WES 237A: Introduction to Embedded System Design (Winter 2026)
Due: 1/11/2026 11:59pm

In order to report and reflect on your WES 237A labs, please complete this Post-Lab report by the end of the weekend by submitting the following 2 parts:

- Upload your lab 1 report composed by a single PDF that includes your in-lab answers to the bolded questions in the Google Doc Lab and your Jupyter Notebook code.
- Answer two short essay-like questions on your Lab experience.

All responses should be submitted to Canvas. Please also be sure to push your code to your git repo as well.

Git Repo Setup

1. Edit your git repo public page to include all of your names, a short bio, and contact emails in the README.md public page. See markdown syntax if needed.

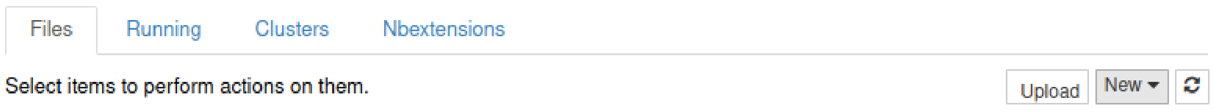
PYNQ Basics

1. Go through the PYNQ Documentation and find the PYNQ Z2 Block Diagram for the Base Overlay
2. **What hardware controls the board peripherals (LEDs, buttons, PMOD headers, etc)?**

Programmable Logic of the Zync-7000 device

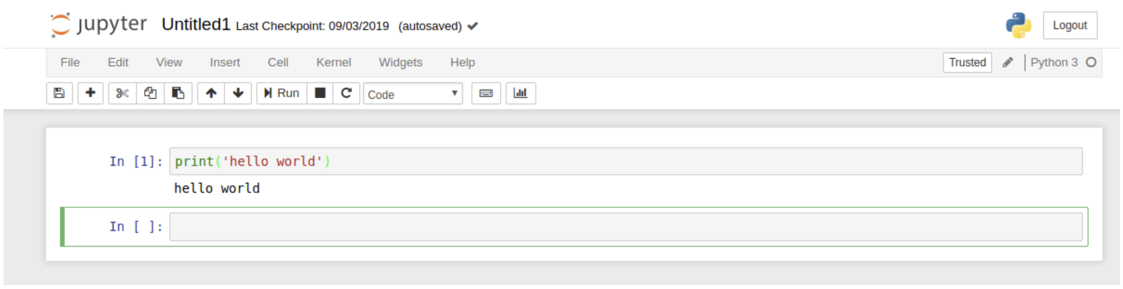
Hello World and LEDs

1. Boot the PYNQ board and connect to your wired private network on 192.168.2.99:9090
2. Select 'New' -> 'Folder'



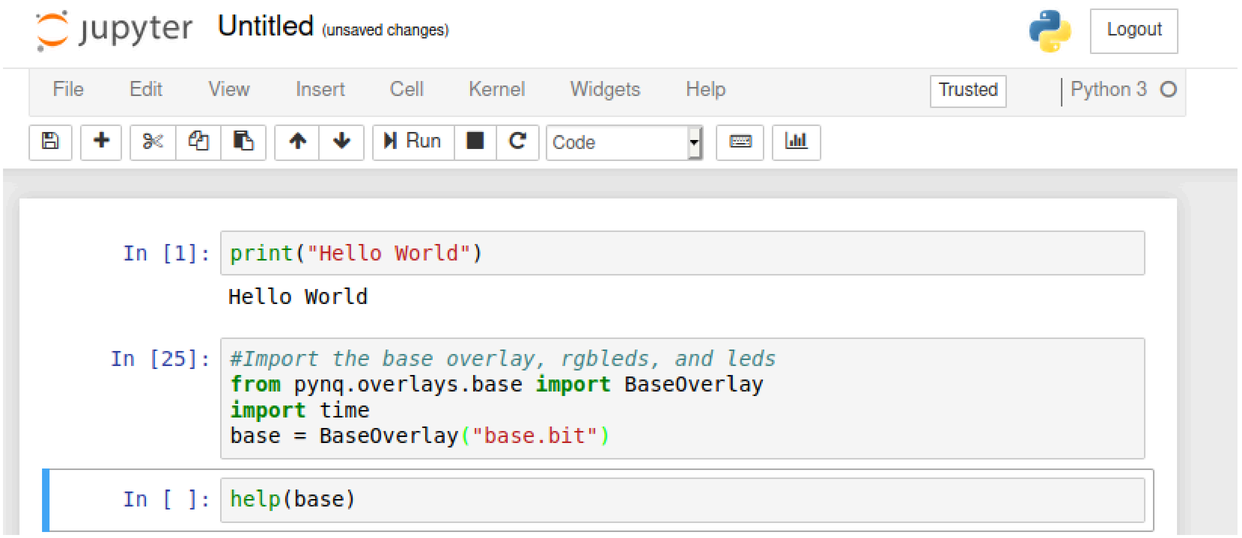
Select items to perform actions on them.

3. Rename the folder to 'Lab1'
4. Go into the folder by double clicking and create a 'New' -> 'Python 3' notebook
5. In the first cell, write 'print("Hello World")'
6. You can run code with the 'Run' button at the top, OR by hitting 'Shift + Enter' at the same time.


jupyter Untitled1 Last Checkpoint: 09/03/2019 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted Python 3
Run Code

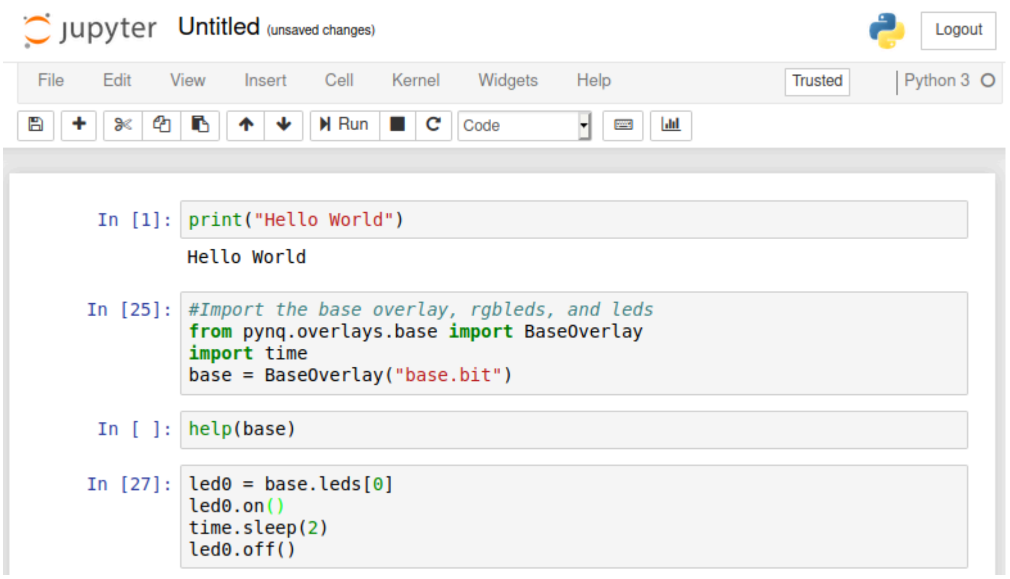
```
In [1]: print('hello world')
        hello world

In [ ]:
```

7. Now let's load the base overlay and access some of LEDs
   a. Import the base overlay and time package with
   ```
   from pynq.overlays.base import BaseOverlay
   import time
   ```
   b. Load the base overlay
   ```
   base = BaseOverlay("base.bit")
   ```
   c. Get the documentation of the base overlay
   ```
   help(base)
   ```


jupyter Untitled (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted Python 3
Run Code

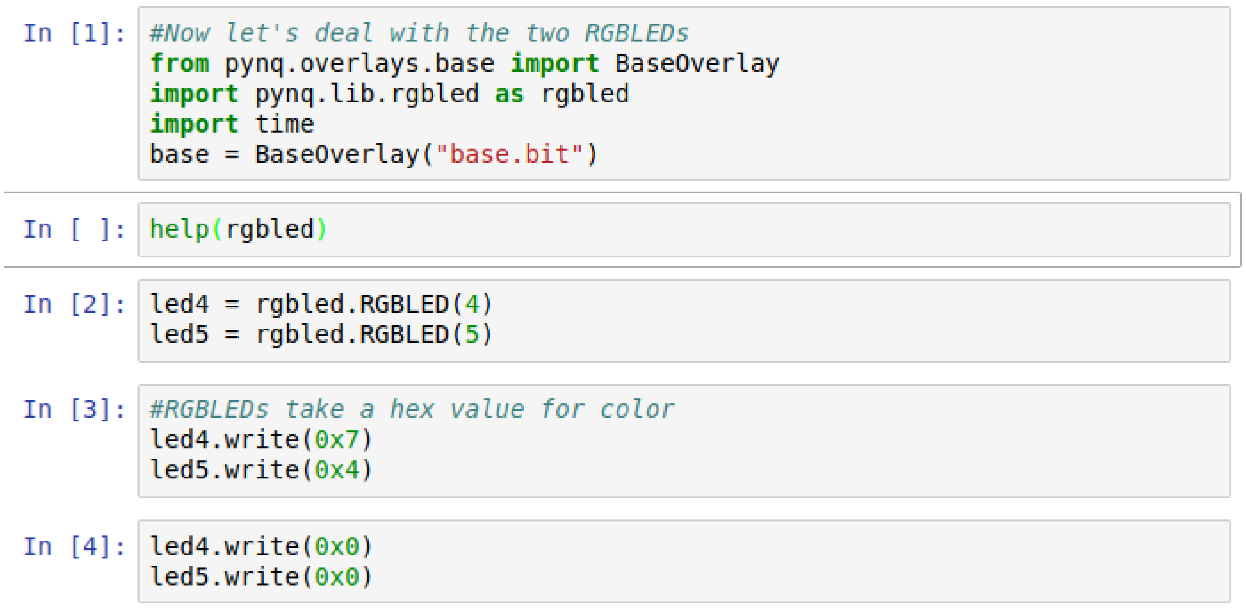
```
In [1]: print("Hello World")
        Hello World

In [25]: #Import the base overlay, rgbleds, and leds
         from pynq.overlays.base import BaseOverlay
         import time
         base = BaseOverlay("base.bit")

In [ ]: help(base)
```

8. Flash the LEDs with an interval of 2 seconds
   ```
   led0 = base.leds[0]
   led0.on()
   time.sleep(2)
   led0.off()
   ```


jupyter Untitled (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

```
In [1]: print("Hello World")

        Hello World

In [25]: #Import the base overlay, rgbleds, and leds
         from pynq.overlays.base import BaseOverlay
         import time
         base = BaseOverlay("base.bit")

In [ ]: help(base)

In [27]: led0 = base.leds[0]
         led0.on()
         time.sleep(2)
         led0.off()
```

9. Now let’s play with the rgb LEDs



```
In [1]: #Now let's deal with the two RGBLEDs
        from pynq.overlays.base import BaseOverlay
        import pynq.lib.rgbled as rgbled
        import time
        base = BaseOverlay("base.bit")

In [ ]: help(rgbled)

In [2]: led4 = rgbled.RGBLED(4)
        led5 = rgbled.RGBLED(5)

In [3]: #RGBLEDs take a hex value for color
        led4.write(0x7)
        led5.write(0x4)

In [4]: led4.write(0x0)
        led5.write(0x0)
```

10. Get a PDF of the jupyter notebook
    a. Go to File->Print Preview then print the print preview page as a PDF
    b. Or try File->Download As->PDF
    c. Only one of the two options needs to work.

ASYNC_IO

1. Download asyncio_example.ipynb from here
2. Upload the asyncio_example.ipynb file to the ‘Lab1’ folder
3. Open the asyncio_example.ipynb
4. Code is organized into ‘cells’. To run the code in a ‘cell’, select the cell and hit ‘Shift + Enter’ at the same time. After running a ‘cell’, you will see [*] which means the code is still executing. Once you see a number in the brackets ([3]), the code has completed.
5. Go through the example code and be able to answer the following with a TA during lab
    a. ***What two lines of code load the FPGA bitstream onto the Programmable Logic (PL) of the PYNQ board?***

```
from pynq import Overlay
ol = Overlay("design.bit")
```

    b. ***Describe in your own words the difference between the ‘looping’ method and the ‘async’ method.***

Looping relies on a single, continuously running control structure.
Async method is based on cooperative multitasking.

6. Write code in the section ‘Lab Work’ to start the LED blinking when ‘button 0’ is pushed and stop when ‘button 1’ is pushed.

GPIO

1. Download gpio_example.ipynb from here
2. Upload the gpio_example.ipynb file to the ‘Lab1’ folder
3. Open the gpio_example.ipynb
4. Go through the example code and be able to answer the following with a TA during lab
    a. ***What is the difference between cells that begin with %%microblaze base.PMODB and cells that don’t?***

Cells that begin with %%microblaze base.PMODB (or %%microblaze base.PMODA) are not ordinary Python notebook cells. They are Jupyter “magic” cells that hand the cell contents to PYNQ’s MicroBlaze subsystem, where the contents are treated as C code, compiled into a small firmware image, and executed on a soft MicroBlaze processor instantiated in the FPGA fabric and wired to that specific PMOD interface.

    b. ***Why do we reload the ‘base’ overlay in the second part of the notebook?***

The base overlay is reloaded in the second part of the notebook to restore a known, clean hardware and software state before instantiating and using a different MicroBlaze subsystem.

5. Write code in the section ‘Lab Work’ to use two pins (0 and 1) for send and two pins (2 and 3) for receive. You should be able to send 2 bits (0~3) over GPIO. You'll need to hardwire from the send pins to the receive pins.
    a. Start the code by copying ‘cells’ 1 and 2 from the beginning of the notebook into the ‘Lab Work’ section.
    b. Then begin editing the %%microblaze cell.