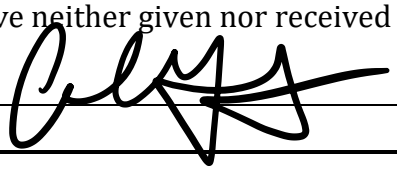
ECE 3544: Digital Design I
Project 3 (Part A) – Design and Synthesis of a Binary Converter System

Student Name: Colin Grundey

Honor Code Pledge: I have neither given nor received unauthorized assistance on this assignment.



Grading: The design project will be graded on a 100 point basis, as shown below:

Manner of Presentation (30 points)

_____ Completed cover sheet included with report (5 points)

_____ Organization: Clear, concise presentation of content; Use of appropriate, well-organized sections (15 points)

_____ Mechanics: Spelling and grammar (10 points)

Technical Merit (70 points)

_____ General discussion: *Did you describe the objectives in your own words? Did you discuss your conclusions and the lessons you learned from the assignment?* (5 points)

_____ Design discussion: *Did you discuss the approach you took to designing any original modules? Did you discuss the approach you took to assembling the top-level module?* (10 points)

_____ Testing discussion: *What was your approach to formulating your test benches? How did you verify the correctness of the modules you designed?* (5 points)

_____ Supporting figures: *Waveforms showing correct operation of the top-level module.* (10 points)

_____ Supporting files: *Do the modules pass any tests applied by the grading staff? Modules whose declarations do not conform to the requirements of the project specification cannot be tested, and will receive no credit.* (10 points)

_____ Validation of the final design on the DE1-SOC board (30 points)

_____ **Project Grade**

ECE 3544: Digital Design I
Project 3A Validation Sheet

Colin Grundey
Student's Name

GTA Validation Instructions:
Program the FPGA on the DE1-SoC Nano board with the student's implementation of the comparator system. When the programming has successfully completed, perform the set of tests described in the table below. For each case, indicate whether or not the student' design demonstrates the behavior described.

Procedure and *Expected Result*	Correct Operation (**Yes** or **No**)
With SW[6] = 0, set SW[5:0] = 011001. *HEX4 should display "b". HEX[3:2] should display "19". HEX[1:0] should display "25".* **Binary 011001 is hex 19. This is equivalent to decimal 25.**	
Change SW[6] to 1. *HEX4 should display "d". HEX[3:2] should display "19". HEX[1:0] should display "13".* **BCD 011001 is decimal 19. This is equivalent to binary 010011, which is hex 13.**	
With SW[6] = 0, set SW[5:0] = 001110. *HEX4 should display "b". HEX[3:2] should display "0E". HEX[1:0] should display "14".* **Binary 001110 is hex 0E. This is equivalent to decimal 14.**	
With SW[6] = 1, set SW[5:0] = 010100. *HEX4 should display "d". HEX[3:2] should display "14". HEX[1:0] should display "0E".* **BCD 010100 is decimal 14. This is equivalent to binary 001110, which is hex 0E.**	
With SW[6] = 0, choose a 6-bit binary number whose value is between 0 and 39, inclusive. Write down the value you choose and its decimal equivalent. SW[5:0] = _____ (binary) = _____ (decimal) *HEX4 should display "b". HEX[3:2] should display the hexadecimal equivalent of the binary value on SW[5:0]. HEX[1:0] should display the decimal value that corresponds to the binary number on SW[5:0].*	
With SW[6] = 1, choose a valid 6-bit binary coded decimal number. It should correspond to a number that is between 0 and 39 inclusive. Write down the binary value you choose and its equivalent in binary. For example, SW[5:0] = 011000 is BCD 18, which is equal to 010010 in binary. SW[5:0] = _____ (BCD) = _____ (binary) *HEX4 should display "d". HEX[3:2] should display the decimal equivalent of the BCD value on SW[5:0]. HEX[1:0] should display the hexadecimal value that corresponds to the binary equivalent of the BCD number on SW[5:0]. (Whew!)*	

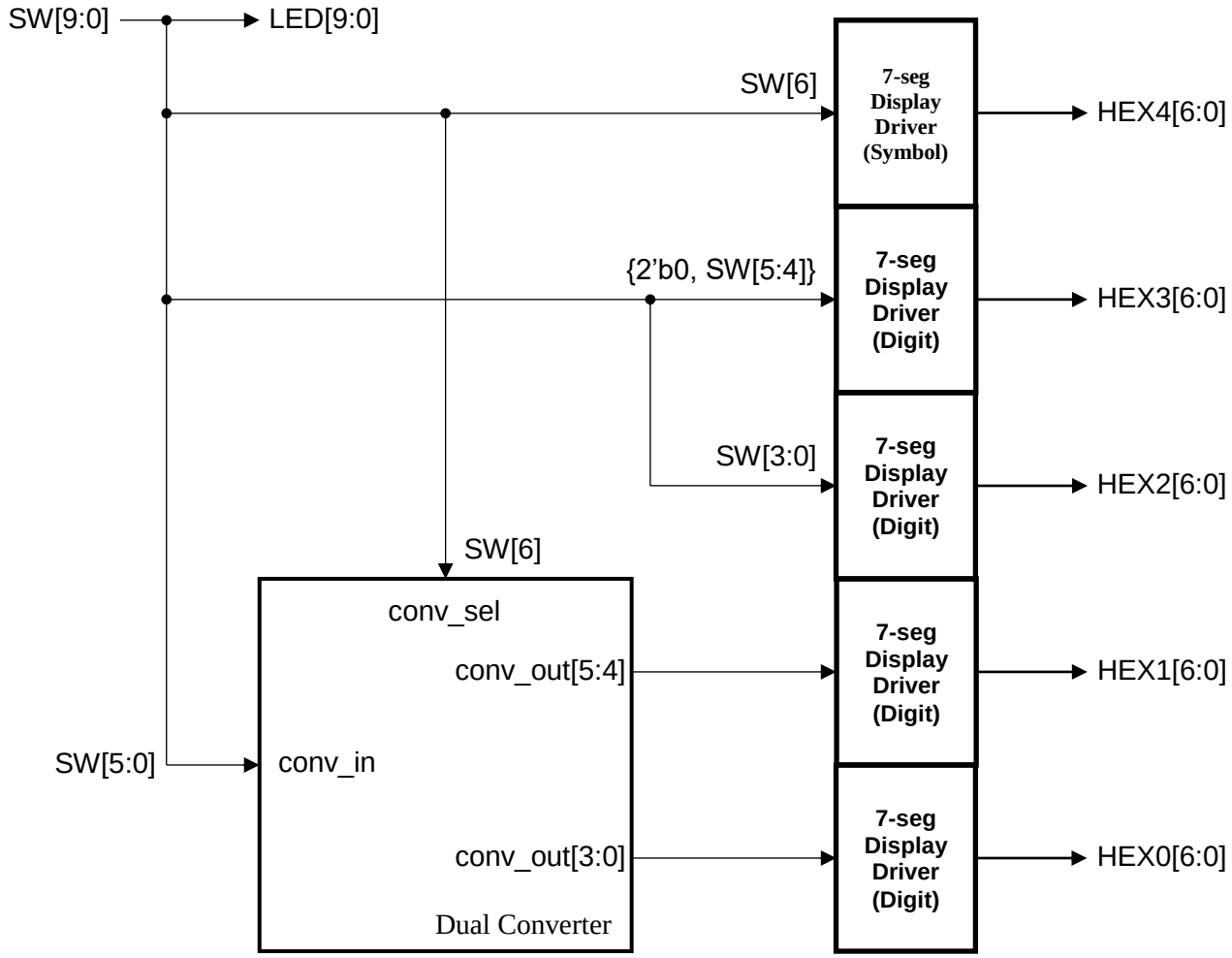
GTA Printed Name and Signature: _____

Date and Time of Validation: _____

Objective

In project 3A, we are tasked with using the converter modules from the previous project in a larger system that eventually can be run on the DE1 board and controlled by various input switches. The changes in the input switches cause changes in the 7-segment hex displays driven by modules we have created in previous assignments. Applying our Verilog skills and modules from before to a piece of hardware is the goal of this project as we will be able to see and control the system built.

Design Process



The diagram above shows the top-level structure for the system we need to implement. It consists of a dual-converter module with the sn184 and sn185 converters. This module is to select which conversion to perform based on the select input. It should perform a binary-to-BCD conversion when the select is low, and BCD-to-binary when the select is high.

There are two other modules clearly defined in the diagram above that drive the 7-segment hex displays. The first is a binary display driver that simply maps a 4-digit binary number to the 7 segments of a display. The second uses a single switch as input and displays the letter "b" or the letter "d" to denote which conversion is taking place on the board.

Implementation

To begin, I outlined the dual-converter module in Verilog and created instances of my converters from the last project. I then used the select input to choose the proper conversion method for generating the output. That was essentially all the dual-converter entailed. Next, I tackled the 7-segment display

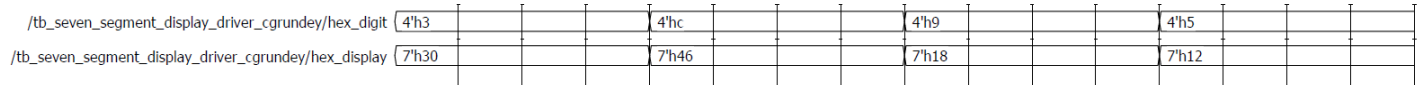
driver. I used combinational logic to drive the hex display as in Homework assignment 3. Finally, the 7-segment symbol display driver only can output a “b” or a “d”. When switch 6 is high, the output should light the display with the letter “d” and when switch 6 is low, “b” should be displayed. It is evident that a basic continuous assignment statement can be used to implement this as shown below.

```
assign hex_driver = (select) ? 6'b0100001 : 6'b0000011;
```

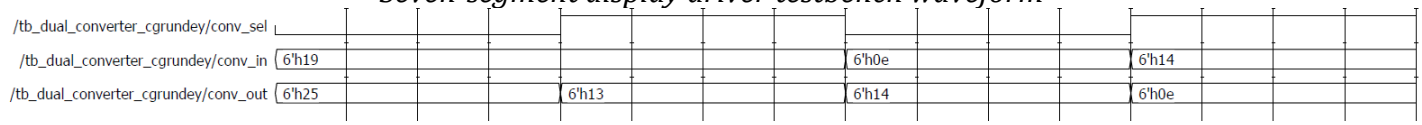
Lastly, I instantiated all of the newly created modules in the top level Verilog file called `project3aTop.v`. I properly placed the switch values as inputs to the various modules as outlined in the first diagram.

Testing

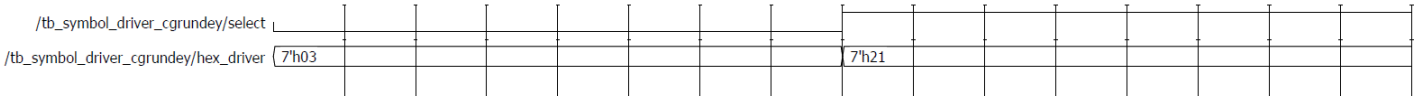
Before trying to program the board to see if my implementation works, I created test benches for the seven-segment display driver, symbol display driver, dual-converter, and full system modules. I used the examples from the validation sheet in my test benches for the converter and entire system to ensure the proper function. The output waveforms are shown below.



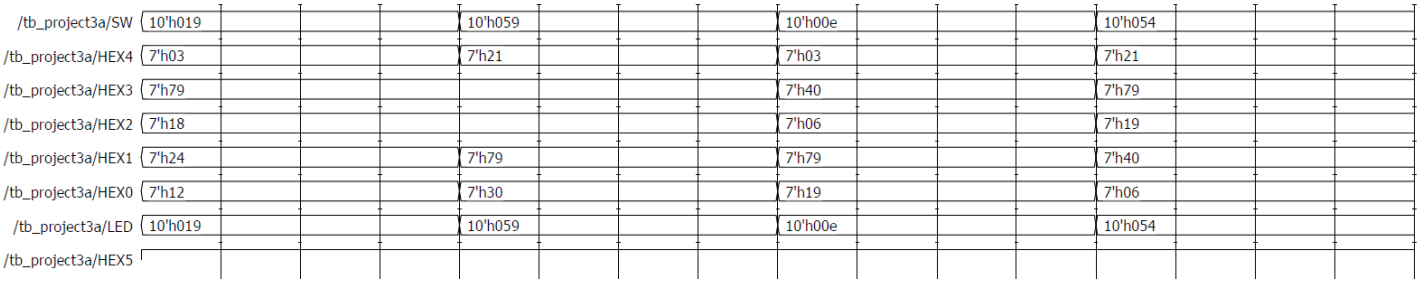
Seven-segment display driver testbench waveform



Dual-converter testbench waveform



Symbol driver testbench waveform



Full system testbench waveform

Conclusion

Project 3A gave us the opportunity to apply our new Verilog skills to a piece of hardware and observe a more physical output as opposed to dissecting waveforms from ModelSim. Most of the difficult logic was already implemented from previous assignments so the main purpose of this was to improve our knowledge in model instantiations within a larger system. It also allowed us to use the Quartus software to program the DE1 board and rearrange switches to observe outputs. This proved to be a tedious project, especially connecting instantiations together, due to large amount of inputs and outputs for the entire system.