

Contents

[KeyStone II Workshop Ubuntu Server 4](#_Toc387929310)

[Introduction 4](#_Toc387929311)

[1. Installing Ubuntu 6](#_Toc387929312)

[Prereqs.pl 6](#_Toc387929313)

[Operating system 6](#_Toc387929314)

[64 BIT Server 6](#_Toc387929315)

[2. Server Directory Structure 7](#_Toc387929316)

[Server Directory Structure 7](#_Toc387929317)

[3. Software Download 9](#_Toc387929318)

[4. Projects Sources 10](#_Toc387929319)

[5. Git Directories 10](#_Toc387929320)

[Utilities 11](#_Toc387929321)

[1. TFTP 11](#_Toc387929322)

[2. FTP 12](#_Toc387929323)

[3. NFS Server 13](#_Toc387929324)

[4. Create Student Accounts 15](#_Toc387929325)

[5. VNC Server 16](#_Toc387929326)

[6. DHCP Server 17](#_Toc387929327)

[Other Utilities that nice to have 20](#_Toc387929328)

[Picocom 20](#_Toc387929329)

[Samba Server 20](#_Toc387929330)

[Connect to the Networks 22](#_Toc387929331)

[Step 1 – Connect the wireless Network 22](#_Toc387929332)

[Step 2 – Adding the local wired Network 23](#_Toc387929333)

[Appendix 24](#_Toc387929334)

[Appendix A: Student laptop and startup summary 24](#_Toc387929335)

[Appendix B: Student Script inside TI network 25](#_Toc387929336)

[Appendix C: git script inside TI network 25](#_Toc387929337)

[Appendix D: Student script for Ti Guest network 26](#_Toc387929338)

[Appendix E: Cheat Sheet (thanks to DZ) 27](#_Toc387929339)

# KeyStone II Workshop Ubuntu Server

## Introduction

With the introduction of KeyStone II, TI multicore offering includes ARM CorePac with one, two or four A15 processors. TI offers an EVM card for its customers to speed up development.

Booting KeyStone II EVM is done using U-Boot and requires tftp server available on the same subnet as the EVM. In addition, the toolchain that are used do not run on the target EVM board, but rather on a Linux server. TI uses Ubuntu as the standard Linux operating system.

In KeyStone II workshop we expect to have 10 KeyStone II EVM and about 20 students. Part of the workshop is to develop new code that runs on the A15 Linux system. Since the tool chain runs of Linux, if we want all the students to take part in developing and build application, each student must have access to an Ubuntu machine on the same subnet as the EVM.

TI training equipment includes a set of Windows laptops. To provide Ubuntu access to each student, either Virtual machine must be installed on the Laptops with Ubuntu operating system, or a dedicated Ubuntu server must be available.

While most of the training sessions happen in TI locations, customers training may take place in customer locations. We need to advice a strategy for both cases.

The virtual machine model is strongly discouraged by TI IT team. It presents multiple IP addresses on the same device.

Updating the release and the development tools requires network access with the right proxy settings. Obviously this must be done using the guess network in TI offices (and a similar customers network if the training is in customer site) so that customers cannot have access to internal TI network.

The next page describes the suggested local network organization:



## Installing Ubuntu

Follow the instructions on <http://blwiki.dal.design.ti.com/index.php/TI_Ubuntu_12.04_Installation>

Do not install Kerberos; do not install the site baseline and not the workstation baseline.

### Prereqs.pl

In the Ubuntu instructions from above there are instructions how to get and run prerrqs.pl. The Wget instruction does not work for me. Instead I do the following procedure:

1. First update the proxy setting in the network configurations to [**http://proxyconfig.itg.ti.com/proxy.pac**](http://proxyconfig.itg.ti.com/proxy.pac)
2. Access the URL <http://ubuntu.dal.design.ti.com/tub.chaut/prereqs.pl> using the browser
3. Copy the complete file and paste it into a file /tmp/prereqs.pl
4. Execute the script – sudo /usr/bin/perl /tmp/prereqs.pl
5. Update the apt-get again sudo apt-get update
6. Do not install Kerberos or the baseline package

### Operating system

TI version of Ubuntu 12.04 with enabled local user is the server operating system. This version is used by TI team to develop the KeyStone Linux code. It is very important to enable super user authority to the local user.

The latest install of Ubuntu on my server is 13.10. The instructions will cover both operating systems

### 64 BIT Server

Some of the installed packages are made for 32-bit machines. In order to run them on 64-bit machines, one has to install an adaption / compatibility library ia32-libs.

Information on installing ia32-libs is available for example in <http://askubuntu.com/questions/143904/how-to-install-ia32-libs-on-12-04-lts-32-bit>

The following instructions are used to install ia32-lins:

* **sudo apt-get install ia32-libs**

Note – if the Ubuntu installation is for 32-bit operating system, the apt-get will not install ia32-libs. So if you get an error message saying that “the package does not exist”, the Ubuntu must be 32-bit version.

The version that is installed currently on the server is 32 bit version, so ia32-libs is not needed.

## Server Directory Structure

# Server Directory Structure

The following directories and sub-directories were added to facilitate the workshop (Complete path):

|  |  |  |
| --- | --- | --- |
| **Directory** | **Purpose** | **Comments** |
| /tiTools | Contains TI tools that are to be used by all students: Linaro cross compiler tool chain, CCS, and MCSDK | Sub-directories are MCSDK, CCS and Linaro tools chain (cross compiler) |
| /tiTools/CCS | CCS installation location |  |
| /ti/TOOLS/MCSDK\_X\_YY | MCSDK installation directory, version number is X\_YY |  |
| /tiTools/ gcc-linaro-arm-linux-gnueabihf-4.7-2013.03-20130313\_linux | Linaro tools chain – cross compiler |  |
| /tftpboot | Root directory for the TFTP server. | Each student has a sub-directory |
| / tftpboot /studentN | TFTP directory for student N, where N is 1 .., 10 | Student has to copy images from the MCSDK to this directory for ramfs boot |
| /opt/filesys | Root directory for the NFS server that enables mounting of the server file system into the EVM | Each student has a sub-directory |
| /opt/filesys /studentN | NFS directory for student N, where N is 1..,10 | Each student should build private file system into this directory |
| /usr/global/scripts | This directory has scripts that initialize environment variables. scriptInsideTI.sh is used when the server is inside TI network, and scriptOutsideTI.sh is used when the server is outside TI firewall. Other scripts may be developed for other locations | The student must run the script for every terminal by doing  Source /tiScript/scriptXXX.sh |
| /usr/global/Projects | Contains the source code for projects that are used during the Lab. | It has two sub-directories, DSP and ARM. Students will copy the source code files from this directory to their private directories. |
| /usr/global/Projects /ARM | Source code for ARM projects | myHello has the myHello.c project |
| /usr/global/Projects /DSP | Source code for DSP projects |  |
| /home/studentN | Home private directory of student N N=1 ..,10 | All changes to files are done in the student private directory |
| /usr/global/git | All sources for TI Arago distribution |  |

Use the command “sudo mkdir /XXX” to create the /XXX directory

## Software Download

**The following instructions are for loading software from TI network. Outside of TI network proxies and scripts may be different**

1. Make sure that the wireless adapter is disabled.
2. Connect the wire network to TI network. Make sure to enable proxy in the

setting->network->proxy tab to http://proxyconfig.itg.ti.com/proxy.pac

1. Create the directories that are defined in the table above
2. Get a set of scripts (either from the business unit trainers or copy them from the appendix) and store them in directory /tiScript. The following three scripts are needed: git-proxy\_ubuntu.sh scriptInsideTI.sh scriptOutside.sh. The scripts define environment variables.
3. Go to <http://software-dl.ti.com/sdoemb/sdoemb_public_sw/mcsdk/latest/index_FDS.html> and install the following:
   1. Latest MCSDK release for Linux
   2. CCS release (enable C6000 multicore) for Linux
   3. Emulation package for Linux
4. Install MCSDK in the directory /tiTools/MCSDK\_X\_Y where X and Y are the release version, currently 3\_18. The directory /tiTools/MCSDK\_X\_Y is built per the table above.
5. Install CCS in directory /tiTools/CCS,.
6. Install the emulation package following the install instructions inside the CCS release location.
7. Install the Linaro toolchain. Download the tool chain from the following link: <https://launchpad.net/linaro-toolchain-binaries/trunk/2013.03/+download/gcc-linaro-arm-linux-gnueabihf-4.7-2013.03-20130313_linux.tar.bz2>
8. Copy the compress file into directory /tiTools and decompress it using the following -

tar xjf gcc-linaro-arm-linux-gnueabihf-4.7-2013.03-20130313\_linux.tar.bz2

1. Change the new directory name gcc-linaro-arm-linux-gnueabihf-4.7-2013.03-20130313\_linux. To gcc
2. Add the following lines to the scripts scriptInsideTI.sh and scriptOutside.sh (if it is not already)

export CROSS\_COMPILE=arm-linux-gnueabihf-

export ARCH=arm

PATH=/tiTools/gcc/bin:$PATH

1. Update apt-get. Do “sudo apt-get update”. Make sure that the update process has no errors. If there are errors, check the setting of the proxies.

## Projects Sources

The directory /usr/global/projects has two subdirectories, ARM and DSP.

The source code for all Lab projects can be downloaded from [\\158.218.104.34\gguser\training\KeyStone\_2\_Labs\tiProjects](file:///\\158.218.104.34\gguser\training\KeyStone_2_Labs\tiProjects) Copy the two subdirectories into /usr/global/projects directory and make sure that eve y user can read the source code.

## Git Directories

* 1. Update the script scriptInsideTI.sh if it was not done before by adding:

export http\_proxy="http://webproxy.ext.ti.com:80"

export ftp\_proxy="http://webproxy.ext.ti.com:80"

export https\_proxy="http://webproxy.ext.ti.com:80"

export no\_proxy=".ti.com"

* 1. Run the script again (source /usr/global/scripts/scriptInsideTI.sh)
  2. Configure git and install packages needed at build time (copy and paste the second long line):

sudo apt-get install git-core

sudo apt-get install build-essential subversion ccache sed wget cvs coreutils unzip texinfo docbook-utils gawk help2man diffstat file g++ texi2html bison flex htmldoc chrpath libxext-dev xserver-xorg-dev doxygen bitbake uboot-mkimage libncurses5-dev

* 1. The purpose of installing the git repository in this point is to provide source code in a case where the external network is not available during the training. The directory /usr/global/git is used to pre-load source code. Switch user to root (sudo –s ) or give full permission to manipulate the /tiTools/git directory (sudo chmod 777 –R /tiTools/git).The following process copies the source code to each of the directories and reset the version to the head version as it is specifies in the release notes. The following uses the labels from release 3.0.18:

cd /tiTools/git

git clone git://git.ti.com/keystone-linux/u-boot.git u-boot-keystone

cd u-boot-keystone

git reset --hard << the TAG name from the release notes>>

Note – If the system does not recognize the tag, do git tag and it will give you all the tags that are available. Choose one that is very similar to the “release notes” tag. The same is true to all git reset commands

make tci6638\_evm\_config

make u-boot-spi.gph

cd ..

git clone git://git.ti.com/keystone-linux/boot-monitor.git

cd boot-monitor

git reset --hard K2\_BM\_13.11 <<or similar>>

make clean

make

cd ..

git clone git://git.ti.com/keystone-linux/linux.git linux-keystone

cd linux-keystone

git reset --hard K2\_LINUX\_03.10.10\_14.03\_03 <<or similar>>

Note – if the kernel build has an error, export the following path: /usr/global/git/tiArago/u-boot-keystone/tools

make keystone2\_defconfig

make uImage

make k2hk-evm.dtb

## Utilities

### TFTP

1. For more information look at - <http://icesquare.com/wordpress/how-to-setup-tftp-on-ubuntu/>
2. Instruction command - “ **sudo apt-get install xinetd tftpd tftp”**
3. Files that are built - **/etc/xinetd.d/tftp**
4. Instructions to build/modify the file /etc/xinetd.d tftp:
   1. “**sudo vi /etc/xinetd.d/tftp**”
   2. **/etc/xinetd.d/tftp** file should have the following lines:

**service tftp**

**{**

**protocol = udp**

**port = 69**

**socket\_type = dgram**

**wait = yes**

**user = nobody**

**server = /usr/sbin/in.tftpd**

**server\_args = tftpboot**

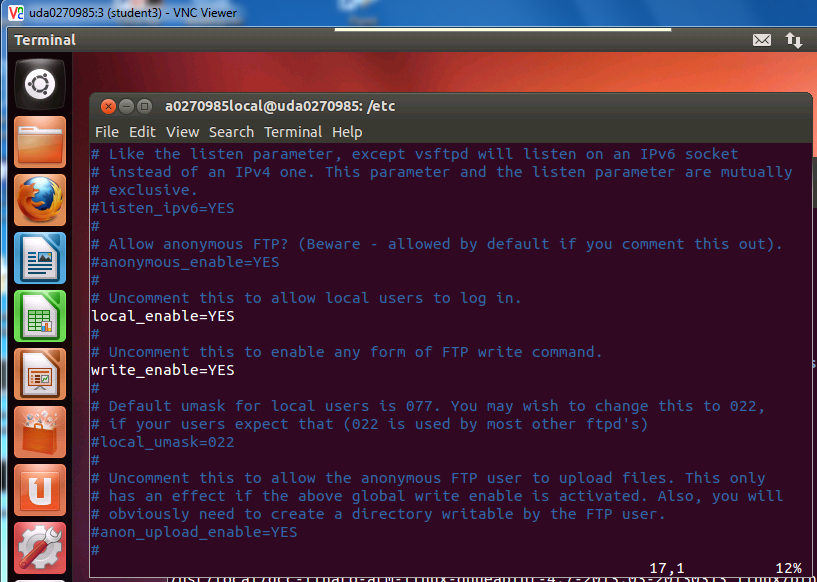
**disable = no**

**}**

1. **/tftpboot** is the root directory for tftp. Each student must have a dedicated sub-directory / **tftpboot** /studentN where N goes from 1 to 10
2. To start the tftp server do **“sudo service xinetd restart”**

### FTP

1. For more information look at in <https://help.ubuntu.com/10.04/serverguide/ftp-server.html>
2. Instruction command - “ **sudo apt-get install vsftpd ”**
3. Files that are built **- /etc/vsftpd.conf**
4. Instructions to build/modify the file **/etc/vsftpd.conf**:
   1. “**sudo vi /etc/vsftpd.conf**”
   2. Comment out anonymous enable
   3. Uncomment out local\_enable and write\_enable
   4. A screenshot of the file is given below
5. To use the ftp server, each user must login with its own user name and password
6. To start the tftp server do **“sudo /etc/init.d/vsftpd restart”**



### NFS Server

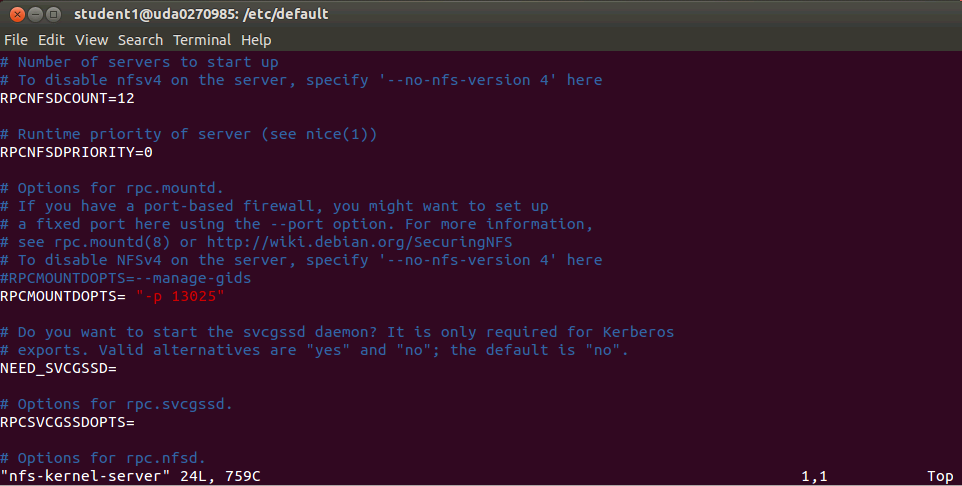
For more information look at in <https://help.ubuntu.com/community/SettingUpNFSHowTo>

1. Instruction command - “ **sudo apt-get install nfs-kernel-server ”**
2. Files that are built **- /etc/default/nfs-kernel-server**  and **/etc/exports**
3. Instructions to build/modify the file **/etc/ default/nfs-kernel-server** :
   1. “**sudo vi /etc/ default/nfs-kernel-server** ”
   2. Change number of servers – RPCNFSDCOUNT to 12 , **RPCNFSDCOUNT = 12**
   3. Specify the port number that NFS use. Change the RPCMOUNTDPORTS to an available port. We use port 13025, **RPCMOUNTDPORTS = “ –p 13025”**
4. A screenshot of the file is given below
5. Instructions to build/modify the file **/etc/exports**:
   1. **“sudo vi /etc/exports”**
   2. Add the following line to it

**/opt/filesys \*(rw, subtree\_check, no\_root\_squash,no\_all\_squash,sync)**

1. A screenshot of **/etc/exports** is given below
2. Build the nfs directories for each student (if students’ sub-directories are not built already)
   1. The main directory is /opt/filesys
   2. **“sudo mkdir /opt/filesys /studentN”** where N is student number, 1 to 10
3. To start the tftp server do **“sudo /etc/init.d/nfs-kernel-server restart”**
4. **Make sure that they are not additional blanks between the items in the configuration files**

Screen Shot of the file /etc/ default/nfs-kernel-server



Screen Shot of the file /etc/ exports



### Create Student Accounts

For more information how to add a user look at <https://www.digitalocean.com/community/articles/how-to-add-and-delete-users-on-ubuntu-12-04-and-centos-6>

1. Repeat the following procedure for each student where N is the student number, N goes from 1 to 10
2. Instruction command to add an account “ **sudo adduser studentN ”**
3. Enter the following values (if the value is not mention here just press return):
   1. Enter new UNIX password **“WsN”** where N goes from 1 to 10
   2. Retype new Unix password **“WsN”** where N goes from 1 to 10
   3. Full Name **studentN**
   4. Is the information correct? **y**
4. To add privileges for sudo the file **/etc/sudoers** should be modified. Here are the instructions:
   1. **Sudo /usr/sbin/visudo**
   2. Add the following line for each student (N goes from 1 to 10)

**studentN ALL= (ALL: ALL) ALL**

1. In addition to the home directory, each user has two more dedicated directories. One is in **/tftpboot/studentN** and a second one is in **/opt/filesys/studentN.** Verify that these directories exist and create them if not when you add any new user. Note, you may have to create the /opt/filesys directory as well.

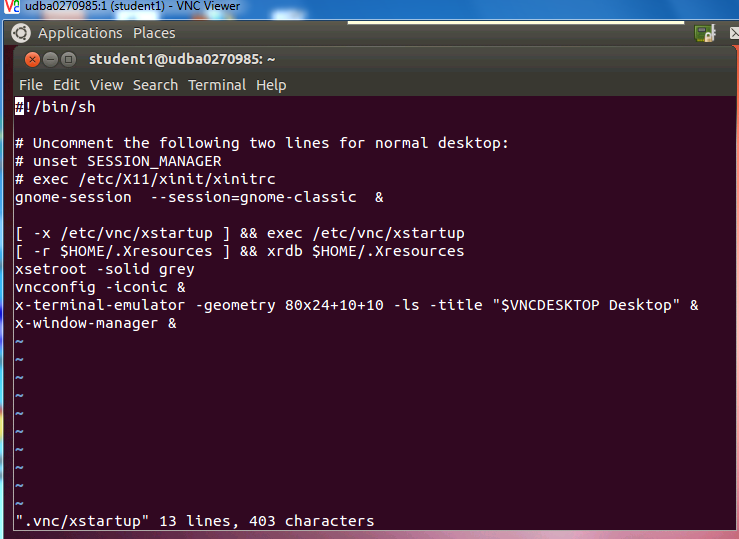
**(“ sudo mkdir /tftpboot/studentN”**

**“sudo mkdir /opt/filesys/studentN”)**

### VNC Server

For more information look at in <http://rbgeek.wordpress.com/2012/06/25/how-to-install-vnc-server-on-ubuntu-server-12-04/>

1. Instruction command for graphic interface - “ **sudo apt-get install gnome-core ”**
2. Instruction command for vnc “ **sudo apt-get install vnc4server ”**
3. Files that are built for each user**- .vnc/xstartup**
4. Instructions to start vnc instance for each student:
   1. IMPORTANT – do the following for each student in the correct order, that is, do student1, and only after it is finished do student2 and so on and so forth
   2. “**su - studentN** ” where N goes from 1 to 10(notice the dash – between the su and the student name)
   3. **“vncserver”**
   4. Set the password to **vncserve**
   5. Make sure that the instance number (the number after the : in the log file) agrees with the student number
5. After vnc start, do the following for each student
   1. vncserver –kill :N where N is the vnc number (the student number)
   2. Add the line **gnome-session –session=gnome-classic &** to .vnc/xstartup file for each student (you can copy the file from the first user after you have changed the first user file.
   3. A screen shot of the is given below
6. Instructions how to connect a remote terminal will be part of the Lab book



### DHCP Server

For more information look at in <http://rbgeek.wordpress.com/2012/04/29/how-to-install-the-dhcp-server-on-ubuntu-12-04lts/>

1. Instruction command - “ **sudo apt-get install isc-dhcp-server ”**
2. Files that are built **- /etc/default/isc-dhcp-server**  and **/etc/dhcp/dhcpd.conf**
3. Instructions to build/modify the file **/etc/ default/ isc-dhcp-server**  :
4. “**sudo vi /etc/ default/ isc-dhcp-server**  ”
5. The DHCP will use the wire interface eth0, thus the value next to the INTERFACE line must be **eth0**
6. A screenshot of the **/etc/default/isc-dhcp-server**  is given below
7. Instructions to build/modify the file **/etc/dhcp/dhcpd.conf**:
   1. **“sudo vi /etc/dhcp/dhcpd.conf”**
   2. Domain name is defined as keyStone2-class
   3. Add (or uncomment and modify the lines)

**subnet 192.168.0.0 netmask 255.255.255.0 {**

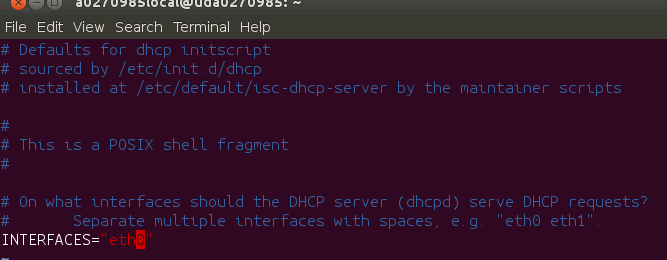
**Range 192.168.0.10 192.168.0.90 ;**

**Option broadcast-address 192.168.0.254 ;**

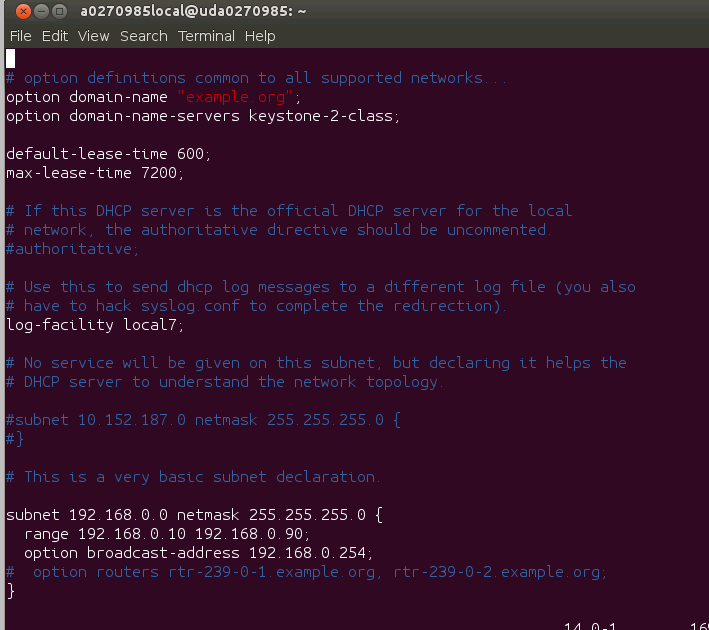
**}**

1. A screenshot of **/etc/ dhcp/dhcpd.conf** is given below
2. **Note – Start the DHCP server ONLY AFTER the wire interface was configured for the local network (see later) and is not connected to external network or TI network.**
3. To start the tftp server do **“sudo service isc-dhcp-server restart”**

The file **/etc/default/isc-dhcp-server**



The file **/etc/dhcp/dhcpd.conf**



# Other Utilities that nice to have

## Picocom

The utility picocom is a simple Ubuntu terminal emulator that can be used to look at the EVM terminal windows. Instructions how to install picocom can be found in <https://developer.ridgerun.com/wiki/index.php/Setting_up_Picocom_-_Ubuntu> . Here is a summary

1. sudo apt-get install picocom
2. Before connecting the USB cable to the evm, do ls -ltr /dev and see what device are currently connected to the Ubuntu machine
3. Connect the USB cable and repeat the process again ls –ltr /dev. Note what new devices were added, it may be /dev/ttyUSBN and ttyUSB(N+1). The first port is the ARM terminal, the second is the BMC terminal that controls the power sequence. Assume that the two new devices are ttyUSB5 and ttyUSB6 then do the following:
4. To connect to the ARM window do: picocom –b 115200 –r –l /dev/ttyUSB5
5. To connect to the BMC terminal do: picocom –b 115200 –r –l /dev/ttyUSB6

## Samba Server

Samba server enables window based laptop to read and write files on the Ubuntu machine and thus eliminates the need for ftp between the Ubuntu server and the students’ laptops.

For more information look at in <http://ubuntuserverguide.com/2012/06/install-samba-server-ubuntu-server-1204-lts.html>

1. Instruction command - “ **sudo apt-get install samba ”**
2. File that is modified **- /etc/samba/smb.conf**
3. Instructions to build/modify the file **/etc/samba/smb.conf** :
   1. “**sudo cp /etc/samba/smb.conf /etc/samba/smb.conf.orig** ”
   2. In the file /etc/samba/smbfs add one or more share section as shown below

**[share]**

**comment = Precise File Server**

**path = /srv/samba/share << or any other path that you want to share>>**

**browsable = yes**

**guest ok = no**

**read only = no**

**create mask = 0755**

* 1. Create directory for file sharing sudo mkdir –p /srv/samba/share
  2. sudo chown nobody.nogroup /srv/samba/share
  3. Start the samba server:

**sudo restart smbd**

**sudo restart nmbd**

Windows explorer will see all the share directories (share, share1, share2, etc.)

# Connect to the Networks

Up to this point the wireless interface on the server was disabled, and the wire interface was connected to TI network or external network. In the case of external network, the proxy definitions are different than in the document. The local IT person will have to advise on the correct proxy. If the external network is TI guest network, getting to the git server might not work.

First the wireless interface is configured to connect to TI guest network. The credentials needed for this connection will be provided by TI IT group. To ask for guest network credentials go to <http://guestnet.ti.com>. The order of connecting the interface is important.

## Step 1 – Connect the wireless Network

The wireless network is connected to the external Web. In my case, it is connected to TI guest network.

The network in North America is net4guest and the Wireless key is tiguest84. You have to ask for new credentials outside of the US

The wireless network appears as eth2. The interfaces file (/etc/network/interfaces) stay the default. Configure the interfaces using the GUI (setup -> network -> wireless and look at options to set the network, the encryption and so on). The following instructions are for TI guest network. For any other location, consult the local IT. The following are the steps that I took to configure the wireless network:

Step 1 – Log in into the machine as super user. Make sure that the wireless adapter is on (the small antenna icon is on). If the server was pre-config the wireless will connect to TI Guest network. To validate it open the setting window (the “gear” icon) and chose the network tab. Select the wireless option and see if the connection is on.

Step 2 – if the wireless status is not connected, (no status or trying to connect for a more than 5 minutes) follow the following procedure:

1. Disconnect the wired cable
2. Delete all the configurations from the /etc/Networkmanagement/system\_connections directory
3. In the setting->network->wireless screen click on the pull down menu of network name. If net4guest is there, select it and enter the key when asked. If net4guest is not in the pull down menu, chose other, enter the name net4guest and continue
4. Reconfigured the netguest network with 128 bit encryption
5. Enter the encryption key
6. Reboot with super user credential
7. When it came up it asked (again) for the key, enter it
8. Connected
9. Start a browser and put user name and password (for the guest account)

Step 3 – Verify connectivity to the external world and especially to apt-get server

1. Open Firefox browser and make sure that you can get to standard web site such as cnn.com or google.com
2. Verify that you can get to the apt-get server. Do “**sudo apt-get update”** and see if the update process works.
3. The following concerning Ubuntu server that was installed with TI installation. For other installations similar procedure must be applied. If you get errors because apt tries to access the cache in dal.design.ti.com and cannot resolve or access (or similar inside firewall addresses) you need to change the proxy configuration. Here are the steps that I took:
   1. At directory /etc/apt/apt.conf.d delete the file proxy (that is change its name, do “sudo **mv proxy proxy.bak**”
   2. At directory /etc modify the file environment (first copy the original to environment.bak and then) delete all the lines that define the proxy do
      * 1. Do “**sudo cp environment environment.bak**”
        2. Open an editor and delete the lines that define any of the proxies
   3. Verify that there is not proxy defined. Do **“printenv | grep proxy”** to verify that no proxy is define. If you still have http, https or ftp proxy run a script that empty these proxies. The script should have lines like **“ export http\_proxy =”**
   4. The script file **studentNO\_TI.sh** in directory /**usr/training** has the needed definitions. The source of this script file is in the appendix.
4. Try “**sudo apt-get update**” again. Verify that it works. If it does not consult with the IT people.

## Step 2 – Adding the local wired Network

The Ubuntu server should have access to two networks. The local network 192.168.0.XXX will support all the EVM and the Laptops in the session. The local network should support at least 12 wired connections for the EVM and 20 wired or wireless connections for students’ Laptops. The local network has a DHCP server on a router or installed on the Ubuntu server.

Note, each EVM is a station, the number of stations is limited to 11.

During the operation, the wire network eth0 is connected to a local switch that supports all the users. The IP address of the server is set to 192.168.0.100. The setting is done using the GUI as explained above. Note, if external network is not available for wireless, the wired network should be connected to an external network and gets its IP address from a DHCP server on the network during the configuration process, and switch to local network afterwards.

Part of the setting is loading a dhcp server that provides IP addresses to all the clients on the local network. Instructions how to install and define the DHCP server are given in this document. The IP address range is between 192.168.0.10 to 192.168.0.90 – 80 addresses all together.

Note that regardless of the way the server is connected to the external world (wireless or wire), the proxy settings must be changed based on the instructions of the local IT person. In this document the proxy settings are for guest TI network.

To set the wired network do the following

1. Connect Ethernet cable to a local switch
2. Start the setting->network
3. Select the wire network and click on option
4. In the Method select manual
5. Add ip address (click on the add tab) with the following
   1. Ip -> 192.168.0.100
   2. Mask 255.255.255.0
   3. Gateway 192.168.0.254
6. Select the routes option and new window will open**. Check the “use this connection for resources only on this subnet”** (or similar).
7. Save the configuration
8. Start the DHCP as describes above
9. Connect a Laptop to the local switch and verify that the DHCP server is working
10. Ping the server from the laptop and back. Verify that the local network is working
11. Open a browser and verify that it still has connectivity to the external world (Google, cnn, yahoo, etc.)

# Appendix

## Appendix A: Student laptop and startup summary

1. Terminal

It is assumed that each student has a Windows Laptop with VNC viewer installed on it and Tera Terminal or similar. In addition, each student should have CCS pre-installed (CCSv5.4 or newer) and a GA version of MCSDK3.

1. Startup Summary

|  |  |
| --- | --- |
| Initial setting | /usr/local/training/studentScript.sh |
| Ifconfig eth0 | 192.168.0.100 |
| Ifconfig eth2 | Ip address of TI guest network – may requires login and password from a browser |
| DHCP | sudo service isc-dhcp-server restart |
|  |  |
| TFTP start | Sudo service xinetd restart (stop and start) |
| nfs | Sudo /etc/init.d/nfs-kernel-server restart |
| Vnc | Su – studentN vncserver |

## Appendix B: Student Script inside TI network

PATH=/tiTools/gcc/bin:/usr/sbin:/usr/bin:$PATH

export CROSS\_COMPILE=arm-linux-gnueabihf-

export ARCH=arm

export http\_proxy="http://webproxy.ext.ti.com:80"

export https\_proxy="http://webproxy.ext.ti.com:80"

export ftp\_proxy="http://webproxy.ext.ti.com:80"

export no\_proxy=".ti.com"

GIT\_PROXY\_COMMAND=/tiScript/git-proxy\_ubuntu.sh

PATH=/sbin:$PATH

## Appendix C: git script inside TI network

#!/bin/bash

exec /usr/bin/corkscrew webproxy.ext.ti.com 80 $\*

chmod +x git-proxy-ubuntu.sh

export GIT\_PROXY\_COMMAND=/tiScript/git-proxy-ubuntu.sh

## Appendix D: Student script for Ti Guest network

PATH=/tiTools/gcc-linaro-arm-linux-gnueabihf-4.7-2013.03-20130313\_linux/bin:/usr/sbin:/usr/bin:$PATH

export CROSS\_COMPILE=arm-linux-gnueabihf-

export ARCH=arm

export http\_proxy=

export https\_proxy=

export ftp\_proxy=

export no\_proxy=

GIT\_PROXY\_COMMAND=/tiScript/git-proxy\_ubuntu.sh

PATH=/sbin:$PATH

## Appendix E: Cheat Sheet (thanks to DZ)

==================================================

|| ||

|| MCSDK 3.0.3.15 quick notes ||

|| v0.1 last modified: 01/16/2014 ||

|| ||

==================================================

A. Prerequisite

==================

Follow Hardware setup guide to make sure you have the latest BMC version & UCD update

http://processors.wiki.ti.com/index.php/EVMK2H\_Hardware\_Setup

User guide: (getting started guide & exploring as well)

http://processors.wiki.ti.com/index.php/MCSDK\_User\_Guide\_for\_KeyStone\_II

Download link:

http://software-dl.ti.com/sdoemb/sdoemb\_public\_sw/mcsdk/latest/index\_FDS.html

Ubuntu Linux 12.04 LTS machine with proxy properly configured

Details about EVMK2H:

http://www.advantech.com.tw/Support/TI-EVM/EVMK2HX.aspx

B. Install CCS & MCSDK

================

1. Install CCSv5.5.x at <CCS default Installed Directory> ,say: ~/ti/

For processor architecture, make sure "C6x DSP + ARM processors" is selected.

2. Install TI KeyStone2 Emupack, ti\_emupack\_keystone2\_setup\_1.0.0.4

3. Install MCSDK 3.00.03.15, same directory as CCS, e.g.: ~/ti/

4. copy the file tci6638-evm.ccxml from mcsdk\_linux\_3\_00\_03\_15/host-tools/loadlin folder

to ~/<user>/ti/CCSTargetConfiguration folder where the CCS saves the user specific configuration file.

5. Jump to step K.5 if you want to run the out of box demo.

C. Build Prerequisite

==================

Toolchain installation:

Download the tool chain from below link:

https://launchpad.net/linaro-toolchain-binaries/trunk/2013.03/+download/gcc-linaro-arm-linux-gnueabihf-4.7-2013.03-20130313\_linux.tar.bz2

cd ~/

tar xjf gcc-linaro-arm-linux-gnueabihf-4.7-2013.03-20130313\_linux.tar.bz2

export CROSS\_COMPILE=arm-linux-gnueabihf-

export ARCH=arm

PATH=$HOME/gcc-linaro-arm-linux-gnueabihf-4.7-2013.03-20130313\_linux/bin:$PATH

Configure git and install packages needed at build time:

sudo apt-get install git-core

sudo apt-get install build-essential subversion ccache sed wget cvs coreutils unzip texinfo docbook-utils gawk help2man diffstat file g++ texi2html bison flex htmldoc chrpath libxext-dev xserver-xorg-dev doxygen bitbake uboot-mkimage libncurses5-dev

D. Building U-boot, boot monitor & kernel

==================

1. Create a separate folder for each MCSDK release

mkdir $HOME/mcsdk\_rls\_xx

cd $HOME/mcsdk\_rls\_xx

2. Set up proxy (please consult IT, refer to "Proxy Setup" in MCSDK User Guide: Exploring the MCSDK)

For TIer within TI network, if your proxy has not been properly setup, type in the following:

export http\_proxy="http://webproxy.ext.ti.com:80"

export ftp\_proxy="http://webproxy.ext.ti.com:80"

export https\_proxy="http://webproxy.ext.ti.com:80"

export no\_proxy=".ti.com"

Use below command to verify, especially pay attention to https\_proxy:

printenv | grep "proxy"

And also create a file git-proxy\_ubuntu.sh under ~/Documents folder, make the content to be:

#!/bin/bash

exec /usr/bin/corkscrew webproxy.ext.ti.com 80 $\*

chmod +x git-proxy\_ubuntu.sh

export GIT\_PROXY\_COMMAND=$HOME/Documents/git-proxy\_ubuntu.sh

Also add below under the [global] section of ~/.subversion/servers

#http-proxy-exceptions = \*.ti.com, \*.nsc.com

#http-proxy-host = webproxy.ext.ti.com

#http-proxy-port = 80

3. U-boot build instructions

git clone git://git.ti.com/keystone-linux/u-boot.git u-boot-keystone

cd u-boot-keystone

git reset --hard K2\_UBOOT\_2013-01\_13.11

option a: if using CCS to load uboot into SRAM (u-boot.bin)

make tci6638\_evm\_config

make

option b: if using the two stage SPI NOR boot

make tci6638\_evm\_config

make spl/u-boot-spl.bin

make tci6638\_evm\_config

make u-boot.img

make tci6638\_evm\_config

make u-boot-spi.gph

make clean before each new build

make distclean if you copied the whole folder from someone else

4. Boot monitor build instructions (skern.bin)

cd ..

git clone git://git.ti.com/keystone-linux/boot-monitor.git

cd boot-monitor

git reset --hard K2\_BM\_13.11

make clean

make

5. Linux kernel & device tree blob build instructions

5.1: to get the code

cd ..

Option 1: as customer to get the release

git clone git://git.ti.com/keystone-linux/linux.git linux-keystone

cd linux-keystone

git reset --hard K2\_LINUX\_03.10.10\_13.11\_01

Option 2 (TIer only): as TI developer to use the latest code still in the development

git clone git://gtgit01.gt.design.ti.com/git/projects/linux-keystone.git

cd linux-keystone

git reset --hard K2\_LINUX\_03.10.10\_13.11\_01

or to get the latest:

git checkout b217010339

replace b217010339 with the latest commit ID one can find from here:

http://gtgit01.gt.design.ti.com/git/projects/linux-keystone.git

5.2 to build kernel and DTB

make keystone2\_defconfig

make uImage

make k2hk-evm.dtb

The built files:

vmlinux is in /linux-keystone folder

uImage is in /linux-keystone/arch/arm/boot folder

k2hk-evm.dtb is in /linux-keystone/arch/arm/boot/dts folder

5.3 To Build Linux for Full RT Pre-empt mode, do the following

git reset --hard K2\_RT\_LINUX\_03.10.10\_13.11\_01

make keystone2\_fullrt\_defconfig

make uImage