VGA Digital Pong Table Tennis Game

List of Authors (Sara Jamu, Everardo Mejia, Armela Gjokaj, and Brandon Brennan)
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
e-mails: sjamu@oakland.edu, emejia@oakland.edu, armelagjokaj@oakland.edu, bbrennan@oakland.edu

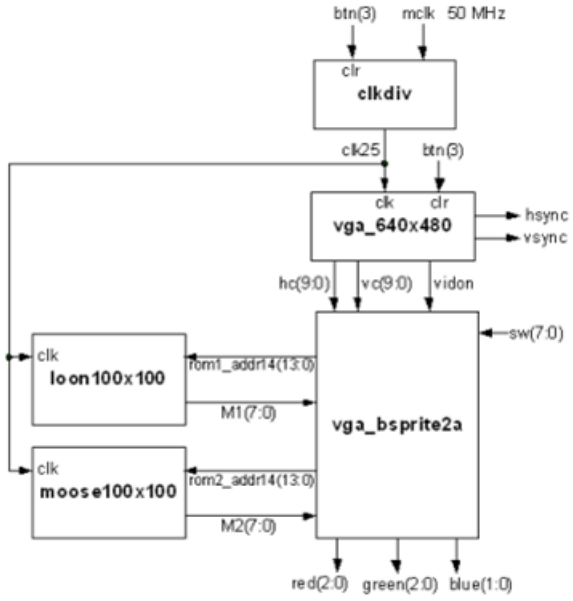
***Abstract* - The project consists of the design and simulation of a classic Pong game, where either two people play against each other, or one plays against a computer simulation. This is a 2D game, which is played by passing the ball until one of the players scores. Once this happens, a new ball will be generated, and the game will restart. The goal of this project is to develop a Pong game by integrating software into hardware and understanding different methods of communication with external peripherals.**

Introduction- The objective of this project is to design a Pong game displayed using a PC monitor. This design will be accomplished utilizing the VHDL hardware description language on the Nexys 4 DDR FPGA board. The purpose of the game is to pass the ball from one player to the other until one of them scores. The user will be able to use SW0 when he wants to pause the game, or the CPU reset to restart the match. The primary external device to be utilized will be a keyboard which is used by each user to move the paddle. Also, we used the buttons of the board to control the paddle of the user. A VGA controller will create communication between the keyboard and the FPGA board, and the game will be displayed either through a VGA monitor. Based on topics we learned in class; we will be able to combine the use of an FPGA board with a keyboard and display screen to output a fun game.

I. Methodology

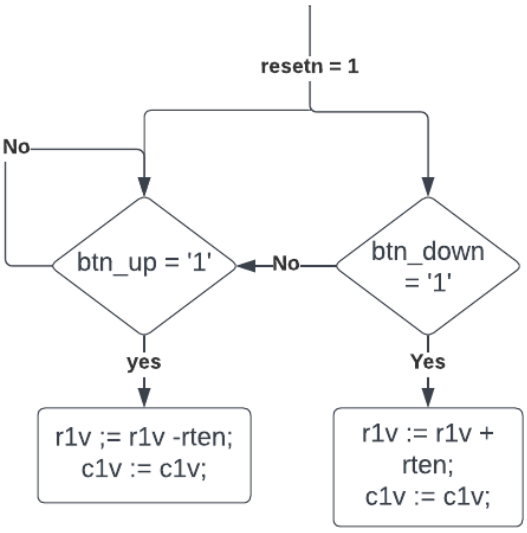
Design Methodology

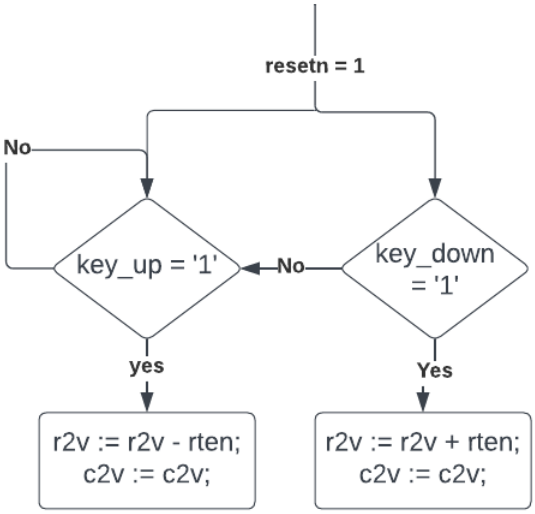
The goal of this project is to develop a game of pong in VHDL using Vivado software and the Nexys4 DDR board. In comparison to current versions of Pong, this version of pong would only feature a PVP mode using a keyboard to receive the inputs for Player 1 and Player 2. Respectively, Player 1 will control the placement of their paddle using the w and s keys and Player 2 using the up and down buttons on the FPGA board. To develop this code, the group referenced Professor Hanna's previous work with VGAs and Sprite development regarding FPGA boards [5]. Mainly, the team referenced multiple different texts written by Darrin E. Hanna and Richard E. Haskel when first creating this project [3]. The team also referenced Professor Hanna's academic work from previous semesters to get a better understanding of how this project could be completed [4]. A good example of the type of sprite/VGA development referenced for this project is shown below.



To begin with, the sprites for Players 1,2, the dotted center line, and the Ball are all created simply using Microsoft Paint and turned into pixelated images. Here, the pixel sizing was defined in MS paint when creating the image (player 1&2: 100x40, ball: 30x30, center line: 500x10), before turning it into a sprite. Then, these paint images are converted into machine code by running MATLAB scripts that can describe the colors of these images at each pixel. Within the code all images had a 12-bit output (= 3 hex values) in the .coe (coefficient files) to describe a pixel's color code so that it can then be used within Vivado to display the sprites for Player 1 and 2's paddles, the dotted line to separate the players, and the ball itself onto the VGA monitor connected to the Nexys4 DDR FPGA board.

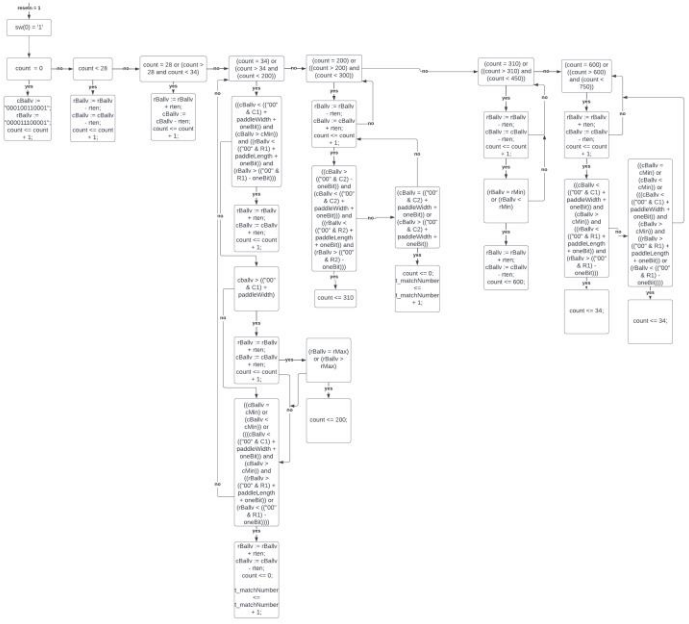
Overall, the Sprites are controlled by two separate Finite State Machines. One FSM is used to control player one and player two's paddles, shown below.





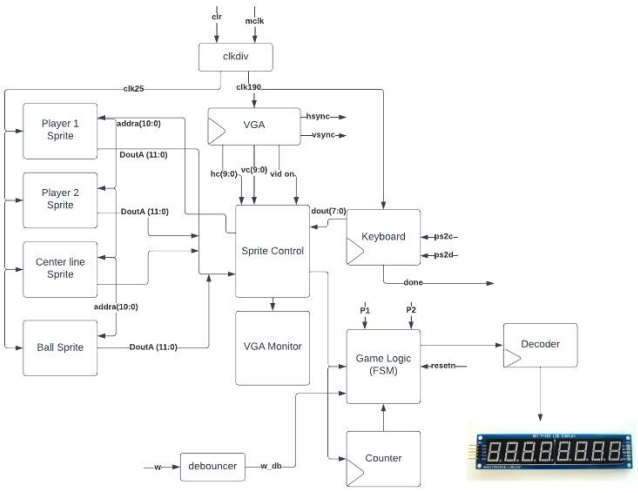
With this FSM, player one can move their paddle up with the left button (BTNL) and down with the down button (BTND). Player two is then able to move up with the up button (BTNU) or the up key (w) on the keyboard and down with the right btn (BTNR) or the down key (s) on the keyboard. Each of these paddles are connected to the buttons located on the FPGA board. The paddles can also be controlled using an attachable keyboard Peripheral that is plugged into the board as well.

The second FSM is what controls the algorithm of the ball movement and bouncing of the ball, also shown below.

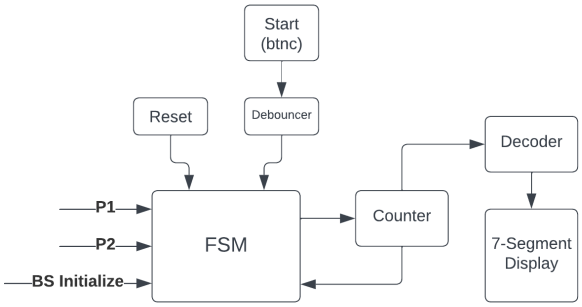


The beginning of the FSM is controlled by the start switch of the game. If the switch is flipped on, then the game will start and therefore start running through the game logic. Otherwise, if the switch is flipped again the game will be paused and the ball will immediately stop moving. The beginning of the game gives the ball the exact coordinates of where to be on the screen (x = 305, y = 225) and then increments the count. Then, the ball movement is initialized to move to the left side of the screen and move a certain number of pixels up and to the left and increment the count. The FSM also has the ball move down and to the left depending on the value of the count variable. The FSM above will also control the AI/algorithm of the ball. The FSM tells the ball to restart the ball's position on the screen and the count variable if the ball goes out of bounds on the screen. If the ball collides with the surface of either paddle, then it will bounce off the paddle to the other side of the screen. Once one of the players loses a match, then the number of matches played will be increased and the ball will restart its place in the middle of the screen.

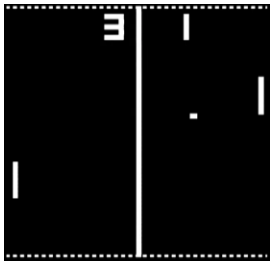
Shown below is a block diagram of the overall design of the VGA Pong board. Shown within the diagram is a counter that receives information from the game logic and displays the number of games played by the users through a decoder on the FPGA boards seven segment display.



The game logic behind pong is ran mainly using an FSM that will update the state of the game and update the win counter for each player, the counter will also display the score for each player using the 7-segment displays available on the Nexys board (in the format "P#: (score)). The FSM will accept inputs from the reset and start (BTNC) buttons to begin the game and set the location of the ball dependent on the state of the game. The FSM is also receiving input from the score counter to determine two things. One, which side of the screen the ball will start and two, when the game ends, which will be programmed to change the state of the game to "Game Over" when one of the players reaches a score of 9. The game logic is structured as shown below.



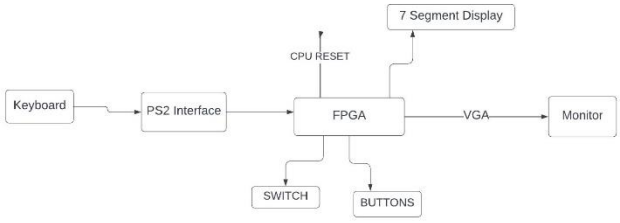
The final product is similar to the image below.



II. Experimental Setup – Hardware Setup

For the setup of this project, we used the following equipment:

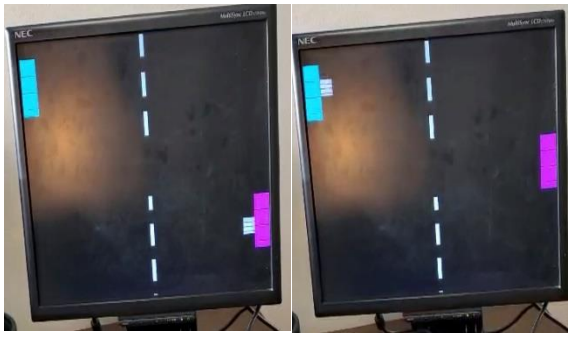
- Vivado 2021.1 Software
- Nexys4 DDR Board
  - CPU reset button.
  - 1 switch
  - Buttons
  - 7 Segment Display
  - 2 LEDs for w key press vs s key press
- Keyboard
- VGA Display monitor



The diagram above is a representation of the integration of the hardware setup for the project. For the implementation of the game, we used a VHDL code that implements a keyboard that is used to control the game. The player controls the game from 2 keys, W, S which control the paddle of one of the users. CPU reset will be used to restart the game. Also, we used a switch from our FPGA board, that was used to pause or start the game. The other opponent that we used was the buttons of the FPGA board to control the other paddle of the game. The losing occurs when a player can't catch the ball and the ball passes and touches the respective edge of the screen, so the other player gains a point. The score of the player and his opponent will be shown on the 7-segment display. The game was displayed on the VGA monitor that gets all the information directly from the FPGA board.

III. Results

The result of this project is a working multiplayer game of pong using the Nexys4 DDR FPGA Board. The user can play the game using the keyboard and the buttons and watch it on the VGA Monitor. We were able to get a working keyboard that would change the lights of the LED on the FPGA board based on the key that we pressed which was either W or S. However, we were not able to implement it with the rest of the project, so the user only used the buttons of the FPGA board.



The figures above are a representation of the VGA display that we were able to get when running the project. The ball bounces at a 45-degree angle and it bounces from one paddle to the other. The seven-segment display shows the points the player gets, and it is incremented each time one of the players scores.

Improvements: Even though we had a working project, there are a few things that we could improve but weren't able to because of time constraint. One of them is to add a trained AI ball. Also, we could add levels once a player reaches a certain score. For example, after one player reaches 9 points, as the maximum points we had for one player to get, it will change the level of the game and the background after each level changes. We could also add limits to the paddles, so they won't go beyond the screen display.

CONCLUSIONS

There are several challenges that are to be expected when creating this FPGA project. Our group had a few days to work on all the project files and integrate them together. When it comes to FPGA projects (or any project for that matter), timing constraints must be met for this project to work correctly. This means that each of us within this group must be able to allocate our time properly to fully complete this project and fully encapsulate the scope of what we wish to achieve with this project. Each person in our group works outside of school, with two students also involved in senior design so it was important to manage our time constraints properly.

Another challenge that faced during the project was working with sprites and VGA as it wasn't a topic that we learned in class. FPGA's have limited resources regarding logic cells, routing resources, and memory blocks. So, it was challenging for our group to create a complex design within the available resources. Our group did a lot of research on the internet, in school resources, and from other students who are skilled in VHDL regarding implementing this project, mainly the sprite work. Some of the sprite work that will be used in the project is not currently covered in the materials from our class lectures so our group must ask for assistance and do our research to figure out how to fully implement this sprite work into our project.

The main challenge was working with the sprites, moving, and controlling their addresses in correspondence to address of the AI ball. Because the ball is not an AI trained module, it doesn't bounce at all angles. Also, a challenge was implementing a multiplayer keyboard which was then replaced by buttons from the FPGA board and controlled by the user.

Overall, the project was very fun to make. Even though it had its challenges and being time consuming, we were able to combine what we learned during this semester with our research and what we learned about sprites to create a fun game.

REFERENCES

[1] Daniel Llamocca / Oakland University, VHDL coding for fpgas. [Online]. Available: https://www.secs.oakland.edu/~llamocca/VHDLforFPGAs.html [Accessed: 20-Mar-2023]

[2] Xilinx, "Vivado ML Overview," Xilinx. [Online]. Available: https://www.xilinx.com/products/design-tools/vivado.html. [Accessed: 20-Mar-2023]

[3] XHaskell, Richard E., and Darrin M. Hanna. Learning by Example Using VHDL: Advanced Digital Design with a Nexys 2 FPGA Board. LBE Books, 2008. [Accessed: 22-Apr-2023]

[4] Haskell, Richard E., and Darrin M. Hanna. Digital Design Using Digilent FPGA Boards: VHDL/Vivado Edition. LBE Books, 2018. [Accessed: 22-Apr-2023]

[5] HASKELL, RICHARD EDMUND. Advanced Digital Design Using DIGILENT FPGA Boards: VHDL/VGA Graphics Examples. LBE, 2016. [Accessed: 22-Apr-2023]