

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 sunset sky with orange and yellow clouds, and distant mountains. The foreground is filled with tall, dry grass.

UltraZed SATA Performance Test Tutorial

© 2017 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

The necessity for high density, low-latency, mass storage is increasing for embedded applications. As data storage demands increase, the ability to objectively measure storage read/write performance when using an embedded operating system becomes essential.

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 SATA storage read/write performance and throughput in addition to transmission overheads, including latency, read/write block size, and system caching such that the calculated throughput typically does not reflect the maximum achievable throughput. As a result, the throughput between the host system and the SATA storage device 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 SATA drive read/write performance tests on Zynq 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:

dd - This utility is from the Coreutils Linux package and is a very simple tool which can be used to perform read/write throughput measurements to target storage devices

hdparm – This tool is often used to measure the read performance of Flash and disk drives. It can also be used to change drive settings and even erase SSDs securely.

bonnie++ - This is a small utility for the purpose of performing extensive benchmarking of file system I/O performance.

For the example SATA test configuration that was used in this tutorial, see the section [Troubleshooting: SATA 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 (WebPACK edition)
- Git SCM toolset (version used for this tutorial is v2.7.4)

Hardware

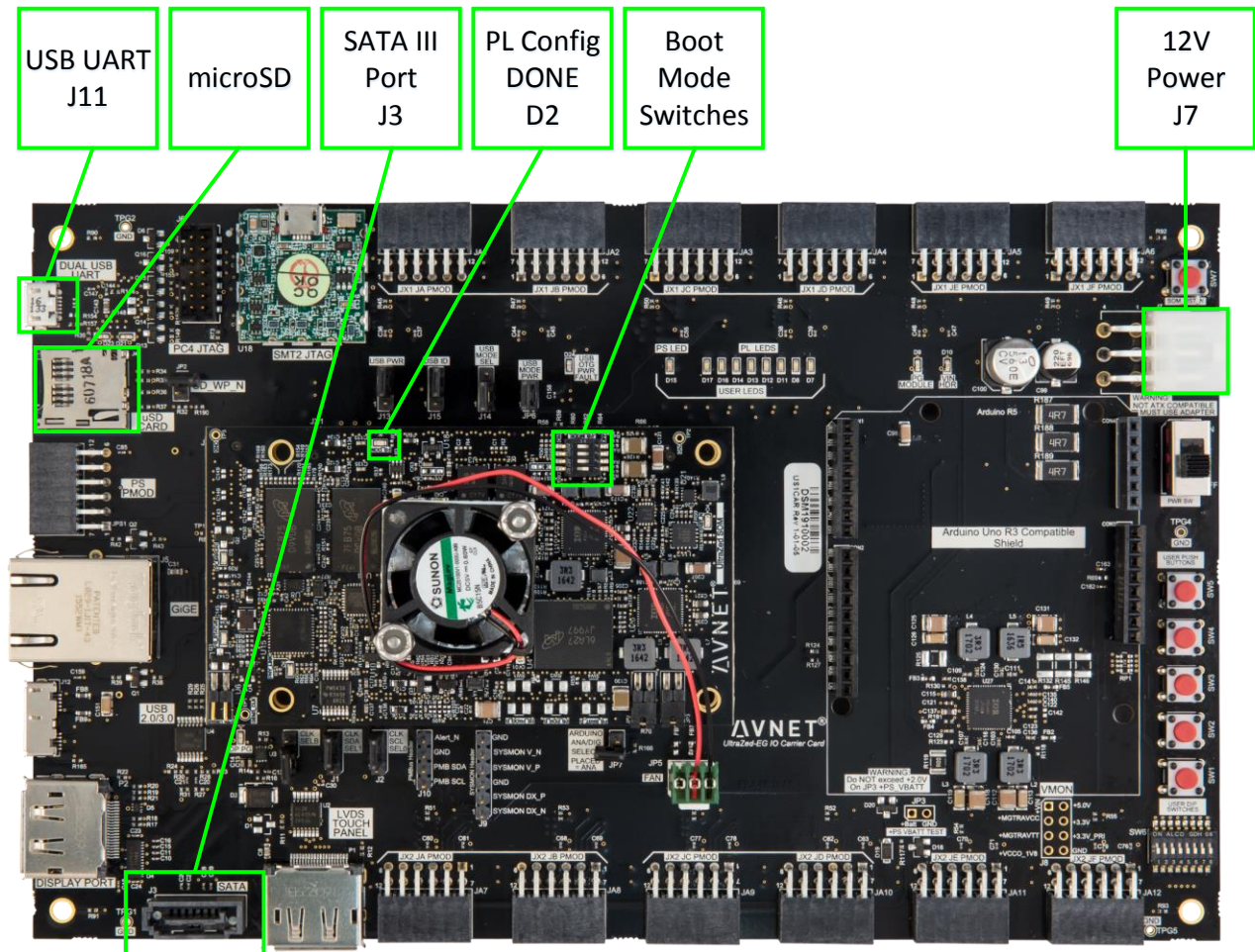
The hardware setup used to test this reference design includes:

- Your choice of UltraZed SOM and carrier card
 - UltraZed-EG SOM (AES-ZU3EG-1-SOM-I-G) + I/O Carrier Card (AES-ZU-IOCC-G)
 - UltraZed-EG SOM (AES-ZU3EG-1-SOM-I-G) + PCIe Carrier Card (AES-ZU-PCIECC-G)
 - UltraZed-EV SOM (AES-ZU7EV-1-SOM-I-G) + 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
- Delkin Utility+ 64GB SATA III SSD
- SATA III data and power combo cable
- 12V/5V 4-pin Molex 2A power adapter

Experiment 0: Setting Up the UltraZed SOM with Carrier Card

UltraZed-EG with I/O Carrier Card

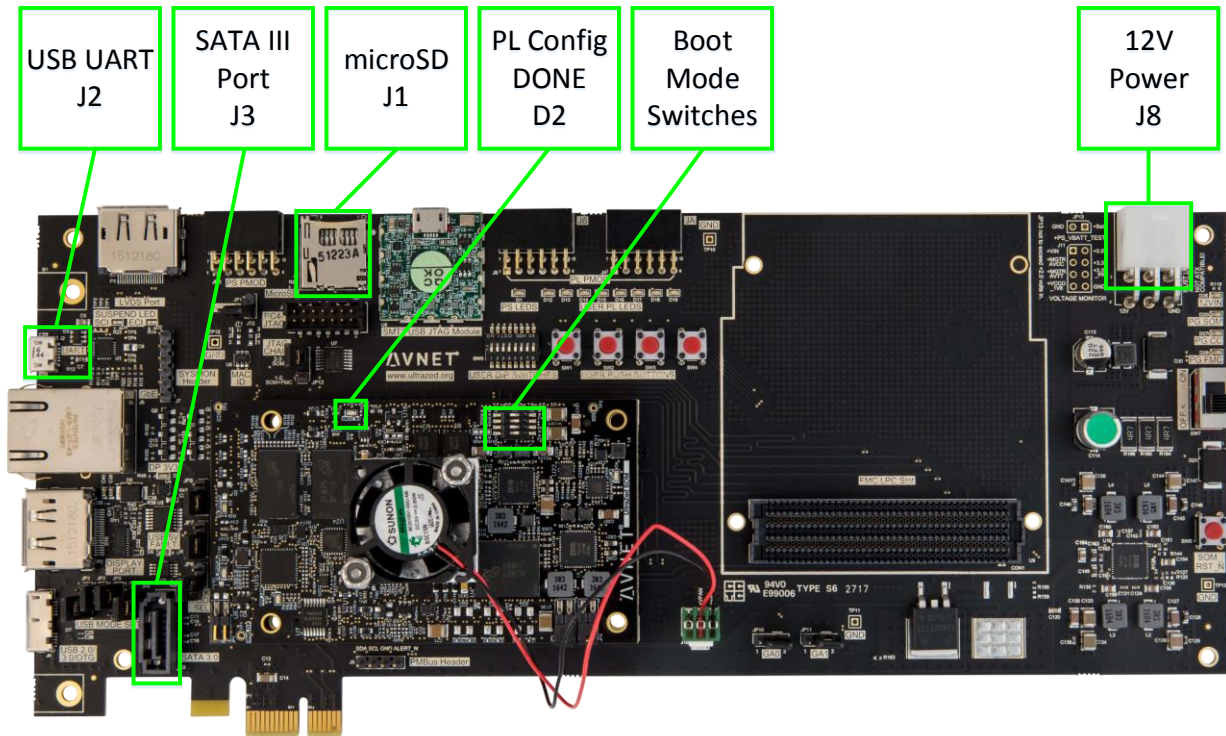
Refer to the following figure and perform the following steps when using the UltraZed-EG and I/O Carrier.



1. Plug the UltraZed-EG SOM onto the I/O Carrier Card via JX1/JX2/JX3 connectors and connect the fan to the fan header (JP5) on the I/O Carrier Card.
2. Set the UltraZed-EG SOM Boot Mode switch (SW2) (MODE[3:0] = SW2[4:1]) to ON, ON, ON, and ON positions (Boot Mode set to SD card, MODE[3:0] = 0x0).
3. Connect the USB-UART port on the I/O Carrier Card (J11) to a free USB port on your PC.
4. Connect the SATA III SSD as described in [Appendix I: Troubleshooting SATA Connection](#).
5. Connect the 12V power cable, but do not turn on the board yet.

UltraZed-EG with PCIe Carrier Card

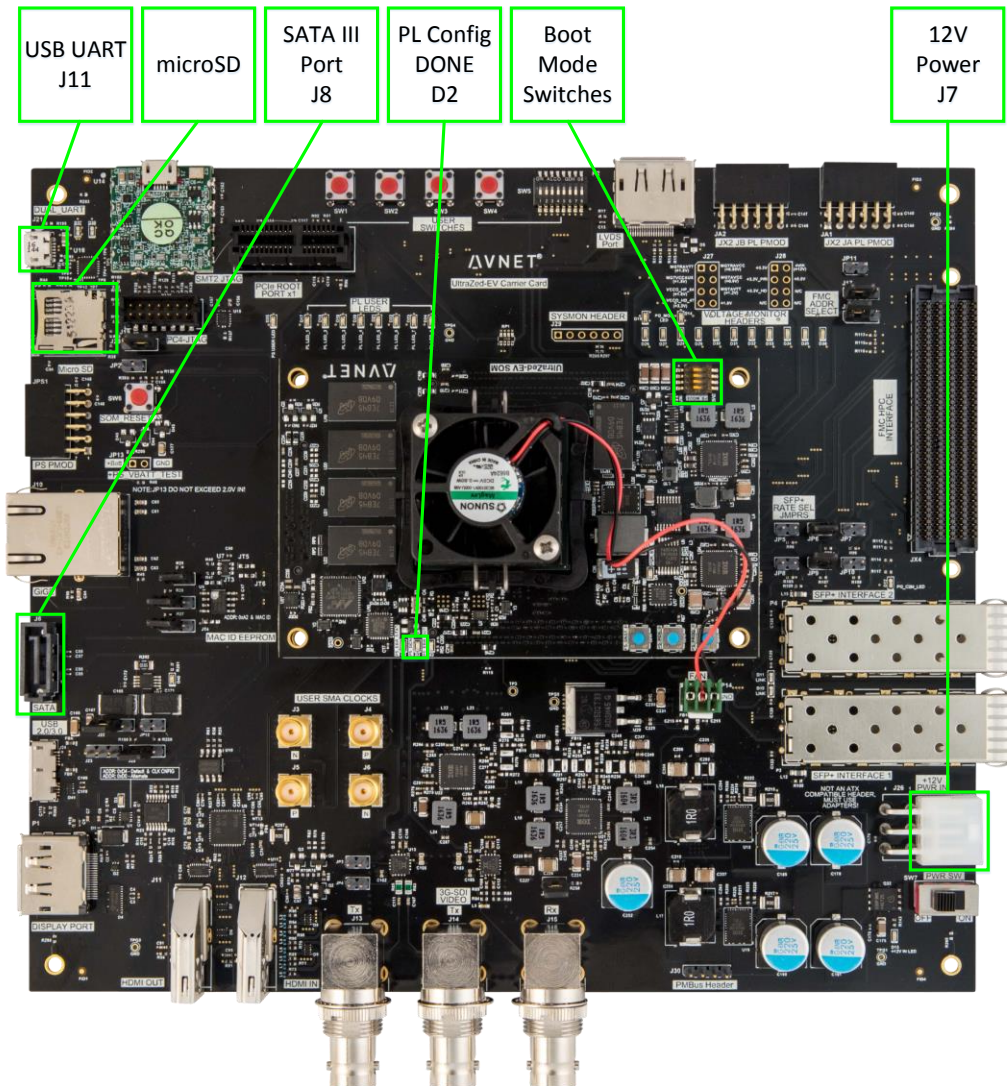
Refer to the following figure and perform the following steps to set up the board when using the UltraZed-EG and PCIe Carrier.



1. Plug the UltraZed-EG SOM onto the PCIe Carrier Card via JX1/JX2/JX3 connectors and connect the fan to the fan header (J10) on the PCIe Carrier Card.
2. Set the UltraZed-EG 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] = 0x5).
3. Connect the USB-UART port on the PCIe Carrier Card (J2) to a free USB port on your PC.
4. Connect the SATA III SSD as described in [Appendix I: Troubleshooting SATA Connection](#).
5. Connect the 12V power cable, but do not turn on the board yet.

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.

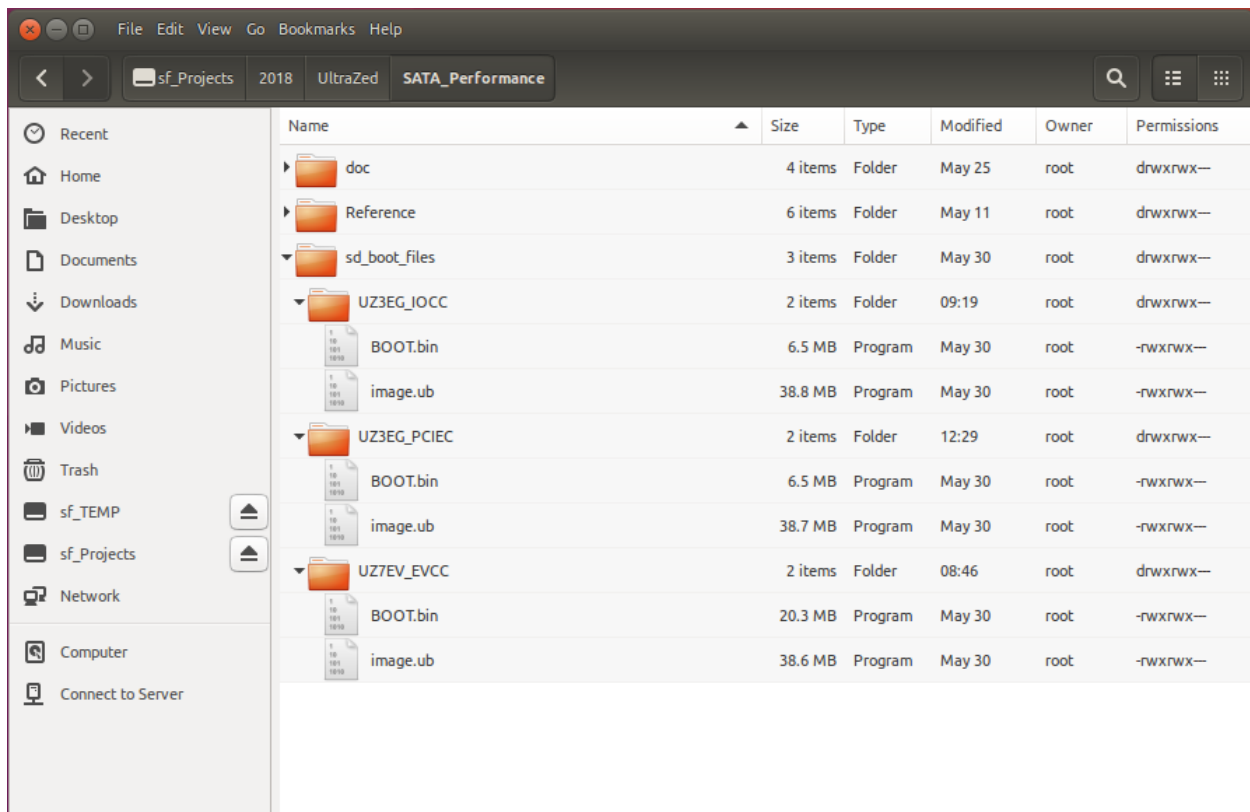


1. Plug the UltraZed-EV SOM onto the EV Carrier Card via JX1/JX2/JX3 connectors and connect the fan to the fan header (JP14) on the EV Carrier Card.
2. Set the UltraZed-EV SOM Boot Mode switch (SW2) (MODE[3:0] = SW2[1:4]) to OFF, ON, OFF, and ON positions (Boot Mode set to SD card, MODE[3:0] = 0x5).
3. Connect the USB-UART port on the EV Carrier Card (J21) to a free USB port on your PC.
4. Connect the SATA III SSD as described in [Appendix I: Troubleshooting SATA Connection](#).
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 this tutorial.

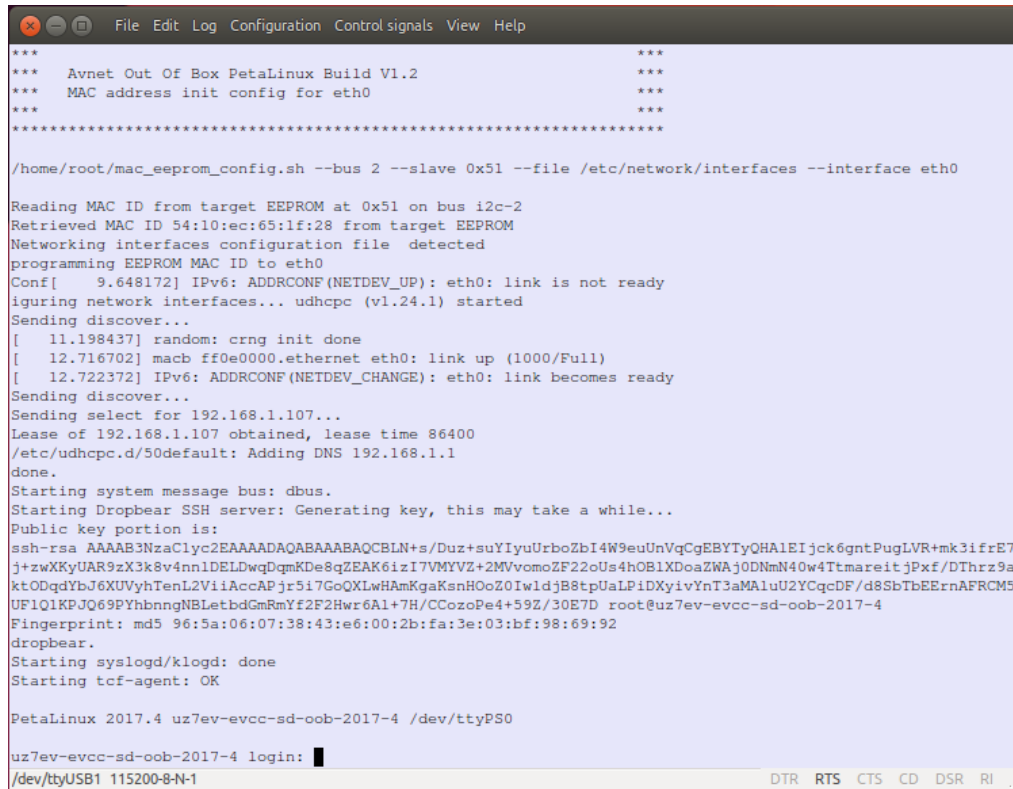
1. Copy the **BOOT.BIN** and **image.ub** files from the accompanying archive into the root folder of the microSD card to match the SOM and Carrier you are using.



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 screenshot shows a gtkterm window with a menu bar (File, Edit, Log, Configuration, Control signals, View, Help) and a terminal output area. The output shows the execution of a script to configure the network interface eth0. The script reads the MAC ID from the EEPROM, programs it, and then starts the DHCP client. The DHCP client successfully obtains an IP address of 192.168.1.107. The output also shows the starting of the system message bus (dbus), the Dropbear SSH server, and the generation of a public key. The public key is displayed as a long string of characters. The output ends with the login prompt for the user 'uz7ev-evcc-sd-oob-2017-4' on the device '/dev/ttyUSB1 115200-8-N-1'.

```
***
*** Avnet Out Of Box PetaLinux Build V1.2
*** MAC address init config for eth0
***
*****

/home/root/mac_eeprom_config.sh --bus 2 --slave 0x51 --file /etc/network/interfaces --interface eth0

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
Programming EEPROM MAC ID to eth0
Conf[ 9.648172] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
ignoring network interfaces... udhcpc (v1.24.1) started
Sending discover...
[ 11.198437] random: crng init done
[ 12.716702] macb ff0e0000.ethernet eth0: link up (1000/Full)
[ 12.722372] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
Sending discover...
Sending select for 192.168.1.107...
Lease of 192.168.1.107 obtained, lease time 86400
/etc/udhcpc.d/50default: Adding DNS 192.168.1.1
done.
Starting system message bus: dbus.
Starting Dropbear SSH server: Generating key, this may take a while...
Public key portion is:
ssh-rsa AAAAB3NzaClyc2EAAAADAQABAAQCBCLN+s/Duz+suYIyuUrboZbI4W9euUnVqCgEBYTyQHA1EIjck6gntPugLVR+mk3ifrE7
j+zWxKyUAR9zX3k8v4nnlDELdWqDqmKDe8qZEAK6izI7VMYVZ+2MVvomoZF22oUs4hOB1XDoaZWAj0DNmN40w4TtmareitjPxf/DThrz9a
ktODqdYbJ6XUVyhTenL2ViiAccAPjr5i7GoQLwHAmKgaKsnH0oZ0IwldjB8tpUaLPiDXyivYnT3aMALuU2YCqCDF/d8SbTbEErnAFRCM5
UF1QIKPJQ69PYhbngNBLetbdGmRmYf2F2Hwr6A1+7H/CCozoPe4+59Z/30E7D root@uz7ev-evcc-sd-oob-2017-4
Fingerprint: md5 96:5a:06:07:38:43:e6:00:2b:fa:3e:03:bf:98:69:92
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: █
/dev/ttyUSB1 115200-8-N-1
```

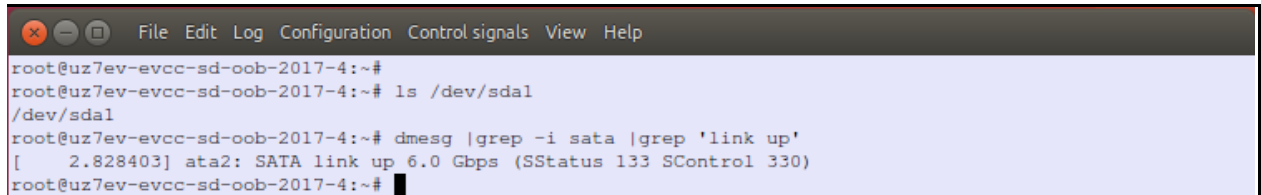

Experiment 2: Running SATA File Read/Write Tests

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 SATA device enumeration to verify that the SATA interface speed is configured correctly and that the target SATA drive is detected correctly. If the **/dev/sda1** device listing does not appear as expected, verify your SATA cable connection matches the one shown in [Appendix I: Troubleshooting SATA Connection](#) further into this document.

```
# ls /dev/sda1

# dmesg | grep -i sata | grep 'link up'
```



```
root@uz7ev-evcc-sd-oob-2017-4:~#
root@uz7ev-evcc-sd-oob-2017-4:~# ls /dev/sda1
/dev/sda1
root@uz7ev-evcc-sd-oob-2017-4:~# dmesg |grep -i sata |grep 'link up'
[ 2.828403] ata2: SATA link up 6.0 Gbps (SStatus 133 SControl 330)
root@uz7ev-evcc-sd-oob-2017-4:~#
```

Running the dd Test

1. At the UltraZed serial terminal launch the included **drive-test.sh** script to perform a write test using the Linux dd command on the /dev/sda1 (SATA SSD) device using the following command:

```
# ./drive-test.sh -d sda1 -t dd
```

[illegible]

- ```
File Edit Log Configuration Control signals View Help
root@uz7ev-evcc-sd-oob-2017-4:~# ./drive-test.sh -d sdal -t dd

*** ** ** ** **** ***** ***
*** ** ** ** ** ** ** ** ** ** ** ***
*** ** ** ** ** ** ** ** **** ** ***
*** ** * * * * * ** ** ** ** ** ***
*** ** ** ** **** ** ***
*** *** **** ** ***
*** *** *** *** *** ***
*** This is a simple script to run the dd, hdparm, and ***
*** bonnie++ test applications to determine the maximum ***
*** maximum achievable read and write performance for SATA ***
*** and USB SSDs and Flash drives. ***
*** ***
*** More information about bonnie++ can be found at ***
*** http://www.coker.com.au/bonnie++/readme.html ***
*** ***
*** This test will unmount the drive if it is already mounted! ***
*** ***

Press enter to continue...
[3653.165249] FAT-fs (sdal): Volume was not properly unmounted. Some data may be corrupt. Please run fsck.
Use the 'dd' command to test how long it takes to write a 4GB file to the disk.
time sh -c dd if=/dev/zero of=/mnt/sdal/test.tmp bs=4k count=1000000 && sync

1000000+0 records in
1000000+0 records out
4096000000 bytes (4.1 GB, 3.8 GiB) copied, 53.7103 s, 76.3 MB/s

real 0m56.797s
user 0m0.476s
sys 0m16.072s
root@uz7ev-evcc-sd-oob-2017-4:~#
```

- Page 11

## Running the hdparm Test

1. At the UltraZed serial terminal launch the included **drive-test.sh** script to perform a series of three cached and uncached read tests using the Linux `hdparm` command on the `/dev/sda1` (SATA SSD) device using the following command:

```
./drive-test.sh -d sda1 -t hdparm
```

[illegible]



- ```

***      *      *      *      *      *      *      *      *      *      *      *      *      *      *      *      *
***      *      *      *      *      *      *      *      *      *      *      *      *      *      *      *      *
***      *      *      *      *      *      *      *      *      *      *      *      *      *      *      *      *
*** This is a simple script to run the dd, hdparm, and
*** bonnie++ test applications to determine the maximum
*** maximum achievable read and write performance for SATA
*** and USB SSDs and Flash drives.
***
*** More information about bonnie++ can be found at
*** http://www.coker.com.au/bonnie++/readme.html
***
*** This test will unmount the drive if it is already mounted!
***
*****
Press enter to continue...
[ 4665.087760] FAT-fs (sdal): Volume was not properly unmounted. Some data may be corrupt. Please run fsck.

Use the 'hdparm' command to test the read times for the disk.
Run this test a few times and calculate the average.
hdparm -T -t /dev/sdal

/dev/sdal:
Timing cached reads:   2312 MB in  2.00 seconds = 1155.82 MB/sec
Timing buffered disk reads: 1408 MB in  3.00 seconds = 468.83 MB/sec

/dev/sdal:
Timing cached reads:   2310 MB in  2.00 seconds = 1154.68 MB/sec
Timing buffered disk reads: 1406 MB in  3.00 seconds = 468.59 MB/sec

/dev/sdal:
Timing cached reads:   2308 MB in  2.00 seconds = 1154.16 MB/sec
Timing buffered disk reads: 1404 MB in  3.00 seconds = 467.43 MB/sec

root@uz7ev-evcc-sd-oob-2017-4:~# █
/dev/ttyUSB1 115200-8-N-1
DTR RTS CTS CD DSR RI

```

- Page 13

Running the bonnie++ Test

1. At the UltraZed serial terminal launch the included **drive-test.sh** script to perform a write test using the Linux **bonnie++** command on the **/dev/sda1** (SATA SSD) device using the following command:

```
# ./drive-test.sh -d sda1 -t bonnie++
```

[illegible]

- Depending upon the read/write throughput capability of the attached SATA drive, the performance test script will complete in about 45 minutes. The bonnie++ test is very thorough and examines throughput using both sequential and random reads and writes. Upon completion of the bonnie++ test, a throughput summary report will be shown which summarizes the total amount of data transferred along with the measured throughput rate over the course of the test.

```

[ 5076.778049] FAT-fs (sdal): Volume was not properly unmounted. Some data may be corrupt. Please run fsck.

Use the 'Bonnie++' command to test the time for sequential and random
reads and writes for the disk.
NOTE: This test takes a few minutes, depending on the speed of the disk
bonnie++ -d /mnt/sdal/tmp -r 4096 -n 16 -u root

Using uid:0, gid:0.
Writing with putc()...done
Writing intelligently...done
Rewriting...done
Reading with getc()...done
Reading intelligently...done
start 'em...done...done...done...
Create files in sequential order...done.
Stat files in sequential order...done.
Delete files in sequential order...done.
Create files in random order...done.
Stat files in random order...done.
Delete files in random order...done.
Version 1.03e
-----Sequential Output----- --Sequential Input- --Random-
-Per Chr- --Block-- -Rewrite- -Per Chr- --Block-- --Seeks--
Machine      Size K/sec %CP K/sec %CP K/sec %CP K/sec %CP K/sec %CP /sec %CP
uz7ev-evcc-sd-oo 8G 10912 89 73385 25 72725 28 12172 99 508666 71 +++++ ++
-----Sequential Create----- -----Random Create-----
-Create-- --Read-- -Delete-- -Create-- --Read-- -Delete--
files /sec %CP /sec %CP /sec %CP /sec %CP /sec %CP /sec %CP
16 39 97 +++++ ++ 470 98 58 96 +++++ ++ 139 96
uz7ev-evcc-sd-oob-2017-4, 8G, 10912, 89, 73385, 25, 72725, 28, 12172, 99, 508666, 71, +++++, ++, 16, 39, 97, +++++, ++, 470, 98, 58, 96, +++++, ++, 139, 96

real    42m44.234s
user    22m13.660s
sys     15m45.272s
root@uz7ev-evcc-sd-oob-2017-4:~#
/dev/ttyUSB1 115200-8-N-1 DTR RTS CTS CD DSR RI

```

- Let's take a closer look at the bonnie++ test results. The test output is difficult to read, so we will explain them a bit further here.

```

Version 1.03e
-----Sequential Output----- --Sequential Input- --Random-
-Per Chr- --Block-- -Rewrite- -Per Chr- --Block-- --Seeks--
Machine      Size K/sec %CP K/sec %CP K/sec %CP K/sec %CP K/sec %CP /sec %CP
uz7ev-evcc-sd-oo 8G 10912 39 73385 25 72725 28 12172 99 508666 71 +++++ ++
-----Sequential Create----- -----Random Create-----
-Create-- --Read-- -Delete-- -Create-- --Read-- -Delete--
files /sec %CP /sec %CP /sec %CP /sec %CP /sec %CP /sec %CP
16 39 97 +++++ ++ 470 98 58 96 +++++ ++ 139 96
uz7ev-evcc-sd-oob-2017-4, 8G, 10912, 89, 73385, 25, 72725, 28, 12172, 99, 508666, 71, +++++, ++, 16, 39, 97, +++++, ++, 470, 98, 58, 96, +++++, ++, 139, 96

```

- uz7e-evcc-sd-oob-2017-4 is the machine name.
- 8G is size of the dataset. By default this is 2x the size of the RAM of the system.
- 10912 is the speed (in KBytes/sec) at which the dataset was written a single character at a time.
- 73385 is the speed (in KBytes/sec) is the speed at which a file is written a block at a time.
- 12172 is the speed (in KBytes/sec) at which the dataset was read a single character at a time.
- 508666 is the speed (in KBytes/sec) at which a file is read a block at a time.
- The +++ indicates that the test completed too quickly and could not be accurately reported.
- The last line is the output in CSV format, which can be used to import into a spreadsheet.

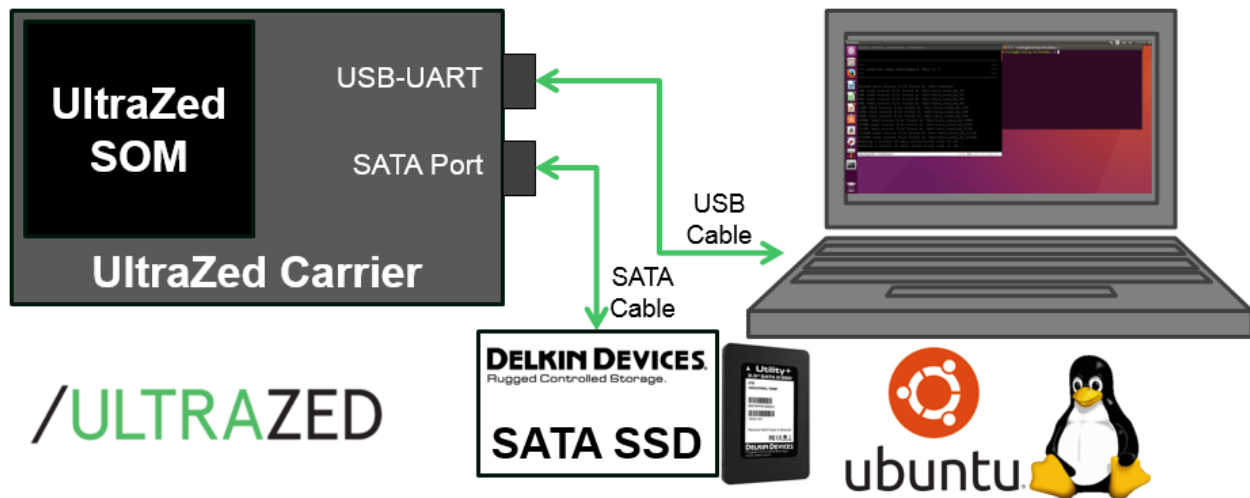
4. Continue to experiment with writing large files to the disk by modifying the test script or manually using the **bonnie++** command until you have collected the type of information that you need for your own SATA performance testing.
5. When you have completed your own SATA performance testing, be sure to shutdown the UltraZed Linux system cleanly by using the command **shutdown -h now** to ensure the system is halted and file systems properly unmounted.

```
# shutdown -h now
```


Appendix I: Troubleshooting SATA Connection

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

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



2. Verify that the SATA III drive is powered externally with 12V/5V adapter.
3. Verify that the SATA III drive is connected with a SATA III rated data cable.
4. Verify that the SATA III drive is partitioned and formatted with a file system that is compatible with the Linux build generated from the PetaLinux project. The FAT32 file system is supported with the provided pre-built image accompanying this tutorial but other file system types might also be supported.

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.

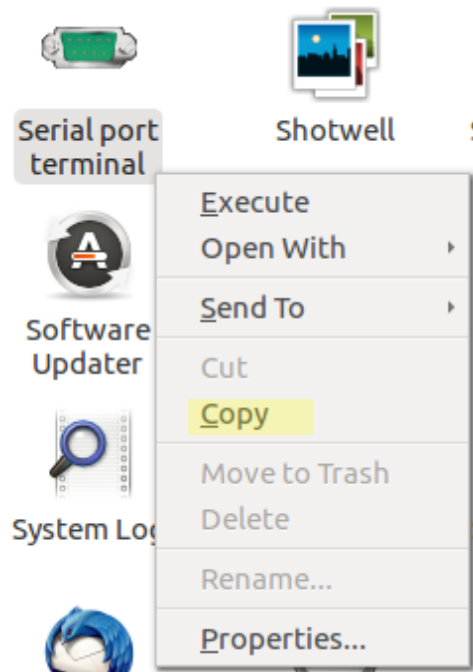
1. To make it easier to launch the terminal app (GtkTerm) without needing to provide the root password each time, open a command window and add the **current username** to the group for the **/dev/ttyUSBx** devices used for USB-UART:

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

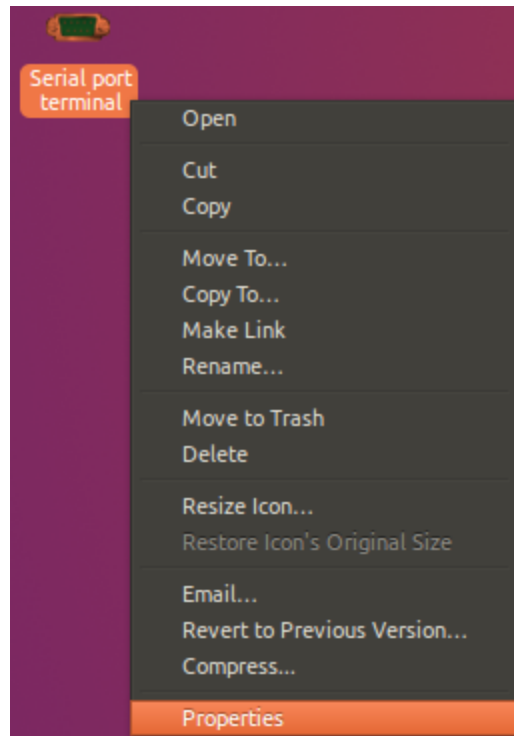
2. Install the **gtkterm** package:

```
$ 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:



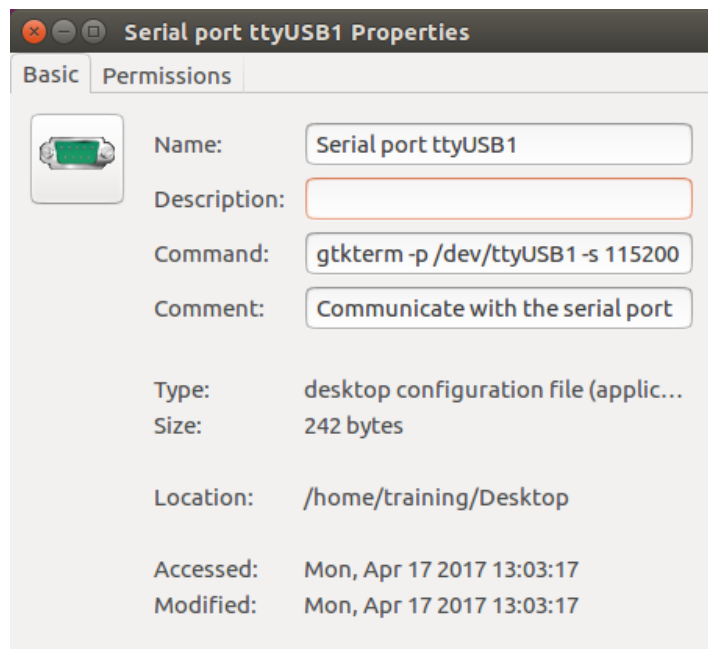
- Right-click on the new **Serial port terminal** (gtkterm) application Desktop icon and select the **Properties** option.



- Within the Properties window, set the app 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-EG 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 IO Carrier Card power switch SW8 is in the OFF position.
11. Insert the UltraZed-3EG SOM module onto the UltraZed IO 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 IO Carrier Card USB-UART port (J11).
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 IO Carrier power switch SW8 to the ON position. The UltraZed system will power on and the Power Good LED (D3) 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.

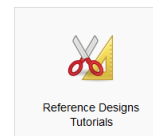
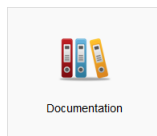
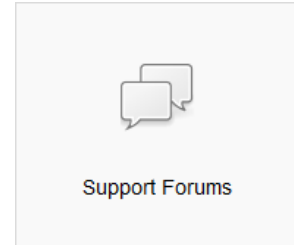
```
# relax the permissions just for ttyUSB0
KERNEL=="ttyUSB0", MODE="0666"
```

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-EG SOM Documentation
<http://zedboard.org/support/documentation/14201>
- UltraZed-EV SOM Documentation
<http://zedboard.org/support/documentation/21811>
- UltraZed-EG I/O Carrier Card
 - Documentation
<http://zedboard.org/support/documentation/17596>
 - Reference Designs
<http://zedboard.org/support/design/17596/131>
- UltraZed-EG PCIe Carrier Card
 - Documentation
<http://zedboard.org/support/documentation/17956>
 - Reference Designs
<http://zedboard.org/support/design/17956/141>
- 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

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

Internet Support

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

dd

- Benchmark disk I/O with dd
<https://www.jamescoyle.net/how-to/599-benchmark-disk-io-with-dd-and-bonnie>

bonnie++

- Simple bonnie++ example
<https://www.jamescoyle.net/how-to/913-simple-bonnie-example>
- Using bonnie++ for filesystem performance benchmarking
<https://www.linux.com/news/using-bonnie-filesystem-performance-benchmarking>
- Active and passive benchmarking with bonnie++
<http://www.brendangregg.com/ActiveBenchmarking/bonnie++.html>

hdparm

- Tune your disk with hdparm
<http://www.linux-magazine.com/Online/Features/Tune-Your-Hard-Disk-with-hdparm>

vi

- A vi cheat sheet
<http://www.lagmonster.org/docs/vi.html>
- A beginner's guide to vi
<https://www.howtogeek.com/102468/a-beginners-guide-to-editing-text-files-with-vi/>
- Basic vi commands
<https://www.cs.colostate.edu/helpdocs/vi.html>
- A vi editor howto
<https://www.washington.edu/computing/unix/vi.html>

Revision History

Date	Version	Revision
03 Apr 17	01	Initial Release
25 May 2018	02	Update to add UltraZed-EG PCIe carrier and add UltraZed-EV SOM and carrier