ELEC 4511/5511

Lab 5

Customize your own AXI IP

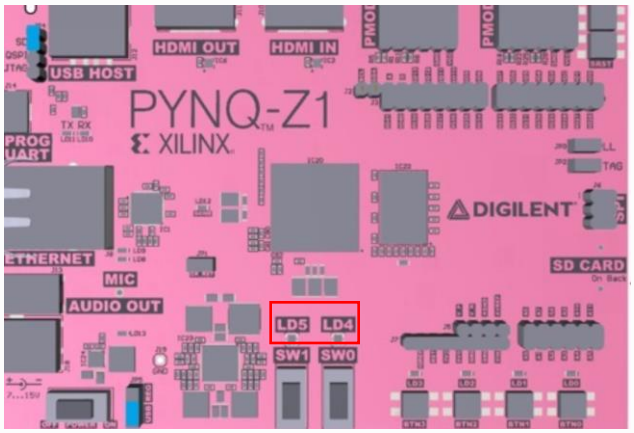
In this lab, you will design a "RGB_LED" control module with AXI interface, and write python program on the PYNQ board to load the hardware overlay to complete the control of the hardware modules by the Processing System.

1. **Experiment goal:**

- Familiar with the process of using Vivado IDE to build hardware IP with AXI interface
- Familiar with using the PYNQ board by loading the system in the SD card
- Understand how the PS and PL systems work separately and how they communicate with each other
- Understand the role of the constraint file

2. **Introduction:**

Lecture18 shows how to customize an IP core with AXI interface to control the LEDs on the PYNQ board. And the steps of this Lab are very similar to that. The only difference is that the IP is not used to control LEDs, but RGB_LEDs, that is, LD4 and LD5 on the PYNQ board. As shown below:



LD4 and LD5 are exactly the same two LEDs that can show different colors. Each LED is controlled by three pins. By outputting high (1) or low (0) voltage to each pin, different states of the LED are displayed. The details as follows:

CLEAR	000
BLUE	001
GREEN	010
CYAN	011
RED	100
MAGENTA	101
YELLOW	110
WHITE	111

In this lab, you can follow Lecture18 and use only one slv_reg0 to control the value of multiple pins; you can also change the design and use multiple slave registers, it all depends on you. But more importantly, you need to fully understand the constraint file. The following is officially provided by Xilinx. You need to make simple modifications according to your design. Simply to say, replace led4_b, led4_g , led4_r , led5_b , led5_g , led5_r with the name of the ports in your design (after # are all comments)

##RGB LEDs

```
set_property -dict { PACKAGE_PIN L15   IOSTANDARD LVCMOS33 } [get_ports { led4_b }]; #IO_L22N_T3_AD7N_35 Sch=led4_b
set_property -dict { PACKAGE_PIN G17   IOSTANDARD LVCMOS33 } [get_ports { led4_g }]; #IO_L16P_T2_35 Sch=led4_g
set_property -dict { PACKAGE_PIN N15   IOSTANDARD LVCMOS33 } [get_ports { led4_r }]; #IO_L21P_T3_DQS_AD14P_35 Sch=led4_r
set_property -dict { PACKAGE_PIN G14   IOSTANDARD LVCMOS33 } [get_ports { led5_b }]; #IO_0_35 Sch=led5_b
set_property -dict { PACKAGE_PIN L14   IOSTANDARD LVCMOS33 } [get_ports { led5_g }]; #IO_L22P_T3_AD7P_35 Sch=led5_g
set_property -dict { PACKAGE_PIN M15   IOSTANDARD LVCMOS33 } [get_ports { led5_r }]; #IO_L23N_T3_35 Sch=led5_r
```

Finally, in the Jupyter notebook of the PYNQ board, you need to write a python program to control the RGB LEDs to display the following patterns:

- Change the color of LD4 every one second in the order of red, green and blue
- Change the color of LD5 every one second in the order of cyan, yellow and magenta
- Requires simultaneous operation of LD4 and LD5

Canvas submission: Please submit your report in "**.pdf**" file format and compress your Vivado project and your python code into a "**.zip**" file.

What needs to be included in your report:

1. Screenshots of the changes you made in the Verilog code when you customize your AXI IP.
2. Screenshot of the changes you made in the constraint file.
3. A short video demo shows LD4 and LD5 changing colors
4. Please write a short analysis of each problem, mainly explaining how you think about the design of the function, what troubles you encountered during the design process, and how you finally solved them, etc.

What needs to be included in your "**.zip**" file:

The folders of your vivado project and IP_repo. And your Python code (Jupyter notebook file)