Lab 06

Part C – The Two-High-Bit Challenge Strikes Back

Description

In this lab module, you will continue working on the two-high-bit detection challenge from Lab 05B, implementing it at the gate level after deriving its boolean equations using Karnaugh maps.

Procedure

K-Map and Equation Derivation

1. Alongside the instructor, derive the Karnaugh map for the two-high-bit detector hardware. A template is below.
    - While it may help, the truth table is not necessary to fill this K-Map. Recall that the output is `1` when two or more of the input bits are `1`.
    - Remember that K-Maps use graycoding (so the order is `00` `01` `11` `10`) and that for n input bits, you will have a K-Map with 2^n cells.

AB \ CD	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$				
$\bar{A}B$				
AB				
$A\bar{B}$				

2. After you have filled the K-Map, circle groups and derive the equations as discussed in class.
    - While you can use product-of-sum (maxterms/circle zeroes), it is likely easier to use sum-of-product (minterms/circling ones).
    - Remember to "kick out" the inputs that differ between rows/columns when transforming your groups to terms of a sum-of-product or product-of-sum equation.
    - Also remember that your groups must cover a number of cells that is a power of 2: 1, 2, 4, possibly 8, or theoretically, but not realistically (for a 4x4 K-Map), 16.

Gate Implementation

3. From your equation, draw the hardware consisting of `AND` and `OR` gates below.
    - You can assume that the inputs are available as both active high and active low (i.e., A and $\bar{\mathrm{A}}$ are both available).
        - However, if using sum-of-product, this should not be needed.

VHDL Implementation

4. Download `lab06c.vhd` and `lab06c.xdc` from Canvas and place them in a new folder titled `lab06c`.
5. Open Vivado and create a new project titled appropriately.
   - If you have not memorized how to do this by now, consult 06B-handout.
   - For your convenience, the board identifier is `xc7a35tcpg236-1`.
     - You can also find the board with the same options as before: `General Purpose`, `Artix-7`, `cpg236`, `-1`.
6. Based on the equations derived earlier, describe the hardware in VHDL using the provided stub hardware description, along with `and`/`or` statements.
   - If stuck, consult the instructor.
   - At this time, only use the new gate-level implementation – keep everything else commented out for now.
   - If you want to use the unary `or` operator, see the appendix "Enabling VHDL-2008" on the last page of this lab handout.
7. `Open Elaborated Design`, review the schematic alongside the instructor, and take screenshots for your report.
8. Run synthesis and implementation and then generate the bitstream to program the device. Then, open the Hardware Manager.
   - Take a screenshot of the resource utilization (LUTs and FFs) to add to your report.
9. Is the resource utilization the same as the previous code?
10. Before proceeding, plug the FPGA development board into your computer using the provided Micro-USB to USB-A cable.
11. At the top, in the green banner (or under the `Open Hardware Manager` dropdown) click `Open Target`, then `Auto Connect`.
12. 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.
13. Click `Program` to upload the bitstream to the FPGA.

VHDL Implementation Testing and Analysis

14. Use the four rightmost switches as the four bit input, and observe the rightmost LED to see the output.
15. Verify the functionality. Is the functionality the same as the previous code?
    - You may choose to do this by looking at the original truth table, or you can un-comment the provided example solution to 05B, re-upload to the board, and test them in parallel.

Deliverables

- Include as part of your **informal report**:
  - Derived 4x4 K-Map and equations (Steps 1 and 2)
  - Pictures of drawn gate implementation (Step 3)
  - Completed VHDL Code
  - Screenshot of Elaborated Design
  - Screenshot of FPGA resource utilization (LUTs, FFs)
  - Answers to handout questions (Steps 9, 15)

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 derive boolean equations from a truth table using Karnaugh Maps.
- Understand how to transform boolean equations to a hardware implementation using AND/OR gates.

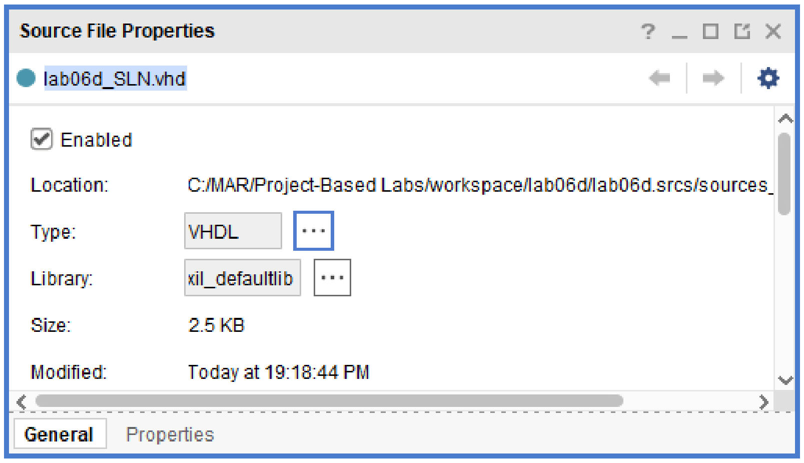
Appendix – Enabling VHDL-2008

- Certain features of VHDL, including the unary `or` and `and` operators, are only available with the newer 2008 standard. By default, Vivado uses the 1993 standard.
- In order to use VHDL-2008, each file must be switched to use the new standard. This can be done in one of two ways.

1. Run the following command in the `Tcl Console`:

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

2. While in the `PROJECT MANAGER` window, click the file in question (here, `lab06c.vhd`).
   1. Under `Source File Properties`, look for `Type: VHDL`, and click the three dots.



Source File Properties

lab06d_SLN.vhd

Enabled ✓

Location: C:/MAR/Project-Based Labs/workspace/lab06d/lab06d.srcs/sources_

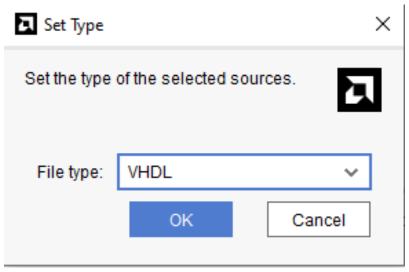
Type: VHDL

Library: xil_defaultlib

Size: 2.5 KB

Modified: Today at 19:18:44 PM

   2. This will open a drop-down menu, as shown below:

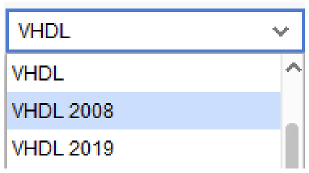


Set Type

Set the type of the selected sources.

File type: VHDL

   3. From the drop-down, select the file type `VHDL 2008`.



- Finally, re-run synthesis/implementation/bitstream generation. If you already did so and received the VHDL-2008 error, this will resolve it.