Lab 05

Part C – Implementing Logic Gates

Description

In this lab module, you will test a hardware description of the seven basic logic gates and verify their truth tables experimentally.

Procedure

Project Creation

1. Download `lab05c.vhd` and `lab05.xdc` from Canvas and place it in a new folder titled `Lab05c`.
2. Open Vivado and create a new project.
    - From the Quick Start menu, select `Create Project`.
    - Click `Next`.
    - On the second page, if desired, give the project an appropriate and change the location. Click `Next`.
    - Ensure `RTL Project` is selected. Click `Next`.
    - On the `Add Sources` page, click `Add Files` and add `lab05c.vhd` from where you downloaded it.
    - Ensure `Copy sources into project` is checked. Click `Next`.
    - On the `Add Constraints` page, click `Add Constraints`
    - Select the following:
        - Category: `General Purpose`
        - Family: `Artix-7`
        - Package: `cpg236`
        - Speed: `-1`
    - Finally, select the middle option: `xc7a35tcpg236-1`.
    - Click `Next`.
    - Click `Finish`.

Hardware Development

3. Open the file in the editor, and follow along with the instructor to review the hardware description.
    - To do this: inside the `Project Manager` pane, under the `Sources` subpane, and under the `Design Sources` folder, double-click the `lab05c.vhd` file.
4. Run synthesis and implementation and then generate the bitstream to program the device.
    - From the left side pane, under the `PROGRAM AND DEBUG` dropdown menu, click `Generate Bitstream`.
        - If a popup appears, click `Yes`.
    - This may take a while, since it is running three steps of the development workflow in series.
    - After the operation completes, you may see a popup with the following options: `Open Implemented Design`, `View Reports`, `Open Hardware Manager` and `Generate Memory Configuration File`.
        - Here, select `Open Hardware Manager` and click `OK`.
            - If you clicked `Cancel`, navigate to that menu manually under the `PROGRAM AND DEBUG` dropdown by clicking on the `Open Hardware Manager` text.
5. Before proceeding, plug the FPGA development board into your computer using the provided Micro-USB to USB-A cable.
    - The Micro-USB end should be plugged into the FPGA at the top left.
    - The USB-A end should be plugged into your computer.
6. At the top, in the green banner (or under the `Open Hardware Manager` dropdown) click `Open Target`, then `Auto Connect`.
    - If you are prompted to grant administrator privileges to a program with a name like `hw_server`, do so.
    - This will connect the FPGA to your computer.

7. Once the device is connected, select `Program Device` (in either of the locations where `Open Target` was previously).
    - Ensure that there is a bitstream file selected (there should be a file path in this field). If there is not, you most likely selected the wrong target device (Step 3).
        - If you have selected the correct device, and the bitstream file has not appeared, you may be able to find it by doing the following:
            1. Click the 3 dots to the right of the field.
            2. Near the top left, click the AMD logo (5th from the left) with the tooltip `Jump to Recent Project Directory`.
            3. Navigate to `./lab05c.runs`, then `impl_1`.
            4. You should see `lab05c.bit` – select this, and hit `OK`.
    - A debug probes file is not necessary.
8. Click `Program` to upload the bitstream to the FPGA.
    - Several lights on the FPGA development board may blink during this process.
        - You will know when the program is complete when the `DONE` light has flashed off and on again, and the 7-segment display is no longer looping.

Testing

9. The gates are arranged as follows (this is also documented in the VHDL file):

Outputs (LEDs)	NOT	X	AND	X	OR	X	X	NAND	X	NOR	X	X	XOR	X	XNOR	X
Inputs (Switches)	NOT	X	AND		OR		X	NAND		NOR		X	XOR		XNOR	
Index	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

10. Determine the truth tables for each of the seven gates. For a truth table with n input bits, you will have 2^n possible outputs.
    - For instance, the `NOT` gate has 1 input and 2 outputs, and the others have 2 inputs and 4 outputs.

Deliverables

- As part of your informal report:
    - Completed VHDL description/code
    - Picture(s) of the programmed FPGA development board functioning as a two-high-bit detector

Outcomes

- Practice designing hardware to solve a specific problem.

- Practice working with VHDL.
- Practice using Vivado for hardware synthesis and implementation.
- Understand what a constraint file does.
- Understand how to program an FPGA with a hardware description.
- Understand how to test a hardware description on an FPGA development board.