Lab 11

Part D – 64 Bit Two-Level Barrel Shifter

Description

- In this module, you will synthesize, implement, upload, and test a hardware description for a two-level barrel shifter, encapsulated within a serial transceiver.
  - You will use the same program for serial communication as in Parts B and C.

Procedure

Project Setup

1. Download `barrel_shifter_generic.vhd` from Canvas.
2. From here you have two options:
   1. **Create a new project** from scratch. To do this, follow Steps 1-8 from 11B-handout, but add `barrel_shifter_generic.vhd` instead of the three files listed for the 2LPE. Then, you can proceed to Step 6 of this handout.
      - Note that you will not need to enable VHDL 2008 on the 2LPE file (since it does not exist), but you must still do so for the hardware container file **and for** `barrel_shifter_generic.vhd`.
      - This is recommended if you had issues with Parts B or C.
   2. **Modify your existing project** by following the Steps 3-5 of this handout.
      - This will be much faster since you will not have to re-customize the Clocking Wizard IP.
3. Disable the decoder file:
   - In the `PROJECT MANAGER` pane, in the `Sources` subpane, expand the `apex` top-level component, and then its `hw : hw_container` subcomponent.
     - Right-click the `decoder : decoder_generic` component and click `Disable File`. Alternatively, press the `ALT` and `-` (minus) keys at the same time.

4. Add `barrel_shifter_generic.vhd` to the project:
   - In the toolbar, click `File`, then `Add Sources`. Alternatively, press `ALT` and `A` simultaneously.
   - If not selected already, check `Add or create design sources`, then click `Next`.
   - The rest of this wizard is identical to the corresponding part of the project creation wizard.
   - Click `Add Files`, then navigate to find `barrel_shifter_generic.vhd`. Once you have added it, click `Finish.`
   - Enable VHDL 2008 for `barrel_shifter_generic.vhd` by clicking on its entry, and changing the `Type` in the `Source File Properties` subpane, as you did for the 2LPE.
5. Comment out the code for Part B and uncomment the code for Part C:
   - Open the hardware container file by double-clicking on the `hw : hw_container` entry in the `Sources` subpane of the `PROJECT MANAGER` pane.
   - There is one labeled section you must comment out: lines 226-235.
   - 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.
   - There are two labeled sections you must uncomment: lines 202-204, and lines 241-257.
   - 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).

Hardware Review and Upload

6. Follow along with the instructor to briefly review the hardware.
7. `Open Elaborated Design`, review the schematic alongside the instructor, and take a screenshot of just the expanded `shifter` block for your report.
8. 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.
9. Before proceeding, plug the FPGA development board into your computer using the provided Micro-USB to USB-A cable.
10. At the top, in the green banner (or under the `Open Hardware Manager` dropdown) click `Open Target`, then `Auto Connect`.
11. 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.

12. Click `Program` to upload the bitstream to the FPGA.

Hardware Testing and Analysis

13. Use the same process as Parts B and C to connect to the FPGA with SerialTool.
    - The relevant section from 11B-handout is Steps 18-25. Steps 16-17 are to install and run the program.
    - If you disconnected the FPGA since Part B or C, you may need to scan for serial ports again.
14. For each input in the table below:
    - Type the `Input` into the text box in SerialTool and hit `SEND`.
    - Enter the `Shift Amount` on the 6 rightmost dip switches on the FPGA development board.
      - Note: the shift amount is in binary for your convenience. Everything else is in hexadecimal.
    - Record whether the expected output corresponds to your actual result.
    - Press the center button (`reset`) to reset the transceiver.
    - Repeat.
    - (Note: see Steps 26 and 27 from 11B-handout for more details).

Input	Shift Amount	Expected Output
`0000 0000 0000 0001`	`000000`	`0000 0000 0000 0001`
`0000 0000 0000 0001`	`000100`	`0000 0000 0000 0010`
`0000 0000 0000 0001`	`111111`	`8000 0000 0000 0000`
`FFFF FFFF FFFF FFFF`	`100000`	`FFFF FFFF 0000 0000`
`0000 0000 0000 0001`	`010101`	`0000 0000 0020 0000`

15. Take a screenshot of the SerialTool window after testing all the inputs above, to attach to your lab report.
16. Notice how shifting significant amounts cuts off data.
    - From your schematic capture in Step 7, you should notice that the output of the barrel shifter is 128 bits.
      - The reason only 64 bits are transmitted is because the transceiver is limited to 64 bits input and 64 bits output.
    - How could you modify the hardware container to view the most significant half of the output? The transceiver?

Deliverables

- Include as part of your **informal report**:
  - A screenshot of the Elaborated Design schematic (Step 7)
  - A screenshot of the FPGA resource utilization (LUTs, FFs) (Step 8)
  - A screenshot of SerialTool containing your testing results (Step 15)
  - A brief comment on whether your experimental results match the expected results.
  - Answers to handout questions (Step 16)

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.