7-Segment Banner

List of Authors (Thomas DeSchutter, Alex Fillmore, Joshua Kulwicki)
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
e-mails: tmdeschutter@oakland.edu, afillmore@oakland.edu, jkulwicki@oakland.edu

***Abstract*—The purpose of this project was to implement a scrolling banner on the eight 7-segment displays present on the Nexys A7-50T. This project is capable of displaying two messages at two different scrolling speeds, both options selectable by a user through switches present on the board. Additionally, the user can hold a button to temporarily pause the scrolling message, to be resumed when released. Another button is used to reset the system entirely. The findings of this project were that this system could be implemented using components studied this semester in ECE 2700. This includes multiplexers, decoders, a finite state machine, counters, and a 4-bit parallel access shift register. This project clearly suggests that powerful and complex digital systems can be constructed out of relatively simple components.**

I. Introduction

This report will cover the methodology and implementation of a scrolling seven segment banner with different speeds on an FPGA.

The motivation behind creating a scrolling seven segment banner was to deepen our understanding of finite state machines and digital systems. The result is a functional digital system that displays the power of the hardware used. The end user can adjust switches to display different messages as well as change the speed of those messages.

Many topics learned in this course were used in completing this project. This includes the various hardware components that were implemented in VHDL. Counters, shift registers, decoders, finite state machines, multiplexers, and a hex to 7-segment decoder are all important components used in this digital system. The information learned about these components and how they function were integral in completing this project [1].

II. Methodology

For this project, the seven segment displays on the Nexys A7-50T are used to display the scrolling messages. Vivado was used to code the components that were needed to make the scrolling banner work. Using a hex to 7-segment decoder like the one used in the lab allowed us to display the scrolling message on the 7 segment displays. The most important component in the design of this project is the shift register. This stores all the data to display the message on the seven segment displays. It also helps in making it appear that the message is being scrolled across the displays. Another important component of this project would be the finite state machine. This allows for sequential control and is essentially the brain of the system. It controls the 8-to-1 mux which selects the correct letter from the shift register and controls which display is on. More detail about the components is explained in the following sections.

A. Counters

Two counters were used to allow for the different scrolling speeds. The two different speeds that can be selected are 0.5s and 1s. The output of these counters is connected to a 2-to-1 mux and they can be selected using a switch on the FPGA. The output of this 2-to-1 mux is then connected to another mux with the other input being tied to '0'. This allows the user to pause the message that is being scrolled using a button. Finally, a 1 ms counter was used to control the finite state machine [3].

B. FSM

The state machine was needed to control the 3-to-8 decoder and the 8-to-1 mux. This ensures that the correct displays are on at the correct time and the correct letter of the selected message is displayed. The output of this state machine is used as the selector of the 8-to-1 mux and the input to the 3-to-8 decoder. The inputs of the state machine include the output of the counter and the clock.

C. 3-to-8 Decoder

The 3-to-8 decoder is used to select which display is on at any given time. The input of this decoder is the output of the finite state machine. This decoder converts that 3-bit input to an 8-bit output that is used to light up the correct display.

D. 4-Bit Parallel Access Shift Register

A shift register was used to shift the letters so that they appear to be scrolling across the seven segment displays. The shift register consists of d flip flops and 2-to-1 muxes. This component selects the correct message to be displayed when the user flips a switch. It also controls the speed that the message is scrolled or pauses it based on input from the user.

E. 4-Bit Parallel Access Shift Register

An 8-to-1 mux is used to select the output that is displayed on the 7-segment display. In other words, which letter of the selected message is displayed at any given time. This is determined by the selector which is the output of the finite state machine.

F. Hex-to-7 Segment Decoder

The output of the 8-to-1 mux enters the hex-to-7 segment decoder. This is then responsible for converting the 4-bit input to an 8-bit output that will display the correct pattern on the 7-segment display.

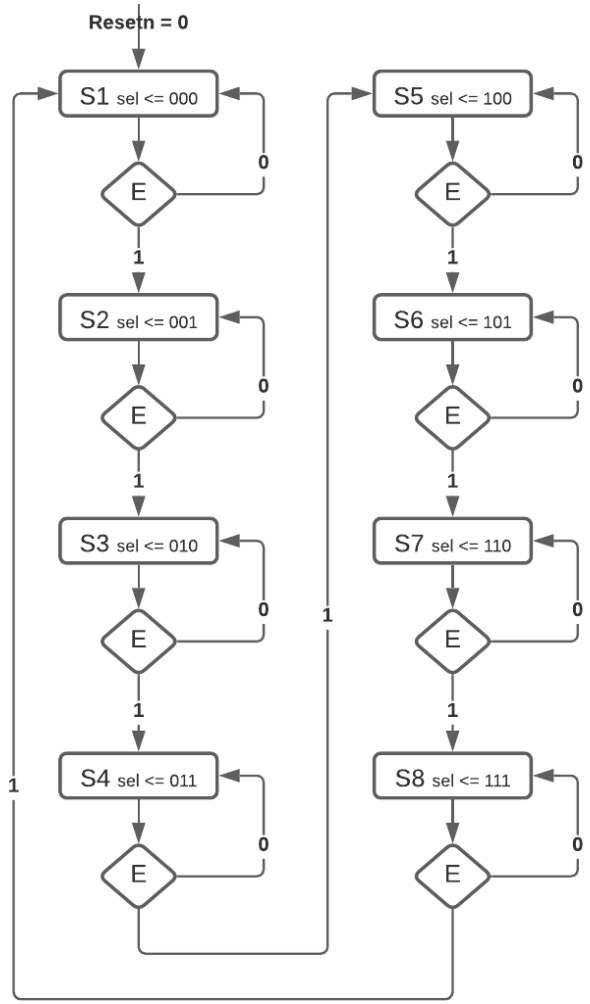

Figure 1: Finite State Machine

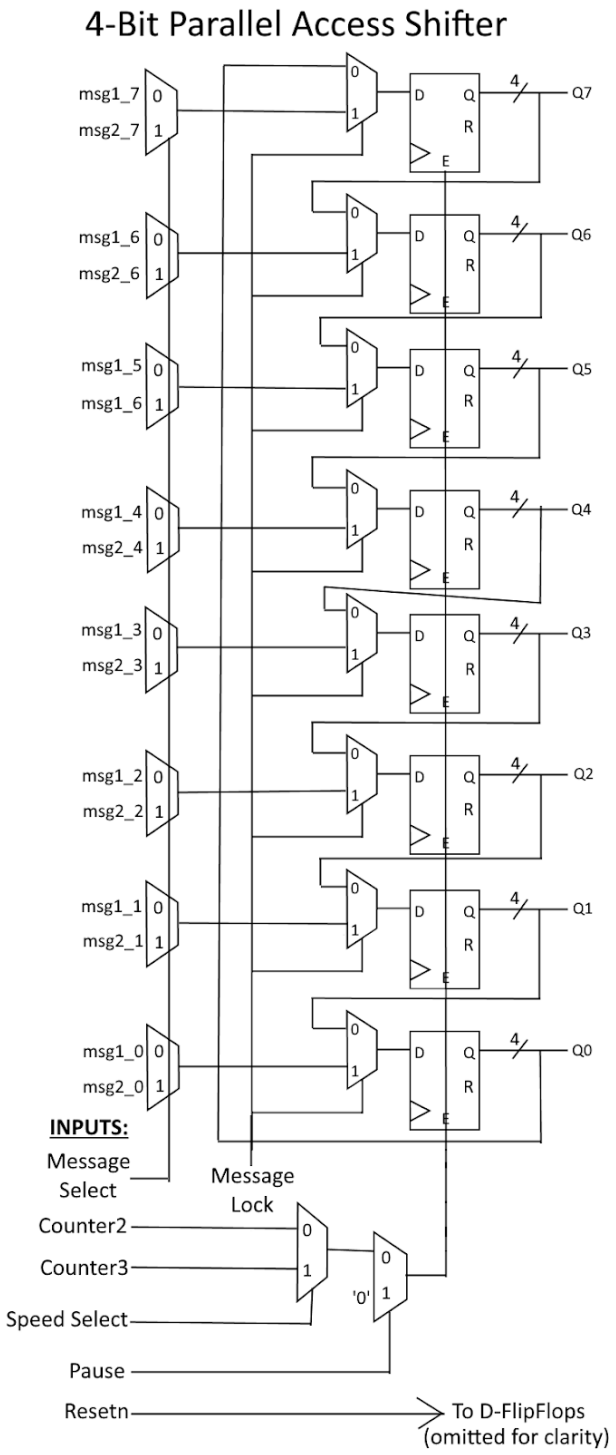

Figure 2: 4-Bit Parallel Access Shift Register

III. Experimental Setup

The components required for this project were programmed in VHDL using Vivado. The final design was displayed on the Nexys A7-50T. Using Vivado we were able to create a testbench to make sure the program was working correctly. Using simulations such as timing and behavioral allowed us to make sure the design was working before implementing it on the FPGA. These tests were also

important because they allowed us to check that the timing was correct so the messages would properly scroll across the seven segment displays. After seeing the results of these tests, we were able to tell if we needed to make modifications.

IV. Results

After generating the bitstream and doing some debugging of our circuit, we were able to get it functioning. We were able to test it by adjusting the different switches and buttons that we assigned the inputs too. A switch could be used to select between the two different messages. The speed of the scrolling message could also be changed by flipping a switch. The message could be paused by holding a button or reset by pressing the reset button. These were the expected results. The results that we had were related to topics we learned in class. For example, we learned that only one 7-segment display can be lit up at once. However, by using a very quick timer and a state machine to cycle through the displays, we were able to make it appear that the message was scrolling across the displays. We also learned a lot about shift registers in class and that was a major component in our project. We were able to implement a shift register and see it in action.

Conclusions

Implementing a digital system onto an FPGA can be simple, but even small mistakes can completely change the results. In specific, we ran into some timing issues with our circuit. By creating a testbench and running simulations, the issue was identified and fixed. We also learned that finite state machines can be very particular. Our finite state machine originally assigned a new output value when the input from the counter was equal to 1. However, this was causing letters and numbers to overlap on the 7-segment displays. To fix this, we needed to assign a new output value once it reached a new state [2]. One improvement that could be made would be to simplify our code. In addition, we could add more messages for the user to select or have an option to have the message scroll the other way, etc.

References

[1] VHDL Coding for FPGAs. [Online]. Available: http://www.secs.oakland.edu/~llamocca/VHDLforFPGAs.html. [Accessed: 14-Apr-2021].

[2] 7-segment serializer (four displays) [Online]. Available: http://www.secs.oakland.edu/~llamocca/Tutorials/VHDLFPGA/Vivado/Unit_7/serializer.zip [Accessed: 14-Apr-2021].

[3] Counter modulo-N (generic pulse generator) with enable and synchronous clear [Online] http://www.secs.oakland.edu/~llamocca/Tutorials/VHDLFPGA/Vivado/Unit_5/my_genpulse_sclr.vhd [Accessed: 14-Apr-2021].

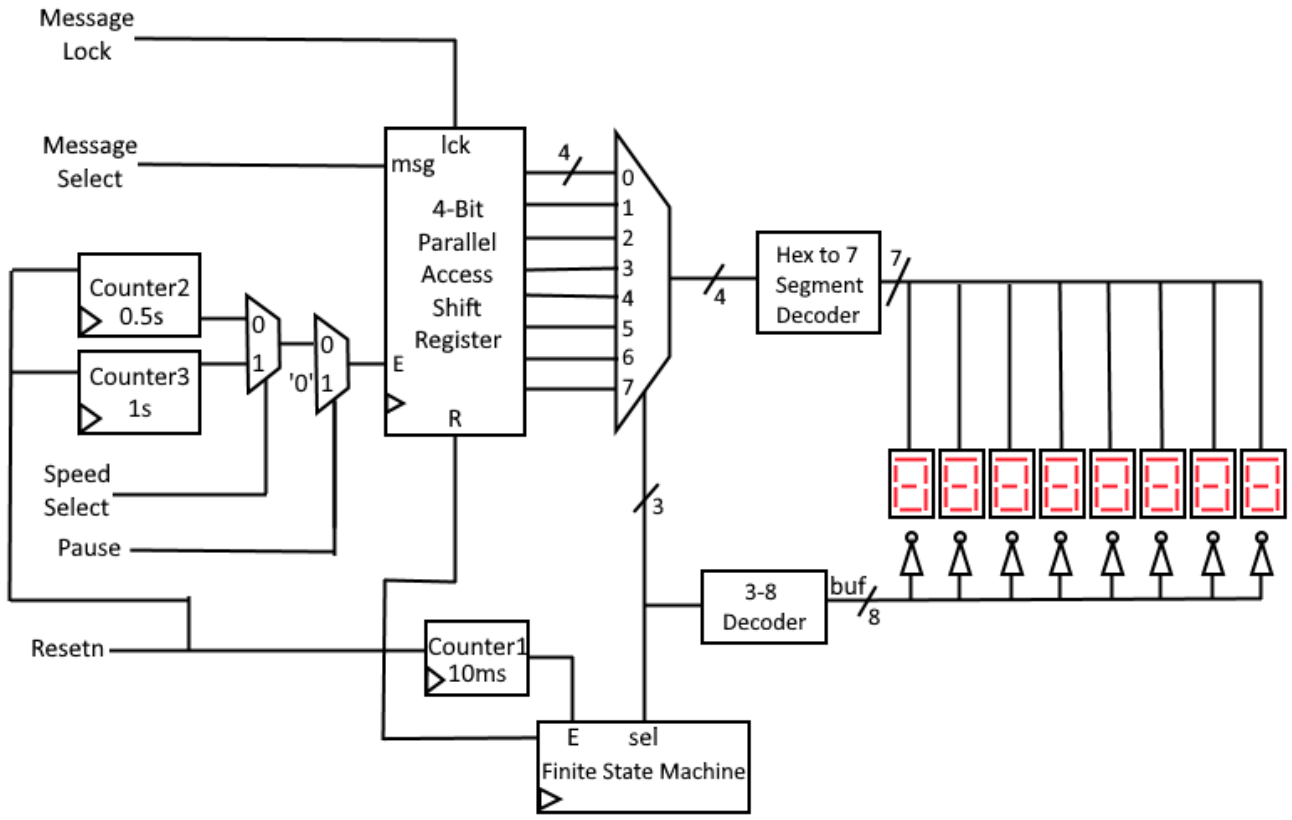

Figure 3: Final Circuit Diagram