# Implementing Linux on the

# Zynq™-7000 SoC

## Lab 1.3

## Creating a Boot Image

## with SDK and Booting ZedBoard

September 2012
Version 05

# Table of Contents

## Lab 1.3 Overview

This lab builds upon the skills covered in the previous labs.

Using the FSBL created in Lab 1.1, the hardware bitstream file, and the U-Boot ELF file that was built on the cross build platform, we will create a boot image using SDK. This will give us a file from which the ZedBoard can be booted into U-Boot.

## Lab 1.3 Objectives

When you have completed Lab 1.3, you will know how to do the following:

- Create a boot image for ZedBoard
- Boot ZedBoard to U-Boot prompt

## Experiment 1: Create the Boot Image

This experiment shows how to export a boot image using SDK.

### *Boot Image Format*

The Zynq BootROM is capable of booting the processor from several different non-volatile memory types but it requires a data structure referred to as the Boot Image Format (BIF) for instructions on how to parse the different boot components. For the most part, simple Linux systems require only three components within the boot image:

1. FSBL (Stage 1 boot loader)

2. Programmable Logic (PL) Hardware Bitstream

3. U-Boot (Stage 2 boot loader)

Refer to Xilinx UG585 document, the Zynq Technical Reference Manual, for further details.

### *Boot Medium*

Although ZedBoard can boot from Quad-SPI Flash and SD card, for the purposes of this lab we will use the SD card as the boot medium. This gives us the advantage of being able to write directly to the FAT32 partition on the SD card from a PC.

**Experiment 1 General Instruction:**

Launch Xilinx Software Development Kit (SDK) and open the workspace from the project in the Lab 1.1 directory. Use SDK **Create Zynq Boot Image** tool to create the Zynq boot image file.

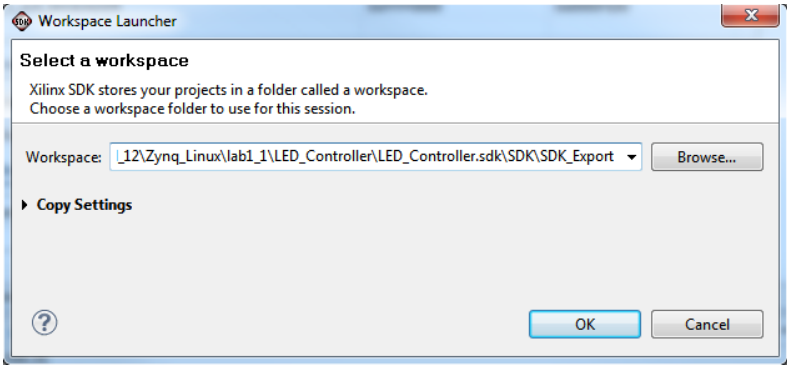
**Experiment 1 Step-by-Step Instructions:**

1. Launch Xilinx Software Development Kit (SDK) if not already open. **Start → All Programs → Xilinx Design Tools → ISE Design Suite 14.2 → EDK → Xilinx Software Development Kit**.

**Figure 1 – The SDK Application Icon**

2. Set or switch the workspace to:

   **C:\Speedway\Fall_12\Zynq_Linux\lab1_1\LED_Controller\LED_Controller.sdk\SDK\SDK_Export\**



Workspace Launcher

Select a workspace

Xilinx SDK stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: _12\Zynq_Linux\lab1_1\LED_Controller\LED_Controller.sdk\SDK\SDK_Export

**Figure 2 – Switching to the Appropriate SDK Workspace**

3. Once SDK is open, launch the **Create Zynq Boot Image** tool using the **Xilinx Tools→Create Boot Image** menu item.

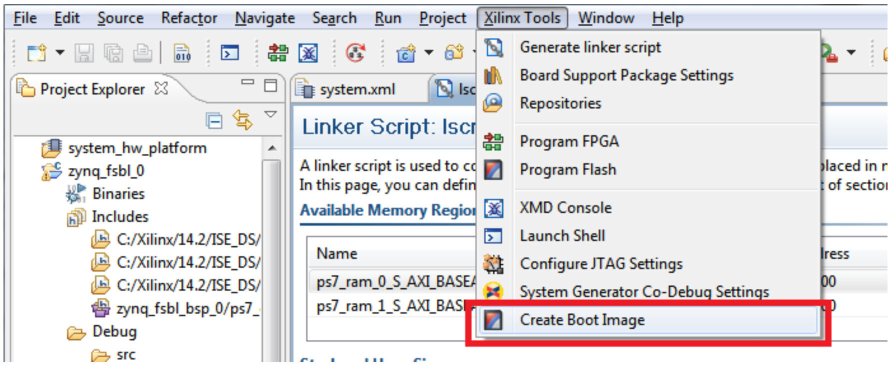


**Figure 3 – Launching the Create Zynq Boot Image Tool**

4. A new Boot Image Format (BIF) file must be created. In the **Bif file** drop down menu, select the **Create a new bif file…** option.

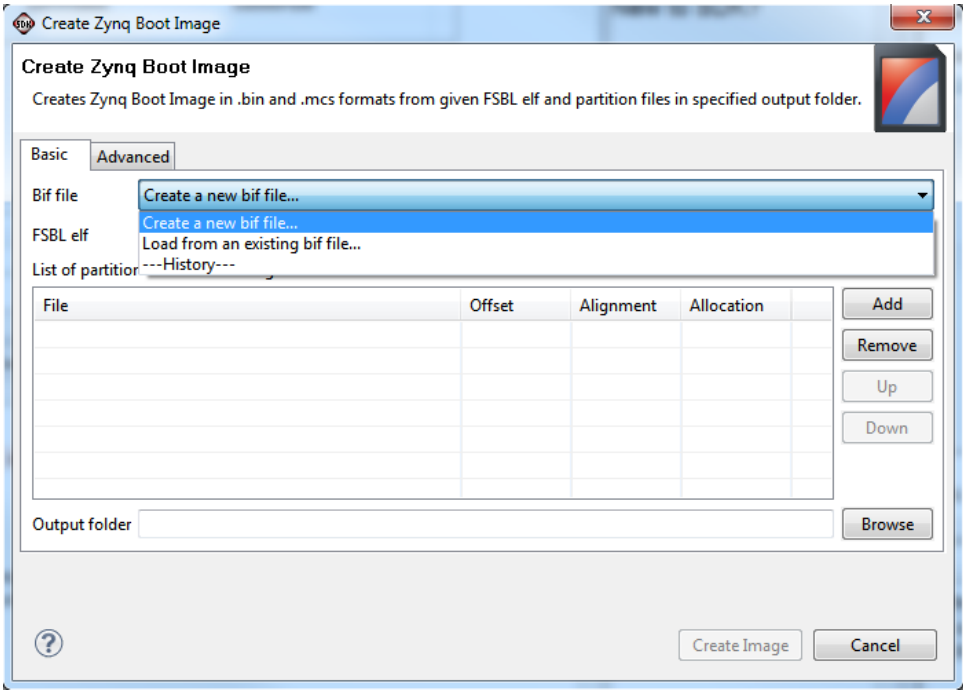


**Figure 4 – Creating a New BIF File**

5. Specify the **FSBL elf** file by browsing to the following file:

   **C:\Speedway\Fall_12\Zynq_Linux\lab1_3\zynq_fsbl_0.elf**

   Keep in mind the importance of order for files placed into a boot image. Specify the hardware bitstream by clicking the **Add** button and selecting the file below. Click the **Open** button to add this file to the list of partitions in the boot image:

   **C:\Speedway\Fall_12\Zynq_Linux\lab1_3\system.bit**

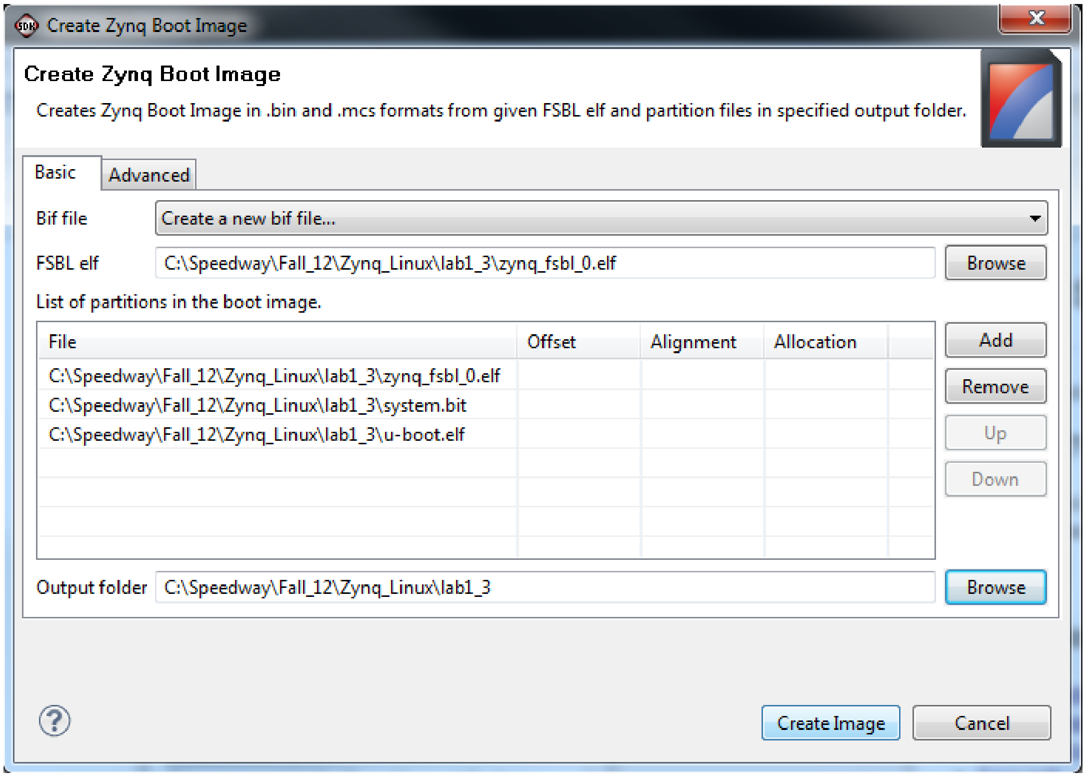
   Specify the application executable last (in our case it is the second stage boot loader U-Boot) by clicking the **Add** button and selecting the following file. Then click the **Open** button to add this file to the list of partitions in the boot image:

   **C:\Speedway\Fall_12\Zynq_Linux\lab1_3\u-boot.elf**

   Browse to the following **Output folder**:

   **C:\Speedway\Fall_12\Zynq_Linux\lab1_3\**

   Click the **Create Image** button which will launch Bootgen in the background, which is a tool for constructing boot images for Zynq-7000 AP SoC configuration.



Create Zynq Boot Image

**Create Zynq Boot Image**

Creates Zynq Boot Image in .bin and .mcs formats from given FSBL elf and partition files in specified output folder.

Bif file: Create a new bif file...

FSBL elf: C:\Speedway\Fall_12\Zynq_Linux\lab1_3\zynq_fsbl_0.elf

List of partitions in the boot image.

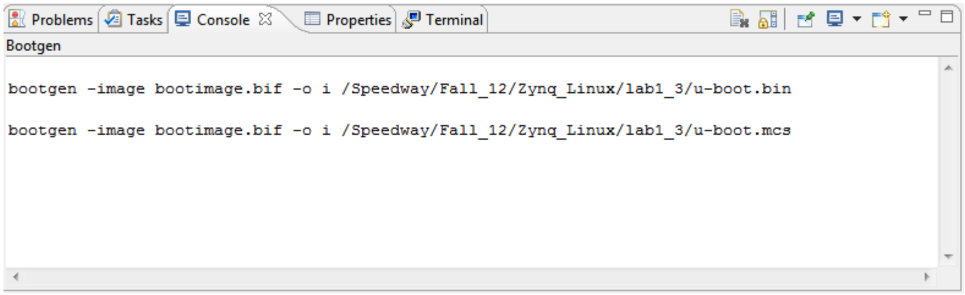
| File | Offset | Alignment | Allocation |
|---|---|---|---|
| C:\Speedway\Fall_12\Zynq_Linux\lab1_3\zynq_fsbl_0.elf | | | |
| C:\Speedway\Fall_12\Zynq_Linux\lab1_3\system.bit | | | |
| C:\Speedway\Fall_12\Zynq_Linux\lab1_3\u-boot.elf | | | |

Output folder: C:\Speedway\Fall_12\Zynq_Linux\lab1_3

**Figure 5 – Selecting Appropriate Boot Image Partitions**

6. Bootgen merges the BIT and ELF files into a single boot image with the format defined in the Boot Image Format (BIF) file to be loaded into Zynq devices at boot time. Evaluate the output seen in the SDK console, notice how bootgen is called against the **bootimage.bif** file, which contains the format which defines which files are integrated and what order they are added to the binary output file **u-boot.bin**.

   It is important to note that the order of the images should always be the FSBL first, followed by the Programmable Logic bitstream, and then finally the software application file (in our case it is the second stage boot loader). The BIF file is automatically created by the Zynq Boot Image Creation tool prior to the bootgen operation.

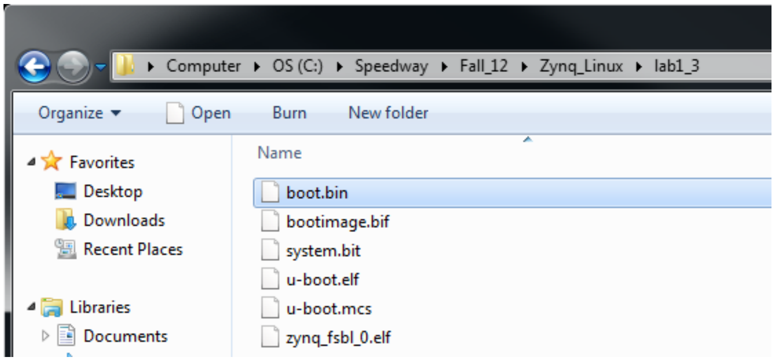


```
bootgen -image bootimage.bif -o i /Speedway/Fall_12/Zynq_Linux/lab1_3/u-boot.bin

bootgen -image bootimage.bif -o i /Speedway/Fall_12/Zynq_Linux/lab1_3/u-boot.mcs
```

**Figure 6 – Bootgen Running in the Background**

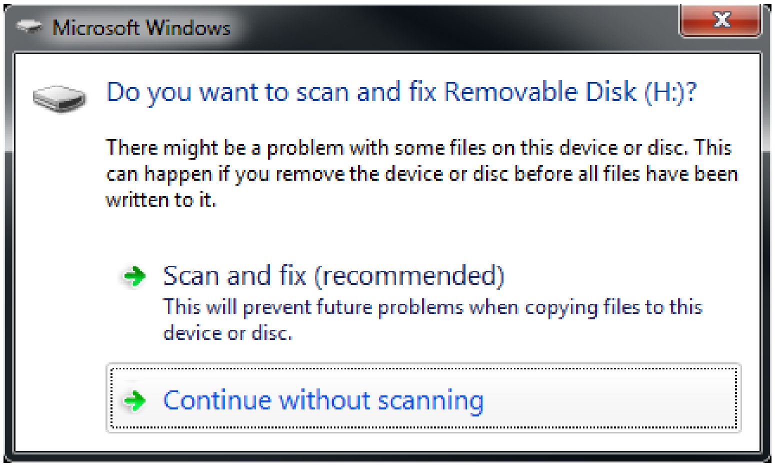
7. BootGen will create a **u-boot.bin** file in the Lab 1.3 folder. The BootROM on Zynq will only be able to locate a boot image file on the SD card if it is named **boot.bin**. Rename the **u-boot.bin** file to the **boot.bin** filename.



boot.bin
bootimage.bif
system.bit
u-boot.elf
u-boot.mcs
zynq_fsbl_0.elf

**Figure 7 – The Resulting Boot Image file Renamed to boot.bin**

8. Insert the SD card into the PC or SD card reader and wait for it to enumerate as a Windows drive. If prompted by Windows when inserting the SD card, select the **Continue without scanning** option.



Microsoft Windows

Do you want to scan and fix Removable Disk (H:)?

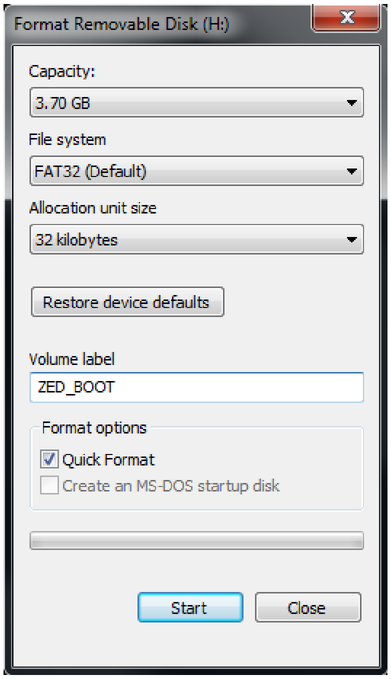
There might be a problem with some files on this device or disc. This can happen if you remove the device or disc before all files have been written to it.

Scan and fix (recommended)
This will prevent future problems when copying files to this device or disc.

Continue without scanning

**Figure 8 – Windows Prompt for Scanning and Fixing an SD Card**

The Zynq BootROM is capable of interpreting the FAT32 file system for SD card boot mode. If the SD card used is not already formatted for FAT32 file system, right click on the drive in Windows Explorer and select the **Format** option. Select the options shown in Figure 9 and click the **Start** button.



Format Removable Disk (H:)

Capacity: 3.70 GB

File system: FAT32 (Default)

Allocation unit size: 32 kilobytes

Volume label: ZED_BOOT

Format options

☑ Quick Format

☐ Create an MS-DOS startup disk

**Figure 9 – Formatting the SD Card with FAT32 File System**

9. The SD card included with ZedBoard is pre-loaded with the factory demo image for the board. You may want to preserve that for your own future use before overwriting. To do so, simply copy the contents to a folder on your PC hard drive before overwriting any of the files.

   Copy the **boot.bin** file from the Lab 1.3 folder to the top level of the SD card. Replace any existing versions of the **boot.bin** file that may be on the SD card.

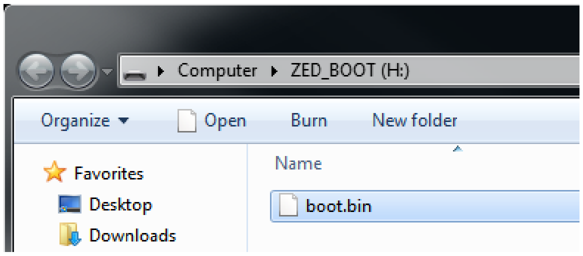


**Figure 10 – The Boot Image File Copied to the SD Card**

   Note: You may encounter problems writing to the SD card when the write-protect button in the "off" position. With the SD card face up, the tiny white plastic switch on the left side should be slid toward the card edge connector (the shortest side).

   For some SD cards, on the opposite side of the SD card from the switch there is a tiny notch cut out. This notch forces some SD cards to work in camera's, MP3 Players, PDA's, etc. that are portable devices, but may not work the same in USB card readers or computers with SD card reader slots. Some card readers for SD cards have this extra built in feature for copy protection (for some reason on blank cards) which makes it useless for custom applications.

   A strategically placed piece of tape, masking, electrical, sometimes transparent may work in overcoming this feature. Be very precise in placing the tape over the notch, but not over the adjacent brass electrical contact. Then place it in your card reader, format it, read and write files to your desire.

   Once you have bypassed the protection feature of the SD card, install your files as needed. If you get the same error again after applying the tape, add a second layer. Double check that you did not cover the connection edge and the plastic tab is closest to the side you insert in the device you are using. Sometimes it takes several tries, but it will usually work with some added patience. If you have existing files on the SD card, insert it, right click on the files you wish to remove or overwrite and select the **Properties** option. Then unselect the **Read Only** check by the box. Try again to delete or add files, if you are prompted with a message indicating that the disk is write protected, then check the tape again.

## *Questions:*

***Answer the following questions:***

- *What are the 3 required files needed to create a Zynq boot image?*

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

- *What filename will the BootROM search for when using the SD card boot mode?*

______________________________________________________________________

## Experiment 2: Boot ZedBoard Using New Boot Image

ZedBoard can now be booted using the Boot Image that was copied to the SD card in the previous exercise.

**Experiment 2 General Instruction:**

Boot ZedBoard using the SD card with the created Zynq boot image and observe the terminal output.

**Experiment 2 Step-by-Step Instructions:**

1. Connect 12 V power supply to ZedBoard barrel jack (J20).

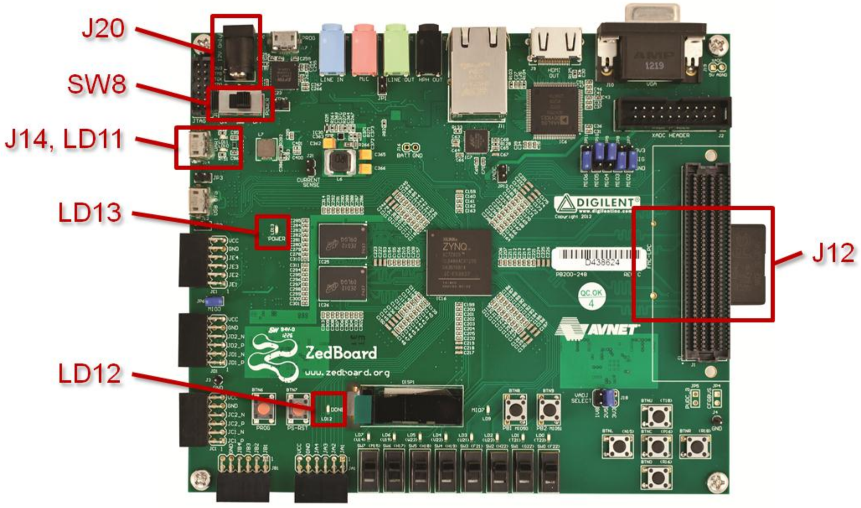


**Figure 11 – ZedBoard Hardware Reference**

2. Connect the USB-UART port of ZedBoard (J14) which is labeled UART to a PC using the MicroUSB cable.

3. Insert the 4GB SD card included with ZedBoard into the SD card slot (J12) located on the underside of ZedBoard PCB.

4. Verify the ZedBoard boot mode (JP7-JP11) and MIO0 (JP6) jumpers are set to SD card mode as described in the Hardware Users Guide:

   http://www.zedboard.org/sites/default/files/ZedBoard_HW_UG_v1_6.pdf

   A copy of the Hardware Users Guide is also located in the SpeedWay **C:\Speedway\Fall_12\Zynq_Linux\support_documents\** folder for your convenience.

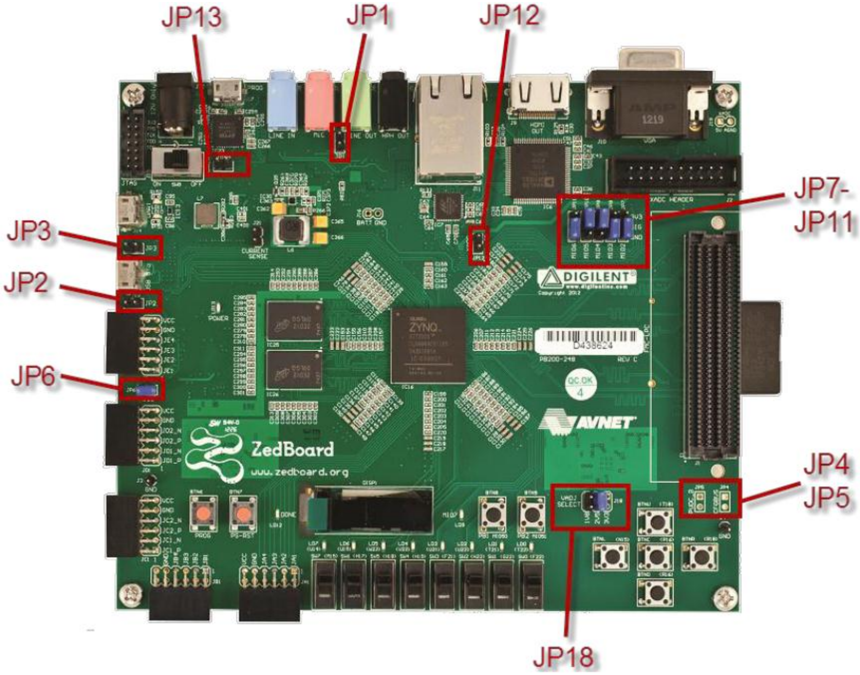


**Figure 12 – ZedBoard Jumper Settings**

5. Turn power switch (SW8) to the ON position.  ZedBoard will power on and the Green Power Good LED (LD13) should illuminate.

6. The PC may pop-up a dialog box asking for driver installation.

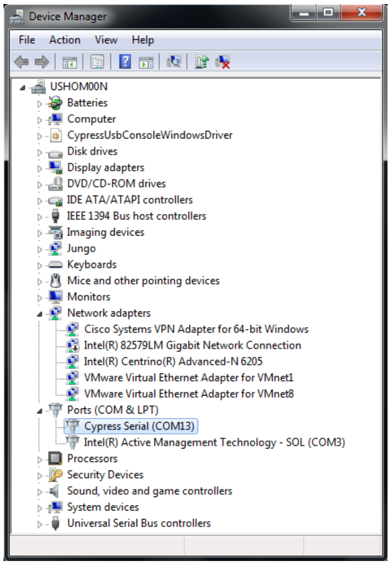
   ZedBoard has a USB-UART bridge based on the Cypress CY7C64225 chipset. Use of this feature requires that a USB driver be installed on your Host PC.

   If Windows recognizes the USB-UART and loads the software driver, go ahead and proceed to the next step. However, if the host PC does not recognize the USB-UART and enumerate it as a COM port device refer to the USB-UART Setup Guide document in the link below for instructions on installing this driver. When driver installation is complete, continue to the next step.

   http://www.zedboard.org/sites/default/files/CY7C64225_Setup_Guide_1_1.pdf

   A copy of the USB-UART Setup Guide is also located in the SpeedWay **C:\Speedway\Fall_12\Zynq_Linux\support_documents\** folder for your convenience.

7. Wait approximately 15 seconds. The blue Done LED (LD12) should illuminate. Use the Windows Device Manager to determine the COM Port.



```
USHOM00N
    Batteries
    Computer
    CypressUsbConsoleWindowsDriver
    Disk drives
    Display adapters
    DVD/CD-ROM drives
    IDE ATA/ATAPI controllers
    IEEE 1394 Bus host controllers
    Imaging devices
    Jungo
    Keyboards
    Mice and other pointing devices
    Monitors
    Network adapters
        Cisco Systems VPN Adapter for 64-bit Windows
        Intel(R) 82579LM Gigabit Network Connection
        Intel(R) Centrino(R) Advanced-N 6205
        VMware Virtual Ethernet Adapter for VMnet1
        VMware Virtual Ethernet Adapter for VMnet8
    Ports (COM & LPT)
        Cypress Serial (COM13)
        Intel(R) Active Management Technology - SOL (COM3)
    Processors
    Security Devices
    Sound, video and game controllers
    System devices
    Universal Serial Bus controllers
```

**Figure 13 – Device Manager Showing Enumerated USB-UART as COM13**

Note: Each unique USB-UART device attached will enumerate under the next available COM port. Here in this example, the Cypress CY7C64225 USB-UART device is enumerated as COM13.

8. On the PC, open a serial terminal program. For this demo, Windows 7 was used which does not come with a built in terminal application such as HyperTerm. Tera Term was used in this example which can be downloaded from the Tera Term project on the SourceForge Japan page:

   http://ttssh2.sourceforge.jp

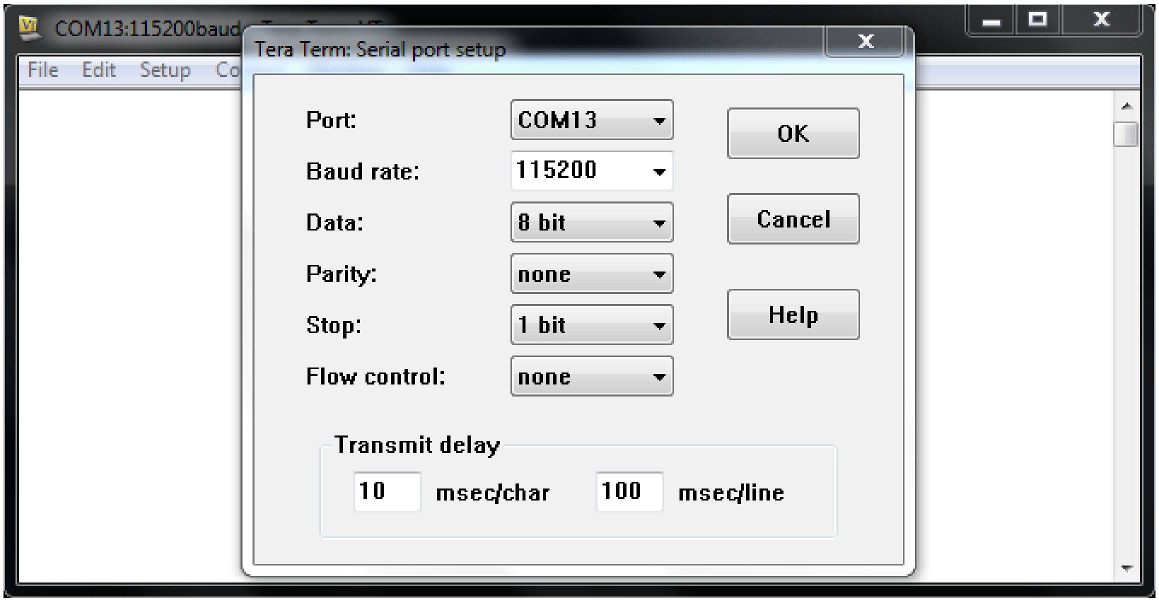
9. Once Tera Term is installed, Tera Term can be accessed from the desktop or start menu shortcuts.

**Figure 14 – Tera Term Icon**

10. To configure baud rate settings, open the Serial Port Setup window from the **Setup→Serial port** menu selection. Select the USB-UART COM port enumeration that matches the listing found in Device Manager.

    Also set the Baud rate option to **115200**, the Data width option to **8-bit**, the Parity option to **none**, the Stop bit option to **1 bit**, and the flow control to **none**.

    Finally, assign the transmit delay parameters to **10 msec/char** and **100 msec/line**, and then click **OK**. This setting will help with later lab exercises were many lines of text will be pasted or sent to the console at one time.



Port: COM13
Baud rate: 115200
Data: 8 bit
Parity: none
Stop: 1 bit
Flow control: none

Transmit delay
10 msec/char 100 msec/line

**Figure 15 – Tera Term Serial Port Setup Page**

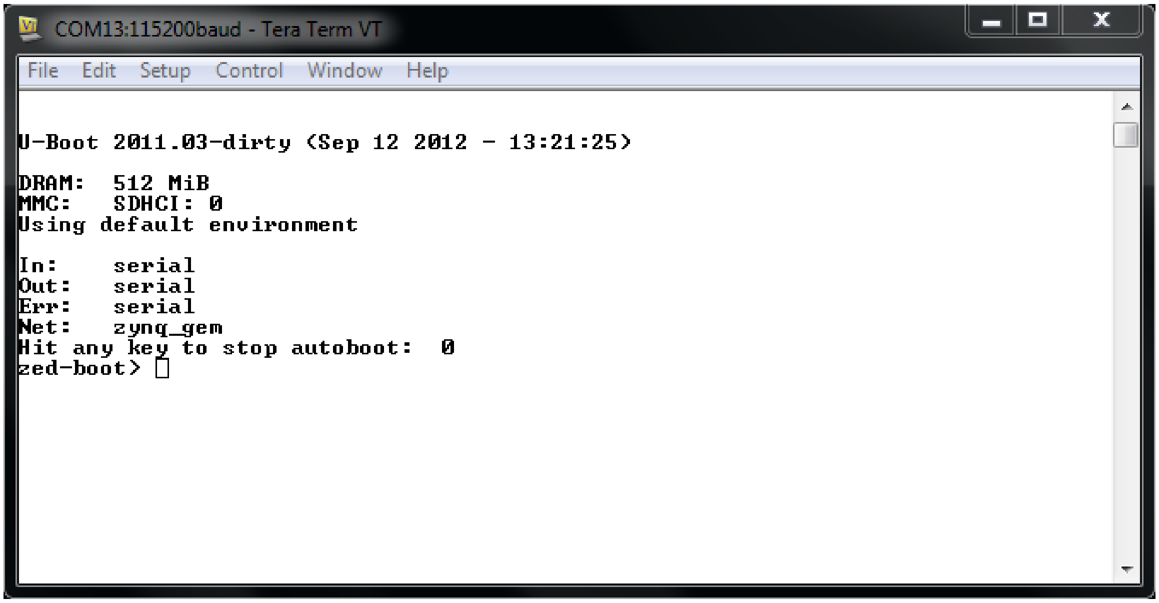
11. Optionally, at this point, the terminal settings can be saved for later use. To do this, use the **Setup→Save** setup menu selection and overwrite the existing TERATERM.INI file.

12. Power cycle the ZedBoard and monitor the Tera Term window carefully.

    When the terminal output from U-Boot and a countdown is observed, **press one of the keyboard keys to interrupt the boot process**.

    If the amber USB-Link Status (LD11) does not flicker to indicate activity and no output is displayed on the terminal after 10 seconds, check the driver installation to determine if the device driver is recognized and enumerated successfully and that there are no errors reported by Windows.

    Take a few minutes to explore the U-Boot environment and command line options. Use the **?** command entry to display a listing of the supported U-Boot commands. Use any of the listed commands to explore U-Boots capabilities.



```
U-Boot 2011.03-dirty (Sep 12 2012 - 13:21:25)

DRAM:  512 MiB
MMC:   SDHCI: 0
Using default environment

In:    serial
Out:   serial
Err:   serial
Net:   zynq_gem
Hit any key to stop autoboot:  0
zed-boot> 
```

**Figure 16 – ZedBoard U-Boot Prompt**

## *Questions:*

***Answer the following questions:***

- *Which document can be used to determine the jumper settings for different boot modes on ZedBoard?*

______________________________________________________________________

- *Why is Tera Term used instead of standard HyperTerm?*

______________________________________________________________________

______________________________________________________________________

## Exploring Further

If you have additional time and would like to investigate more…

- Experiment with the order of the three files used to create the boot image. Does the order these files are added **really** matter?

- Experiment with the naming of the boot image file. Does the file **really** need to be named boot.bin?

- Experiment with the U-Boot environment. Enter the command **help** to see the list of supported commands.

This concludes Lab 1.3.

## Revision History

| Date | Version | Revision |
|---|---|---|
| 07 Sep 12 | 00 | Initial Draft |
| 28 Sep 12 | 01 | Initial Draft |
| 18 Oct 12 | 02 | Course Release |
| 14 Jan 13 | 05 | ZedBoard.org Training Course Release |
| | | |

## Resources

http://www.zedboard.org

http://www.xilinx.com/zynq

http://www.xilinx.com/planahead

http://www.xilinx.com/sdk

# Answers

## Experiment 1

- *What are the 3 required files needed to create a Zynq boot image?*

1. FSBL

2. Bitstream

3. Application or Second Stage Boot Loader

- *What filename will the BootROM search for when using the SD card boot mode?*

boot.bin

## Experiment 2

- *Which document can be used to determine the jumper settings for different boot modes on ZedBoard?*

The ZedBoard Hardware Users Guide contains this information.

- *Why is Tera Term used instead of standard HyperTerm?*

HyperTerminal is not included in a standard Windows 7 install and Tera Term is more flexible in terms of scripting capability.