Lab0: Start with Ultra 96v2 and Pynq

This document assumes that you have installed beforehand the “DPU on Pynq”. The documentation and steps are at github. https://github.com/Xilinx/DPU-PYNQ
We generated a detailed internal doc called “Install Pynq DPU and FINN” and as result the “image” to download to the SD card.
You can download the IMG that we generate from moodle link (Ultra96v2_DPU_FINN_image.7z)

1. Setup the ultra96v2 board.

1.1 burn the image to an SD card (only if you start here)

Use (minimum 16GB) SD. In Windows: Win32DiskImager or BelenaEtcher.
In Linux: use BalenaEtcher or use `dd` command (not recommended if you are not an advanced user)

```
$ sudo dd bs=4M if={image_name}.img of=/dev/sd{X} status=progress conv=fsync
```

It will create two partitions into the SD:

- BOOT – partition of type FAT (size=400MB)
- ROOTFS – partition of type EXT4

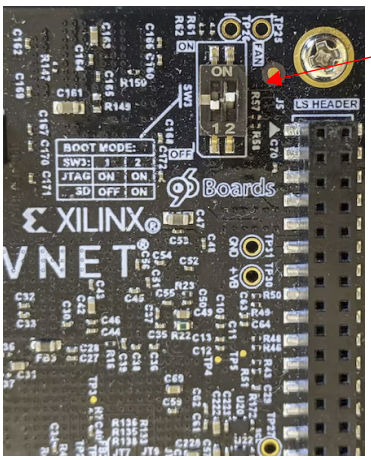
The first BOOT partition was created with a size of 400MB, and contains the following files:

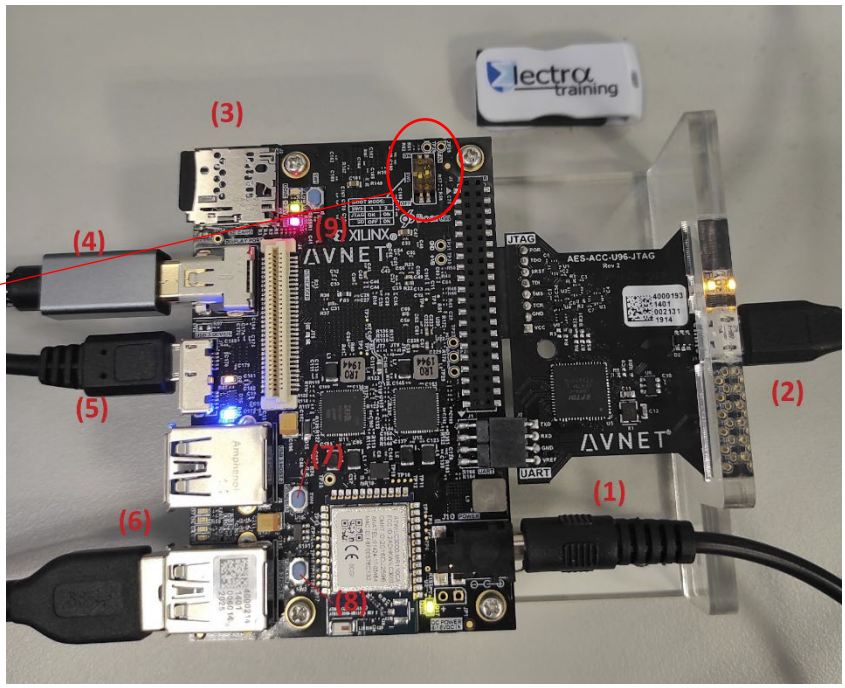
- BOOT.BIN
- boot.scr
- image.ub

The second ROOTFS partition contains the rootfs.tar.gz content, and is pre-installed with the Pynq image including the DPU and FINN.

1.2. Connect the board. Plug the following components:

Power plug (1)
SD-Card (2)
Micro-USB1: Connect to a terminal (3)
MiniDisplay port: Connect to a external monitor (4)
Micro-USB2 (o mini usb-B): give an ethernet connection (5)
USB (x2). Connect camera, mouse, keyboard, pendrive, etc. (6)
Power On button (7)
Reset Button (8)





2. Boot the Ultra96v2 board

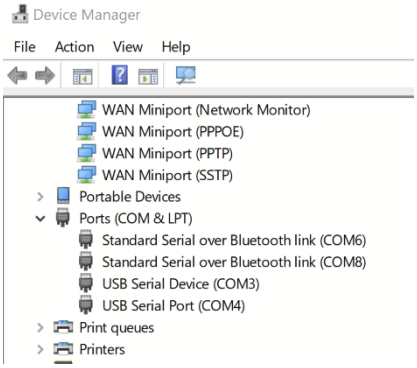
Press the Power On button (7), the board will boot, and several LEDs will be on. After a few seconds, the monitor connected through Mini-DisplayPort will show a welcome page.

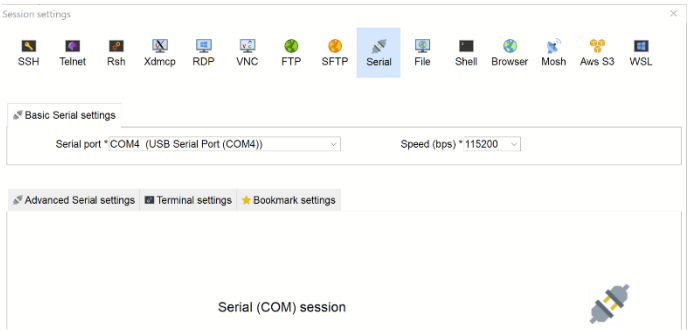
2.1. Connect using a terminal to the board:

Prerequisite: It is necessary to connect properly the micro-usb to serial connector (2 in figure) to the host computer

In Linux: open a terminal (putty, minicom, etc) to /dev/ttyUSB1 at 115200

In Windows: open a terminal (mobaXterm, Teraterm, etc).
In the device manager see where the board is connected (In this example COM4).





Open the terminal. In MobaXterm. New connection -> serial

Note1: If you want to see booting again, you can press the reset button (8 in figure). You will see boot process in the console (and in the monitor too if it is connected)

You can list the content of the embedded Linux (ls -l). Furthermore, you will list the user “xilinx” home directory

```
(pynq-venv) xilinx@pynq:/$ pwd
/
(pynq-venv) xilinx@pynq:/$ ls
bin   dev  home  lib64       media  opt   root  sbin  srv  tmp  var
boot  etc  lib   lost+found  mnt    proc  run   snap  sys  usr
(pynq-venv) xilinx@pynq:/$
```

The **ifconfig** show that usb0 is a network interface with address 192.168.3.1

```
xilinx@pynq:~$ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 6982  bytes 513475 (513.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 6982  bytes 513475 (513.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

usb0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.3.1  netmask 255.255.255.0  broadcast 192.168.3.255
        inet6 fe80::50bb:a5ff:fe8a:3866  prefixlen 64  scopeid 0x20<link>
        ether 52:bb:a5:8a:38:66  txqueuelen 1000  (Ethernet)
        RX packets 2881  bytes 711508 (711.5 KB)
        RX errors 0  dropped 4  overruns 0  frame 0
        TX packets 1324  bytes 555204 (555.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        ether f8:f0:05:76:c1:32  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

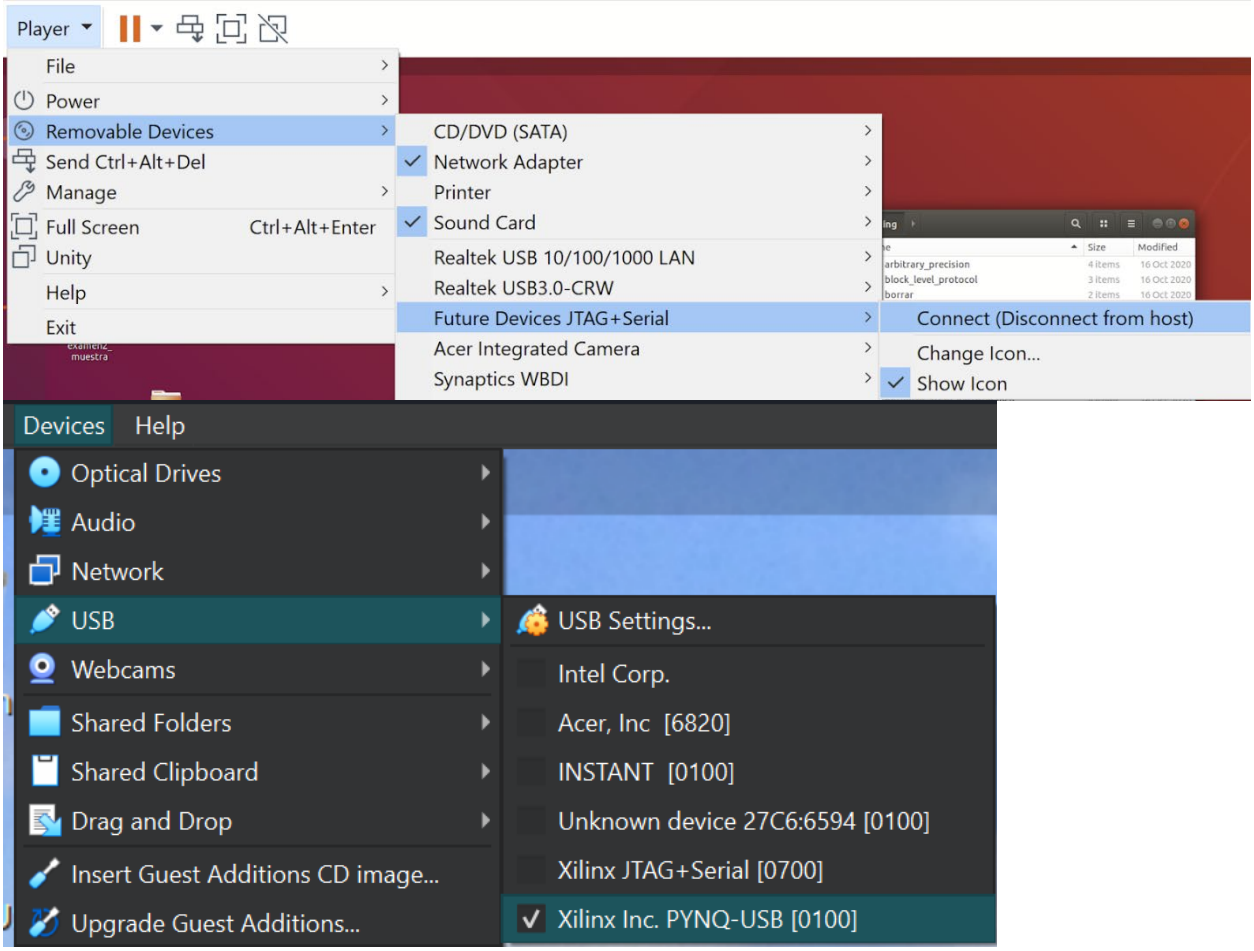
2.2. Connect the new Ethernet (usb0 in Ultra96v2) into the host machine.

Prerequisite: It is necessary to connect properly the micro-usb (5 in figure) to the computer

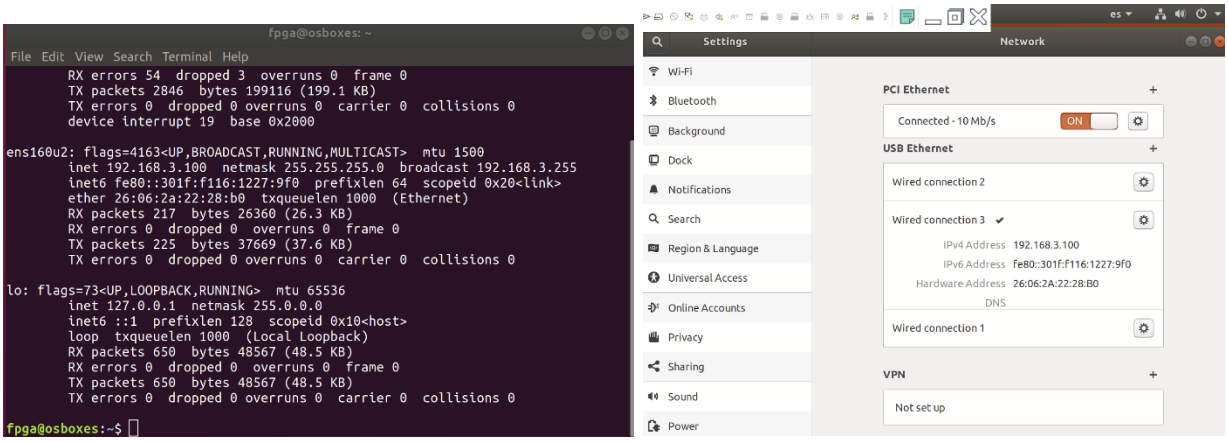
2.2a. In Linux

If you are using a virtual machine (optional).

First capture the device (**removable device -> Linux Foundation PYNQ-USB**)



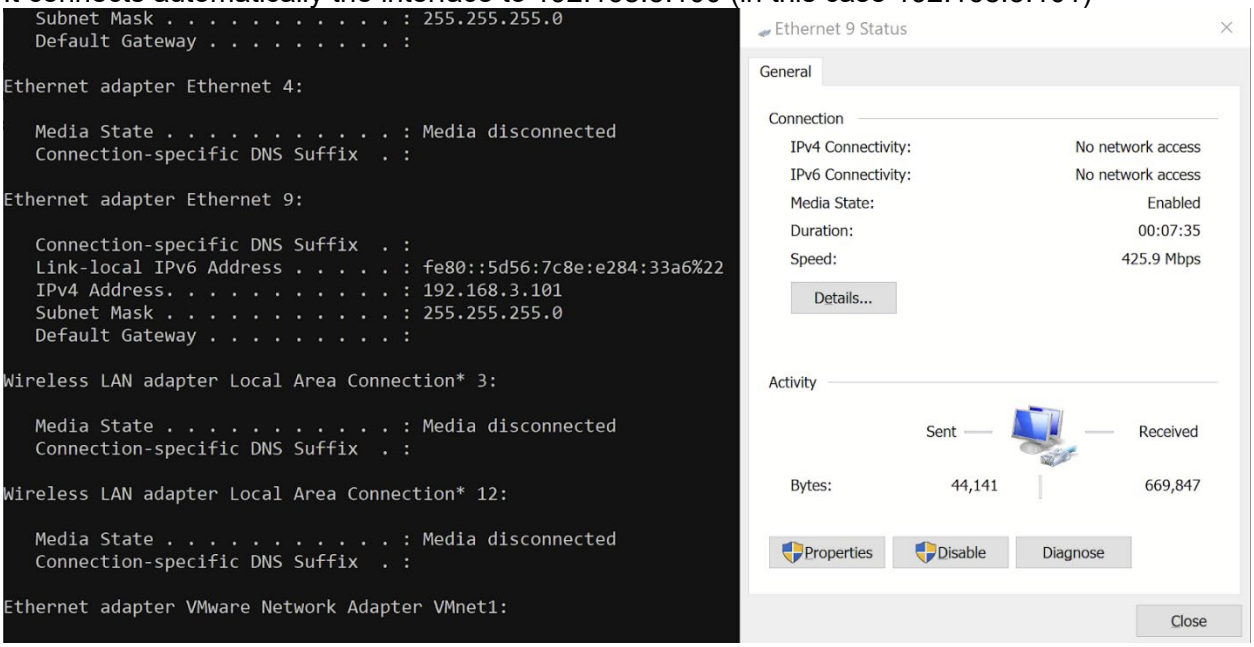
The new cable connects to a new Ethernet connection. By default, connects automatically this interface to 192.168.3.100



Now you can open a web browser and connect to the embedded system at 192.168.3.1

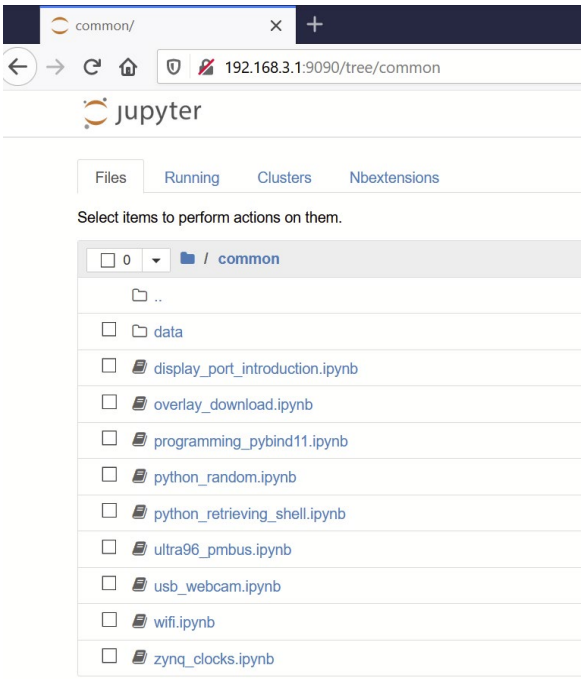
2.2b. In window

It connects automatically the interface to 192.168.3.100 (in this case 192.168.3.101)



2.3. Connect to the Jupyter server into the board

Connect to the Jupyter server in the Ultra96v2 board at 192.168.3.1. The Jupyter notebook has the password: Xilinx (lowercase).
(optional) review the common folder and interact with the notebook.



2.4. (optional) connect to the board using ssh

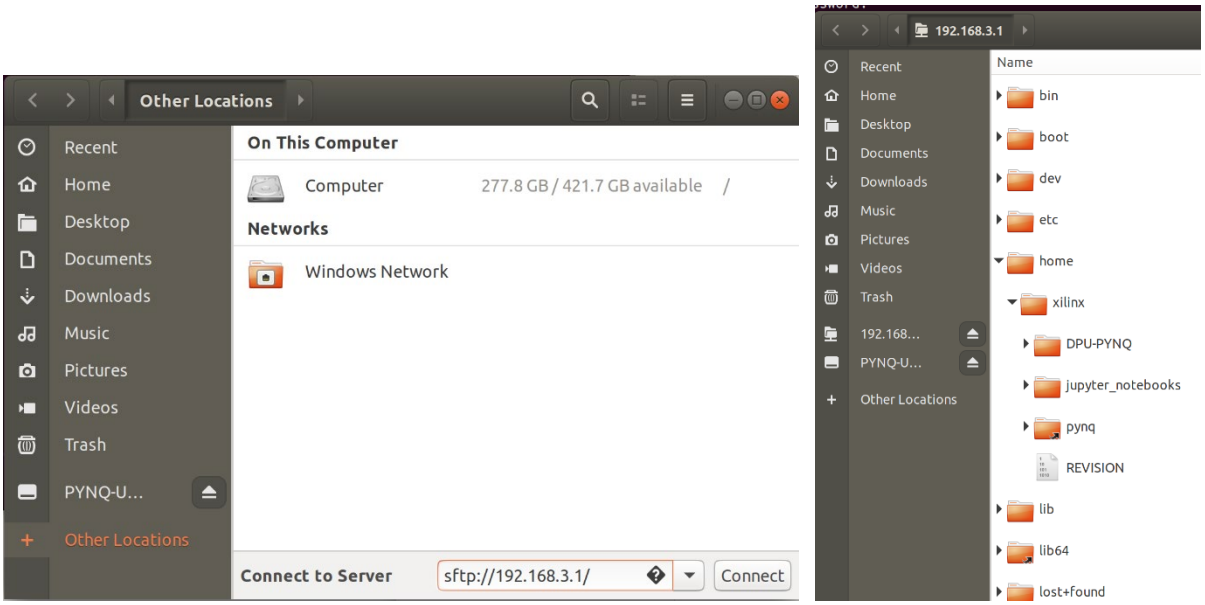
Since you have an Ethernet connection, it is possible to use ssh and sftp.

2.4.a In Linux (Virtual machine)

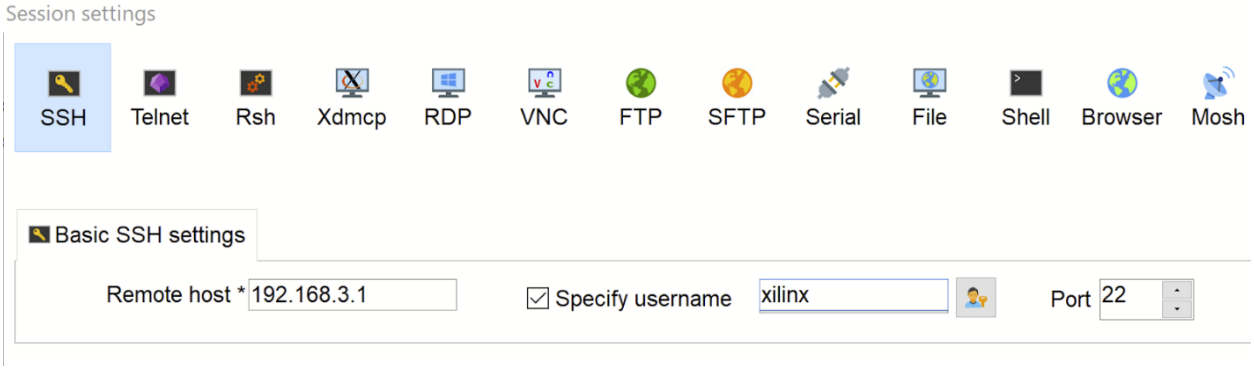
In a terminal run ssh **xilinx@192.168.3.1**

```
electra@ubuntu:~$ ssh xilinx@192.168.3.1
xilinx@192.168.3.1's password:
Welcome to PYNQ Linux, based on Ubuntu 18.04 (GNU/Linux 5.4.0-xilinx-v2020.1 aarch64)

Last login: Mon Feb  8 13:34:24 2021
xilinx@pynq:~$
```

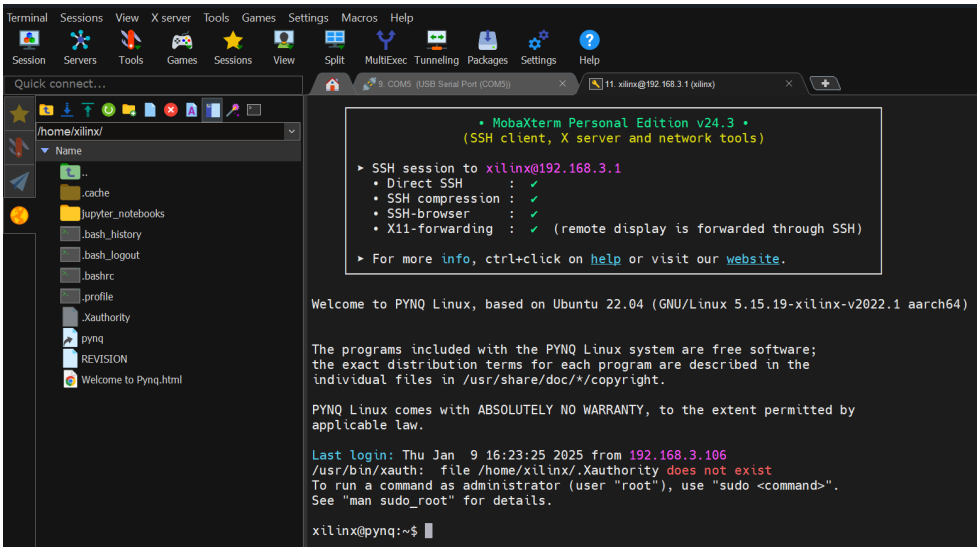
In a file browser, you can open an SFTP (secure FTP) connection using a file browser and use connect **sftp://xilinx@192.168.3.1**



2.4.b In Windows (using mobaXterm)



The ssh opens a very useful sftp window in the left pane.



Note: if you find a “permission denied” to access to /home/xilinx

```
Last login: Thu Jan  9 16:17:39 2025
Could not chdir to home directory /home/xilinx: Permission denied

/usr/bin/xauth:  timeout in locking authority file /home/xilinx/.Xauthority
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

-bash: /home/xilinx/.bash_profile: Permission denied
```

Change to root (`sudo su`) and change the owner of /home/xilinx subdirectory (`chown xilinx:Xilinx /home/xilinx`)

```
(pynq-venv) xilinx@pynq:/$ sudo su
[sudo] password for xilinx:
root@pynq:/# ls -l /home/
total 8
drwxr-xr-x 3 root root 4096 Mar 18  2023 fred
drwxr-x--- 3  127  135 4096 Mar 18  2023 xilinx
root@pynq:/# chown xilinx:xilinx /home/xilinx/
root@pynq:/# ls -l /home/
total 8
drwxr-xr-x 3 root   root   4096 Mar 18  2023 fred
drwxr-x--- 3 xilinx xilinx 4096 Mar 18  2023 xilinx
root@pynq:/#
```

Add writing permission for everyone, so you can write with sftp from different user (`chmod -R 777 /home/xilinx`)

```
root@pynq:/home# chmod -R 777 /home/xilinx/
root@pynq:/home# ls -l /home/
total 8
drwxr-xr-x 3 root   root   4096 Mar 18  2023 fred
drwxrwxrwx 4 xilinx xilinx 4096 Jan  9 16:35 xilinx
root@pynq:/home#
```

2.4.c Connect as root user to ssh

It is a “potential security risk” but is very practical to access as root to the embedded system.

Firstly you must know the root passw. To reset root passw to the board.

```
(pynq-venv) xilinx@pynq:/etc/ssh$ sudo su
[sudo] password for xilinx:
root@pynq:/etc/ssh# passwd root
New password:
Retype new password:
passwd: password updated successfully
root@pynq:/etc/ssh#
```

If you try to access as root using aterminal ssh **root@192.168.3.1**

```
root@192.168.3.1's password:
Access denied
root@192.168.3.1's password:
```

You will need to edit the /etc/ssh/sshd_config file (for example use nano or vi)

```
root@pynq:/etc/ssh# nano sshd_config
```

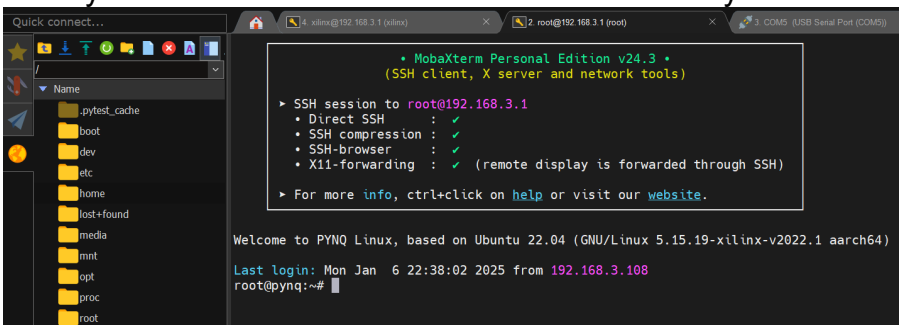
Add “PermitRootLogin” yes below # Authentication.

```
  GNU nano 6.2                        sshd_config *
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:
PermitRootLogin yes

#LoginGraceTime 2m
#PermitRootLogin prohibit-password
#StrictModes yes
#MaxAuthTries 6

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^/ Go To Line
```

Save changes and then, restart ssh service to apply the changes:

```
sudo service sshd restart
```

Then you can connect as root and access the file system without restrictions

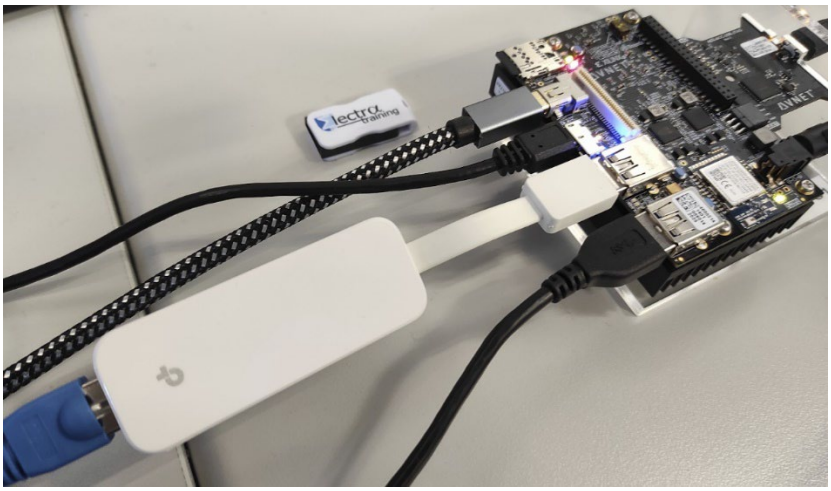


2.5. Connect the embedded system to the internet

There are several alternatives.

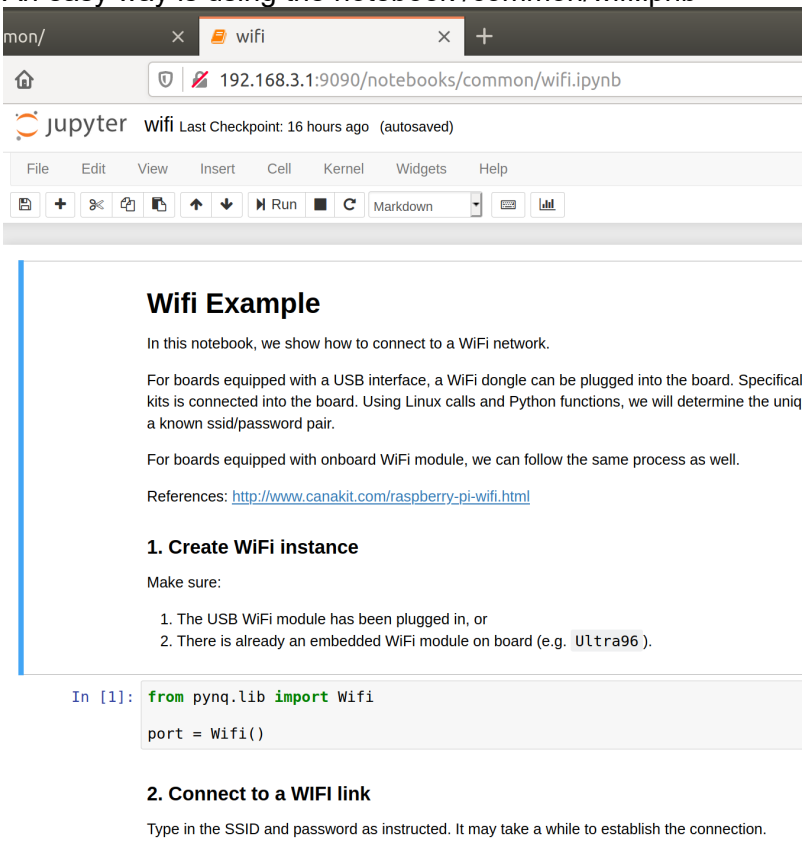
2.5.a. Connect an Ethernet adaptor and plug into an Ethernet plug.

Depending on the configuration, check the internet connection.



2.5.b. Connect using Wi-Fi

Ultra96v2 has a Wi-Fi connection. There are several ways to connect to a Wi-Fi connection. An easy way is using the notebook /common/wifi.ipnb



Then you can see using ifconfig the new connection

```
xilinx@pynq:~$ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 13928  bytes 962639 (962.6 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 13928  bytes 962639 (962.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

usb0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.3.1  netmask 255.255.255.0  broadcast 192.168.3.255
        inet6 fe80::8c67:f0ff:fe20:f163  prefixlen 64  scopeid 0x20<link>
        ether 8e:67:f0:20:f1:63  txqueuelen 1000  (Ethernet)
        RX packets 7344  bytes 1045559 (1.0 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 11097  bytes 12877066 (12.8 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.47  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::faf0:5ff:fe76:bfda  prefixlen 64  scopeid 0x20<link>
        ether f8:f0:05:76:bf:da  txqueuelen 1000  (Ethernet)
        RX packets 21930  bytes 32730220 (32.7 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10144  bytes 680413 (680.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

xilinx@pynq:~$ ping www.google.com
PING www.google.com (142.250.184.164) 56(84) bytes of data.
64 bytes from mad07s23-in-f4.1e100.net (142.250.184.164): icmp_seq=1 ttl=119 time=8.51 ms
64 bytes from mad07s23-in-f4.1e100.net (142.250.184.164): icmp_seq=2 ttl=119 time=9.62 ms
64 bytes from mad07s23-in-f4.1e100.net (142.250.184.164): icmp_seq=3 ttl=119 time=8.53 ms
64 bytes from mad07s23-in-f4.1e100.net (142.250.184.164): icmp_seq=4 ttl=119 time=8.68 ms
^C
--- www.google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 8.513/8.841/9.624/0.456 ms
xilinx@pynq:~$
```

2.6. Basic commands (reboot and power off)

You can reboot the board using the Linux command

```
$ sudo reboot
```

Or a hardware reboot, pressing the physical reboot button (8).

To power off the board

```
$ sudo poweroff
```

Or pressing the physical PowerOn-PowerOff button (7).

3. Run the Basic - Examples

If you are familiar with Jupyter, probably you do not need to run all the examples.

3.1 (optional) Getting started

The getting started is for the ones who have no experience using Jupyter notebooks. There are three basic notebooks:
1_jupyter_notebooks.ipynb
2_python_environment.ipynb
3_jupyter_notebooks_advanced_features.ipynb

Ensure that you have the knowledge of “1_jupyter_notebooks” and “2_python_environment”

3.2 Basic platform examples (Common folder)

These are basic examples, but using specific contents of the FPGA platform

python_random.ipynb Nothing specific from FPGA, only generates random number

python_retrieving_shell.ipynb. Shows how to use Linux command from a Python notebook

Ultra96_pmbus.ipynb. The Ultra96 has some support for monitoring power rails on the board using PMBus. PYNQ exposes these rails through the get_rails function that returns a dictionary of all the rails available to be monitored.

Explore all projects in folder

4. Run the DPU Examples (next lab)

You can review the DPU folder. Running and interacting with the DPU (Deep Learning Processing Unit) will be the next laboratory.

Deliverables:

Nothing. Be sure that you understand the setup