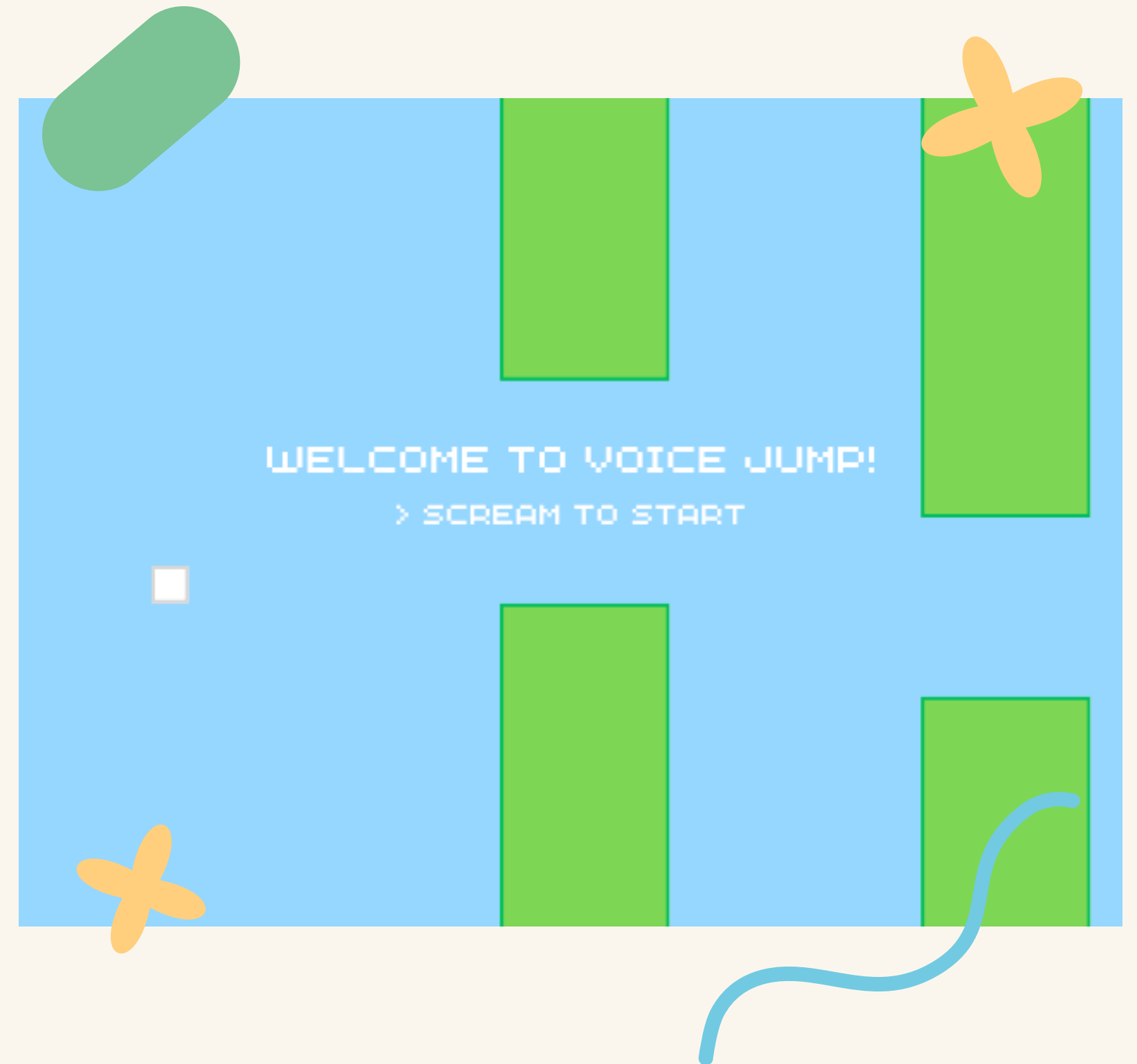
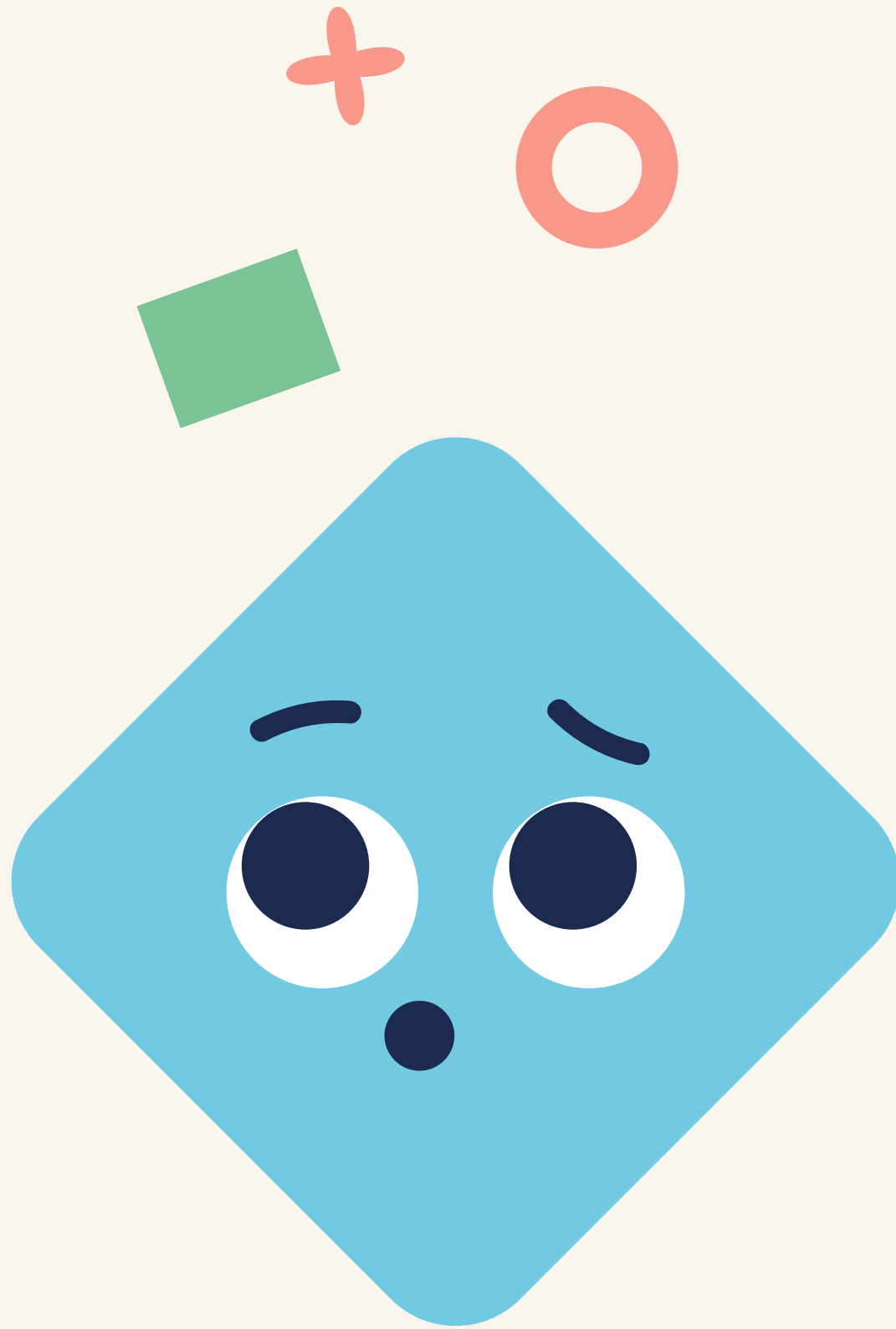


Let's Play!

# Voice Jump!

Georgia Zhang and Jay Lin





# How to Play?

- Flappy bird inspired video game
- Player movement controlled by audio amplitude input
- Collision detection with pillars and screen boundaries
- Score tracking implemented in BCD, displayed on 7-segment HEX
- Start/reset buttons with start screen and game restart

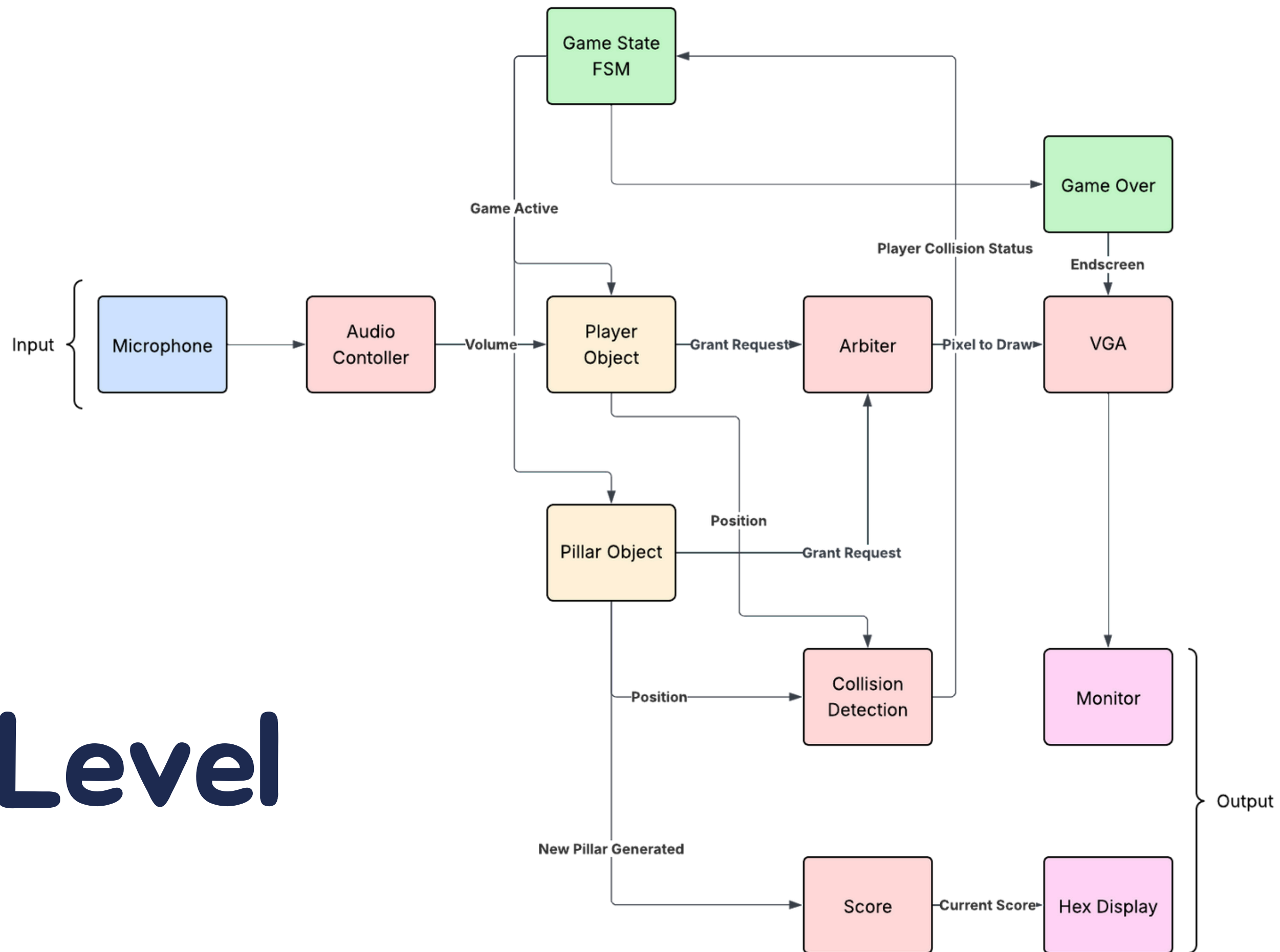


# Block Diagrams

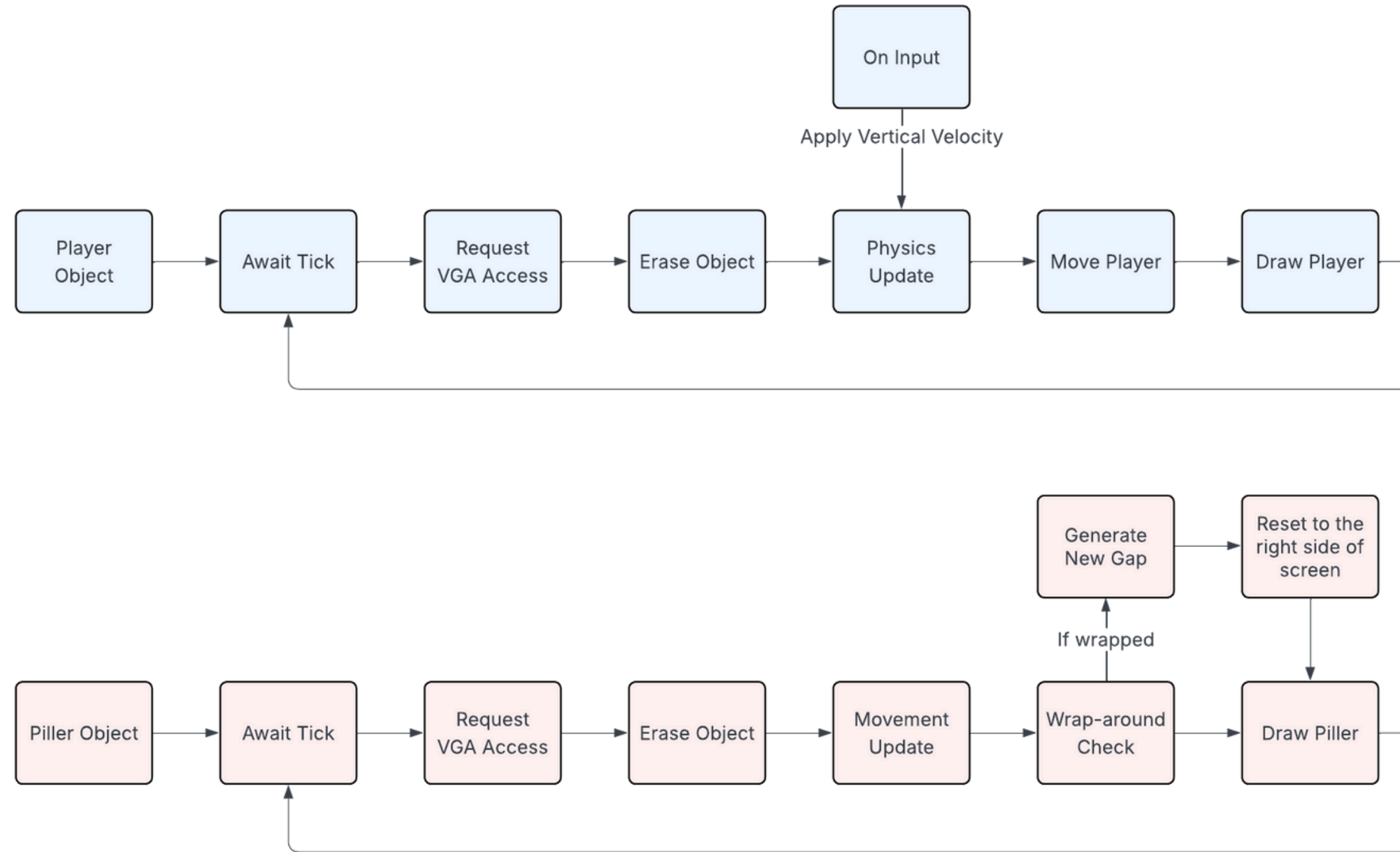


**How our code works**

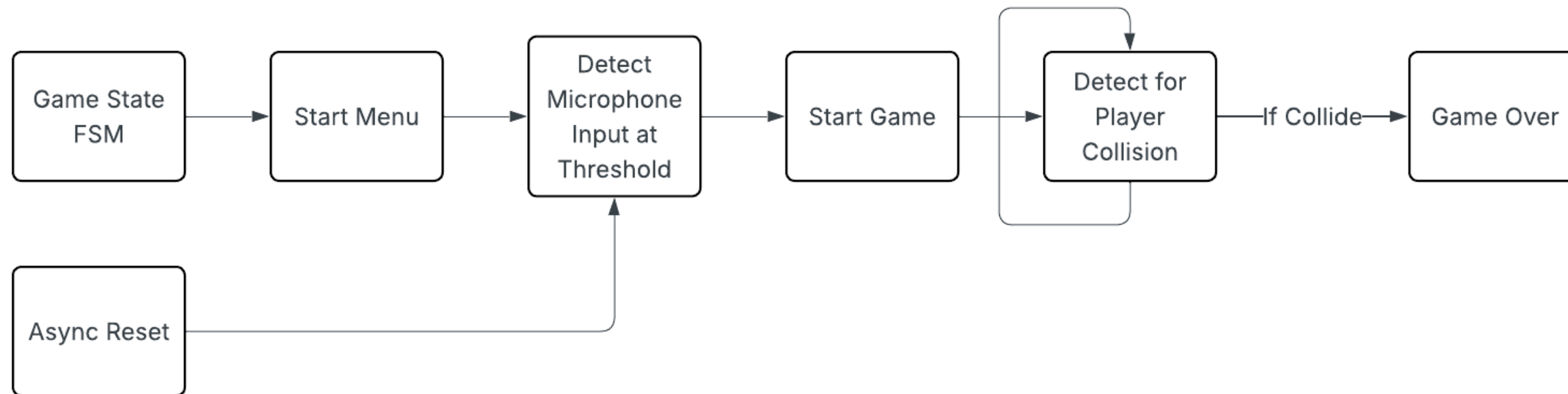
# High Level

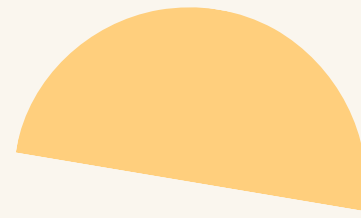


# Player and Pillar Block Diagram



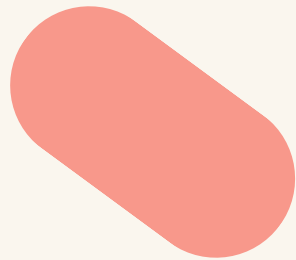
# Game State FSM





# Project Demo

**Let's play!**

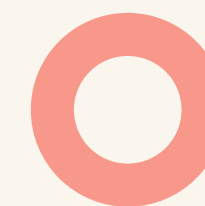
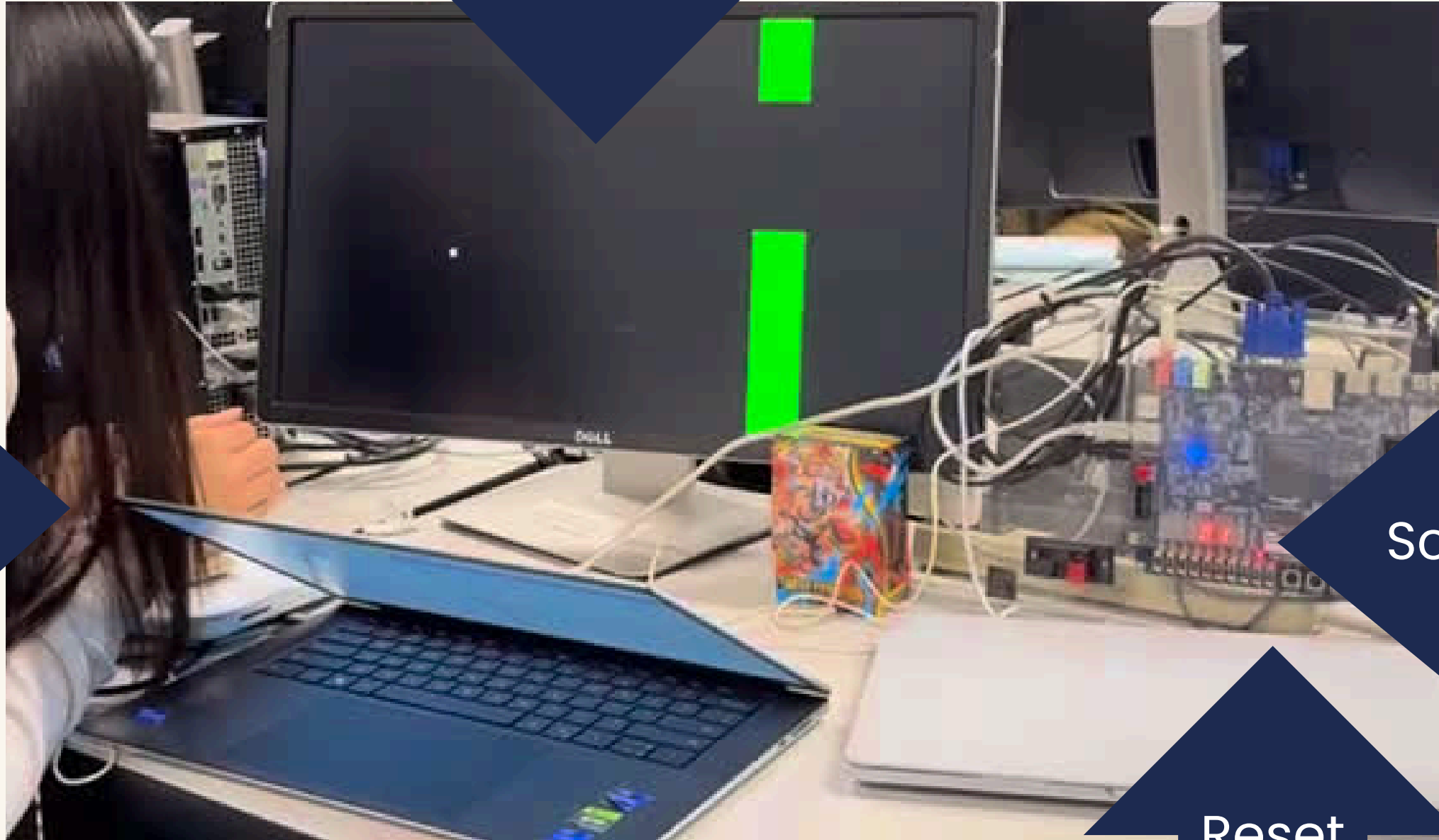


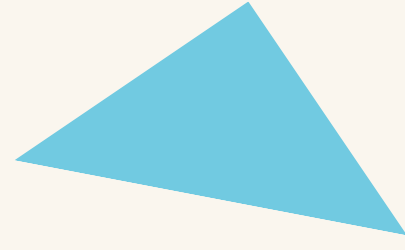
VGA  
Display

Microphone input

Score counter

Reset  
button





# Bugs and Fixes

**Challenges we faced**



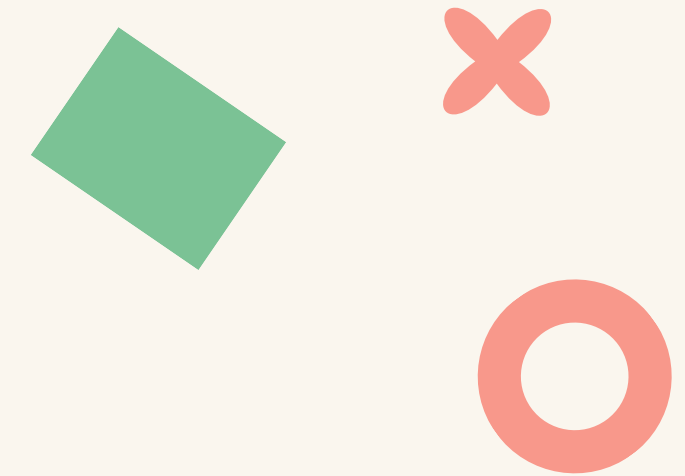
# Game Logic

- Pillar gap RNG not working
- Resolved by cycling through an array of pre-determined gap position

```
next_index = (gap_index == 3'd4) ? 3'd0 : gap_index + 1'b1;  
gap_index <= next_index;  
  
// Update Gap Position  
new_center = MIN_GAP_CENTER + (next_index * STEP_SIZE);  
current_gapTop <= new_center - (FIXED_GAP_SIZE >> 1);  
current_gapBottom <= new_center + (FIXED_GAP_SIZE >> 1);
```

- Score being incorrectly counted
  - Score counter increments each time pillar position resets in pillar FSM

```
H: begin  
    req=1;  
    if (X == 0) begin Lx=1; scored = 1; end  
    LXC=1; LYC=1; |  
end
```



# VGA

- Streaking pillar drawing
  - Switched from drawing row by row to column by column

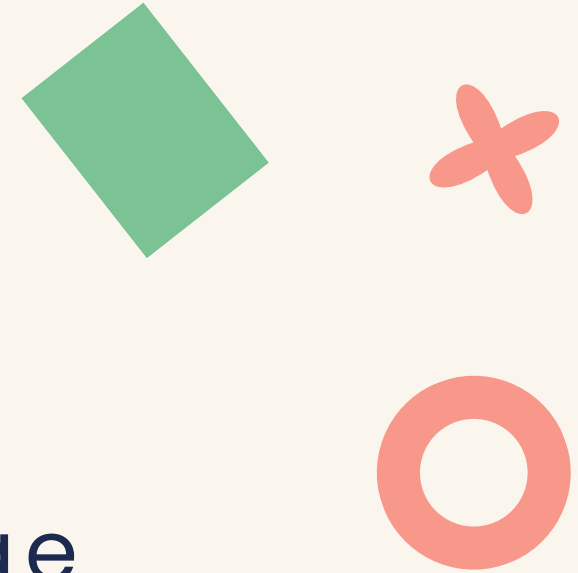
```
// Draw leftmost column (XC = 0, increment YC) - but only if X > 59
I: begin
    req=1;
    if (X > XDIM-1 && !collision) begin // Only draw if pillar is fully on screen
        Eyc=1;
        write=1;
    end else begin
        Eyc=1; // Still increment YC to complete the loop
    end
end
```

- Background image remaining on screen between game states
  - Create a background clearing module

```
case (y_Q)
    IDLE: begin
        Lx = 1'b1; Ly = 1'b1; Lxc = 1'b1; Lyc = 1'b1;
        ready = 1'b1;
    end

    DRAW_BLACK_X: begin
        req = 1'b1;
        write = 1'b1;
        drawingBlack = 1'b1;
        Exc = 1'b1;
    end

    DRAW_BLACK_Y: begin
        req = 1'b1;
        drawingBlack = 1'b1;
        Lxc = 1'b1;
        Eyc = 1'b1;
    end
end
```





# Future Work



**Improvements to make**

# Game State

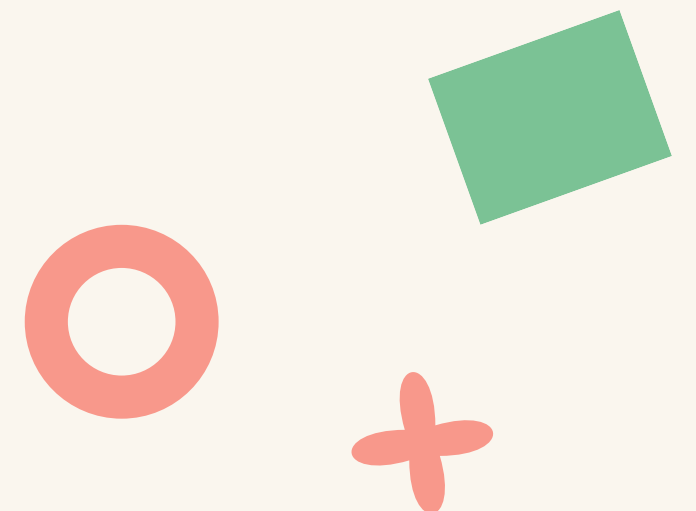
- Extend the game state FSM to include a game over state
- Add additional difficulty levels, changing pillar speed and gap width

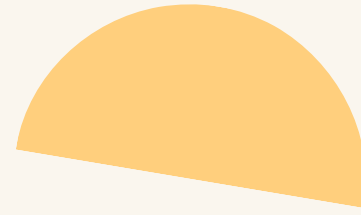
# Controls

- Use Fourier transforms to use pitch control for player movement

# Display

- Use more detailed game graphics
  - Bird character for player
- Have multiple pillars on screen at once
- Display score on the VGA





# Work Distribution

**Who built what**



# Georgia

# Jay

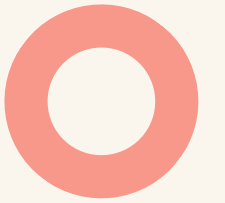


---

## Milestone 1

- Pillar module
- VGA drawing arbiter

- Player module



---

## Milestone 2

- Hex score display
- Collision detection module
- ModelSim simulations

- Audio input for player movement control
- Gap randomizer module
- Score counter

---

## Final Presentation

- Presentation slides
- Game state FSM
- Background reset module

- Block Diagrams
- Gap height randomizer

Thank You  
For Playing!



Any  
questions  
?