VHDL Stopwatch with Lap ROM Functions

List of Authors (Ryan Paye, Nathan Fazecas, Jenna Grajewski)
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
e-mails: rpaye@oakland.edu, nfazecas@oakland.edu, jennagrajewski@oakland.edu

***Abstract*—In this project, the group utilized aspects and skills developed during the winter semester of 2022 at Oakland University in ECE 2700 Digital Logic Design to develop a stopwatch using VHDL. The Digital Design was outputted to a FPGA (Nexys-A7 50t) using VHDL code inputted into Vivado. For this project, the group decided to implement a stopwatch that incorporates the use of a lap function. (Inputs outputs) In this project, the experimental results strayed away from the expected results but in such a manner that was not entirely not feasible. In the future, given time constraints it would be nice to read further information about how errors might interact with the stopwatch.**

I. Introduction

For this project, a simple digital stopwatch was created. This design was optimized for the Nexys-A7 50t which incorporates the use of many buttons, switches, and a 7-segment display. This stopwatch system incorporates the use of many circuit/programming blocks developed from material from each lab and class session to ultimately display on the 7-segment displays of the board. This project heavily relied on aspects, such as counters, ROMs, and a litany of systems that incorporate FSM's. The required components of the stopwatch will be further illustrated in the appended sections below.

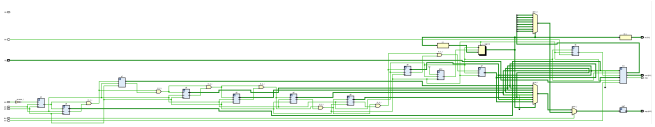
The motivation behind this project was to effectively design a functioning stopwatch with consistent and useful information given from the device. The main purpose was for the stopwatch to be reliable and completely functional at any point in its function. The dependability of the laps and the reset functions were focused on to ensure absolute functionality.

In further inspection, with simple adjustments to some aspects of the stopwatch, there might be due to the high precision of the counters, this could be useful for things such as timing a wide variety of competitive sports if presented in a more compact form of hardware that utilizes a 7-segment display.

Think of the rest of the report as an expansion of some of the points in the introduction.

II. Methodology

All in all, this project closely mirrors the functions of a basic digital stopwatch that can be found in a store. To correctly design the stopwatch, one must take into account both the functionality and ports used.



These should be functionally similar to how a stopwatch operates and what it outputs. On a stopwatch there are three major inputs: a start/pause, a reset, and a lap input.

How might this be incorporated into the top file? Two questions arise from this accordingly, how can aspects of design be used to implement this, and furthermore, how can aspects of VHDL coding be used to develop a streamlined version of the design without much error? In order to do this, in the top file a few custom circuit blocks were designed to be implemented into the development of the stopwatch. Overall, a Count Circuit is connected to the inputs of the Lap ROM, and it is connected to a 7-segment display serializer which is connected to the Lap ROM along with the count values. Furthermore, to increment the lap a debounced push button must be added to carefully increment the encoded bits of the lap ROM.

A. COUNT BLOCK

Perhaps the most important aspect of a stopwatch was the count section. Since a stopwatch has much to do with the assortment of a wide variety of counters, a

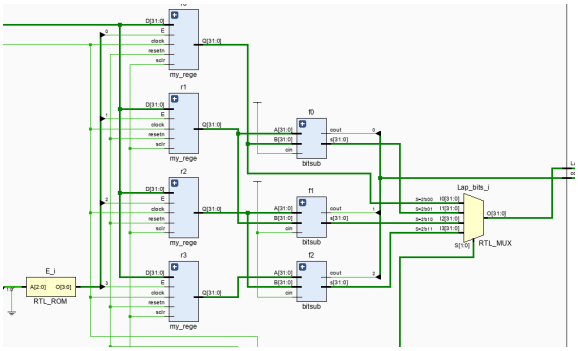
well crafted counter block must be designed to implement this clock. To design this, a wide variety of counters were incorporated so that the stopwatch could display the correct time format. Furthermore, after each counter reaches the highest bit of the count, the z input is passed through an and gate with the previous enable to enable the next counter. In this project 2 different counter architectures were used to develop the count. This consisted primarily of 1.) 6 BCD counters and 2.) 2 modulo-6 counters.

The BCD counter was used when the desired count was to be incremented until the output reached 9. This then causes the z value to go high.

Modulo-6 counters were used when the desired count was to be incremented until 6. This was used primarily when the minute or hour time was expected to increment by 1. Like the BCD counter, the value goes high when the integer is 6.

B. LAP ROM

The lap input was used with intents to output the lap function. This function was perhaps the most challenging and intricate aspect of the stopwatch to design. Before the function could be developed, research was needed to be done on the functionality of lap functions. When comparing this with the variety of previous examples from this class, it was decided that a ROM should be designed.

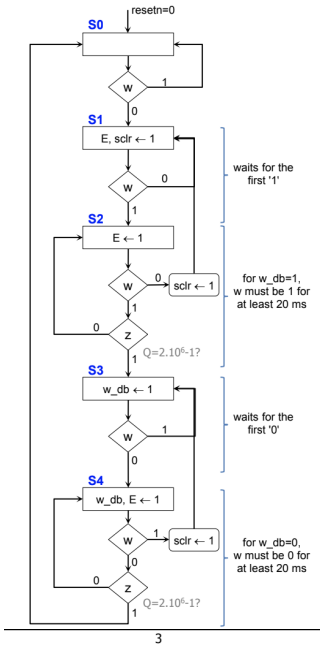


This design was similar to what was designed in lab 5, but it would have to be manipulated to work in the project. Instead of using an address as the encoded bits of the decoder input, the lap number should be encoded. To incorporate this as a function of the lap enable button, a counter must be implemented with a push-button enable. Once the enables are given, this enables the different register which correlates to each lap. Furthermore, to get the true lap split values, three 32-bit subtractors were needed so that the previous lap time could be subtracted from the current lap time, which would give the split as a result.

C. DEBOUCNER

Prior to implementing a debouncer, there were some errors with the lap times. If the lap enable were to be left as a simple push button, this would usually result in a wide variety of cycling errors in the lap counter due to button bouncing at the boards high clock frequency 100 MHz. Using this system along with the clock divider in the lap counter, the lap increment was designed to be smooth with limited chances for unwanted cycling.

Button bouncing is due to elasticity between the connection of the two metal contacts. This causes a lot of noise, and this can be seen in the form of bouncing between the two connection points. This can last for up to 20 ms until there is a uniform signal. This is much too slow for the clock's period of 10 ns. To counteract this a FSM/ASM architecture was used to implement a smooth transition between debounced values.



This state machine relies on the assortment of the input values of the push-button enable value. When the push button remains high from S1 to S2 to, S3 comes with the output of the debounced signal going

high. This then must go low from S3 to S4 and stay stable from the debounced value to go low.

III. Experimental Setup

To implement this, the optimal board used for the design was a Nexys A-7 50t. In the constraints, 4 push buttons were implemented, and 3 switches were implemented. These inputs all go to the different pathways and are outputed to the enables and the segs LED of the 7-Segment Display.

IV. Results

This project was exceedingly stressful at times to implement the functionality of the stopwatch. Most of the time developing this project was spent eliminating warnings such as multi driven nets. As such, the optimization of the circuit seemed to possibly cut away to some features. This caused a slight deviation in the way the lap values were formatted during the final lap enable.

https://youtu.be/KVwZg4nt2ps

Table of Expected Results

lap #	Expected	Real
lap 1	1:35	1:34
lap 2	2:52	2:49
lap 3	5:17	5:09
lap 4	10:11	XXXXXXXX

More discussion should be conducted on whether the discrepancy was caused from a multi driven. If a read write function were to be correctly implemented into the Lap ROM, this may have caused more favorable results when switching from 4 laps. Overall, this error remains a challenge to pinpoint with direct accuracy.

Programming was then possible on the board to display the clock which displayed each lap. This is related to much learned whilst in class. For example, using a 7-Segment display in our labs 5 and 6.

Conclusions

In conclusion, this final project made us more knowledgeable of all the aspects of circuit design and VHDL code in Vivado. Formatting was much easier when using these different techniques. This was reinstated from the aspects of collaboration done within the group. The techniques that were used included Streamline code and design to reduce error during implementation and Bitsteam Generation. This helped our code look neater and helped us stay organized. Generic Mapping was used which made it much easier to recall one class in the top modules than having to recall multiple counter components of different integers. A lot was learned about how to troubleshoot with a wide variety of errors and critical errors. The number of bits used was a big issue that was overcome when writing the code, but eventually got figured out. With doing this over, making a stopwatch with more functionalities that involve more elaborate designs would be a higher focus and desire.

References

[1] D. Llamocca, “Unit 7 – Introduction to Digital System Design,” Reconfigurable Computing Research Laboratory (RECRLab), Electrical and Computer Engineering Department, Oakland University. [Online]. Available: http://www.secs.oakland.edu/~llamocca/Winter2021_ece2700.html.

[2] D. Llamocca, “VHDL Coding for FPGAs Unit 7,” Reconfigurable Computing Research Laboratory (RECRLab), Electrical and Computer Engineering Department, Oakland University.[Online].Available:http://www.secs.oakland.edu/~llamocca/Tutorials/VHDLFPGA/Unit%207.pdf

[3] D. Llamocca, VHDL Coding for FPGAs. [Online]. Available: http://www.secs.oakland.edu/~llamocca/VHDLforFPGAs.html.

[4] *Button Bounce - Clemson University*. (n.d.). Retrieved April 19, 2022, from https://people.cs.clemson.edu/~yfeaste/855Assignments/presentations/team6-button-bounce.pdf

[5] *Digital Stopwatch - Oakland University*. (2021, April). Retrieved April 19, 2022, from https://www.secs.oakland.edu/~llamocca/Courses/ECE2700/W21/FinalProject/Group1_Stopwatch_presentation.pdf