Digital Stopwatch

Designed in VIVADO Using VHDL

Elena Huizar, Emilee Otten, Patty Hasa
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
ehuizar@oakland.edu, esotten@oakland.edu, pattyhasa@oakland.edu

***Abstract*— This project's goal is to prove the group's understanding of digital circuit and VHDL concepts covered in this course. Through the combination of these concepts, a digital stopwatch, with capabilities similar to an ordinary stopwatch, was designed in VIVADO using VHDL. A detailed block diagram was created and a simulation of the stopwatch's functionalities was successfully conducted. Furthermore, this design was implemented on a Basys-3 board, resulting in a physical prototype of this stopwatch that successfully functioned as intended and as designed.**

I. Introduction

The goal of this project is to design a digital stopwatch in VIVADO using VHDL and implement this stopwatch on a Basys-3 Board. This stopwatch will measure minutes and seconds using a variety of counters and display these times on four seven-segment displays. This stopwatch also has start, reset, pause, count up, count down, and lap capabilities, similar to a traditional stopwatch or timer. These capabilities are possible due to the combination of logic gates, registers, multiplexers, counters, and a finite state machine (FSM). This digital stopwatch can be used as a component in a digital watch that contains other functionalities, like a display of the current time and an alarm.

II. Methodology

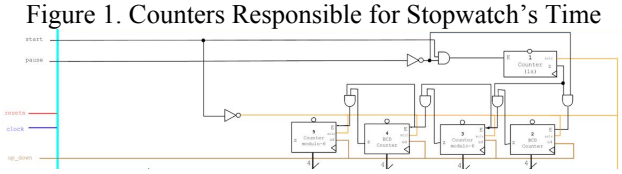
This project can be classified as a digital system, with seven inputs and two main components, the datapath circuit and the FSM. The seven inputs of this digital system are *start*, *pause*, *up_down*, *lap1*, *lap2*, *select_lap*, and *display_lap*, all programmed as switches. *Start* begins the stopwatch's counting of time when its switch is high (1). When its switch is low (0), the stopwatch resets to a time of zero minutes and zero seconds. *Pause* freezes the stopwatch's counting of time when its switch is high. *Up_down* controls if the stopwatch counts up or counts down in time. When its switch is high, the stopwatch's time will count up. When its switch is low, the stopwatch's time will count down. The lap inputs, *lap1* and *lap2*, act as memory states for the stopwatch. When *lap1* is high, the lapped time is recorded in a set of four four-bit registers. Similarly, when *lap2* is high, the lapped time is recorded in a separate set of four four-bit registers. *Select_lap* selects between the first and second laps in order to potentially display one of the lapped times. When *select_lap* is low, the first lap is selected. In contrast, when *select_lap* is high, the second lap is selected. *Display_lap* can display this selected lap on the seven-segment displays. When its switch is low, the stopwatch's time will be displayed on the seven-segment displays. When its switch is high, the selected lap will be displayed on the seven-segment displays. It is important to note that whenever these lapped times are displayed, the stopwatch continues to run in the background.

The datapath circuit can be broken down into the main counters, which are responsible for the stopwatch's time, and the circuitry responsible for the functionality and the laps and their displays. These elements, along with the FSM, are discussed in more detail in the following sections. Furthermore, the full block diagram for this digital system is included at the end of this report.

A. Main Counters

The stopwatch consists of six counters. Five of these counters are used to count the minutes and seconds of the stopwatch and are shown in Figure 1. Note that this digital circuit was adopted from one of Professor Llamocca's VHDL tutorials and revised to add additional stopwatch capabilities [1]. Two of these counters, labeled as 3 and 5 in the figure, are modulo-6 counters, meaning their count is from zero to five. Two of these counters, labeled as 2 and 4, are binary coded decimal (BCD) counters, meaning their count is from zero to nine. The fifth counter is a one second counter, labeled as 1, with a count from zero to 10^8. This counter allows the other four counters to count every one second, rather than every ten nanoseconds.

Figure 1. Counters Responsible for Stopwatch's Time



The inputs of all five counters are clock (*clock*), asynchronous reset (*resetn*), enable (*E*), synchronous clear (*sclr*), and an up/down input (*ud*). The counter outputs are a four-bit output (*D*) and a single bit output (*z*). Note that the output *D* of counter 1 is not important to the design or functionality of the stopwatch.

Sclr acts as the stopwatch's reset capability and is implemented using the start switch and a NOT gate. If *start* is high, *sclr* is low and the stopwatch will not reset. Similarly, if *start* is low, *sclr* is high and the stopwatch will

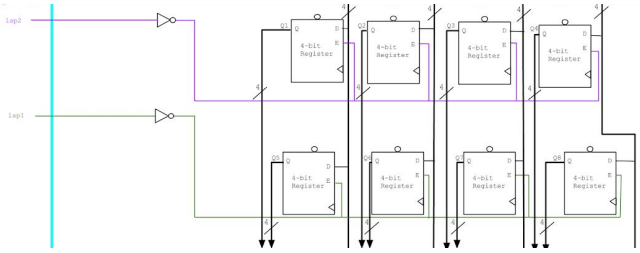
reset. The enable of counter 1 is controlled by an AND gate with *start* and NOT *pause* as inputs, acting as both the stopwatch's start and pause capabilities. This ensures that if *start* is high and *pause* is low, the enable of this counter is one and the counter will count. If *pause* is high, the enable of this counter will be zero and the counter will pause in its count.

The enable for counter 2 also acts as the stopwatch's pause capability and is controlled by an AND gate with counter 1's output *z* and NOT *pause* as inputs. This ensures that if *pause* is high, the enable of this counter will be zero and the counter will pause in its count. In contrast to counters 1 and 2, the enables of counters 3, 4, and 5 are not directly wired to *pause*. Instead, their enables are the previous counter's output *z* AND the previous counter's input *E*. This ensures that once a single counter reaches its maximum value, the next counter will be temporarily enabled to start counting.

B. Laps and Lap Displays

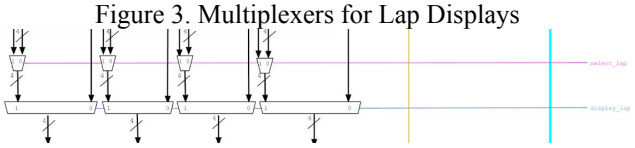
As previously mentioned, the four-bit registers of the digital system play a crucial role in the stopwatch's lap capability. There are a total of eight four-bit registers, with four responsible for recording the first lapped time and another four responsible for recording the second lapped time. These registers are shown in Figure 2. The enable of each register is crucial to the lap capability of the stopwatch. As discussed in class, registers have a memory state when their enable is zero. Thus, the enable must be wired so that it is zero when the user wants to lap a certain time. This was accomplished using a lap switch and a NOT gate. When the switch is low, the enable of each register is one and the stopwatch's time is not recorded into the memory state of the registers. When the switch is high, the enable of each register is zero and the stopwatch's time is recorded into the memory state of the registers. *Lap2* controls the enables of the first row of registers, whereas *lap1* controls the enables of the second row of registers

Figure 2. Registers Used to Implement Lap Function



Before displaying the lapped time, the user must select the first or second lapped time using the *select_lap* switch. *Select_lap* is the select line of a set of four two-to-one multiplexers. Thus, the first lap is selected when the select line is zero, and the second lap is selected when the select line is one. The outputs of these multiplexers are then fed into one of the inputs of another set of four two-to-one multiplexers. However, the select line of this set of multiplexers is *display_lap*. Normally, *display_lap* is zero and the stopwatch's time is selected to be displayed on the seven-segment displays. However, when the user sets the *display_lap* switch to high, the select line becomes one, and the lapped time is selected to be displayed on the seven-segment displays. These eight two-to-one multiplexers are shown in Figure 3.

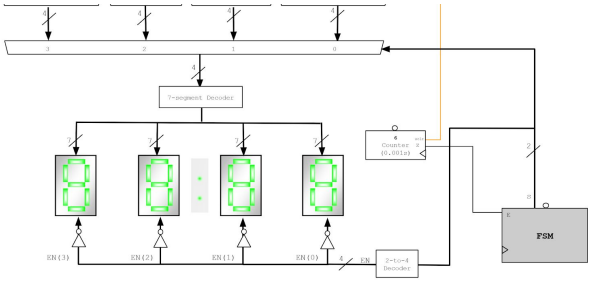
Figure 3. Multiplexers for Lap Displays



C. FSM

The FSM plays a crucial role in displaying the stopwatch's current or lapped time on the seven-segment displays. Because only one of the seven-segment displays can be used at a time, the use of the FSM gives the stopwatch the ability to use all four seven segment displays. This is accomplished through the circuit shown in Figure 4. Note that this digital circuit was adopted from one of Professor Llamocca's VHDL tutorials [1].

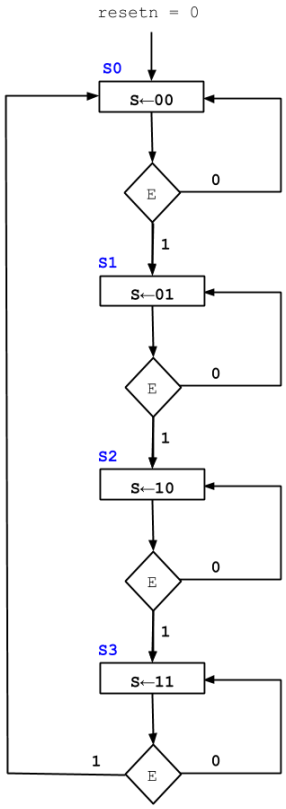
Figure 4. FSM Used to Choose the State



The enable of the FSM is controlled by the output *z* of the 0.001 second counter. This counter enables the FSM every 0.001 seconds, which is also equal to the refresh rate of the seven-segment displays. Thus, each of the four displays will be illuminated for 0.001 seconds and un-illuminated for 0.003 seconds. This is accomplished through a four-to-one multiplexer and a two-to-four decoder.

The Algorithmic State Machine (ASM) chart for the FSM was adopted from one of Professor Llamocca's VHDL tutorials and is shown in Figure 5 [1]. This ASM chart depicts the functionality of the FSM. As shown, *s* is "00" in state S0, "01" in state S1, "10" in state S2, and "11" in state S3. *S* is then used to control which multiplexor input is selected, imputed to the seven-segment decoder, and displayed on one of the seven-segment displays. *S* is also imputed to a two-to-four decoder, whose output is imputed to NOT gates before enabling or disabling the seven-segment displays. The NOT gate is crucial to accurately enabling the seven-segment displays due to their active-low nature.

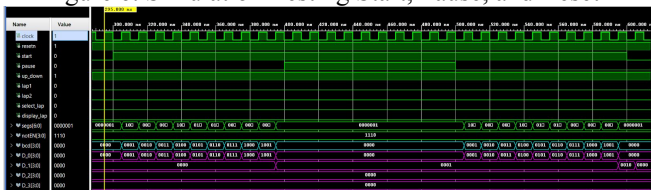
Figure 5. ASM for the FSM



III. Experimental Setup

To verify the functionality of this stopwatch, a testbench within VHDL was used. This testbench simulated various values for *start*, *pause*, *up_down*, *lap1*, *lap2*, *select_lap*, and *display_lap* in order to verify the various capabilities of this stopwatch. Because of the large number of inputs into the digital system, these capabilities were tested in four separate simulations. The first simulation, shown in Figure 6, tested the start, pause, and reset capabilities of the stopwatch.

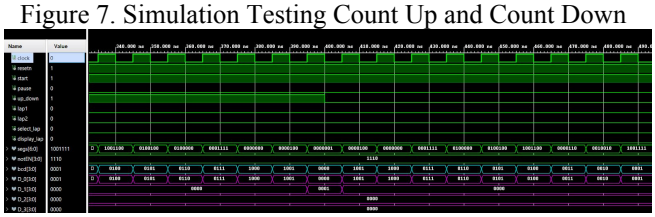
Figure 6. Simulation Testing Start, Pause, and Reset



Before *start* is high, the outputs of counter 2 (*D_0*), counter 3 (*D_1*), counter 4 (*D_2*), and counter 5 (*D_3*) are "0000". When *start* is high, the counters act as expected, and the counting of time begins. Additionally, when *D_0* is "1001", *D_1* becomes "0001" on the next positive edge of the clock. This confirms that the output *z* of each counter as well as the enable of each counter behaves as expected. Furthermore, the counters pause in their counting when *pause* is high. When *pause* is low, the counters continue in their count. Lastly, *D_0*, *D_1*, *D_2*, and *D_3* all become "0000" when *start* is low, confirming the stopwatch's reset capability.

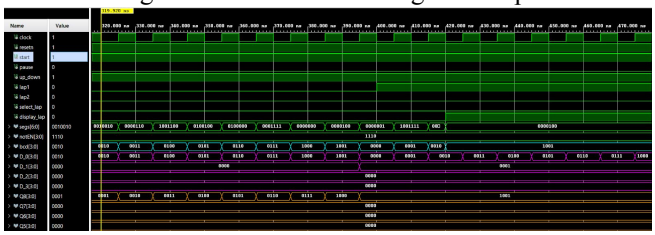
The second simulation, shown in Figure 7, tested the count up and count down capabilities of the stopwatch. As expected, the counters counted up when *up_down* was high and counted down when *up_down* was low.

Figure 7. Simulation Testing Count Up and Count Down



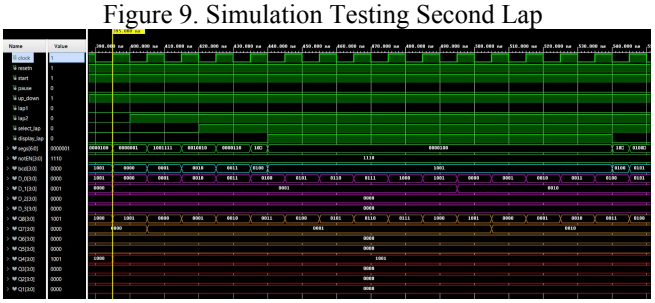
The third simulation, shown in Figure 8, tested the functionality of the first lap using *lap1*. Before *lap1* is high, *Q8* records *D_0*, *Q7* records *D_1*, *Q6* records *D_2*, and *Q5* records *D_3* on every clock tick. However, the value of *Q8* is always one less than *D_0*. This was assumed to be due to the high clock frequency and was left to be tested on the Basys-3 board before modifications to the circuit were made.

Figure 8. Simulation Testing First Lap



When *lap1* is high, the value of each counter is stored in the corresponding memory state of the register. This is then displayed when *display_lap* is high. This can be seen as the output *bcd* is equal to *Q8* when *display_lap* is high. The fourth and final simulation tested the functionality of the second lap using *lap2*. Before *lap2* is high, *Q4* records *D_0*, *Q3* records *D_1*, *Q2* records *D_2*, and *Q1* records *D_3* on every clock tick. As with the first lap, the value of *Q4* is always one less than *D_0*. Despite this, it is still clear that the value of each counter is stored in the corresponding memory state of the register when *lap2* is high. Then, *select_lap* and *display_lap* were both set to be high in order to display the lapped value. This can be seen as the output *bcd* is equal to *Q4* when *display_lap* is high.

Figure 9. Simulation Testing Second Lap



Not much attention was given to the outputs *bcd* or *segs* within each of these simulations. Due to the length of the simulations, the state of the FSM was always zero; therefore, only *D_0*, *Q4*, and *Q8* were selected as *bcd* and imputed to the seven-segment decoder to become *segs*. Because the four-to-one multiplexer and seven-segment decoder were modified from one of Professor Llamocca's VHDL tutorials, these outputs were assumed to be correct [1].

IV. Results

Lastly, the stopwatch was implemented on a Basys-3 board and can be seen at this link: https://drive.google.com/file/d/1qtMjPt_YUe1DLOftuVjQ_FgYC6ToEedz/view?usp=sharing. As expected, the stopwatch and its features functioned as designed. When the *start* switch was high, the stopwatch started counting, and the count was displayed on the seven-segment displays. If the *start* switch was low, the stopwatch was reset to a time of zero minutes and zero seconds. When the *pause* switch was high, the stopwatch froze its count at the current time. Once the *pause* switch was low, the stopwatch continued counting. If the *up_down* switch was high, the stopwatch counted up in time. In contrast, if the *up_down* switch was low, the stopwatch counted down in time.

In terms of the laps, when the *lap1* or *lap2* switches were high, the current counter value was recorded into the stopwatch's memory. The correct lap, influenced by the *select_lap* switch, was then displayed on the seven-segment displays when the *display_lap* switch was high. Note that when either of the laps was displayed, the stopwatch continued to count in the background. Once the *display_lap* switch was low, the stopwatch's current time was displayed. If the *lap1* or *lap2* switches were low, there was no lapped time, and any previously saved times were "deleted".

As expected from the simulations shown in Figure 8 and Figure 9, the lapped time occasionally was one second behind the actual time that the user wanted to lap. This issue was not unexpected after understanding the reasoning for this error. During class, it was discussed that when implementing a synchronous component into any design, there would be a slight delay within the component's output. Professor Llamocca also reiterated that the output would appear after the clock's rising edge, not right on the clock's rising edge.

Once tested on the Basys-3 board, this issue was found to only occur if the *lap1* or *lap2* switches were set high too close to the transition in time from one second to the next. However, the lapped time would equal the correct time if the *lap1* or *lap2* switches were set high immediately after the transition in time from one second to the next. This method was employed in the testing of the stopwatch shown in the video, resulting in a lap capability that appeared to function as one would expect. However, this issue should be resolved, such as through the addition of a set of four four-bit adders, in order to be used within a more complex design, like in a digital watch.

Conclusions

As discussed, a digital stopwatch was designed with the knowledge learned in class about digital logic and VHDL. With the addition of adding two different lap functions and an up-down counter, this project has become an extensive version of a regular stopwatch seen on a personal phone or used in races. Furthermore, the combination of logic gates, registers, multiplexers, counters, and a finite state machine was implemented on a Basys-3 board to create a physical prototype of the stopwatch. After conducting the implementation of these components onto the board, it was concluded that everything worked as intended, with the exception of one issue that could be avoided with patience or a more complex design. With this issue in mind, as stated previously, the user must use the *lap1* or *lap2* switch immediately after the transition of time or the circuit design will need four four-bit adders in order to resolve this error of the delayed lap. Besides this minor issue, the results were exactly as expected and the only improvement that could be made would be the user interface of this stopwatch. This project could have a more user-friendly display with the switches properly labeled and a bigger seven segment display. Through the addition of this user-friendly display, this stopwatch would have the ability to be used in a variety of real world applications.

References

[1] D. Llamocca. Digital System Design [PowerPoint slides]. Available: http://www.secs.oakland.edu/~llamocca/VHDLforFPGAs.h

Figure 10. Stopwatch Block Diagram