Tic-Tac-Toe Using VGA Output

Alexander Ivanovic, Shane Mahaffy, Johnathan Hannosh, Luca Wagner

Electrical and Computer Engineering Department

School of Engineering and Computer Science

Oakland University, Rochester, MI

aivanovic@oakland.edu, spmahaffy@oakland.edu , jonathanhannosh@oakland.edu, lwagner2@oakland.edu

***Abstract*- A game of Tic-Tac-Toe was created on a Nexys 4 ddr board using VHDL, and played on a VGA screen. In a traditional game of tic-tac-toe, the users position is indicated by an X or an O, but our game will use colors red and blue. The players position was decided using switches 3 down to 0 on the board, and was saved in registers. A component called a VGA controller was responsible for displaying the users color.**

I. Introduction

Tic-Tac-Toe is a fun and simple two player game where players try to match 3 objects in a row vertically, horizontally, or diagonally. Our game has a grid where players can choose their position by using the binary representation of their desired grid position using switches 3 down to 0. The top left box is 0, the top middle box is 1, the top right box is 2, the middle left box is 3, the middle middle box is 4, the middle right box is 5, the bottom left box is 6, the bottom middle box is 7, and the bottom right box is 8. In order for the player to choose which color is displayed they must use switch 15. Turning switch 15 on makes the color blue, and turning it off makes the color red. Pressing the center button will confirm the users position and color, and will allow it to appear on the VGA screen. For example if the user selects switches 3 down to 0 to be "0010" and switch 15 to be "1", then hits the center button, the color blue would appear in top right box. When the game is over, the user can press the reset button to clear the colors and play a new game. The main motivation for doing this project was to gain an understanding on how to create a VGA output in VHDL since it wasn't something that was done in our previous laboratory assignments. Other than VGA output, the rest of the concepts that will be discussed have been practiced in laboratory assignments. The Main components used in the top-level design are the decoder, registers, combinational circuit, the multiplexor, and the VGA controller.

II. Methodology

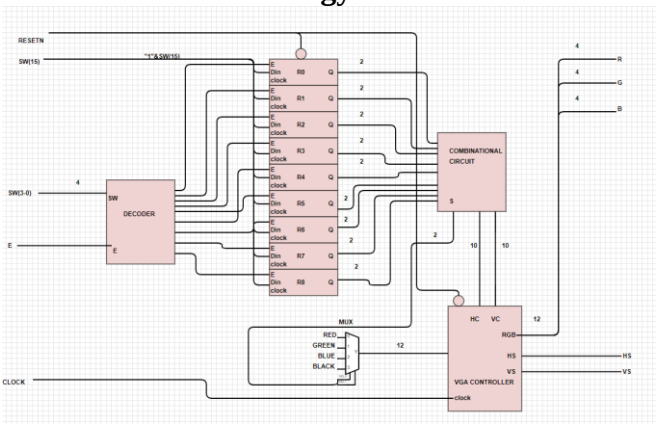

Figure 1: The Top level design of the circuit created in this project.

A. The Decoder

The decoder is the fist component that the user input goes through and is responsible for storing the color into the correct register. The way the decoder works is it takes the users input from switches 3 down to 0, and the center button and uses

them to enable the correct register. For example if switches 3 down to 0 are “0000” and the center button has been pressed then the color selected by switch 15 will be stored into register 0. If the user selects switches 3 down to 0 to be “0001” and the center button is pressed then the color will be stored in register 1. The decoder follows this sequence for all 9 registers.

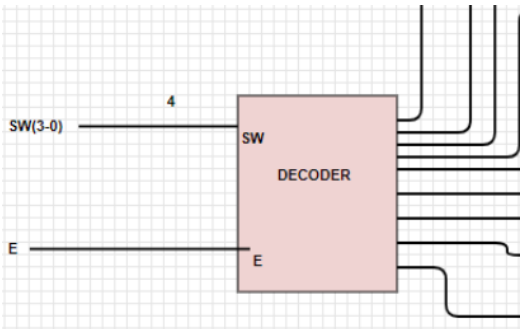

Figure 2: The decoder used in the project.

B. *The Registers*

The registers acted as a memory for switch 15, which represented the color. The registers were enabled by the decoder, and the input and output were the users selection for switch 15. There was also a 1 concatenated with switch 15 as the input and output of the register to validate that there was something in the register and that it was not just in its original state of having a 0 written into it. Without the registers, the game would have no way of remembering what color the user wanted in their square which would defeat the purpose of the game.

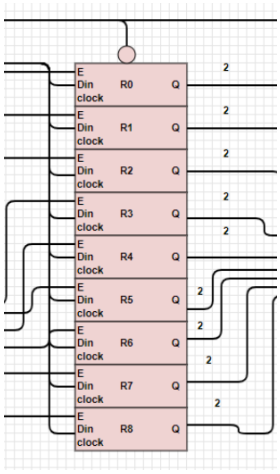

Figure 3: The registers used in the project.

C. *The Combinational Circuit*

The combinational circuit used a series of if statements to determine what color would be displayed and where it would be displayed. The combinational circuit uses the Hcount and Vcount values obtained from the VGA controller and the output of the register, to determine what the select line will be for the multiplexor. It is in the combinational circuit where the positions for each square are declared. The if statements for the grid lines are set up such that the grid lines are always visible. Box 0 would be declared as being between Hcount 0-200 and Vcount 0-150, box 1 is between Hcount 210-400, and Vcount 0-150, box 2 is between Hcount 410-630 and Vcount 0-150, box 3 is between Hcount 0-200 and Vcount 160-300, box 4 is between Hcount 210-400 and Vcount 160-300, box 5 is between Hcount 410-630 and Vcount 160-300, box 6 is between Hcount 0-200 and Vcount 310-470, box 7 is between Hcount 210-400 and Vcount 310-470, and box 8 is between Hcount 410 -630 and Vcount 310-470. The grid lines appear when the Hcount is between 201-209 and 401-409 and when the Vcount is between 151-159 and 301-309.

An example of what the logic would look like for the combination circuit for box 0 would be “If the Hcount is 0-200 and the Vcount is 0-150 then if Register 0 = “11” then the output is blue.” This basically says that if the Hcount and Vcount values are in the box 0 range and register 0 has the color blue written into it that it should send the blue signal to the multiplexor which will be discussed next.

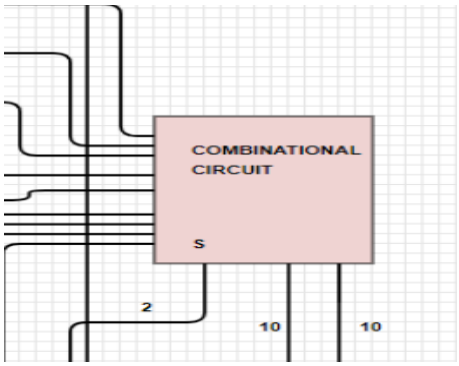

Figure 4: The combinational Circuit used in the project.

D. *The multiplexor*

The Multiplexor used in this project was a 12bit 4 to 1 multiplexor with a select line that was controlled by the output of the combinational circuit. The 4 inputs were the 12 bit binary representation of the colors needed, which were “1111_0000_0000” for Red, “0000_1111_0000” for Green, “0000_0000_1111” for Blue, and “0000_0000_0000” for black. The output of the multiplexor is sent to the VGA controller to display the color chosen. The VGA controller will be discussed next.

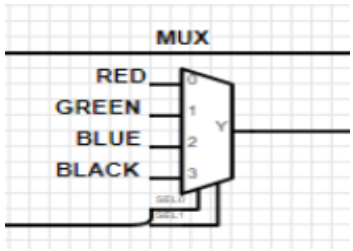

Figure 5: The Multiplexor used in the project.

E. The VGA controller

The VGA controller used in this experiment was The VGA_ctrl_simple which was created by Professor Llamocca. This component worked exactly the way we needed it to for our project so we did not need to create our own. The way The VGA controller works is has a horizontal and vertical sync that keep track of the pixels being written along with the Hcount and Vcount that tell the other components where the horizontal and verticle sync are in terms of writing information on the screen. It also outputs an RGB signal which is the information that is being written into the pixels to show a color. The color that it needs to display is determined by output of the multiplexor.

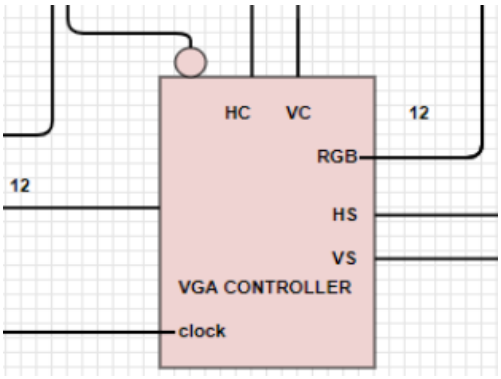

Figure 6: The VGA Controller used in this project.

III. Experimental Setup

The initial testing was done using a testbench and a behavioral simulation in Vivado. This was helpful to see that all of our signals were being used however most of the testing was done by actually plugging in our fpga board into a VGA monitor and going through every possible scenario to ensure that it functions correctly every single time. By doing this we assured that our project would function properly for the live demonstration during the presentation.

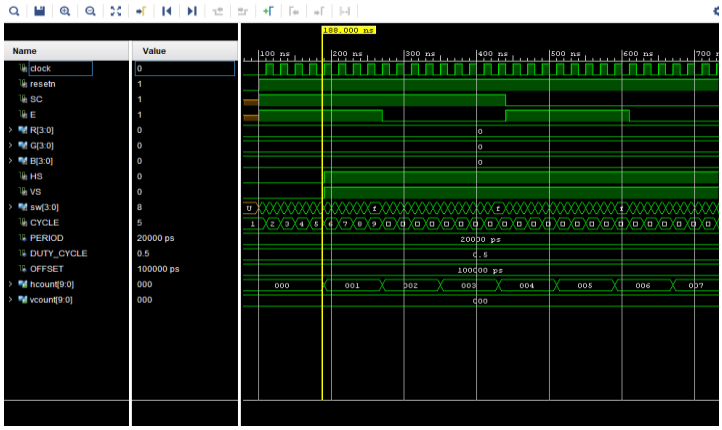

Figure 7: The behavioral simulation for the experimental setup.

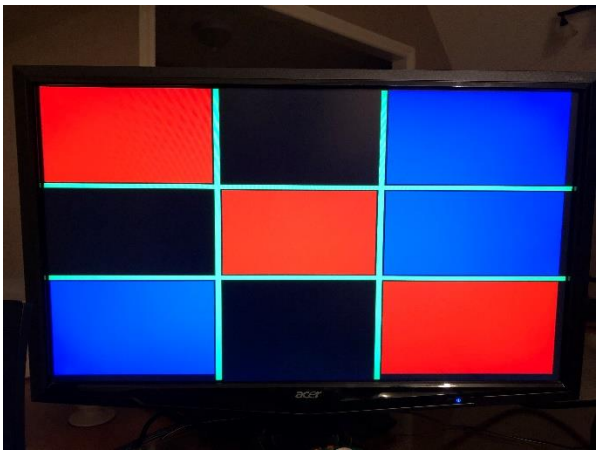

Figure 8: One of the test runs of the project on a VGA screen.

IV. Results

The result of the hard work put into this project is a fully functional game of tic-tac-toe that worked as expected. This game used digital logic, VHDL coding, an fpga board, and a VGA output. Switch number 15 was used to pick the color used, and switches 3 down to 0 were used to pick the position of the color. By using the decoder, registers, combinational circuit, multiplexor, and VGA controller, we were able to play our simple game on a monitor. The results can be viewed on Figure 7 and Figure 8.

V. Conclusions

In conclusion, we were able to use digital logic to create a functioning game of tic-tac-toe. Looking back if we had to re make the game there are some things that we would have done differently. One of the things we would have added on would be a screen that would display who one or say if the game resulted in a tie. Another thing we would have added on would be a scoring component to keep track of the number of wins a color has. Something else we could have added is a component that automatically keeps track of whose turn it is which would eliminate the need for switch 15, and we could have added something that would make sure players could not select squares that already have a color selected. Other than that, we are very pleased with the way our game turned out.

VI. References

[1.] Llamocca, Daniel. "VHDL Coding for FPGAs." VHDL Coding for FPGAs. http://www.secs.oakland.edu/~llamocca/VHDLforFPGAs.html