Application Note AN15813

Monitoring the EZ-USB FX2LP™ VBUS

Author: Rich Peng
Associated Project: No
Associated Part Family: CY7C68013A
CY7C68014A
CY7C68015A
CY7C68016A
Software Version: None
Associated Application Notes: None

Abstract

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

Introduction

One of the USB-IF specification requirements is that you must never drive the USB pins when they are not connected to the bus; you may only drive them when VBUS is present. Refer to section 7.2.1 of the Universal Serial Bus Specification, Revision 2.0. If you violate this requirement, you risk causing several failures to the system. One documented failure causes reset problems on the upstream devices which in turn may result in PC cold boot problems, or it causes hubs to fail to enumerate downstream devices. Other documented problems are failure to properly resume from a suspend state and forcing other enumerated devices off the bus.

The method used to detect if a device is connected to the bus is to monitor the VBUS signal. A bus powered design does not require VBUS monitoring because it cannot drive the bus when disconnected. A self powered design uses the power input to detect connection; it knows when it can assert its D+ or D- pull up resistor or when it must disconnect. This application note is intended to help the designer understand the methods of how to monitor the VBUS inside an application and know how to implement this feature.

When your application is for a self powered device, VBUS monitoring is very important. Do not drive the D+ / D- bus before VBUS is present or after it has been removed. This means that the D+ / D- signal can only be driven while the USB device is electronically connected to the USB host.

VBUS Function

Figure 1. USB Physical Connection

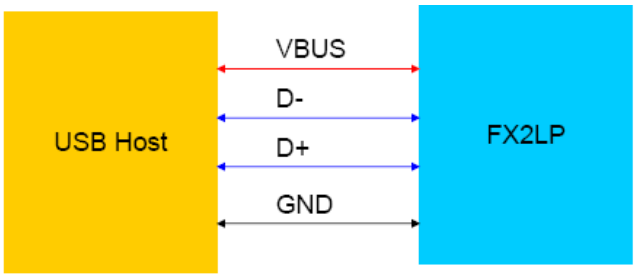


Figure 1 shows a basic diagram to illustrate the physical USB connections between device and host. The FX2LP is the device in the system; it is never the host. Based on this concept, we can more easily understand how the VBUS monitoring works in the USB system.

According to the USB specification, a USB device can have two power configurations: bus-powered and self-powered. In the bus-powered device, the device power relies on the USB host to provide a limited amount of power over the cable. Because VBUS provides power for the device to function, the D+ / D- is not driven before power is provided to the device.

For a self-powered device design, you must make certain that the device monitors VBUS and disables or enables the pull up on the D+ accordingly. In the self-powered

device, the power is typically provided by a power adaptor or a battery, not the VBUS. What kind of problems can possibly happen in this kind of design? The USB specification states “Devices may not provide power to the pull-up resistor on D+ / D- unless VBUS is present (see section 7.1.5 of USB 2.0 Specification). When VBUS is removed, the device must remove power from the D+ / D- pull-up resistor within 10 seconds”. If you require more detailed information, you can refer to the USB 2.0 Specification, section 7.2.1. After the reset period, the FX2LP device comes up connected on power up. That means that when the FX2LP gets out of reset the D+ is driven. From the time that the FX2LP gets out of reset, you have 100 ms to respond to the request otherwise the design violates the USB 2.0 specification. If you do not process VBUS carefully, your design can fail both the USB-IF logo certification tests and the Microsoft Windows WHQL or DTM tests.

The USB-IF checks for compliance on this by checking the back-voltage on the D+, the D-, and VBUS pins into a 15K ohm load. Refer to section F (Back Voltage Testing) of the *Universal Serial Bus Implements Forum Full and Low Speed Electrical and Interoperability Compliance Test Procedure* which you may download from http://www.usb.org/developers/docs/USB-IFTestProc1_3.pdf

This document explains why your design must comply with the back-voltage test. This document also explains what errors may occur if the device does drive D+ / D- prior to seeing VBUS. Failures that may occur are as follows:

- The PC fails to cold boot due to the back-voltage affecting the motherboard reset sequence.
- The hub fails to enumerate downstream devices due to reset anomalies.
- The motherboard fails to properly resume from a suspend state.
- Introduction of a device or hub forces one or more upstream devices off the bus.

These are only a few of the failures or anomalies that have been reported.

From this brief introduction, you can see why VBUS must be monitored if the product is a self-powered device with an FX2LP. If you do not monitor VBUS in this type of design, then you will spend more effort in debugging your board, resulting in wasted time and money. The suggestions presented here must be considered at the very beginning of a USB self-powered device design.

VBUS Monitor Methodology

When designing with the FX2LP as a self-powered device, there are two different design modes. One mode is a standalone design with the FX2LP and the other mode is using a coprocessor to control the FX2LP. Regardless of which mode you use, you have to implement the VBUS monitor feature in your design. Since the FX2LP is a programmable chip, you can use the firmware to implement the VBUS monitor feature. Inside the FX2LP, the DISCON bit controls the D+ pull up resistor to enable or disable the connection. You can use this bit to control the connection when monitoring VBUS. In hardware you must use one GPIO pin to detect the presence or absence of VBUS, then you can enable or disable the DISCON bit to match the VBUS behavior. Figure 2 shows the standalone mode application with FX2LP; Figure 3 shows the coprocessor mode.

Figure 2. Standalone Mode of FX2LP Design

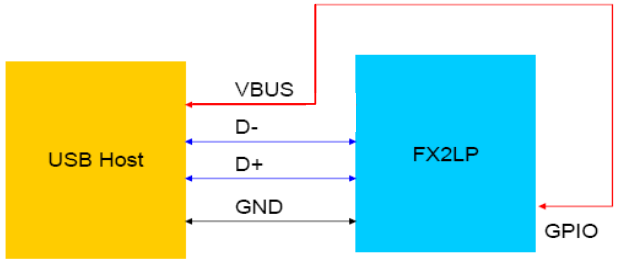
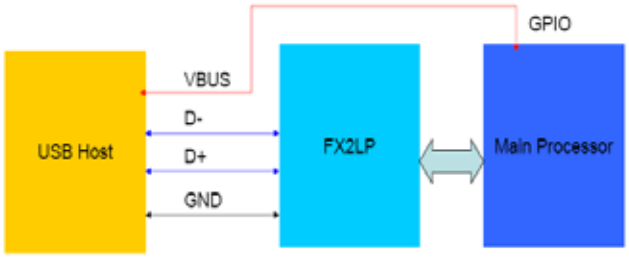
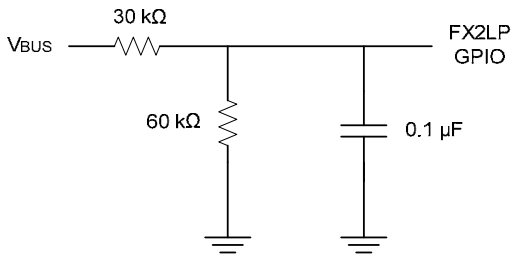


Figure 3. Coprocessor Mode of FX2LP Design



Monitoring VBUS can be accomplished by connecting VBUS to a GPIO, through a resistive bridge and have firmware monitor the status of the GPIO. The purpose of the resistive bridge is to add a bleed path for the VBUS signal. If you do not implement a bleed path, when the cable is unplugged the GPIO pin consumes less than 10 µA and causes the input to stay high for some time. A path of approximately 100K ohms causes this charge to dissipate within a short period of time. Typically a 30K ohm resistor between the VBUS pin and the GPIO pin with a 0.1 µF capacitor in parallel with a 60K ohm resistor from the GPIO pin to ground will accomplish this, see Figure 4.

Figure 4. Sample Schematic



Typically you place the firmware to monitor the GPIO status in the foreground loop (TD_Poll). This loop executes frequently and keys in on this as an event trigger. To accomplish this, the firmware must configure any GPIO port pin to be an input and in the loop. TD_Poll firmware has something similar to the example in Code 1.

Code 1. Configure a GPIO Port Pin

```
If ( !(IOA & 0x80) ) // if VBUS not present (using a PORTA.7 pin for example)
{
                     // user take action and set the DISCON bit to disconnect FX2LP from USB
                     // Setting DISCON bit disables the pull up on D+
USBCS |= bmDISCON;
}
Else
{
                     //Clear DISCON
USBCS&=~bmDISCON;
                     //Proceed to do other tasks
}
```

This code shows how to implement the VBUS monitor in the FX2LP standalone mode design. In some designs, other items may need to be considered such as in ATA applications (see the firmware in the CY4611 RDK).

If your application is in coprocessor mode, the connections are almost the same; the only difference is that you have to make the VBUS detection with the main processor chip GPIO, and follow the firmware design flow implemented in the main chip side. Then send the command to control the FX2LP DISCON bit through the Slave FIFO interface. Then the application in coprocessor mode will have the same behavior as that of the standalone mode.

Summary

This application note is intended to help the designer realize the importance of VBUS monitoring and to implement it at the start of the design to avoid having to reconfigure IOs and redo board layouts. VBUS monitoring helps minimize the design time and the time required for firmware debugging. If your design is complete, use the information in this application note to help debug your application and implement an appropriate VBUS monitor to eliminate any back-voltage problems you may have.

About the Author

Name: Rich Peng
Title: Application Engineer Sr Staff
Background:

Contact: lip@cypress.com
+886-27255515#207

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone: 408-943-2600
Fax: 408-943-4730
http://www.cypress.com