# Lab 12

## Part C – 64 Bit Two-Level Multiplier

### Description

- In this module, you will finalize the lab work for the project. You will investigate, synthesize, implement, upload, and test the hardware description for a 64-bit two-level multiplier, encapsulated within a serial transceiver.

## Procedure

### Project Creation

- If you already have an existing project from 12B:
  - Add the "2LMR Files" and "New CLA Files" listed in Step 1 below (this should be everything in the 12C folder on Canvas)
  - Set `multiplier_mk2.vhd`, `priority_encoder_64_2l.vhd`, and `barrel_shifter_generic.vhd` to VHDL 2008 as in Step 3.
  - Proceed to Step 9.

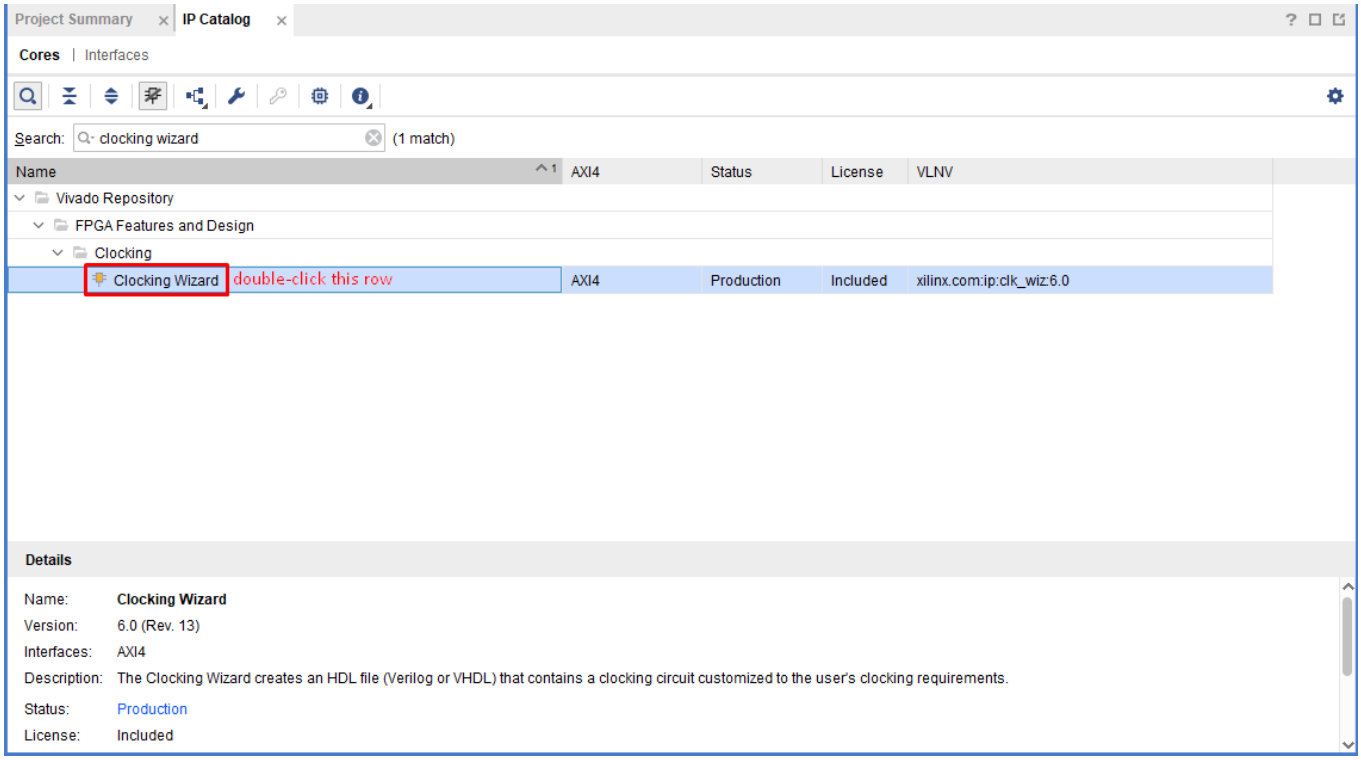
1. Download the files below from Canvas and place them in a new folder titled `lab12b`.
   - 2LMR Files
     - `mux_generic.vhd`
     - `priority_encoder_8.vhd`
     - `priority_encoder_64_2l.vhd` (VHDL 2008)
     - `barrel_shifter_generic.vhd` (VHDL 2008)
     - `decoder_generic.vhd`
     - `multiplier_mk2.vhd` (VHDL 2008)
   - New CLA Files
     - `cla_pow_2.vhd`
     - `cla_half_group_logic.vhd`
   - CLA Files (from Part B)
     - `cla_pow_4.vhd`
     - `cla_group_logic.vhd`
     - `partial_full_adder.vhd`
   - Transceiver Files
     - `basys3_mk8_apex.xdc`
     - `d_flip_flop.vhd`
     - `mk8_apex_128.vhd`
     - `mk8_container_lab12.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 `adder : cla_pow_4` 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 `adder : cla_pow_4`.
        - In the `Source File Properties` subpane below, look for the `Type` property. Click the three dots next to the box containing `VHDL`.
        - In the dialog that opens, expand the dropdown and select `VHDL 2008`.
        - Click `OK`.
    - Repeat for `multiplier_mk2.vhd`, `priority_encoder_64_2l.vhd`, and `barrel_shifter_generic.vhd`.
    - Alternatively, you can open the `Tcl Console` subpane at the bottom by selecting the corresponding tab, and enter the following command:

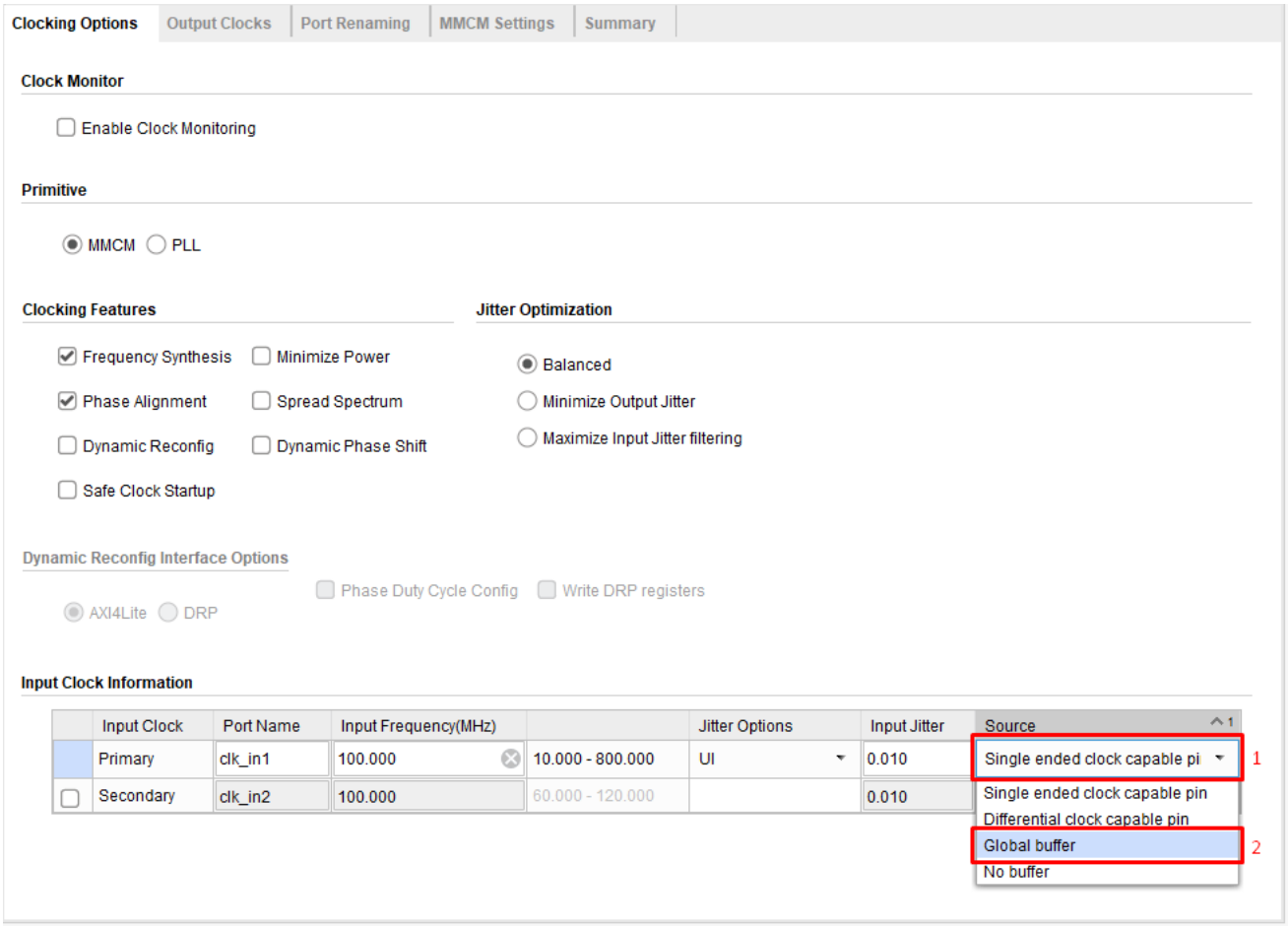
```
set_property file_type {VHDL 2008} [get_files -filter {FILE_TYPE == VHDL}]
```

# Adding the Clocking Wizard IP Core

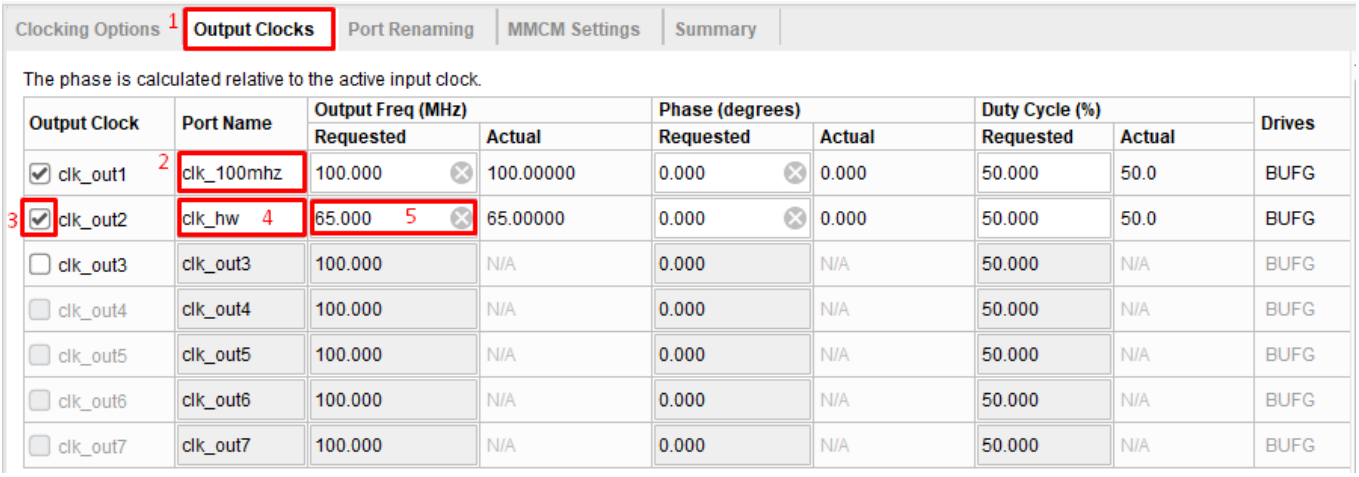
4. On the left sidebar, at the bottom of the `PROJECT MANAGER` dropdown, click on `IP Catalog`.
5. In the search bar, type "clocking wizard", then double-click on the search result for Clocking Wizard.



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!



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 `65.000`.




8. Finally, click `OK` at the bottom right. A dialog may appear, if so, click `Generate`.
    - Note: if you did not properly set up the clock wizard or exited the dialog before setting everything up, do NOT click on the Clocking Wizard entry in the IP catalog again. Instead, double-click the `hw_mmcm : clk_wiz_0` entry in the `Sources` subpane (or alternatively, right-click and select Re-customize IP). This will bring up the correct window for changing the settings on that IP block.

## Hardware Review and Upload

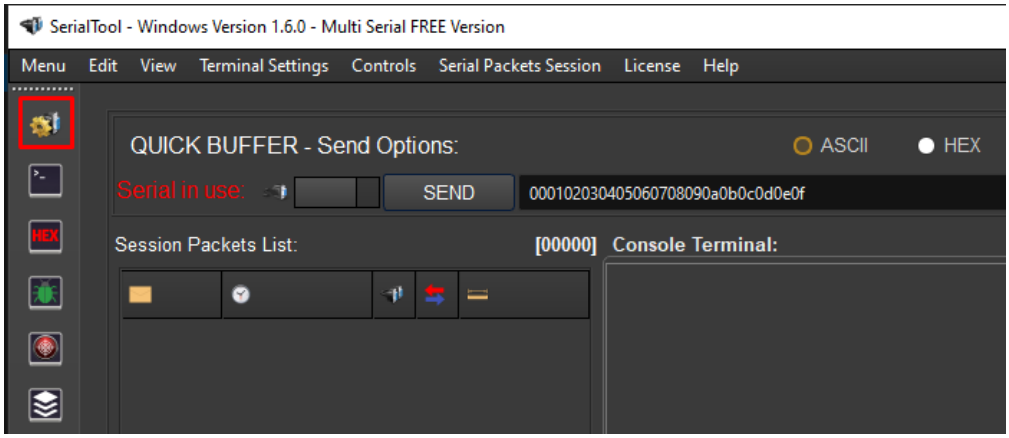
9. Modify `mk8_container_lab12.vhd` as follows:
    - Open `mk8_container_lab12.vhd` by double-clicking on the corresponding entry in the `Sources` subpane of the `PROJECT MANAGER` pane.
    - Comment out lines 187 to 211.
        - Instead of doing this manually, use your mouse to select all rows of one section, then press `CTRL` and `/` at the same time. This will automatically comment each selected line.
    - Uncomment lines 217 to 236.
        - Use `CTRL + /` in the same way to uncomment these lines. This shortcut toggles line comments on the selected lines, so it can be used in either way.
    - Save the file (`CTRL + S`, or hit the save icon at the top of the editor).
10. Follow along with the instructor to briefly review the hardware.

11. `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 `multiplier` block inside it. Take a screenshot of just the multiplier hardware.
12. 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.
13. Before proceeding, plug the FPGA development board into your computer using the provided Micro-USB to USB-A cable.
14. At the top, in the green banner (or under the `Open Hardware Manager` dropdown) click `Open Target`, then `Auto Connect`.
15. 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.
16. Click `Program` to upload the bitstream to the FPGA.

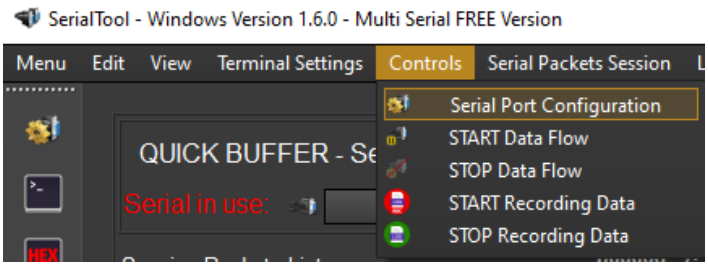
## Enabling Serial Communication

17. Launch SerialTool.
    - 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.

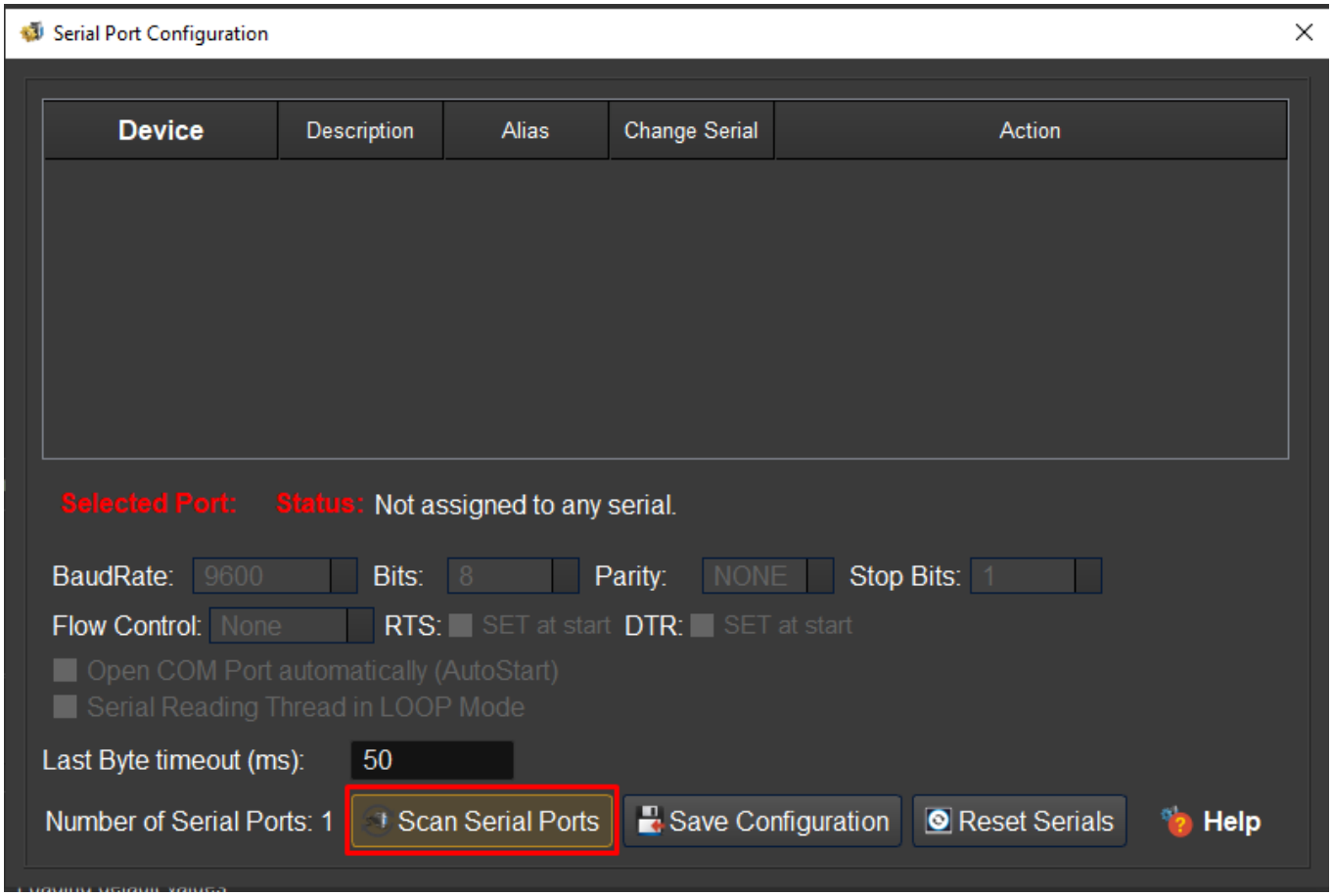
18. Select the first option on the sidebar (a gear with a serial connector).



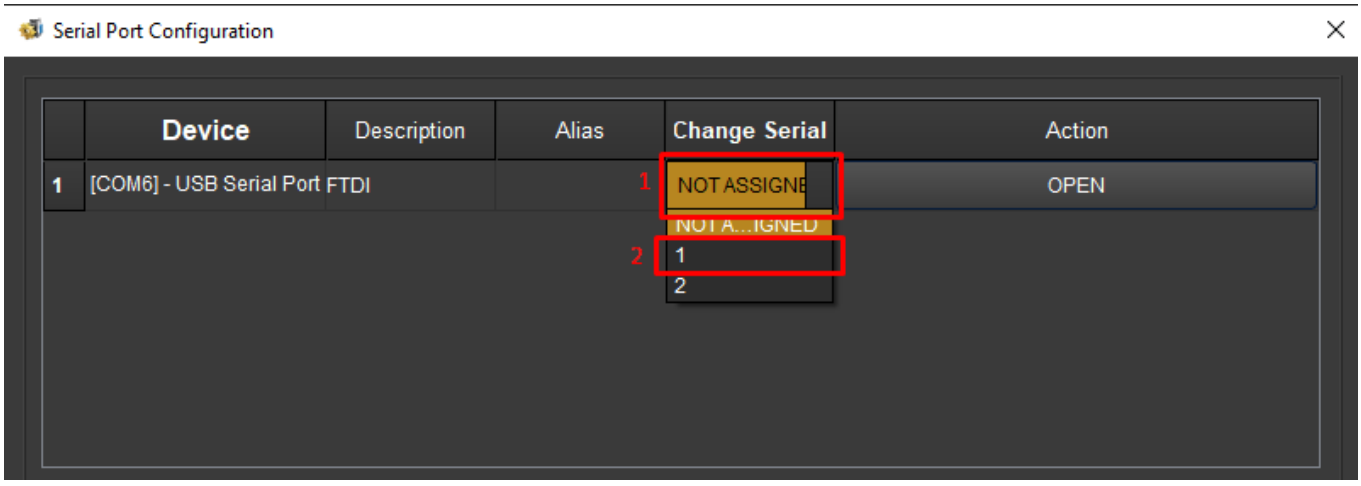

- Alternatively:
  - Top menu bar: `Controls`
  - First option: `Serial Port Configuration`



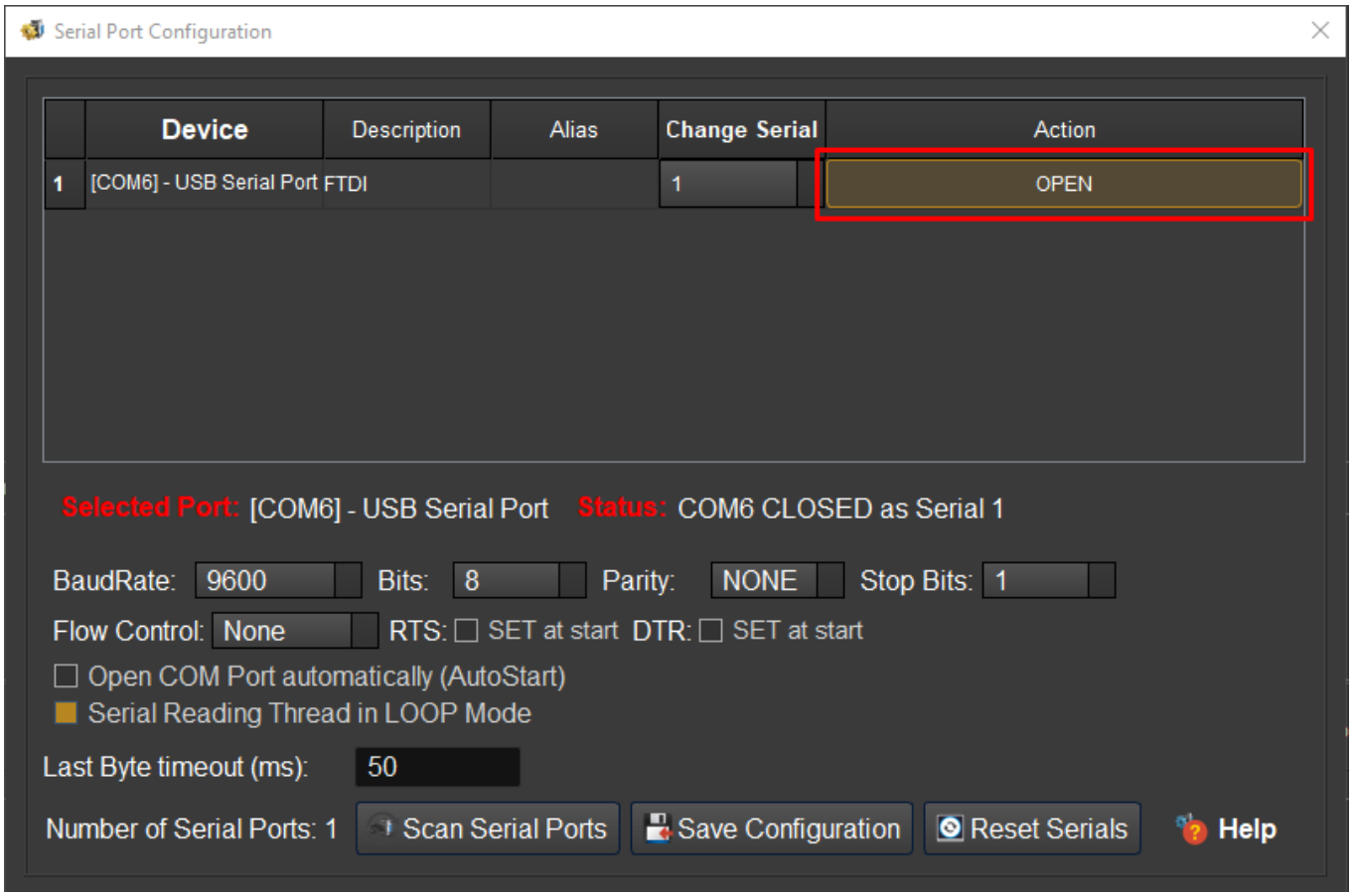
19. At the bottom of the dialog that appears, if no ports are already shown, `Scan Serial Ports`.



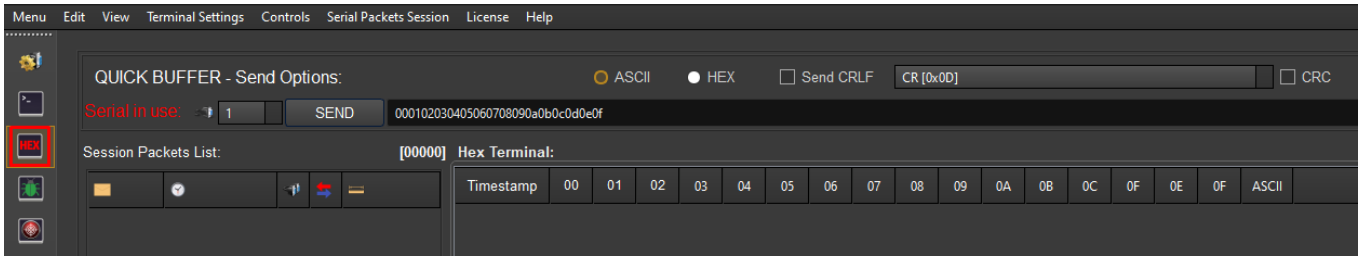

20. . A serial port should appear. Click the dropdown that says `NOT ASSIGNED`, then click `1`:



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:

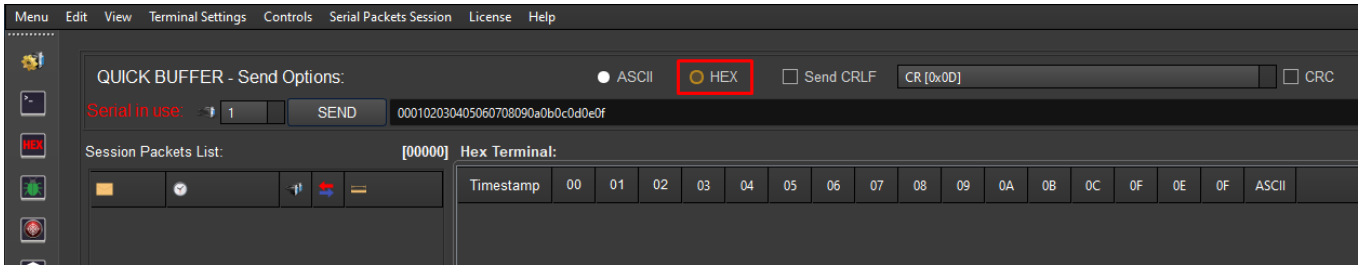



23. Close the dialog box (*not* the serial port!) if it does not disappear automatically.
24. On the sidebar, click the `HEX` terminal button, shown below. Your screen should then appear how it does on the right side of the image below.




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

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




## Hardware Testing and Analysis

26. The text box that is by default filled with `000102030405060708090a0b0c0d0e0f` is what you are sending to the FPGA.
27. Press `SEND` to send the data to the FPGA. For the 64-bit two-level multiplier, you should receive a 128-bit response.
28. Press the center button (`reset`) to reset the transceiver and prepare it for the next transmission.
29. Repeat for each input below (the first is provided again for you). Record your results in the spreadsheet. This data should be included in your final project report.
    - Note that everything is in hexadecimal.
    - A template spreadsheet is provided at the following link (make a copy): https://docs.google.com/spreadsheets/d/13vskDKgJuBocyXxhSi5Db3hFf6AFB9sTuGOXItceWrs/edit?usp=sharing

| MR Sign (SW1) | Multiplier (127..64) | MD Sign (SW0) | Multiplicand (63..0) |
|---|---|---|---|
| 0 | 0001020304050607 | 0 | 08090A0B0C0D0E0F |
| 0 | 8D8B667EC0A9E8E2 | 0 | B378F62F36596213 |
| 0 | 0000000000000000 | 0 | 0000000000000001 |
| 0 | 0000000000000001 | 0 | 0000000000000001 |
| 0 | 0000000000000001 | 1 | 0000000000000001 |
| 1 | 0000000000000001 | 1 | 0000000000000001 |
| 1 | 0000000000000001 | 0 | 0000000000000001 |
| 0 | FFFFFFFFFFFFFFFF | 0 | 0000000000000001 |
| 0 | FFFFFFFFFFFFFFFF | 0 | FFFFFFFFFFFFFFFF |

30. 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 11)
  - A screenshot of the FPGA resource utilization (LUTs, FFs) (Step 12)
  - A printout of your data spreadsheet (Step 29)
  - A screenshot of SerialTool containing your testing results (Step 30)

# 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.
- Practice using serial communication to test hardware descriptions by sending data to and receiving data from an FPGA development board.
- Practice working with Intellectual Property (IP) Cores.
- Practice using an onboard MMCM to generate a slower clock from the base 100 MHz clock on the FPGA development board.
- Understand a complex hardware description.
- Understand how high-precision integer multiplication can be implemented efficiently.