

ECEN 449: Microprocessor System Design
Department of Electrical and Computer Engineering
Texas A&M University

Prof. Sunil P. Khatri

Lab exercise created and tested by:
Kushagra Gupta, Cheng-Yen Lee, Abbas Fairouz, Ramu Endluri, He Zhou,
Andrew Douglass and Sunil P. Khatri

Laboratory Exercise #1

Using Vivado

Jan2024

Objective

The aim of this week's lab exercise is to familiarize you with the Xilinx FPGA design flow via Vivado by stepping through a simple example. We will use Vivado to create hardware, which lights up the LEDs on the ZYBO Z7-10 board depending on the status of the on-board DIP switches. The hardware will be using an FPGA and designed in Vivado using Verilog. After completing the aforementioned example, you will be expected to implement a simple counter and jackpot game on your own with the knowledge gained from the first part of this lab.

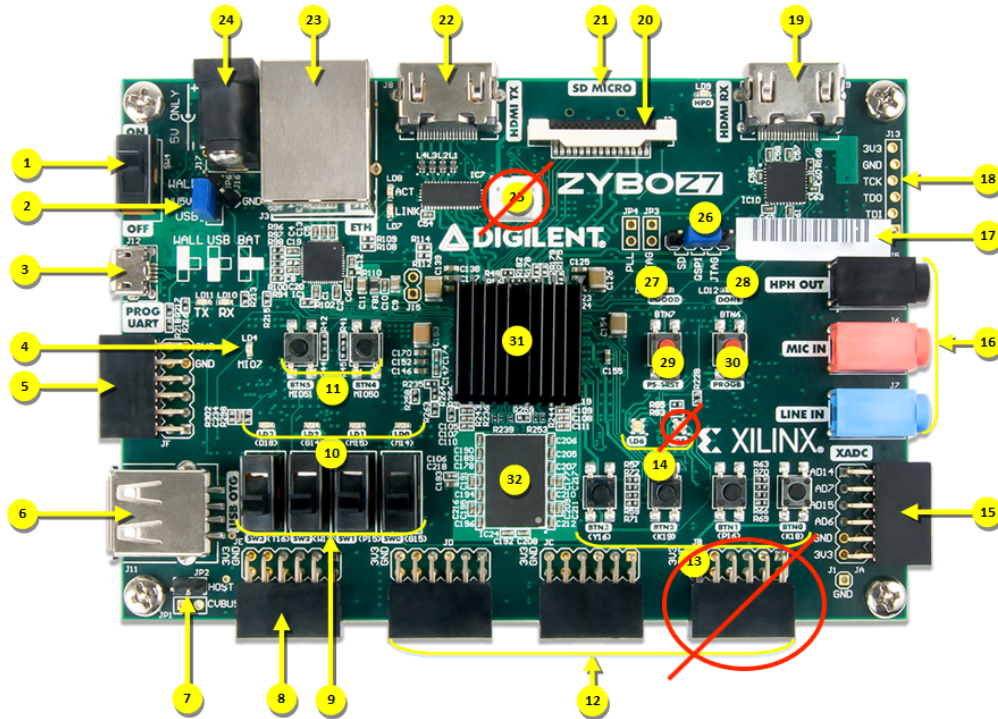


Figure 1: ZYBO Z7 picture callout – *Zybo Z7-20 pictured

Callout	Description	Callout	Description	Callout	Description
1	Power Switch	12	High-speed Pmod ports	23	Ethernet port
2	Power select jumper	13	User buttons	24	External power supply connector
3	USB JTAG/UART port	14	User RGB LEDs	25	Fan connector (5V, three-wire) *
4	MIO User LED	15	XADC Pmod port	26	Programming mode select jumper
5	MIO Pmod port	16	Audio codec ports	27	Power supply good LED
6	USB 2.0 Host/OTG port	17	Unique MAC address label	28	FPGA programming done LED
7	USB Host power enable jumper	18	External JTAG port	29	Processor reset button
8	Standard Pmod port	19	HDMI input port	30	FPGA clear configuration button
9	User switches	20	Pcam MIPI CSI-2 port	31	Zynq-7000
10	User LEDs	21	microSD connector (other side)	32	DDR3L Memory
11	MIO User buttons	22	HDMI output port	* denotes difference between Z7-10 and Z7-20	

Table 1: ZYBO Z7 table callout (Figure 1) – *Zybo Z7-20 pictured

Procedure

1. Create a folder for your ECEN449/ECEN749 lab work
2. Launch the Vivado and create a new design project.

- (a) Open a terminal window in the CentOS workstation and run the following commands:

```
source /opt/coe/Xilinx/Vivado/2022.1/settings64.sh
vivado
```

The first sets up the environment in order to run Vivado and the second command starts the Vivado Suite

Note: it is recommended to add the source command in your 'bashrc' file.

```
vi ~/.bashrc
```

Add the following line at the end of your 'bashrc' file:

```
source /opt/coe/Xilinx/Vivado/2023.1/settings64.sh
```

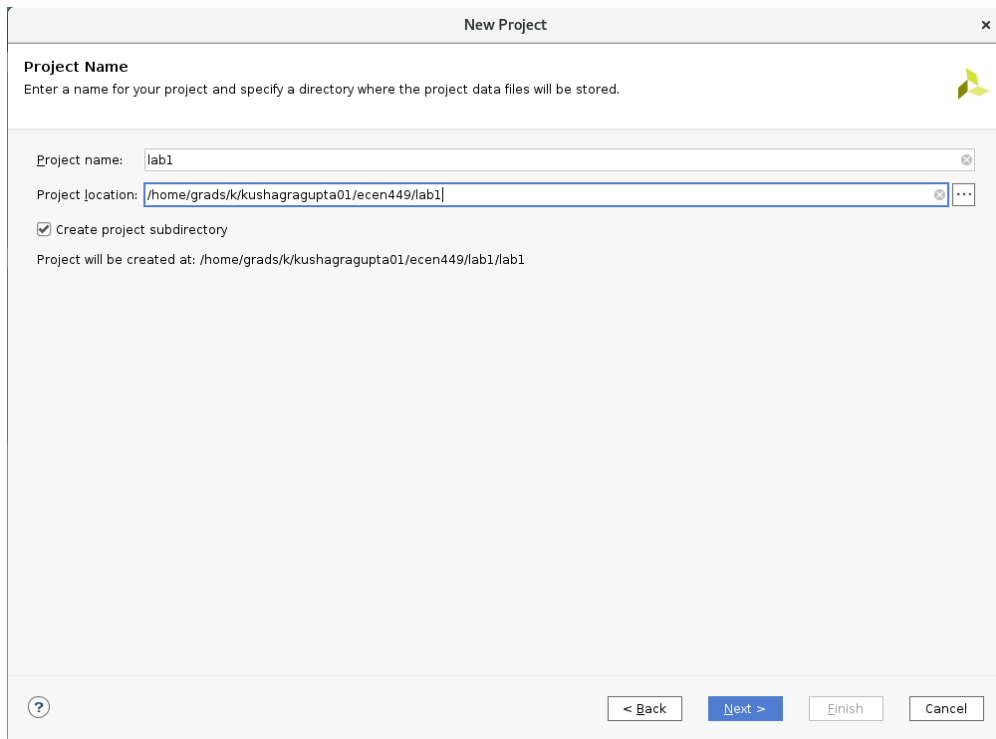
To save the 'bashrc' file, press 'ESC', then type ':wq', then press 'Enter'.

- (b) Once in Vivado, select **File** → **Project** → **New**

The New Project Wizard opens. Click **Next**. (Figure 2).

- Select a Project Name (ex. lab1) and a Project Location (ex. /ecen449/lab1 in your home directory). Then check the create project subdirectory and click Next.
- Select RTL project and leave 'Do not specify sources' unchecked at this time. Click Next.
- You will see 'Add Sources' window, select 'Target language' as Verilog and 'Simulator language' as Mixed.
- Click on the '+' button and select 'Create file', a window pop up will appear.
- Select 'File type' as verilog, 'File name' as 'switch', and select 'File location' as '<local to project>', click 'OK' to create the Verilog source file.
- Click 'Next' and you will see the 'Add Constraints(optional)' window. We will add the Constraints File later in the lab. Click 'Next'.

- (c) Next, the 'Default Part' window appears (Figure 3). We can select our hardware from the 'Parts' tab or from the 'Boards' tab. The 'Parts' tab lists Xilinx supported Parts(FPGAs) and the 'Boards' tab lists the supported boards. The ZYBO Z7-10 (Zynq Board) is an entry-level digital circuit development platform built around the Xilinx Zynq-7000 family, the Z-7010. The Z-7010 is based on the Xilinx All Programmable System-on-Chip (AP SoC) architecture, which integrates a dual-core ARM Cortex-A9 processor with a Xilinx 7-series Field Programmable



New Project

Project Name
Enter a name for your project and specify a directory where the project data files will be stored.

Project name:

Project location:

☒ Create project subdirectory

Project will be created at: /home/grads/k/kushagragupta01/ecen449/lab1/lab1

Figure 2: Create New Project

Gate Array (FPGA) logic. In this lab and the next we will select the hardware from the 'Parts' tab and in the later labs we will select the hardware using the 'Boards' tab. Select the hardware using the following parameters.

Set the device properties to the following:

Family: Zynq-7000

Package: clg400

Speed: -1

You will see four devices. Select the second device with part number 'xc7z010clg400-1' and click 'Next'. Finally, review the information in the 'New Project Summary' window and hit 'Finish' to create project.

Next, the 'Define Module' window appears (Figure 4). This allows us to define the ports for our hardware module. Xilinx will then auto generate part of our source file based on the information

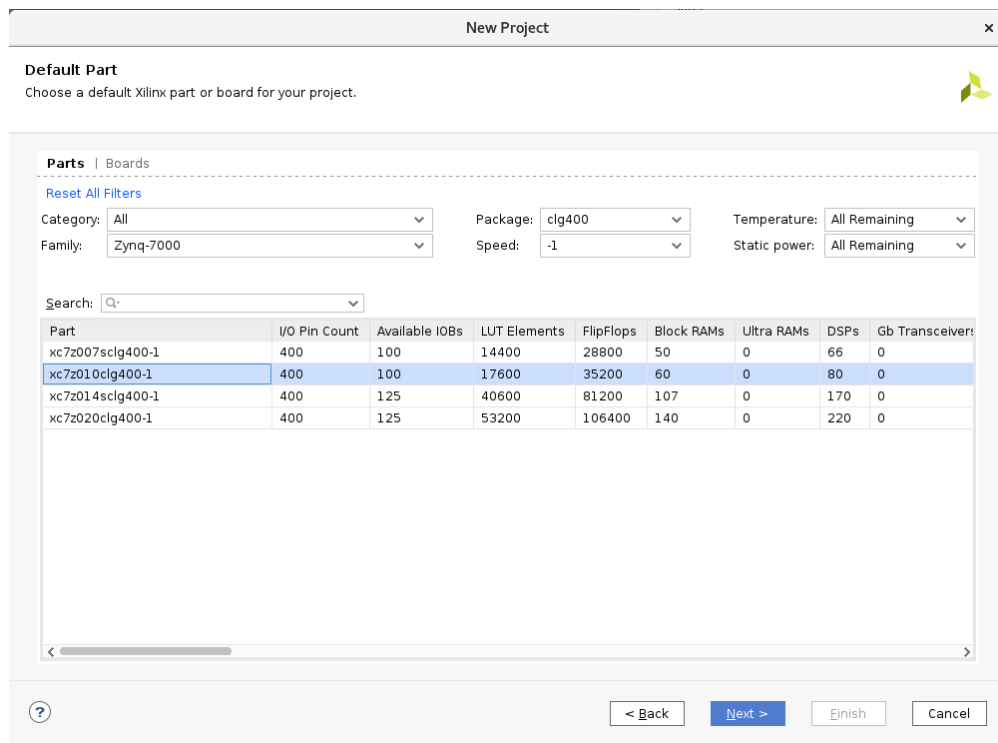


Figure 3: Device Properties

provided. Specify a port called 'SWITCHES'. Set its direction to 'input', check Bus, set the Most Significant Bit (MSB) to 3, and set the Least Significant Bit (LSB) to 0. Specify another port called 'LEDS' and set the direction to 'output', check 'Bus', set MSB to 3, and set LSB to 0. This will create a 4-bit input port, which will connect to the on-board slide switches, and a 4-bit output port, which will connect to the on-board LEDs. Click 'OK'

3. At this point, Vivado has created a new project and source file for us to modify. We will now create code to provide the desired functionality (i.e. to turn on LED[i] when Switch[i] is high).
 - (a) From the 'Sources' window, open the 'switch.v' file. It will contain a Verilog module with the port declarations described in part 2(b).
 - (b) Above 'endmodule', add the following line of code:
`assign LEDS[3:0] = SWITCHES[3:0];`

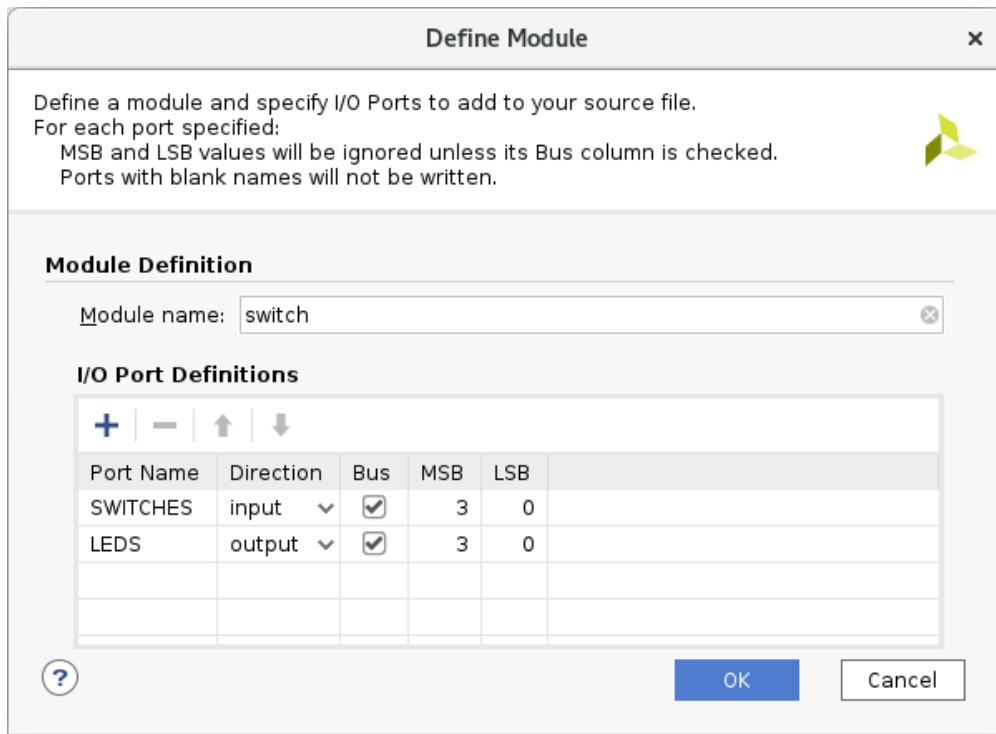


Figure 4: Define Module

The resulting verilog module should be as follows:

```

module switch(SWITCHES,LEDS);
    input [3:0] SWITCHES;
    output [3:0] LEDS;

    assign LEDS[3:0] = SWITCHES[3:0];

endmodule

```

- (c) Click on the save file icon to save your changes. Saving the source file will perform a 'Syntax Check'. When you save the file, if you have made any syntax errors in the source file, Vivado will show error messages corresponding to syntax errors in the messages panel. Please clear the errors and save your source file by pressing **Ctrl+S** or click on the save file icon.
4. We now need to create the 'Xilinx Design Constraints(XDC)' file containing the location of the DIP switches and LEDs on the ZYBO Z7-10 Board. The .xdc file will be used to connect signals described

in the Verilog file (LEDS[3:0] and SWITCHES[3:0] in our case) to pins on the FPGA, which are hardwired to the LEDS and DIP switches on the ZYBO Z7-10 board.

- (a) Use your favorite text editor to create a new file called 'switch.xdc' in your lab1 project directory and copy the following text into the new file:

```
##Switches
```

```
set_property PACKAGE_PIN G15 [get_ports {SWITCHES[0]}]
set_property IOSTANDARD LVC MOS33 [get_ports {SWITCHES[0]}]
```

```
set_property PACKAGE_PIN P15 [get_ports {SWITCHES[1]}]
set_property IOSTANDARD LVC MOS33 [get_ports {SWITCHES[1]}]
```

```
set_property PACKAGE_PIN W13 [get_ports {SWITCHES[2]}]
set_property IOSTANDARD LVC MOS33 [get_ports {SWITCHES[2]}]
```

```
set_property PACKAGE_PIN T16 [get_ports {SWITCHES[3]}]
set_property IOSTANDARD LVC MOS33 [get_ports {SWITCHES[3]}]
```

```
##LEDs
```

```
set_property PACKAGE_PIN M14 [get_ports {LEDS[0]}]
set_property IOSTANDARD LVC MOS33 [get_ports {LEDS[0]}]
```


```
set_property PACKAGE_PIN M15 [get_ports {LEDS[1]}]
set_property IOSTANDARD LVC MOS33 [get_ports {LEDS[1]}]
```

```
set_property PACKAGE_PIN G14 [get_ports {LEDS[2]}]
set_property IOSTANDARD LVC MOS33 [get_ports {LEDS[2]}]
```

```
set_property PACKAGE_PIN D18 [get_ports {LEDS[3]}]
set_property IOSTANDARD LVC MOS33 [get_ports {LEDS[3]}]
```

Note that the above pin assignments were taken from the ZYBO Z7-10 Master Constraint File accessible from the course website.

- (b) After saving 'switch.xdc', return to the Sources panel in vivado, right click on the constraints folder and select 'Add Sources'. Next, the 'Add Sources' window will open. Select 'Add or create constraints' and click 'Next'. Click on the + button and select 'Add files' and navigate to the directory where you saved the constraint file. Select the constraint file and click 'OK'. Click 'Finish' to add the XDC file to your project.
5. At this point, both 'switch.v' and 'switch.xdc' should show up in the 'Sources' window. It is now time to create the hardware configuration for our specific FPGA and download the generated configuration to the ZYBO Z7-10 board.

- (a) Select 'switch.v' in the 'Sources' window. In the 'Flow Navigator' under 'Program and Debug' panel, click on 'Generate Bitstream'. A warning will appear indicating 'No implementation results available'. Click 'Yes' to launch 'Synthesis and Implementation'. This will run all the processes necessary to create a bitstream, which can be downloaded to the FPGA. Running these processes may take several minutes; progress is indicated by the spinning icon and output to the console. You can check the progress in the 'Design Runs' panel located at the bottom of the screen. When a process completes, a  appears next to the appropriate process name. Vivado follows these steps before creating the Bitstream File : synthesize the Verilog, map the result to the FPGA hardware, place the mapped hardware, and route the placed hardware.
 - (b) Once the bitstream generation is completed, we need to download the bitstream to the FPGA on the ZYBO Z7-10 board. Turn on the power to the ZYBO Z7-10 board and make sure that the jumper JP5 is set in 'JTAG' mode. In the 'Flow Navigator' window, under the 'Program and Debug' panel click on 'Open Hardware Manager'. Click on 'Open Target' and in the pop up select 'Open New Target' which will open the 'Open New Hardware Target' window. Click 'Next'. select 'Local Server' in the 'Connect to' field. Click 'Next'. Select 'xilinx_tcf' in Hardware Targets and 'xc7z010_1' in Hardware Devices as in (Figure 5). The 'arm_dap.0' device is ARM Cortex Processor which is not needed for this lab. Click 'Next' and go through the summary and Click 'Finish'. Click on 'Program Device' under 'Hardware Manager' and select the FPGA 'xc7z010_1'. Click 'Program' to program the FPGA on the ZYBO Z7-10 board.
6. At this point, the FPGA should be programmed to function as described in 'switch.v'. Verify this by toggling the DIP switches 0 through 3 and observe LEDs 0 through 3. Demonstrate your progress to the TA.
7. Implement a 4-bit counter using the LEDs (You do not need the switches for this exercise). The count value should update approximately every 1 second. Use the BTN0 and BTN1 push buttons on the ZYBO Z7-10 board to control the direction of the count. For example, when the BTN0 button is pressed, the counter should count up. Likewise when the button BTN1 is pressed, the counter should count down. When neither button is pressed, the count should remain the same. Demonstrate this operation to the TA upon completion.

Hints:

- Do not forget to add clock and reset as input pins to your verilog module.
- Skim through the user manual for the ZYBO Z7-10 board to determine the pin assignments for additional signals. The user manual may be found on the course website.

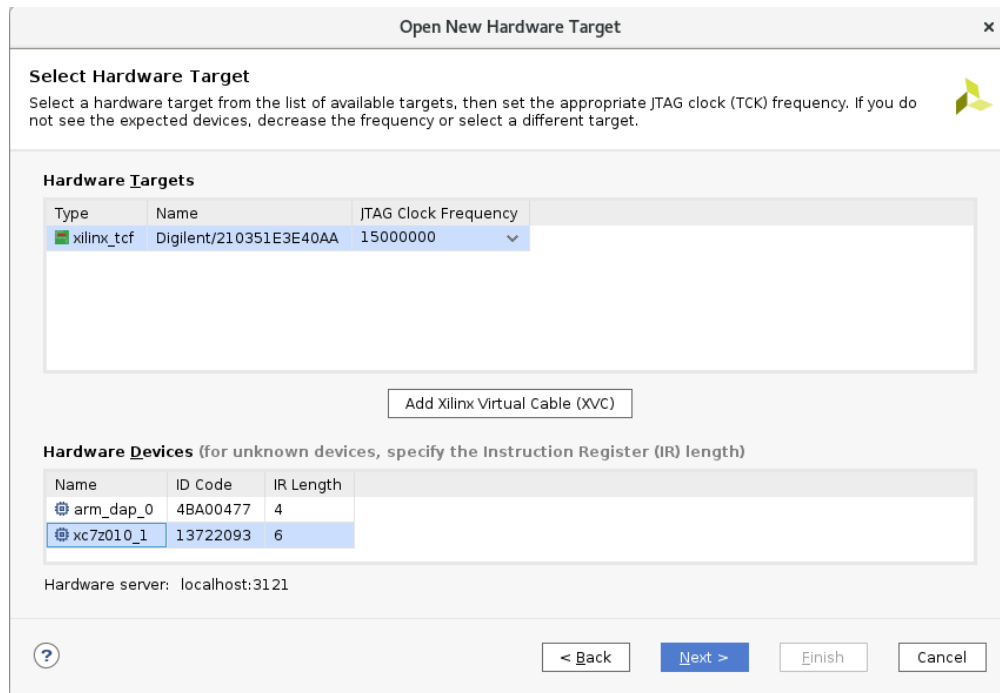


Figure 5: Hardware Target

- After modifying the XDC to include the new ports (and removing unused ports) append the following text to the XDC:

```
set_property PACKAGE_PIN K17 [get_ports CLOCK]
set_property IOSTANDARD LVCMOS33 [get_ports CLOCK]
```

Note: The above XDC lines provide Vivado with timing constraints necessary to ensure proper design operation and assume your signal for clock is labeled 'CLOCK'.

- The K17 pin is the onboard clock with frequency 125MHz. Updates to the LEDs at this rate will not be visible, and thus, the incoming clock must be divided. Think back to your introductory digital logic class to determine how to divide a clock!
8. 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!

Deliverables

1. [8 points.] Demonstration of working portions of the lab.

Submit a lab report with the following items:

2. [8 points.] Correct format including an Introduction, Procedure, Results, and Conclusion.
3. [4 points.] Answers to the following questions:
 - (a) How are the user push-buttons wired on the ZYBO Z7-10 board (i.e. what pins on the FPGA do each of them correspond to and are the signals pulled up or down)? You will have to consult the Master XDC file for this information.
 - (b) What is the purpose of an edge detection circuit and how should it have been used in this lab?