Introduction to Verilog (FPGA)

Roshni Pande[1] , **Sri Laxmi Ganna**[2]
ee16btech11032@iith.ac.in, ee16btech11008@iith.ac.in

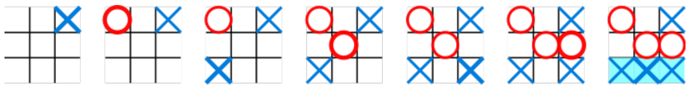

Figure 1: Game rules

1 Objective

In this report, the detailed explanation of how to prepare the infamous pen-paper game Tic-Tac-Toe on FPGA is given. The Software and Hardware specifications are given below. The code and all the other related files have been uploaded on github and can be publically accessed on this link https://github.com/ee16btech11032/FPGA-Project

2 Introduction

Tic-tac-toe (American English), Noughts and Crosses (British English), or Xs and Os is a paper-and-pencil game for two players who are labelled as X and O. Each takes turns marking the spaces in a 33 grid. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row wins the game.
The game can be generalized to an m,n,k-game in which two players alternate placing stones of their own color on an mn board, with the goal of getting k of their own color in a row.If played properly, the game will end in a draw.
For the current project, we have built a (3,3,3)-game where X is represented by Green LED lights and O is represented by Red LED lights. The winner is shown with their respective color of LED.

3 Requirements

3.1 Software Requiremnts

Each author name must be followed by:

- Language - VHDL

3.2 Hardware Requiremnts

Each author name must be followed by:

- Icoboard
- Breadboard / PCB

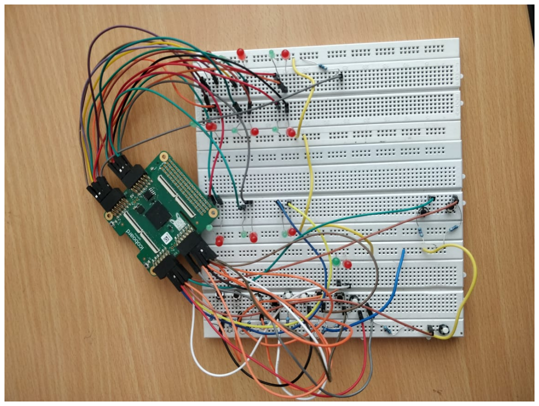

Figure 2: Working Model on Bread Board

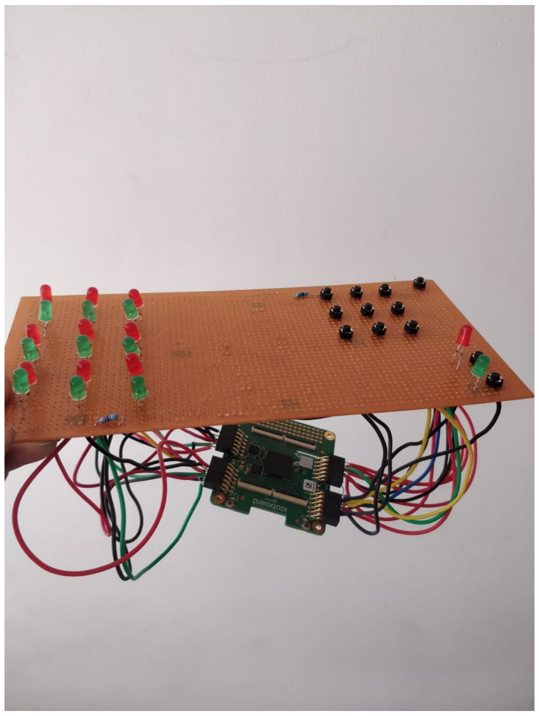

Figure 3: Working Model on PCB

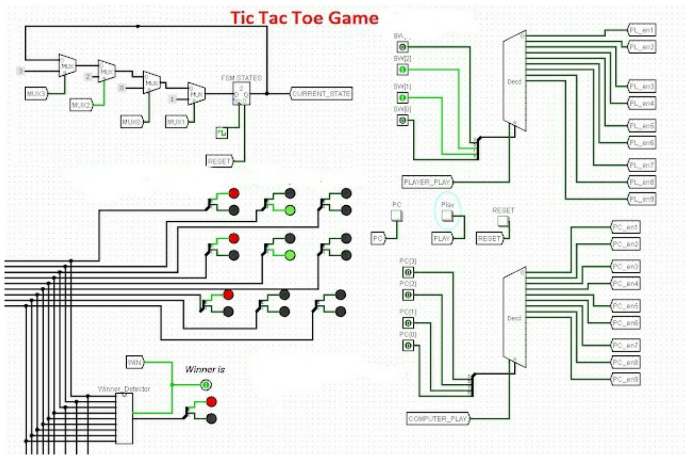

Figure 4: Circuit Diagram

- Push Buttons
- 2 colors LEDs (here, Green and Red)
- male-to-male wires
- RaspberryPi
- Resistors

4 Working

When the player plays the game, a 2-bit value is stored into one of the nine positions in the 3x3 grid like Xs/ Os in the real paper-and-pencil version. 2'b00 is stored into a position when neither the player or computer played in that position. Similarly, 2'b01 (X) is the value to be stored when the player played in the position and 2'b10 (O) is the value to be saved when the next player played in the position.
The player1/ player 2 plays the game by pressing their corresponding button. Red/ Green LED is lit in a position when the position is played by player1 / player2 respectively.
The player1/ player2 wins the game when successfully placing three similar (01-Xs) or (10-Os) values in the following row pairs: (1,2,3) ; (4,5,6); (7,8,9) ; (1,4,7) ; (2,5,8); (3,6,9); (1,5,9); (3,5,7).
The winner detecting circuit is designed to find the winner when the above winning rule is matched. To detect an illegal move, a comparator is needed to check if the current position was already played by either the computer or player. Moreover, "No space" detector is to check if all the positions are played and no winner is found.

5 Implementation

5.1 Software Implementation

1. IDLE(00): when waiting for the player1 / player 2 to play or when resetting the circuit, the FSM is at the IDLE state.
2. PLAYER(01): The player1 turns to play and "01" to be stored into the decoded position.
3. COMPUTER(10): The player 2 turns to play and "01" to be stored into the decoded position.
4. Game-over(11): The game is finished when there is a winner or no more space to play.

Inputs of the controller of the Tic Tac Toe game:
a. Reset :
Reset = 1: Reset the game when in the Game-Over state.
Reset = 0: The game begins.
b. Play:
Play = 1: When in the IDLE state, play = 1 is to switch the controller to the PLAYER1 state and the player plays.
Play =0: Stay in the IDLE state.
c. PC
PC = 1: When in player 2 state, PC = 1 is to switch to the IDLE state and the player 2 plays.
PC =0 : stay in player 2 state.
d. Illegal-move
Illegal-move = 0: When in PLAYER1 state, Illegal-move = 0 is to switch to player 2 state and let player 2 plays when PC = 1.
Illegal-move = 1: Illegal moving from the player1/ player 2 and switch to the IDLE state.
e. No-space
No-space = 0: still have space to play, continue the game.
No-space = 1: no more space to play, game over, and need to reset the game before playing again.
f. Win
Win = 0: Still waiting for the winner
Win = 1: There is a winner, finish the game, and need to reset the game before playing again.

5.2 Hardware Implementation

Assign pins to all outputs and inputs and connect outputs and inputs according to the pin configuration in .pcf file that has been in the aforementioned Github link. Use raspberryPi as CPU so we connect icoBoard to raspberryPi and run the codes.

6 Conclusion

In this report we have seen how to implement the game Tic-Tac-Toe on FPGA. We first learnt the principles of the game. Then we have seen the how to go about the working of this game and to implement it on our FPGA in software implementation. So, by this we conclude that the major bottleneck in this project is to write the algorithm and write corresponding code in verilog. Through this project, the age-old game has finally reached a new technological level and can now be played with a proper setup, having proper turns between players without any cheating.