Pong

FPGA Video Game

List of Authors (Ardian Djekovic, Randy Tomka, Divon Yono)
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
e-mails: adjekovic@oakland.edu, tomka@oakland.edu, divonyono@oakland.edu

***Abstract*— Our final project was to create a popular, interactive video game using a Nexys-DDR board with VHDL programming. We found that we learned most of this programming in our previous labs, it was up to us to use our previous labs and have them work simultaneously. Implementing a display into the project had seemed easy but took a long time to program. Together with teamwork, we were able to create a functional interactive video game: Pong.**

I. Introduction

Pong is a renowned arcade game dating back decades ago. The scope of the project is to design this arcade game and implementing it onto a FPGA board. Pong will be a two-player game. The ping pong ball will start in the middle of the screen with two pongs (paddles) on both ends. Each user will be controlling its paddle by using the buttons on the board. The user will turn on a switch to initiate the start of the game. The ball will then move in a random direction and users will move their paddle to hit the ball. The ball will then bounce off the paddle and go in the opposite direction. When the ball is hit beyond the paddle (this acts such as goal posts), the opposing player will get a point and win the game and the other player will lose the game. The FPGA board will be connected to a monitor through VGA.

The motivation of this project was to create an entertaining game with a FPGA board as well as having a learning experience from it. We will be implementing what we learned from our previous labs as well as in class lectures such as the external peripherals: interfacing. We also conducted further research into the VGA component than our class covered. Our projects application is being used in the gaming industry, as well as any external output to monitors.

II. Methodology

A. Layout

The layout of our project's controls can be seen in Figure 1. The monitor was connected through VGA which was plugged into the VGA port in the Nexys 4 board. Then the CPU reset button was used to clear the board and restart the game. The 7-segment display was utilized to display the score of what level the user got to. The up and right button was utilized to control the movements of the 1st player and the left and down buttons were utilized for the 2nd player. The center button initiated the start of the game.

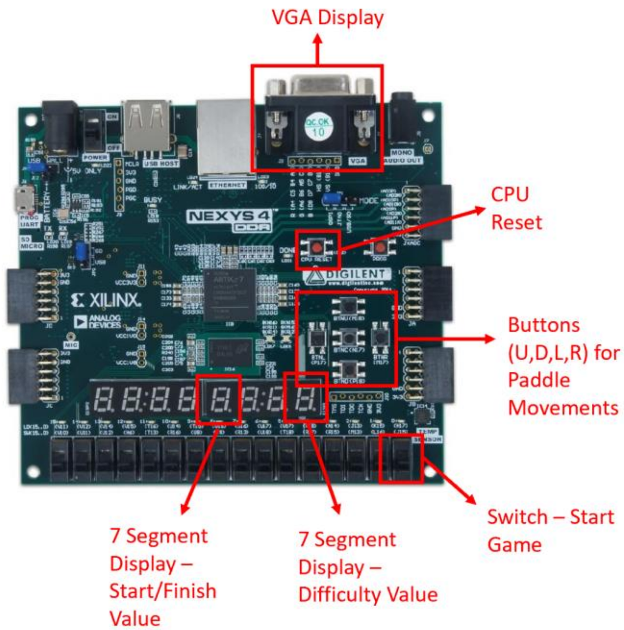


Figure 1: Board Layout

B. Inputs

The design for the input methods was a long procedure that involved a board input. We are utilizing the Nexys board as our method of input with 4 buttons, U, D, L, R, (Up, Down, Left, Right) to control each user's paddles movement. Buttons U and R were used for player 1 movement to go up and down. Buttons L and D were used for player 2 movement to go up and down. Switch SW0 was used to initiate the start of the game – this is when the ball starts moving. Also, the CPU reset button was programmed to clear the game and restart it.

C. Top Level

The top-level design of the project can be found below figure 2 shows the block that is in charge of generating and controlling all the object on the screen. And in figure 3 the expand top level design is shown with the component that are used and they are all inside the top file that is shown in figure 2 of the report.

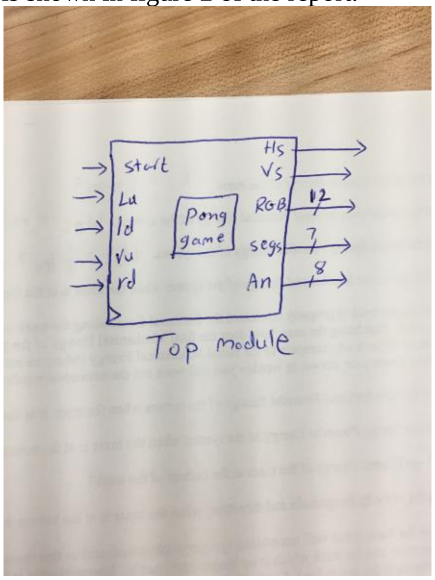


Figure 2: Top file

D. Datapath Circuit

The main part of the project dealt with creating and moving the objects on the VGA screen. The VGA display and controller were taken from Dr. LIamocca. The objects were generated by assigning constant values to the sides of each project. For example, both paddles were given a length of 10 pixels, also the distances from the sides of the screen were initially assigned for each object. Every distance was measured in relation to the top and left sides of each object. If the bottom and right sides of each object were required, the height and width were added, respectively. These distances were chosen in reference to the top and left sides of the screen, since this is the aspect in which the VGA display code writes to the screen. Some of these values, like the positions of the walls and the horizontal positions of the paddles, were static and were assigned as constants. The vertical positions of the paddles and both the horizontal and vertical positions of the ball were constantly changing, so these were assigned as signals. The object was draw to the screen by VGA controller. It was done by having a statement that checked the current position of the object with vertical and horizontal count of the VGA controller. When the count was in between coordinates that were assigned to an object, an "on" signal went high. There was one of these signals for every object, and they were all concatenated together with the "video_on" signal from the VGA display. This new signal was then used as a select line for a multiplexer. Each object also had its own RGB color code assigned to it, and these codes were the inputs to the multiplexer. Whenever the count reached the area on the screen where an object was supposed to be drawn, the RGB color of the object was drawn on the screen. The purpose of this multiplexer is to toggle through the different objects, depending on what "on" signal is currently high. It toggles fast enough that it looks like every object is displayed on the screen at the same time. The paddles and the balls needed to be able to move around the screen. To do this, a constant was assigned for each object, and this was denoted as the velocity of the object. This velocity was added to the position of the object so at the next tick, the object would move as many pixels as the velocity was defined as. For example, the paddles had a velocity of 10, so they moved 10 pixels every clock tick. This allowed the paddle to move from one end of the screen to the other in about a second, which was reasonable for the purpose of the game. This allowed the paddle to move from one end of the screen to the other in about a second, which was reasonable for the purpose of the game. The animation of the ball was much more challenging than the paddle, since the ball had to move both vertically and horizontally, and it was not controlled by an input like the paddles were. The vertical and horizontal speeds were equal. This occurred if the ball was in the bounds of the screen. The ball also needed to have its direction change whenever it came touched another object. This hit detection was the most challenging area of the project. To accomplish this, the coordinates of the ball were checked with the coordinates of the other objects through comparison operators. The ball also had to have its horizontal coordinates match up with the position of the side of the paddle. This would then reverse the horizontal speed of the ball, sending it in the opposite direction. However, if the ball missed the paddle then the round would end, the winner would be displayed on the 7-segment display, and a new round would begin once the reset button is pressed.

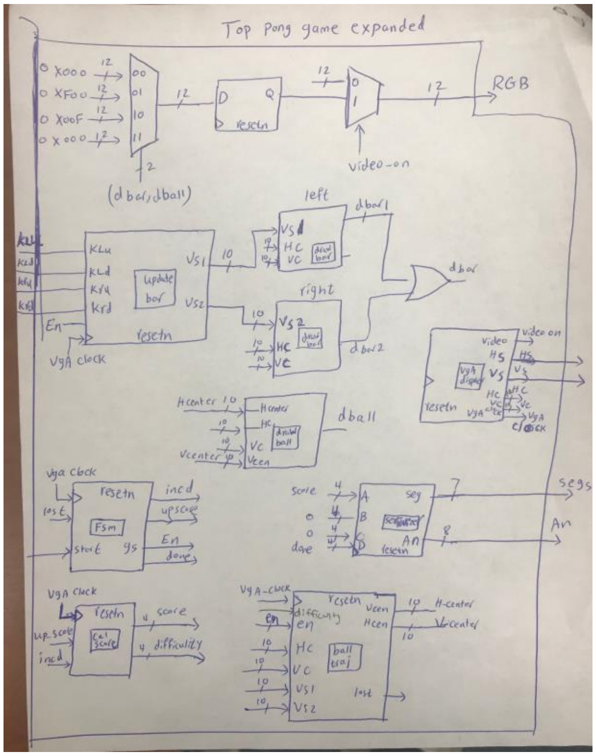


Figure 3: Top file design expanded with data path

E. Control Circuit

State1: is the initaliazation state it's the beginning of the game starting positions.

State2: is the play state where gs is set to 1 and done is set t0 0101 and if the player doesn't lose the game it increments the difficulty of the game.

State3: the up score is set to 1 and the difficulty increases if the player doesn't lose the game.

State4: if player loses the game then gs s set to 0 and done is 1111 which is on. Then the game starts by pressing the reset button on the board.

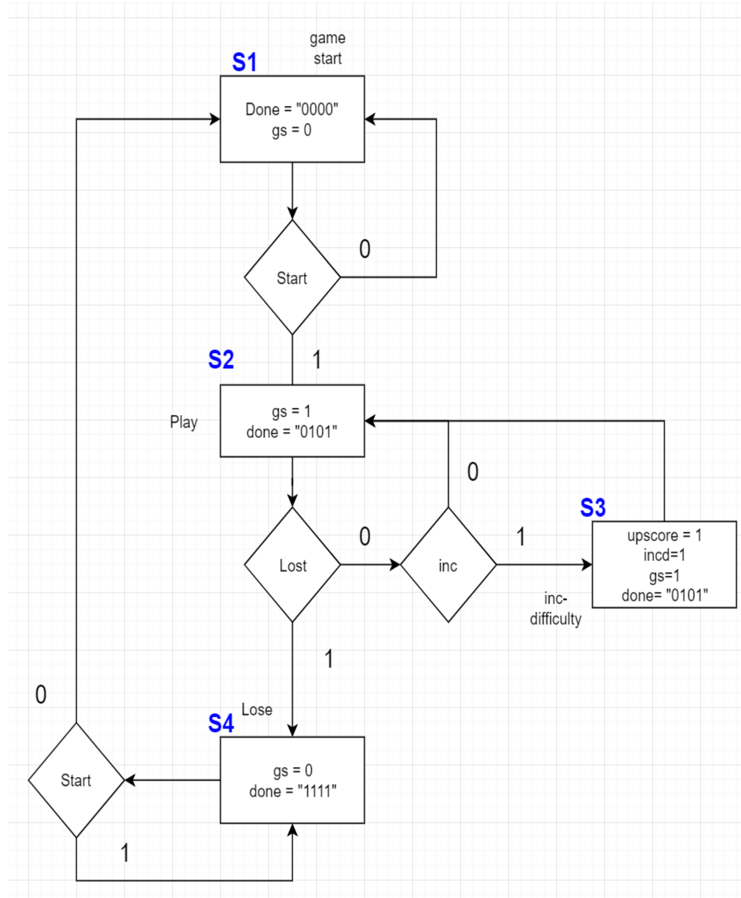


Figure 4: states machine of the pong game

F. Output

One of the last steps of the project was to display the output of the game through VGA to monitor. The results were outputted on the 7-segment display in hexadecimal to display the result of the users score. The component displays an unsigned hexadecimal value of the result signal into the 7-Segment display.

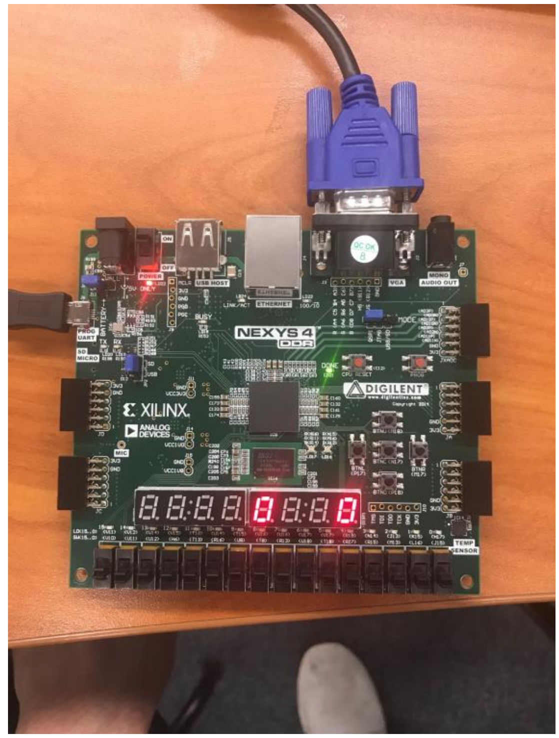
III. EXPERIMENTAL SETUP

The setup that was used was a Nexys4-DDR board along with VHDL coding. The main software tool used to do the coding was all written inside Vivado program. Each component/module was designed and simulated separately from one another to ensure success between each component. It was possible to see if any mistakes were made in the code from the timing and behavioral diagrams. After completing any troubleshooting needed, we implemented the code in the overall design and then moved on to the next component. This allowed the project to be completed smoothly in increments and in a timely manner. As a team, it became more effective to work out problems and bugs with each component one at a time. After simulating all the components and verified everything worked, it was time to implement the monitor in the design. Implementing the VGA components into the project was not as easy as planned, but after some troubleshooting, the team was able to get the job done. The projects hardware tools that were used was the Nexys board itself along with the monitor. To test our cases, the buttons on the Nexys were first used for the paddle movement. This verified the code and components were correct and slowly paved the way to implement the VGA component.

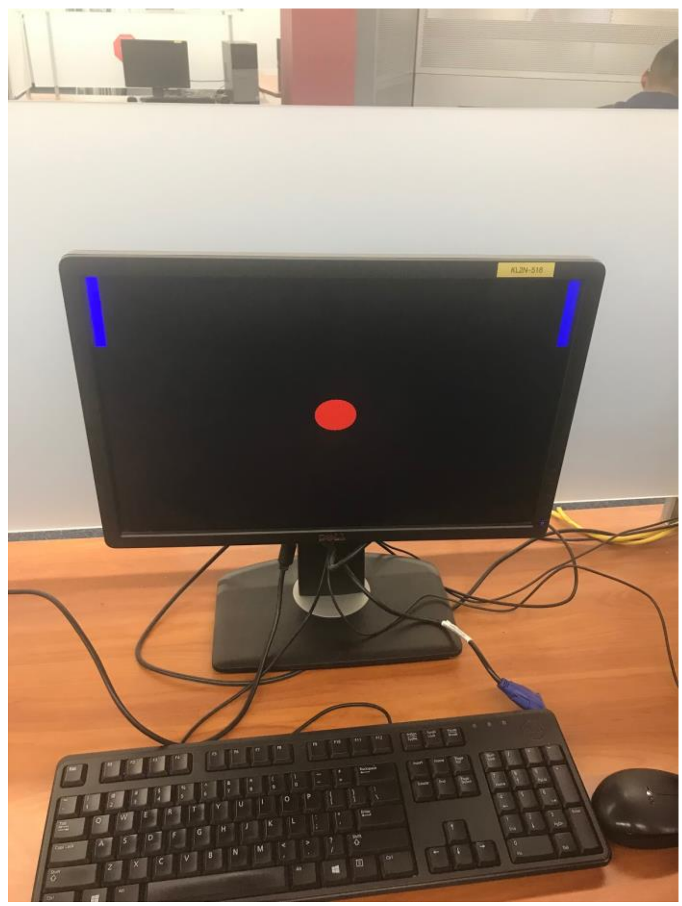
The expected results were to have a functional pong video game that operates by buttons, performs visually onto the monitor through VGA, and displays the result on the 7-segment display of the Nexys board. It was decided early on to input the movement of the paddle from the buttons and display the result on the 7-segment display.

IV. RESULTS

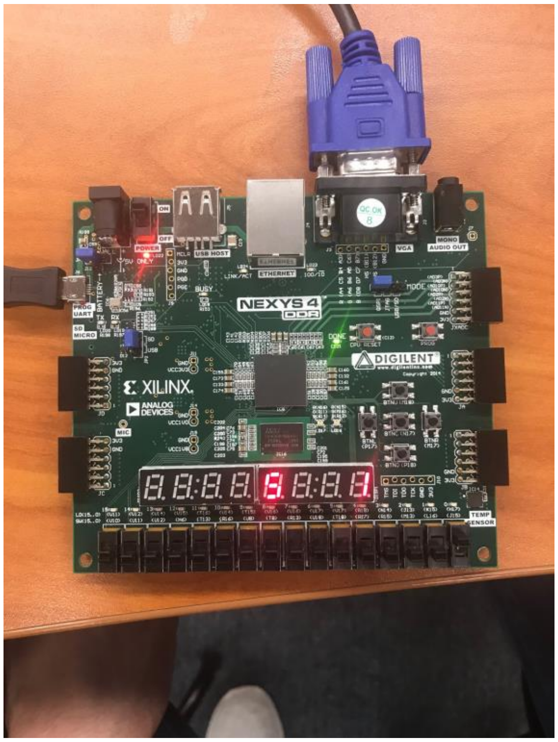
Overall, we were happy with the results. Everything was performing as functional as we expected. The ball and bars moved accordingly and very smooth. The response of the hardware buttons and switches were timed correctly as well. We were also very happy with the 7-segment display. It displayed the level we were currently playing, as well as telling us the state of the game (start or finish).



Picture1: Starting position of the game

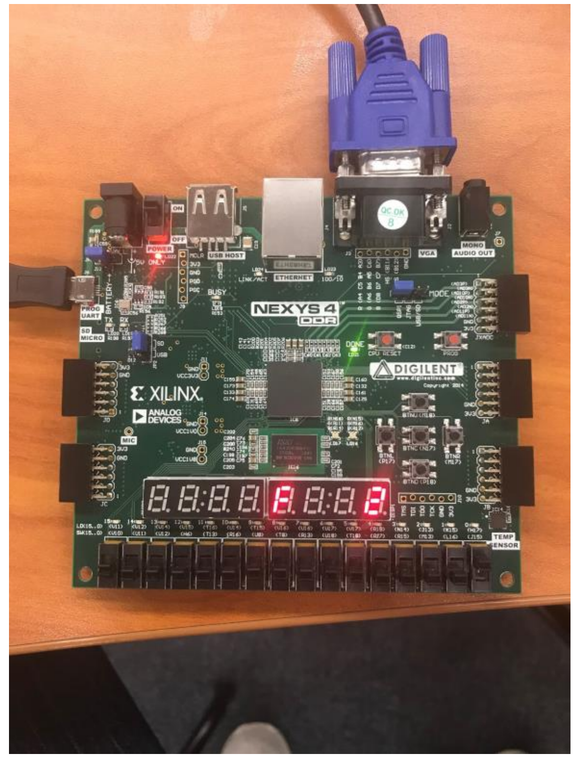


Picture2: Display the game through VGA



Picture3: first level of the game

Picture4: playing the game using nexys 4



Picture5: Losing states of the game

CONCLUSIONS

The constant challenges of the project were a great learning experience along with including past labs and researching additional material that was not covered in class. Integrating multiple components such as adders, state machines, and registers together in one big design became a challenge which was overcome. Areas not covered in class such as the implementation of the monitor to the Nexys board was further research done by the group to meet the needs of the project. One of the biggest improvements we would like to make is to add two controllers that way our fingers are not jammed together pressing buttons closely to each other. Another improvement we would like to make is to add a background picture with nice scenery at each level. This will make the game much more enjoyable and less dull. The most difficult part of the whole project was trying to make it a single player pong game with one of us versing the "computer". We could not even know where to begin to start that. We switched the project around and made it a two-player game instead. Although these improvements were not made in this version of pong, we still feel as if our final project laid out a well-represented version of the game.

REFERENCES

List and number all bibliographical references in 9-point Times, single-spaced, at the end of your paper. When referenced in the text, enclose the citation number in square brackets, for example [1]. Where appropriate, include the name(s) of editors of referenced books. The template will number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use "Ref. [3]" or "reference [3]" except at the beginning of a sentence: "Reference [3] was the first . . ."

[1] Llamocca, Daniel. "VHDL Coding for FPGAs." *VHDL Coding for FPGAs*. Web 14 Apr. 2018.