

Design of Digital Systems: Lab Assignment 0

This assignment provides an introduction to working with VHDL on FPGA. The objective of this assignment is to be able to write VHDL code and program the FPGA with the written VHDL code.

Background

In this lab, we explain how to install and use Xilinx Vivado. Xilinx Vivado is a software program that provides step-by-step process to build a Register-Transfer Level (RTL) design using VHDL and program it on an FPGA. The FPGA board used for all labs is the Nexys 4 DDR which has an Xilinx Artix-7 FPGA .

Installation

Please make sure to follow the instructions below to obtain the free edition of Vivado as the default setup will install a trial paid version with a lot of unnecessary tools. Xilinx Vivado requires around 60 GB of disk space with around 120 GB space during installation, so please make sure to have enough disk space. The process might take several hours, so please make sure to do this as early as possible and not wait for the last day.

1. Go to Xilinx Vivado website and download the latest Vivado version. You might need to create an account. This guide is written for Xilinx Vivado ML 2021.1.
2. Run the downloaded installation program. In the first couple of windows, you will need to enter the account you registered with earlier.
3. Once you reach the **Select Product to Install** window, select Vivado as shown in Figure 1
4. In the **Select Edition to Install** window, select Vivado ML Standard as shown in Figure 2
5. In the **Vivado ML Standard** window, only select Vivado, Artix-7 and cable drivers as shown in Figure 3
6. In the **Select Destination Directory** window, keep the default settings.
7. Start the installation process.

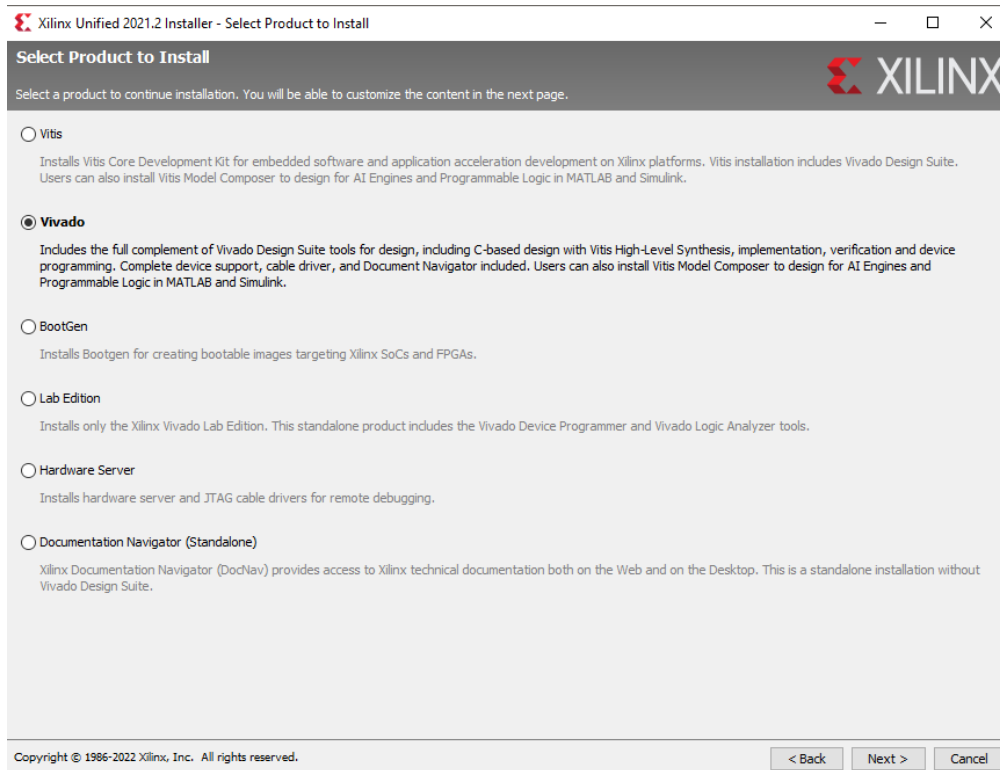


Figure 1: “Select Product to Install” window

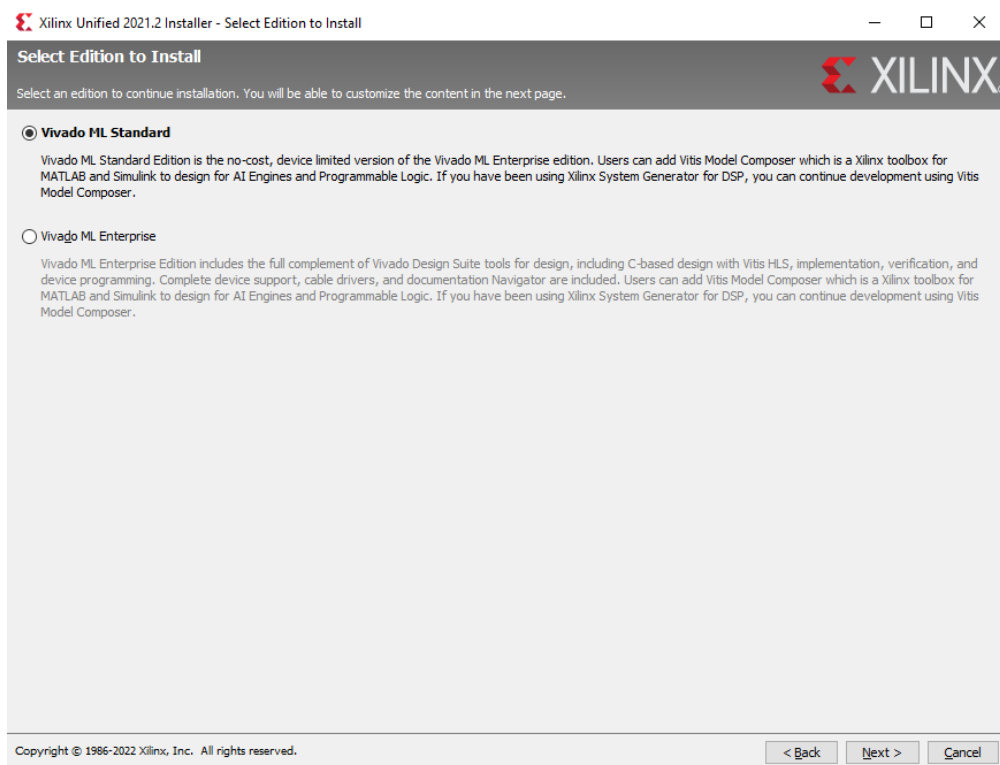


Figure 2: “Select Product to Install” window

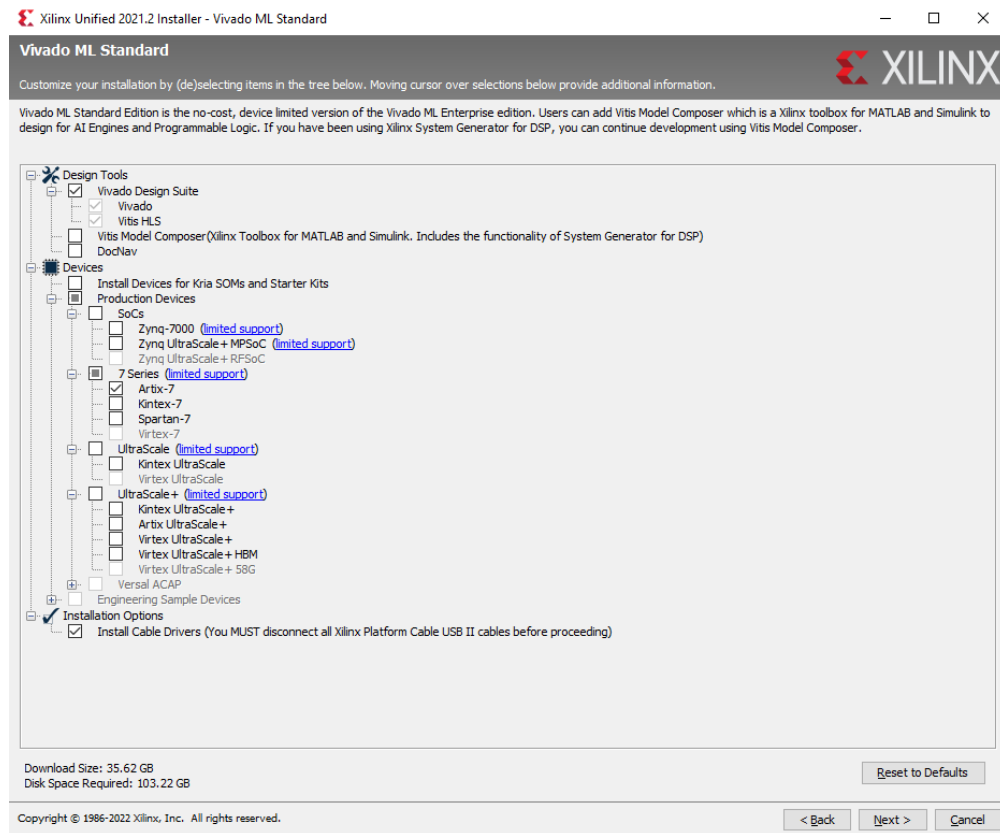


Figure 3: “Select Product to Install” window

Lab Procedure

Once you have completed installing Vivado, you can start building an RTL project. If Vivado requests firewall permission at anytime, then click allow access.

1. Create a **project**.
 - (a) Create a new directory for all your lab assignments.
 - (b) Open Xilinx Vivado.
 - (c) Click on **Create Project**.
 - (d) Set the **project name** to lab0 and **project location** to the directory you created in the first step.
 - (e) Choose **RTL Project** and tick **Do not specify sources at this time**.
 - (f) Search for the Nexys4 FPGA with filters from Table 1.

Table 1: Nexys4 FPGA Info

Family	Atrix7
Package	CSG324
Speed Grade	-2

(g) Choose the part xc7a100tcsg324-2, as shown in Figure 4.

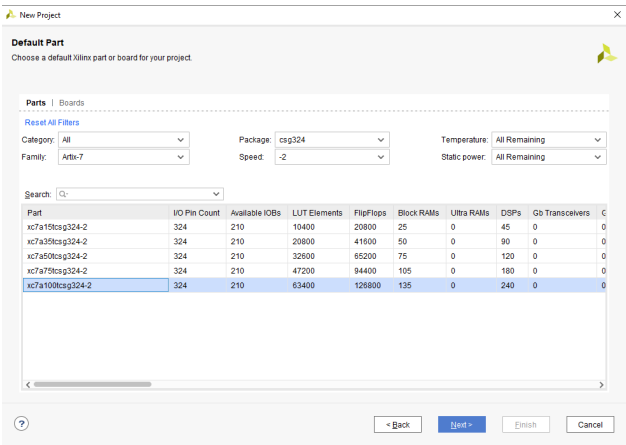


Figure 4: FPGA parts

2. Figure 5 provides a general **overview** of the windows you will see in Vivado

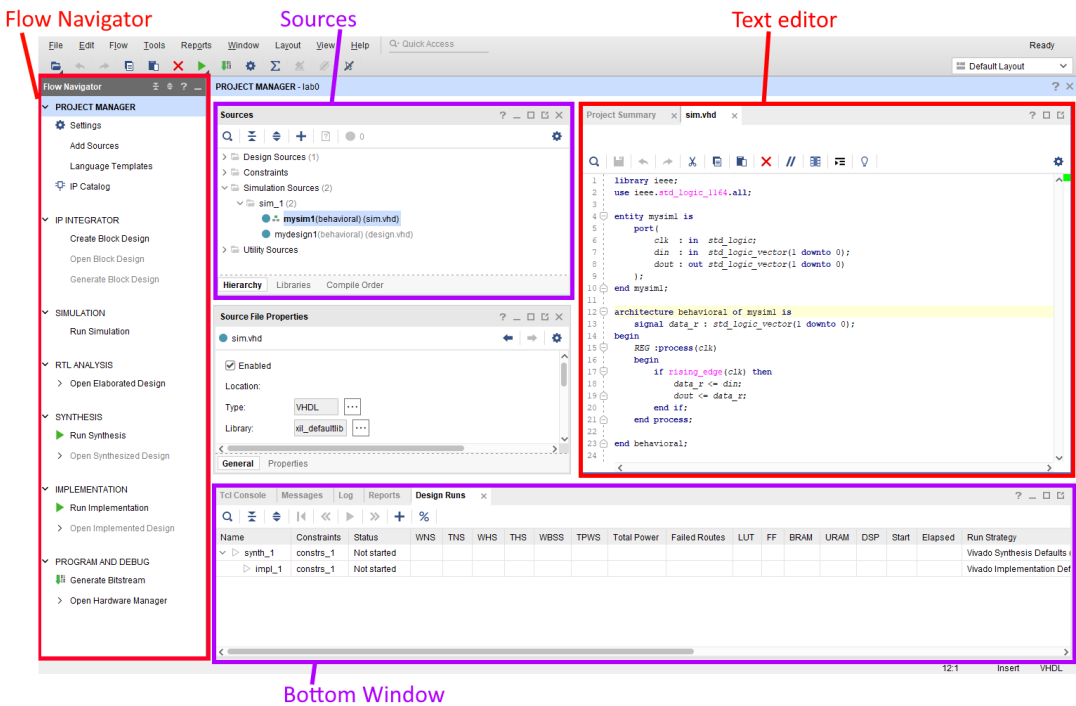


Figure 5: Vivado overview

- (a) The **Flow Navigator** window provides a step-by-step process on how to design your RTL project. The process includes creating sources, simulating the sources, synthesizing and implementing the sources, generating bitstream and flashing the bitstream into the FPGA.
 - (b) The **Sources** window shows your design sources, constraint sources and simulation sources.
 - (c) The text editor window lets you write VHDL code.
 - (d) The bottom window provides additional useful tools, which we will talk about later.
3. **Design sources:** Design sources contains all the code that will be programmed into the FPGA.
- (a) In the **Flow Navigator**, click on **Add Sources**.
 - (b) Click on **Add or Create Design Sources**
 - (c) In the next window, click on **Create File**. Select VHDL as file type and type **design** as filename. Click OK and Next.
 - (d) In the define module window, click Cancel and Yes.
 - (e) Double click on **design.vhd** file from the Sources window
 - (f) Using the text editor, copy the content of **lab0_design.vhd** file into this file.
 - (g) Save the file. You will now see **mydesign1** in the Sources window.
4. **Simulation Sources:** Simulation sources contains all the code that will test the design sources. The code will NOT be programmed into the FPGA.
- (a) In the **Flow Navigator**, click on **Add Sources**.
 - (b) Click on **Add or Create Simulation Sources**
 - (c) In the next window, click on **Create File**. Select VHDL as file type and type **sim** as filename. Click OK and Next.
 - (d) In the define module window, click Cancel and Yes.
 - (e) Double click on **sim.vhd** file from the Sources window
 - (f) Copy the content of **lab0_sim.vhd** file into this file.
 - (g) Save the file. You will now see **mysim1** in the Sources window.
5. **Code explanation:**
- (a) The design source is simply two inputs and two outputs. Each input controls one output. The output updates up to 20ns (2 clock cycles) after the input is changed.
 - (b) The simulation source tests every case in VHDL, a.k.a when the two inputs are 00, 01, 10, or 11.

- (c) Finding coding errors: when coding, it is very common to make mistakes. Figure 6 shows you where you can find errors in your code as early as possible. If you fix all errors and you are still getting errors when running any of the building or testing processes (simulation, synthesis, ...), then you will find the error in the TCL Console window at the bottom.

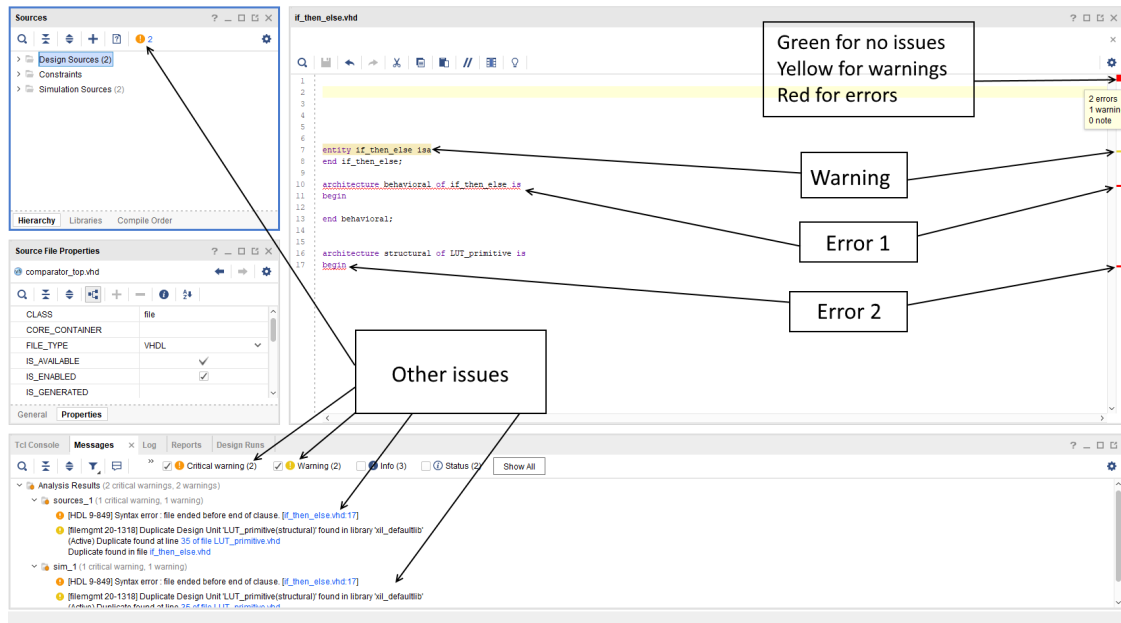


Figure 6: Coding errors in Vivado

6. **Simulation:** Simulation is a very important step to test that your design works before running it on the FPGA.

- In the Sources window, right click on `mysim1` and click **Set as Top**. This will set `mysim1` as the simulation to test.
- Click on **Run Simulation** in the **Flow Navigator**. Then click on **Run Behavioral Simulation**.
- You will now see the waveform window similar to Figure 7. You might need to zoom out (magnifier minus button) to see what is happening properly.
- Right click on `din` and `dout`, then click **Radix→Binary** to see the numbers in binary form.
- As you can see from the waveform, `dout` (output) changes after 20ns from when `din` (input) changes
- Take a screenshot of the waveform for the report.

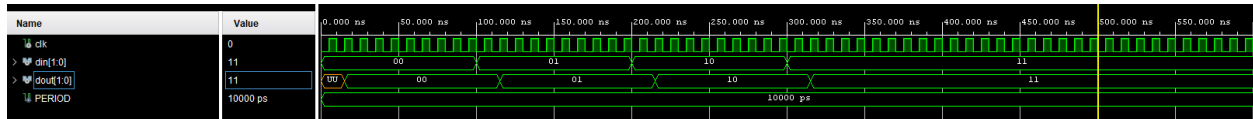


Figure 7: Waveform in Vivado

7. Constraints

- In the Sources window, right click on `mydesign1` and click **Set as Top**. This will set `mydesign1` as the top design. In this project, the button will not work because `mydesign1` is the only source and is already set as top. You might need to do this step in future labs.
- Click on **Open Elaborated Design** from the Flow Navigator. The schematic of the design will show up.
- From the top menu in Vivado, click on **Window→I/O ports**. Set the ports as shown in Figure 8. The ports values are taken from the nexy4ddr reference manual PDF.
- Press `ctrl+s` to save the constraints file. Name the file `constraints`
- If the design uses a `clk`, you will need to set the clock period of the clock. Double click on `constraints.xdc` from the sources window and add the following line which will set the clock with 10ns period.

```
create_clock -period 10.000 -name clk [get_ports clk]
```
- Save the file

Useful tip: To return to the initial view at any time, simply click on the **Project Manager** in the **Project Navigator**

Name	Direction	Neg Diff Pair	Package Pin	Fixed	Bank	I/O Std	Vcco	Vref	Drive Strength	Slew Type	Pull Type	Off-Chip Termination	IN_TERM
▼ All ports (5)													
▼ din (2)	IN			✓	(Multi)	LVCMOS33*	3.300				NONE	▼ NONE	▼
din[1]	IN		L16	✓	14	LVCMOS33*	3.300				NONE	▼ NONE	▼
din[0]	IN		J15	✓	15	LVCMOS33*	3.300				NONE	▼ NONE	▼
▼ dout (2)	OUT			✓	15	LVCMOS33*	3.300		12	▼	▼ NONE	▼ FP_VTT_50	▼
dout[1]	OUT		K15	✓	15	LVCMOS33*	3.300		12	▼	▼ NONE	▼ FP_VTT_50	▼
dout[0]	OUT		H17	✓	15	LVCMOS33*	3.300		12	▼	▼ NONE	▼ FP_VTT_50	▼
▼ Scalar ports (1)													
clk	IN		E3	✓	35	LVCMOS33*	3.300				NONE	▼ NONE	▼

Figure 8: I/O ports in Vivado

8. Synthesis and Implementation

- Click on **Run Implementation**. This will run synthesis followed by implementation.
- Wait until both tasks complete. You can view the progress in the **Design Runs** from the bottom window.

- (c) Once the implementation is complete, a pop-up might show up. Click cancel on it.
- (d) Look at the following values in the **Design Runs** window: **WNS**, **LUT** and **FF**. Check Figure 9 for reference.
- (e) The area results to be reported in the report are Look-up tables (**LUT**) and Flip-flops (**FF**).
- (f) The timing results to be reported in the report are worst negative slack (**WNS**) in **ns**, critical path delay (**CPD**) in **ns** and maximum frequency (**freq**) in **MHz**. The latter two can be computed from **WNS** using

$$\text{CPD} = 10 - \text{WNS}$$

$$\text{freq} = 1000 / \text{CPD}$$

Note: The frequency result from the above formula will be in MHz.

Name	Constraints	Status	WNS	TNS	WHS	THS	WBSS	TPWS	Total Power	Failed Routes	LUT	FF	BRAM	URAM
✓ synth_1	constrs_1	synth_design Complete!									0	4	0	0
✓ impl_1	constrs_1	route_design Complete!	9.134	0.000	0.229	0.000		0.000	0.099	0	0	4	0	0

Figure 9: Implementation results in Vivado

9. Program the board

- (a) From the **Flow Navigator**, click **Generate Bitstream**
- (b) Once completed, open **Hardware Manager**.
- (c) Connect the USB cable between the computer and the board.
- (d) Click on **Open target** and select **Auto Connect**
- (e) Click on **Program Device**.
- (f) Press **Program**
- (g) Test the board by playing with switch 0 and switch 1. Report your observation in the report.