Seven Segment Banner Display

Brian Conlon, Lucas Costello, George Habeb
Electrical and Computer Engineering Department School of Engineering and Computer Science
Oakland University, Rochester, MI
conlon@oakland.edu, georgehabeb@oakland.edu, lcostello@oakland.edu

Introduction

Our objective was to create a seven segment banner display to be used within the Oakland University Engineering Building. We decided to go with an architecture based application by demonstrating our design on a Nexys A7 (100T) board. The board has a 7 segment display, switches, and everything else needed to code and test our design. The code consists of 16 source files within Vivado (VHDL), which includes the top file. For more information on specific parts within this, please see the block diagram listed on page 3 (Figure 3). Our motivation for this project was that we knew as people start their first semesters at Oakland, it can be confusing to differentiate the buildings on campus. We thought a seven segment banner display not only helps the students, but better shows what to expect from their future classes. The main implications of our project were based around the planning process. The planning process includes making a block diagram of how all of our components would interact, which took some time to work out. Within our Digital Logic Design class at Oakland University, we learned a lot about shift registers, mux, finite state machines (FSM), decoders, counters, and much more we applied to our project. We learned on our own how to apply these components together to this specific application. Although this application was made to be used at Oakland University, it can be used anywhere.

Methodology

In our project, the Nexys A7-100T FPGA board's seven-segment displays feature one of two chosen messages scrolling from left to right. Users can control functions like variable scrolling speeds and pausing the display. The message selection is controlled by a 2-to-1 mux, directing the chosen message to a parallel access shift register with enable. The shift register plays a crucial role in storing and shifting the characters of our message, delivering them to an 8-to-1 mux. The finite state machine (FSM) manages the 8-to-1 mux and the 3-to-8 decoder, ensuring the correct display is enabled for the selected character. The FSM operates on a 10ms counter, shifting the selected character and display every tenth of a second, creating the illusion of multiple segments being on simultaneously. To achieve variable speed settings and pause functionality, we employ two individual counters and two multiplexers, working together to create the desired effects. All synchronous circuits in our system share the same clock and reset input, ensuring a synchronized operation. Further details

about our components and their workings can be found in the following sections.

A. multipleShiftReg :

The multipleShiftReg component is designed to maintain the original values of characters and shift them to the right. Comprising 2-to-1 multiplexers and D flip-flops, it remains simple while ensuring efficient data manipulation. The enable signal is sourced from another component named displayControl. The resultant output of the shifted message is then conveyed to the 8-to-1 Mux. A more detailed view of the multipleShiftReg component can be found below (Figure 1).

B. 8-to-1 Mux

Taking the outputs from the multipleShiftReg and the FSM, the 8-to-1 Mux outputs the selected character to the Binary-to-Seven Segment Decoder.

C. displayControl :

The way we went about implementing the variable speed and pause options was to use two different counters along with two muxes. As can be seen in the block diagram (Figure 3), the enable switch on the shift register was connected to the output of this component.. For example, if the “pause display” option was selected, the mux would choose a zero instead of the clock. This would pause the shift register, and continue displaying what was currently on the display. The other mux for the “speed select” has a mux connected to two different speed clocks, which the user can use a switch to select between the two.

D. Finite State Machine

The finite state machine (FSM) was instrumental in coordinating the selected character from the 8-to-1 Mux with the 3-to-8 Decoder. To achieve a smooth presentation of the message across all eight displays, we serialized the FSM. This involved syncing its enable with a 10ms counter, allowing a systematic shift through each character and display, revealing the complete message. The FSM's output, a "sel" value, is then directed to both the 8-to-1 Mux and the 3-to-8 Decoder. The state diagram for the FSM can be seen below (Figure 2).

E. Binary-to-Segment Decoder

In order to include every letter and number along with a character for "space" we had to use 6-bits to represent each character. The way our Binary-to-Segment Decoder works is by taking the input and checking for the corresponding case. After it finds the case it outputs the 7-segment equivalent of the character.

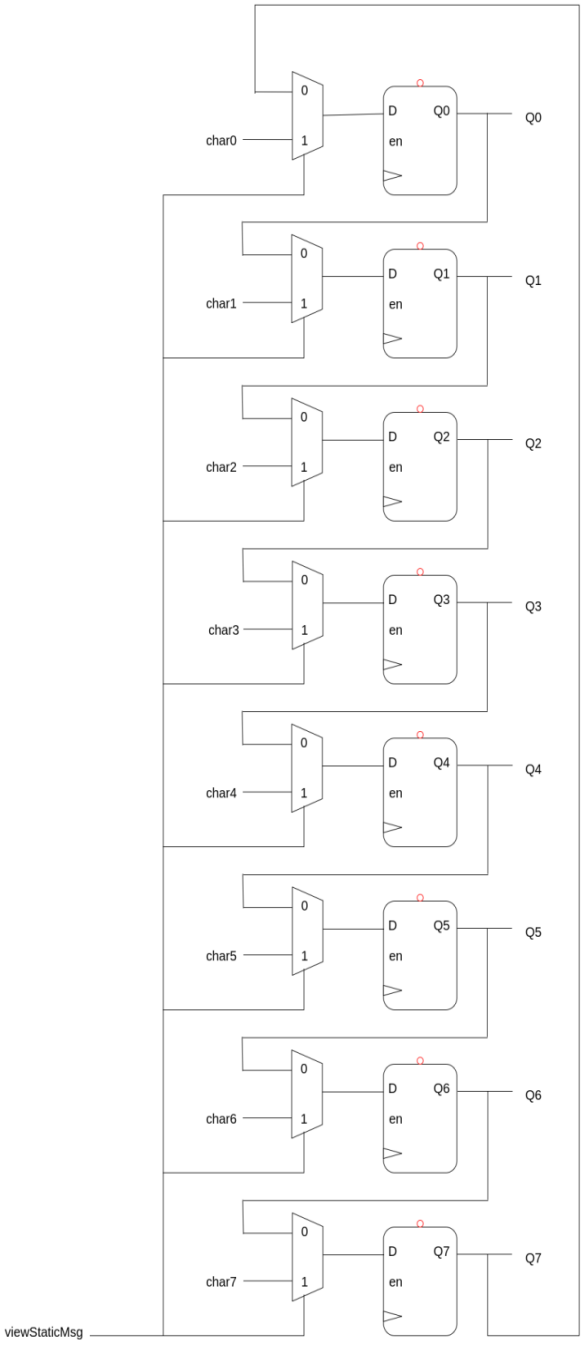

Figure 1: multipleShiftReg Component

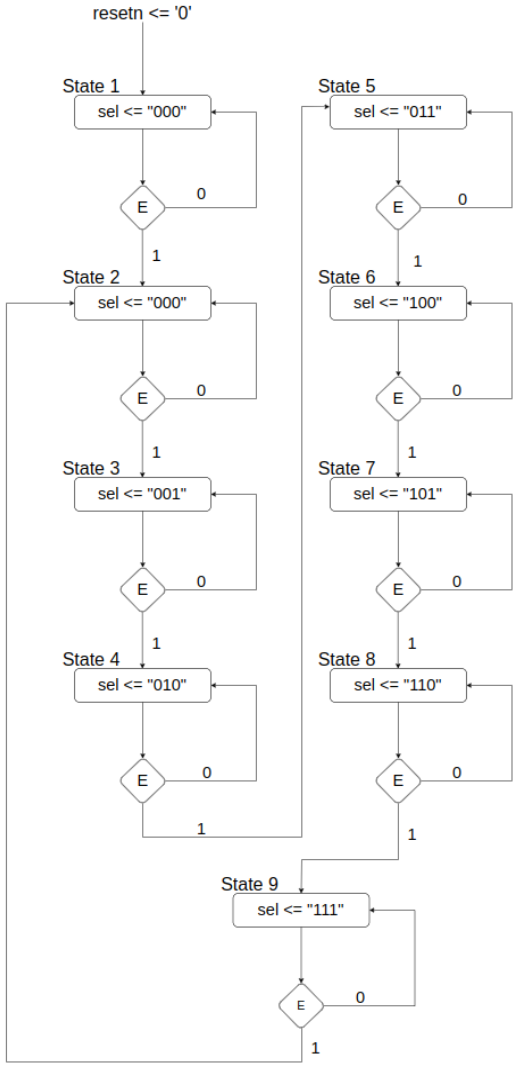

Figure 2: State Diagram

Experimental Setup

In this project, VHDL was used to describe the circuit, and Xilinx Vivado 2021 played a crucial role in software simulations. The circuit's outputs were closely linked to the 7-segment display, following the constraints file for the Nexys A7-100T board. The subsequent synthesis and implementation steps in Vivado resulted in a bit-stream for programming the FPGA board, which then underwent rigorous testing. The user interface featured four switches: one for

selecting the banner message, another for choosing variable speed (with two options), a third for pausing the display, and the fourth to show a static version of the selected message. VHDL programming within Vivado was essential for developing the project's components, and the testing process, including simulations and adjustments, ensured the accuracy and reliability of the implemented system.

Results

After refining the bitstream and conducting thorough circuit debugging, our project achieved successful functionality. Testing involved utilizing the four assigned switches: one for message selection, another for scrolling speed adjustment, a third for pausing, and a fourth for viewing a static message version. The inclusion of a reset button facilitated efficient message resetting. These outcomes met our expectations and aligned with fundamental concepts discussed in class, particularly the constraint of illuminating only one 7-segment display at a time. Leveraging a serialized Finite State Machine (FSM) and implementing parallel access shift registers allowed us to create the illusion of simultaneous activation across all 8 displays and seamless message shifting. Overall, our project not only showcased effective implementation but also incorporated essential theoretical concepts into a functional and engaging display system.

Conclusions

In conclusion, the seven-segment display project proved to be successful, and the learning experience was invaluable. Our observations indicate potential improvements, such as expanding the project by incorporating additional seven-segment displays. The current setup accommodates phrases of up to 8 characters, and enhancing it could involve integrating more shift registers to increase the message capacity

References [1] Llamocca, Daniel. "Digital System Design." Oakland University, Rochester, MI

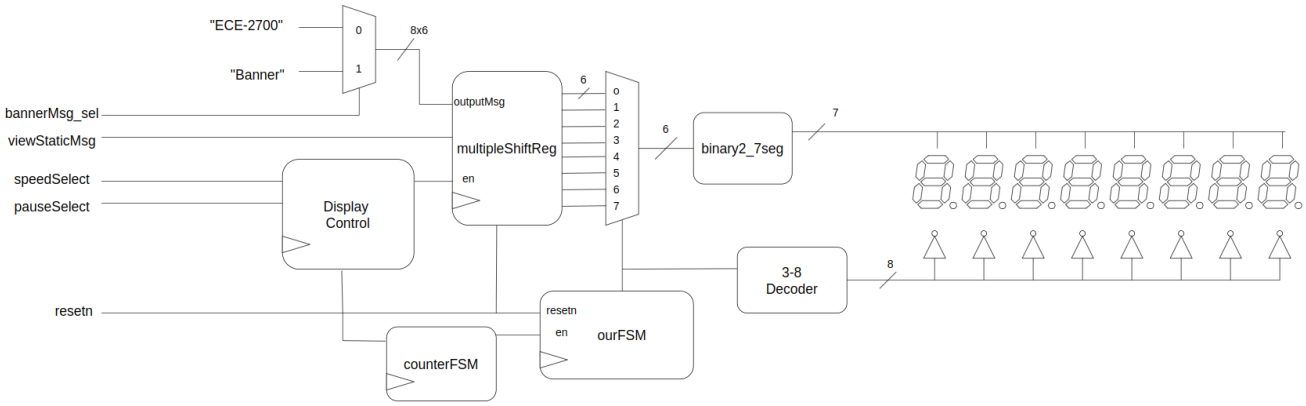

Figure 3: Complete Circuit Diagram