Lab 11

Part B – 64 Bit Two-Level Priority Encoder

Description

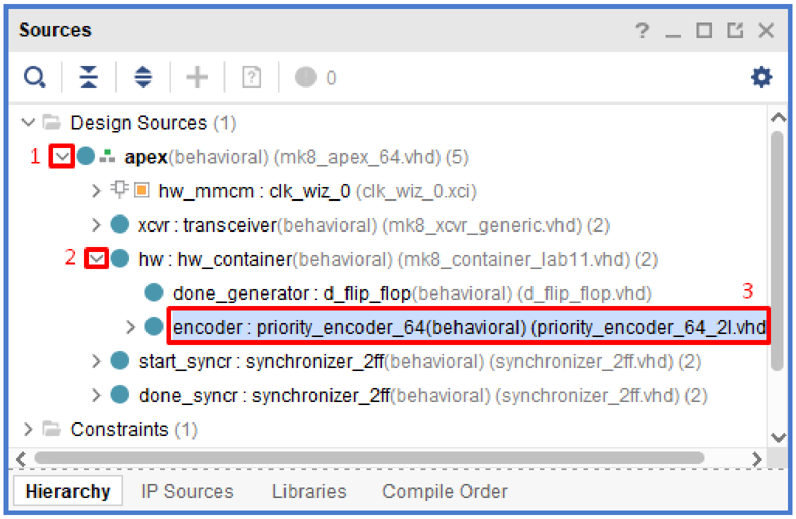
- In this module, you will:
  1. setup a program that can communicate using a serial protocol over USB with the FPGA.
     - You can install this program either on your laptop or a lab computer, but the computer you choose to install on must be able to connect to the FPGA normally with Vivado.
  2. synthesize, implement, upload, and test a hardware description for a two-level priority encoder, encapsulated within a serial transceiver.

Procedure

Project Creation

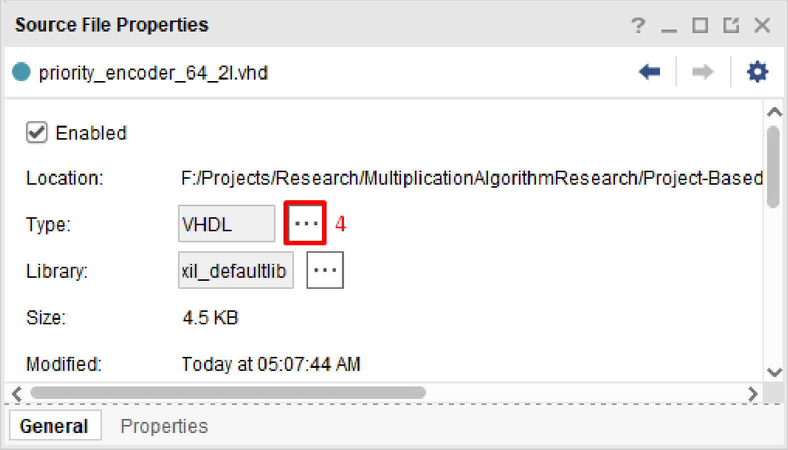
1. Download the files below from Canvas and place them in a new folder titled `lab11b`.
    - Note: You may wish to separate the transceiver files from the 2LPE files as you will re-use the former in Parts C and D.
    - 2LPE Files – for Part B only
        - `mux_generic.vhd`
        - `priority_encoder_8.vhd`
        - `priority_encoder_64_2l.vhd`
    - Transceiver Files – for Parts B, C, and D
        - `basys3_mk8_apex.xdc`
        - `d_flip_flop.vhd`
        - `mk8_apex_64.vhd`
        - `mk8_container_lab11.vhd`
        - `mk8_rx_module.vhd`
        - `mk8_tx_module.vhd`
        - `mk8_xcvr_generic.vhd`
        - `synchronizer_2ff.vhd`
2. Open Vivado and create a new project titled appropriately.
    - If you have not memorized how to do this by now, consult a previous lab handout (ex. 06B).
    - For your convenience, the board identifier is `xc7a35tcpg236-1`.
        - You can copy and paste this directly into the search bar in the device select menu.
        - You can also find the board with the same options as before: `General Purpose`, `Artix-7`, `cpg236`, `-1`.

3. Enable VHDL 2008 for the 2LPE file.
    - To do this graphically:
        - In the `PROJECT MANAGER` pane, in the `Sources` subpane, expand the `apex` top-level component, and then its `hw : hw_container` subcomponent.
        - Click on the entry `encoder : priority_encoder_64`.
        - In the `Source File Properties` subpane below, look for the `Type` property. Click the three dots next to the box containing `VHDL`.



Sources

Design Sources (1)
1 apex(behavioral) (mk8_apex_64.vhd) (5)
hw_mmcm : clk_wiz_0 (clk_wiz_0.xci)
xcvr : transceiver(behavioral) (mk8_xcvr_generic.vhd) (2)
2 hw : hw_container(behavioral) (mk8_container_lab11.vhd) (2)
done_generator : d_flip_flop(behavioral) (d_flip_flop.vhd)
3 encoder : priority_encoder_64(behavioral) (priority_encoder_64_2l.vhd
start_syncr : synchronizer_2ff(behavioral) (synchronizer_2ff.vhd) (2)
done_syncr : synchronizer_2ff(behavioral) (synchronizer_2ff.vhd) (2)
Constraints (1)



Source File Properties

priority_encoder_64_2l.vhd

☑ Enabled

Location: F:/Projects/Research/MultiplicationAlgorithmResearch/Project-Based

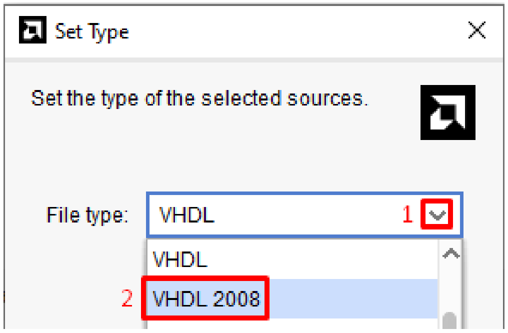
Type: VHDL ··· 4

Library: xil_defaultlib ···

Size: 4.5 KB

Modified: Today at 05:07:44 AM

- In the dialog that opens, expand the dropdown and select `VHDL 2008`.



Set Type

Set the type of the selected sources.

File type: VHDL 1

VHDL
2 VHDL 2008

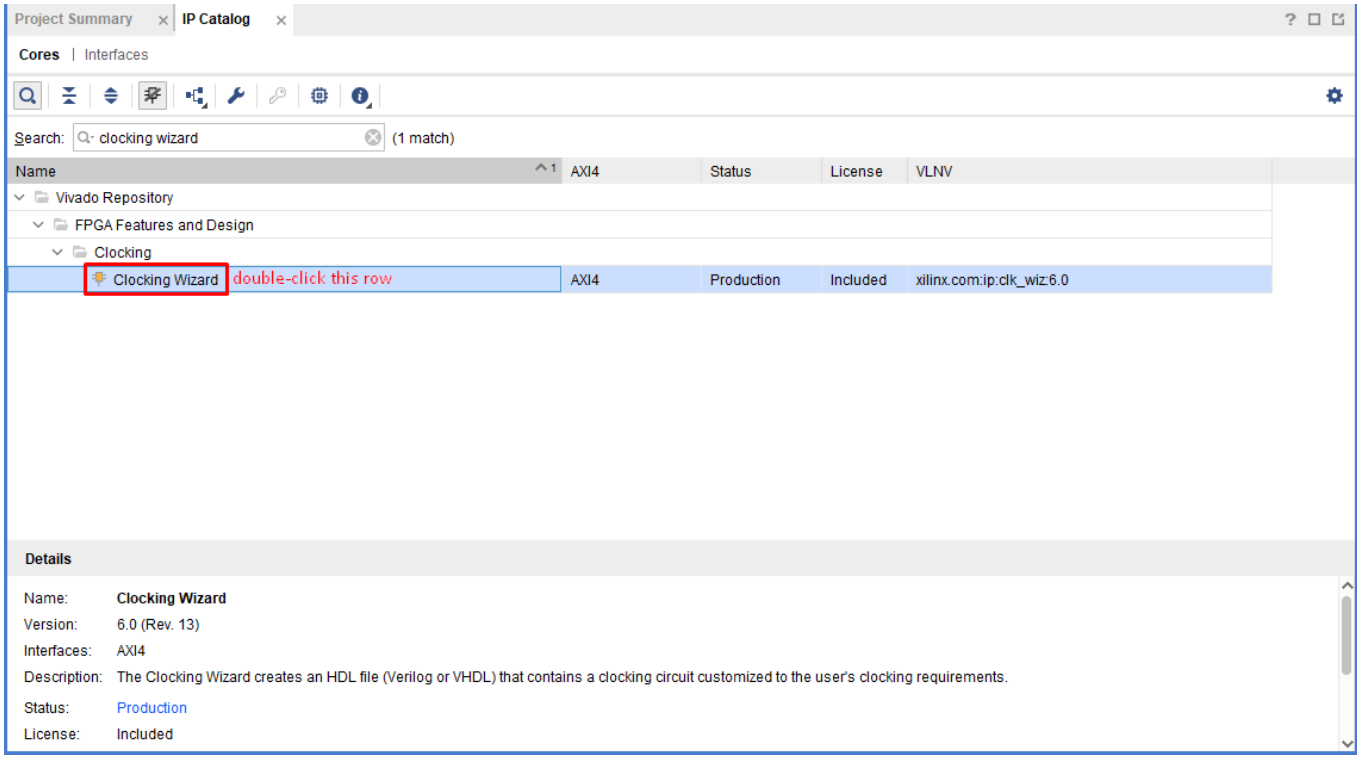
- Click `OK`.
- Important: Repeat for the `hw : hw_container` entry/file.

- Alternatively, you can open the `Tcl Console` subpane at the bottom by selecting the corresponding tab, and enter the following command:

```
set_property file_type {VHDL 2008} [get_files -filter {FILE_TYPE == VHDL}]
```

Adding the Clocking Wizard IP Core

4. On the left sidebar, at the bottom of the `PROJECT MANAGER` dropdown, click on `IP Catalog`.
5. In the search bar, type "clocking wizard", then double-click on the search result for Clocking Wizard.



Search: clocking wizard (1 match)

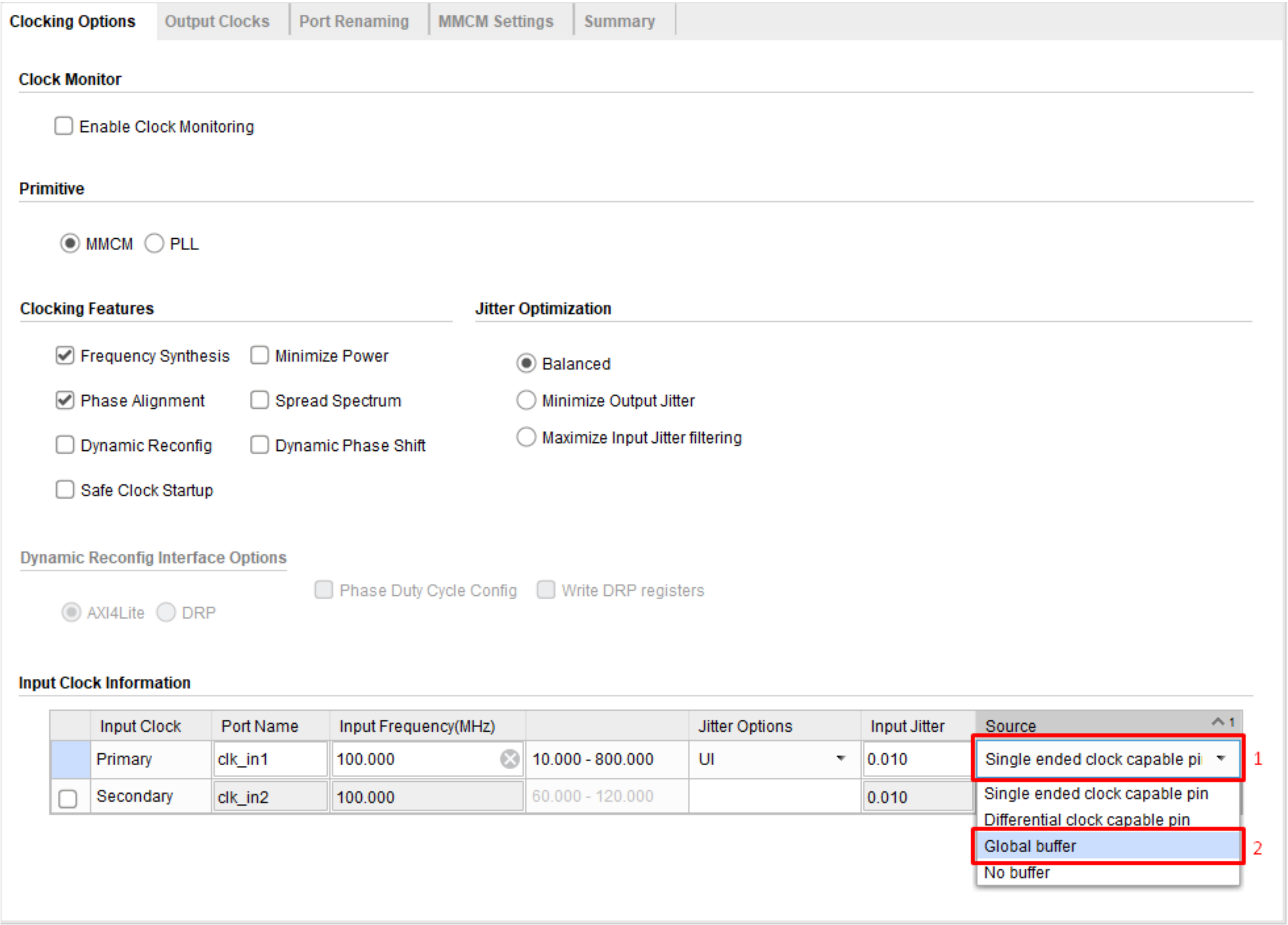
Name	AXI4	Status	License	VLNV
Vivado Repository				
FPGA Features and Design				
Clocking				
Clocking Wizard	AXI4	Production	Included	xilinx.com:ip:clk_wiz:6.0

double-click this row

Details

Name: **Clocking Wizard**
Version: 6.0 (Rev. 13)
Interfaces: AXI4
Description: The Clocking Wizard creates an HDL file (Verilog or VHDL) that contains a clocking circuit customized to the user's clocking requirements.
Status: Production
License: Included

6. In the clocking wizard dialog that opens, at the bottom under `Input Clock Information`, scroll to the right to reveal the `Source` column of the table. For the value in that column in the row for the "Primary" input clock, click the dropdown and switch it to **Global Buffer**.
   - This step is critical, otherwise implementation will fail!



Clock Monitor

□ Enable Clock Monitoring

Primitive

● MMCM ○ PLL

Clocking Features

☑ Frequency Synthesis □ Minimize Power

☑ Phase Alignment □ Spread Spectrum

□ Dynamic Reconfig □ Dynamic Phase Shift

□ Safe Clock Startup

Jitter Optimization

● Balanced

○ Minimize Output Jitter

○ Maximize Input Jitter filtering

Dynamic Reconfig Interface Options

● AXI4Lite ○ DRP

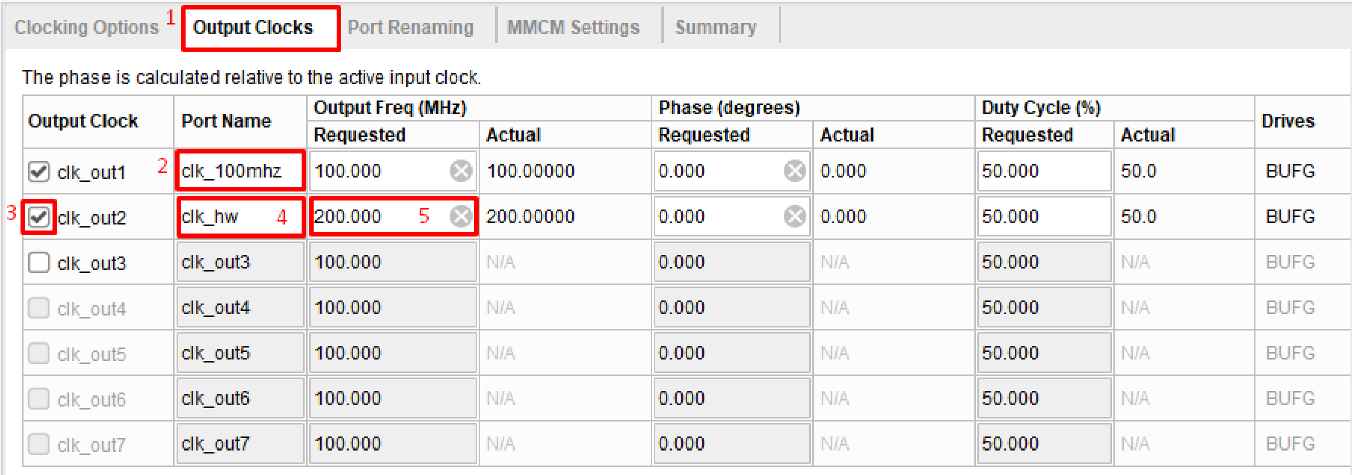
□ Phase Duty Cycle Config □ Write DRP registers

Input Clock Information

	Input Clock	Port Name	Input Frequency(MHz)		Jitter Options	Input Jitter	Source
	Primary	clk_in1	100.000	10.000 - 800.000	UI	0.010	Single ended clock capable pi
□	Secondary	clk_in2	100.000	60.000 - 120.000		0.010	

Single ended clock capable pin
Differential clock capable pin
Global buffer
No buffer
1
2

7. Near the top of the dialog, switch to the Output Clocks tab. Under the `Port Name` column, rename the port `clk_out` to `clk_100mhz`. Check the box in the `Output Clock` column labeled `clk_out2`. Rename the port to `clk_hw` in the `Port Name` column. In the same row, under `Output Freq (MHz)`, change `Requested` to `200.000`.



The phase is calculated relative to the active input clock.

Output Clock	Port Name	Output Freq (MHz)		Phase (degrees)		Duty Cycle (%)		Drives
		Requested	Actual	Requested	Actual	Requested	Actual	
☑ clk_out1	clk_100mhz	100.000	100.00000	0.000	0.000	50.000	50.0	BUFG
☑ clk_out2	clk_hw	200.000	200.00000	0.000	0.000	50.000	50.0	BUFG
□ clk_out3	clk_out3	100.000	N/A	0.000	N/A	50.000	N/A	BUFG
□ clk_out4	clk_out4	100.000	N/A	0.000	N/A	50.000	N/A	BUFG
□ clk_out5	clk_out5	100.000	N/A	0.000	N/A	50.000	N/A	BUFG
□ clk_out6	clk_out6	100.000	N/A	0.000	N/A	50.000	N/A	BUFG
□ clk_out7	clk_out7	100.000	N/A	0.000	N/A	50.000	N/A	BUFG

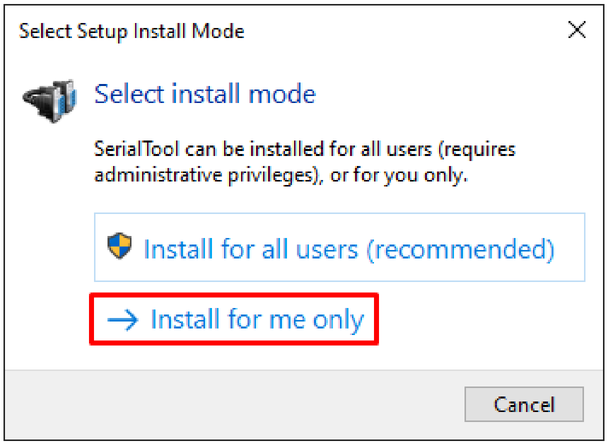
8. Finally, click `OK` at the bottom right. A dialog may appear, if so, click `Generate`.

Hardware Review and Upload

9. Follow along with the instructor to briefly review the hardware.
10. `Open Elaborated Design`, review the schematic alongside the instructor, and take screenshots for your report.
    - Take a screenshot of the main schematic (no components expanded).
    - Expand the `hw` block, and then the `encoder` block inside it. Take a screenshot of just the two-level priority encoder.
11. Run synthesis, implementation, and bitstream generation. Then, open the Hardware Manager.
    - Take a screenshot of the resource utilization (LUTs and FFs) to add to your report.
12. Before proceeding, plug the FPGA development board into your computer using the provided Micro-USB to USB-A cable.
13. At the top, in the green banner (or under the `Open Hardware Manager` dropdown) click `Open Target`, then `Auto Connect`.
14. Once the device is connected, select `Program Device` (in either of the locations where `Open Target` was previously).
    - See 06B-handout for troubleshooting steps, or ask the instructor or TA.
15. Click `Program` to upload the bitstream to the FPGA.

Enabling Serial Communication

16. Install the SerialTool program.
    - Navigate to https://www.serialtool.com/_en/index.php.
    - Click `Download` on the main page. This prompts a donation page.
    - Click `Download` on the donation page. You do not have to donate.
    - Download the appropriate version for your operating system:
        - Windows or Linux – MacOS is supported by SerialTool, but not by Vivado, so you will not be able to use a Mac for this.
    - Run the installer.
        - If running on a lab computer, you will need to install "for me only."

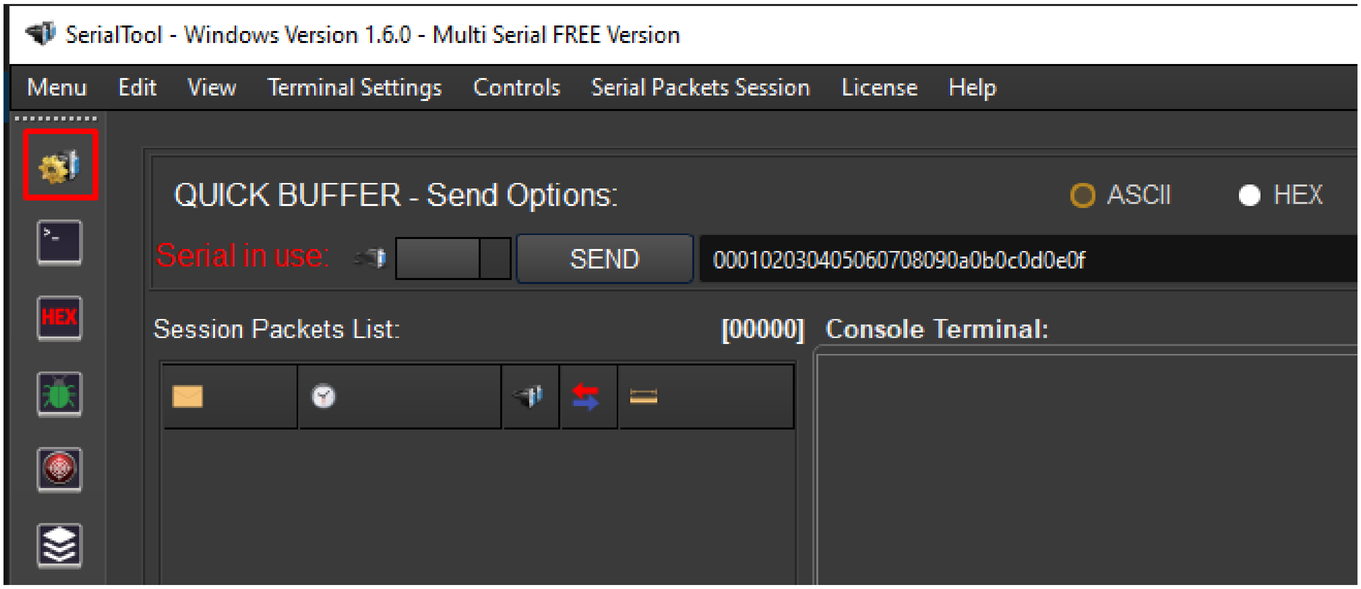

Select Setup Install Mode
Select install mode
Install for all users (recommended)
Install for me only
Cancel

SerialTool can be installed for all users (requires administrative privileges), or for you only.

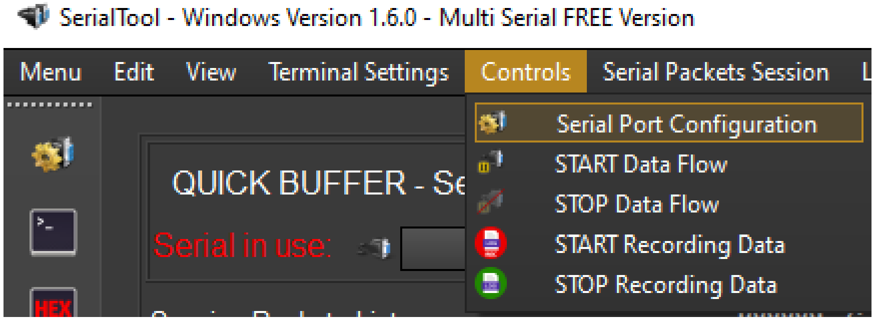
- Other than that, it is fairly straightforward. You can accept all default settings, just keep clicking `Next`, until reaching the final page, and click `Finish`.

17. Run SerialTool.
    - If you just installed it, it may start automatically.
    - You will have to press one of three buttons to start the application.
        - It will tell you which, it is different every time you start it up. This is because the application is freeware.
    - Note that each session (each time you launch the program) is limited to 100 packets. However, this is will not be an issue for the scope of our lab work unless you spam data over the serial connection far beyond what is instructed in this handout.

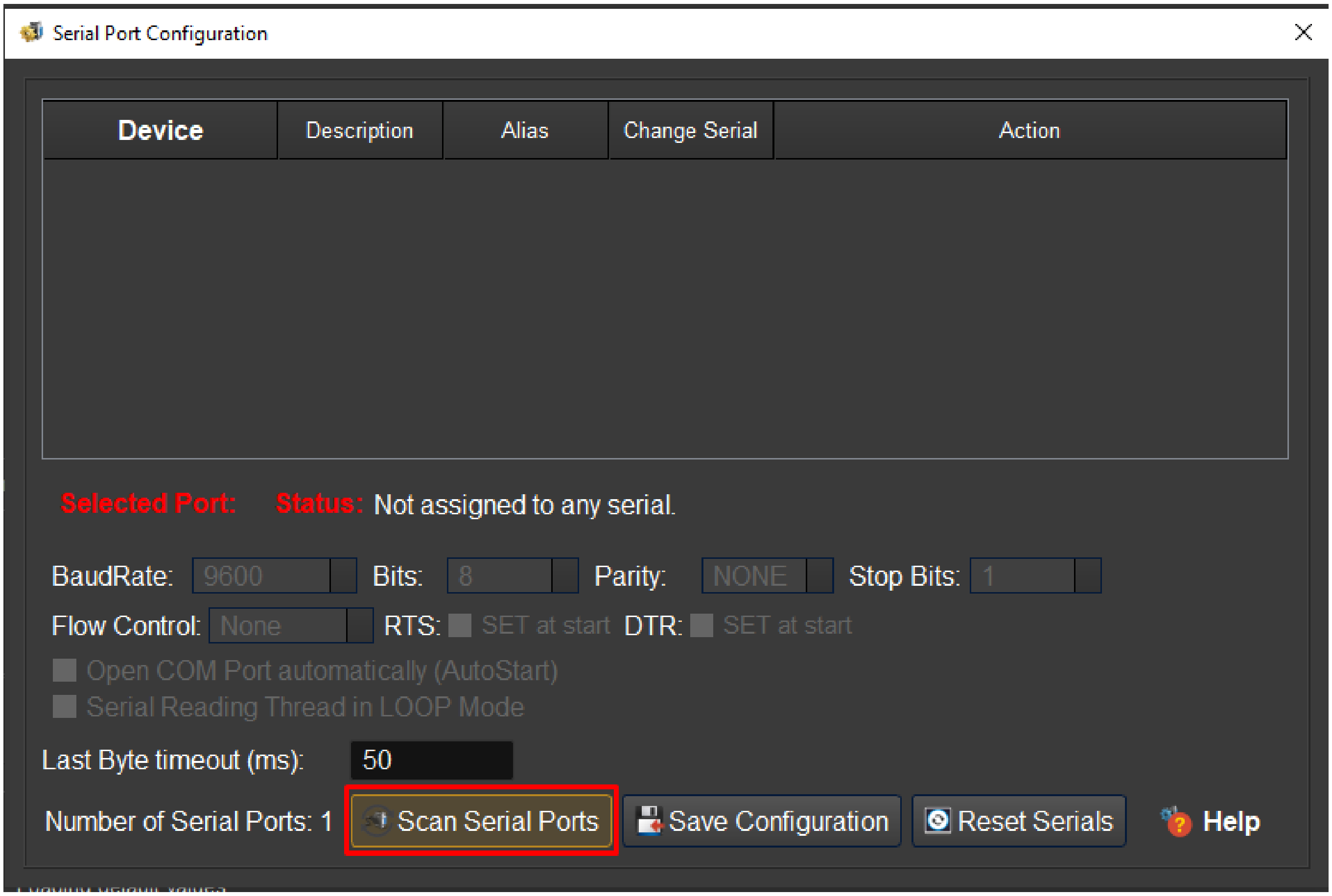
18. Select the first option on the sidebar (a gear with a serial connector).



- Alternatively:
  - Top menu bar: `Controls`
  - First option: `Serial Port Configuration`



19. At the bottom of the dialog that appears, if no ports are already shown, `Scan Serial Ports`.



Serial Port Configuration

Device	Description	Alias	Change Serial	Action

Selected Port: Status: Not assigned to any serial.

BaudRate: 9600 Bits: 8 Parity: NONE Stop Bits: 1

Flow Control: None RTS: ■ SET at start DTR: ■ SET at start

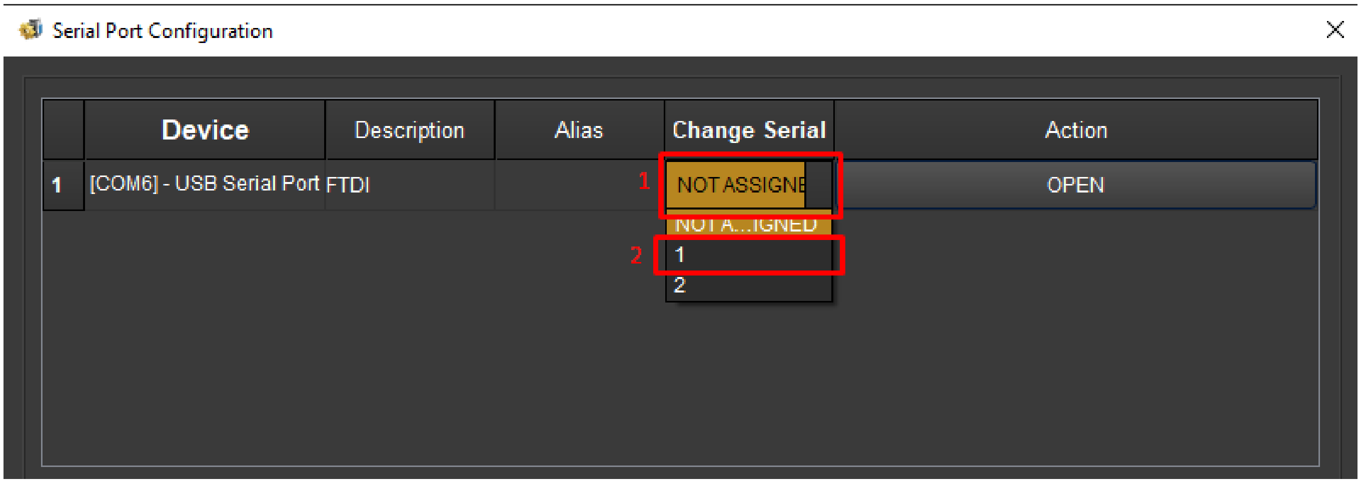
■ Open COM Port automatically (AutoStart)

■ Serial Reading Thread in LOOP Mode

Last Byte timeout (ms): 50

Number of Serial Ports: 1

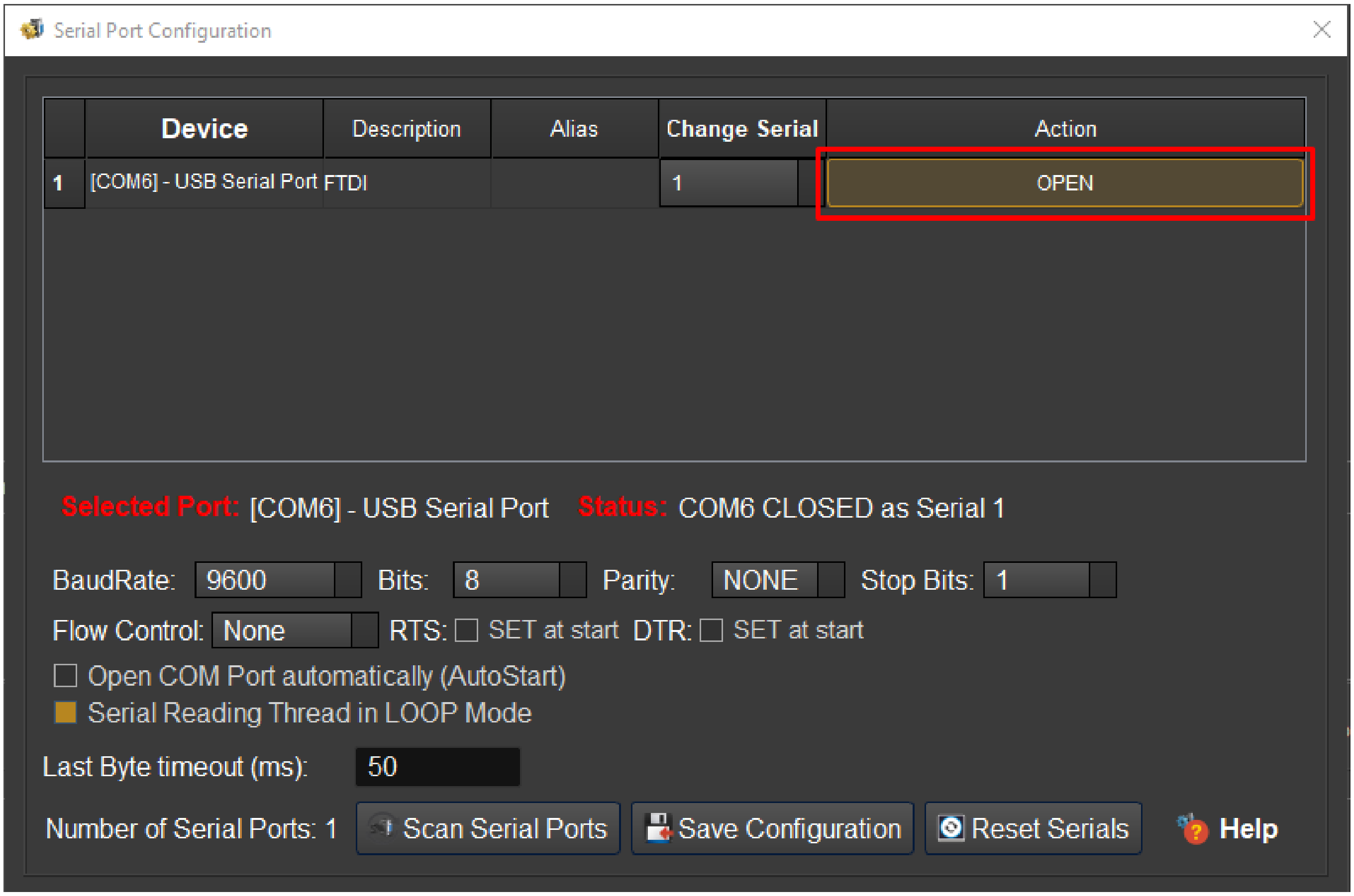
20. A serial port should appear. Click the dropdown that says `NOT ASSIGNED`, then click `1`:



Serial Port Configuration

	Device	Description	Alias	Change Serial	Action
1	[COM6] - USB Serial Port	FTDI		NOT ASSIGNE	OPEN

21. Use the following (mostly default) configuration:
    - BaudRate: 9600
    - Bits: 8
    - Parity: NONE
    - Stop Bits: 1
    - Serial Reading Thread in LOOP Mode: Checked
        - This setting is the only one that is not default, you will need to check this box.
            - If you do not, SerialTool will prompt you to do so after the next step.
22. Click `OPEN` on the port:



Serial Port Configuration

	Device	Description	Alias	Change Serial	Action
1	[COM6] - USB Serial Port	FTDI		1	OPEN

Selected Port: [COM6] - USB Serial Port Status: COM6 CLOSED as Serial 1

BaudRate: 9600 Bits: 8 Parity: NONE Stop Bits: 1

Flow Control: None RTS: □ SET at start DTR: □ SET at start

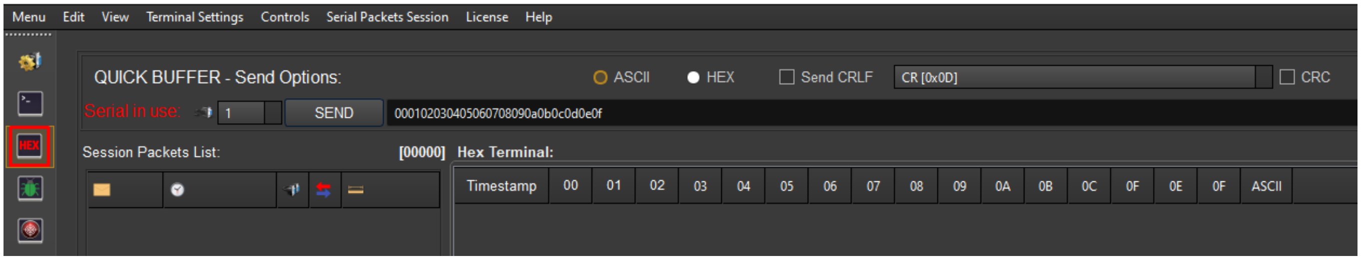
□ Open COM Port automatically (AutoStart)

☑ Serial Reading Thread in LOOP Mode

Last Byte timeout (ms): 50

Number of Serial Ports: 1

23. Close the dialog box (*not* the serial port!) if it does not disappear automatically.
24. On the sidebar, click the `HEX` terminal button, shown below. Your screen should then appear how it does on the right side of the image below.



QUICK BUFFER - Send Options: ○ ASCII ● HEX □ Send CRLF CR [0x0D] □ CRC

Serial in use: 1 SEND 000102030405060708090a0b0c0d0e0f

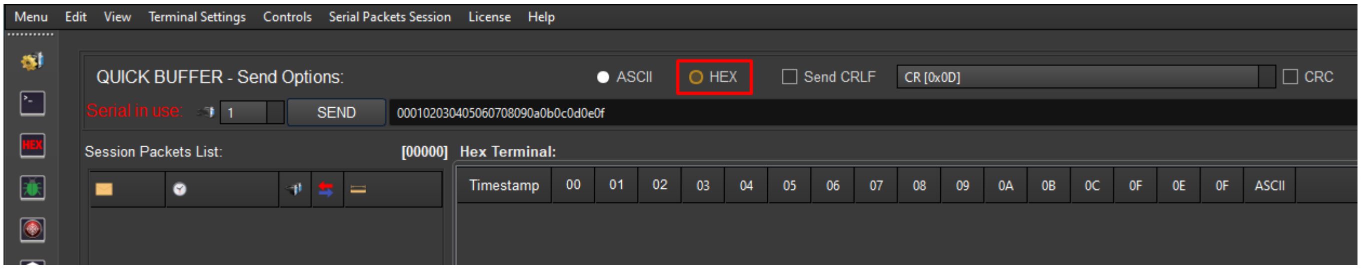
Session Packets List:

[00000] Hex Terminal:

Timestamp	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0F	0E	0F	ASCII

- Alternatively, on the top menu, click `View`, then `Hex Terminal`.

25. Select the `HEX` radio button at the top.



QUICK BUFFER - Send Options: ● ASCII ○ HEX □ Send CRLF CR [0x0D] □ CRC

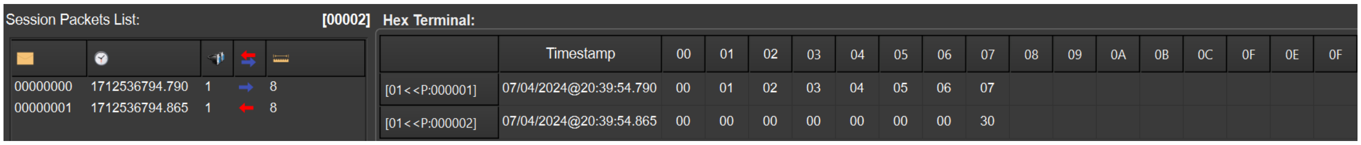
Serial in use: 1 SEND 000102030405060708090a0b0c0d0e0f

Session Packets List: [00000] Hex Terminal:

Timestamp	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0F	0E	0F	ASCII

Hardware Testing and Analysis

26. The text box that is by default filled with `000102030405060708090a0b0c0d0e0f` is what you are sending to the FPGA.
    - Remove the last half, so that what remains is `0001020304050607`. This is 64 bits in hexadecimal.
27. Press `SEND` to send the data to the FPGA. For the two-level priority encoder, you should receive a response that looks like this:



Session Packets List: [00002]

00000000	1712536794.790	1	→	8
00000001	1712536794.865	1	←	8

Hex Terminal:

	Timestamp	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0F	0E	0F
[01<<P:000001]	07/04/2024@20:39:54.790	00	01	02	03	04	05	06	07								
[01<<P:000002]	07/04/2024@20:39:54.865	00	00	00	00	00	00	00	30								

    - How to read this:
        - The pane on the left:
            - The blue is transmission and the red is receipt.
            - The decimal number is the timestamp (in unix time).
            - The column labeled with what looks like a ruler is the transmission length, in bytes.
                - For this module, it will be 8.
        - The pane on the right:
            - This will show both incoming and outgoing data.
28. If entering an input that is not all zeroes, you should also see the leftmost LED light up. This is the `valid` output.
29. Press the center button (`reset`) to reset the transceiver and prepare it for the next transmission.
30. Repeat for each input below (the first is provided again for you). Do your results match the expected output?
    - Note that everything is in hexadecimal.
    - Do not enter spaces into SerialTool, the spaces are there only to help you read the inputs.

Input	Expected Output	Valid LED
`0001 0203 0405 0607`	`30`	`1`
`0000 0000 0000 0000`	`00`	`0`
`0000 0000 0000 0001`	`00`	`1`
`8000 0000 0000 0000`	`3F`	`1`
`0123 4567 89AB CDEF`	`38`	`1`
`0000 0F50 0B20 0A70`	`2B`	`1`

31. Take a screenshot of the SerialTool window after testing all the inputs above, to attach to your lab report.

Deliverables

- Include as part of your **informal report**:
  - A screenshot of the Elaborated Design schematic (Step 10)
  - A screenshot of the FPGA resource utilization (LUTs, FFs) (Step 11)
  - A screenshot of SerialTool containing your testing results (Step 30)
  - A brief comment on whether your experimental results match the expected results.

Outcomes

- Practice working with VHDL.
- Practice using Vivado for hardware synthesis and implementation.
- Practice programming and testing a hardware description on an FPGA development board.
- Understand how to use serial communication to test hardware descriptions.
- Learn how to use a serial communication software to send data to and receive data from an FPGA development board.
- Learn how to add an Intellectual Property (IP) Core to a Vivado project.
- Learn how to use an onboard MMCM to generate a faster clock from the base 100 MHz clock on the FPGA development board.