Memory Match

Simon Says Children's Game

List of Authors (Kevin Harper, Alex Rice, Ravi Prajapati,Joey Georgees)

Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
Emails:
kharper2@oakland.edu,Alexrice@oakland.edu,rprajapati@oakland.edu,jgeorgees@oakland.edu

Abstract—

The project our group is creating for ECE-2700 is a memory match game that runs on a Nexys A7-50t board coded with vivado. The purpose of this project is to learn how to make simple video games that are fun to play to encourage others to create their own for a fun learning experience. We started by looking over examples we had from the teacher in the hand out for the group project in the beginning of the semester. Once we saw the simple game option, Which was to build a simple game with the board and program from class, we chose that idea. Next was picking the game we wanted to create which was a much more challenging part of the project than one might think because each of the students has their own strong areas that they are confident in doing and it doesn't always match up with other students. We settled on Memory match because we were able to come up with a simple block diagram quickly that we knew we were able to build. Major Findings of this project are the PRNG that was used in the project. We had to learn everything for that online from what it is why we need it and how it is used. A very useful thing we learned quickly was being able to read through other programs that were made for other things and being able to understand what was going on in the code. The last major thing we have learned from this project was learning to work with everyone during a pandemic, all having different lifes, jobs to work, and bills to pay. We had to be understanding and helpful to each other to be able to do what we need to do in a timely manner. Our main recommendations for creating this project is to first read as much as you can about what you are making before you start because if you dont you will be working backwards changing things you were positive that were going to work but you come to find out it won't so you have wasted time creating something just to get rid of it. Knowing what everything needs to do will cut the time wasted creating things you don't need in half.

I. Introduction

The Scope of this Project Included but not limited to Methodology, Experimental setup, results, conclusions and references for anything that helped us create this project. The project for ECE-2700 was created with the program vivado and nexys a7 50t board. The Project was created to create a project that could be fun to build and fun to use after it was built. Being able to accomplish both of these goals included building something that is doable by other students and a game easy

enough where anyone would be able to play it. This way the game is something that you are able to show to other people of all ages and they are not confused by what you have built. The implications of this project are that it could be made into a lab and become something that students can build and show their families, that way they understand what you built. The Two things we used that we had to teach ourselves that we did not learn in class were the button debounce which I have made in another class. The second was the PRNG which was the harder part and took a lot more research and understanding of what it is and how it works. The button debounce keeps the project from taking multiple inputs from a single button press and the PRNG created the code for each level.

II. Methodology

The Project Memory match game is based on the Simon says, this is quite an interesting idea how exactly, it can be created the same game in digital design system, which is operated by A one Nexys-A7 50T board, and USB cable that plugged into the computer. The heart of the game is Coding, which is happening inside the Vivado software.

the methodology of the Memory match is as following:

Once the start button is pressed, the Pseudo random number generator (PRNG) generates the value from the defined range in the code. We had to find PRNG online and use it not copying it directly but making sure that we understood what the original maker was using it for and how we could modify it to make it our own. We use the code for the PRNG from reference 6. Then the signal passes through priority encoder and present the value on the 7-segment display then expect the user's input after the user's input, the signal got sent out to Multiplexer and display the value on the 7-segment display, the user input value has to be in same order, and get the feedback either the she won or loose, the register keeps track of the stages that the user is in, by feedback system, and display the levels of the user.

A. Design Methodology

//Fsm; (I'm just typing this here while i have the time guys, we can fit it in where it needs to go with the rest of this section later).//

The finite state machine, or control circuit of our design project has nine states. The initial state S0 has our program begin by flashing the word begin written as "bEgn" on the seven segment display of the A50-T. The state stays S0 unless the variable start is true. Start becomes '1' or true when the center button is pressed by the user. When start is '1' the FSM moves on to S1. In S1 the pseudo random number generator (PRNG) is enabled and the timer is initiated. From S1 the next state is always S2. In S2 the timer is loaded and enabled, and the level counter is enabled. From S2, the next state is always S3. In S3 is when the level number and pattern from the PRNG will begin being displayed to the user. The PRNG will output the first step in the pattern to be displayed, it will then wait for a specific amount of time and if the sequence is not complete for that level it goes back to S3 to generate and display the next step in the sequence. Once the sequence for that level is complete, the state moves on to S4. S4 stops the PRNG and holds the memory of the sequence to check for a match from the user's inputs. From S4, the next state is always S5. In S5 is where the user begins entering the sequence to see if they can match the output from the PRNG. Each input is checked individually to see if it is a match. If at any point the user presses a wrong button, the next state is S7. If all of the inputs from the user match the expected sequence, the next state is S6. In S6 since the user entered the correct sequence the display shows "pass" to alert the user they have completed the level. If the user is at the max level of the game the next state is S8, else the next state goes all the way back to S1 where the level is incremented by one and the whole process is restarted for the next more difficult sequence. Since S7 happens when an incorrect sequence is entered by the user, "fail" is displayed to the user and then the next state is S8. Once in

S8, either the max level has been reached or an incorrect sequence was entered and so “End” is flashed on the display and the state goes back to S0 flashing “bEgn” until the user presses the center button to start a new game.

B. Experimental Methodology

Unfortunately, we don't have anything substantial to experiment with just yet. Not being able to get together and write VHDL code to troubleshoot and fix bugs within the code is very difficult right now since all the members don't have access to VHDL but we plan on finding solutions around these challenges and having substantial code to test and fix. We do believe from experience we have from writing code in lab and doing homework that writing the initial code is the easy part, but making work how we want it to work and be able to mesh together with other aspects of the project and its code is the difficult part and will require strenuous testing and debugging. Also we plan on playing testing the game a lot and trying to break it in the sense that we want it to be as perfect as it can be. And sometimes it’s easier to find problems after implementing code and testing it together than just scrolling through code and trying to fix issues. We want the game to feel exactly as it did from people's childhood and bring a sense of nostalgia and familiarity, which will require a lot of testing until it's perfect.

III. Experimental Setup

Testing and setting up our project was quite a bit different then in previous semesters. Given the new COVID restrictions in place till the end of the semester, we were unable to meet together or in the labs to work on our project. This presented us with some difficulties as only one of our group members had purchased the Nexys A7-50T and all other members have Apple computers, leaving them unable to use Vivado.

Due to these circumstances, our setup for testing and verifying our project was to have video calls on google meets. During the online meetings we programed and tested our game using the Vivado 2018.3 software. The code written in the Vivado software was then implemented on the Nexys A7-50T FPGA board. We didn’t end up using any external components (LCD or a piezo buzzer) but rather used only the five push buttons on the A7-50T and the seven segment display. Only one of us was able to use the software and hardware but by using screen sharing in the video calls we were able to see what was going on and debug any issues as a group.

IV. Results

After testing the finished product several times, we have reached adequate results that we are satisfied with. The implantation of the code works perfectly as we intended with the game working in a satisfactory manner by outputting the expected results that we intended to output. The multiple phases work as intended, for example the pass state works as you would expect, telling the player that they passed for a few seconds and seamlessly transitioning to the next phase where the player is expected to remember what order the LEDs flashed and inputting the buttons. The only issue we had with some of the results is a debouncing issue where when a button is pressed it inputs more than one input resulting in a loss state but that seems to be fixed now and that issue seems to be fixed after taking another look at the debouncer. All in all we are very satisfied with the results and the game works as intended

Conclusions

After working tirelessly to achieve the intended results and a satisfactory project that all group members are proud of, we can all say that we now have a better understanding of VHDL and the work process that would need to be undergone for a bigger project that we undergone in the real world. We understand that communication is key and dividing the workload between a group to maximize work efficiency between members in accordance with their best suited skills. We also understand that working with VHDL isnt just code, its working with hardware and understanding its limits and its strengths to achieve results that are satisfactory. We know we have only touched the tip of the iceberg and you can use these skills to achieve much more. The game we created is something all group members are proud of and we all came out of it with a tighter grasp of the skills that engineers need for success. We believe we have created an awesome project and worked proficiently to achieve the results we intended at the beginning of the project and now know how to get there. From creating a blueprint using data paths and state machines, to researching how this project works in other systems and implementing it in VHDL. To get a better understanding of systems we had to teach ourselves such as creating PRNGs and implementing systems we learned in class such as registers and clocks. With the additional challenges of COVID, it was a strenuous process but all members believe we have created a project that we are all proud of.

References

1. Chami, Claudio Avi. “Pseudo Random Generator Tutorial in VHDL (Part 1/3).” *Semiwiki*, Semiwiki, 4 Sept. 2016, semiwiki.com/fpga/6129-pseudo-random-generator-tutorial-in-vhdl-part-1-3/.
2. Collinger, Zachary. “How to Play Simon Says Game.” *Considerable*, Considerable, 30 Oct. 2019, www.considerable.com/entertainment/games/simon-says/.
3. Colvin, James. “Simon Says... with LEDs!” *Digilent Blog*, 4 Aug. 2015, blog.digilentinc.com/simon-says-with-leds/.
4. Gandiaga, Xabier. “EmbDev.net.” *Need Help with Simon(Game) VHDL Code - EmbDev.net*, 3 Dec. 2016, embdev.net/topic/413682.
5. geeks, geeks for. “Pseudo Random Number Generator (PRNG).” *GeeksforGeeks*, 6 Sept. 2019, www.geeksforgeeks.org/pseudo-random-number-generator-prng/.
6. harmon, Dave. “Daveharmon/SimonGameVHDL.” *GitHub*, 16 Aug. 2015, github.com/daveharmon/SimonGameVHDL.
7. “Introduction to FPGA Design with Vivado High-Level Synthesis.” *Xilinx*, Initial Xilinx Release., 22 Dec. 2019, www.xilinx.com/support/documentation/sw_manuals/ug998-vivado-intro-fpga-design-hls.pdf.
8. Irfan, Muhammad, et al. “Pseudorandom Number Generator (PRNG) Design Using Hyper-Chaotic Modified Robust Logistic Map (HC-MRLM).” *MDPI*, Multidisciplinary Digital Publishing Institute, 6 Jan. 2020, www.mdpi.com/2079-9292/9/1/104/htm.
9. JamesdelukJamesdeluk 11588 bronze badges, et al. “VHDL: Button Debouncing (or Not, as the Case May Be).” *Stack Overflow*, 1 June 1969, stackoverflow.com/questions/61630181/vhdl-button-debouncing-or-not-as-the-case-may-be.
10. Kallaher, Brandon. “Getting Started with Vivado.” *Getting Started with Vivado [Digilent Documentation]*, reference.digilentinc.com/vivado/getting_started/start.

11. Larson, Scott. “Debounce Logic Circuit (with VHDL Example).” *Debounce Logic Circuit (with VHDL Example) - Logic - Eewiki*, 28 June 2019, www.digikey.com/eewiki/pages/viewpage.action?pageId=4980758.
12. MISHRA, SHATADAL. “FPGA BASED RANDOM NUMBER GENERATION.” *Department of Electrical Engineering National Institute of Technology*, Department of Electrical Engineering National Institute of Technology Rourkela, 2010, ethesis.nitrkl.ac.in/1960/1/final_prj.pdf.
13. Pasini , Antonio. “Fig 1- Uploaded by Antonio Pasini Content May Be Subject to Copyright.” *Www.researchgate.net*, ResearchGate, 10 Apr. 2007, www.researchgate.net/figure/General-architecture-of-a-digital-PRNG_fig1_235790455.
14. *Pushbutton DeBounce Circuit in VHDL*, vhdlguru.blogspot.com/2017/09/pushbutton-debounce-circuit-in-vhdl.html.
15. Rtc. “How To Make Simon Says Game With Your FPGA.” *Gadget Factory*, 18 Apr. 2016, blog.gadgetfactory.net/2016/04/how-to-make-simon-says-game-with-your-fpga/.
16. Startups, Augmented, director. *Coding and Simulating Simple VHDL in Vivado*, YouTube, 21 Apr. 2016, www.youtube.com/watch?v=ShjXQdKdxsE.
17. Startups, Augmented, director. *Introduction to the Vivado Design Suite Interface and Creating a New Project*, YouTube, 19 Apr. 2016, www.youtube.com/watch?v=BVNHbJ2WZRU.
18. tsuker, Dennis, and Instructables Instructables. “'Simon' CPE 133 Final Project.” *Instructables*, Instructables, 12 Oct. 2017, www.instructables.com/Simon-CPE-133-Final-Project/.
19. “VHDL Code for Debouncing Buttons on FPGA.” *FPGA Projects, Verilog Projects, VHDL Projects - FPGA4student.Com*, 2016, www.fpga4student.com/2017/08/vhdl-code-for-debouncing-buttons-on-fpga.html.
20. Wikipedia. “Pseudorandom Graph.” *Wikipedia*, Wikimedia Foundation, 16 Oct. 2020, en.wikipedia.org/wiki/Pseudorandom_graph.
21. Wishman, Max . *Simon Game with VGA* , Microprocessor-Based Systems , 11 Dec. 2009, pages.hmc.edu/harris/class/e155/projects09/GlickWishman.pdf.
22.
23.