ECE 4730: Embedded Systems II

Project 1: Hardware PWM in Vivado

Objectives

1. Familiarity with Xilinx FPGA design flow in Vivado 2019.1
2. Experience with Verilog
3. Hardware input and output
4. Pulse-Width-Modulated (PWM) LED
5. Clock Experimentation, Jackpot Game

Deliverables

6.	Verilog code for switches	2
7.	Verilog code for PWM	5
8.	XDC file with clock	3
9.	Verilog code for Jackpot Game	5
10.	lab writeup	4
11.	Instructor/TA-approved Jackpot Game	6
Total:		25

Procedure

Part 1- Creating a new project

1. Create a directory (folder) for your ECE 4730 lab1. Spaces or special characters will cause problems (+,”,’,;,’,’,.., etc.). You may use _ and -.
2. Launch Vivado and select “Create Project”.
3. Select project name and location, select “Create project subdirectory,” then

click “Next”.

4. Select RTL project and leave "Do not specify sources" unchecked. Click "Next”.

5. In the "Add Sources" window, set the following:
   Target language: Verilog
   Simulator language: Mixed.
   Click on the blue "+" button and select "Create file"; a pop-up window will appear.

6. In the popup window, select the following:
   File type: Verilog
   File name: switches
   File Location: '<Local to project>'
   Then click on 'OK' to create the Verilog source file. lick 'Next'

7. The 'Add Constraints (optional)' window appears. We'll do that later; click 'Next'.

8. The ‘Default Part' window appears. We can select the hardware from either the ‘Parts’ or ‘Boards’ tab. The ‘Parts’ tab allows you to design for only the FPGA portion, while the ‘Boards’ tab allows you to design for the FPGA and other built-in hardware portions (such as the Zynq processor). We’ll be working with only the FPGA today, so in the parts tab, select:
   Category: All Package: clg400 Temperature: All
   Family: Zynq-7000 Speed: -1 Static Power: All Remaining
   Select the device with part ID ‘xc7z010clg400-1’ and click ‘Next’, then ‘Finish’.

9. A ‘Define Module’ window appears (* figure <to be added…>). This auto-generates the module header for us. Create the following two ports:

Port Name	**Direction**	**Bus**	**MSB**	**LSB**
SWITCHES	input	yes	3	0
LEDS	output	yes	3	0

Part 2- Coding and Constraints

6. From the ‘Sources’ window, open the created Verilog file (‘.v’ file extension). It will contain a Verilog module with the port declarations described above.

7. Above ‘endmodule’, add the following line of code:
   assign LEDS[3:0] = SWITCHES[3:0];

3. Saving the file performs a ‘Syntax Check’ (errors will be seen as red text in the ‘Messages’ tab); save the file by pressing Ctrl + S or selecting ‘File → Text Editor → Save All Files’.

Part 3- Bitstream and Programming

8. Find prj1.xdc in the folder 'lab1Files' and place it in the directory created above (Part 1, step 1: ECE4730_Lab1).
9. Click the ‘+’ button under ‘Sources’. Select ‘Add or create constraints’ and ‘Next’. Then click ‘Add Files’ and navigate to and select prj1.xdc, select ‘OK’ then ‘Finish’. See that prj1.xdc now shows up in the ‘Constraints’ directory in Vivado.
10. In the left-most section (**Flow Navigator**), click on ‘Generate Bitstream’. Click ‘Yes’ to run synthesis and implementation, then OK in the following window to launch ‘Synthesis and Implementation’. (Continue with step 4. Then verify generation completes in step 5).
11. Verify the top right jumper on the physical FPGA (JP5) is set to JTAG mode (). Then plug the FPGA into the computer using the PROG and UART micro-USB port, flip the switch to ON.
12. The ‘Bitstream Generation Completed’ window will pop up when completed. You may select any of the first three options and click ‘OK’, or simply click ‘Cancel’.
13. Select ‘Open Hardware manager’ (in the ‘Flow navigator’ under ‘Program and Debug’). Select ‘Open Target’ and in the pop up select ‘Open New Target’. Click ‘Next’.
14. Select ‘Local Server’, and ‘Next’. In the next window, select ‘xilinx_tcf’ in ‘Hardware Targets’ and the FPGA (xc7z010_1) in Hardware Devices instead of the arm processor. Click ‘Next’ and then ‘Finish’.
15. Click ‘Program Device’ under ‘**Open Hardware Manager**’ and select the FPGA. Select ‘Program’.
16. The FPGA should now be programmed according to the Verilog file. If the switches turn on the LEDS, then you did it!
17. Good Job!

Part 4- Your Turn

12. What does the .xdc file do?
    a. Extra: What does IOSTANDARD LVCMOS33 mean?
13. What is JTAG mode? Why are we using it?

14. What pin are the user push buttons wired to? Hint: consult the zybo z7-10 xdc file.

15. What is the purpose of an edge detection circuit? How should it have been used in the lab?

16. Create another project called PWM and use the switches to adjust the brightness of an LED.
Hint: You will need a clock for this part. Add the following lines to your .xdc file:

```
#clk_100MHz
set_property PACKAGE_PIN K17 [get_ports clk_100MHz]
set_property IOSTANDARD LVCMOS33 [get_ports clk_100MHz]
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk_100MHz]
```

6. Design a ‘Jackpot game which works as follows: The LEDs glow in a one-hot fashion, which means that the LEDs are turned on one at a time in a sequential manner. Get the transition to happen as fast as you can, while you can still make out which LED is on at a given of time. Assign a DIP switch to each of the LEDs. At any point in time, if you turn on the switch corresponding to the glowing LED, you win a Jackpot and all the LEDs start glowing! Use the reset button to reset the game.