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

Name(s) Shane Snediker

|  |  |  |
| --- | --- | --- |
| **CATEGORY** | **POINTS** |  |
| Majority4 | 52 | 50 |
| TOTAL | 52 | 50 |
| Great work! I gave extra credit for the great report you made! Good job! |  |  |

## 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:

Majority module truth table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| a | b | c | d | m |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

Majority module K-map:

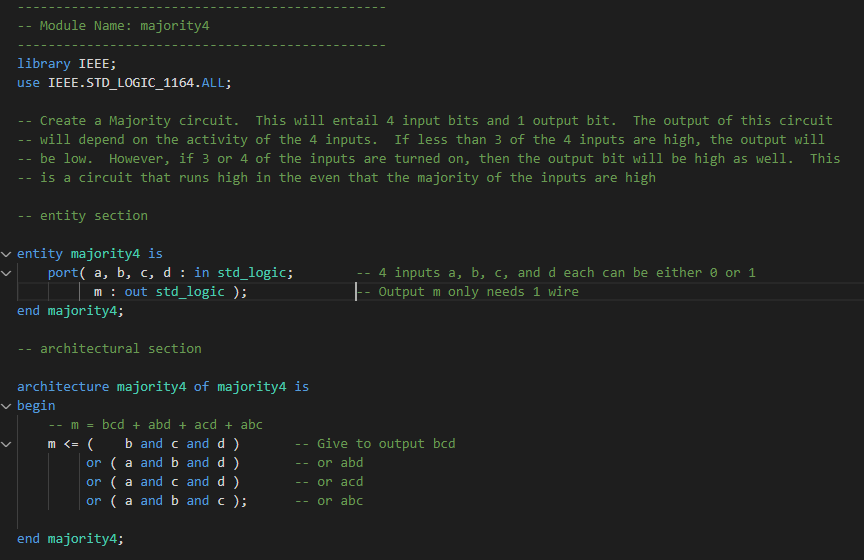
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| cd\ab | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 0 | 1 | 0 |
| 11 | 0 | 1 | 1 | 1 |
| 10 | 0 | 0 | 1 | 0 |

a b c d a b c d a b c d a b c d

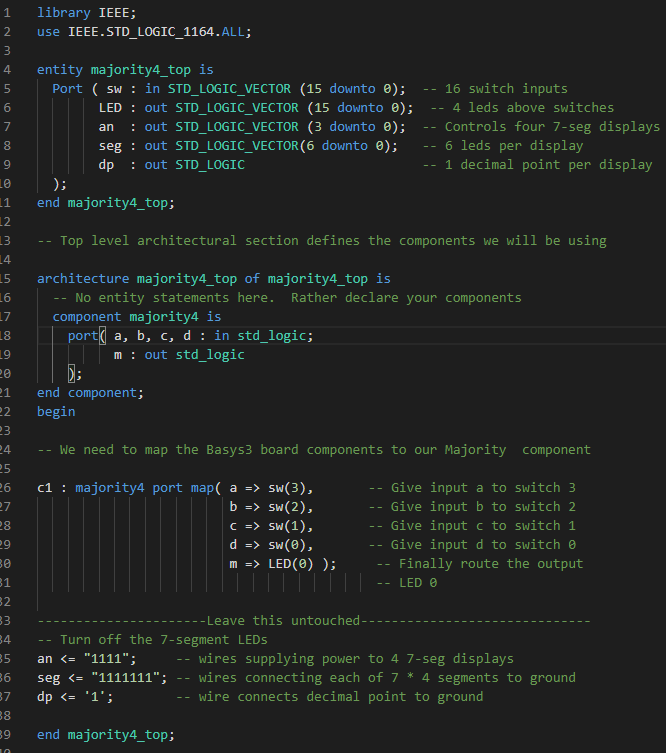
a b c d a b c d a b c d a b c d

\*\*\* m = b c d + a b d + a c d + a b c

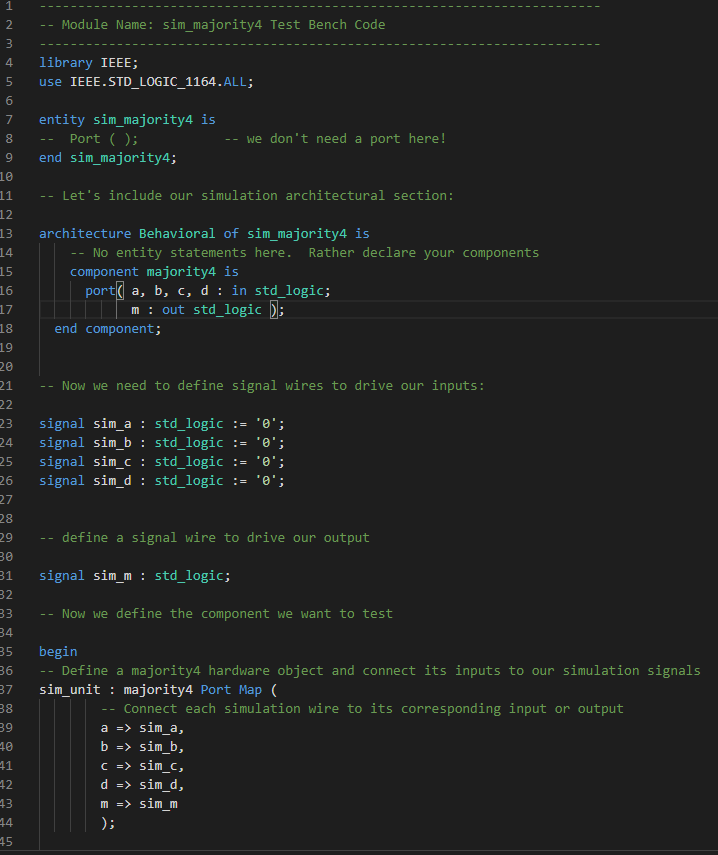
1. Include a copy of your Majority VHDL module code here. Make sure it is commented well!

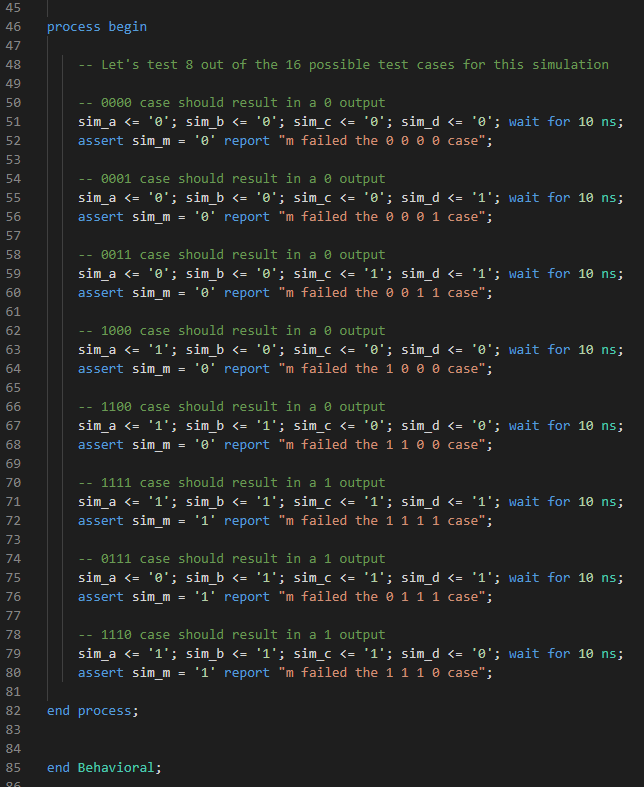


1. Include a copy of your Majority\_top level VHDL module code here. Make sure you comment it well!



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



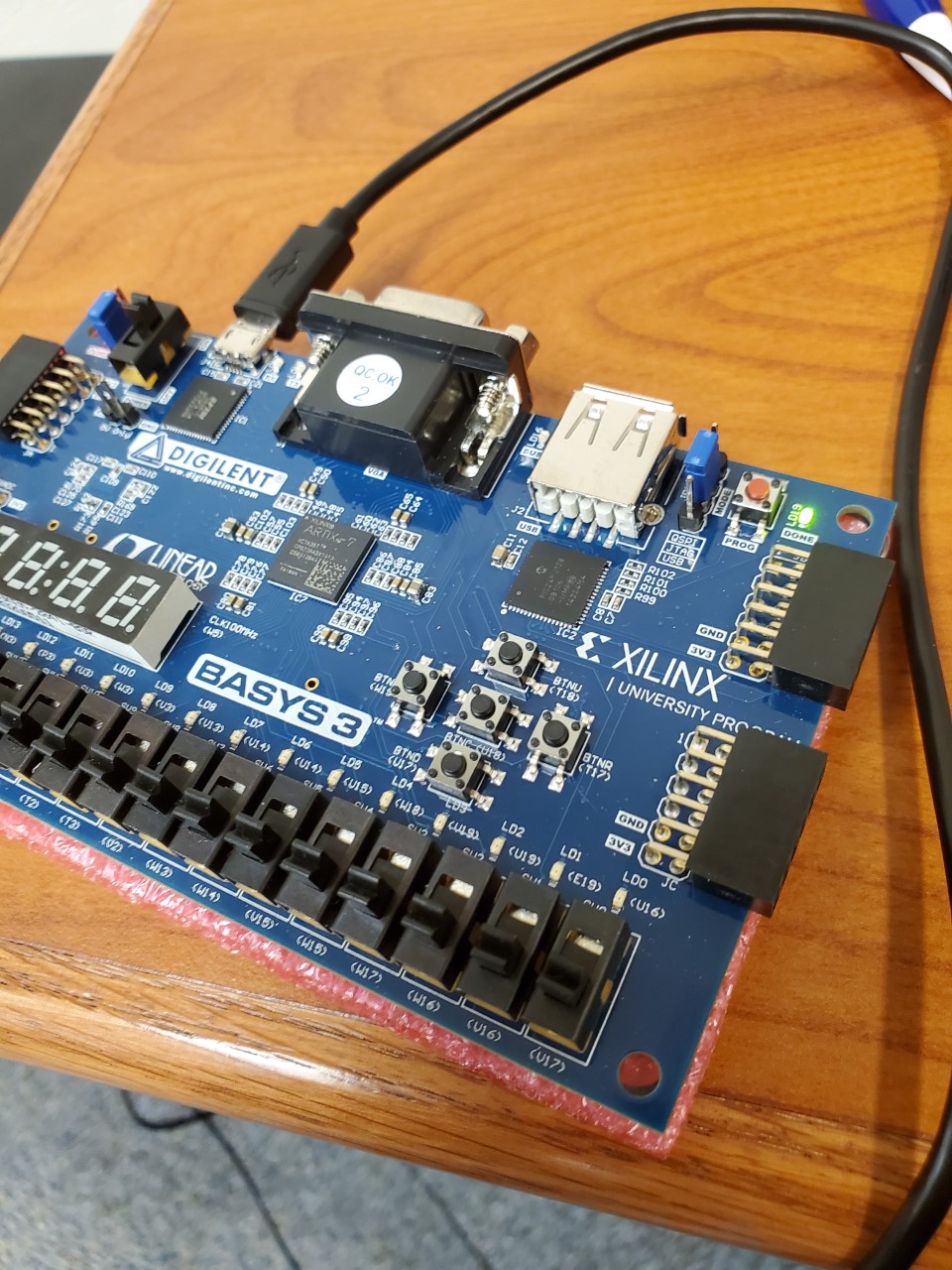


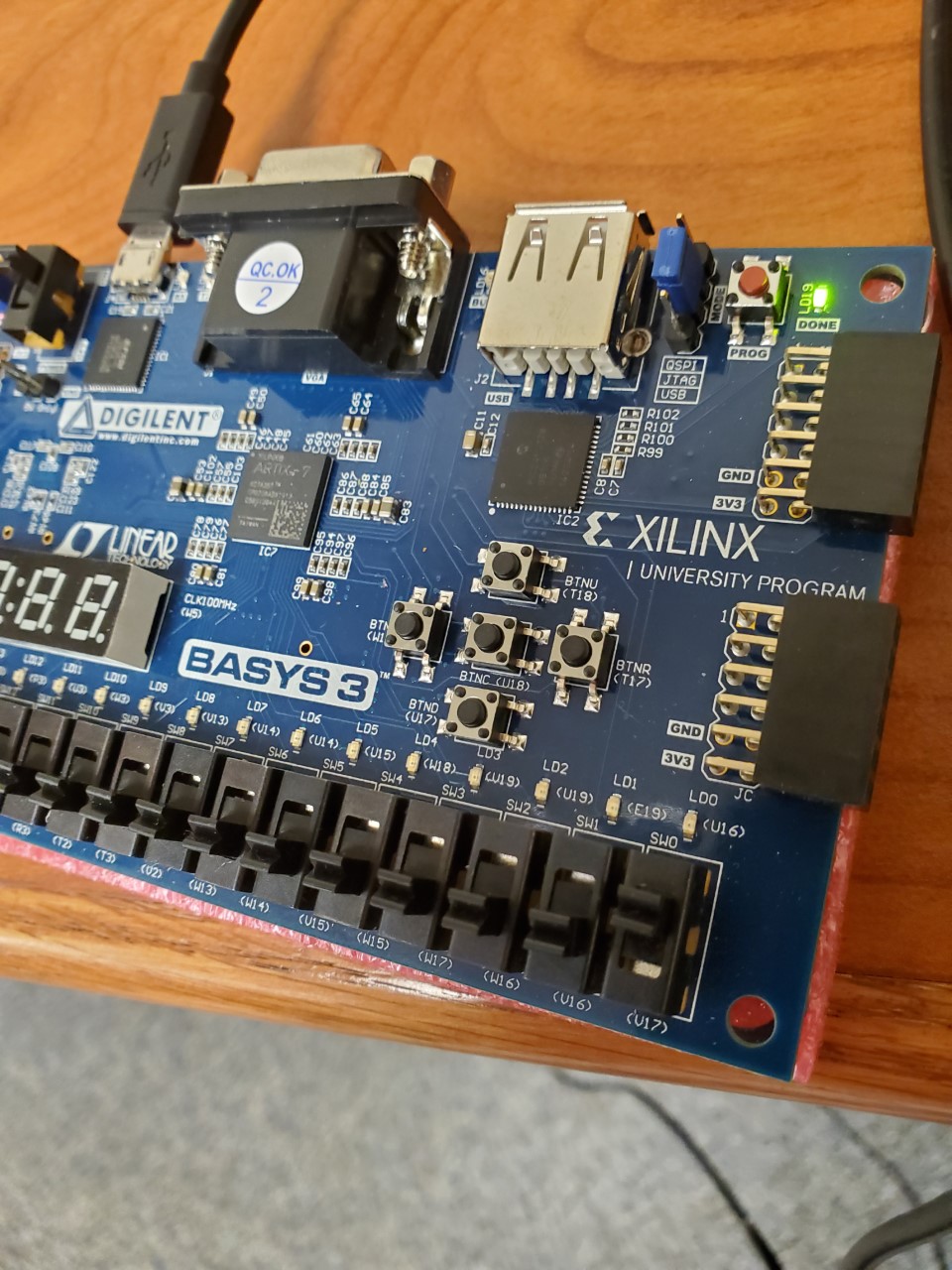
1. Describe the design process, what issues you faced, and how it works.

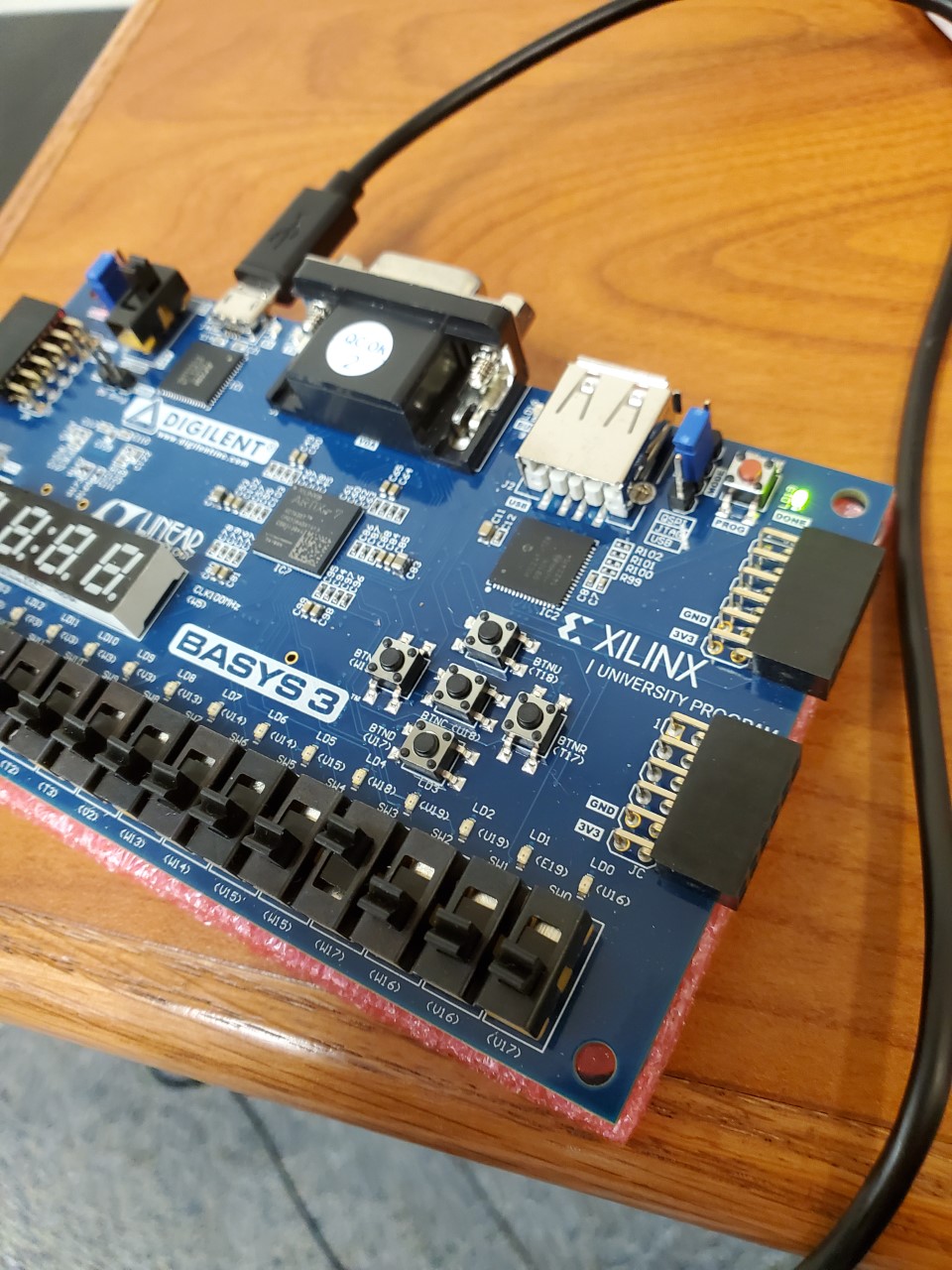
This was a fantastic project. I mean it. I genuinely had fun finally getting my hands on VHDL, Vivado and the Basys3 board. The tutorials and lectures that you’ve done walked us through everything so well that it was easy for me to put together this lab. The only real issue that I would have faced in this project if I wasn’t so OCD and asked you so many questions that the issue presented itself in one of my snippets shared through Discord, was that I didn’t realize that std\_logic\_vector signals were distinguished by double quotations and std\_logic signals with single quotations in the simulation test bench file. This would have likely created an error that may have taken me quite awhile to figure out if I hadn’t have encountered it through our chat discussion thread. Other than that, my greatest hurdle with this project was that I was so far behind due to the issue of not being able to have Vivado function on my personal machine that I ended up having to go back and watch every single one of the last 4 lectures and follow along very closely to get myself to the point where I was understanding all of the different file types, the differences between behavioral, structural and algorithmic code, and other syntactical nuances involved in learning VHDL. However, having such a strong foundation in learning computer languages, I’m not finding the process of learning VHDL all that difficult. I’m really enjoying the process. Taking the time to write out the truth table for the majority component, constructing the K-map, writing the majority4.vhd file, writing the majority4\_top.vhd file and writing the test bench file and then to run the ./gen.sh script and have the project compile successfully, to open the elaborated design and see the logic come out how I would expect it to come out, to run the simulation and have no issues arise with it and then connect my board to Vivado and have the switches and LED function exactly how I was expecting it to work was truly gratifying.

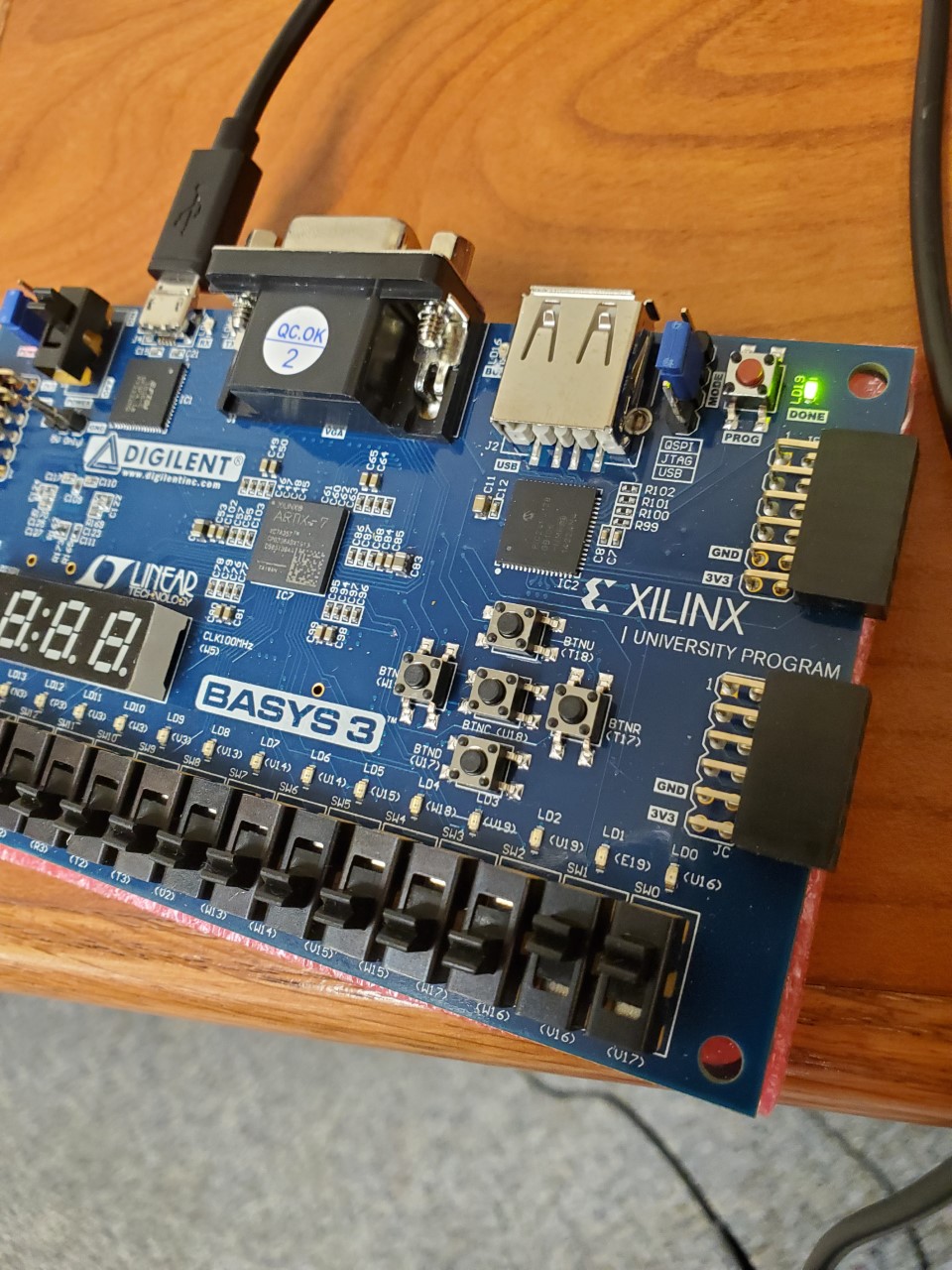
The majority component is a logical function that provides a single output based on 4 inputs. The output will be high, or 1, which will light up the LED light bulb connected to the output if a majority of the input signals are high. A majority of 4 signals is anything above 2 signals. Therefore, if 3 of the 4 signals are high or all 4 of the signals are high, the output light bulb will be high. If a majority of the inputs are not high, then the output will be low, or 0, and therefore the LED light bulb will not be turned on. This configuration means that there are 5 possible input configurations that will lead to a high output, and 11 input configurations that lead to a low output. My majority vhdl code connects the 4 single-bit inputs to the first 4 switches on the Basys3 board (sw(0), sw(1), sw(2) and sw(3)) while the output gets routed to the very first LED light bulb (LED(0)). Therefore, the light bulb turns on whenever at least 3 of those switches are turned on high. I won’t bog down this report with pictures of all 16 cases to prove that the FPGA was connected accurately, but here’s a few to show that it was working.

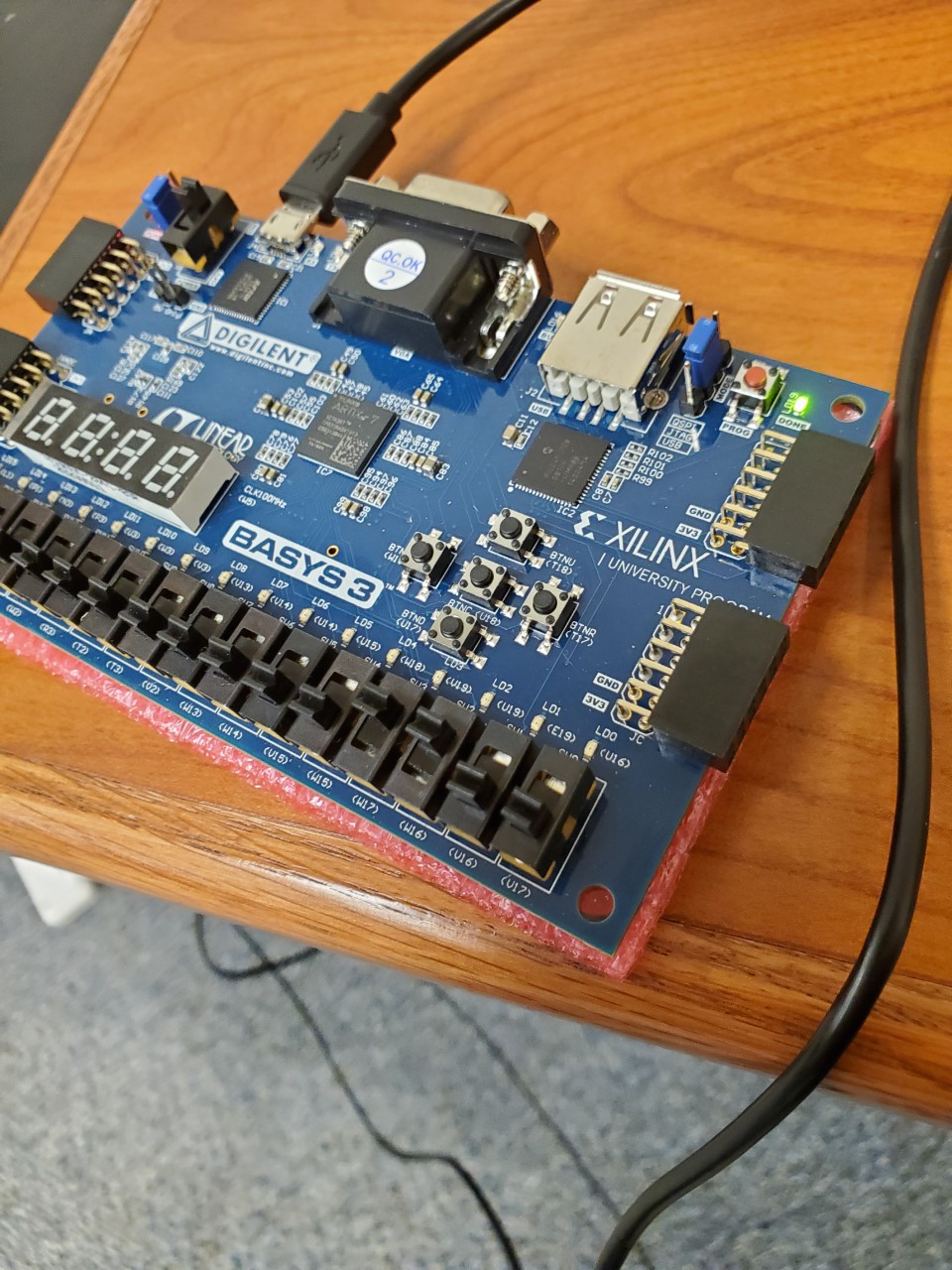
5 of the low output cases:











And here’s the 5 high output cases:

