



# **ECE241 Final Project**

# **Snake Game**


Karen Weng & Tyra Lehn



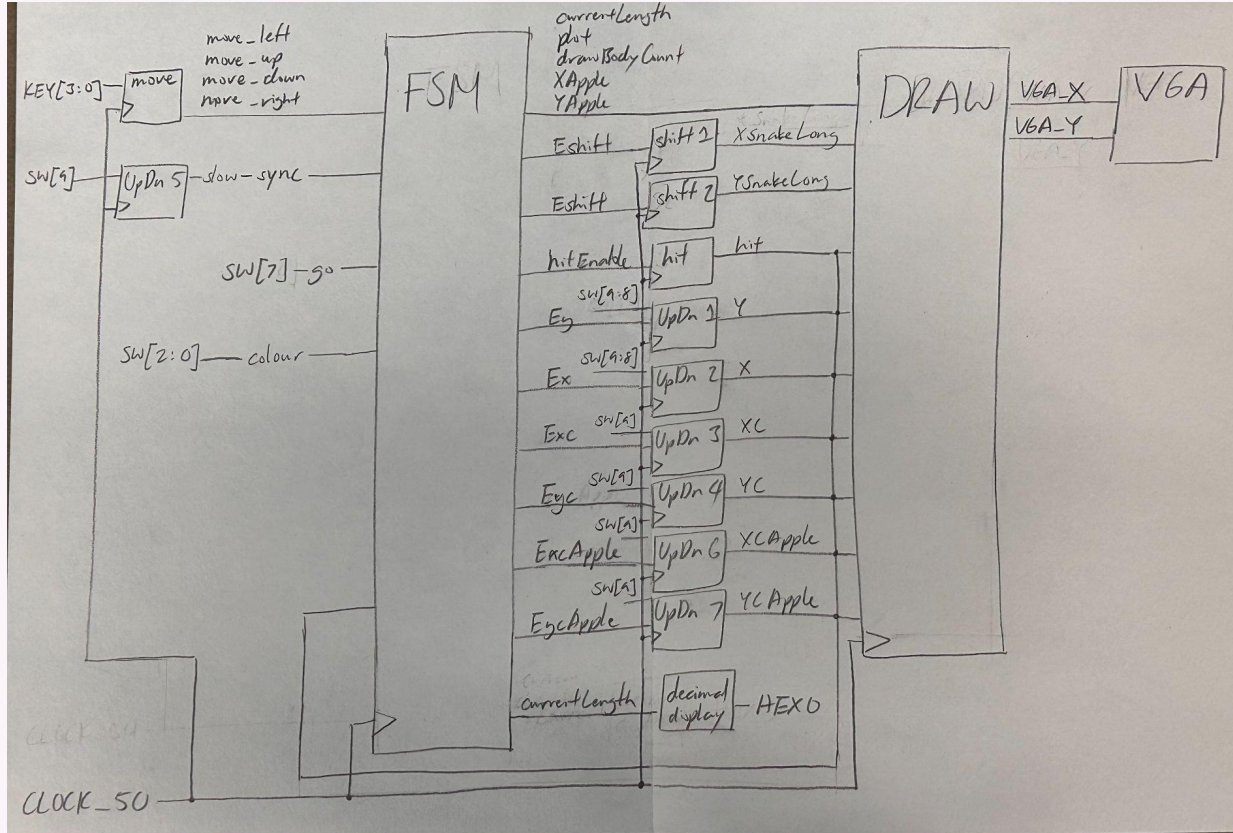


# Game Description

## Recreation of classic Snake Game

- **How to Play:**
    - Move snake around with key inputs
    - Avoid hitting walls and itself
    - Eat apples to grow longer
  - **Goal**
    - Eat as many apples as possible
    - Grow as long as possible
- 

# Block Diagram

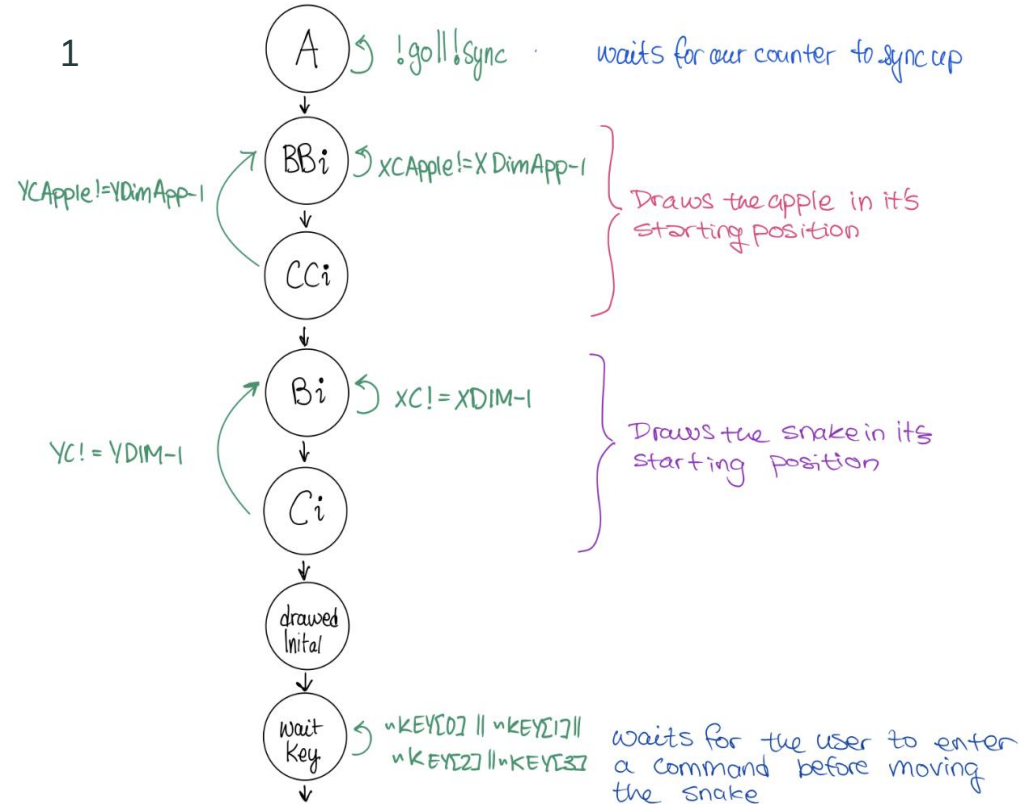




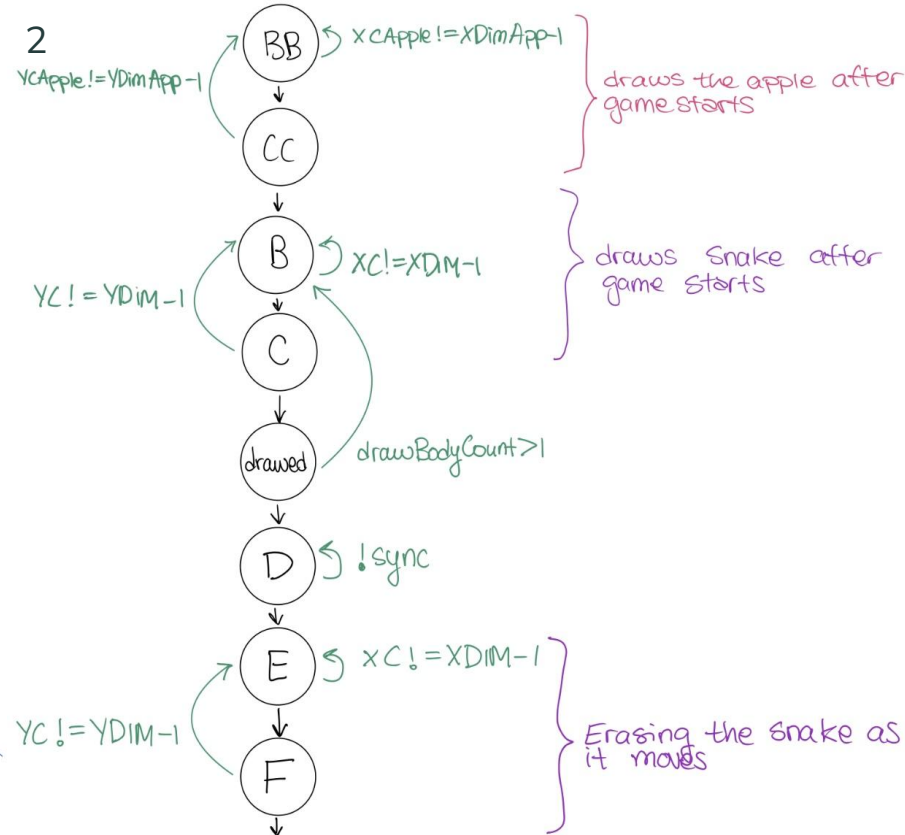
# PART 1:

## State Diagram (FSM)

1

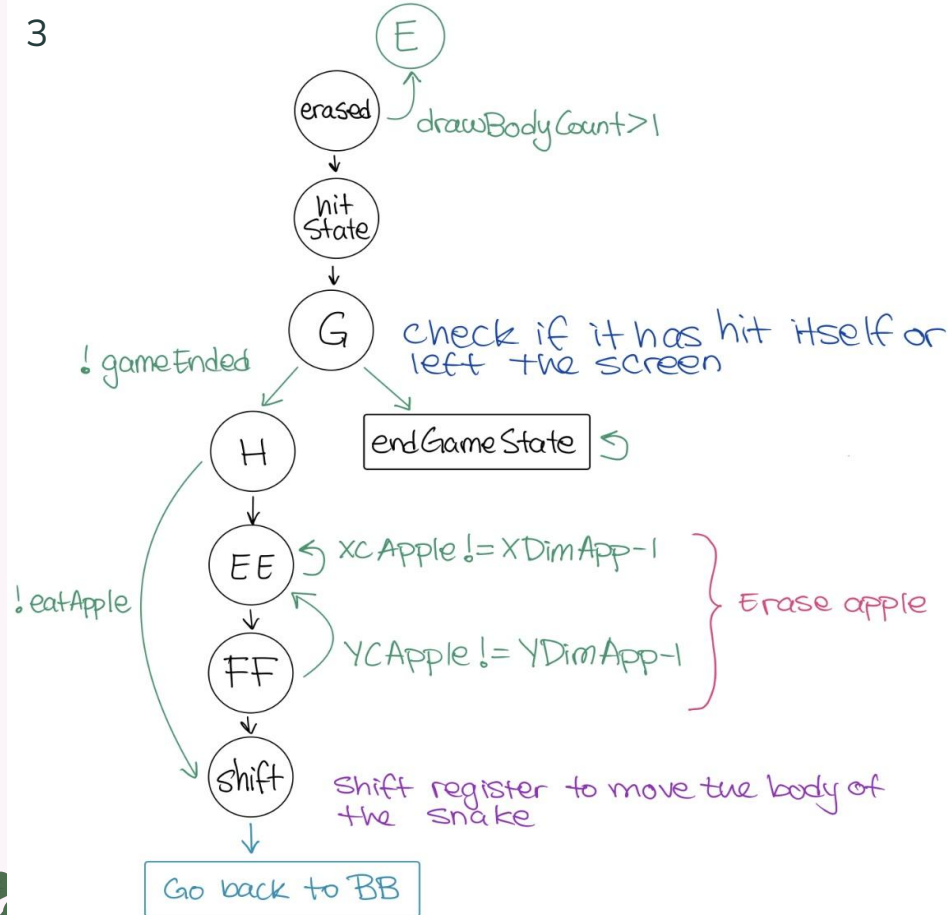


2

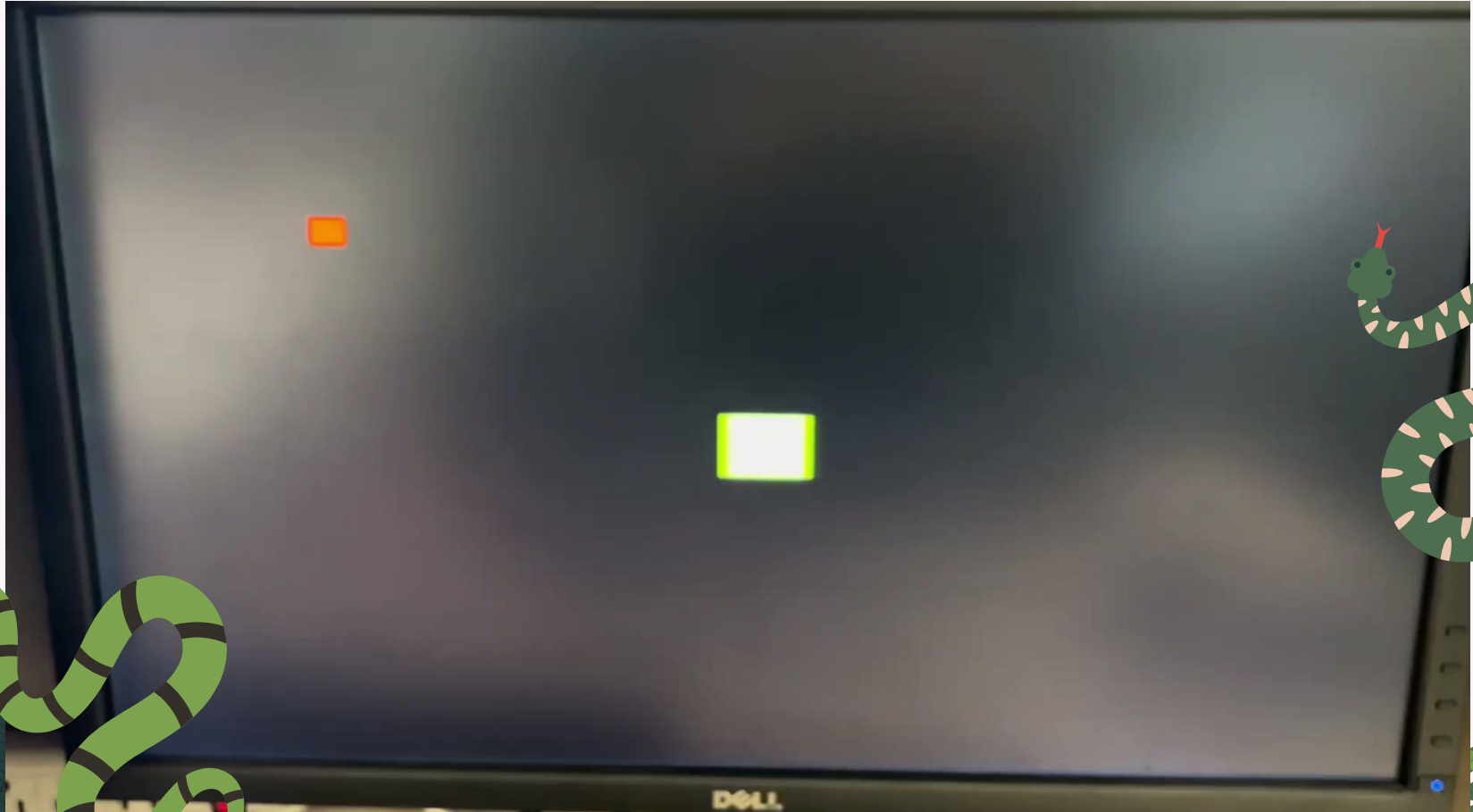


# PART 2: State Diagram (FSM)

3

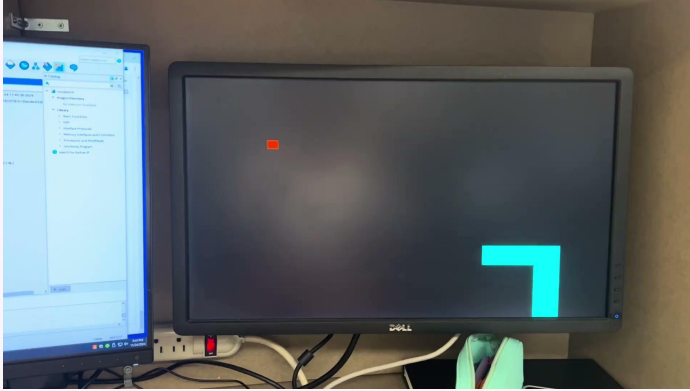


# Process Work: Eating an apple with baby snake





# Eats Apple & End Game



# Bugs, Issues & Solutions

## Apple Not Displaying

- Y\_D and y\_Q became 5 bit variables but I hadn't changed the size of the reg

## Too Many Apple Generating at Once

- After eating one apple it would generate multiple other apples at once
- Fixed by changing the location that the generate function is called in the FSM

## Snake Dies Right After Starting Game

- Snake head wasn't moving but body is shifting into itself

## Counter

- The counter would increment too many times per apple collision





# Multiple Lengths

```
if;
tEnable;

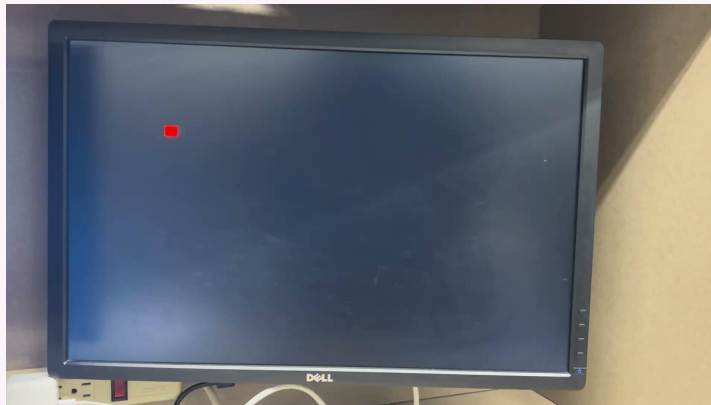
always @(posedge Sw[4] or posedge Sw[5])
begin
    if (Sw[5])
        begin
            currentLength <= 4'b1; // Reset to zero when sw[5] is high
        end
    else if (currentLength < 4'b1111)
        begin
            currentLength <= currentLength + 1; // Increment on the pos
        end
end

[3:0] currentLength;
currentLength = 1;
drawBodyCount;
[8 : maxLength * XDIM -1 : 0] XSnakeLong;
[7 : maxLength * YDIM -1 : 0] YSnakeLong;

assign XSnakeLong = {x0, x1, x2, x3};
assign YSnakeLong = {y0, y1, y2, y3};

shift;

initial begin
```



```
parameter maxLength = 6;
// wire maxLength;
assign maxLength = 2;

// reg [3:0] currentLength;
wire hit;
reg hittable;

// always @(posedge Sw[4] or posedge Sw[5])
// begin
//     if (Sw[5])
//         begin
//             currentLength <= 4'b1; // Reset to zero when sw[5] is high
//         end
//     else if (currentLength < 4'b1111)
//         begin
//             currentLength <= currentLength + 1; // Increment on the pos
//         end
//     end

wire [3:0] currentLength;
assign currentLength = 2;
reg [3:0] drawBodyCount;
wire [8 : maxLength * XDIM -1 : 0] XSnakeLong;
wire [7 : maxLength * YDIM -1 : 0] YSnakeLong;

// assign XSnakeLong = {x0, x1, x2, x3};
// assign YSnakeLong = {y0, y1, y2, y3};

reg Eshift;

initial begin
```

er was successful. 0 errors, 14 warnings  
tion was successful. 0 errors, 291 warnings

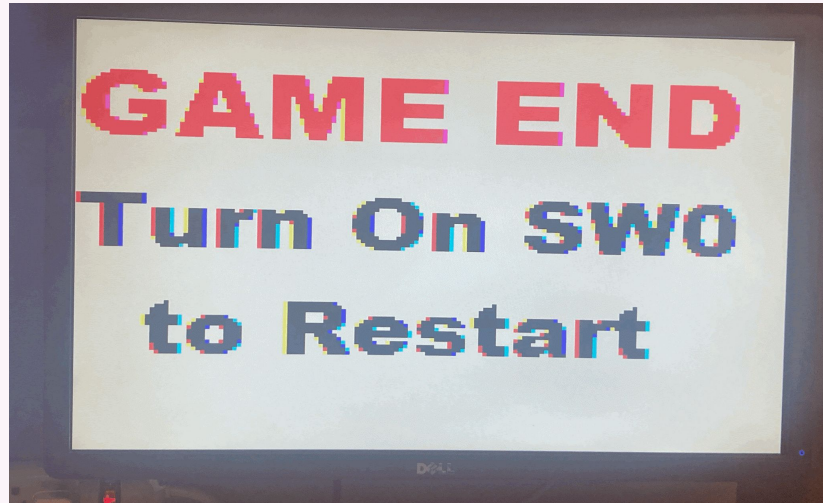
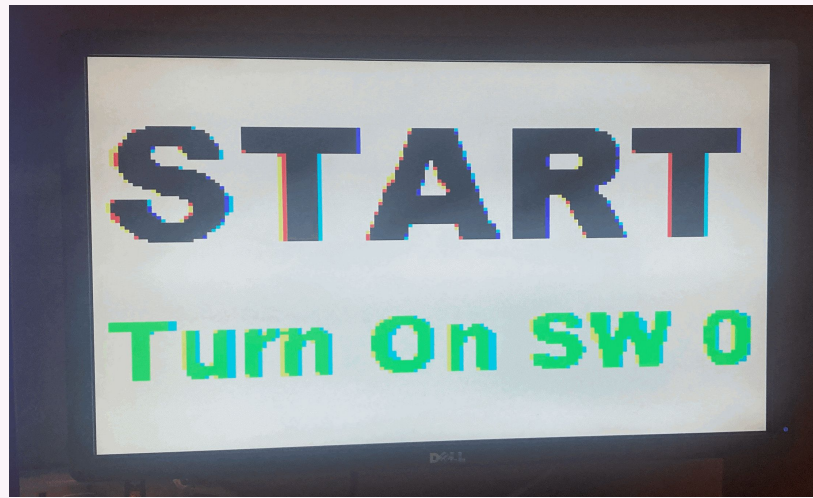


# Future Work

01

Switch to the Start and  
End Screen  
Be able to restart

Note: we were able to draw the screens individually (to the right), but couldn't switch to them at the appropriate time



# Future Work

## 02

### Correct counter when eat apple Grow Snake

Our “grow snake” function is already coded, however it relies on the counter because with each apple it grows one. Once counter gets fixed, the “Grow Snake” will work too.

## 03

### Implement keyboard & speaker

The code should accept code from the keyboard, instead of the keys. Also should make noise when it eats an apple and when it dies.



# Final Work Distribution

## Karen

- Snake length dependant on currentLength
- Snake movement with shift register
- End Game when snake hits itself

## Tyra

- Apple Generation
- Apple eating
- End screen
- End game with hit edge of screen





# THE END!

**Additional Miscellaneous &  
Process Work Beyond this Point**



