Implementing Linux on the

Zynq™-7000 SoC

Lab 2.1

Building the Linux Kernel from Source Code

September 2012
Version 05

Table of Contents

Lab 2.1 Overview

The Xilinx Linux project combines the benefit of open source Linux operating system together with a customized solution geared towards developing software on its processing platform.

Xilinx provides a freely downloadable Linux solution that has been tested on Xilinx Zynq-7000 All Programmable SoC development boards and the source files are hosted on Xilinx's GIT repository. You can use the Xilinx Software Development Kit (SDK) to develop Linux applications for Zynq-7000 devices. This solution is also used as the foundation for Xilinx Targeted Reference Designs (TRDs) in different market segments.

In this lab, the process to rebuild the kernel from the source repository is explored.

Linux can also help to accelerate your next embedded development project providing a Unix-like platform upon which to build a system, allowing you to focus on the features of your embedded application. Also, if the need arises, however, you have the source code, and can add support for new hardware or add a special feature.

Lab 2.1 Objectives

When you have completed Lab 2.1, you will know how to do the following:

- Retrieve Linux kernel source code from Xilinx repository
- Configure kernel for ZedBoard target
- Build kernel for ZedBoard target

Experiment 1: Clone the Xilinx Linux Kernel Git Repository

This experiment shows how to clone the Xilinx Linux kernel Git repository for Zynq. To successfully complete this lab, you will need Internet access to retrieve the repository information from the Xilinx website.

Experiment 1 General Instruction:

Clone the Xilinx Linux Kernel Git repository for Zynq.

Important Note: If performing this lab on an Avnet SpeedWay training laptop, the appropriate Git repository has already been cloned for you and Experiment 1 is for reference only.

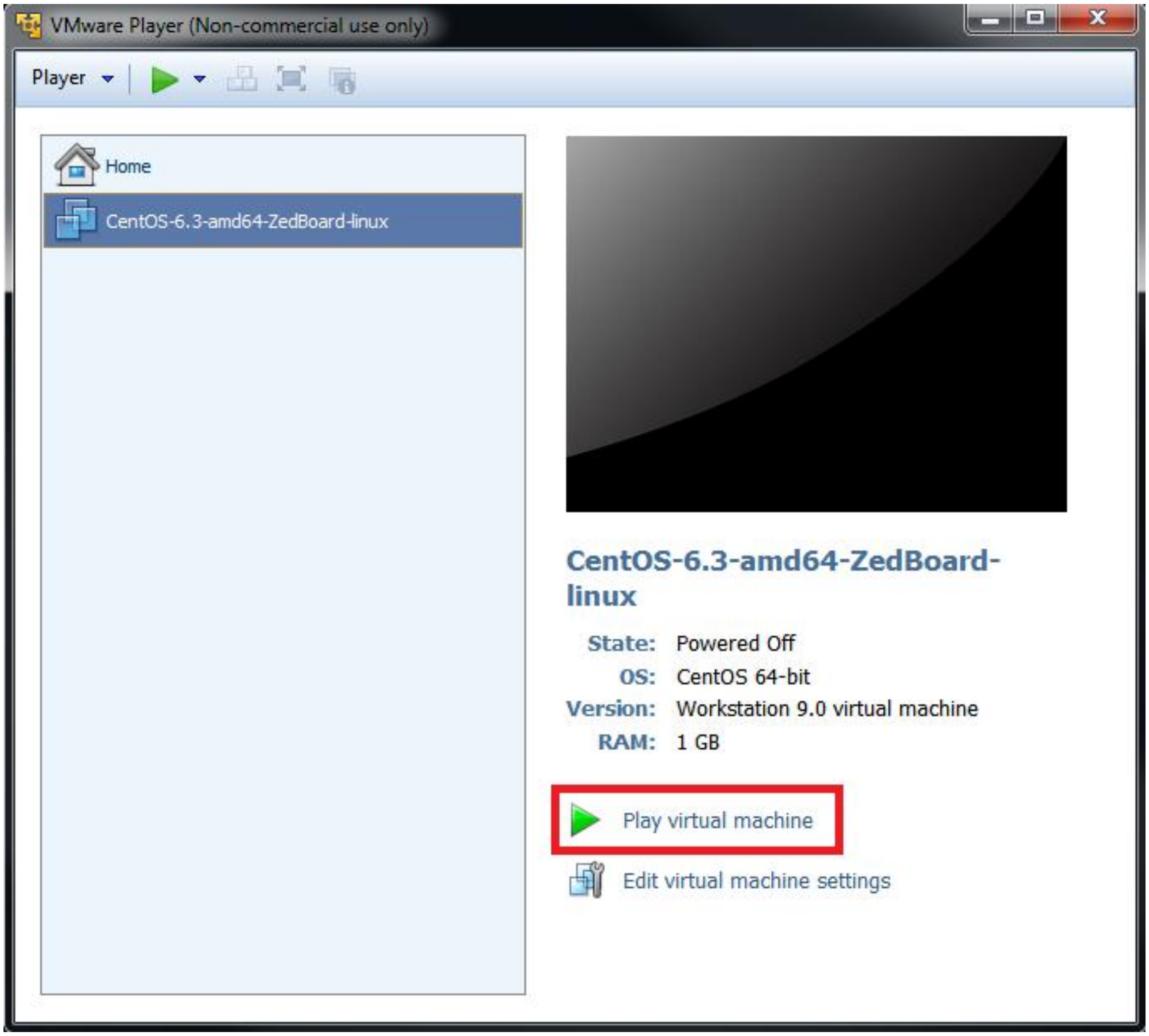
If the Linux kernel source already exists in the **/home/training/linux-xlnx/** folder on the Virtual Machine, Experiment 1 can be skipped and Experiment 2 can be started.

Experiment 1 Step-by-Step Instructions:

1. If the CentOS virtual machine is not already open, launch the VMware Player application from by selecting **Start → All Programs → VMware → VMware Player**. If the CentOS virtual machine is already open, skip ahead to Step 4.

Figure 1 – The VMware Player Application Icon

2. Select the virtual machine named **CentOS-6.3-amd64-ZedBoard-linux** from the selections on the left and then click on the **Play virtual machine** button to the right.



VMware Player (Non-commercial use only)
Player
Home
CentOS-6.3-amd64-ZedBoard-linux

CentOS-6.3-amd64-ZedBoard-linux

State: Powered Off
OS: CentOS 64-bit
Version: Workstation 9.0 virtual machine
RAM: 1 GB

Play virtual machine
Edit virtual machine settings

Figure 2 – The VMware Player Application

3. If prompted for a workstation login, click on the user entry **training** and enter the password **Avnet** in order to log into the system.

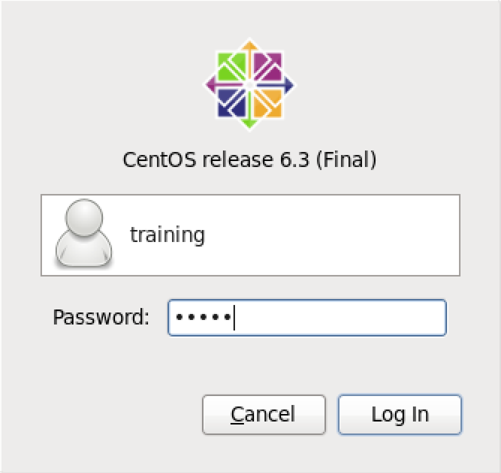


Figure 3 – The CentOS Workstation Login

4. If a terminal is not already open on the desktop of the CentOS guest operating system, open a terminal window through the **Applications→System Tools→Terminal** menu item. If a terminal is already open, bring that terminal session into focus on the desktop.

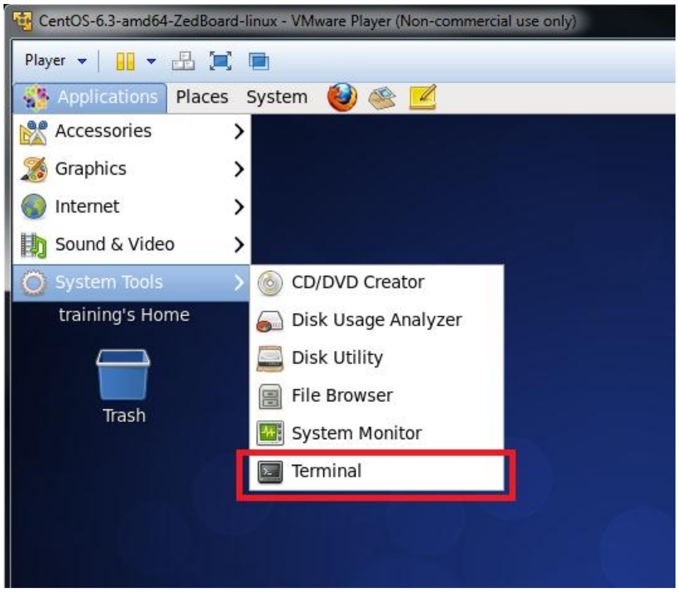


Figure 4 – Launching the CentOS Terminal from the Desktop

5. Change the current working directory to the training user home directory.

```
$ cd ~
```

6. The Xilinx Linux kernel Git repository is located at the following URL:

   git://git.xilinx.com/linux-xlnx.git

   To get a working copy of the codebase, clone the remote repository to your local machine. Cloning creates the repository and checks out the latest version, which is referred to as HEAD.

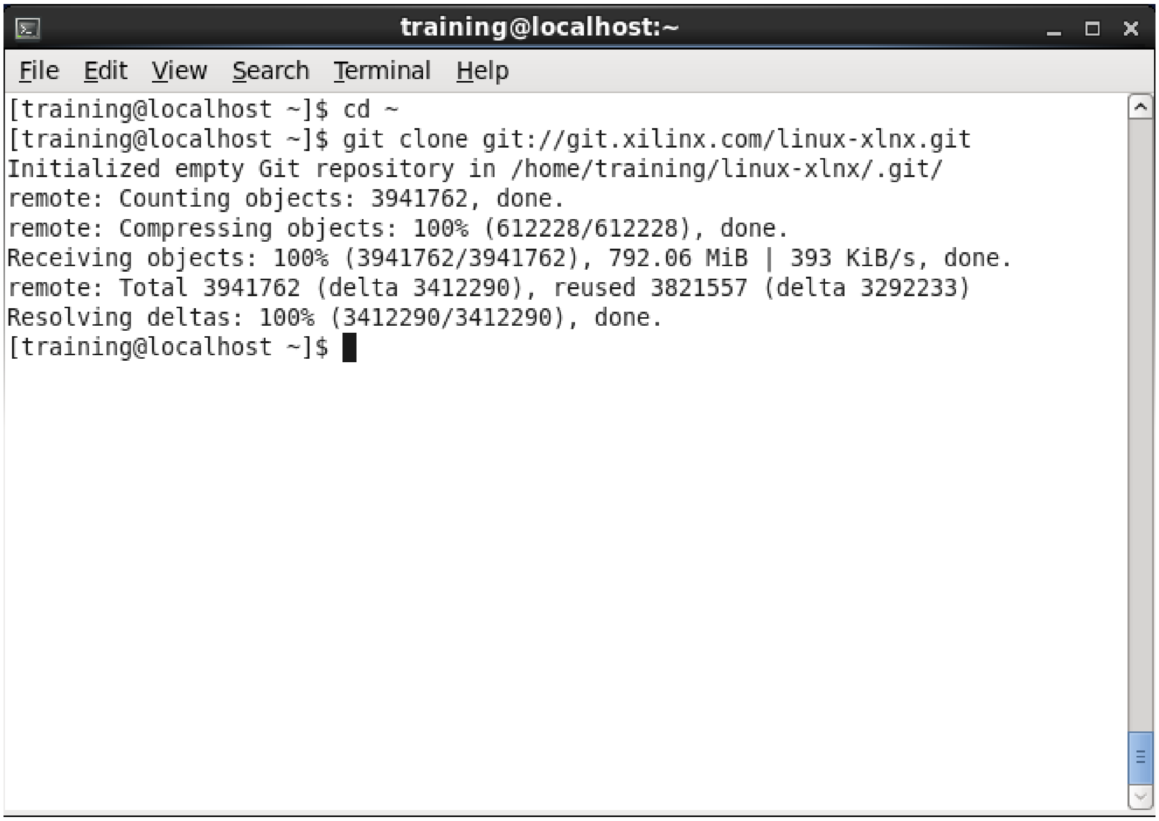
   Use the following git command to clone the repository. Perform this step only if you are not performing this exercise as part of a Live SpeedWay course.

```
$ git clone git://git.xilinx.com/linux-xlnx.git
```

7. Wait until the clone operation completes, this could take 20-60 minutes depending upon your connection speed.

   The clone command sets up a few convenience items for you by:

   - Keeping the address of the original repository
   - Aliasing the address of the original repository as origin so that changes can be easily sent back (if you have authorization) to the remote repository



```
[training@localhost ~]$ cd ~
[training@localhost ~]$ git clone git://git.xilinx.com/linux-xlnx.git
Initialized empty Git repository in /home/training/linux-xlnx/.git/
remote: Counting objects: 3941762, done.
remote: Compressing objects: 100% (612228/612228), done.
Receiving objects: 100% (3941762/3941762), 792.06 MiB | 393 KiB/s, done.
remote: Total 3941762 (delta 3412290), reused 3821557 (delta 3292233)
Resolving deltas: 100% (3412290/3412290), done.
[training@localhost ~]$ 
```

Figure 5 – Using the Git Clone Command

8. Change from the home directory into the Xilinx Linux kernel source directory.

```
$ cd linux-xlnx/
```

9. Checkout the code changes related to the 14.2 tools release which are kept under the **xilinx-14.2-build1** tag.

```
$ git checkout -b xilinx-14.2-build1-trd \

xilinx-14.2-build1-trd
```

Questions:

Answer the following questions:

- *How is a working copy of the Xilinx Linux kernel code base obtained?*

____

Experiment 2: Configuring the Kernel for a Zynq Target

A working copy of the Xilinx Linux kernel repository is now available on the local development machine and can be used to configure the source tree so that it can be built for a Zynq target platform.

Keep in mind that the Linux kernel must be configured for a target platform before being compiled. This is done using the make defconfig command format:

```
make ARCH=arm <PLATFORM>_defconfig
```

Where <PLATFORM> is the name of the target platform (but without the .h) found in the include/configs/ folder.

Experiment 2 General Instruction:

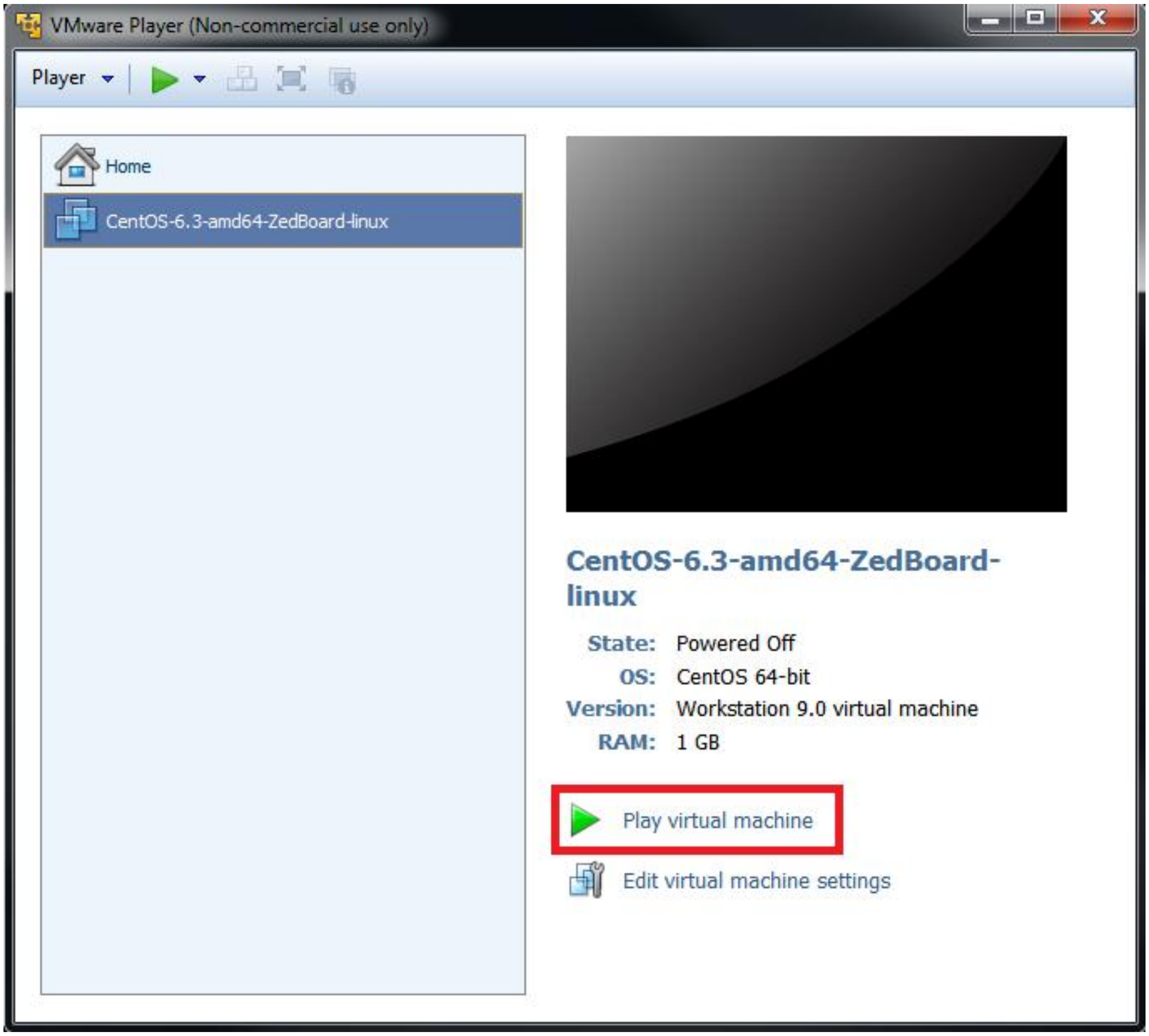
Configure the Xilinx Linux kernel source for the Zynq target platform.

Experiment 2 Step-by-Step Instructions:

1. If the CentOS virtual machine is not already open, launch the VMware Player application from by selecting **Start → All Programs → VMware → VMware Player**. If the CentOS virtual machine is already open, skip ahead to Step 4.

Figure 6 – The VMware Player Application Icon

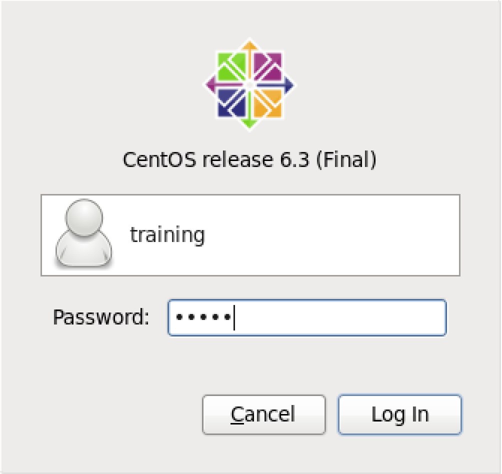
2. Select the virtual machine named **CentOS-6.3-amd64-ZedBoard-linux** from the selections on the left and then click on the **Play virtual machine** button to the right.


VMware Player (Non-commercial use only)
Player
Home
CentOS-6.3-amd64-ZedBoard-linux
CentOS-6.3-amd64-ZedBoard-linux
Play virtual machine
Edit virtual machine settings

State: Powered Off
OS: CentOS 64-bit
Version: Workstation 9.0 virtual machine
RAM: 1 GB

Figure 7 – The VMware Player Application

3. If prompted for a workstation login, click on the user entry **training** and enter the password **Avnet** in order to log into the system.


CentOS release 6.3 (Final)

training

Password: •••••

Figure 8 – The CentOS Workstation Login

4. If a terminal is not already open on the desktop of the CentOS guest operating system, open a terminal window through the **Applications→System Tools→Terminal** menu item. If a terminal is already open, bring that terminal session into focus on the desktop.

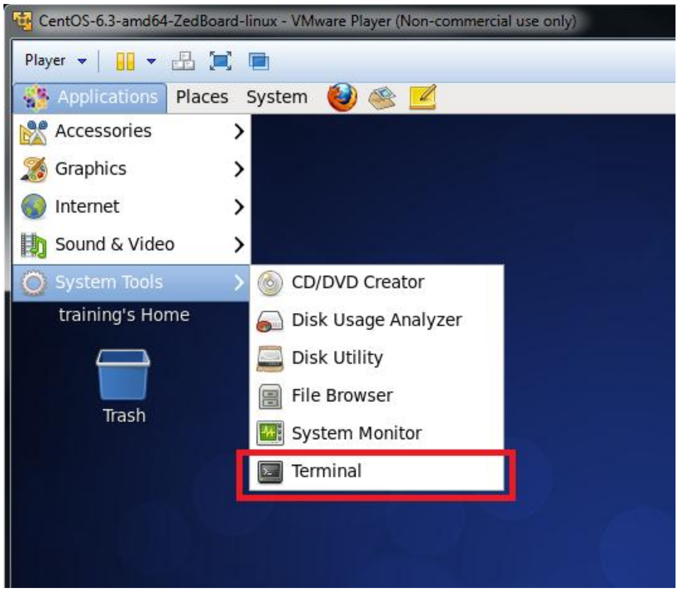


Figure 9 – Launching the CentOS Terminal from the Desktop

5. Change from the home directory into the Xilinx Linux kernel source directory.

```
$ cd ~/linux-xlnx/
```

6. For good measure (sometimes a necessity) run a make distribution clean command against the kernel source code. This command will remove all intermediary files created by config as well as any intermediary files created by make and it is a good way to clean up any stale configurations.

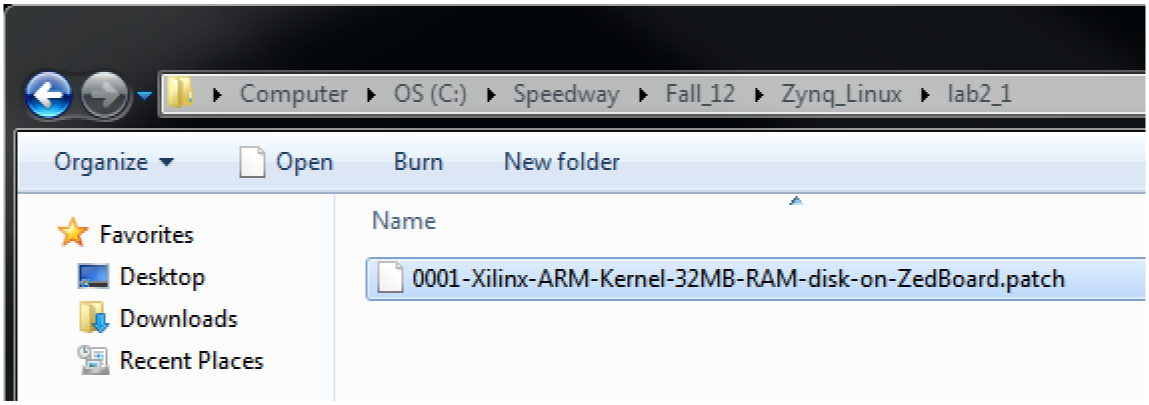
```
$ make distclean
```

7. Locate the **.patch** file in the Lab 2.1 folder on the host operating system by using Windows explorer to navigate to the following folder:

   C:\Speedway\Fall_12\Zynq_Linux\lab2_1\

   Right click on this file and select the **Copy** option which will place the selected file in the Virtual Machine clipboard.

   This patch modifies the default kernel build configuration for Zynq targets so that up to 32MB RAM disks can be supported. This support will be needed for the following labs 3.2 and 3.3 to be completed successfully.



0001-Xilinx-ARM-Kernel-32MB-RAM-disk-on-ZedBoard.patch

Figure 10 – Copying the Kernel Patch File to Virtual Machine Clipboard

8. Open a file browser window through the **Applications→System Tools→File Browser** menu item.

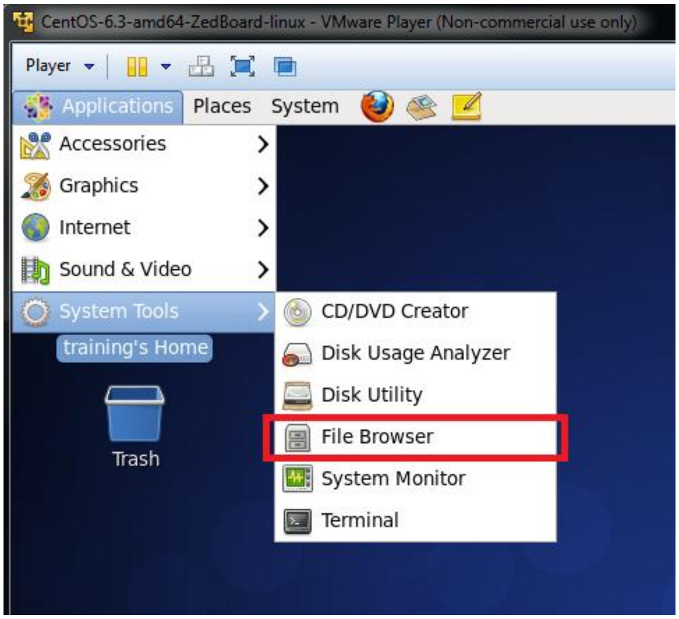


Figure 11 – CentOS File Browser

9. Using the CentOS File Browser on the guest operating system, browse to the **/home/training/linux-xlnx/** folder. Right click in open white space between folder icons and select the **Paste** option to paste the file into the following source code tree folder:

/home/training/linux-xlnx/

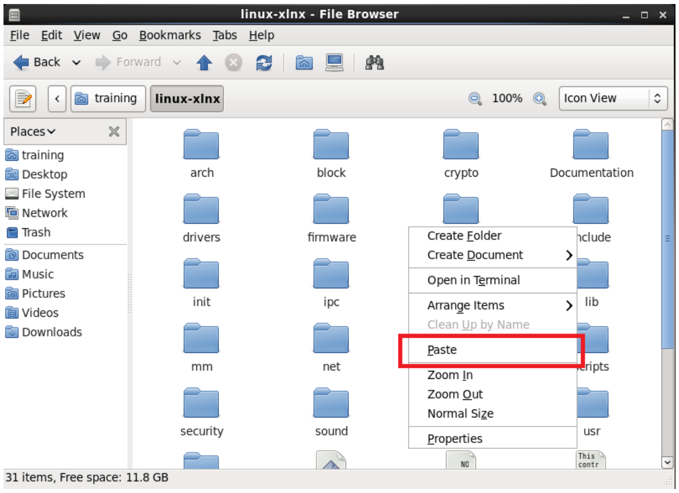


Figure 12 – Kernel Patch Pasted to the Guest Machine

10. Using the CentOS terminal, apply the patch to the kernel tree.

    Ignore any warnings about file permissions or types.

```
$ git apply 0001-Xilinx-ARM-Kernel-32MB-RAM-disk-on-ZedBoard.patch
```

11. A Zynq configuration is included for Zynq targets and the Zynq kernel configuration can be seen in the **/arch/arm/configs/xilinx_zynq_defconfig** file. This configuration was modified by the source code patch from the previous steps.

    Configure the Linux Kernel for the Zynq target by using the Zynq defconfig.

```
$ make ARCH=arm xilinx_zynq_defconfig
```

Questions:

Answer the following questions:

- *Which folder contains the Linux Kernel ARM target platform default configuration files?*

____

Experiment 3: Building the Kernel from a Configured Source

Now that the kernel source tree has been configured for the Zynq target platform, it can be built using the cross toolchain.

Experiment 3 General Instruction:

Build Xilinx Linux kernel for the Zynq target platform using the cross toolchain and copy the Kernel Image to the Lab2.3 folder under the host operating system.

Experiment 3 Step-by-Step Instructions:

1. In the previous experiment, the kernel source tree was configured for a Zynq target platform.

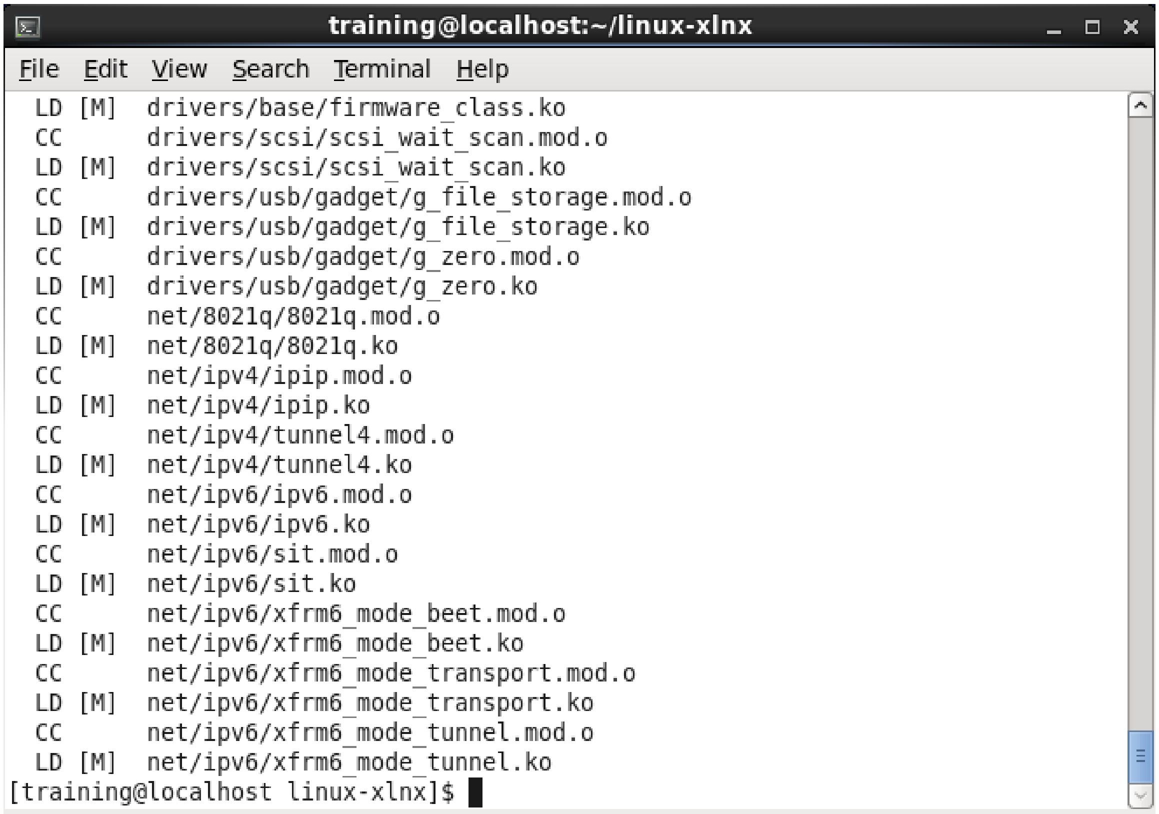
   First, make sure the **/home/training/linux-xlnx/** folder is the current working directory by using the pwd command. If the working directory shown is not the **/home/training/linux-xlnx/** folder change directories to that folder.

```
$ pwd
```

2. Once the working directory is the **/home/training/linux-xlnx/** folder, build the kernel source with the make command.

   The build process should take about 10 to 20 minutes to complete. If the build is successful and the console output looks similar to that shown in Figure 13, proceed to Step 4.

```
$ make ARCH=arm
```


training@localhost:~/linux-xlnx
File Edit View Search Terminal Help

```
  LD [M]  drivers/base/firmware_class.ko
  CC      drivers/scsi/scsi_wait_scan.mod.o
  LD [M]  drivers/scsi/scsi_wait_scan.ko
  CC      drivers/usb/gadget/g_file_storage.mod.o
  LD [M]  drivers/usb/gadget/g_file_storage.ko
  CC      drivers/usb/gadget/g_zero.mod.o
  LD [M]  drivers/usb/gadget/g_zero.ko
  CC      net/8021q/8021q.mod.o
  LD [M]  net/8021q/8021q.ko
  CC      net/ipv4/ipip.mod.o
  LD [M]  net/ipv4/ipip.ko
  CC      net/ipv4/tunnel4.mod.o
  LD [M]  net/ipv4/tunnel4.ko
  CC      net/ipv6/ipv6.mod.o
  LD [M]  net/ipv6/ipv6.ko
  CC      net/ipv6/sit.mod.o
  LD [M]  net/ipv6/sit.ko
  CC      net/ipv6/xfrm6_mode_beet.mod.o
  LD [M]  net/ipv6/xfrm6_mode_beet.ko
  CC      net/ipv6/xfrm6_mode_transport.mod.o
  LD [M]  net/ipv6/xfrm6_mode_transport.ko
  CC      net/ipv6/xfrm6_mode_tunnel.mod.o
  LD [M]  net/ipv6/xfrm6_mode_tunnel.ko
[training@localhost linux-xlnx]$
```

Figure 13 – Linux Kernel Build Completed

3. If you run into strange errors regarding problems locating your cross toolchain, the terminal **PATH** environment variable may not be set correctly. One way to resolve this is to pick up the updated user profile using the source command from the **/home/training/** folder.

   Return to the Linux kernel source folder, then clean the source tree with **make distclean** and go back to Step 2 to rebuild. This should help resolve any segmentation faults encountered with the toolchain.

```
$ cd ~

$ source .bash_profile

$ cd linux-xlnx/

$ make distclean
```

The terminal PATH environmental variable tells the terminal shell which directories to search for executable files in response to commands issued by a user. This environment variable increases both the convenience and the safety of command line based operating systems and is widely considered to be the single most important environmental variable. The PATH environment variable can be checked by echoing it to the command line. Check it to be sure that it contains a path to the Sourcery CodeBench installed /bin folder.

```
$ echo $PATH
```

4. If a file browser window is not already open from a previous exercise, open a file browser window through the **Applications→System Tools→File Browser** menu item.

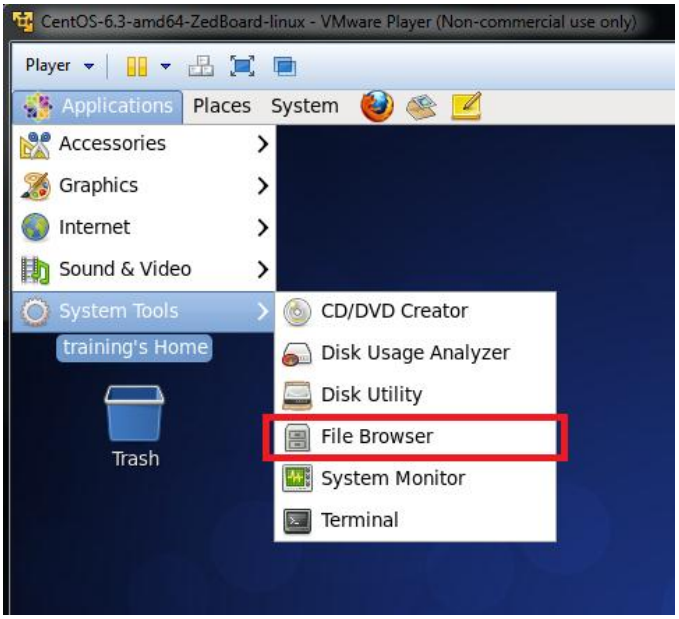
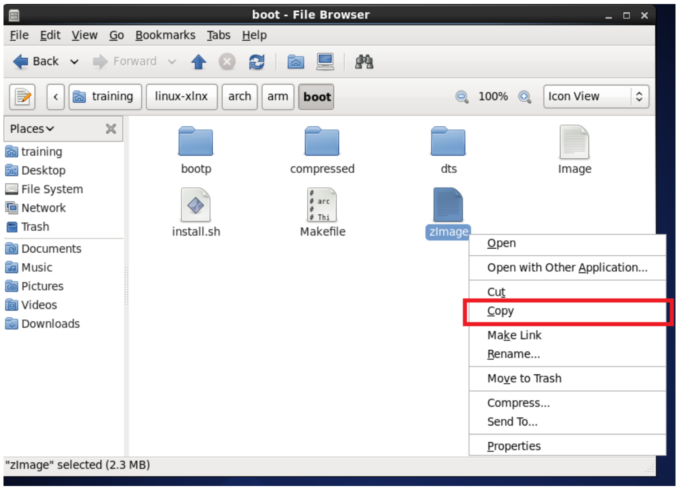


Figure 14 – CentOS File Browser

5. Locate the file **/home/training/linux-xlnx/arch/arm/boot/zImage** which is the target executable image needed to execute on Zynq. Right click on this file and select the **Copy** option which will place the selected file in the Virtual Machine clipboard.



"zImage" selected (2.3 MB)

Figure 15 – Copying Kernel Image to Virtual Machine Clipboard

6. Paste the kernel image into the Lab 2.3 folder under the host operating system by using Windows Explorer to navigate to the following folder:

   C:\Speedway\Fall_12\Zynq_Linux\lab2_3

   This kernel image file will be used in a later lab session. You may have also noticed a **devicetree.dtb** file in this folder. This file serves as the platform description used by the kernel to initialize the hardware and related drivers. This file is provided as a starting point for our system but we will discuss this file in further detail later in Part 3 and Lab 3.3.

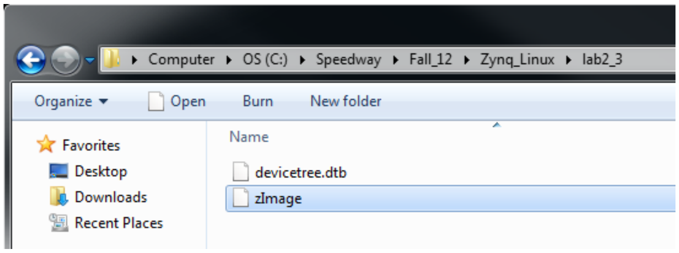


Figure 16 – Kernel Image Copied to the Host Machine

Questions:

Answer the following questions:

- *Which kernel build target file has the image format that is needed for execution on the Zynq platform?*

____

Exploring Further

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

- Take a look through the Zynq default configuration file **/arch/arm/configs/xilinx_zynq_defconfig** and determine which components are used to build the Linux kernel.

This concludes Lab 2.1.

Revision History

Date	Version	Revision
10 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://git-scm.com/

http://www.kernel.org

Answers

Experiment 1

- *How is a working copy of the Xilinx Linux Kernel code base obtained?*

A working copy is obtained using the Git clone command against the repository URL.

Experiment 2

- *Which folder contains the Linux Kernel ARM target platform default configuration files?*

/home/training/linux-xlnx/arch/arm/configs/

Experiment 3

- *Which kernel build target file has the image format that is needed for execution on the Zynq platform?*

zImage