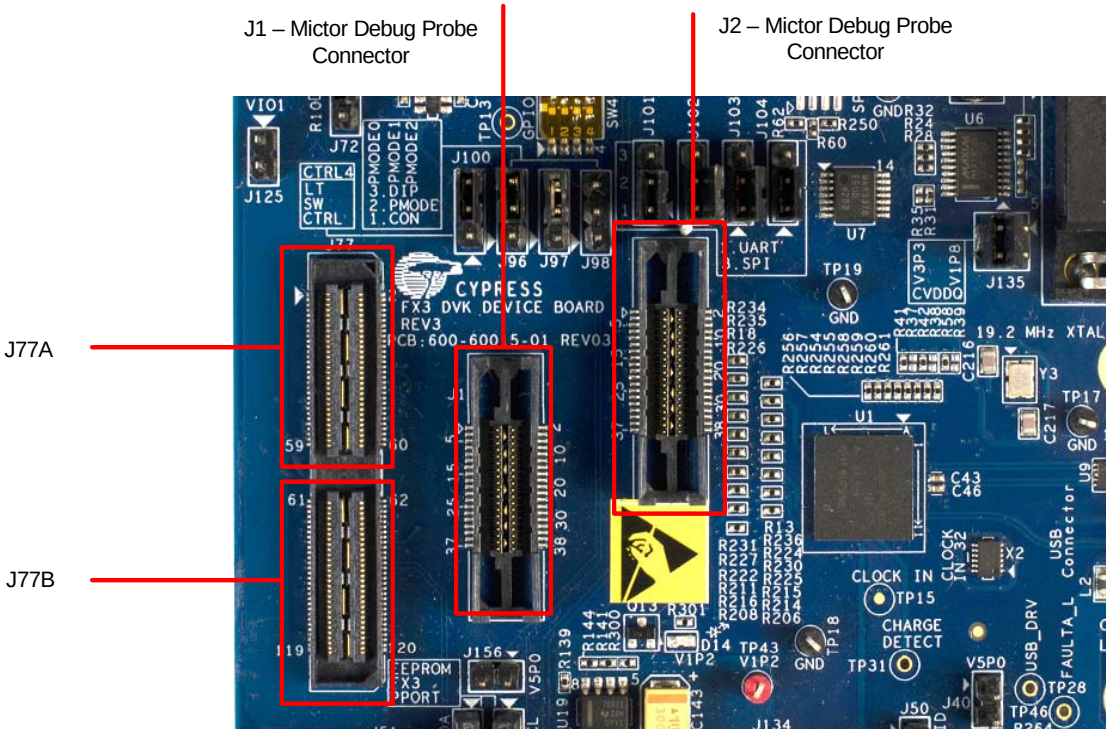


Figure 4-6 and Figure 4-7 show the schematic diagrams of the Samtec expansion connector with the GPIF II signals.

Figure 4-6. Samtec Expansion Connector Circuit1

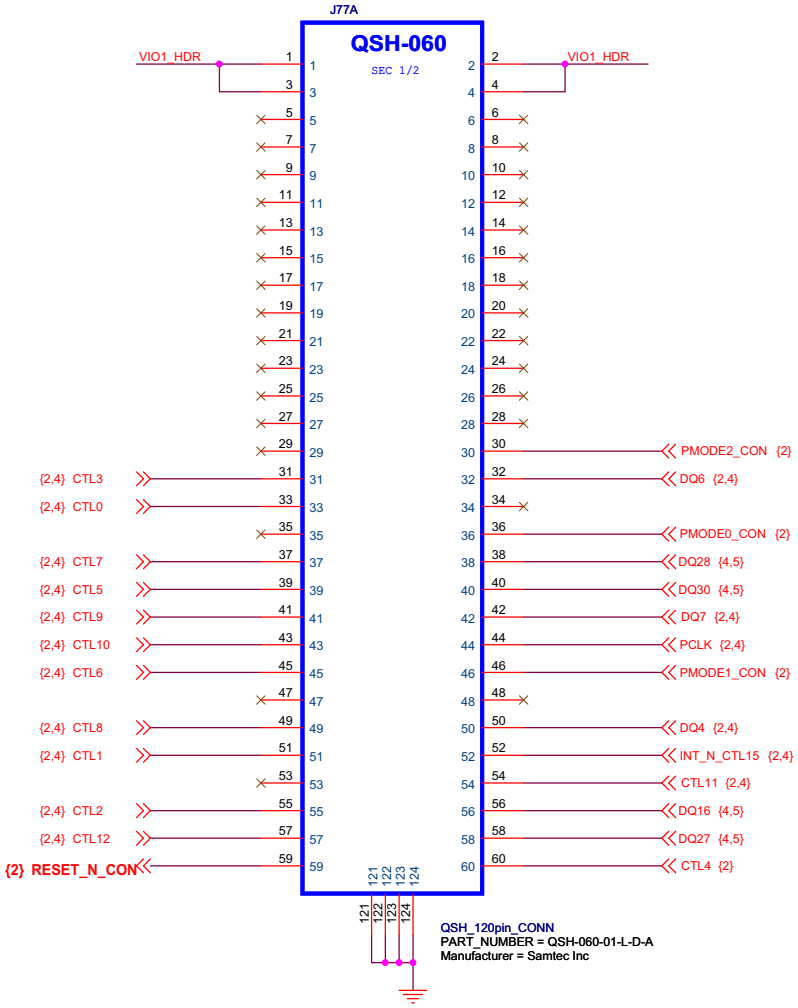
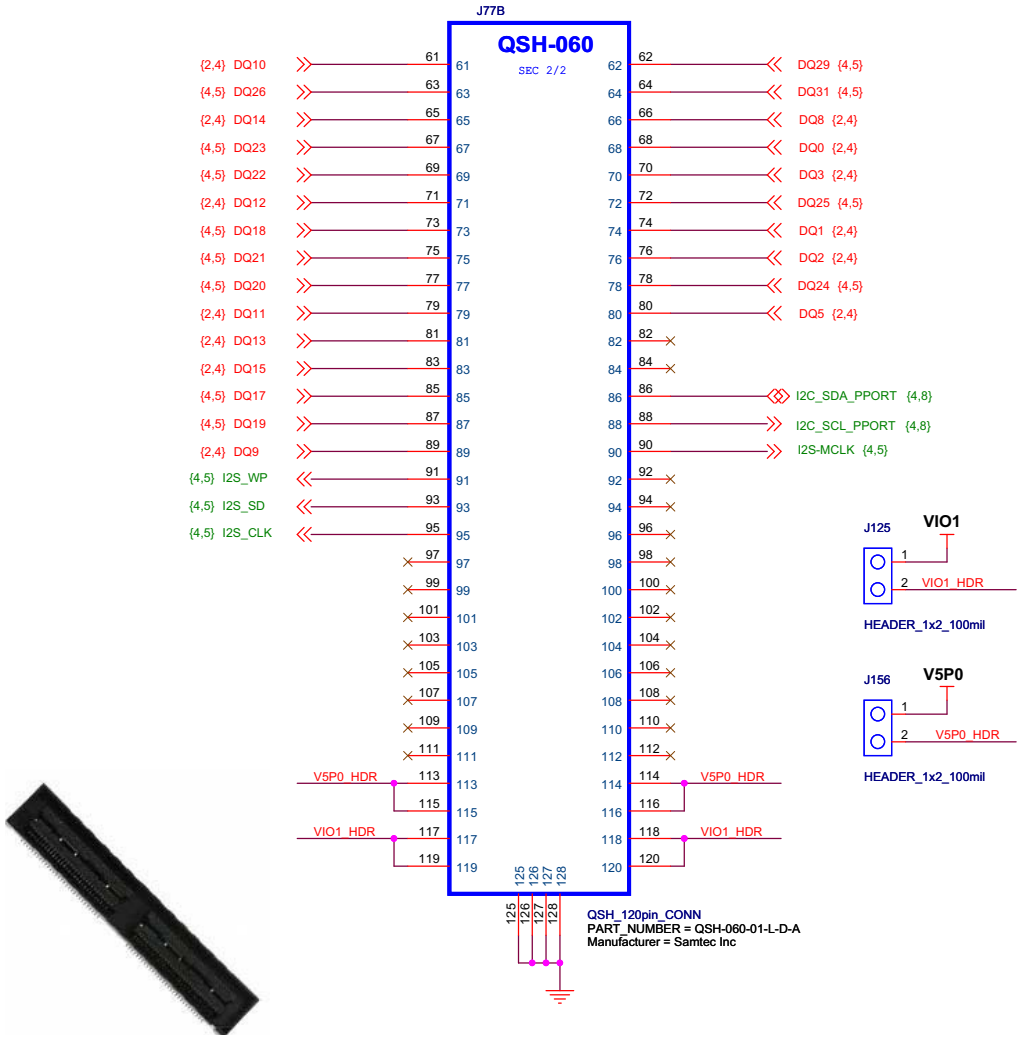


Figure 4-7. Samtec Expansion Connector Circuit 2



Note GPIF II I/O pins are shared between the J77A and J77B connectors. In addition I^2S and I^2C interface I/O lines are brought out onto the J77B connector.

Table 4-4 shows the detailed pinout of the GPIF II interface on the Samtec expansion connector.

Table 4-4. FX3 GPIF II Signal Pinout on Samtec Connector

Connector J77 Pin No.	FX3 Signals	GPI/O
68	D0	GPIO[0]
74	D1	GPIO[1]
76	D2	GPIO[2]
70	D3	GPIO[3]
50	D4	GPIO[4]
80	D5	GPIO[5]
32	D6	GPIO[6]
42	D7	GPIO[7]

Table 4-4. FX3 GPIF II Signal Pinout on Samtec Connector *(continued)*

Connector J77 Pin No.	FX3 Signals	GPI/O
66	D8	GPIO[8]
89	D9	GPIO[9]
61	D10	GPIO[10]
79	D11	GPIO[11]
71	D12	GPIO[12]
81	D13	GPIO[13]
65	D14	GPIO[14]
83	D15	GPIO[15]
44	CLK	GPIO[16]
33	CTL0	GPIO[17]
51	CTL1	GPIO[18]
55	CTL2	GPIO[19]
31	CTL3	GPIO[20]
60	CTL4	GPIO[21]
39	CTL5	GPIO[22]
45	CTL6	GPIO[23]
37	CTL7	GPIO[24]
49	CTL8	GPIO[25]
41	CTL9	GPIO[26]
43	CTL10	GPIO[27]
54	CTL11	GPIO[28]
57	CTL12	GPIO[29]
36	PMODE0	GPIO[30]
46	PMODE1	GPIO[31]
30	PMODE2	GPIO[32]
52	INT	int#
59	RESET	reset#
56	D16	GPIO[33]
85	D17	GPIO[34]
73	D18	GPIO[35]
87	D19	GPIO[36]
77	D20	GPIO[37]
75	D21	GPIO[38]
69	D22	GPIO[39]
67	D23	GPIO[40]
78	D24	GPIO[41]
72	D25	GPIO[42]
63	D26	GPIO[43]
58	D27	GPIO[44]
38	D28_UART-RTS	GPIO[46]

Table 4-4. FX3 GPIF II Signal Pinout on Samtec Connector *(continued)*

Connector J77 Pin No.	FX3 Signals	GPI/O
62	D29_UART-CTS	GPIO[47]
40	D30_UART-TX	GPIO[48]
64	D31_UART-RX	GPIO[49]
95	I2S-CLK	GPIO[50]
93	I2S-SD	GPIO[51]
91	I2S-WS	GPIO[52]
90	I2S-MCLK	GPIO[57]
88	I2C-SCL	GPIO[58]
86	I2C-SDA	GPIO[59]

As shown in Figure 4-7, a jumper J125 can be inserted to enable the VIO1 (1.8 V, 2.5 V, and 3.3 V – see Table 4-1) power to the Samtec connector (GPIF II interface). Similarly, J156 can be used to connect power to the Samtec connector.

The CTRL_4 line can be routed either to the J77A Samtec connector (as a GPIF II interface I/O line) or it can be used to enable or disable the on-board USB switch. As explained in USB Receptacle on page 66, the J100 jumper can be used to control the OTG power. Figure 4-8 shows how the selection for pin2-CTL4_SW is made.

Figure 4-8. CTRL_4 Selection Jumper

As shown in Figure 4-8, if a jumper connects J100 pins 1 and 2, the CTRL_4 line connects to the GPIF II interface on the Samtec connector (J77A). If pins 2 and 3 are connected with a jumper, the CTRL_4 line routes to the USB switch (U8) for OTG power control (VBUS).

4.4.1 Interconnecting to the GPIF II Connector

You can connect the external processor board to the FX3 device using an interconnection board that mates with the Samtec expansion connector J77. The mating connector is a QTH-060 series Samtec connector. The following figures show the clearances and location of the Samtec connector on the FX3 DVK board. Note that these dimensions apply only to Rev 02 and Rev 03 releases of the DVK board.

Figure 4-9 shows the clearances between the Samtec connector and nearby components. The white colored outline on the J77 connector is the outline of the connector body.

Figure 4-9. Clearance to Samtec Connector (all dimensions in mils)

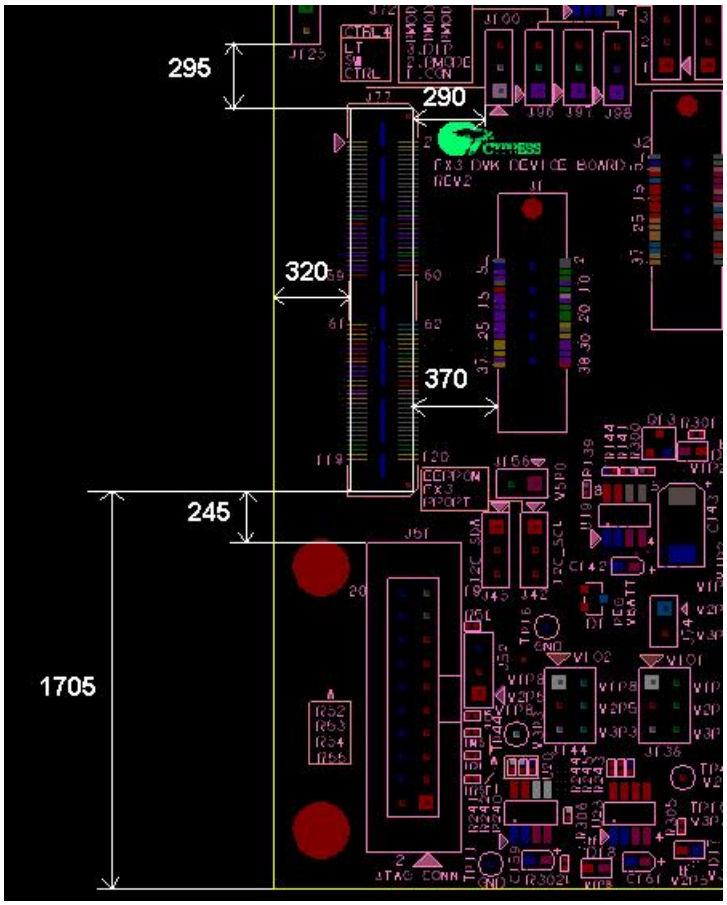


Figure 4-10. Distance from Center of Pin 1 to Top Edge of Board (all dimensions in mils)

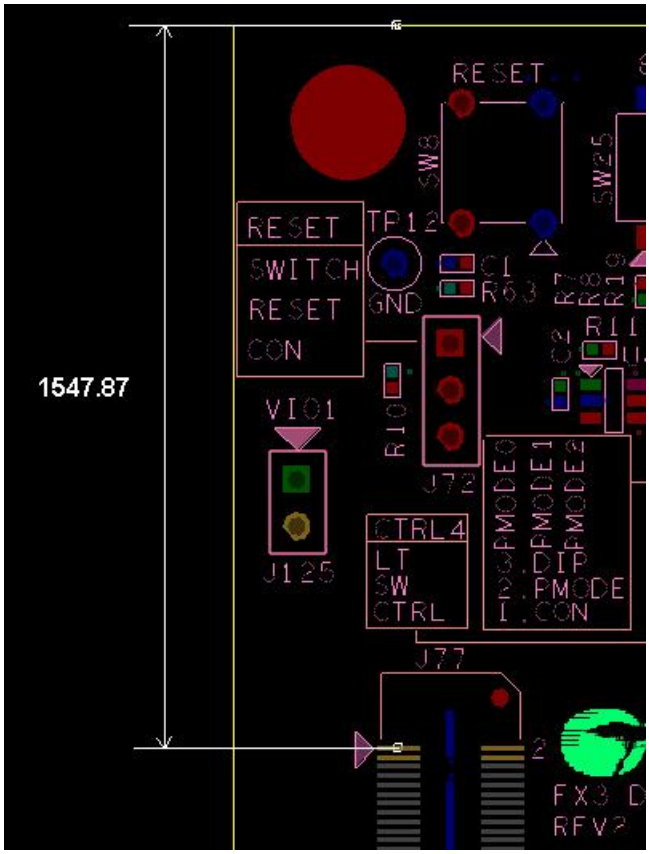
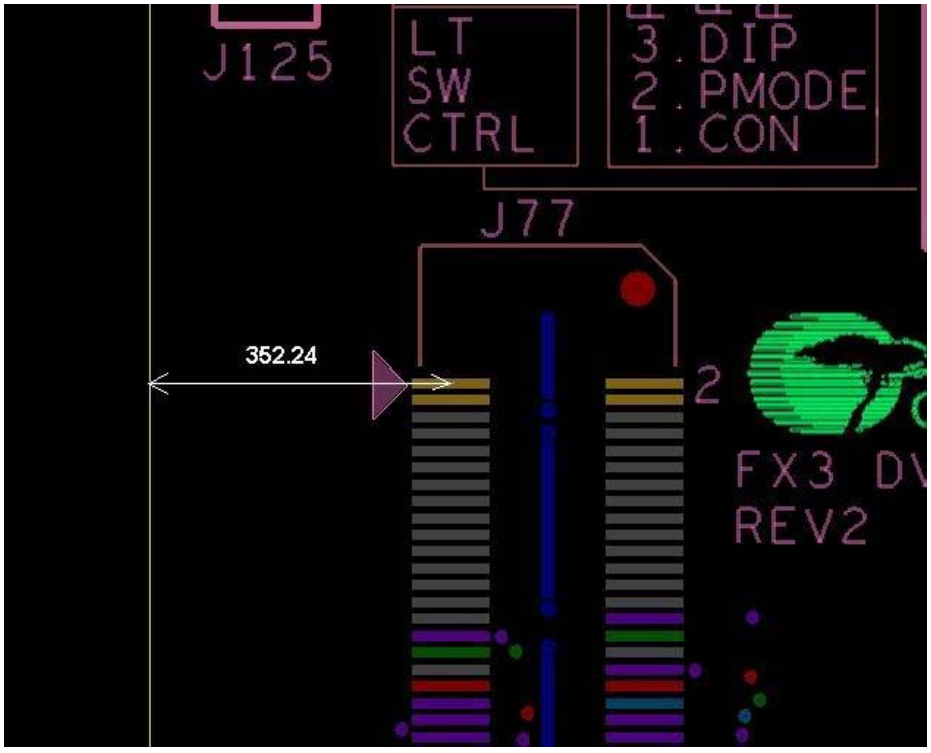


Figure 4-11. Distance from Center of Pin 1 to Left Edge of Board (all dimensions in mils)



4.4.2 FX3 DVK Interconnect Boards

Three interconnect boards are available to interface the FX3 GPIF II (using the Samtec connector, J77) to external devices.

- FMC interconnect board: Connects Xilinx FPGA development board to the FX3 DVK.
- HSMC interconnect board: Connects Altera FPGA development board to the FX3 DVK.
- Aptina interconnect board: Connects Aptina image sensor headboard to the FX3 DVK.

How do I get an interconnect board?

Contact Cypress at fx3@cypress.com to obtain an interconnection board.

How do I connect these interconnect boards to the FX3 DVK?

The connection of the Xilinx FPGA to the FX3 DVK is shown in the Design Example1 section of the application note AN65974 - Designing with the EZ-USB® FX3™ Slave FIFO Interface

The connection of the Altera FPGA to the FX3 DVK is shown in the Design Example2 section of the application note AN65974 - Designing with the EZ-USB® FX3™ Slave FIFO Interface

The connection of Aptina image sensor headboard is shown in the Hardware Setup section of the application note AN75779 - How to Implement an Image Sensor Interface with EZ-USB® FX3™ in a USB Video Class (UVC) Framework

Where can I find the schematics and gerber files of these interconnect boards?

For the FMC interconnect board, visit http://www.cypress.com/?app=forum&id=167&rID=84748

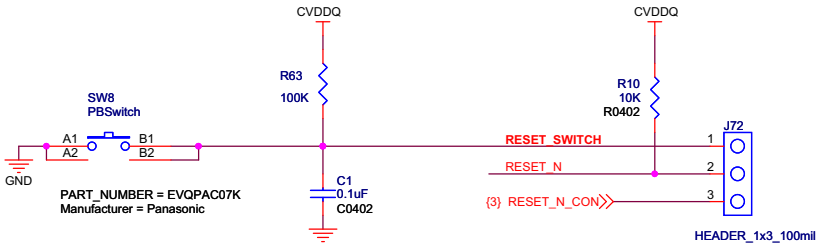
For the HSMC interconnect board, visit http://www.cypress.com/?app=forum&id=167&rID=84756

For the Aptina interconnect board, contact Cypress at fx3@cypress.com.

4.5 Reset Circuit

The FX3 device can be reset either from an external processor wired to the Samtec connector or from an on-board push button. This selection can be made on J72; see Figure 4-12.

Figure 4-12. Reset Circuit and Selection Headers



Based on the jumper setting on J72, either this RESET_N signal goes to the FX3 device (J72 pins 1 and 2 connected) or a signal from the external processor resets the FX3 device (J72 pins 2 and 3 connected).

4.6 Serial Interfaces

EZ-USB FX3 supports the following serial interfaces:

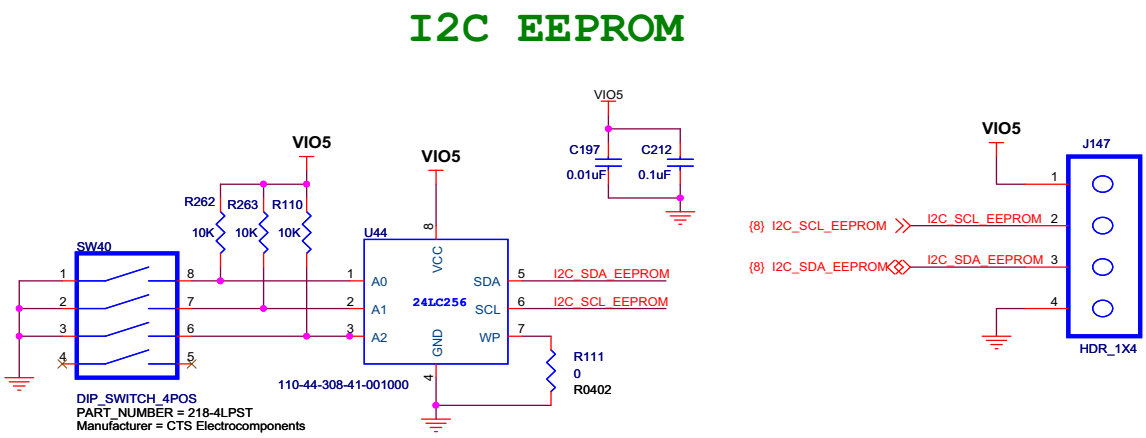
- I2C
- I2S
- SPI and UART
- JTAG

4.6.1 I2C Interface

The I2C interface lines on the FX3 device are available on headers for probing and expansion, as well as to connect to an on-board EEPROM device. You can set the EEPROM address bits A2, A1, and A0 using the on-board dip switch SW40.

As shown in Figure 4-13, setting the SW40 dip switch to the ON position connects the output to ground and connects a logic LOW to the EEPROM address lines. The OFF position connects VIO5 to the address lines of the EEPROM. Jumper J134 of the FX3 DVK sets the VIO5 voltage level.

Figure 4-13. EEPROM and Address Selection Switches



See the “I2C EEPROM Boot” section of the application note AN76405 - EZ-USB FX3 Boot Options for more details.

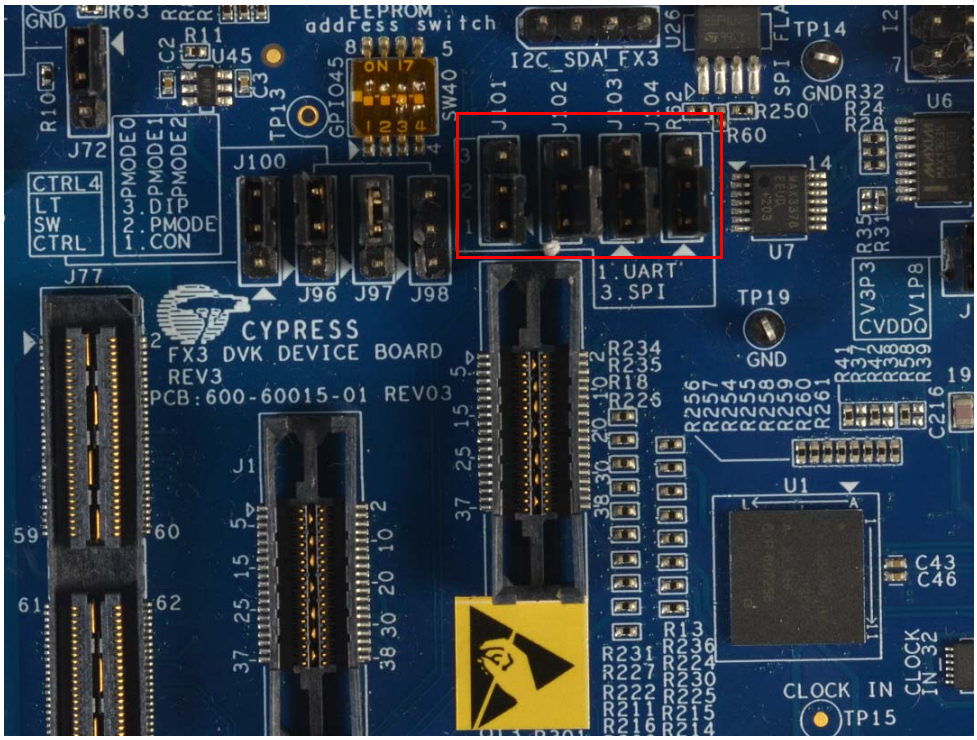
4.6.2 I2S

FX3 has an I2S port to support external audio codec devices. FX3 functions as an I2S master (transmitter only). The I2S interface consists of four signals: clock line (I2S_CLK), serial data line (I2S_SD), word select line (I2S_WS), and master system clock (I2S_MCLK). FX3 can generate the system clock as an output on the I2S_MCLK line or accept an external system clock input on the same line. All four I2S lines come out on header J20.

4.6.3 SPI and UART

EZ-USB FX3 supports an SPI master interface on the serial peripheral port. The SPI I/O lines are shared with the UART I/O lines on the FX3 device. The FX3 DVK jumper settings do not allow both the SPI and UART interfaces to operate simultaneously.

Figure 4-14. SPI/UART Selection Headers on DVK



Note The arrow marked near the jumpers indicates pin1 of the jumper.

Figure 4-15 shows the jumper J101–J104 schematic symbols and the signal names (UART/SPI) on each of these jumper pins.

Figure 4-15. UART/SPI Selection Headers

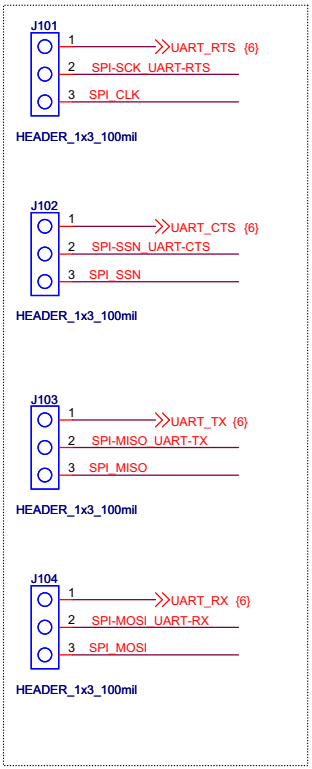


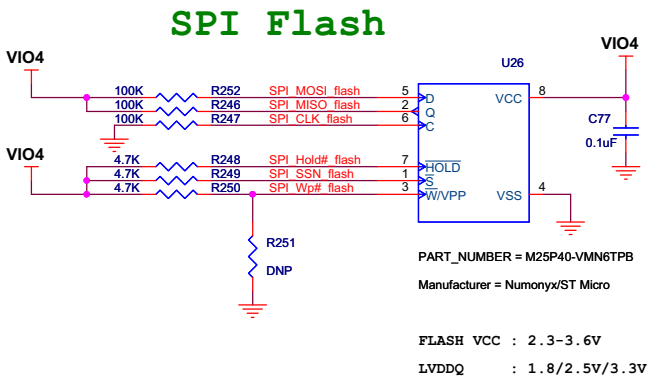
Table 4-5 shows the jumper J101–J104 pin combination to select either the UART or SPI interface.

Table 4-5. Jumper Pin Selection for UART and SPI I/O Lines

Jumper	Jumper Pins Shorted (UART I/O Lines)	Jumper Pins Shorted (SPI I/O Lines)
J101	1–2 (UART_RTS)	2–3 (SPI_CLK)
J102	1–2 (UART_CTS)	2–3 (SPI_SS)
J103	1–2 (UART_TX)	2–3 (SPI_MISO)
J104	1–2 (UART_RX)	2–3 (SPI_MOSI)

The SPI lines also connect to an onboard SPI flash, as shown in Figure 4-16.

Figure 4-16. Flash Circuit of DVK Board



Note Pull-up and pull-down resistors are not recommended on SPI_MISO/SPI_MOSI signals. They are not populated on the REV-03 FX3 DVK board.

4.6.4 JTAG

The FX3 JTAG interface provides a standard five-pin interface for connection to a JTAG debugger. The JTAG circuit on the DVK board provides an option to debug the firmware through the CPU core's on-chip debug circuitry. Industry standard debugging tools for the ARM926E-J-S core can be used for FX3 application development. The JTAG pins of FX3 come out on J51.

The introductory chapters in this note explain how to use the JTAG interface to debug FX3 interface using the Segger J-Link.

A. Appendix - Troubleshooting

A.1 Self-Power Mode

Problem	Possible Cause	Possible Solution
FX3 DVK board does not power on when power supply is connected	Incorrect switch setting	Set switch SW9 to “VP50” (down) position instead of “VBUS_IN” (up) position
	DVK board or power adapter failure	Insert the power plug firmly into the J49 power jack
		If problem persists, then the power supply unit may be faulty. Test the DVK board with another power supply unit with the same current/voltage ratings
		If the FX3 DVK board does not power up even after the power adapter has been replaced, the board may be faulty. Contact http://www.cypress.com/go/support for technical assistance

A.2 Bus-Power Mode

Problem	Possible Cause	Possible Solution
FX3 DVK board does not power on when USB cable is connected	Incorrect switch setting	Set switch SW9 to “VBUS_IN” (up) position.
	Incorrect setting of jumper J53	Verify that pins 1–2 of J53 are shorted

A.3 USB 2.0 Enumeration

Problem	Possible Cause	Possible Solution
FX3 DVK board does not enumerate when connected to PC host	Sequence of operations	If the board is powered first and then connected to the PC, it may not enumerate. User should hit the reset button in this case.
	Incorrect power mode jumper settings	If the DVK board is self-powered, ensure that the jumper settings are as shown in "Self-Power Mode" on page 36
	Incorrect setting of jumper J53	Verify that pins 1–3 or 2–4 of J53 are shorted to enable bus-power mode
	USB 3.0 cable failure	Verify that the USB cable is properly connected. Use the cable supplied with the kit or a USB-IF certified cable
	Windows OS fails to indicate enumeration to user	The FX3 device enumerates correctly but sometimes Windows OS does not display a window to indicate that the device has enumerated. See Figure 3-4 and Figure 3-5 and locate the FX3 device entry in the Windows OS Device Manager

A.4 USB Driver Installation

Problem	Possible Cause	Possible Solution
Cannot Install cyusb3.sys driver	Unsigned cyusb3.sys driver	FX3 SDK earlier than version 1.1.1 contains unsigned USB drivers. To use these drivers, reboot the PC host and during boot press **[F8]** and select **Disable Driver Signature enforcement** in the list of PC BIOS options
		Use drivers from SDK version 1.1.1 and later, which contain Microsoft certified *cyusb3.sys* driver along with a secure catalog file (*cyusb3.cat*). The latest SuiteUSB can also be downloaded separately from the FX3 SDK webpage.
	Incompatible driver for OS platform	Install the correct SuiteUSB installer. For 32-bit Windows XP/Vista/7 install *CyUSB3_x86_<Build_no>.msi.* For 64-bit OS platform install *CyUSB3_x64__<Build_no>..msi.* Perform manual driver re-binding as described in section 3.4.1. Using Windows Hardware Wizard.

A.5 USB Boot

Problem	Possible Cause	Possible Solution
Control Center displays **Programming Failed** when FX3 RAM is programmed with firmware	Incorrect PMODE settings	Verify that PMODE [2:0] pin settings for USB boot are as specified in Table 3-5
	Limitation of Etron host controller. (The Control Center application in SDK builds prior to version 1.2.1 downloads firmware to FX3 in pieces of 4096 bytes. The Etron host controller further splits this data into separate chunks of 4058 bytes and 38 bytes)	Migrate to FX3 SDK version 1.2.1 or later. (In FX3 SDK 1.2.1, the control transfer size for firmware download is reduced to 2048 bytes in the Control Center application)
	USB 3.0 cable failure	Verify that the USB cable is connected firmly. Use the cable supplied with the kit or a USB-IF certified cable.

A.6 I2C Boot

Problem	Possible Cause	Possible Solution
"**Programming of I2C EEPROM failed"** message appears in Control Center	Incorrect EEPROM mounting	Verify correct EEPROM orientation in socket U44
	EEPROM size	Make sure that the EEPROM has enough capacity to hold the .img file. To find the size of the .img file, right click on the .img file and choose "properties".
	Incorrect combination of PMODE jumpers and dip switch settings	Before programming the EEPROM, set the board in USB boot mode. After the DVK board enumerates using the FX3 bootloader, ensure that the address pins are configured correctly using SW40 before programming the EEPROM. Refer to the "Download Firmware Image to FX3 RAM" on page 43 for details.
	Incorrect settings on jumpers J42 and J45	Verify that pins 1-2 are shorted
	Incorrect settings on SW40 for EEPROM address pins (A[2:0])	Ensure that the EEPROM address pins A[2:0] are configured according to the requirement specified for the EEPROM being used (refer to the EEPROM datasheet)
	Faulty EEPROM	Replace EEPROM if programming still fails and redo the steps above
FX3 DVK does not boot from I2C EEPROM after programming	Incorrect PMODE settings	Refer to Table 3-6 to verify the correct PMODE settings for I2C boot
	Incorrect boot firmware header format	Refer to AN76405 - EZ-USB® FX3 Boot Options and verify that the elf2img command line utility generates the correct header format, as explained in the application note

A.7 SPI Boot

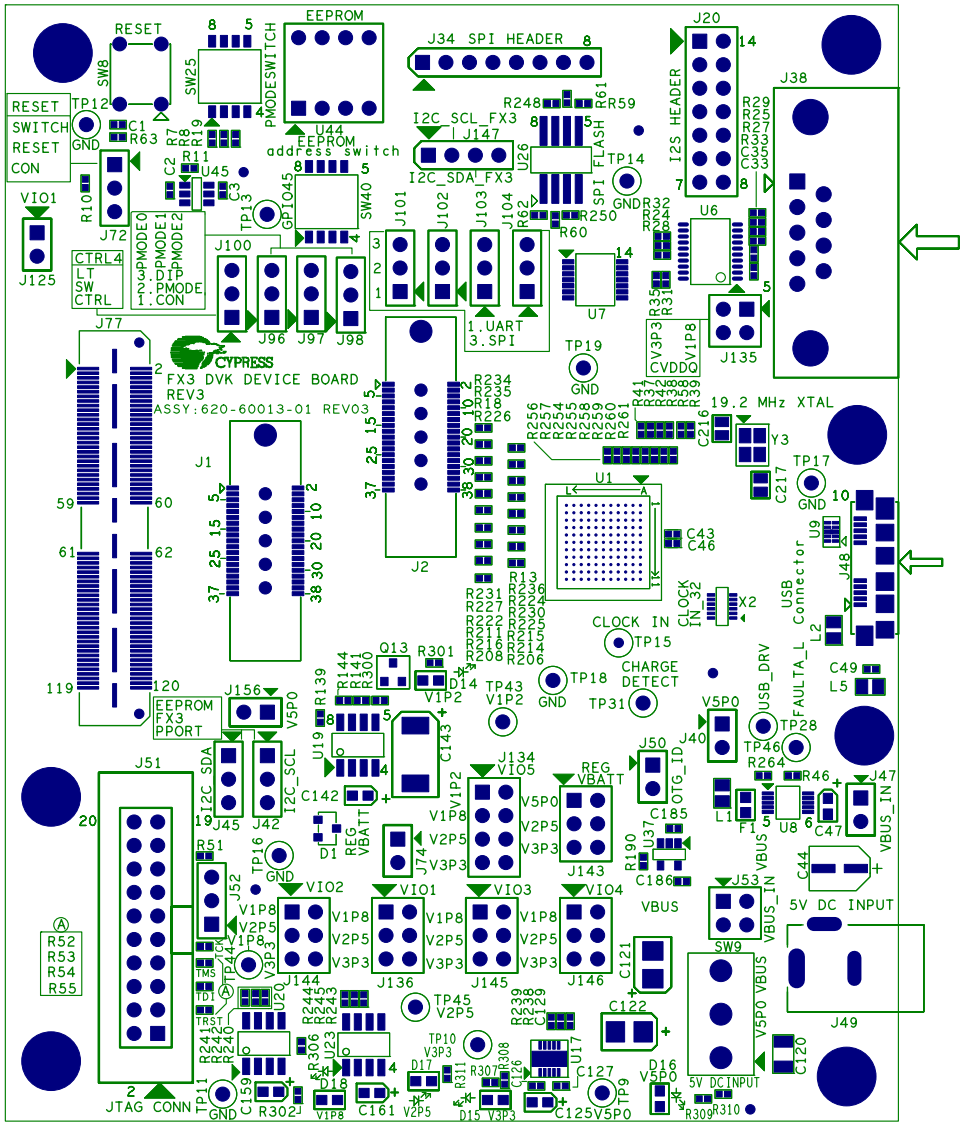
Problem	Possible Cause	Possible Solution
Programming of SPI Flash failed message appears in the Control Center	Incorrect PMODE settings	See Table 3-9 for the jumper and dip switch (SW25) settings. The settings vary before and after programming the firmware
	Incorrect Jumper positions	J101, J102, J103, and J104 pins 2-3 should be shorted before programming. This setting should remain for entire SPI boot sequence
FX3 DVK does not boot from SPI flash after programming	Incorrect PMODE settings	See Table 3-7 for the jumper and dip switch (SW25) settings. The settings vary before and after programming the firmware
	Pull-up resistors on SPI I/O lines	Some FX3 Rev-03 DVK boards contain pull-up resistors R246 and R252 on the MISO and MOSI lines of the SPI. Remove these pull-up resistors and retry the booting sequence
		FX3 DVK boards earlier than REV-03 do not support SPI boot.

A.8 USB3.0 Enumeration

Problem	Possible Cause	Possible Solution
FX3 enumerates as a USB 2.0 device even though it boots firmware that enables USB 3.0	Applicable only to NEC host controllers - Excessive spread spectrum reference clock from motherboard chipset. (The NEC host controller does not have its own reference clock for USB 3.0. If the motherboard has an out-of-spec PCIe reference clock (either incorrect frequency or excessive spread spectrum), the device attached to the USB 3.0 controller will not be able to track the signal and will fail to enumerate on the system.)	Check the Spread Spectrum setting in the PC BIOS. If enabled, disable Spread Spectrum manually
	Incorrect firmware configuration	Verify that the FX3 firmware has **CyU3PConnectState** set to CyTrue. This enables USB PHY on the FX3 device and connects to the USB host. The CyU3PGetConnectState API returns CyTrue if the FX3 device is connected and VBUS is detected by FX3. Also, the CyU3PUsbGetSpeed API provides information on the connection speed. See the FX3 API Guide for details

B. Appendix - PCB Layout

B.1 Assembly Top



B.2 Assembly Bottom