Digital Stopwatch/Timer

Riana Matheis, Michelle Mertz, Denisa Nazarko

Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
E-mails: rlleek2@oakland.edu, mmertz@oakland.edu, dnazarko@oakland.edu

***Abstract*—The goal of this project was to formulate logic to create and implement a timer/stopwatch in VHDL, using an FPGA. The stopwatch/timer can display measurements in minutes, seconds, and hundredths of seconds on a 7 segment display.**

I. Introduction

The objective of this project was to create a Stopwatch/Timer that is capable of displaying tens of minutes, minutes, tens of seconds, seconds, tenths and hundredths of seconds on a seven-segment display. The user is able to interact through the implementation of a pause and reset functionality on the FPGA. The project has been made using the Nexys Artix7-50T board.

II. Methodology

The design and application of this project utilized concepts learned throughout the semester. Aside from a generic pulse counter and a few modifications to the inputs and outputs on certain design components, everything used had been seen before. To implement the logic, two counters, a multiplexer, four gen-pulses with a count of 10, two gen-pulses with a count of 6, a 3-to-6 decoder, a seven-segment decoder and a final state machine was utilized.

A. Clock Counters

Two clock counters were used in the implementation of the design. The Nexys Artix7-50T board has an internal clock that outputs a signal every 10 nanoseconds (ns), however for the design, a signal was needed to be output every 10 micro-seconds (ms). To do this, the internal clock of the board needed to be slowed down. So, a clock counter was used to regulate the signal to be output every 0.01 seconds.

B. Genpulse Counter

Originally, the group was planning on using BCD counters, alongside modulo-6 and modulo-10 counters in the design. After facing issues during simulation, a switch to a generic pulse counter was made. The gen-pulse was used six different times, each to represent an increment of time displayed on the seven-segment. To replace the modulo-6 counter, two gen-pulses were given a count of 6. Their range covers digits 0-5. Therefore, these covered the tens of seconds, as well as, the tens of minutes. Furthermore, to replace the modulo-10 counter, a four gen-pulses were given a count of 10. Their range covers digits 0-9. Therefore, these covered the hundredths of seconds, the tenths of seconds, ones of seconds and ones of minutes. Figure 1 depicts the logic followed for the counters.

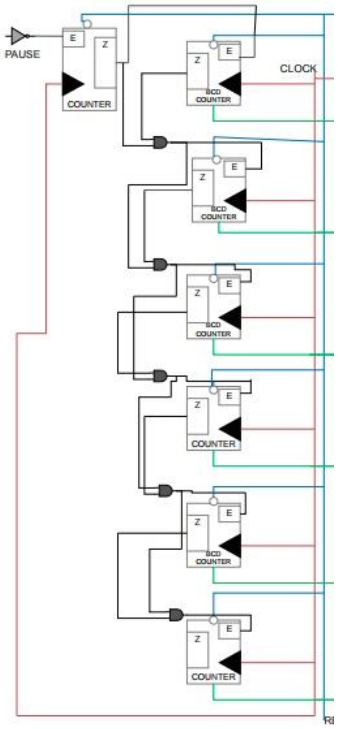

Fig.1. Counter Logic

C. Multiplexer

A multiplexer was used to take all the outputs from the counters and provide only one input to the seven-segment decoder. By using a multiplexer, the logic of the circuit was greatly simplified, as it eliminated the need to use a separate seven-segment decoder for each output of the gen-pulses.

D. Finite State Machine (FSM)

The finite state machine regulated by a clock counter and acted as a way to ensure the that each digit in the timer

changed properly. Furthermore, it was used to regulate what would be input to the 3-to-6 decoder, and thus be displayed on the seven-segment display. It also provided would be input to the selector of the multiplexer. The figure below shows the logic it followed in ASM form.

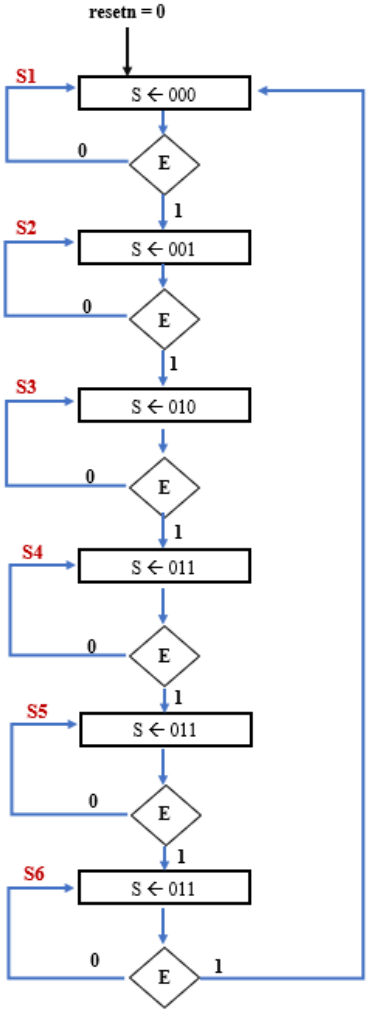

Fig. 2. FSM

E. 3-to-6 Decoder

A 3-to-6 decoder was used because only six digits of the seven-segment decoder were needed to display the timer/stopwatch output as its range is from 00:00:00 to 59:59:99.

F. Seven-Segment Decoder

A seven-segment decoder was used to take in the output of the multiplexer and send it to the seven-segment display. Figure 3 shows the logic of the multiplexer, FSM and decoders.

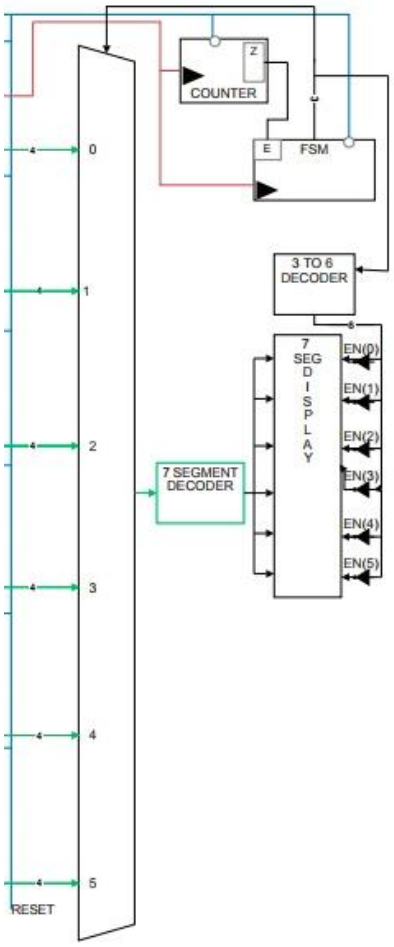

Fig. 3. Seven-Segment Logic

III. EXPERIMENTAL SETUP

The Nexys Artix7-50T board has been used to implement the design and well as Xilinx Vivado to write the code.

1. Design

Figure 4 shows the entire logic implemented onto the board. As mentioned before, it utilizes two counters, a multiplexer, four gen-pulses with a count of 10, two gen-pulses with a count of 6, a seven-segment decoder and a finite state machine.

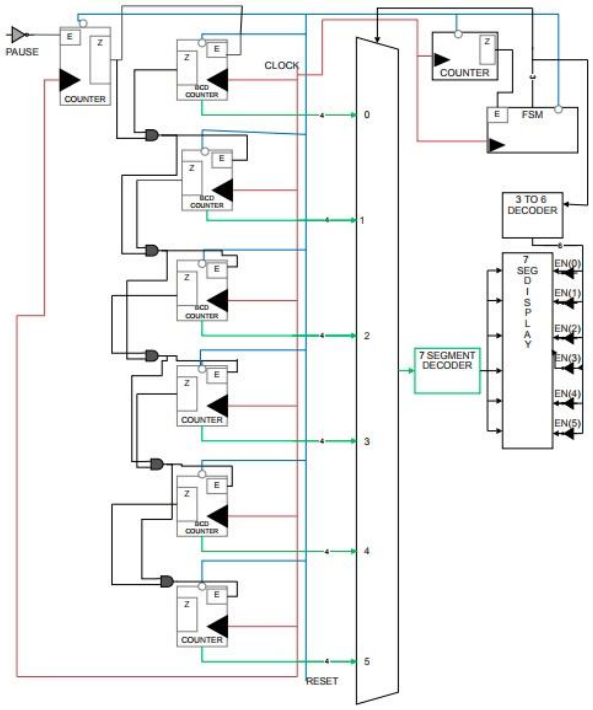

Fig.4 Digital Stopwatch Logic

For user interaction, a button to reset the timer, as well as a button that allows the timer to start/stop, has been integrated. Figure 5, shown below, depicts the FPGA that was used and boxed in red, are the components on the board that have been used in the design.

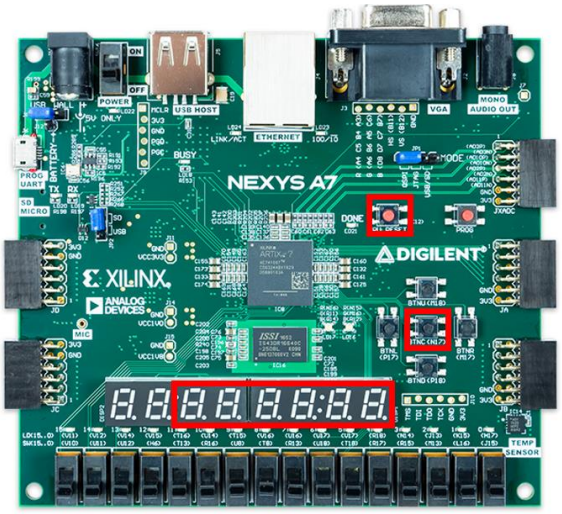

Fig.5. Nexys Artix-7 50T Board

2. *Testing and Implementation*

To verify the functionality of the project, Vivido was used. The testbench will be created so that the user can test the start, stop and reset capability of the circuit. Figure 5 and Figure 6, shown below, depict the behavioral simulation that was formed upon proper coding.

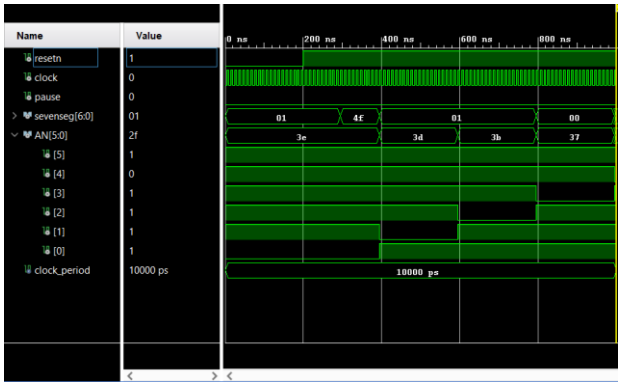

Fig.6. Timing Diagram 1

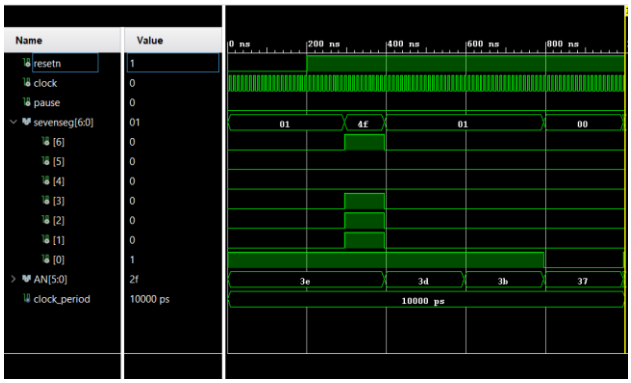

Fig.7. Timing Diagram 2

It is important to note that in order to capture the behavioral simulation, it was needed for the clock speed to changed, significantly. The board operates using an internal clock speed of 100MHz. So, had the group left the speed as it was, they would not have been able to receive any data from the behavioral simulation. In Figure 8, the commented section (text shown in light gray) is what the group needed to have the clock set to when generating a bitstream and uploading to the board. The section that is uncommented (text with color) is what had to be used in order to enable a behavioral simulation.

```
-- Counter: 0.01s
--gz: MYGENPULSEI generic map (COUNT => 10**6)
gz: MYGENPULSEI generic map (COUNT => 10)
    port map (clock => clock, resetn => resetn, E => PNOT, z => BCDIN);
--    z <= '1'; -- only for simulation

-- BCDIN<='1'; -- why is it '1'?

-- Counter: 0.001s
--gz: MYGENPULSEI generic map (COUNT => 10**5)
gfsm: MYGENPULSEI generic map (COUNT => 20)
    port map (clock => clock, resetn => resetn, E => '1', z => IICOUT);
```

Fig.8. Accounting for Clock Speeds

IV. RESULTS

The group was able to achieve the final design, they set out to accomplish. As mentioned before, a few modifications were needed to be made. To ensure a proper change between digits of the display, the group decided to switch to using generic pulse counters. This allowed for their code to be simplified, taking away the need for BCD, modulo-6 and modulo-10 counters.

The group faced many issues upon testing, as they expected, and spent a majority of their time troubleshooting the design. Finally, finding errors in the XDC file, their multiplexer, as well as, their 3-to-6 decoder, the group was able to resolve the issues and have a design that was fully functional.

Some areas where there could have been improvement regard a more intuitive design. The most apparent, and simple fix that the group realized after getting their design working, was the placement of the pause function. The group utilized a button for this functionality and now understand that using a switch would have made more sense, as it holds the value for the user, without having them have to hold it down themselves. Furthermore, had there been more time, the group could have implemented an LCD display to show the time, rather than the seven-segment display. Additionally, they could have implemented more user-interaction capabilities on the board. This could have been done by using a lap function.

CONCLUSIONS

This project reiterated much of what was learned in the course and further advanced the group's understanding of VHDL and using it to program FPGA's. While working on the project, the group faced several roadblocks. Luckily, they were able to overcome them through a great deal of time and efforts spent troubleshooting. While stressful at times, this helped better their abilities in problem-solving, as well as, knowing how to work in a team. Despite the issues, the group found the project to be a great learning experience. They enjoyed understanding how a common tool, used every day, operates and what the logic behind its functionality is. The group hopes to further their knowledge in using FPGA's, and eventually, be able to design something more complex and intuitive.

REFERENCES

[1] Llamocca, D. (n.d.). VHDL Coding for FPGAs. http://www.secs.oakland.edu/~llamocca/VHDLforFPGAs.html