CS373 LM8: Introduction to FPGA’s and VHDL using   
Xilinx Vivado® and the BASYS 3 Board

Name(s) Stephen Haugland

|  |  |  |
| --- | --- | --- |
| **CATEGORY** | **POINTS** |  |
| Majority4 |  | 50 |
| TOTAL |  | 50 |

## Learning Goals for This Study Guide

* Understand where to locate additional information about FPGA design with the Basys 3
* Gain an introductory understanding of how and FPGA works and how they are programmed.
* Learn the key features of the Bassys 3 board.
* Understand how to test and verify that the Basys 3 board is working correctly.
* Learn beginning VHDL.
* Learn how to simulate a design.
* Challenge critical thinking skills by implementing a new project.

## Instructions for This Project

* You may work by yourself or with another person. You must **both** submit the code to whitgit along with your changes to this document. You must each answer the questions in this document by yourself, but you may consult each other on your answers.
* Answer the questions on the last page of this document and submit this document (with your answers in it) to Blackboard.
* Follow the instructions given in class to install Vivado on your machine (or use the machines in the lab – if you use lab machines which already have Vivado installed, you will need to install the board files for the BASYS 3)
* Create the majority project using the TCL/Bash scripts provided in class. You will need to configure the TCL script so that it builds your majority project.

# Majority4 Project

Now it’s time to put your newly learned skills to use. For this project you will design and implement the logic for a **Majority Circuit** that has four input bits.

## Marjority4 Project Specifications

* **Pull the LM8\_Majority starter project from whitgit using vscode**
* Create a **Majority** VHDL module that has:
  + Four STD\_LOGIC inputs, a, b, c and d.
  + One STD\_LOGIC output m :
    - This output will be high if 3 or 4 of the a,b,c,d inputs are active.
    - This output will be low if 0, 1, or 2 of the a, b, c, d inputs are active.
* **Create a testbench vhdl file to test all combinations of a, b, c, d of your Majority component**.
  + You could do this using nested simulation code loops for loops in a test bench, but this is not required for this lab. We will cover how to write more complex test bench code in class.
  + Make sure to configure the setup TCL script so that this simulation will be generated.
* Create a **Majority\_top** VHDL module that interfaces to the BASYS 3 board.
  + This should include the Majority component and map the board inputs/outputs to this component.
  + Setup the project so that SW(3), SW(2) … SW(0) map to the inputs of your majority circuit **a**, **b, c**, and **d** respectively.
  + Setup the project so that LED(0) is mapped to the output **m** of your majority circuit.
* **You must program the FPGA board and verify the majority circuit works: if any 3 of the four switches are high the LED should turn on. It should also turn on if all four switches are set to high.**
  + Modify the setup TCL script so that it will generate both the simulation and the bitstream

## Majority Circuit Design

1. Create a Karnaugh map for the Majority circuit include the k-map and the resulting Boolean equation for the Majority circuit here:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD/AB | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 0 | 1 | 0 |
| 11 | 0 | 1 | 1 | 1 |
| 10 | 0 | 0 | 1 | 0 |

M = BCD + ACD + ABC + ABD

1. Include a copy of your Majority VHDL module code here. Make sure it is commented well!

------------------------------------------------

-- Module Name: majority4

------------------------------------------------

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity majority4 is

-- complete the port statement for this module

-- defining inputs and output

    Port (  a : in STD\_LOGIC;

            b : in STD\_LOGIC;

            c : in STD\_LOGIC;

            d : in STD\_LOGIC;

            m : out STD\_LOGIC

        );

end majority4;

architecture majority4 of majority4 is

begin

-- complete the behavioral code for this module

-- Boolean logic equation derived from k-map simplification

    m <= (b and c and d) or (a and c and d) or (a and b and c) or (a and b and d);

end majority4;

1. Include a copy of your Majority\_top level VHDL module code here. Make sure you comment it well!

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity majority4\_top is

  Port ( sw : in STD\_LOGIC\_VECTOR (15 downto 0);  -- 16 switch inputs

        LED : out STD\_LOGIC\_VECTOR (15 downto 0);  -- 4 leds above switches

        an  : out STD\_LOGIC\_VECTOR (3 downto 0);  -- Controls four 7-seg displays

        seg : out STD\_LOGIC\_VECTOR(6 downto 0);   -- 6 leds per display

        dp  : out STD\_LOGIC                       -- 1 decimal point per display

  );

end majority4\_top;

architecture majority4\_top of majority4\_top is

-- add the component statement here for the majority4 entity

component majority4 is

  -- defining 4 single bit inputs and 1 single bit output

  Port( a : in STD\_LOGIC;

  b : in STD\_LOGIC;

  c : in STD\_LOGIC;

  d : in STD\_LOGIC;

  m : out STD\_LOGIC );

end component;

begin

-- map the the wires from the board to the majority 4 compoment as described in the lab instructions

-- hooking up inputs to board switches, and output to board led

m1 : majority4 port map (

  a <= SW(3),

  b <= SW(2),

  c <= SW(1),

  d <= SW(0),

  m <= LED(0)

);

----------------------Leave this untouched------------------------------

-- Turn off the 7-segment LEDs

an <= "1111";     -- wires supplying power to 4 7-seg displays

seg <= "1111111"; -- wires connecting each of 7 \* 4 segments to ground

dp <= '1';        -- wire connects decimal point to ground

end majority4\_top;

1. Include a screen snip of your Majority circuit simulation testbench running here:

Chart, treemap chart

Description automatically generated

1. Describe the design process, what issues you faced, and how it works.

First, we started with defining our majority4 component by describing the combinational logic for our output. In order to create this behavioral code, we had to first make a k-map to deduce the simplest Boolean expression for the output m. Once we had that, we could code that logic into our majority4 component. This file acted as a template or object that we could create an instance of later on in our simulation testbench and our top-level file to program the board. Next, we tested this majority 4 component to make sure our logic was correctly defined by feeding it different inputs and assuring the correct outputs were given through assert statements. In addition to looking for fail reports in our logs from failed assertions, we also were able to view the trace diagram (see question 4) to confirm our logic was correct. After we confirmed that our component was designed correctly, we edited the majority4\_top file in order to connect our majority4 component up to the input switches and output LED on the board. Then we generated a bitstream which was loaded onto our board and finally we tested our program on our board. To our surprise, once we had worked out all of the compilation errors and loaded the program to our board, it worked as expected and when 3 or 4 switches were high, the led turned on.

Overall our design process started by determining the combinational logic for our output using “and” and “or” gates and we slowly abstracted these details away until we had an instance of a majority\_4 module that we had tested, and then connected and programmed to our board.

Some of the main issues we had was with the syntax and getting used to how VHDL code works. At first we defined our inputs a,b,c,d to be STD\_INPUT\_VECTOR(0 downto 0) because we were following the example from the slides that had an input with multiple bits. This did not work however because we were trying to assign a vector to a single integer. Besides this, one main conceptual issue I had was with not realizing that the simulation test bench was not at all related to programming our boards. I was a little confused as to why we were defining our majority4 component in multiple files but we needed to in order to first run a simulation on it and second, connect it to the board.