****

**Final Project**

**Fall 2021**

**10/18/2021**

**College of Engineering and Computing**

**Department of Electrical and Computer Engineering**

**ECE 287: Digital Systems**

**Hartup, Dave Ph.D.**

*Zack Burton*

*Riley Taylor*

**Introduction**

For this project, we are going to be making a tic-tac-toe game in VHDL. It will consist of switches for the users to place their x or o in a VGA (3x3) grid, one switch that can expand the game board into a (5x5) grid. It will also include three buttons, two to confirm both users’ inputs, and one for a reset.

Four switches will be for player 1 and player 2, excluding the switch that changes the board. If all switches are off, the destination of the x or o will be the first box in a (3x3) grid for both players. We will be using a for loop that will repeat until all boxes are filled or the game detects that a player has three x’s or o’s in a row, column or diagonal. Player 1 (x) will always go first, thus limiting confirmations on player 1’s side to five and player 2 (o) will have four confirmations. When player 1’s button is pressed, it will loop to player 2’s turn and keep looping back and forth until player 1 has pressed their confirmation button five times or a player has already won. To prevent any user from placing their x or o in a spot that is already used, they will receive a red light when they try to do so and if a user tries to go outside the range of the VGA grid they will see “DNE” displayed on the board when they try and confirm their move.

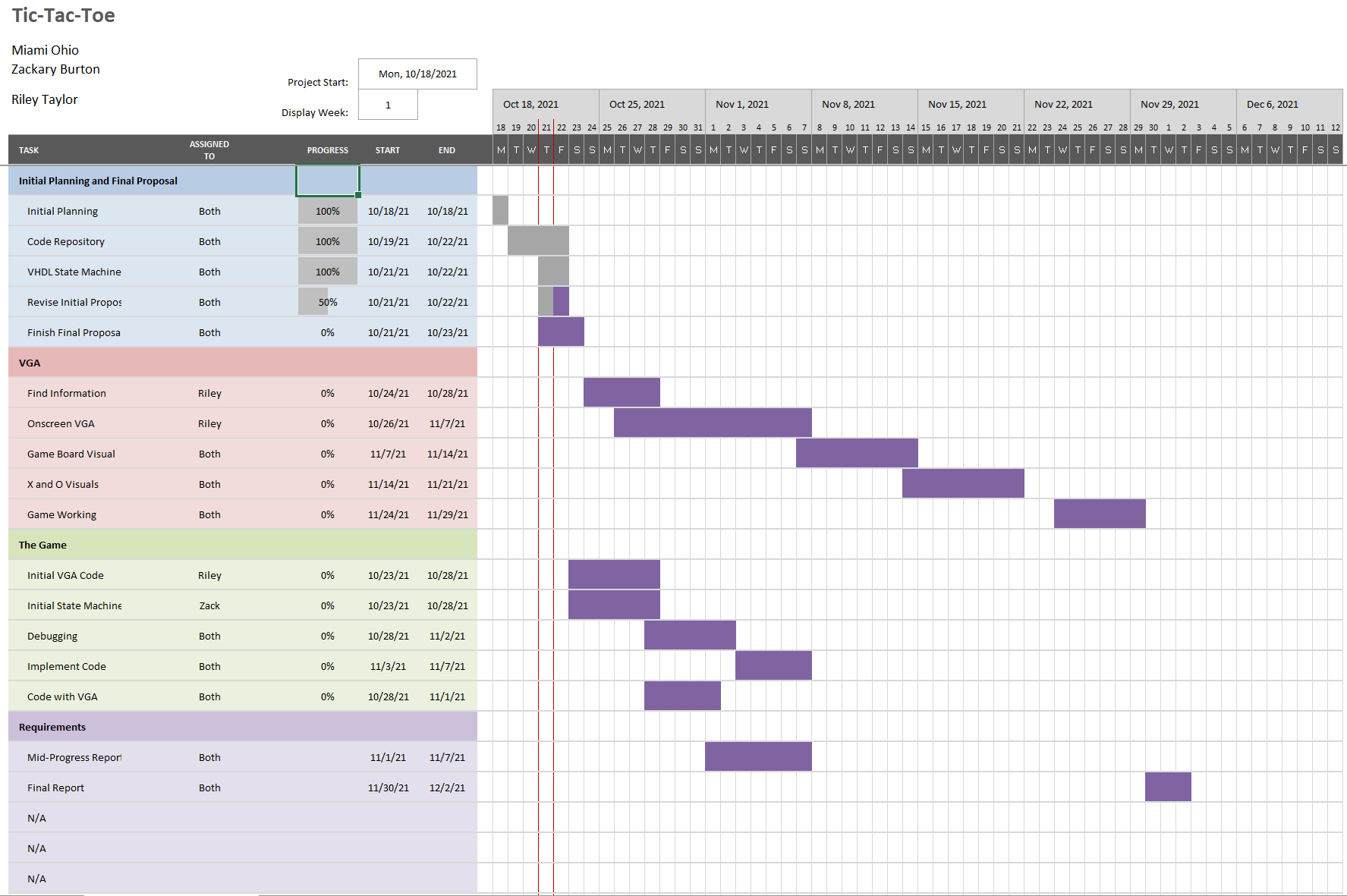
Once player 1 has used their confirmation button five times, the system will check if there are three bits in a row, column, or diagonal from the same user. If there is none, “TIE” will be displayed in hex. If player 1 wins, “PL 1 WINS” will be displayed, and if player 2 wins, then “PL 2 WINS” will be displayed. A player may win before player 1 uses all five confirmations, so a check will be implemented after using the confirmation button three times and then moving forward after each confirmation. The game will reset 10 seconds after the end, or if the user resets it manually. Finally, when making the expanded board, we will expand the code we already have for the first one to fit a 5x5 grid. The preceding will be modeled using a state machine so slowly build up the code of the game.

**Code Repository**

We will be using Git as our code repository for this project. This allows us to work on the project from different devices and push our work to the repository for the other person to see what was worked on, as well as access our Gantt chart and proposal. Our final work will be submitted by zipping the files from the repository.

**Gantt Chart and Planning**

The following image is our Gantt chart. The chart will also be available in the repository as we may need to adjust certain time blocks in the future. This chart will help us in knowing if we are ahead or behind schedule, which is important for a big project like this. We can also fill in percentages when we are complete or in progress with one of the subsections.



**Meetings**

We are planning to meet four times a week plus our lab time for two hour intervals. Tuesday, Thursday, Saturday and Sunday look like our best days to meet. We will work in accordance with the Gantt chart.

**Challenges**

In the process of completing the project, we will encounter some risks and challenges. We might not complete the 5x5 due to the amount of tasks that need to be done. To help minimize this risk we will work on the project part by part throughout the upcoming weeks with the functionality of the game. Once we have the basics and functionality down then we will move onto more user friendly assets including some visual queues and displays on our board. Another issue we might come across is getting lost in our code, which is why following the state machine will be imperative to our success.