

A large satellite dish antenna is positioned in a field of tall, dry grass. The dish is white and mounted on a metal structure. The background shows a clear blue sky with some clouds, and the sun is setting, creating a warm glow. The dish is angled towards the left side of the frame.

UltraZed-EV PCIe Root Complex Performance Test Tutorial

© 2018 Avnet. All rights reserved. All trademarks and registered trademarks are the property of their respective owners. All specifications are subject to change without notice.

NOTICE OF DISCLAIMER: Avnet is providing this design, code, or information "as is." By providing the design, code, or information as one possible implementation of this feature, application, or standard, Avnet makes no representation that this implementation is free from any claims of infringement. You are responsible for obtaining any rights you may require for your implementation. Avnet expressly disclaims any warranty whatsoever with respect to the adequacy of the implementation, including but not limited to any warranties or representations that this implementation is free from claims of infringement and any implied warranties of merchantability or fitness for a particular purpose.

Introduction

This document describes a Zynq UltraScale+ PCIe Root Complex design implemented and tested on the Avnet UltraZed-EV SOM + EV Carrier development board.

The Zynq UltraScale+ (ZU+) All Programmable System on Chip (SoC) includes the serial transceivers and an Integrated Block for PCI Express that can be configured as an Endpoint or Root Port, compliant to the PCI Express Base Specification Revision 2.1. The Root Port can be used to build the basis for a compatible Root Complex, to allow custom communication between the ZU+ SoC and other devices via the PCI Express protocol, and to attach ASSP Endpoint devices such as Ethernet Controllers or Wireless Adapters to the ZU+ SoC. This example describes a PCIe Root Complex System on an Avnet UltraZed-EV platform with the existing Xilinx IPs and standard Linux software drivers. Connectivity with an off the shelf Ethernet NIC endpoint card is demonstrated with this design.

The Zynq UltraScale+ (ZU+) SoC integrates a quad-core ARM Cortex-A53 based processing system (PS) and Programmable Logic (PL) in a single device. This example design demonstrates the ZU+ device being used as PCI Express Root Complex System and it shows the following components working together on UltraZed-EV development board platform.

Zynq UltraScale+ Processing System with integrated PCIe Root Complex interfacing with

- External 4GB DDR4 memory
- Software drivers (installed on Linux running on the UltraZed) to enumerate and exercise a PCI Express Endpoint connected to the Root Complex System.

The intent of this design is to provide basic use cases for customers to build their own applications. The example will be a building block for designs which use the PCI Express link for passing control/data from Zynq UltraScale+ PS to a PCIe endpoint.

DISCLAIMER: This tutorial is provided for reference/educational purposes only and may not reflect results observed with other test equipment.

There are a number of factors which can impact Ethernet network performance and throughput in addition to transmission overheads, including latency, packet size, and system caching such that the calculated throughput typically does not reflect the maximum achievable throughput. As a result, the throughput over the Ethernet network can be substantially lower than the theoretical limits.

Design Objectives

This UltraZed tutorial offers system developers an example of how to:

- Target a prebuilt Xilinx release of PetaLinux to UltraZed
- Launch PCIe and Ethernet performance tests on Zynq UltraScale+ platform using a test script running a prebuilt open source Linux build created with Xilinx PetaLinux Tools

Experiment Setup

This tutorial builds upon the concepts and lab activities of the Avnet UltraZed Tutorials which cover the use of Xilinx Vivado Design Suite in creating/testing a basic Zynq UltraScale+ MPSoC hardware platform and running software applications. Please refer back to this reference material on the UltraZed community website for further information on how to configure the underlying UltraZed hardware platform.

The experiments in this tutorial use the following Linux applications:

iperf3 - This utility measures maximum achievable throughput on TCP/IP networks

For the example PCIe and Ethernet test configuration that was used in this tutorial, see the section [Appendix I: Troubleshooting Ethernet Connection](#) later in this document for further information.

The instructions in this tutorial assume that the cross build platform is an Ubuntu 16.04 LTS installation running in a virtual machine on an x86-based host. Though other systems may work using the same or similar instructions, those systems are not supported.

Example Design Requirements

Software

The software used to test this reference design is:

- Ubuntu 16.04 LTS 64-bit Desktop
 - VirtualBox v5.1.30 virtual machine
 - Windows-7 64-bit host OS
- Xilinx Vivado Design Suite 2017.4 (Design Edition)
- Xilinx PetaLinux 2017.4
- Git SCM toolset (version used for this tutorial is v2.7.4)

Hardware

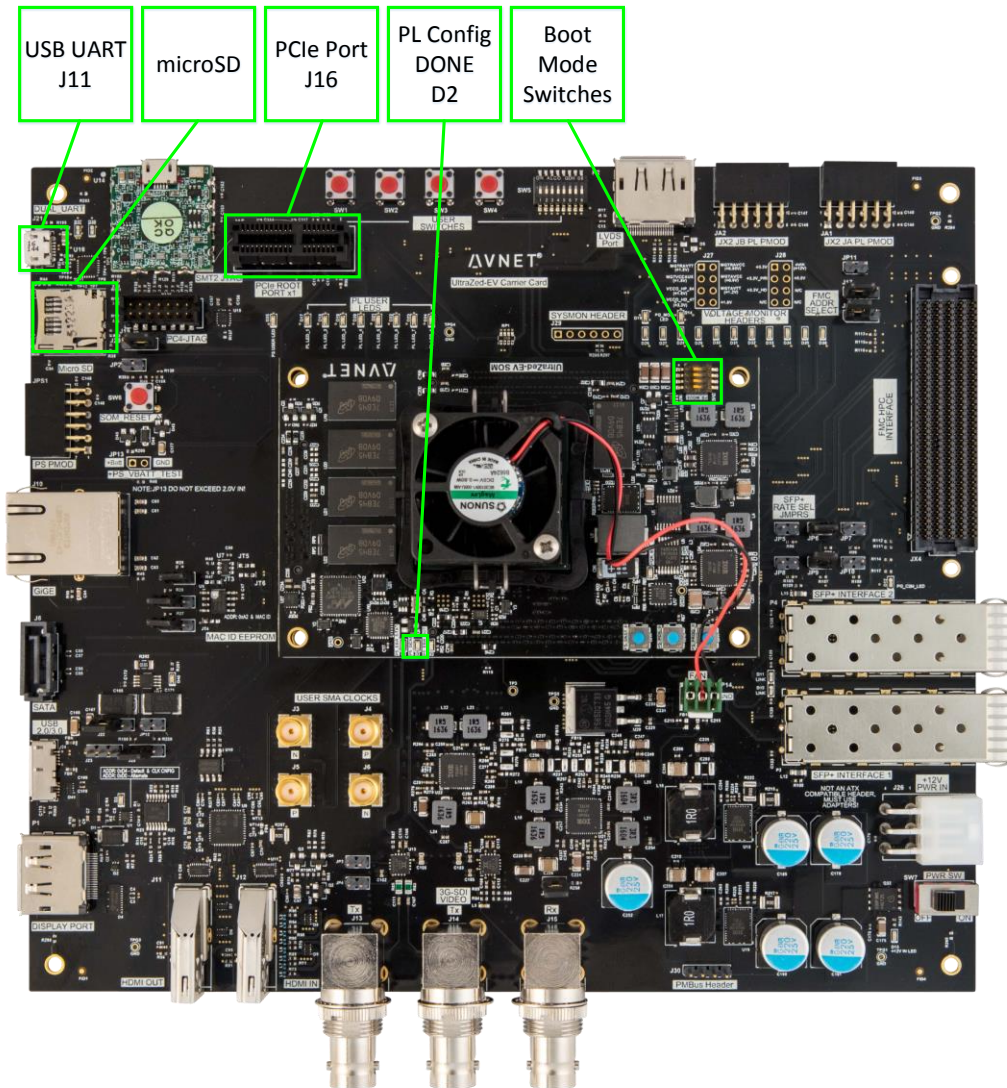
The hardware setup used to test this reference design includes:

- UltraZed-EV SOM (AES-ZU7EV-1-SOM-I-G) and EV Carrier Card (AES-ZUEV-CC-G)
- Lenovo ThinkPad T420 Laptop
 - Intel® Core i5-2540M CPU - 2.60 GHz
 - 4GB DDR3 Memory
 - SD card slot on PC or external USB SD card reader
- USB cable (Type A to Micro-USB Type B)
- 8GB MicroSD card
- CAT-5E Ethernet cable
- Broadcom "TIGON3" "BCM5751" (BCM95721A211) PCIe Ethernet NIC
 - <https://www.amazon.com/Broadcom-BCM95721A211-PCI-E-Network-Adapter/dp/B001G2K18M>
 - <https://www.newegg.com/Product/Product.aspx?Item=14U-004V-00034>

Experiment 0: Setting Up the UltraZed SOM with Carrier Card

UltraZed-EV with EV Carrier Card

Refer to the following figure and perform the following steps to set up the board when using the UltraZed-EV and EV Carrier.

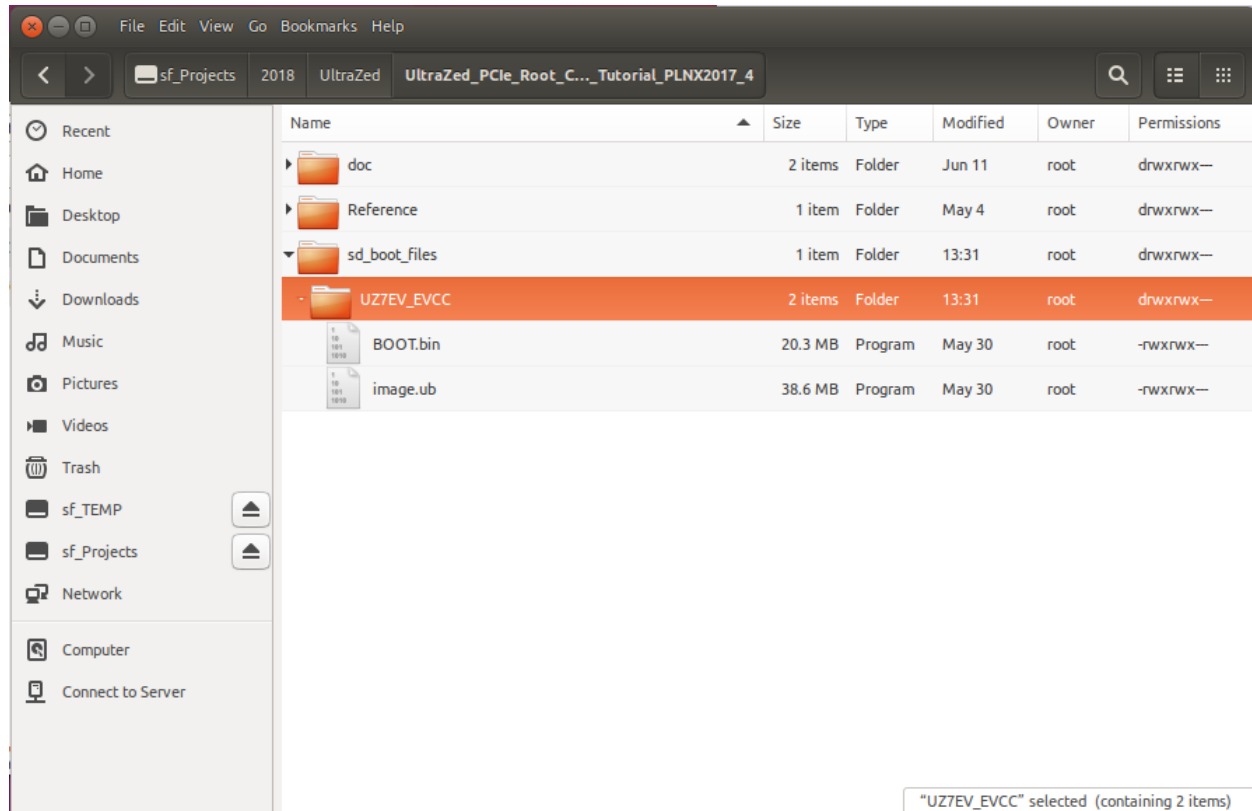


1. Plug the UltraZed-EV SOM onto the EV Carrier Card via JX1/JX2/JX3 connectors and connect the fan to the fan header (JP14) on the EV Carrier Card.
2. Set the UltraZed-EV SOM Boot Mode switch (SW2) (MODE[3:0] = SW2[1:4]) to OFF, ON, OFF, and ON positions (Boot Mode set to SD card, MODE[3:0] = 0x5).
3. Connect the USB-UART port on the EV Carrier Card (J21) to a free USB port on your PC.
4. Insert the PCIe Ethernet NIC into the PCIe slot connector (J16).
5. Connect the 12V power cable, but do not turn on the board yet.

Experiment 1: Setup Linux for UltraZed

The experiments in this tutorial are based upon the Linux build that is provided by Avnet as part of the UltraZed-EV PetaLinux 2017.4 Enhanced BSP available [here](#). For your convenience the SD card bootable binaries are included with this design tutorial.

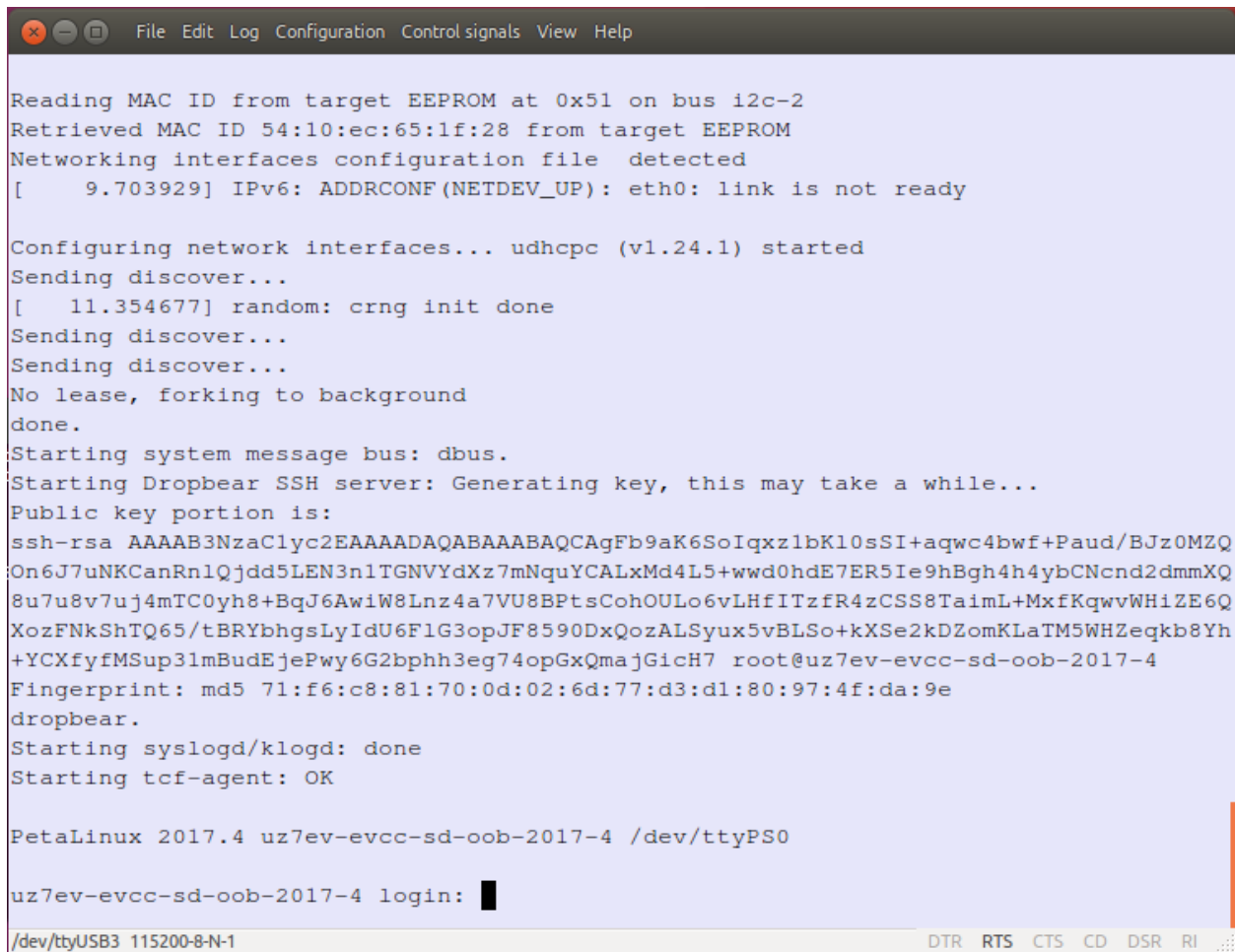
1. Copy the **BOOT.BIN** and **image.ub** files from the accompanying archive into the root folder of the microSD card.



Replace any existing versions of these files that may already be on the microSD card.

2. Insert the microSD card, prepared using the steps [above](#), into the UltraZed Carrier microSD card cage.
3. Set the UltraZed Carrier power switch to the ON position. The UltraZed system will power on and the Power Good LED should illuminate.
4. Launch a terminal program with the 115200/8/n/1/n settings. For the example output shown here, **gtkterm** was used. For information on setting up **gtkterm** to use with the UltraZed USB-UART port, see the section [Appendix II: Troubleshooting Serial Connection](#) later in this document for further information.

5. You should observe terminal output from U-Boot and then Linux output appear in the **gtkterm** window.



The image shows a screenshot of a terminal window titled 'gtkterm'. The window has a menu bar with 'File', 'Edit', 'Log', 'Configuration', 'Control signals', 'View', and 'Help'. The terminal output shows the following sequence of events:

```
Reading MAC ID from target EEPROM at 0x51 on bus i2c-2
Retrieved MAC ID 54:10:ec:65:1f:28 from target EEPROM
Networking interfaces configuration file detected
[ 9.703929] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready

Configuring network interfaces... udhcpc (v1.24.1) started
Sending discover...
[ 11.354677] random: crng init done
Sending discover...
Sending discover...
No lease, forking to background
done.
Starting system message bus: dbus.
Starting Dropbear SSH server: Generating key, this may take a while...
Public key portion is:
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCAgFb9aK6SoIqxz1bKl0sSI+aqwc4bwf+Paud/BJz0MZQ
On6J7uNKCanRnlQjdd5LEN3nlTGNVYdXz7mNquYCALxMd4L5+wwd0hdE7ER5Ie9hBgh4h4ybCNcnd2dmmXQ
8u7u8v7uj4mTC0yh8+BqJ6AwIW8Lnz4a7VU8BptsCohOULO6vLHfITzfR4zCSS8TaimL+MxfKqvwWHiZE6Q
XozFNkShTQ65/tBRYbhgsLyIdU6F1G3opJF8590DxQozALSyux5vBLSo+kXSe2kDZomKLaTM5WHZeqlb8Yh
+YCXfyfMSup3lmBudEjePwy6G2bphh3eg74opGxQmajGicH7 root@uz7ev-evcc-sd-oob-2017-4
Fingerprint: md5 71:f6:c8:81:70:0d:02:6d:77:d3:d1:80:97:4f:da:9e
dropbear.
Starting syslogd/klogd: done
Starting tcf-agent: OK

PetaLinux 2017.4 uz7ev-evcc-sd-oob-2017-4 /dev/ttyPS0

uz7ev-evcc-sd-oob-2017-4 login: █
```

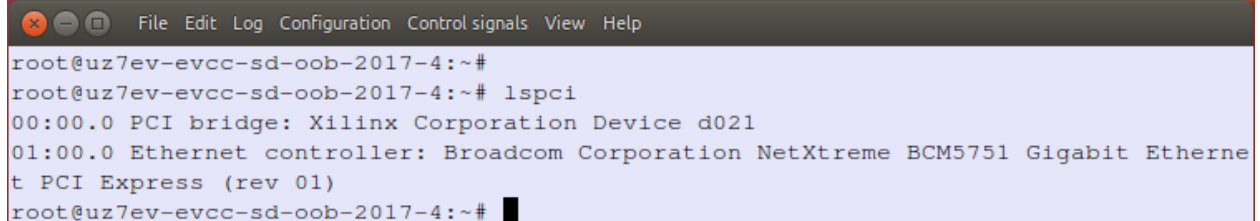
The status bar at the bottom of the window shows the device path as '/dev/ttyUSB3 115200-8-N-1' and several control signal checkboxes: 'DTR', 'RTS', 'CTS', 'CD', 'DSR', and 'RI'.

Experiment 2: Verify that the PCIe NIC is Recognized

Now that the embedded target software has been setup and UltraZed is booted with Linux to a login prompt, the file Read/Write Tests can be launched on the attached SATA drive.

1. Use the terminal window to enter the login **root** along with password **root** in order to gain access
2. Check the PCIe device enumeration to verify that the Ethernet NIC is recognized and that the Ethernet interface is functioning correctly. If the **eth1** device listing does not appear as expected, verify your PCIe Ethernet NIC installation and Ethernet cable connection matches the one shown in [Appendix I: Troubleshooting Ethernet Connection](#).

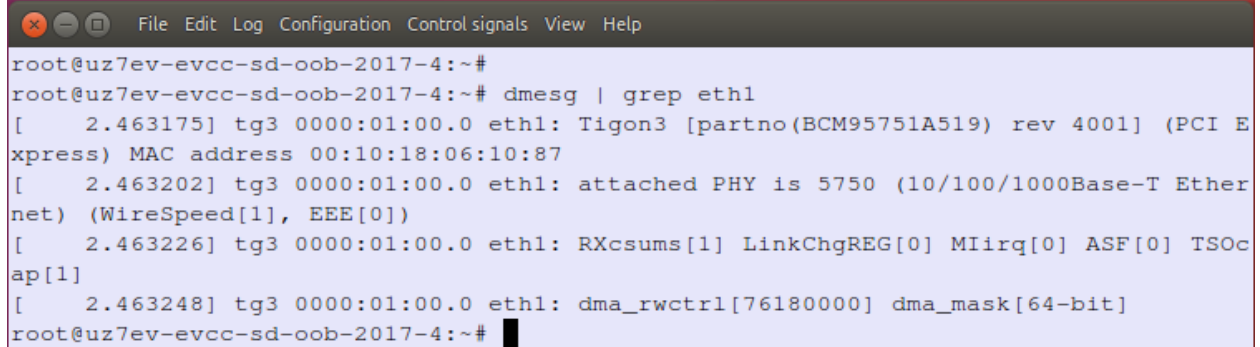
```
# lspci
```



```
root@uz7ev-evcc-sd-oob-2017-4:~#  
root@uz7ev-evcc-sd-oob-2017-4:~# lspci  
00:00.0 PCI bridge: Xilinx Corporation Device d021  
01:00.0 Ethernet controller: Broadcom Corporation NetXtreme BCM5751 Gigabit Ethernet PCI Express (rev 01)  
root@uz7ev-evcc-sd-oob-2017-4:~#
```

3. It is also helpful to examine the Linux kernel boot messages to confirm the PCIe Ethernet NIC interface (eth1) was recognized successfully.

```
# dmesg | grep eth1
```



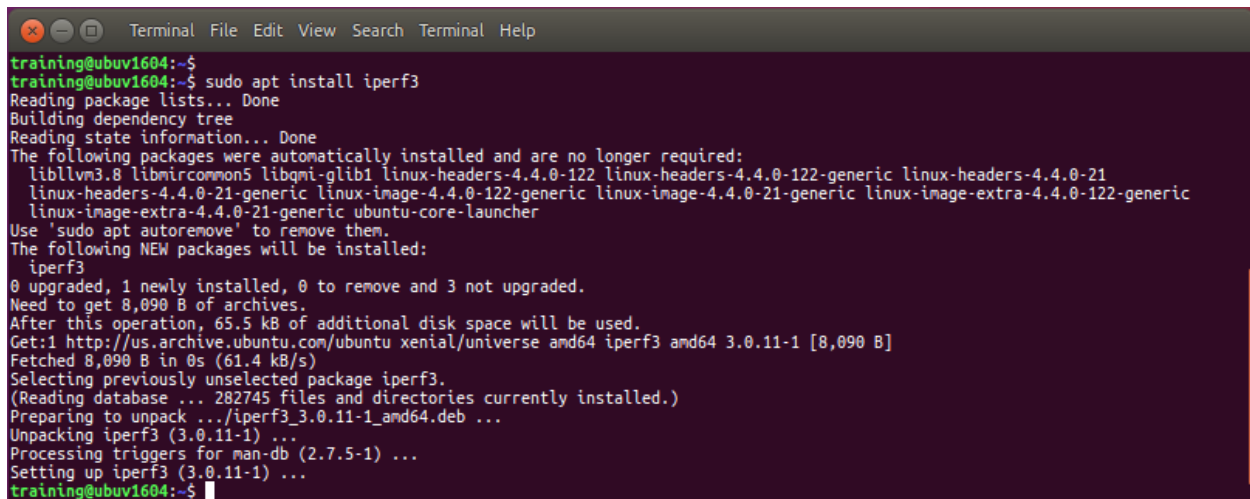
```
root@uz7ev-evcc-sd-oob-2017-4:~#  
root@uz7ev-evcc-sd-oob-2017-4:~# dmesg | grep eth1  
[ 2.463175] tg3 0000:01:00.0 eth1: Tigon3 [partno(BCM95751A519) rev 4001] (PCI Express) MAC address 00:10:18:06:10:87  
[ 2.463202] tg3 0000:01:00.0 eth1: attached PHY is 5750 (10/100/1000Base-T Ethernet) (WireSpeed[1], EEE[0])  
[ 2.463226] tg3 0000:01:00.0 eth1: RXcsums[1] LinkChgREG[0] MIirq[0] ASF[0] TSOcap[1]  
[ 2.463248] tg3 0000:01:00.0 eth1: dma_rwctrl[76180000] dma_mask[64-bit]  
root@uz7ev-evcc-sd-oob-2017-4:~#
```


Experiment 3: Install iperf3 in the Ubuntu Virtual Machine

The Ethernet performance test runs the iperf3 application to test the achievable Ethernet throughput. The iperf3 application works by setting up a server on one end and a client on the other end of the connection. This requires that iperf3 also be installed in the Ubuntu Linux host.

1. Connect the Ubuntu Linux host to the internet.
2. Open a Ubuntu command window and use the Linux apt package manager utility to install iperf3:

```
$ sudo apt install iperf3
```

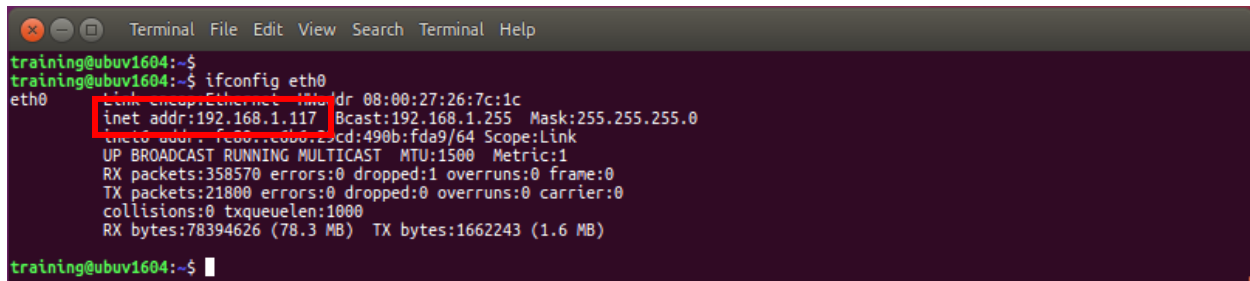
A screenshot of a terminal window titled "Terminal" with a menu bar containing "File", "Edit", "View", "Search", "Terminal", and "Help". The terminal shows the command "training@ubuv1604:~\$ sudo apt install iperf3" and its output. The output indicates that several packages were automatically installed and are no longer required, lists them, and then shows that the new package "iperf3" will be installed. It also shows the disk space requirements and the source of the package. The terminal ends with the prompt "training@ubuv1604:~\$".

```
training@ubuv1604:~$ sudo apt install iperf3
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libllvm3.8 libmircommon5 libqmi-glib1 linux-headers-4.4.0-122 linux-headers-4.4.0-122-generic linux-headers-4.4.0-21
  linux-headers-4.4.0-21-generic linux-image-4.4.0-122-generic linux-image-4.4.0-21-generic linux-image-extra-4.4.0-122-generic
  linux-image-extra-4.4.0-21-generic ubuntu-core-launcher
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  iperf3
0 upgraded, 1 newly installed, 0 to remove and 3 not upgraded.
Need to get 8,090 B of archives.
After this operation, 65.5 kB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu xenial/universe amd64 iperf3 amd64 3.0.11-1 [8,090 B]
Fetched 8,090 B in 0s (61.4 kB/s)
Selecting previously unselected package iperf3.
(Reading database ... 282745 files and directories currently installed.)
Preparing to unpack .../iperf3_3.0.11-1_amd64.deb ...
Unpacking iperf3 (3.0.11-1) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up iperf3 (3.0.11-1) ...
training@ubuv1604:~$
```

Experiment 4: Running iperf3 Ethernet Tests

1. Use the same command window as in the previous step to determine the IP address assigned to the Ubuntu Linux host.

```
$ ifconfig eth0
```

A terminal window titled 'Terminal' with a menu bar (File, Edit, View, Search, Terminal, Help). The prompt is 'training@ubuv1604:~\$'. The command 'ifconfig eth0' has been entered. The output shows details for the 'eth0' interface, including MAC address, IP address (192.168.1.117), broadcast address, and MTU. The IP address '192.168.1.117' is highlighted with a red box.

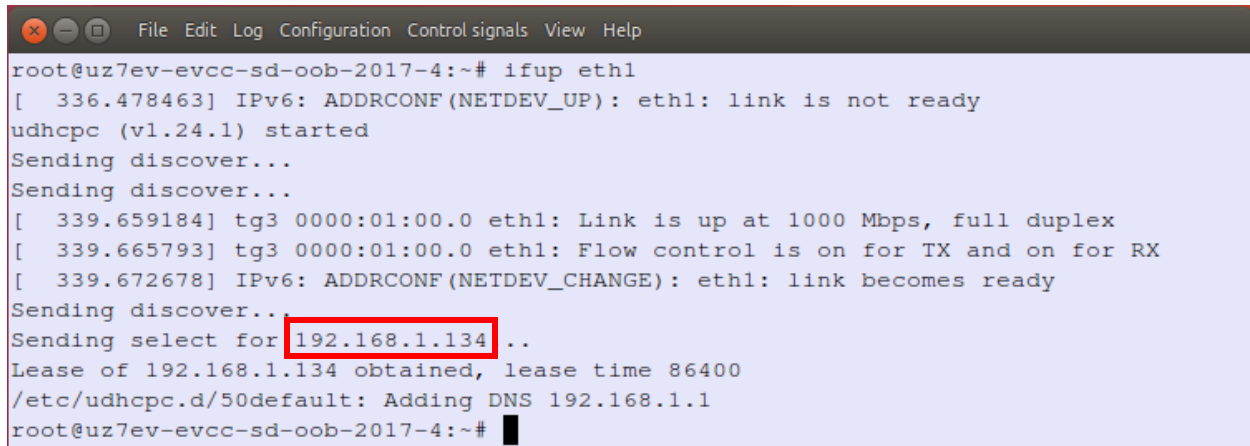
```
training@ubuv1604:~$ ifconfig eth0
eth0: flags=4096<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    link/ether 08:00:27:26:7c:1c brd ff:ff:ff:ff:ff:ff
    inet addr:192.168.1.117 Bcast:192.168.1.255 Mask:255.255.255.0
    inet6 addr: fe80::e0b0:2cd:490b:fda9/64 Scope:Link
    UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
    RX packets:358570 errors:0 dropped:1 overruns:0 frame:0
    TX packets:21800 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:1000
    RX bytes:78394626 (78.3 MB) TX bytes:1662243 (1.6 MB)

training@ubuv1604:~$
```

In this case the IP address of the Ubuntu host is set to 192.168.1.117. This may be different for your network.

2. Use the GtkTerm serial terminal to enable the eth1 Ethernet interface on the PCIe NIC. You may first see a message that the “link is not ready”. You may safely ignore this message.

```
# ifup eth1
```

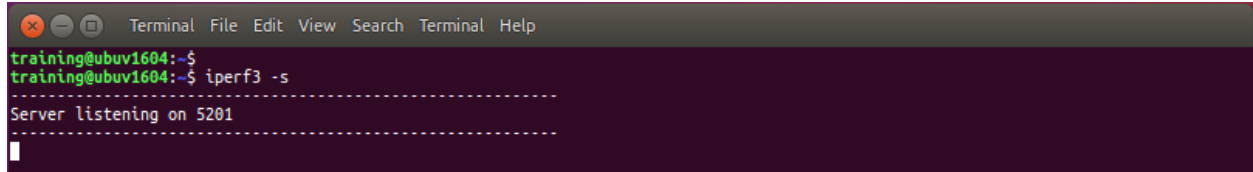
A terminal window titled 'File Edit Log Configuration Control signals View Help'. The prompt is 'root@uz7ev-evcc-sd-oob-2017-4:~#'. The command 'ifup eth1' has been entered. The output shows the process of bringing up the 'eth1' interface, including sending discover messages and obtaining an IP address (192.168.1.134). The IP address '192.168.1.134' is highlighted with a red box.

```
root@uz7ev-evcc-sd-oob-2017-4:~# ifup eth1
[ 336.478463] IPv6: ADDRCONF(NETDEV_UP): eth1: link is not ready
udhcpc (v1.24.1) started
Sending discover...
Sending discover...
[ 339.659184] tg3 0000:01:00.0 eth1: Link is up at 1000 Mbps, full duplex
[ 339.665793] tg3 0000:01:00.0 eth1: Flow control is on for TX and on for RX
[ 339.672678] IPv6: ADDRCONF(NETDEV_CHANGE): eth1: link becomes ready
Sending discover...
Sending select for 192.168.1.134 ..
Lease of 192.168.1.134 obtained, lease time 86400
/etc/udhcpc.d/50default: Adding DNS 192.168.1.1
root@uz7ev-evcc-sd-oob-2017-4:~#
```

In this case the IP address of the UltraZed-EV is set to 192.168.1.134. This may be different for your network.

3. Use the same command window as in the previous step and start iperf3 in server mode on the Ubuntu Linux host.

```
$ iperf3 -s
```

A terminal window with a dark background and light text. The title bar shows 'Terminal' and standard window controls. The prompt is 'training@ubuv1604:~\$'. The command 'iperf3 -s' has been entered, and the output is 'Server listening on 5201'.

```
training@ubuv1604:~$ iperf3 -s
Server listening on 5201
```

4. At the UltraZed serial terminal launch the included **network-test.sh** script to perform Ethernet performance tests using the Linux iperf3 command as a client on the eth1 interface using the following command. Press enter to continue when prompted at the splash screen.

```
# ./network-test.sh -i eth1 -l 192.168.1.134 -m c -r 192.168.1.117
```

A serial terminal window with a light blue background and black text. The title bar shows 'File Edit Log Configuration Control signals View Help'. The script displays a star border, instructions, and a prompt to press enter to continue. The status bar at the bottom shows '/dev/ttyUSB3 115200-8-N-1' and hardware control signals.

```
*****
***                                     ***
***      ****  **      **  ****      **  ****      ****      ****      ***
***      **  **  **      **  **  **  **  **      **      **      ***
***      **      **  **  **      **  **  **  ****      **      ***
***      **      **  ****      **  **  **  **      **      **      ***
***      **      **  **      **      ****      ****      ****      **      ***
***      **      **  **      **      ****      ****      ****      **      ***
***      ....  **  **      **      ****      ****      ****      **      ***
***      .....                                     ***
***                                     ***
*** This is a simple script to run the iperf3 test application ***
*** to determine the maximum achievable Ethernet ***
*** network bandwidth. ***
***                                     ***
*** More information about iperf3 can be found at ***
*** http://software.es.net/iperf/ ***
***                                     ***
*** Make sure the Ethernet cable is connected to only the ***
*** selected interface before continuing!!! ***
*** eth0 == Carrier ***
*** eth1 == PCIe NIC ***
***                                     ***
*** If running test in CLIENT mode make sure the iperf3 server ***
*** is started on a network host before continuing!!! ***
*** $ iperf3 -s ***
***                                     ***
*****

Press enter to continue...
```

- The network test script will run for 30 seconds and report the observed data throughput every 2 seconds. Upon completion of the iperf3 test, a throughput summary will be shown which shows the total amount of data transferred along with the measured throughput rate over the course of the test.

```

Start the eth1 interface...
Set the static IP address for the device 192.168.1.134...
Start the iperf3 test as a client connected to the server at IP address 192.168.1.117...
Connecting to host 192.168.1.117, port 5201
[ 4] local 192.168.1.134 port 57438 connected to 192.168.1.117 port 5201
[ ID] Interval           Transfer     Bandwidth   Retr   Cwnd
[ 4]  0.00-2.00      sec    208 MBytes  871 Mbits/sec    0    779 KBytes
[ 4]  2.00-4.00      sec    206 MBytes  863 Mbits/sec    0    865 KBytes
[ 4]  4.00-6.00      sec    209 MBytes  878 Mbits/sec    6    700 KBytes
[ 4]  6.00-8.00      sec    191 MBytes  799 Mbits/sec   20    580 KBytes
[ 4]  8.00-10.00     sec    204 MBytes  856 Mbits/sec   87    395 KBytes
[ 4] 10.00-12.00     sec    206 MBytes  865 Mbits/sec    0    484 KBytes
[ 4] 12.00-14.00     sec    208 MBytes  874 Mbits/sec    0    532 KBytes
[ 4] 14.00-16.00     sec    201 MBytes  844 Mbits/sec   35    410 KBytes
[ 4] 16.00-18.00     sec    205 MBytes  859 Mbits/sec  136    365 KBytes
[ 4] 18.00-20.00     sec    206 MBytes  865 Mbits/sec    0    450 KBytes
[ 4] 20.00-22.00     sec    201 MBytes  841 Mbits/sec   90    434 KBytes
[ 4] 22.00-24.00     sec    208 MBytes  874 Mbits/sec   56    406 KBytes
[ 4] 24.00-26.00     sec    205 MBytes  858 Mbits/sec    0    474 KBytes
[ 4] 26.00-28.00     sec    201 MBytes  842 Mbits/sec    0    519 KBytes
[ 4] 28.00-30.00     sec    206 MBytes  866 Mbits/sec   35    337 KBytes
-----
[ ID] Interval           Transfer     Bandwidth   Retr
[ 4]  0.00-30.00     sec  2.99 GBytes  857 Mbits/sec  465
[ 4]  0.00-30.00     sec  2.99 GBytes  856 Mbits/sec
                                     sender
                                     receiver

iperf Done.
root@uz7ev-evcc-sd-oob-2017-4:~#
/dev/ttyUSB3 115200-8-N-1
DTR  RTS  CTS  CD  DSR  RI

```

- Go to the Ubuntu host command window and stop the iperf3 server with a <ctrl>-c.

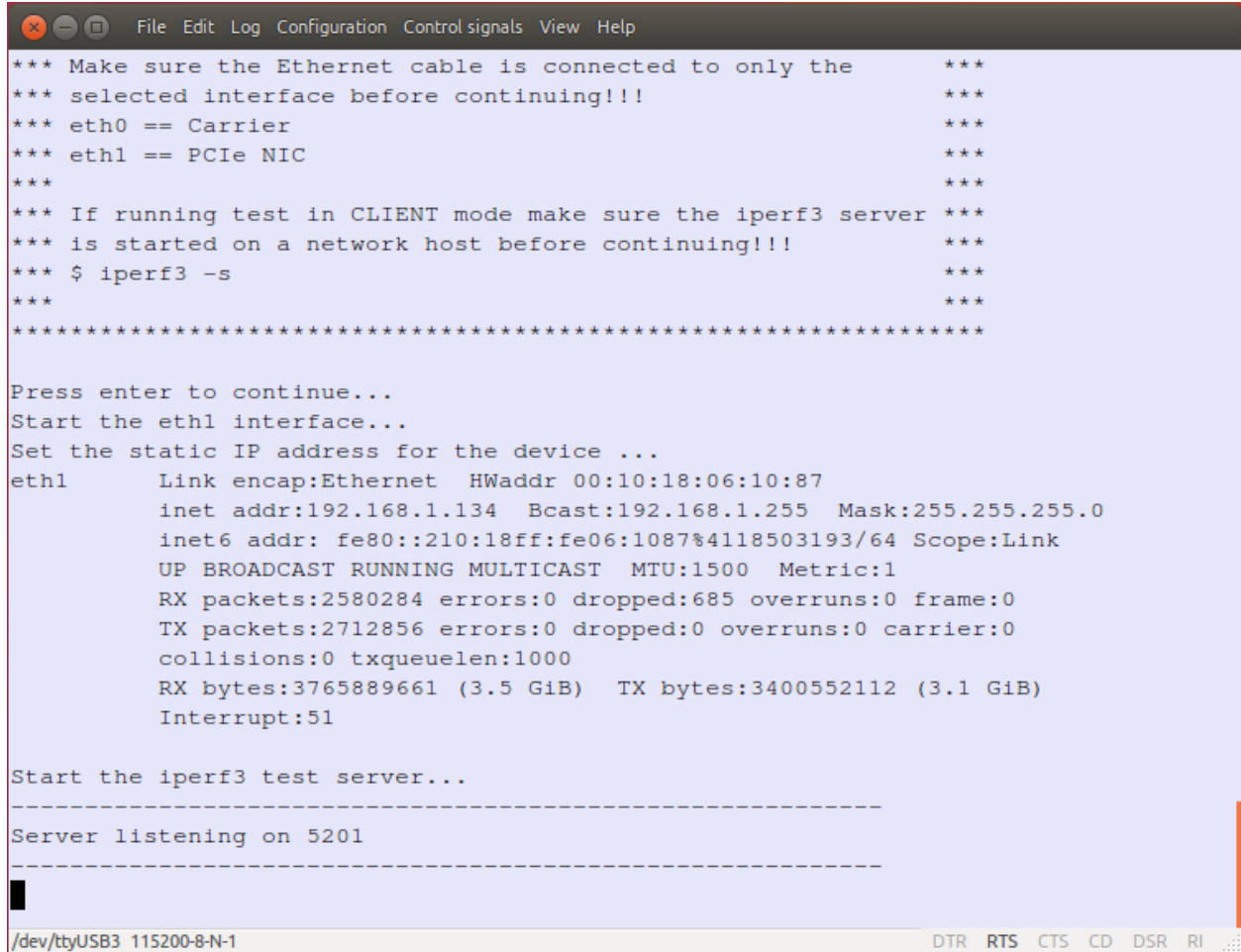
```

Terminal File Edit View Search Terminal Help
[ 5] 15.00-16.00     sec    104 MBytes  871 Mbits/sec
[ 5] 16.00-17.00     sec    98.6 MBytes  827 Mbits/sec
[ 5] 17.00-18.00     sec    104 MBytes  872 Mbits/sec
[ 5] 18.00-19.00     sec    104 MBytes  875 Mbits/sec
[ 5] 19.00-20.00     sec    105 MBytes  879 Mbits/sec
[ 5] 20.00-21.00     sec    104 MBytes  874 Mbits/sec
[ 5] 21.00-22.00     sec    105 MBytes  882 Mbits/sec
[ 5] 22.00-23.00     sec    105 MBytes  880 Mbits/sec
[ 5] 23.00-24.00     sec    107 MBytes  900 Mbits/sec
[ 5] 24.00-25.00     sec    104 MBytes  874 Mbits/sec
[ 5] 25.00-26.00     sec    105 MBytes  880 Mbits/sec
[ 5] 26.00-27.00     sec    97.5 MBytes  818 Mbits/sec
[ 5] 27.00-28.00     sec    104 MBytes  869 Mbits/sec
[ 5] 28.00-29.00     sec    103 MBytes  861 Mbits/sec
[ 5] 29.00-29.38     sec    39.0 MBytes  855 Mbits/sec
-----
[ ID] Interval           Transfer     Bandwidth   Retr
[ 5]  0.00-29.38     sec  2.96 GBytes  865 Mbits/sec  509
[ 5]  0.00-29.38     sec  2.96 GBytes  864 Mbits/sec
                                     sender
                                     receiver
Server listening on 5201
^Ciperf3: interrupt - the server has terminated
root@uz7ev-evcc-sd-oob-2017-4:~#

```

7. Now experiment with running the iperf3 tests as a server on the UltraZed-EV using the following command. Press enter when prompted at the splash screen to start iperf3 in server mode.

```
# ./network-test.sh -i eth1 -m s
```



```
*** Make sure the Ethernet cable is connected to only the ***
*** selected interface before continuing!!! ***
*** eth0 == Carrier ***
*** eth1 == PCIe NIC ***
***
*** If running test in CLIENT mode make sure the iperf3 server ***
*** is started on a network host before continuing!!! ***
*** $ iperf3 -s ***
***
*****

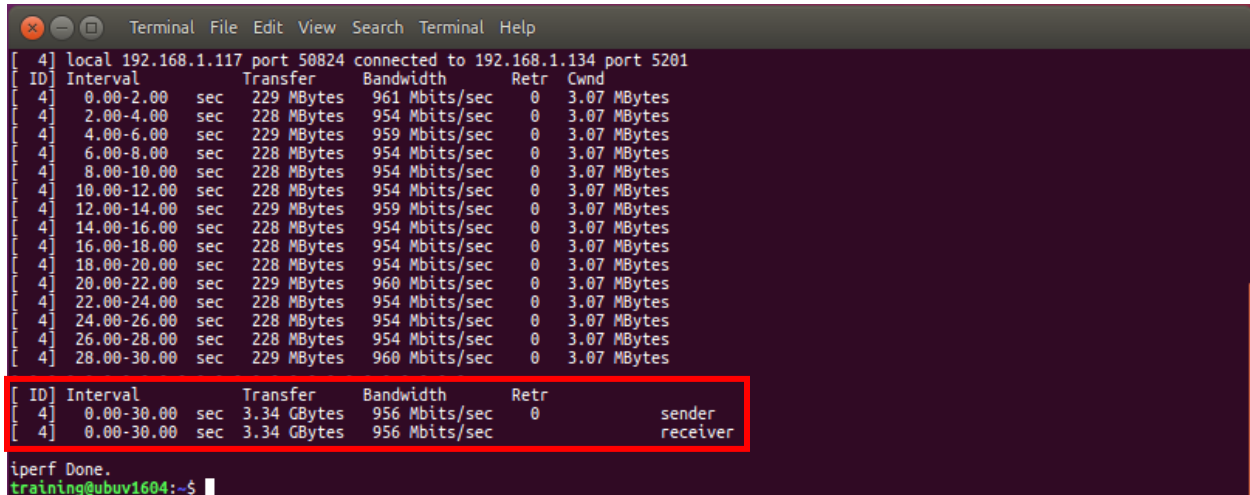
Press enter to continue...
Start the eth1 interface...
Set the static IP address for the device ...
eth1      Link encap:Ethernet  HWaddr 00:10:18:06:10:87
          inet addr:192.168.1.134  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::210:18ff:fe06:1087%4118503193/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2580284 errors:0 dropped:685 overruns:0 frame:0
          TX packets:2712856 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3765889661 (3.5 GiB)  TX bytes:3400552112 (3.1 GiB)
          Interrupt:51

Start the iperf3 test server...
-----
Server listening on 5201
-----
█

/dev/ttyUSB3 115200-8-N-1 DTR RTS CTS CD DSR RI
```

- Go back to the Ubuntu host command window and start iperf3 as a client to connect to the UltraZed-EV iperf3 server. Recall that the IP address of the UltraZed-EV is set to 192.168.1.134. This may be different on your network. The command below will run the iperf3 application in client mode and will run for 30 seconds and report the throughput every 2 seconds.

```
$ iperf3 -c 192.168.1.134 -t 30 -i 2
```



```
4] local 192.168.1.117 port 50824 connected to 192.168.1.134 port 5201
ID] Interval      Transfer      Bandwidth    Retr    Cwnd
4]  0.00-2.00    sec    229 MBytes    961 Mbits/sec    0    3.07 MBytes
4]  2.00-4.00    sec    228 MBytes    954 Mbits/sec    0    3.07 MBytes
4]  4.00-6.00    sec    229 MBytes    959 Mbits/sec    0    3.07 MBytes
4]  6.00-8.00    sec    228 MBytes    954 Mbits/sec    0    3.07 MBytes
4]  8.00-10.00   sec    228 MBytes    954 Mbits/sec    0    3.07 MBytes
4] 10.00-12.00   sec    228 MBytes    954 Mbits/sec    0    3.07 MBytes
4] 12.00-14.00   sec    229 MBytes    959 Mbits/sec    0    3.07 MBytes
4] 14.00-16.00   sec    228 MBytes    954 Mbits/sec    0    3.07 MBytes
4] 16.00-18.00   sec    228 MBytes    954 Mbits/sec    0    3.07 MBytes
4] 18.00-20.00   sec    228 MBytes    954 Mbits/sec    0    3.07 MBytes
4] 20.00-22.00   sec    229 MBytes    960 Mbits/sec    0    3.07 MBytes
4] 22.00-24.00   sec    228 MBytes    954 Mbits/sec    0    3.07 MBytes
4] 24.00-26.00   sec    228 MBytes    954 Mbits/sec    0    3.07 MBytes
4] 26.00-28.00   sec    228 MBytes    954 Mbits/sec    0    3.07 MBytes
4] 28.00-30.00   sec    229 MBytes    960 Mbits/sec    0    3.07 MBytes

ID] Interval      Transfer      Bandwidth    Retr
4]  0.00-30.00   sec    3.34 GBytes    956 Mbits/sec    0    sender
4]  0.00-30.00   sec    3.34 GBytes    956 Mbits/sec    0    receiver

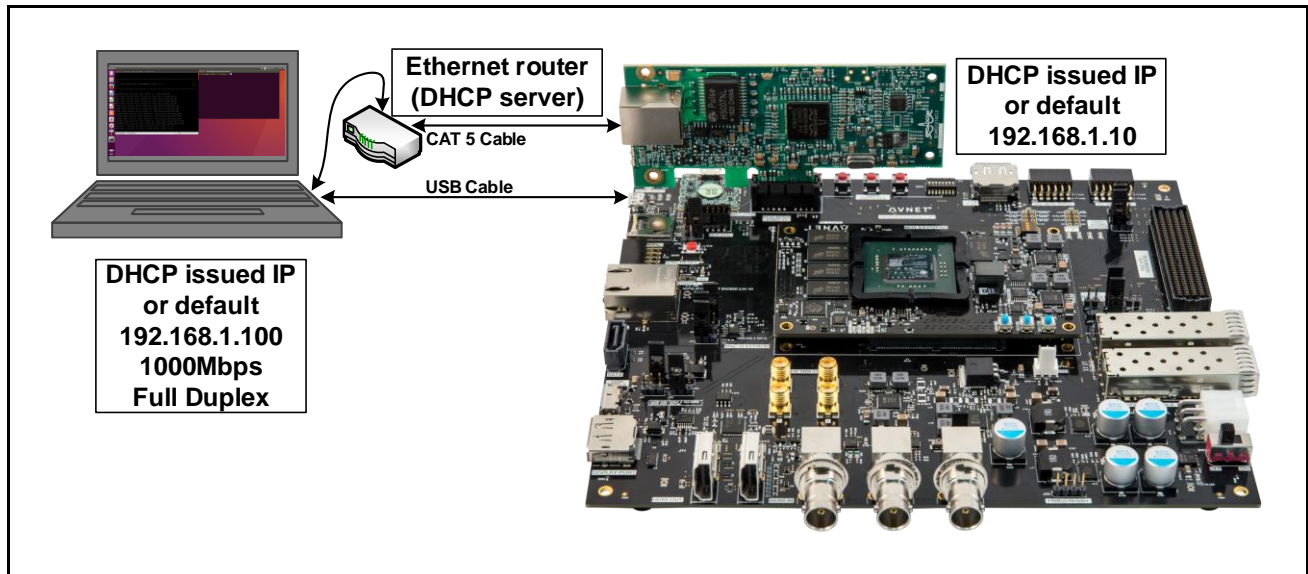
iperf Done.
training@ubuv1604:~$
```

- You may wish to experiment with the iperf3 command to specify different TCP/IP packet sizes, etc. to examine their effect on the Ethernet throughput to aid in tuning the performance of the Ethernet interface.

Appendix I: Troubleshooting Ethernet Connection

This section provides troubleshooting information for the Ethernet connection used in this UltraZed Open Source Linux PCIe Performance Test Tutorial.

1. The basic configuration for the UltraZed Open Source Linux PCIe Performance Test Tutorial is shown below:



2. Verify that the Broadcom BCM5751 PCIe Ethernet NIC add-in card is installed in the UltraZed-EV Carrier Card correctly.
3. Verify that the Ethernet cable is connected to an Ethernet router/switch or directly to a PC. If connected directly to a PC the IP address will need to be statically assigned and the link speed will need to be set to 1000Mbps full duplex.
4. Verify that the USB UART cable is connected between the PC and the UltraZed-EV Carrier Card. See [Appendix II: Troubleshooting Serial Connection](#) for more details.

Appendix II: Troubleshooting Serial Connection

This section provides troubleshooting information for the USB-UART serial connection used in this UltraZed Open Source Linux SATA Performance Test Tutorial. The experiments in this tutorial use Ubuntu **Serial port terminal** (gtkterm) as the serial terminal application is recommended for this tutorial but other serial terminal applications might work as well.

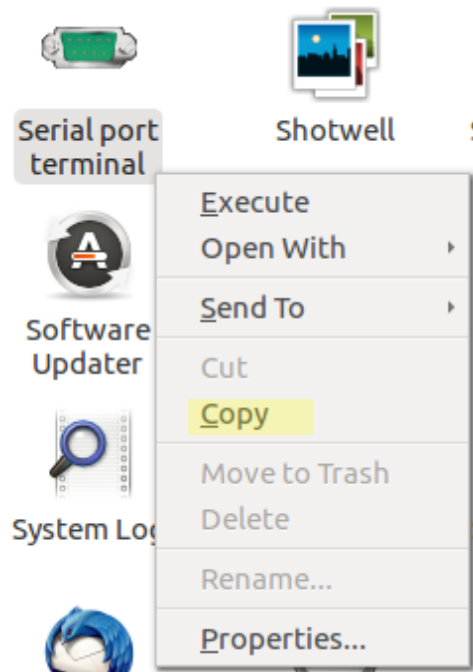
1. To make it easier to launch the terminal app (GtkTerm) without needing to provide the root password each time, open a command window and add the **current username** to the group for the **/dev/ttyUSBx** devices used for USB-UART:

```
$ sudo usermod -a -G dialout <current username>
```

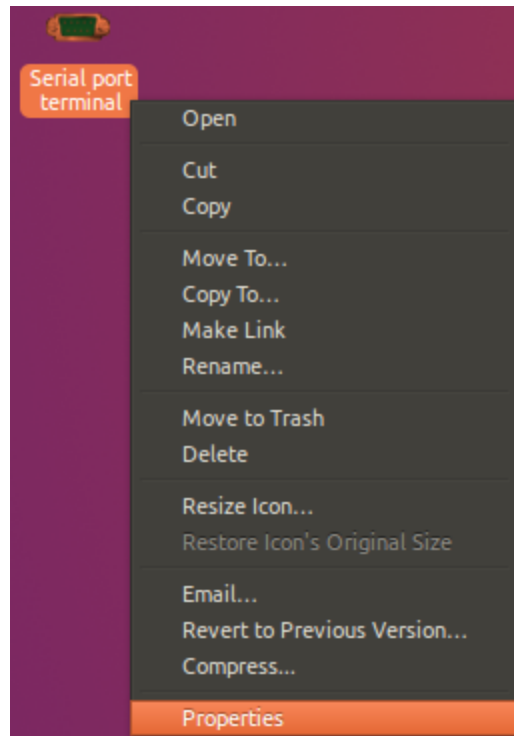
2. Install the **gtkterm** package:

```
$ sudo apt-get install gtkterm
```

3. **Reboot the Virtual Machine to force the changes to take effect.**
4. Create a Desktop icon by copying and pasting **Serial port terminal** (gtkterm) application from the **/usr/share/applications** folder directly to the **~/Desktop** folder:



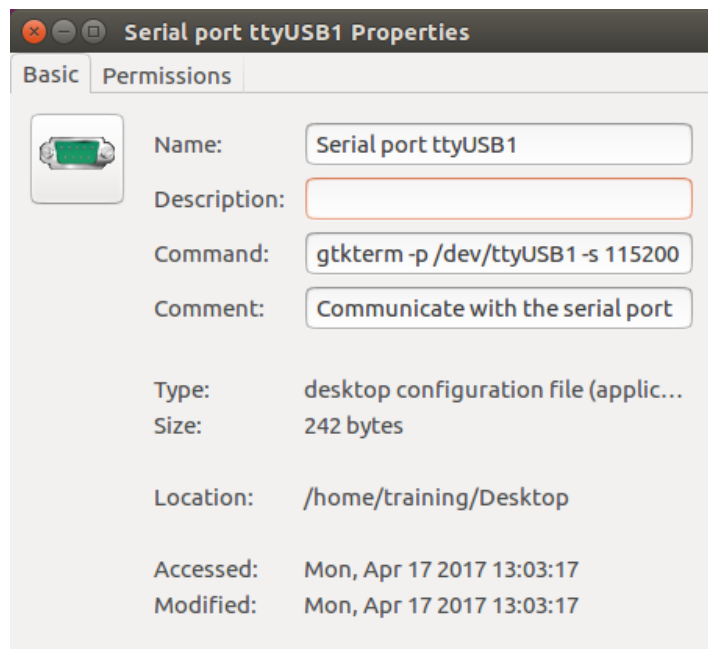
- Right-click on the new **Serial port terminal** (gtkterm) application Desktop icon and select the **Properties** option.



- Within the Properties window, set the attributes to match the USB-UART device attached to the system. In this example the USB-UART is attached to the **/dev/ttyUSB1** device entry. Close the properties window when finished.

Name: **Serial port ttyUSB1**

Command: **gtkterm -p /dev/ttyUSB1 -s 115200**



7. Insert a bootable microSD card prepared with the prebuilt binaries available with this tutorial into the UltraZed Carrier Card J4 slot.
8. Setup the UltraZed hardware as described in [Experiment 0: Setting Up the UltraZed SOM with Carrier Card](#).
9. Set the UltraZed SOM Boot Mode switch (SW2) (MODE[3:0] = SW2[4:1]) to ON, OFF, ON, and OFF positions (Boot Mode set to SD Card, MODE[3:0] = 0xA).
10. Make sure the UltraZed Carrier Card power switch SW7 is in the OFF position.
11. Insert the UltraZed SOM module onto the UltraZed Carrier Card using the JX1, JX2, and JX3 connectors.
12. Close or disconnect the terminal that may have previously been open on your PC.
13. Plug in the UltraZed USB-UART cable between the host PC and the UltraZed Carrier Card USB-UART port (J21).
14. Insert the appropriate country plug into the 12V AC/DC adapter. Plug it into the J7 2x3 power connector. (NOTE – this 2x3 connector is NOT compatible with ATX power supplies.)
15. Set the UltraZed Carrier power switch SW7 to the ON position. The UltraZed system will power on and the Power Good LED (D2) should illuminate.
16. Check the kernel output log for signs that the USB-UART device has enumerated and note the ttyUSB device that is enumerated. USB-UART device should enumerate as **/dev/ttyUSB0** or similar.

```
$ dmesg |grep ttyUSB
```

17. Create system default udev rules to give USB-UART devices sufficient permissions for all users similar.

```
$ sudo cp /lib/udev/rules.d/50-udev-default.rules /etc/udev/rules.d/.
```

18. Edit the system default udev rules with **vi** text editor.

```
$ sudo vi /etc/udev/rules.d/50-udev-default.rules
```

19. Add the following 2 lines to **50-udev-default.rules** somewhere just after the 'tty' section.

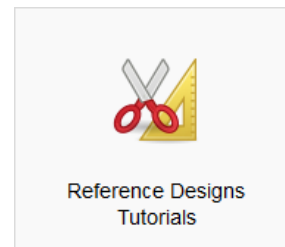
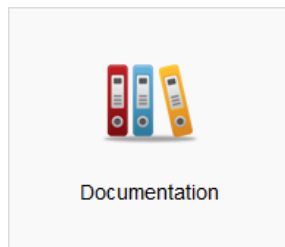
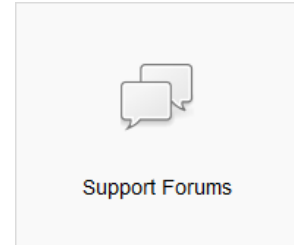
```
# relax the permissions just for ttyUSB0
KERNEL=="ttyUSB0", MODE="0666"
```

20. Save the changes to the **50-udev-default.rules** file and exit **vi** text editor.
21. Run **gtkterm** again in normal user mode and check for terminal output.

Appendix III: Getting Support

Avnet Support

- Technical support is offered online through the ultrazed.org website support forums. UltraZed users are encouraged to participate in the forums and offer help to others when possible.
<http://ultrazed.org/forums/zed-english-forum>
<http://ultrazed.org/forums/software-application-development>
- For questions regarding the UltraZed community website, please direct questions to the ultrazed.org Web Master (webmaster@ultrazed.org).
- To access the most current collateral for the UltraZed, visit the community support page (www.ultrazed.org/content/support) and click one of the icons shown below:



- UltraZed-EV SOM Documentation
<http://zedboard.org/support/documentation/21811>
- UltraZed-EV Carrier Card
 - Documentation
TBD
 - Reference Designs
TBD
- Instructions for how to setup the Ubuntu virtual machine if using a Windows host PC
[http://zedboard.org/sites/default/files/design/VirtualBox Installation Guide 2017 2.zip](http://zedboard.org/sites/default/files/design/VirtualBox%20Installation%20Guide%202017%202.zip)

Internet Support

Here are some helpful links regarding some of the Linux software applications mentioned in this tutorial:

iperf3

- Benchmark Ethernet throughput with iperf3
<https://software.es.net/iperf/>
- iperf3 FAQ
<https://software.es.net/iperf/faq.html>

Xilinx Support

For questions regarding products within the Product Entitlement Account, send an email message to the Customer Service Representative in your region:

- Canada, USA and South America - isscs_cases@xilinx.com
- Europe, Middle East, and Africa - eucases@xilinx.com
- Asia Pacific including Japan - apaccase@xilinx.com

For technical support, including the installation and use of the product license file, contact Xilinx Online Technical Support at www.xilinx.com/support. The following assistance resources are also available on the website:

- Software, IP and documentation updates
- Access to technical support Web tools
- Searchable answer database with over 4,000 solutions
- User forums

Revision History

Date	Version	Revision
13 Jun 2018	01	Initial Release