EGR 680
Fall 2018
Laboratory #10

PYNQ Embedded Design using Jupyter Notebooks

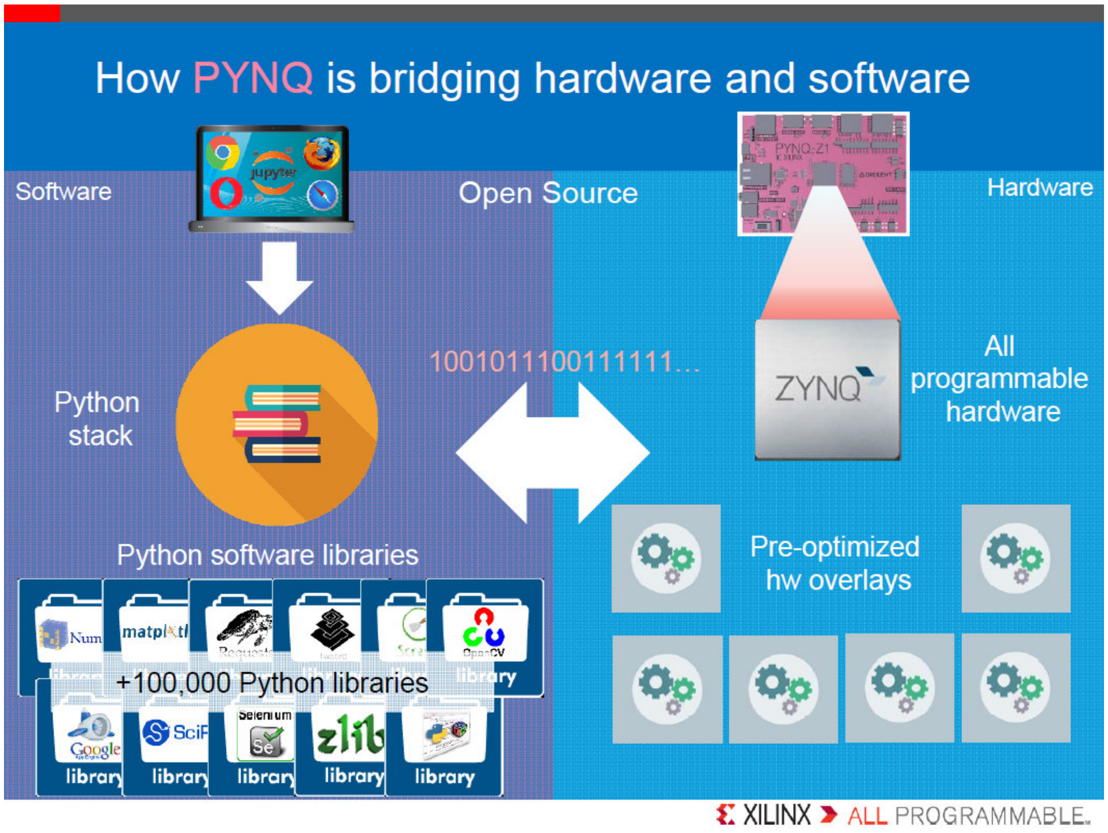
Due Date: By beginning of next lab section

Objectives

- Set up the PYNQ board to communicate with Jupyter Notebooks
- Learn how to use and understand the PYNQ python drivers
- Create an interactive GUI that communicates with the PYNQ board

---

Part I – Getting Started with PYNQ


How PYNQ is bridging hardware and software
Software
Open Source
Hardware
Python stack
1001011100111111...
ZYNQ
All programmable hardware
Python software libraries
+100,000 Python libraries
Pre-optimized hw overlays
XILINX ALL PROGRAMMABLE.

Now that you have experienced hardware design using Verilog, microcontroller app design using C, and object oriented programming using Python, it's time to pull everything together and make a complete embedded system utilizing all three. By using Python on an FPGA, developers can take advantage of FPGA resources without having to use ASIC-style design tools such as Vivado. However, the base layer (overlay) on the FGPA that allows the user to accomplish this is quite complicated. The design of a custom overlay would still need to be designed by a hardware engineer. What Xilinx has done to limit the need for custom overlays is put together a variety of pre-made overlays that can be loaded onto the board just like a software library. Below is a block diagram of the **PYNQ** architecture to better understand how the three programming layers communicate.

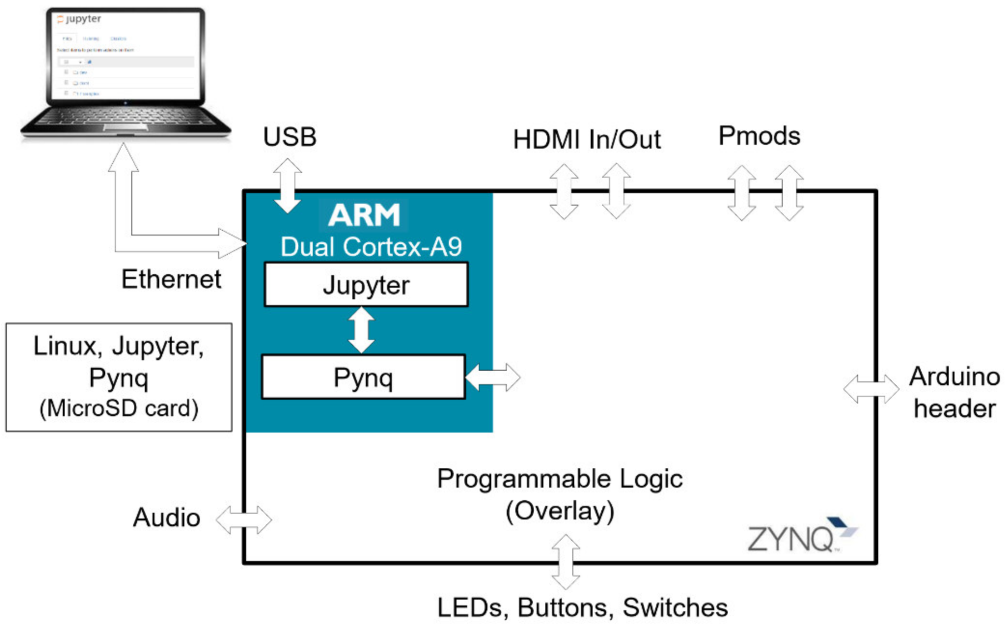


Figure 1: PYNQ block diagram

Before we can get started, we need to configure the **PYNQ** board to communicate with Jupyter Notebooks. The **PYNQ** board requires the **PYNQ** image, which is a bootable Linux image that includes the **PYNQ** Python package and other open-source packages. The **PYNQ** image needs to be installed on an 8GB or greater SD card using a disk imaging utility.

1. In your web browser, navigate to the **PYNQ** website at *www.pynq.io*, or go to the direct link below for the direct download of the current image. The current **PYNQ** image is **v2.1**. The image download is about 1GB, so it may take some time to download.
    a. http://files.digilent.com/Products/PYNQ/pynq_z1_v2.1.img.zip

2. Navigate to the location of the download on your PC and extract the .ZIP file.

3. A disk imaging utility needs to be installed on your PC to install the PYNQ image on the SD card. Download and install the **Etcher** disk imaging utility from the link below. You can also use **Win32DiskImager**, but for this tutorial, **Etcher** will be used.
    a. https://etcher.io/
    b. https://sourceforge.net/projects/win32diskimager/

4. Once installed, launch the **Etcher** desktop app.

5. Insert the SD card into your PC or an external SD card reader. Make sure the SD card is completely empty. If it is not, you will need to format the SD card by right-clicking on the drive in Windows File Explorer and selecting **Format**.

6. Once the SD card is inserted and properly formatted, click **Select image** in the **Etcher** app as shown in Figure 2 below.

7. Next, click **Select drive** and select your SD card and then click **Flash**. This will take about 10 minutes to complete.

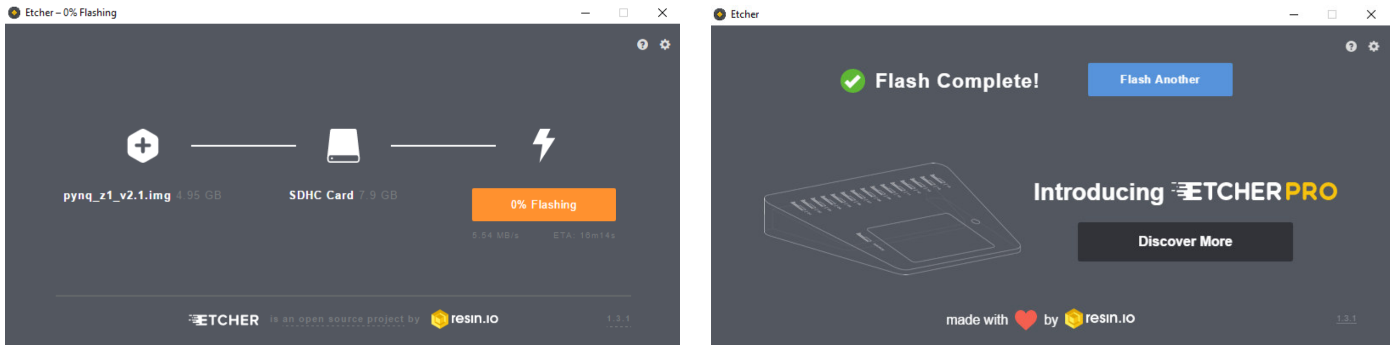


Figure 2: Etcher disk imaging utility

8. Remove the SD card from your PC and insert it into the SD card slot on the **PYNQ** board.

9. Set the **JP4** jumper to the SD position. This sets the board to boot from the SD card.

10. Connect the board to the PC with a micro USB cable and an Ethernet cable. See the figure below for the proper configuration.

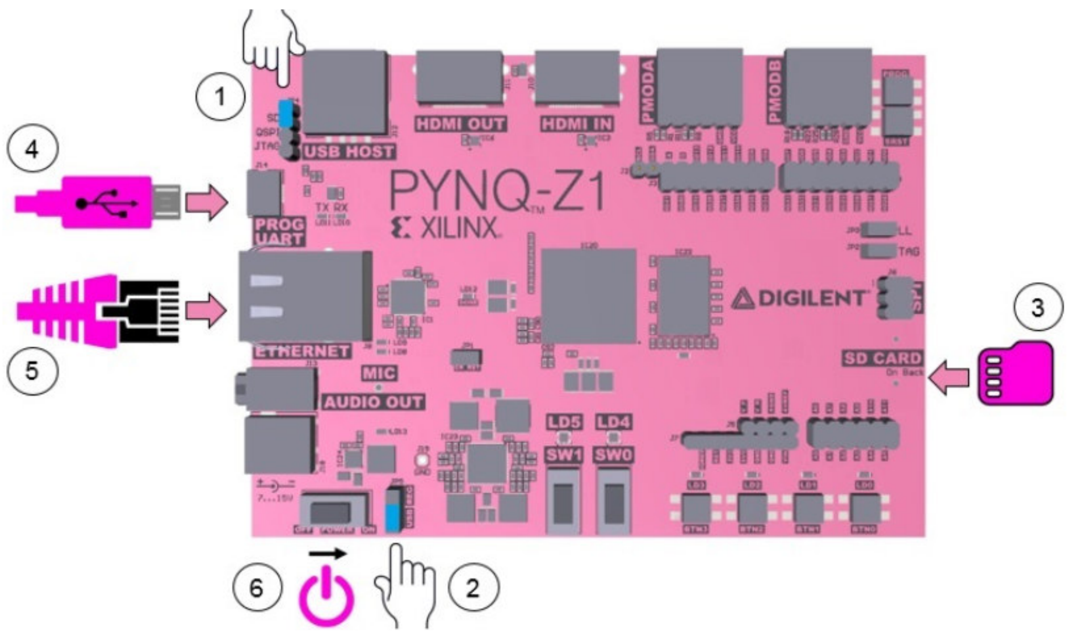


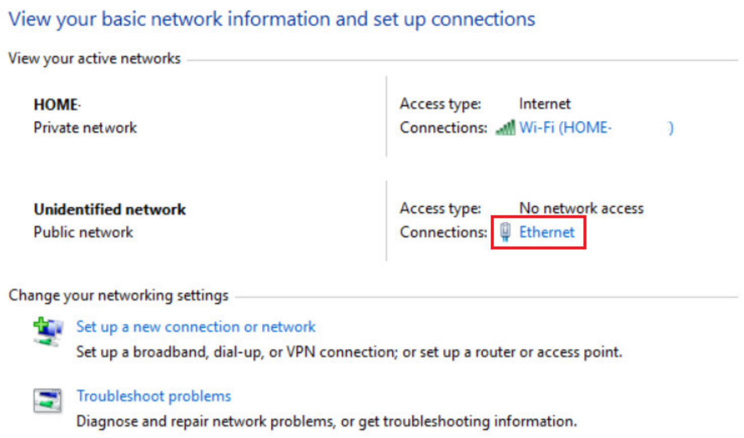
Figure 3: Configuring the PYNQ board

11. Turn on the **PYNQ** board with the on-board **power** switch. The red LD13 LED will turn ON to indicate power and then the green LD12 LED will turn ON after a few seconds to indicate a successful bitstream upload to the board. After a minute, you should see two blue LD4-LD5 LEDs and four green LD0-LD3 LEDs flash simultaneously. The blue LEDs will then turn OFF while the greens stay ON. This indicates that the system is now booted and ready for use.

12. To establish an Ethernet communication between the **PYNQ** board and a PC, the network card on the PC must be configured with a static IP address. To do this, go to **Control Panel → Network and Internet → Network and Sharing Center**. You should see your Wi-Fi and Ethernet connections here.

13. Click on the Ethernet link as shown below in red. This brings up the Ethernet Status window. Click **Properties** and click **Yes** when you get the **User Account Control** pop-up.

Note: If you do not have a dedicated Ethernet port on your PC, you will need you use a USB to Ethernet adapter. The following setup instructions should be the same regardless of the method you perform. The only difference should be the name of the Ethernet connection shown on your PC. It may be called **Ethernet 3** or something similar.



View your basic network information and set up connections

View your active networks

HOME
Private network

Access type: Internet
Connections: Wi-Fi (HOME-)

Unidentified network
Public network

Access type: No network access
Connections: Ethernet

Change your networking settings

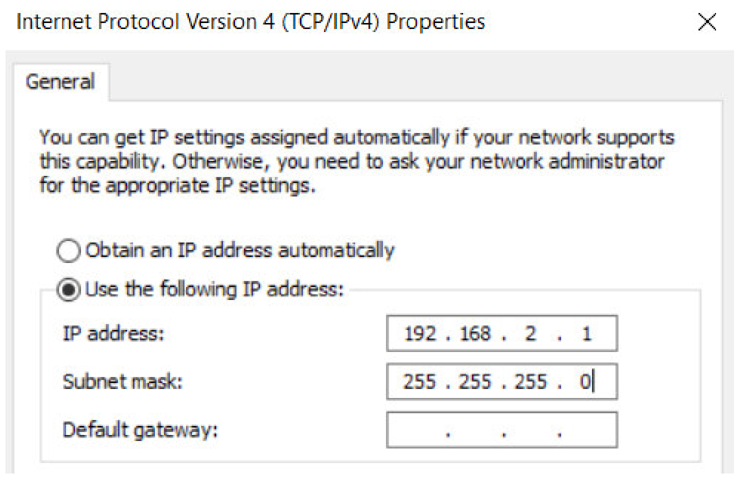
Set up a new connection or network
Set up a broadband, dial-up, or VPN connection; or set up a router or access point.

Troubleshoot problems
Diagnose and repair network problems, or get troubleshooting information.

Figure 4: Network connection setup

14. In the Ethernet Properties window, double-click on **Internet Protocol Version 4 (IPv4)**. This will bring up the window to set a static IP as shown in Figure 5.

15. Select the **Use the following IP address** radio button and enter **192.168.2.1** for the IP address and **255.255.255.0** for the subnet mask and then click **OK → OK → Close**. *You can set the last number in the IP address to any number between 0 and 255 except 99. **192.168.2.99** is the IP address of the **PYNQ** board.* Your Ethernet connection is now configured for a static IP address.



Internet Protocol Version 4 (TCP/IPv4) Properties

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

○ Obtain an IP address automatically
● Use the following IP address:

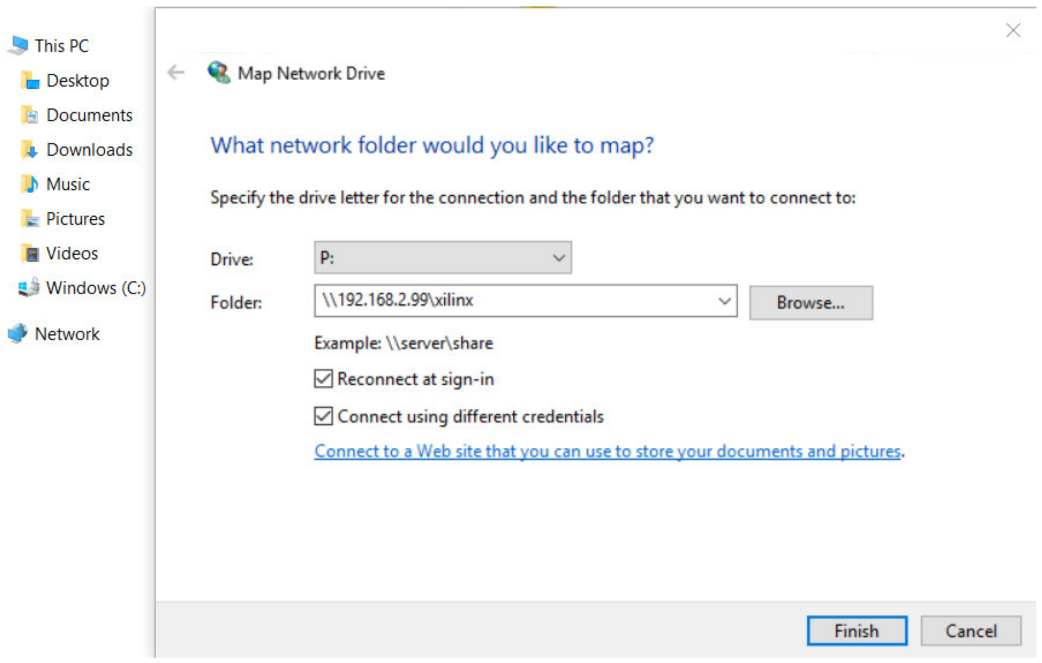
IP address: 192 . 168 . 2 . 1
Subnet mask: 255 . 255 . 255 . 0
Default gateway: . . .

Figure 5: IPv4 settings for a static IP address

16. To make file sharing and Python library viewing easier between the **PYNQ** board and your PC, a network drive can be mapped from your PC to the **PYNQ** board. Open the **Windows File Explorer** and right-click **Network**. Select **Map Network Drive** from the menu

17. Select any drive letter that is not already in use and enter the address as shown in figure 6 below. Also,

make sure to select **Connect using different credentials** before clicking **Finish**. Click **Finish.**



What network folder would you like to map?

Specify the drive letter for the connection and the folder that you want to connect to:

Drive: P:

Folder: \\192.168.2.99\xilinx

Example: \\server\share

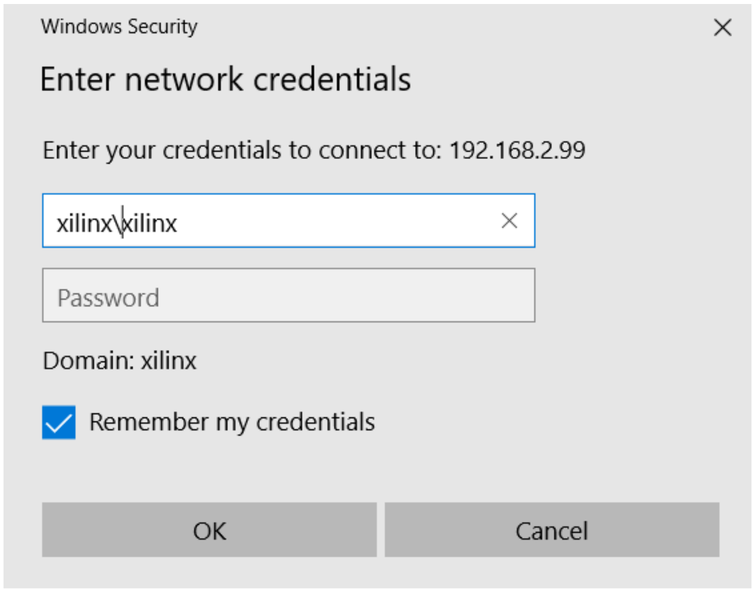
☑ Reconnect at sign-in

☑ Connect using different credentials

Connect to a Web site that you can use to store your documents and pictures.

Figure 6: Mapping a network drive

18. The **PYNQ** board username is ***xilinx*** and the password is ***xilinx***. You may need to append the domain name to the username if you are having trouble connecting. See the figure 7 below and click **OK**.



Enter network credentials

Enter your credentials to connect to: 192.168.2.99

xilinx\xilinx

Password

Domain: xilinx

☑ Remember my credentials

Figure 7: PYNQ login credentials

19. If the **PYNQ** board has been successfully mapped to the PC, it should now be visible in the **Windows File Explorer** as an available drive (Figure 8). You can view/edit/add to the contents of the SD card on the **PYNQ** board through the PC.

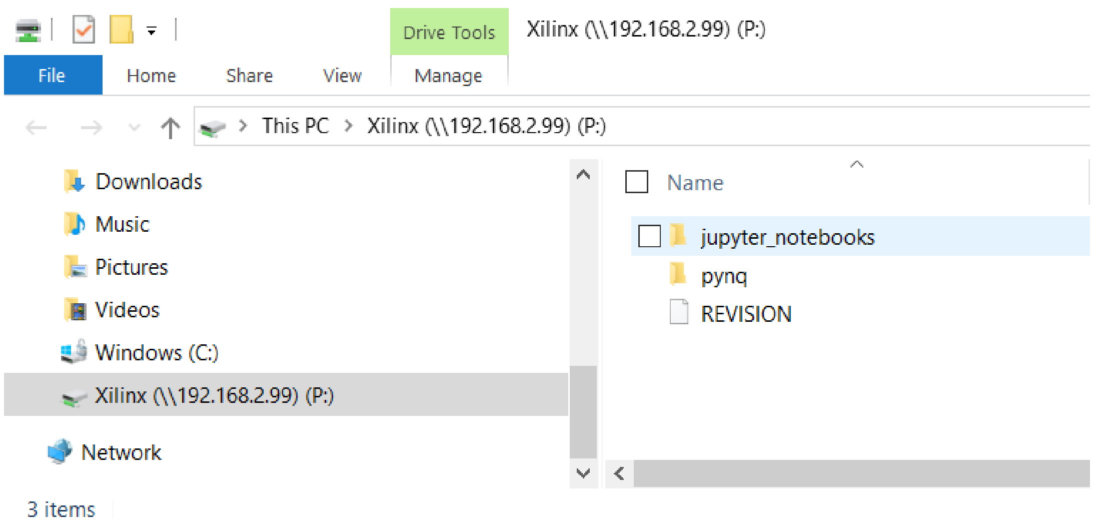


Figure 8: PYNQ mapped as a network drive to a PC

20. As mentioned, **PYNQ** is accessed via Jupyter Notebooks through a web browser. It is recommended that you use Chrome as the web browser. You can use Firefox, but you may run into issues with the security settings. **Launch** the Google Chrome web browser.

21. To connect to Jupyter Notebooks on the **PYNQ** board type **192.168.2.99:9090** in the address bar.

22. The Jupyter Notebook login screen will appear and prompt you for a password. Enter ***xilinx*** for the password and click **Log in**.

23. You should now see all of the files on the **PYNQ** board. This is where all of the Notebooks are stored and can be accessed from. Let's run a demo Notebook to test out the board. Click **base → board → board_btns_leds.ipynb** to open the Buttons and LEDs demo Notebook. All of the Notebooks in the base category are related to the **PYNQ** base overlay.

24. A Notebook is broken up into **cells** that can include Python code or **Markdown** comments and notes. Each **cell** can be executed individually by selecting the **cell** and clicking **Run**. Select the Python code **cell** and then click **Run** to execute the program.

25. The **Markdown cell** describes the functionality of the demo. Make sure that the program operates properly before moving on.

    Note: It is highly recommended to create a copy of any Notebook that you edit

26. Try changing the delay times and the starting color of the RGB LED by editing the variables in the **cell**.

27. For more information on **PYNQ** Jupyter Notebooks visit ***pynq.readthedocs.io***. This is a great resource for **PYNQ**.

Part II – Jupyter Notebook Applications

A Jupyter Notebook is an interactive computing environment that opens the programmers design capabilities to a variety of useful and powerful tools. Within a Jupyter Notebook, a programmer can run live iPython code, create interactive widgets, and generate "Matlab-like" plots, just to name a few. The notebook server is running on the ARM processor of the ZYNQ chip on the PYNQ board, with each running notebook associated to one kernel. In this part, you will gain a better understanding on how the notebooks work by re-creating the *Let's Make a Deal* game that you did back in lab 4. However, this time you will be using iPython and Jupyter Notebooks to design the game.

1. Navigate to **Base → Board** in the Jupyter Notebook file explorer and create a new notebook as shown in Figure below.

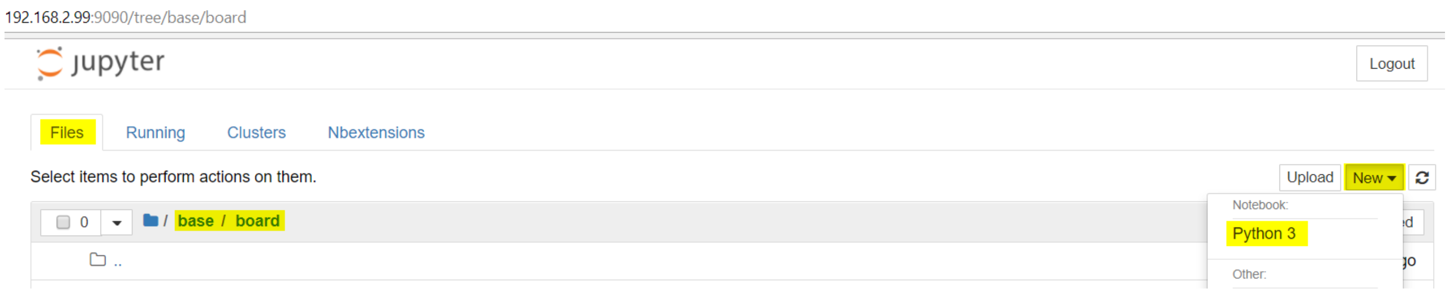


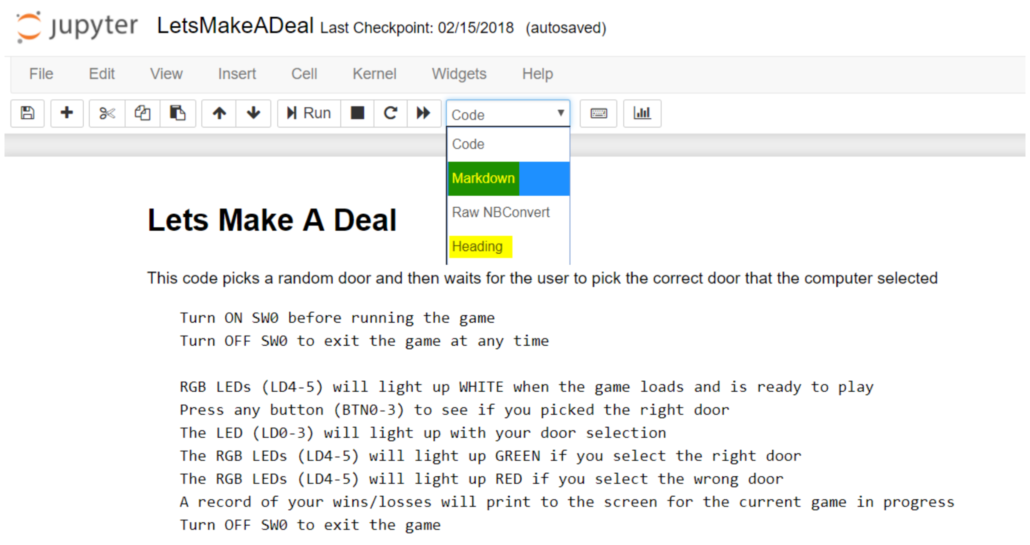
Figure 9: Creating a new notebook

2. Rename the notebook ***Rand_game*** by clicking **File → Rename…**

3. Using the Part I demo program as an example, re-create the *Let's Make a Deal* game from lab 4.

4. Open the **PYNQ → lib** folder in the **PYNQ** shared network drive. Reference step 19 of Part I if needed. Browse through the Python driver files for the buttons and LEDs (*leds.py, buttons.py, rgbled.*py) to view the classes and functions that are available.

5. Start by creating a proper **Heading** and **Markdown** cell for your notebook. Figure 10 is an example **Heading** and **Markdown** that describes expected behavior of your game. You can click on **+** option to add more cells and for each cell you can select between **Heading**, **Markdown** and **Code** from the drop-down menu.



Lets Make A Deal

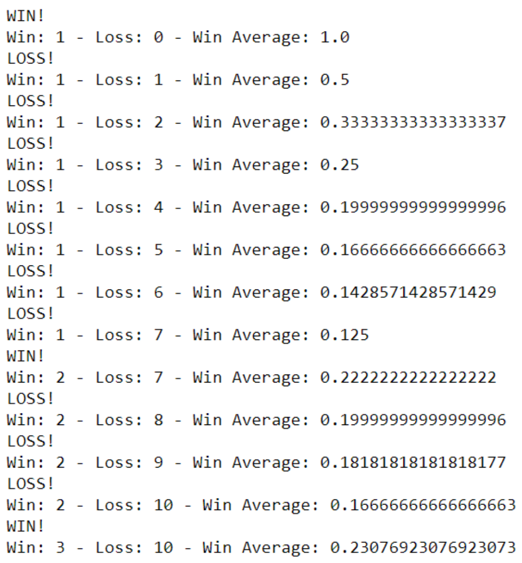
This code picks a random door and then waits for the user to pick the correct door that the computer selected

```
Turn ON SW0 before running the game
Turn OFF SW0 to exit the game at any time

RGB LEDs (LD4-5) will light up WHITE when the game loads and is ready to play
Press any button (BTN0-3) to see if you picked the right door
The LED (LD0-3) will light up with your door selection
The RGB LEDs (LD4-5) will light up GREEN if you select the right door
The RGB LEDs (LD4-5) will light up RED if you select the wrong door
A record of your wins/losses will print to the screen for the current game in progress
Turn OFF SW0 to exit the game
```

Figure 10: Expected game behavior

6. The notebook output of your game should be similar to figure 11. Play your game for a while and note the win percentage of your game. If you used the **Rand** function, note that it is only a pseudo-random number generator.



```
WIN!
Win: 1 - Loss: 0 - Win Average: 1.0
LOSS!
Win: 1 - Loss: 1 - Win Average: 0.5
LOSS!
Win: 1 - Loss: 2 - Win Average: 0.33333333333333337
LOSS!
Win: 1 - Loss: 3 - Win Average: 0.25
LOSS!
Win: 1 - Loss: 4 - Win Average: 0.19999999999999996
LOSS!
Win: 1 - Loss: 5 - Win Average: 0.16666666666666663
LOSS!
Win: 1 - Loss: 6 - Win Average: 0.1428571428571429
LOSS!
Win: 1 - Loss: 7 - Win Average: 0.125
WIN!
Win: 2 - Loss: 7 - Win Average: 0.2222222222222222
LOSS!
Win: 2 - Loss: 8 - Win Average: 0.19999999999999996
LOSS!
Win: 2 - Loss: 9 - Win Average: 0.18181818181818177
LOSS!
Win: 2 - Loss: 10 - Win Average: 0.16666666666666663
WIN!
Win: 3 - Loss: 10 - Win Average: 0.23076923076923073
```

Figure 11: Notebook output

Part III –Jupyter Notebook GUI using ipywidgets

A Jupyter Notebook is not limited to just text output. By using the **iPywidgets** library, an interactive **GUI** can be created to interact with the I/O of the **PYNQ** board. For more information on ipywidgets, you can refer to the document "***ipywidgets_Userguide.pdf***" available on the course website.

In this part, you will create an interactive **GUI** to communicate with the **PYNQ** board's RGB LEDs and green LEDs. Figure 12 shows an example interactive **GUI** for your reference.

Application behavior

- You should be able to control each green LED individually with an interactive button
- You should be able to control each RGB LED individually with an interactive slider
- Your application should allow you to control the state of any LED at any time without delay

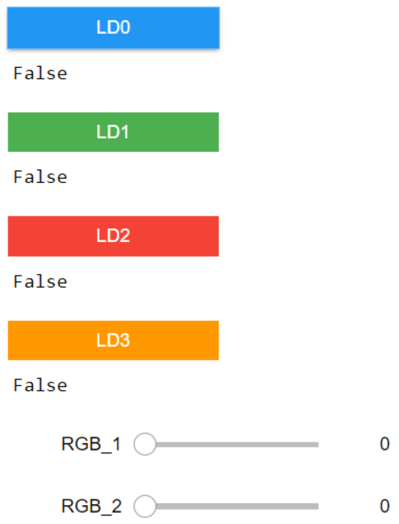


Figure 12: Interactive GUI for PYNQ

<u>For Fun and a little extra credit…</u>

- *In addition to the above GUI, create a menu that allows a single or multiple selection of all the LEDs. Then create another slider that allows you to adjust the blink rate of the selected LED while still allowing full control from the buttons and RGB sliders previously created. For example: if the flash feature is enabled for the RGB LED, you should still be able to use the slider to change its color while maintaining the flash functionality.*

<u>Laboratory Deliverables</u>

You are required to turn in a hard copy of the report. Report should have the following items:

- Cover page with one page description of your design for Parts II and III
- Code printouts (with comments) for Parts II and III

You also have to demonstrate your design and turn in a zipped version of the project