# Tic-Tac-Toe VGA

Matthew Bayer, James Khoury, Francesco Parrinello, Logan Verstraete
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
mjbayer@oakland.edu, lpkhoury@oakland.edu, fsparrinello@oakland.edu,
ldverstraete@oakland.edu

**Abstract**
For this project the students created a Tic-Tac-Toe game using VHDL, a Nexys board, and a VGA screen. By using the switches, 0 to 8, the user can display their color in one of the 9 spaces of the board. After the game is completed and a winner is determined the screen will display the color of the winner. The users will then press the reset button to start a new game.

## I. Introduction

This report will cover the team's strategies, methodologies, and ideas on the project. The team created a Tic-Tac-Toe game in VHDL code with the VGA screen displaying the game and the two users playing the game on the Artrix-7 FPGA board using nine switches, two buttons and a reset switch. The users will use the switches to select which block to put their color. For example, if BTNL is pressed and switch 2 is pushed up a colored block will appear in block 3 on the VGA screen. The switch is then set back down. The users will alternate their turns and the colors by pressing the two buttons labeled BTNL and BTNR on the board until someone wins (three in a row) or until the board is full. The motivation behind the project is to create a fun and creative game using the concepts learned in class. Some topics that the project covers that the team learned in class is VHDL coding, multiplexers, and VGA screen implementations. VGA screen implementation was vaguely covered in class, the team will need to learn more about them on their own in order to be successful. The application of the project is to provide simply entertainment using concepts learned in class.

## II. Methodology

### A. Concept(Rules)

Tic-Tac-Toe is a simple game where there are two players who alternate marking spots in a tic-tac-toe board. The game board is a 3 by 3 grid with 9 sections. Typically the game is played using "X"s and "O"s but for this application on the VGA screen the board will display two different colored blocks. The game is won when the user has three inputs in a row, column, or diagonal. When there is a winner the screen will display the winner's color. If all the blocks are filled, no one wins it is called a cat's game. The user must press the reset button for a new game.

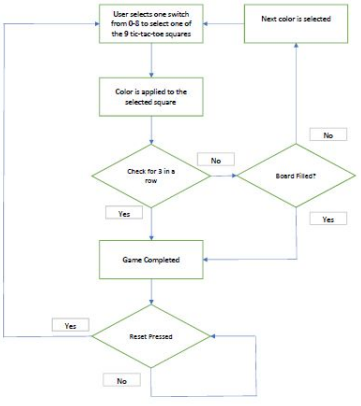


Figure 1: Flow Chart

The flowchart in figure 1 shows the logic behind the game and how it can be implemented in Vivado in the top file. This was done during the first part of the design process. This helped the team understand the logic behind what needed to be done to build a successful game.

## B. FPGA Board Implementation

We will have nine blocks that will indicate each position of the tic-tac-toe position.The nine blocks will correlate to switch 0 to 8 on the FPGA board. Starting with the top left as position one then going through the row and ending with position three. The next row will start with position four and ending with position six. Finally having the last row at position seven and ending with position nine.

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |

Figure 2: Block Numbering Diagram

The game will start by a user pressing the left or right button on the nexys board (BTNL or BTNR). This will have the board to either display the red or green color on the block the user selects. The resetn button will be used as a reset button to restart the game after user wins or all the blocks are filled. In order to reset the game the user also has to move all switches to zero.

## C. Display

The display needed to be sectioned into 9 squares for the game. The VGA supports a resolution of 640x480. Each of the nine blocks had a resolution of 190x140 pixels.

| | | |
|---|---|---|
| HC:10-200<br>VC:10-150 | HC:210-400<br>VC:10-150 | HC:410-600<br>VC:10-150 |
| HC:10-200<br>VC:160-300 | HC:210-400<br>VC:160-300 | HC:410-600<br>VC:160-300 |
| HC:10-200<br>VC:310-450 | HC:210-400<br>VC:310-450 | HC:410-600<br>VC:310-450 |

Figure 3: Pixel mapping of the 9 squares

These specific pixels were chosen by the team to ensure that there was no pixel overlap on the VGA screen. HC/VC stands for horizontal count and vertical count respectively. These indicated which pixels were being written at a specific instance. Also, by leaving slight gaps between the blocks, the tic-tac-toe board was automatically created in black.

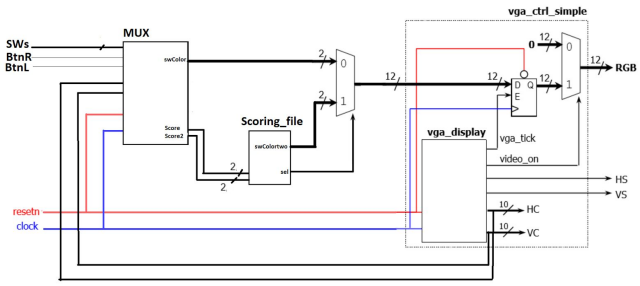


Figure 4: Top level Schematic

Figure 4 above is the team's overall top level design. The MUX was created with switches, buttons, HC/VC, clock and resetn as inputs. The output of the MUX was the two bit color as well as each of the players score count. The score counts were then sent into the scoring_file then color and

select were outputted into the next mux. If select was one the MUX then outputted the winners color into the vga_ctrl_simple to display the winners color. Else, the specific color block was outputted out of the MUX and vga_ctrl_simple wrote the correct player's color onto the VGA screen in the block selected by the switch.

## III. Experimental Setup

The setup that will be used to verify the functionality of the project will be a VGA screen, a computer with Vivado software, and a Artrix-7 FPGA board. The VHDL code will be written in Vivado which will create the game. The code will then be programmed to the FPGA board. The FPGA board will be connected to the VGA screen where the game can be viewed as the user plays it on the FPGA board. The expected outcome of the project will a successful friendly game of tic-tac-toe being played between the two users.

## IV. Results

The team successfully made the tic-tac-toe game using a vga screen and the nexys board. The team had to make changes to original ideas due to complexities that arose. A big issue was displaying the correct color on specific pixels. In earlier versions of the project the pixels would overlap for some blocks and cause a kind of fluctuation in the colors of that block. This slowed the developing process but was fixed. In the end all switches and buttons worked accordingly.

## V. Alternate Approaches / Improvements

The team learned many techniques about how to implement VGA displays using VHDL code and Vivado. The team quickly learned that there were many approaches to producing the same successful game. A counter was originally implemented in order to keep track of the two players colors. The issue was that the counter was continuously counting which lead to the blocks switching colors between red and green rapidly. By using the buttons to alternate colors for the users, it creates more of a video game feeling for them.

Another change that could have been made in the MUX code is the use of decimal HC/VC values. HC/VC were outputted from the VGA display as 10 bit binary numbers. In order to keep it simple, the team wrote the if statements for HC/VC in the MUX in 10 bit binary, which proved to be confusing at times when trying to debug the issues in the pixel counts. For example, if the pixels were overlapping it would create overlapping blocks on the screen, and if the pixels were not written they would pick up the previous color and fill the extra pixels with that color. This was solved by assigning the remaining outside pixels to black. In order to eliminate the error of mistyping a 10 bit binary number, it would be easier to use a signal and convert the 10 bit binary HC/VC to decimal to ensure that there are no overlapping pixels.

One possible improvement to the game would be to implement a way to eliminate the user's ability to overwrite a block already written by another player. The team tried to design this but ran out of time, and it was agreed upon that in the original game of tic-tac-toe one simply does not just draw an 'X' over someones 'O'.

Another possible improvement to our game would be to find a way to allow the user to leave the switch up ('1') after each turn, this would eliminate the overwriting block issue, because the users would know which switches are up during each game.

## VI. Conclusions

Overall, the team had an exponential learning curve while completing this project. By combining the knowledge from class lectures along with the VHDL notes on Moodle the team was able to fully understand how to implement a VGA display using VHDL coding and the FPGA board. Once the display was implemented, the game logic was developed using specific code blocks learned in lab such as multiplexers and registers. By implementing Professor Llamocca's VHDL files such vga_ctrl_12b, vga_display, and mydebounce functions and editing them to fit the games needs combined with the the teams code blocks such as MUX and scoring_file, the team was able to make a successful game of tic-tac-toe.

## VI References

[1] VHDL Unit 7/ Unit 7 pdf (moodle.oakland.edu)