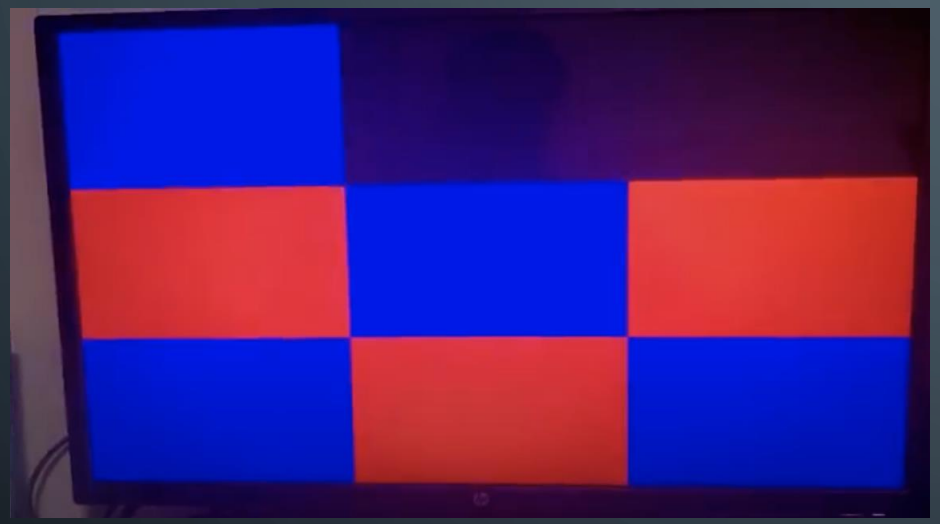
TIC-TAC-TOE

PROJECT COMPLETED BY:

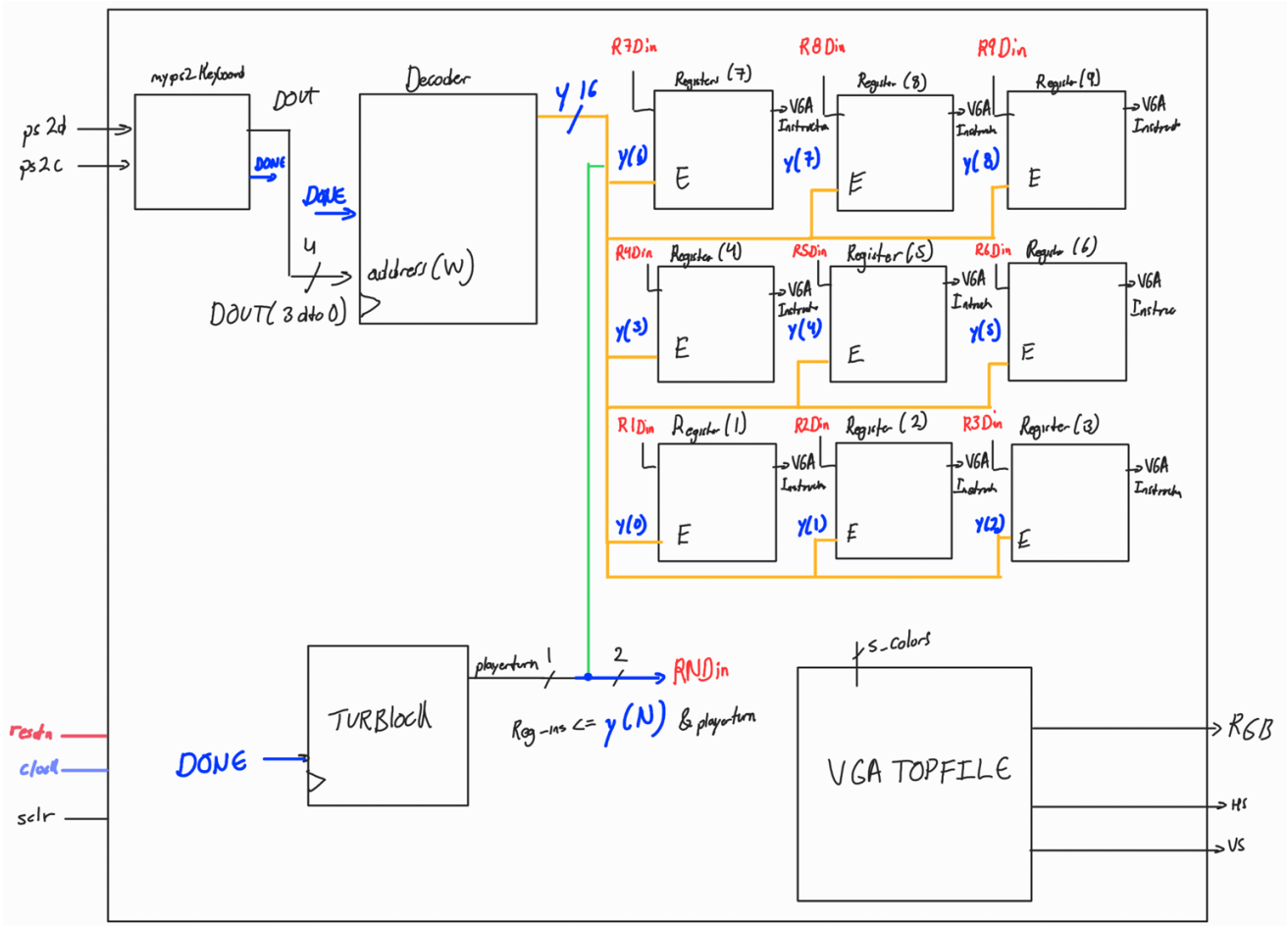
ANGEL BAUTISTA, ANTONIO MONTAGNA , PAOLO NIKAJ AND IAN BETZ

Agenda



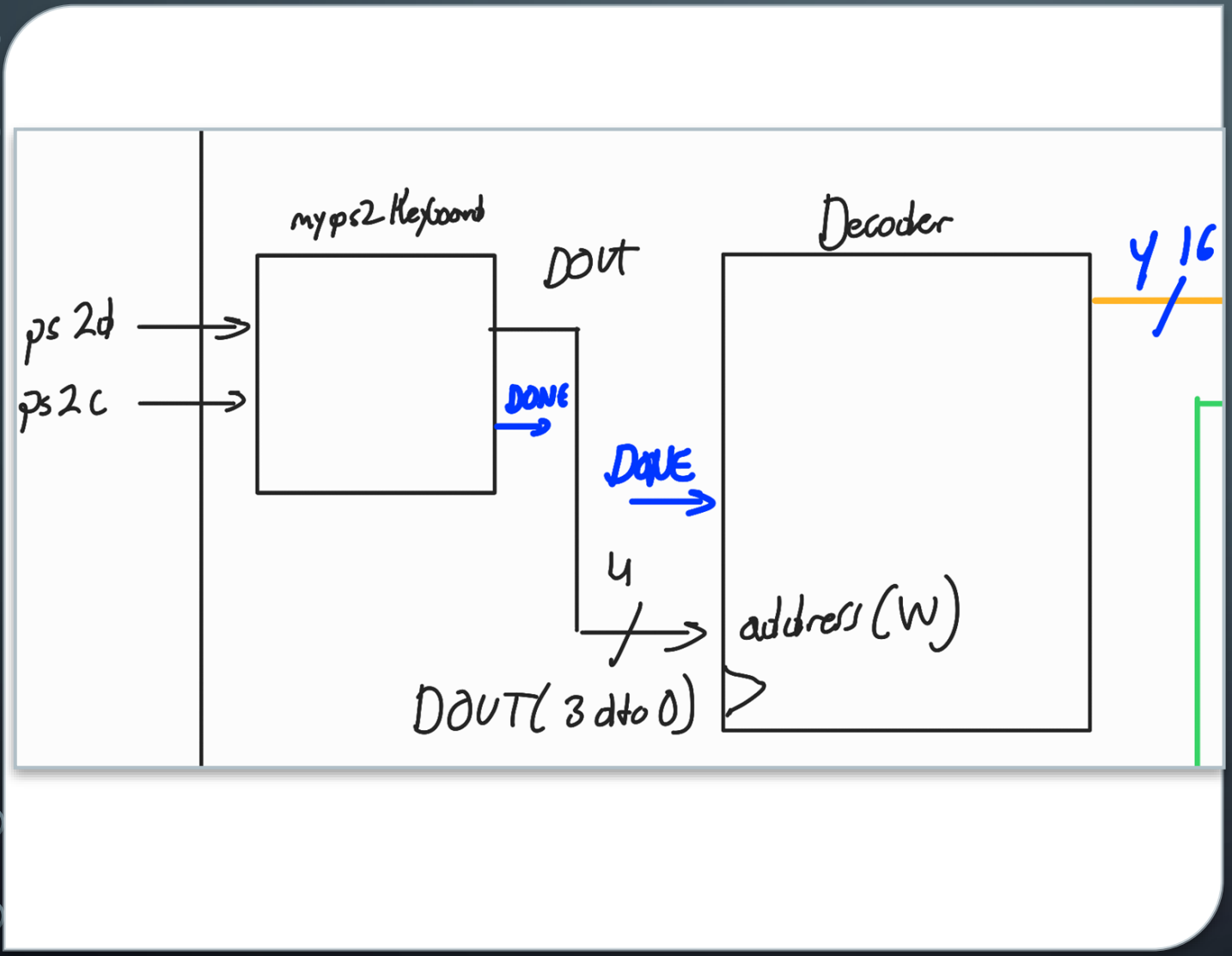
- Operation of a Tic Tac Toe game
- Circuit Components
  - PS2/Keyboard and Decoder
  - Memory
  - Game Logic
    - FSM
  - VGA display
- Hardware Implementation
  - Video Demo

TIC TAC TOE game



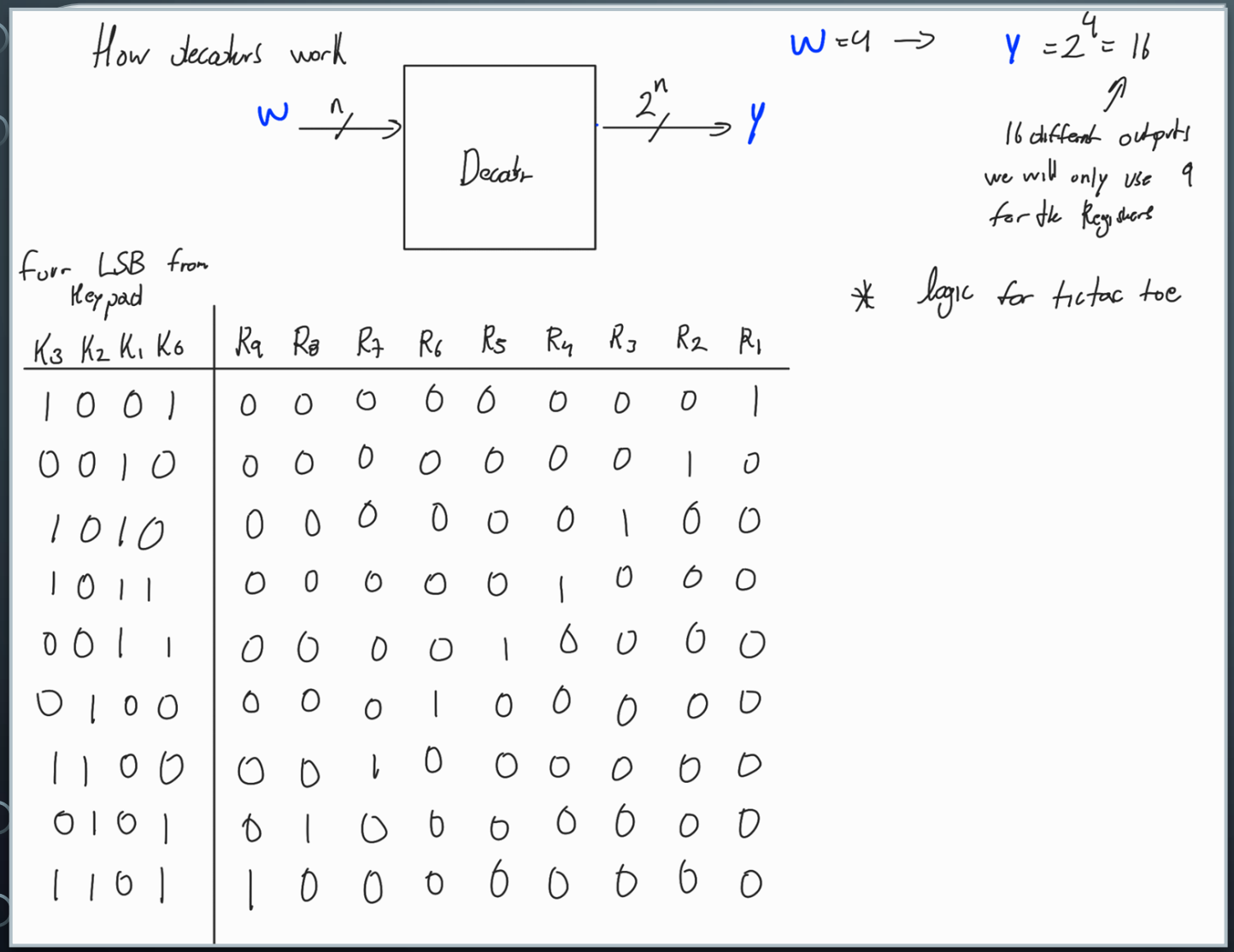
- The purpose of this project is to make a functional Tic Tac Toe game.
- Game data is saved on memory registers, these registers are enable by a decoder.
- The game logic is performed by the Turn Block Circuit as well as the decoder register.

PS/2 Keyboard/Decoder

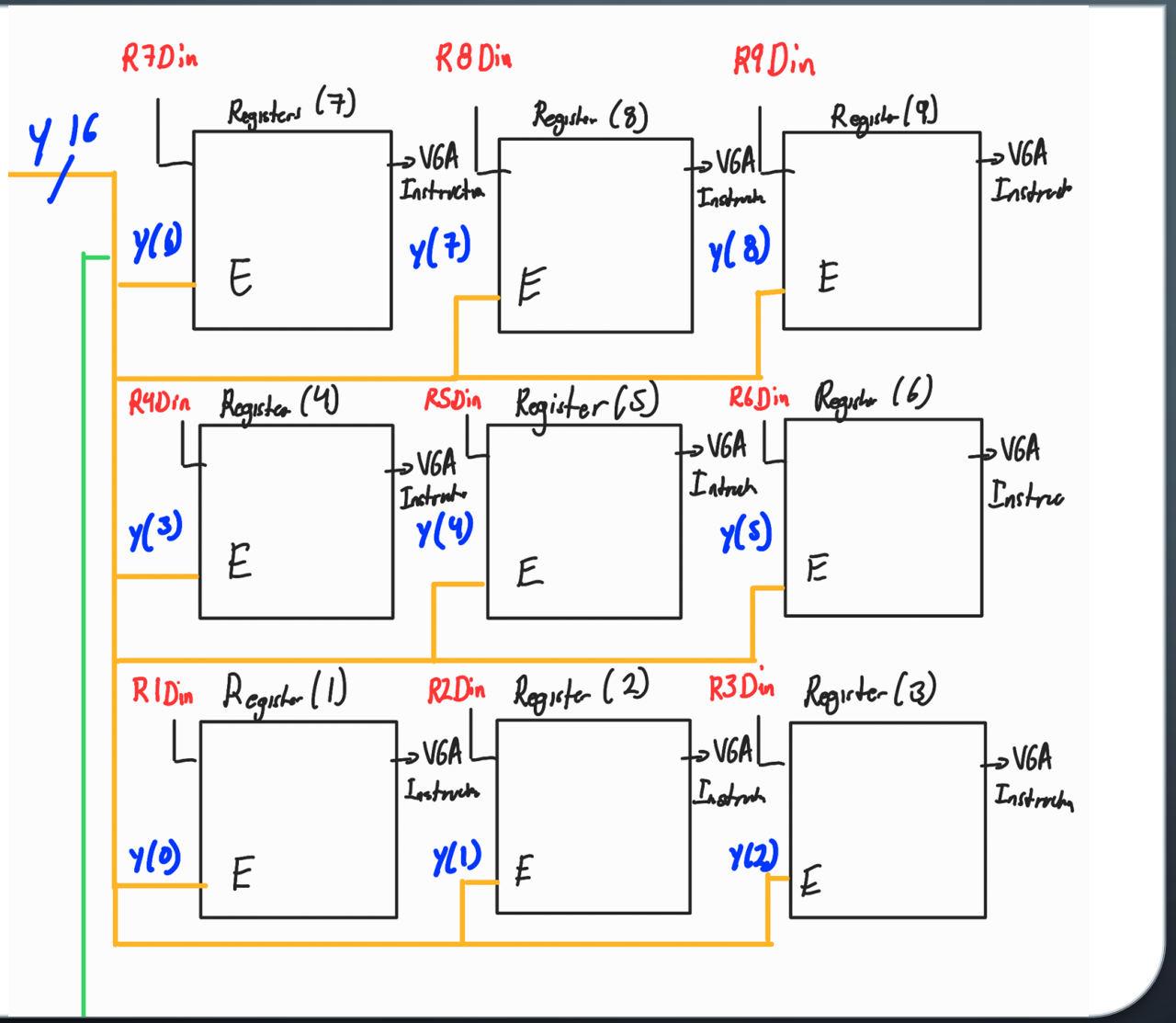


- "myps2_keyboard" was implemented in this project
- When a key is pressed a "Done" signal enable the decoder and game logic circuit.

PS/2 Keyboard/Decoder

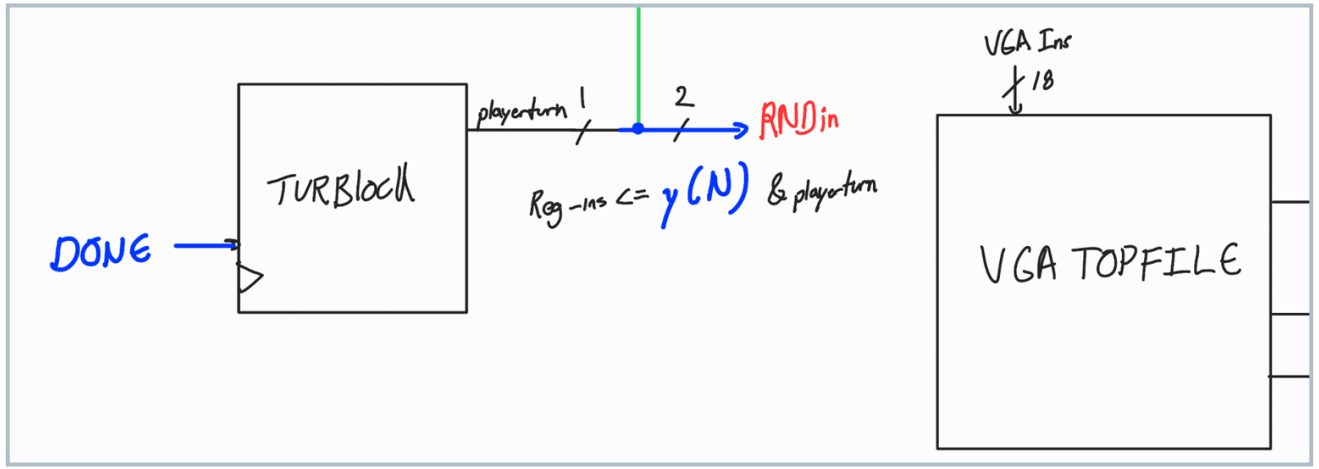


- Each value from the keypad is mapped to a single enable bit.
- Only 9 bits were used out of 16 as enable register.
- Only one enable line is active per turn.



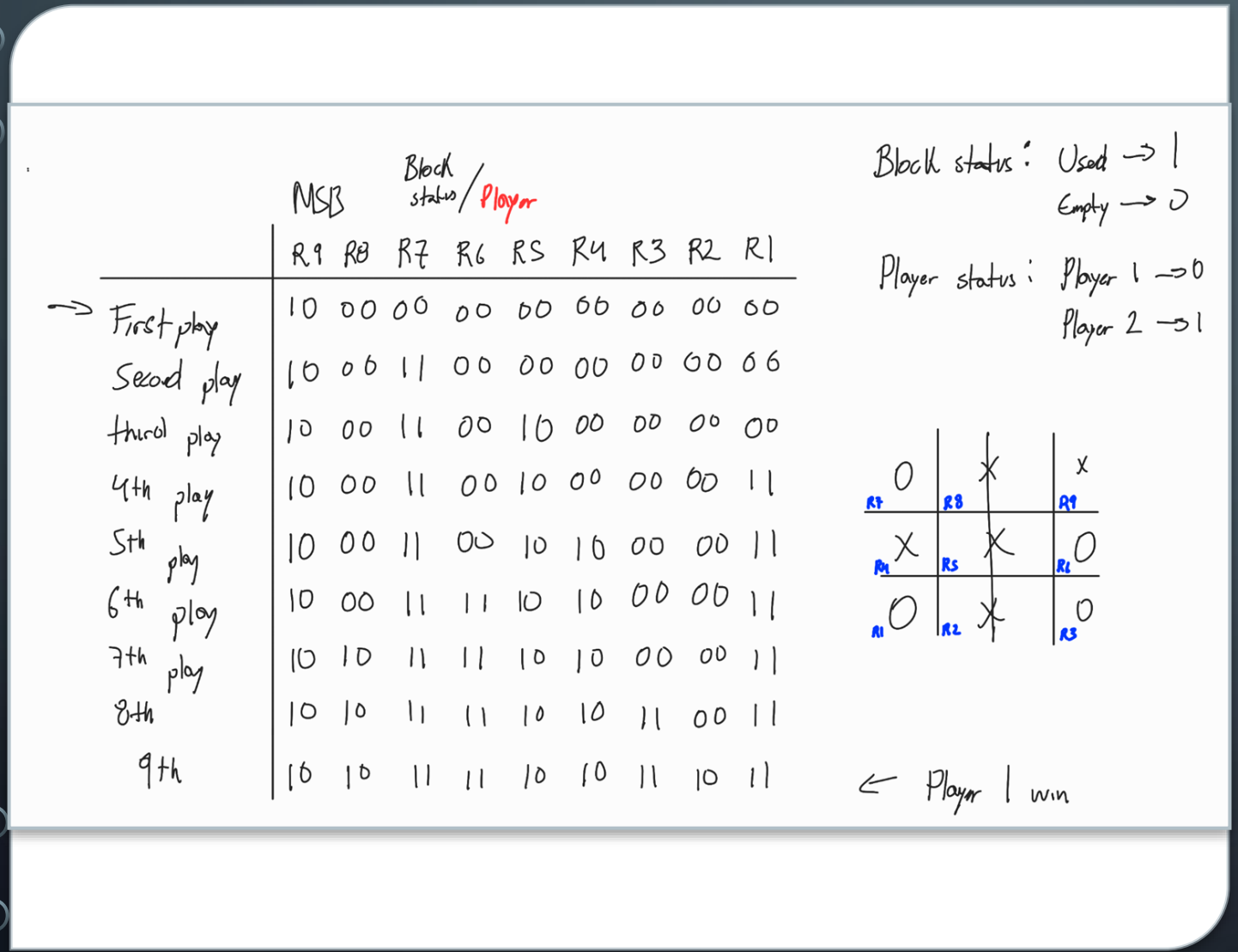
Game Registers

- 2-bit register with enable and reset.
- Sequential circuits that store the values of signals.
- 2 bit register that has a storage element that can hold 2 bits.
- The registers used in this Game go from R1 to R9.



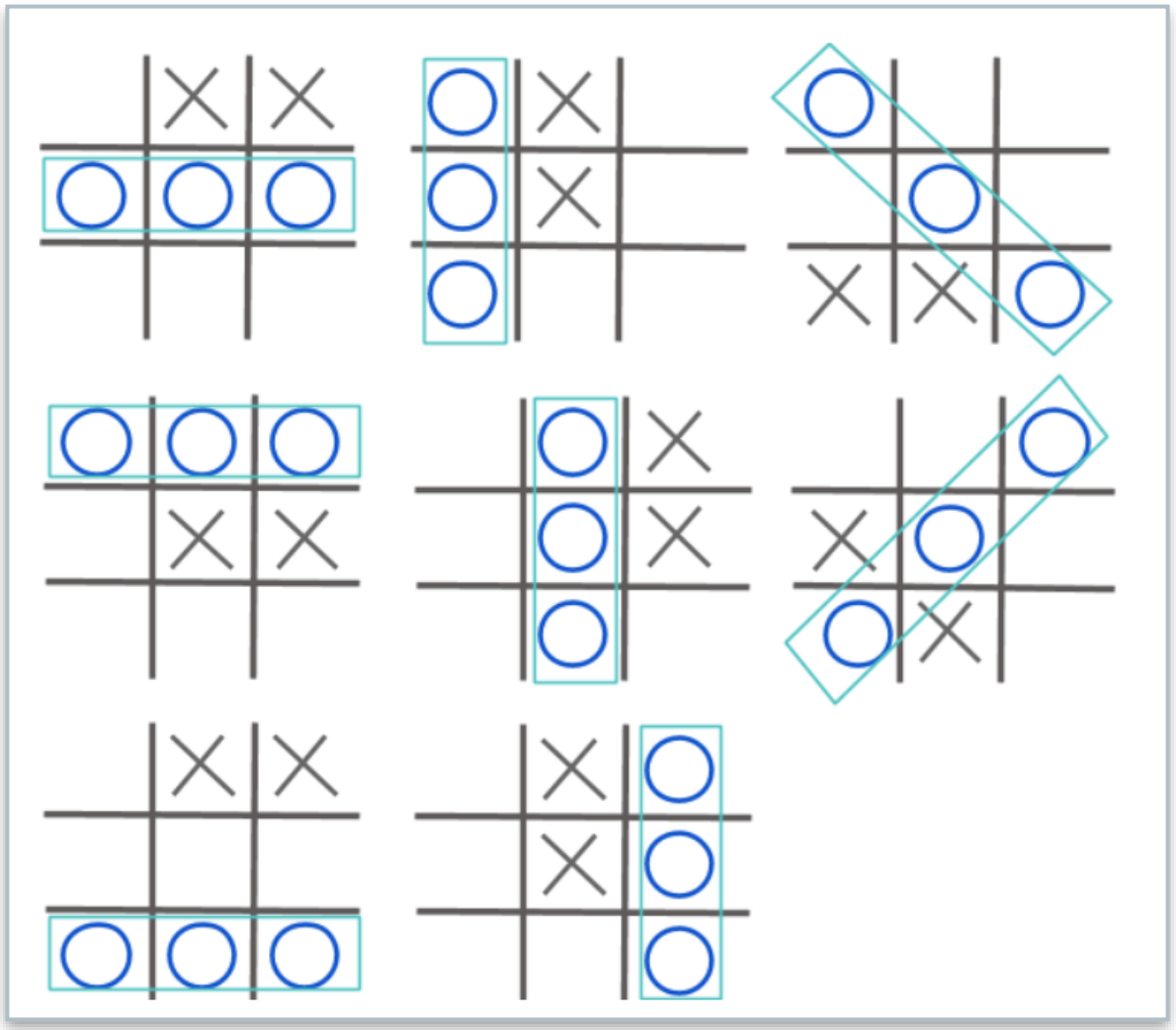
Game Logic

- TurnBlock is a simple logic instruction that alternates a single bit every time a key is pressed on the keyboard.
- Reg_ins is a concatenation of the playerturn bit and the numpad output.



Game Logic

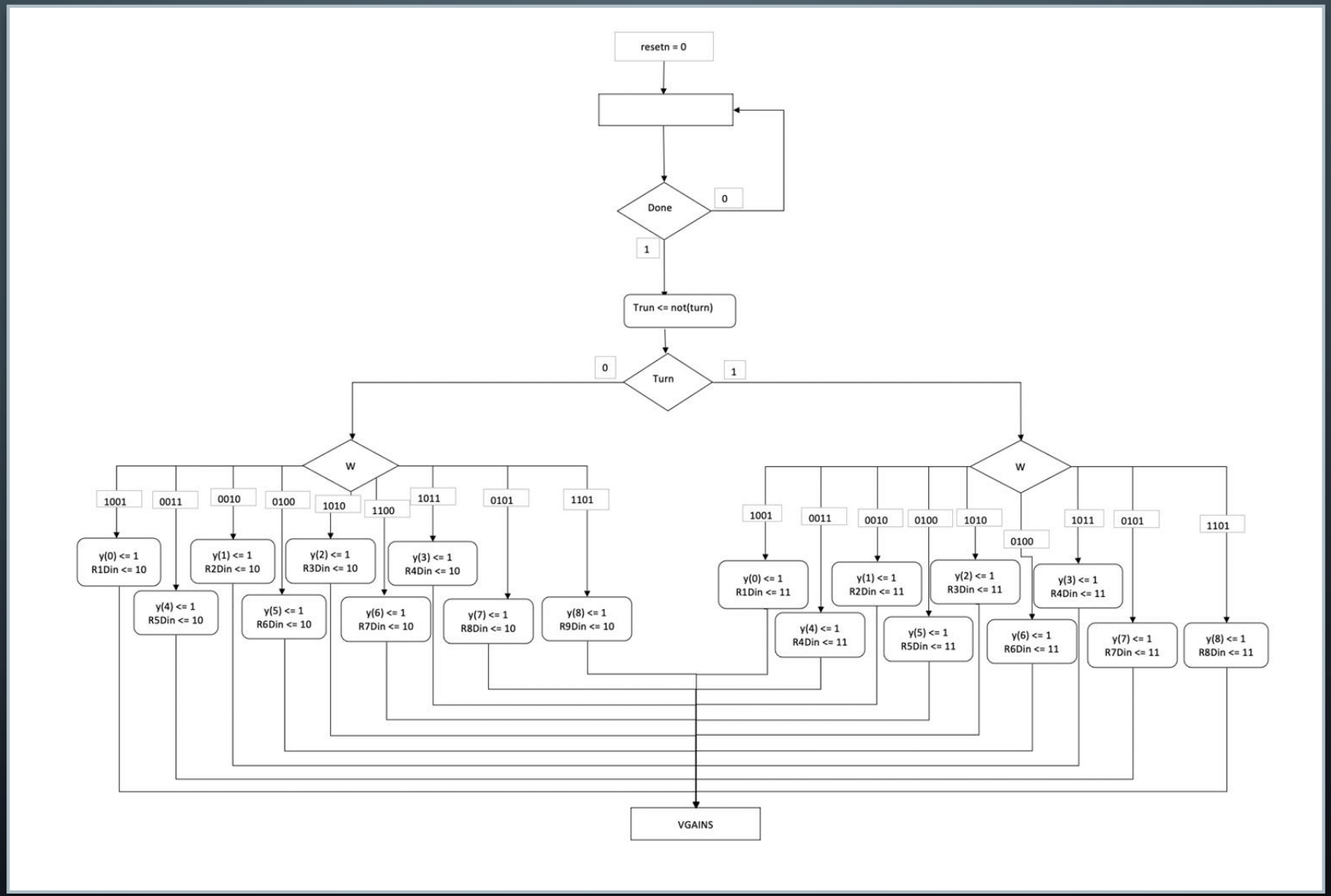
- An 18-bit binary value (VGAins) is used in order to track the game progress and instruct the VGA.
- This truth table shows the VGAins value during a sample gameplay.



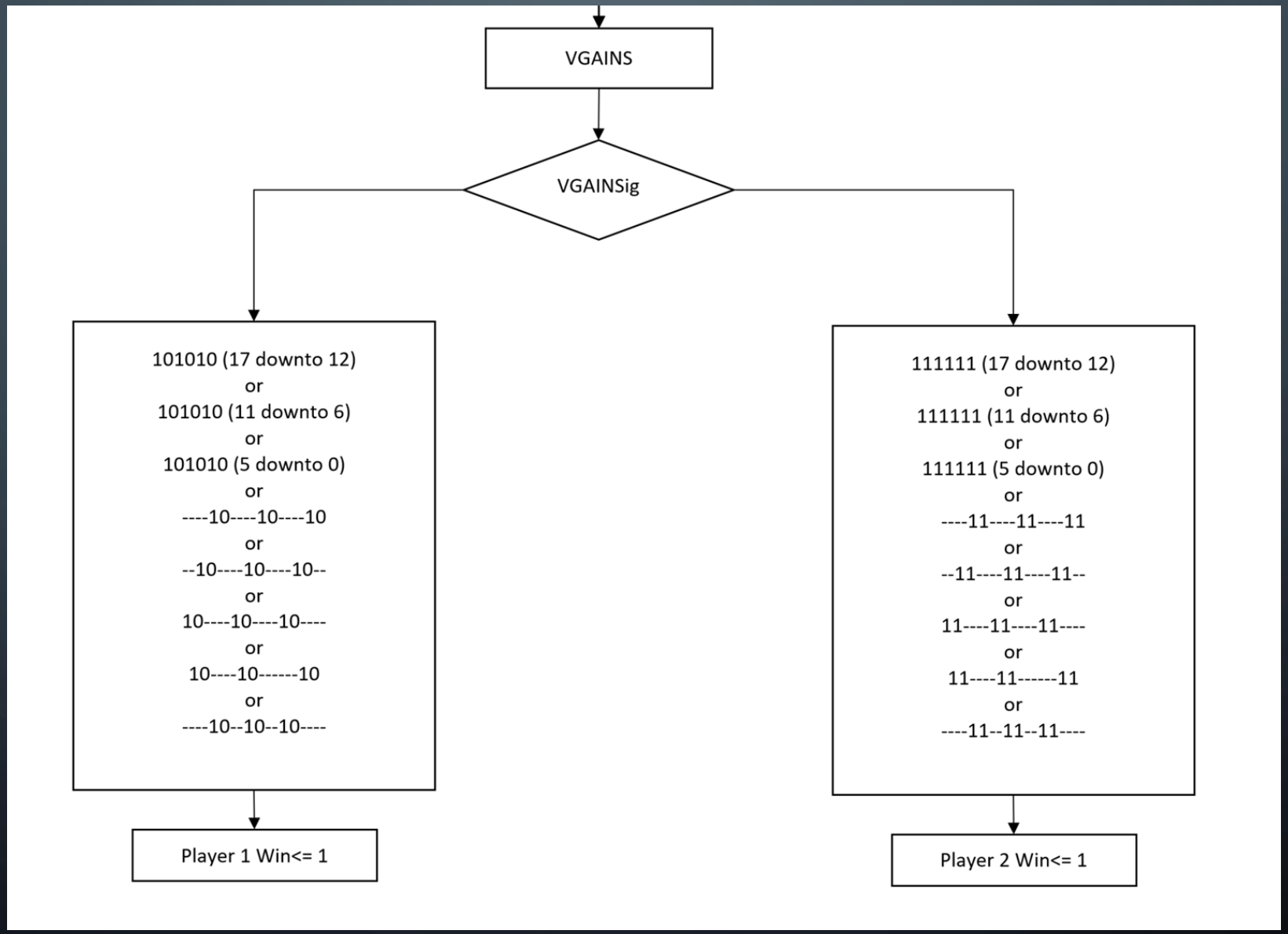
Game Logic

- Winner is displayed on the built-in LEDs
- Right half LEDs-P1 WIN
- Left half LEDs-P2 WIN
- Detecting a winner is done with concatenating the corresponding signals for each combination along with the player turn bit.

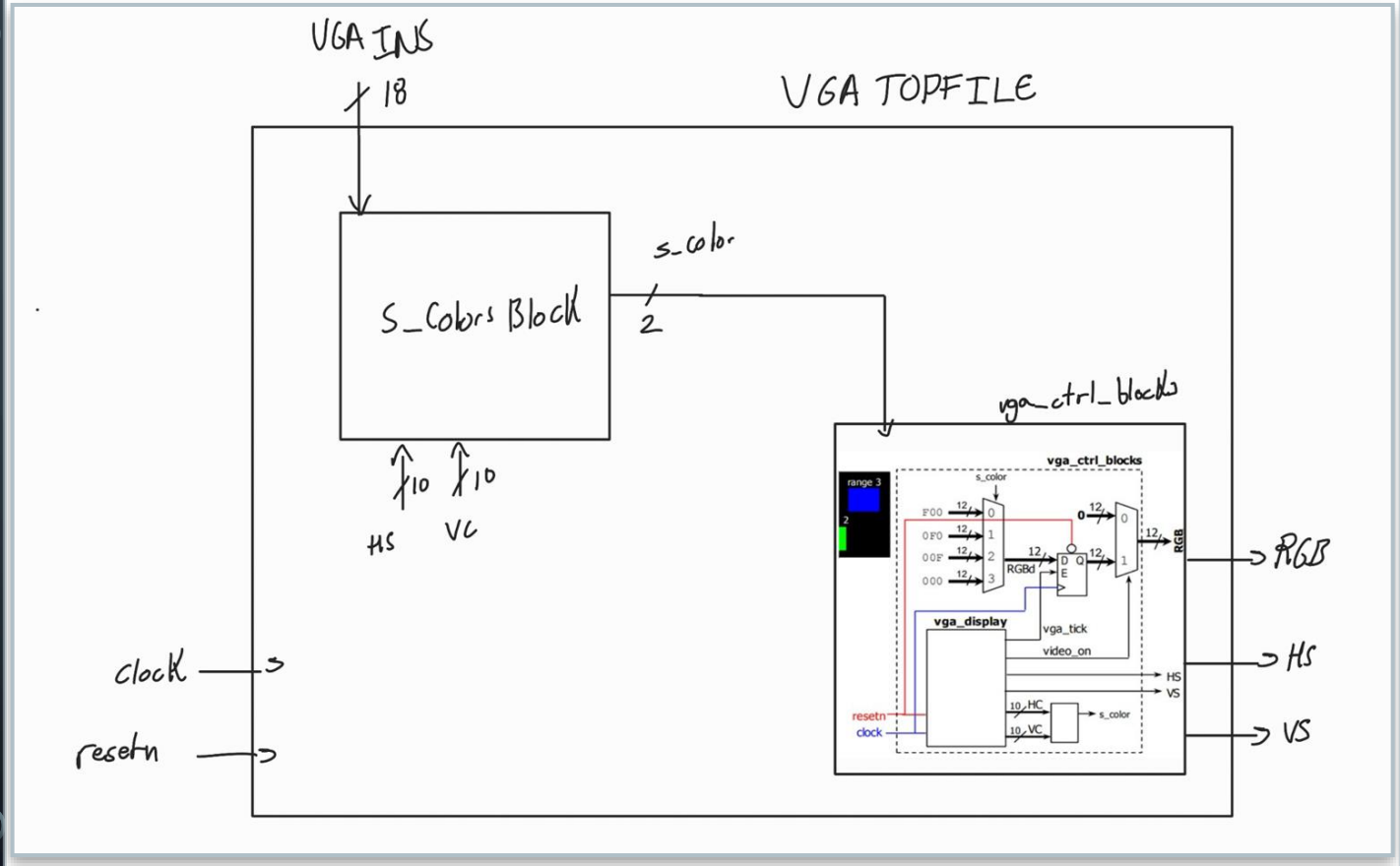
Game Logic FSM



Game Logic FSM

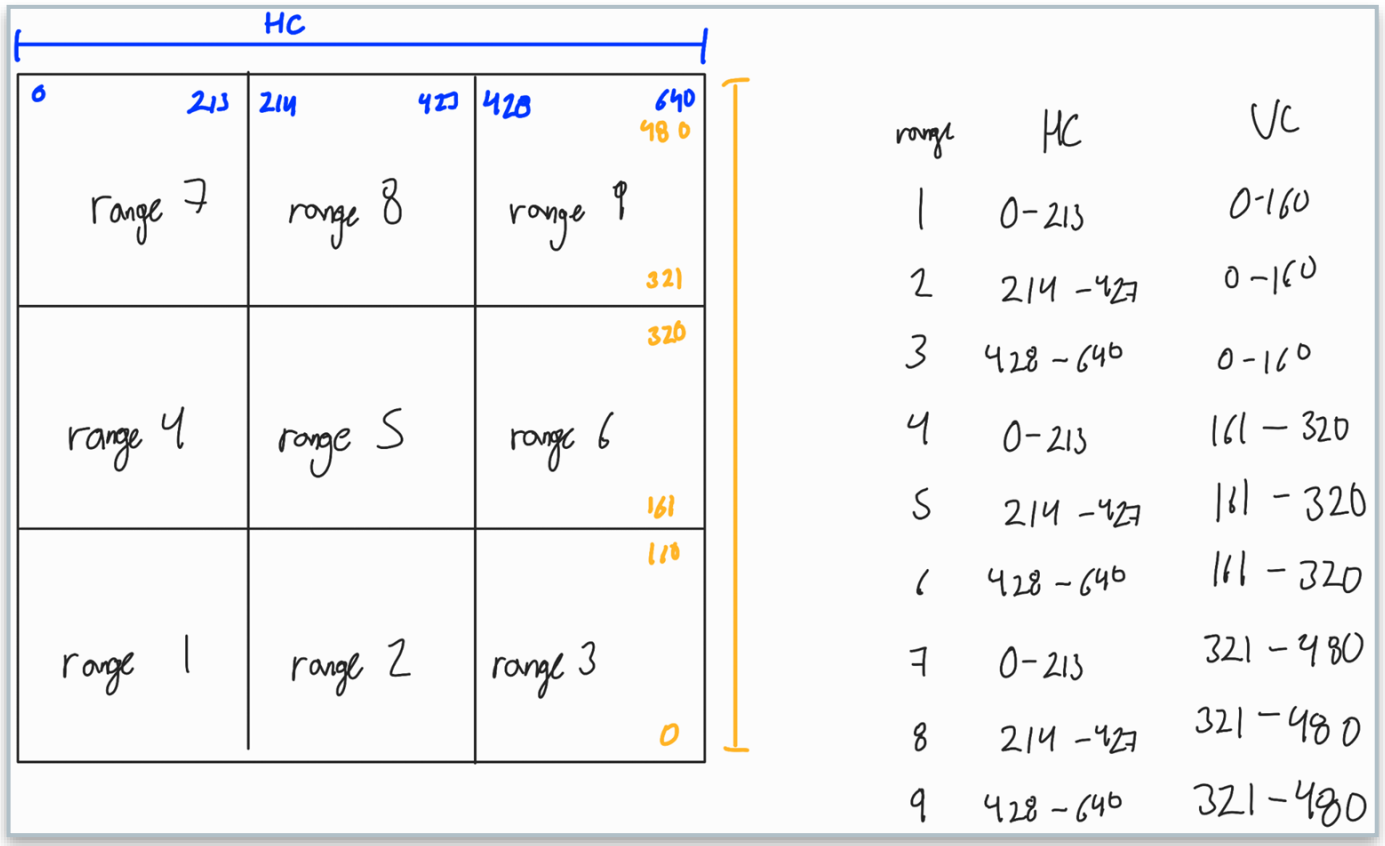


VGA



- The VGA topfile uses the inputted VGAINS value in order to determine what will be outputted to the display.
- S_colors block functions to decode the VGAINS value and determine when and where to display on the screen

VGA

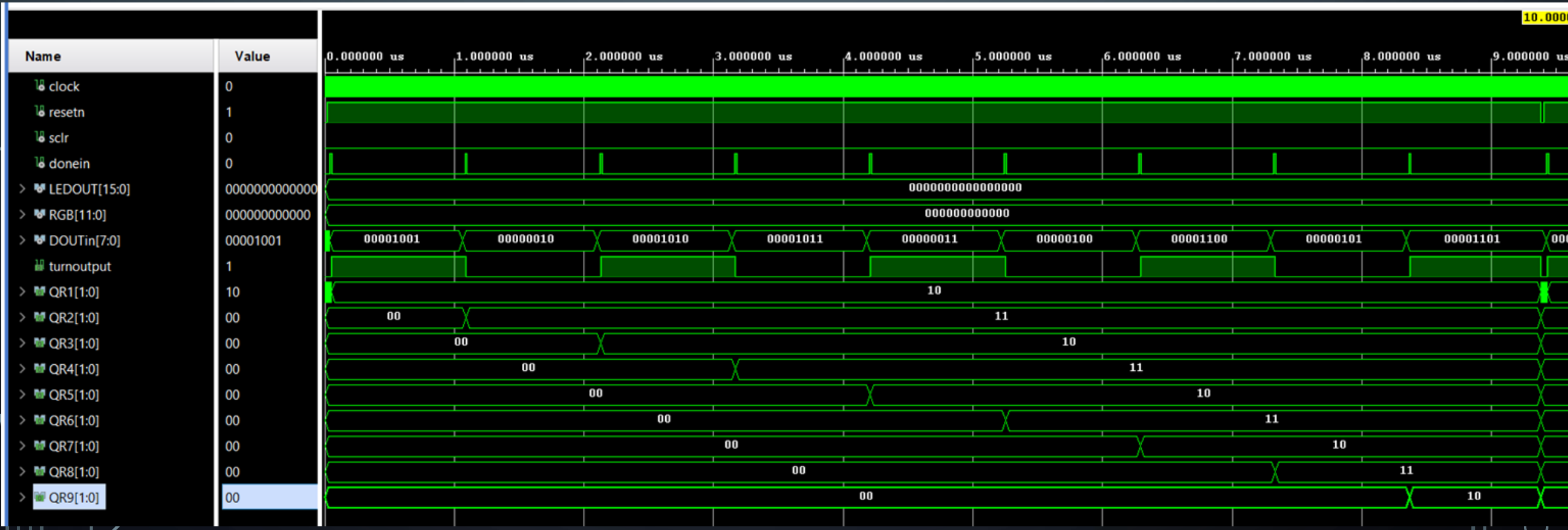


range	HC	VC
1	0-213	0-160
2	214-427	0-160
3	428-640	0-160
4	0-213	161-320
5	214-427	161-320
6	428-640	161-320
7	0-213	321-480
8	214-427	321-480
9	428-640	321-480

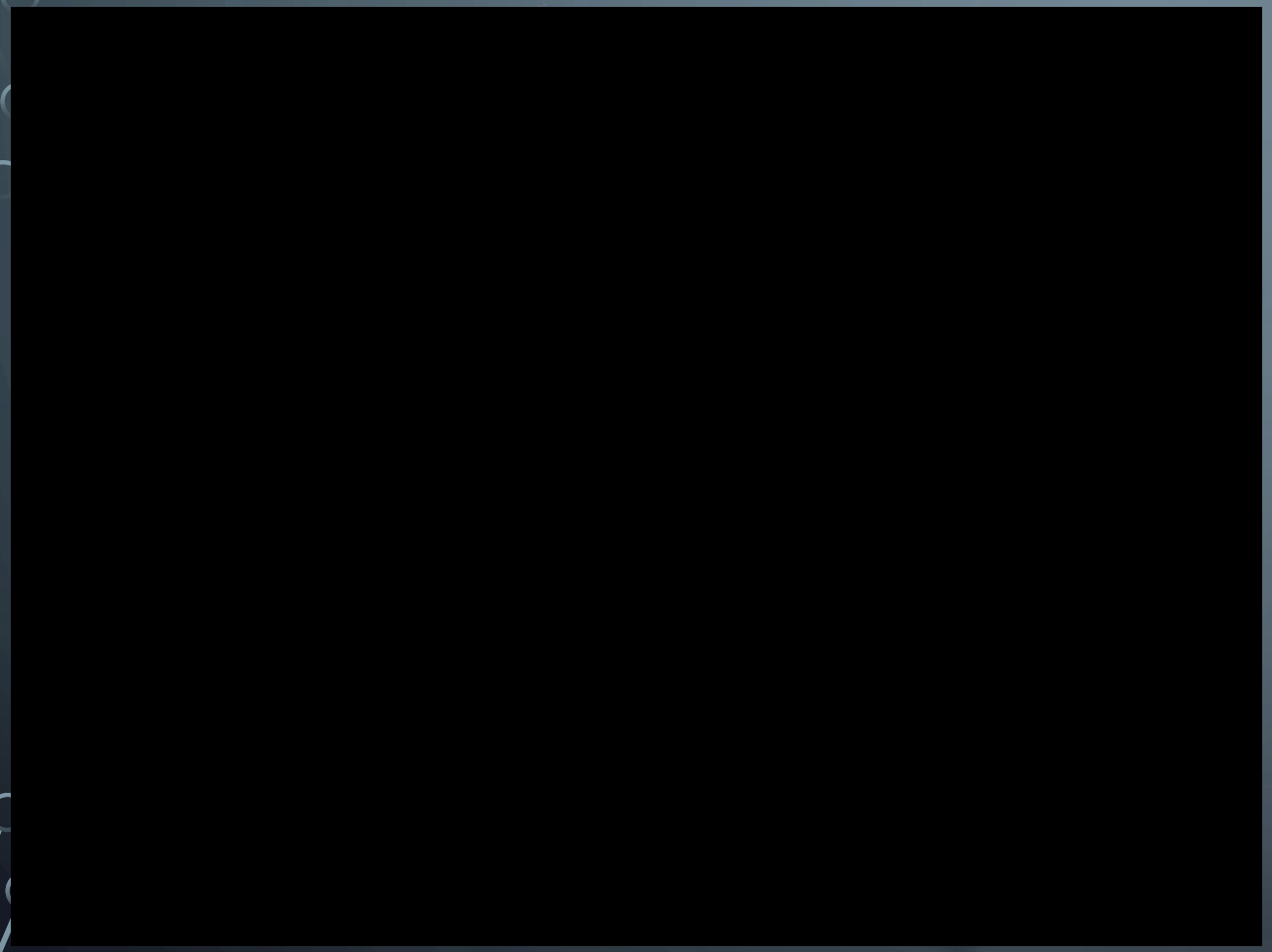
- The Display is divided into nine rectangles. each range will illuminate when selected. red for player 1 and blue for player 2.
- the VGA was mapped using HC values 0-640, and VC values 0-480 to properly fit our display.

Testbench

- Demonstrates that the keyboard inputs fill the respective registers as we would expect.



Hardware Implementation



- Circuit Implementation!
- Link to Video of Circuit Implementation
  - (Link to our video goes here)

Questions?