Digital Stopwatch

Robert Kozubiak, Shawn Waite, Peter Isho
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
rjkozubiak@oakland.edu, smwaite@oakland.edu, pdisho@oakland.edu

Abstract—**The purpose of this project is to implement a digital stopwatch and display it on a Nexys 4 FPGA board. This is possible by programming the logic in VHDL, which is a strong language for digital logic. This stopwatch works perfectly with no noticeable time discrepancies. The only recommendation would be to use a different method of time measurement if accuracy down to nanoseconds is necessary.**

I. Introduction

Having an accurate way to tell time is a necessity in all aspects of life. Having the convenience of a stopwatch allows for untold tales of greatness. To achieve the design of a stopwatch, multiple components needed to be incorporated perfectly. The internal design required eight counters, one 8-to-1 multiplexer (MUX), one 2-to-1 MUX, one 7-segment decoder, one 3-to-8 decoder, and one FSM.

Utilizing a cascading method that connects each counter to the next one, the time discrepancy is gone. This will produce a time that will be almost exact to the actual time. The miniscule time delay would be in the nanoseconds due to the registers used for the lap function. The lap function acts as follows: As a switch is activated, the current time will be displayed for a brief period while the true time will be running in the background. When the switch goes down, the real time displays.

This project can be used in any situation where time is critical. Cooking, physical activity, and lab experiments are merely three different ways this could be used. Whether or not someone would want to carry this board, and a computer, with them is another question.

II. Methodology

Our stopwatch has all of the basic functionalities that would be expected, including start, pause, lap, and reset. A switch starts the stopwatch, and the same switch pauses the time. The time will be displayed on eight, 7-segment displays but only six of them will be utilized. The other two, the two leftmost displays, will always display 0. A max time of 99 minutes and 59.99 seconds can be achieved, and anything after that will reset the clock to 0 while continuing the stopwatch. The second switch that is used controls the lap function. This will display the current time to the 7-segment displays as long as the switch is on. The main feature of this is that the real time is still running so as soon as the switch is turned off the actual time will be displayed. The last function is to reset the time to allow the user to use the stopwatch again.

A. Counters

Eight counters were used to make this stopwatch work, as shown in Figure 1. Our stopwatch counts in increments of 10 ms, but the board has a native clock signal of 10 ns. To adjust for this time discrepancy, an initial counter was placed that produces one output signal for every 1,000,000 input signals. This counter essentially changes the clock signal from 10 ns to 10 ms and allows the rest of the stopwatch to proceed. This output signal will go two places. The first will be the hundredths place for the seconds. This counter, along with the next five counters, has three inputs and two outputs. An enable, reset, and clock are the inputs, while the outputs are one 4-bit signal and one 1-bit signal. The enable tells the counter when to increment by one, the 4-bit output produces a signal that reflects the current count, and the 1-bit output acts as a signal that tells the next counter to increment by one. When the count is at 9 and it goes to 0, a signal is also produced that goes to the next counter's enable at the next clock signal. This type of cascading counting continues down to the last counter, where the process will begin again. The fourth counter is for the second's tens place so the count goes to 5 before sending out a signal to inform the next counter to increment. The last counter deals with the FSM and will be discussed in a later section.

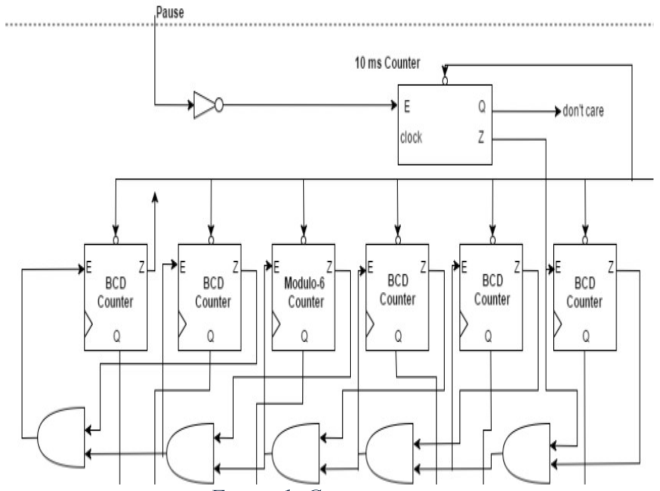

Figure 1. Counters

B. Registers

The 4-bit signal that is sent out from each counter will go to a register and a 2-to-1 MUX, as shown in Figure 2. This step is for the lap function to work properly. There are six total registers and each of them have the same inputs and outputs. The inputs are an enable, reset, a 4-bit input, and a clock, and the output is a 4-bit output. The output goes to the same 2-to-1 MUX that the output of the corresponding counter goes to. The select of each of the six, MUX's is connected to a switch which represents the lap function. When the switch is down, the output of the MUX is the counter's signal. When the switch is up, the lap signal will be the output. Another key component in this is the enable of each of the registers. They are the opposite of the switch status, so when the switch is down the enable is 1 and vice versa. This allows the same 4-bit signal to be outputted from the register if the lap function is activated. If the switch is down the enable is 1, so the lap signal is always in sync with the actual time. The output of each MUX will be discussed next.

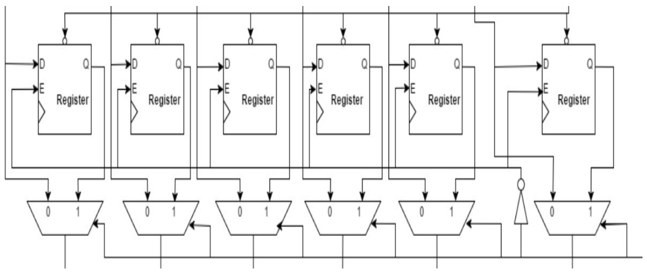

Figure 2. Register and 2-to-1 MUX for lap function

C. 8-to-1 Multiplexer

The signals from the previous MUX's are eight inputs to this 8-to-1 MUX, as shown in Figure 3. This MUX has all the data that now needs to be displayed. Firstly, the eight most significant bits are all 0 as this will correspond to having 0's on the two leftmost displays. The output goes to a Hex to 7 segment decoder but first comes the question of which input to select. This will come from a combination of a counter and a FSM. The FSM being used is essentially a counter that counts up to 7 and then resets to 0. Each count, which is in the form of a 3-bit signal, gets sent out as the select to the multiplexer, and that signal selects which 4-bit signal to output to the decoder. Since there can only be one display on at a time, a way around this is to switch the display that is on very quickly. The human eye won't be able to tell the difference so it will seem as though all the displays are on. To select which display to use, the same 3-bit signal that acts as the select will be sent to a 3-to-8 decoder. The output of this signal is then passed to a not gate followed by the AN signal, which decides which display to turn on. The FSM's enable comes from a similar counter to the initial 10 ms counter. The only difference between that counter and this one is that it is a 1 ms counter. This is so the outputs and displays are being updated rapidly which makes it seem as though all the displays are on.

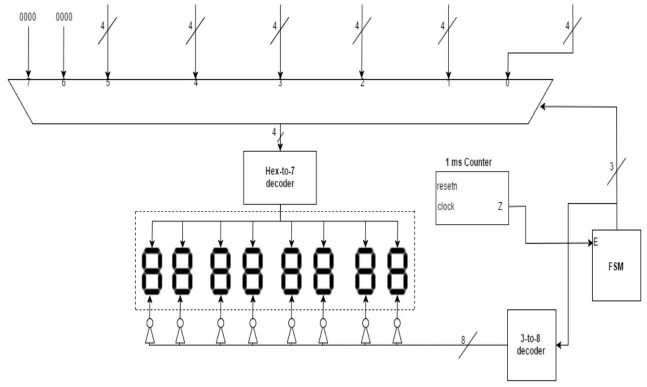

Figure 3. The components needed to display the 8, 4-bit signals.

The finite state machine has the same functionality that a Modulo-8 counter would have. There are eight states that correspond to an output of 000 to 111, as shown in Figure 4. These are for both the selector of the MUX and the input of the 3-to-8 decoder. The thing that allows the signal to go to the next state is the enable, which is connected to a 1 ms counter. If the enable is 0, the state stays the same until it is 1. This process repeats until the user decides to turn off the stopwatch.

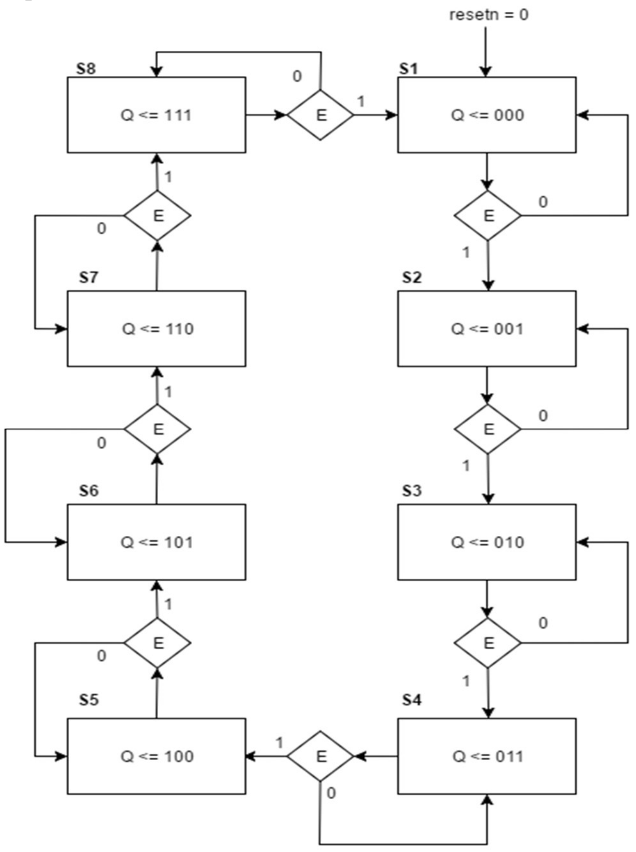

Figure 4. FSM (ASM Form)

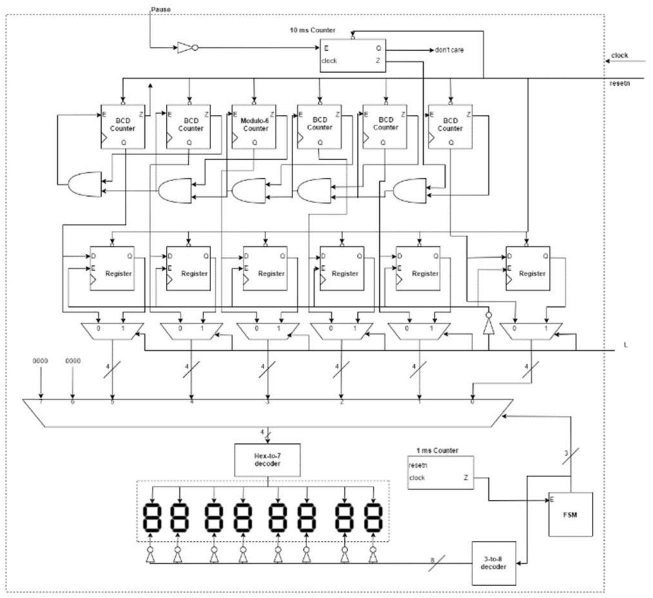
Figure 5. Circuit Design

III. Experimental Setup

The hardware used for this project is the FPGA board known as the Nexys 4 Artix-7. The FPGA has eight 7-segment displays that are the basis of our stopwatch project. The software used to program the Nexys 4 Artix-7 FPGA board is called ISE Design Suite 14.7. This software allows an interface with the FPGA and also programs the various functions of the stopwatch. Using an internal clock, the FPGA keeps track of milliseconds, seconds, minutes, and hours and displays them on the 7-segment displays. Using a switch on the FPGA, the lap function "pauses" the 7-segment display while the clock is still counting internally. The expected results of these functions, is a working stopwatch that can keep time accurately with and without the lap function enabled. To test that the clock is accurate, a working timer will be compared to the output of the stopwatch to see if there are any discrepancies in the timing of the board. The expected results should be extremely close to the actual time. The only time discrepancy would be when the lap function is activated. There would be a miniscule delay, somewhere around 10 ns, that wouldn't be noticeable since this stopwatch is incrementing with 10 ms. Nonetheless, this should be kept in mind if this stopwatch is ever used to measure time with an accuracy of nanoseconds.

A testbench simulation was created to test the functionality of the counters and the lap function, as shown in Figure 6. The following picture shows three significant signals, which include the output of the counters (d), the lap signal (l), and the output of the 2 to 1 MUX. As the lap switch is down, all three signals are identical. When the switch is up, the counter output continues while the lap signal stays the same. The output of the MUX is the same as the lap signal, and this signal will eventually be on the 7-seg displays. As the switch is pushed down, the signals, once again, become identical.

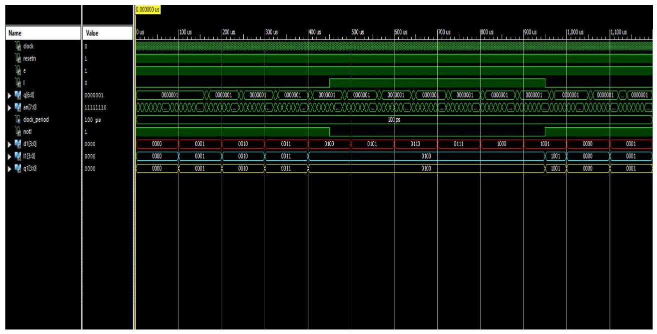
Figure 6. Lap Function Testbench

This next picture, as shown in Figure 7, is a testbench of the counters working. As they go to 9 they reset to 0 and the next counter increments by 1. This logic is implemented for every counter output in the program. This process is continued the entire time the stopwatch is on, so it is essential to have this working.

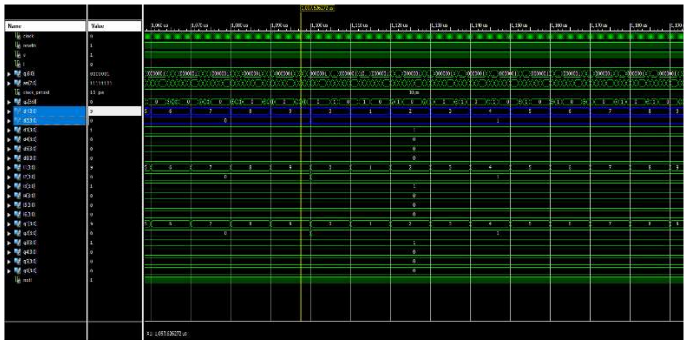
Figure 7. Counter Testbench

IV. Results

The results obtained met our expectations as there was no noticeable delay between this stopwatch program and a real stopwatch If measured down to the nanosecond, there would probably be a delay, but that isn't the case. The lap function seems to work flawlessly, even for long periods of time in between turning the lap input on and off. This is because of the two different signals used, one for the real time and one for the lap time. Even as the lap function is activated, the real time is always continuing with no hesitation. As the time passes its max of 99:59.59, it resets itself to 00:00:00 and continues. The reset button works as intended, as it forces every one of the counter and lap outputs to 0. The enable also works as intended, as the counters all turn on and off immediately as the switch is turned on and off, respectively. Overall, the results were sufficient and were obtained as expected in all aspects of the project.

CONCLUSION

This project gave the group a better understanding of digital logic combined with VHDL. Implementing this stopwatch required knowledge of multiple components, including counters, multiplexors, decoders, registers, and finite state machines. All of these needed to be connected with each other in a way to produce what was needed. A project like this is a perfect stepping stone into the field of embedded systems.

REFERENCES

[1] VHDL Coding for FPGAs http://www.secs.oakland.edu/~llamocca/VHDLforFPGAs.html