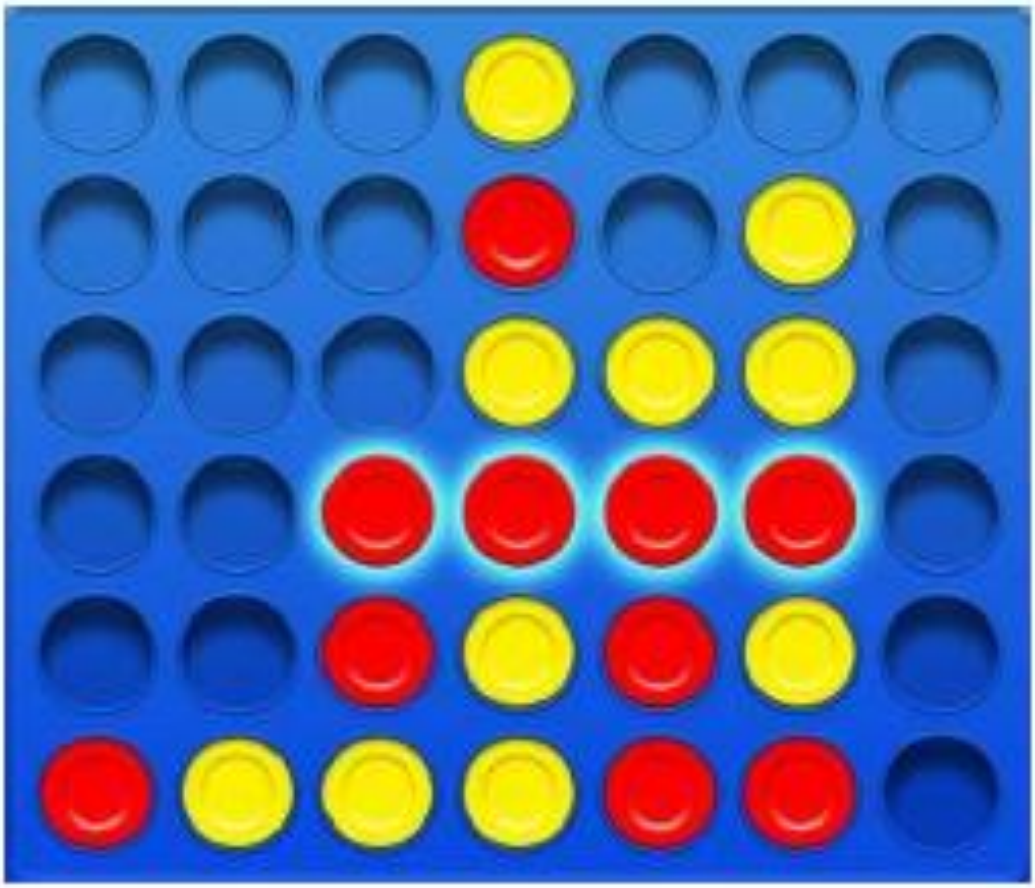


SCORE 4

VLSI DESIGN

DEMOCRITUS UNIVERSITY OF THRACE
COMPANY ADDRESS

PROJECT SCORE 4

PROJECT SCORE 4

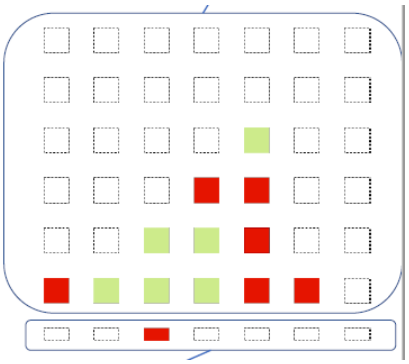
You are asked to implement the game 'SCORE 4' on the lab's FPGA board:

- In the game, the first one to arrange 4 squares of the same color in the same line wins.
- Each state of the game will be depicted in a VGA screen.
- Input can be given either from the FPGA board's buttons, or from keyboard buttons.

WHAT SHOULD BE DEPICTED ON THE SCREEN

A 6x7 panel with red or green squares.

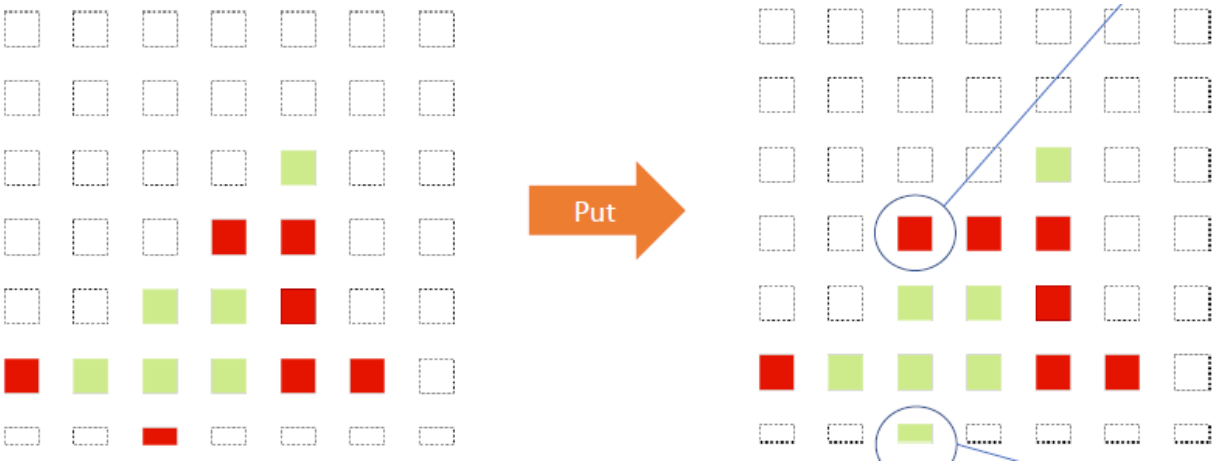
Empty squares should not be visible on the screen.



→ A line with a narrow rectangular showing the active column and the active color.
→ The next square should be added to the active column by pressing the button 'PUT'.
→ The square entered should be the same color as the active color.
→ Square moves left or right by pressing the 'LEFT' and 'RIGHT' buttons.

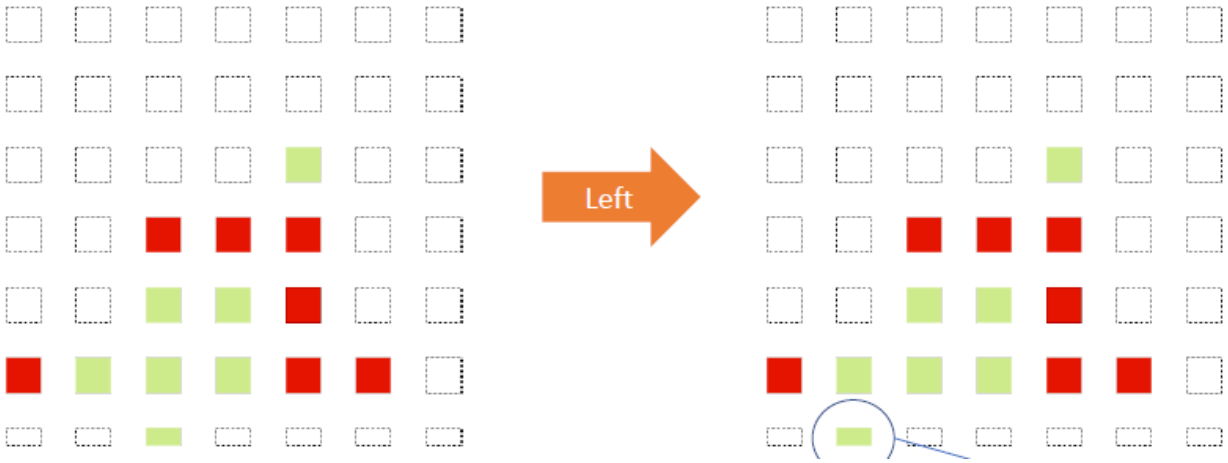
PUT FUNCTION

→ Assign the 'PUT' function on a button of the FPGA board or of the keyboard.
→ On the first free line (starting from the bottom) of the active column, a square of the active colour is added.
  ○ If column is full, it should not change.
→ The active colour changes right away (different player).



ACTIVE COLUMN CHANGE

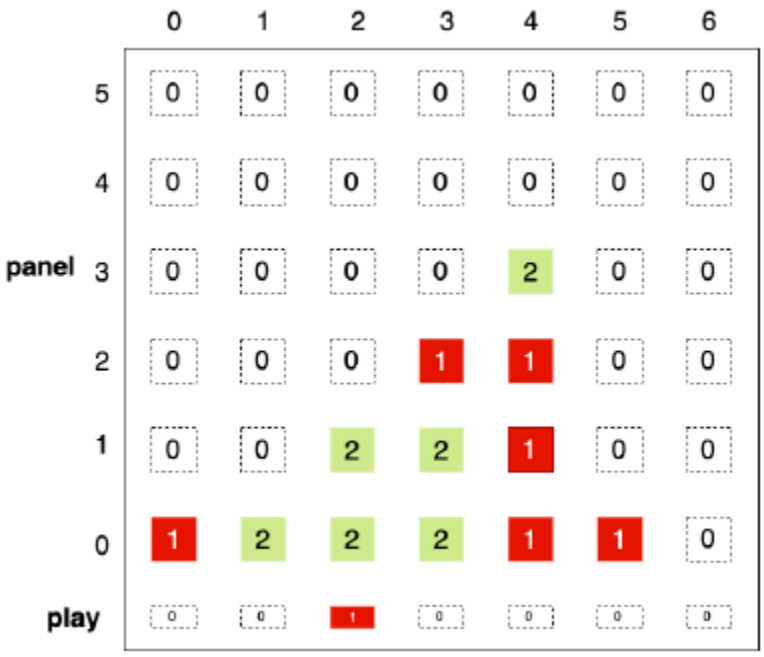
→ Assign 'RIGHT' and 'LEFT' functions on two buttons of the FPGA board or of the keyboard.
→ Left button press: Change of the active column to the left.
→ Right button press: Change of the active column to the right.



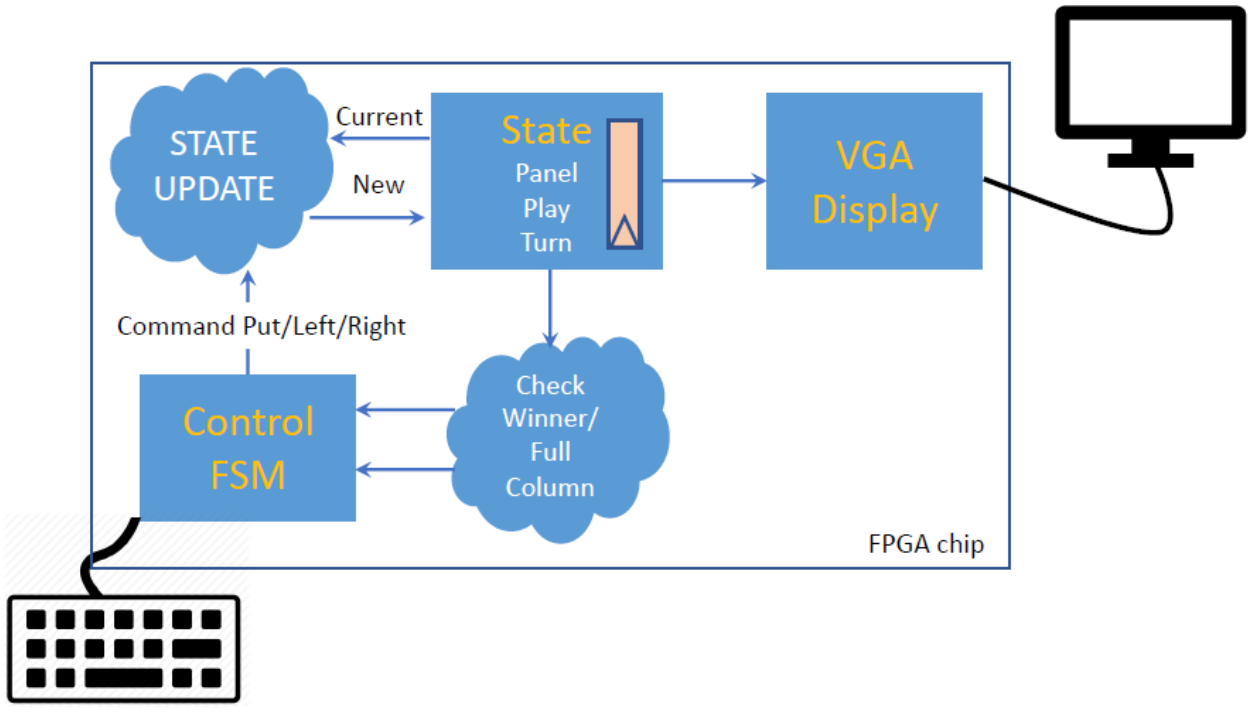
CHANGE MANAGEMENT PROCESS

The circuit's state is depicted in 3 different variables:

→ Panel: 2D array using 2 bits to describe each position on the panel.
panel[i][j] = 0 (for empty), 1 (for red) or 2 (for green).
→ Play: 1D array 1x7 using 1 bit to describe if each column is active or idle.
play[i] = 0 (idle) or 1 (active).
→ Turn: 1-bit variable
0: red's turn
1: green's turn



ORGANIZING THE CIRCUIT



Control FSM: Defines the states of the game.

- → Wait for input (Left, Right, Put)
- → Pressing the right action button:
  - LEFT / RIGHT -> Update Play State
  - PUT -> Update Panel State, Update Turn State
  - Update of the state is implemented with the logic state update.
- → Check for winner / full column
- → VGA display
  - Regardless of all the rest. It reads the varying states and depicts them accordingly.

DESIGN STEPS

1. Score 4 table
   - → Depicting current state on the screen.
   - → Panel state variables ‘play’ and ‘turn’ would be constant.
2. State check and refresh.
   - → Finding of the first free line in a column (for ‘put’ function).
   - → Check for full table (end of the game).
   - → Check for winning 4-squares combination.
   - → Functions mentioned above could be implemented either as a combination logic block (1 cycle) or sequential logic (repetitive).
3. FSM and final design.
   - → Merging of the steps in states and actions on the FSM.

ADDITIONAL BONUS FEATURES

- → Highlighting the winning 4 squares.
- → Usage of sprites (images) instead of squares.
  - Use a simple sprite ROM to do that.
- → Change of state of 2 boards through general purpose IO pins on the FPGA.
  - Each player using their own FPGA board and playing while they see only their screen.
- → Automatic Score 4 player.
  - One player playing manually for the one color
  - The other color's moves are automatically decided by the automatic player.

SCREENSHOT OF SCREEN OUTPUT