ECE 3544: Digital Design I
Project 3 (Part B): Design and Synthesis of a Synchronous Finite State Machine

Student Name: Jacob Abel

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 other conclusions and the lessons you learned from the assignment?* (5 points)

_____ State machines: *Does your state diagram for the* `keypressed` *module correctly represent its behavior? Did you address the questions on structuring state machines in Verilog?* (10 points)

_____ Implementation discussion: *Did you discuss the approach you took to modifying the counter logic?* (5 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 that do not conform to the requirements of the project specification will receive no credit.* (10 points)

_____ Validation of the final design on the DE1-SOC board (25 points)

_____ **Project Grade**

ECE 3544: Digital Design I
Project 3B Validation Sheet

GTA Validation Instructions:
Program the FPGA on the DE1-SoC board. When the programming has successfully completed, press and release KEY1 to reset the design. Record the value of the seven-segment displays:

_____
Press KEY1

Set SW[6:0] to 0000000. (The switches are 0 when they are down.) Press and release KEY0. Record the values of the seven-segment displays.

SW[6:0] = 0000000 _____
Press KEY0

If the value does not match the last four digits of the student's Student ID Number, stop the validation.

DO NOT RESET THE DESIGN. Set SW[6:4] to 101. Choose a non-zero value for SW[3:0]. Record the value.

_____
SW[3:0] value

Press and release KEY0 five times. After each release, record the value of the seven-segment displays.

SW[6:4] = 101	_____	_____	_____	_____	_____
	1st Press	2nd Press	3rd Press	4th Press	5th Press

Reset the design using KEY1. Set SW[6:4] to 100. Choose a new non-zero value for SW[3:0]. Record the value.

_____
SW[3:0] value

Press and release KEY0 five times. After each release, record the value of the seven-segment displays.

SW[6:4] = 100	_____	_____	_____	_____	_____
	1st Press	2nd Press	3rd Press	4th Press	5th Press

DO NOT RESET THE DESIGN. Set SW[6:4] to 111. Press and release KEY0 five times. After each release, record the value of the seven-segment displays.

SW[6:4] = 111	_____	_____	_____	_____	_____
	1st Press	2nd Press	3rd Press	4th Press	5th Press

DO NOT RESET THE DESIGN. Set SW[6:4] to 110. Press and release KEY0 five times. After each release, record the value of the seven-segment displays.

SW[6:4] = 110	_____	_____	_____	_____	_____
	1st Press	2nd Press	3rd Press	4th Press	5th Press

_____ _____
GTA Printed Name GTA Signature

_____
Date and Time of Validation

Project 3B

ECE3544 CRN:82989

Jacob Abel

November 10, 2018

Objective

The objective of this project is to demonstrate a capability to synthesise sequential designs onto a real world system such as an FPGA. This project also demonstrates a basic capacity to interpret and implement state diagrams.

Button FSM Diagram

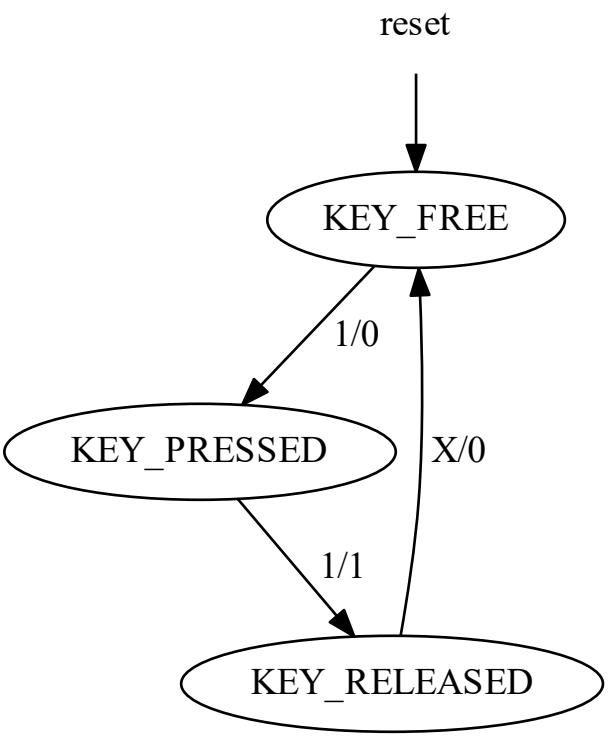


Figure 1: `keypressed` Module FSM (Input: `enable_in`, Output: `enable_out`)

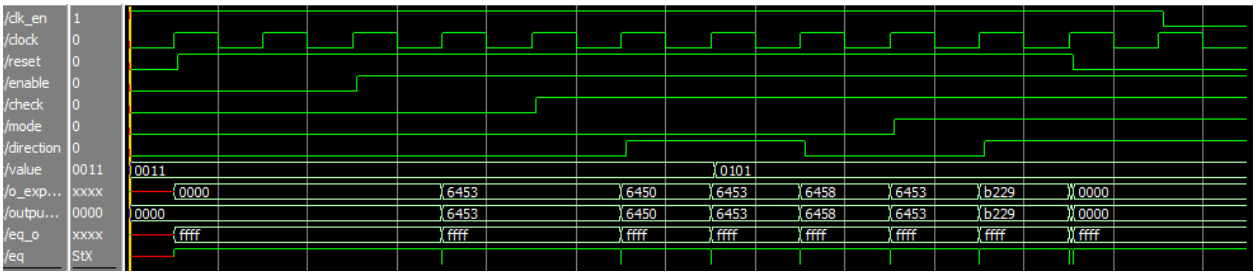
FSM Module Design

The design process was relatively straightforward. The initial stage is outlining project requirements on scratch paper and verifying it against the official project specification. At this point the starter program was adapted to include 7 segment displays and the proper pin assignments. Functionality of the starter module was verified and the button module was analysed to develop a basic state diagram. Now the final system design started by replacing the bulk of the FSM module with a `casez` block that corresponded to each stage of the specified state machine. The test-bench was also implemented at this point by walking through the state diagram and matching its output against expected values. Several small bugs were ironed out and the module was tested on the physical device with equivalent functionality.

There were some failed attempts and discoveries made during this project. One attempt was to use the Accellera Open Verification Library which ultimately delayed the project and was removed. Utilising this library will be attempted again with the next project given sufficient time. The discovery was of the usefulness of the `casez` statement and the restrictiveness of normal `case` statements. The prior permits wildcard evaluation and is extremely useful.

While the implemented FSM did not follow the standard FSM pattern, due to its simplicity, implementing it as a single procedural always block resulted in significantly cleaner code. A more complex FSM would utilise the standard FSM pattern and be divided into two to three procedural always blocks. The button module opted to use the standard three block pattern with the first block handling internal register states, the second block handling changes of states from inputs, and the final block handling outputs. The primary FSM does all of this in one block with a `casez` statement.

Project FSM Module Simulation



This simulation demonstrates that all the basic inputs produce the correct outputs. The `reset`, `enable`, `check`, `mode`, `direction`, and `value` values are the inputs that compose the switch input values and the `outputValue` output is the output as 7 segment display values. The `eq_` wires demonstrate equivalence on each output pin and the `eq` wire demonstrates equivalence on all outputs. The equivalences are computed using an XNOR against expected results.

Conclusion

This project demonstrated the process of implementing, simulating, and physically verifying state machines in Verilog for FPGAs. Despite starting this project early, the project encountered delays due to inexperience during an attempt to utilise the OVL library for the first time. Otherwise, little to no issues arose during the project. As such the project can easily be considered a success with the exception of the the delays due to the lack of experience with the OVL library and the decision to pursue use of said library in lieu of experience.