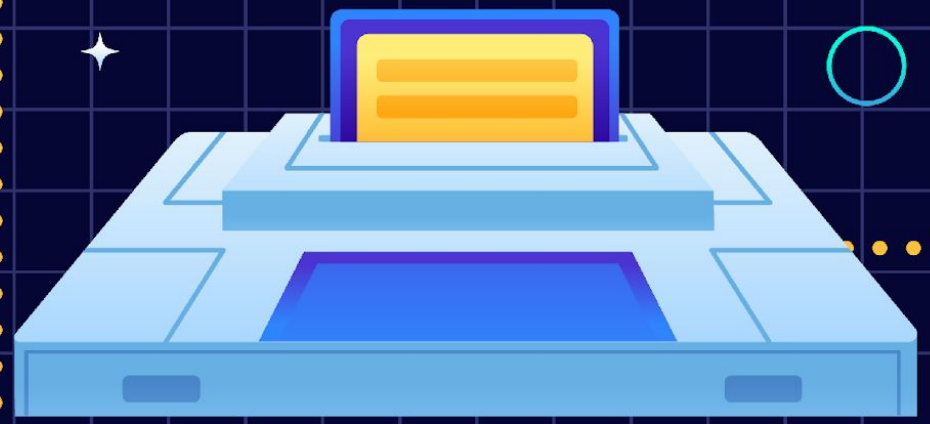
PAC-MAN

EGR 3093 Final Project
Dorian, Kyle, and Mikel

EGR 3093



How To Play

1. Select a 3-character username with the joystick (0-9, A-F).
2. Control Pac-Man with left/right/up/down joystick.
3. Navigate the maze and attempt to collect all the dots.
4. Each time a ghost catches Pac-Man, he loses a life.
5. Lose 3 lives and it’s GAME OVER!

Introduction

- Binary numbers (1s and 0s) are the basis for all **digital electronics**.
- Designers can utilize **logic gates** to form decision-making circuits.
- Basic operations include **AND**, **NOT**, & **OR**.
- **State Machines** prescribe certain outputs based on a range of input conditions.
- User **inputs** (4-way joystick) and internal signals (processor clock) drive **outputs** (LEDs and score/HDMI displays).
- **Memory** allows the system to read game data (map/player).

Technical Details

- All the driver modules for this project were written in **VHSIC Hardware Description Language** (VHDL).
- Programming was completed in the **Vivado** integrated development environment.
- Team collaboration was aided by a cloned **GitHub** repository.
- Our Xilinx Spartan-7 XC7S50-CSGA324 **Field Programmable Gate Array** (FPGA) interprets VHDL instructions as a physical circuit.

01

Design Overview

Game Design Goals

01

3-Character username selection using external joystick

02

Increasing difficulty over time (chase vs. scatter)

03

Score counting on seven segment display

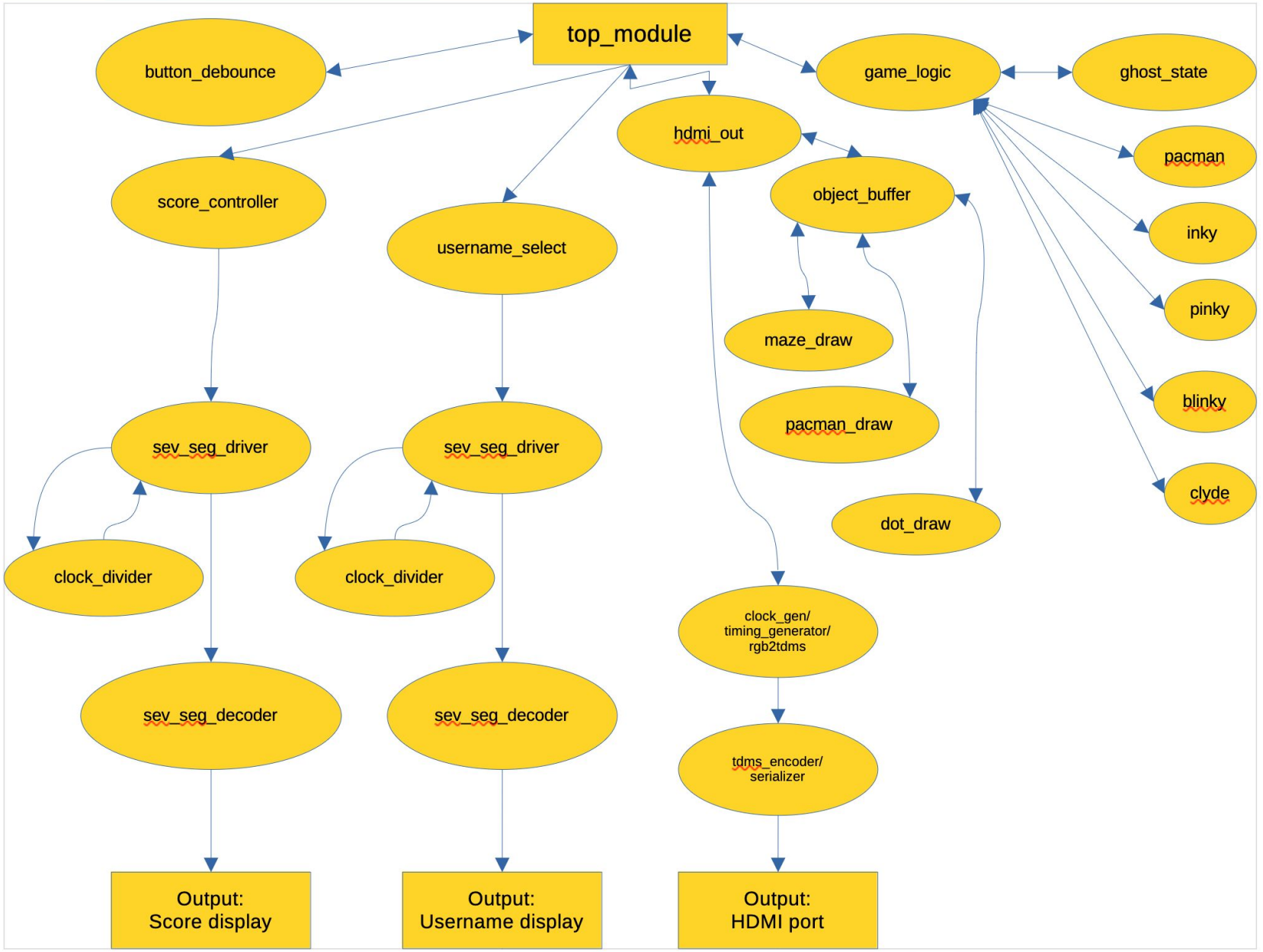
04

Video output on 480p LCD display (VGA-to-HDMI)

05

Game continuation until all 3 lives are depleted.

System Flow Chart



Pac-Man & Ghost ROM

```
"0000111111110000"
"0001111111111000"
"0011111111111100"
"0111111100111001"
"0011111100111001"
"0001111111111001"
"0000011111111111"
"0000011111111111"
"0001111111111001"
"0011111111111101"
"0111111111111001"
"0011111111111001"
"0001111111111000"
"0000111111110000"
```

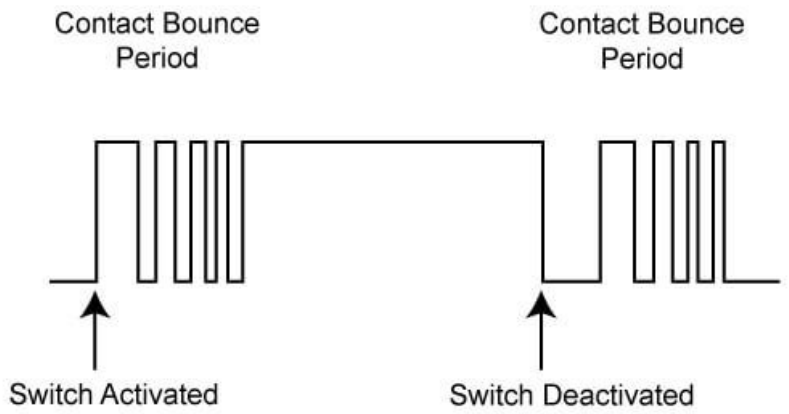
```
"0000001110000000"
"0001111111111000"
"0011111111111100"
"0111111111111100"
"0110001100011001"
"0110001100011001"
"1110001100011111"
"1111111111111111"
"1111111111111111"
"1111111111111111"
"1111111111111111"
"1111111111111111"
"1100110011001101"
"1000010010000001"
```

02

Input/ Output

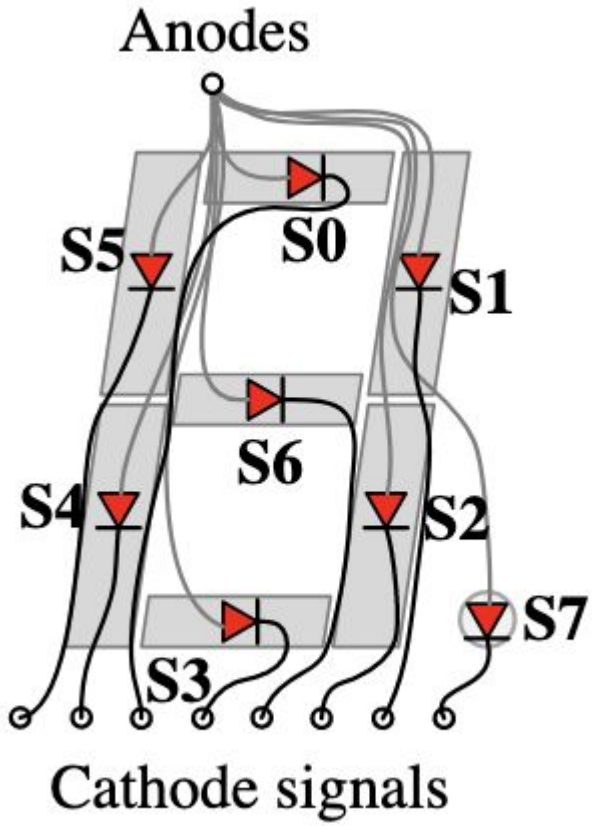
Joystick Input

- Due to the circuit's electrical connection, some inconsistency ("bouncing") of the input signal is to be expected.
- A debouncer module receives raw input and delivers a stable output to the internal logic module.



Seven Segment Display

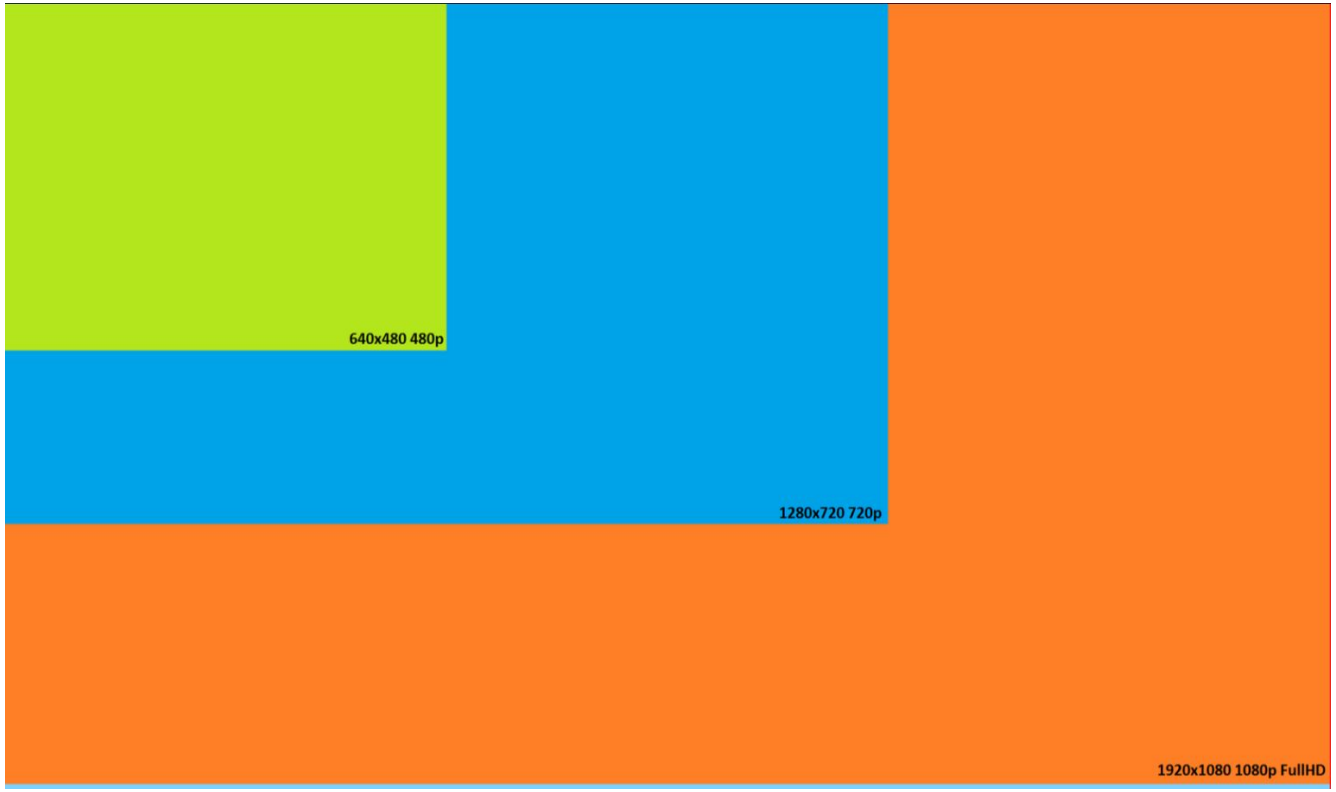
- This display converts a binary input into a user-readable hexadecimal output.
- The anodes and cathodes of each LED segment is controlled by a decoder module.
- The custom username is stored in memory, and the displayed score is incremented every time Pac-Man eats a pellet.



Video Graphics Array (VGA)

Our VGA graphics use these signals:

- A 25MHz pixel clock, which defines when pixels are written on the display.
- A 125MHz clock, which aids in driver internal processing.
- HSYNC determines the start and end of each horizontal line of pixels
- VSYNC determines the start and end of each frame or vertical refresh.
- Red, Green and Blue (RGB) control the current pixel's color.



A challenge our team encountered was implementing a driver capable of providing VGA signals to a VGA-to-HDMI module.

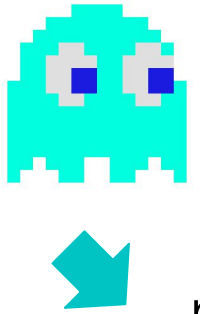
03

Game Logic

Ghost AI

Each Ghost has a different way of catching Pac-Man

Inky

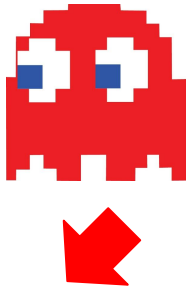


Inky follows a set course around the maze.

Pinky

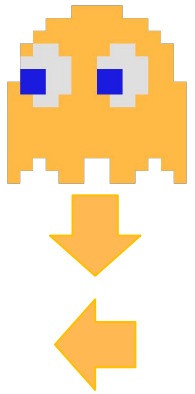
Pinky hunts Pac-Man prioritizing Pac-Man's x coordinates first in a direct manner.

Blinky



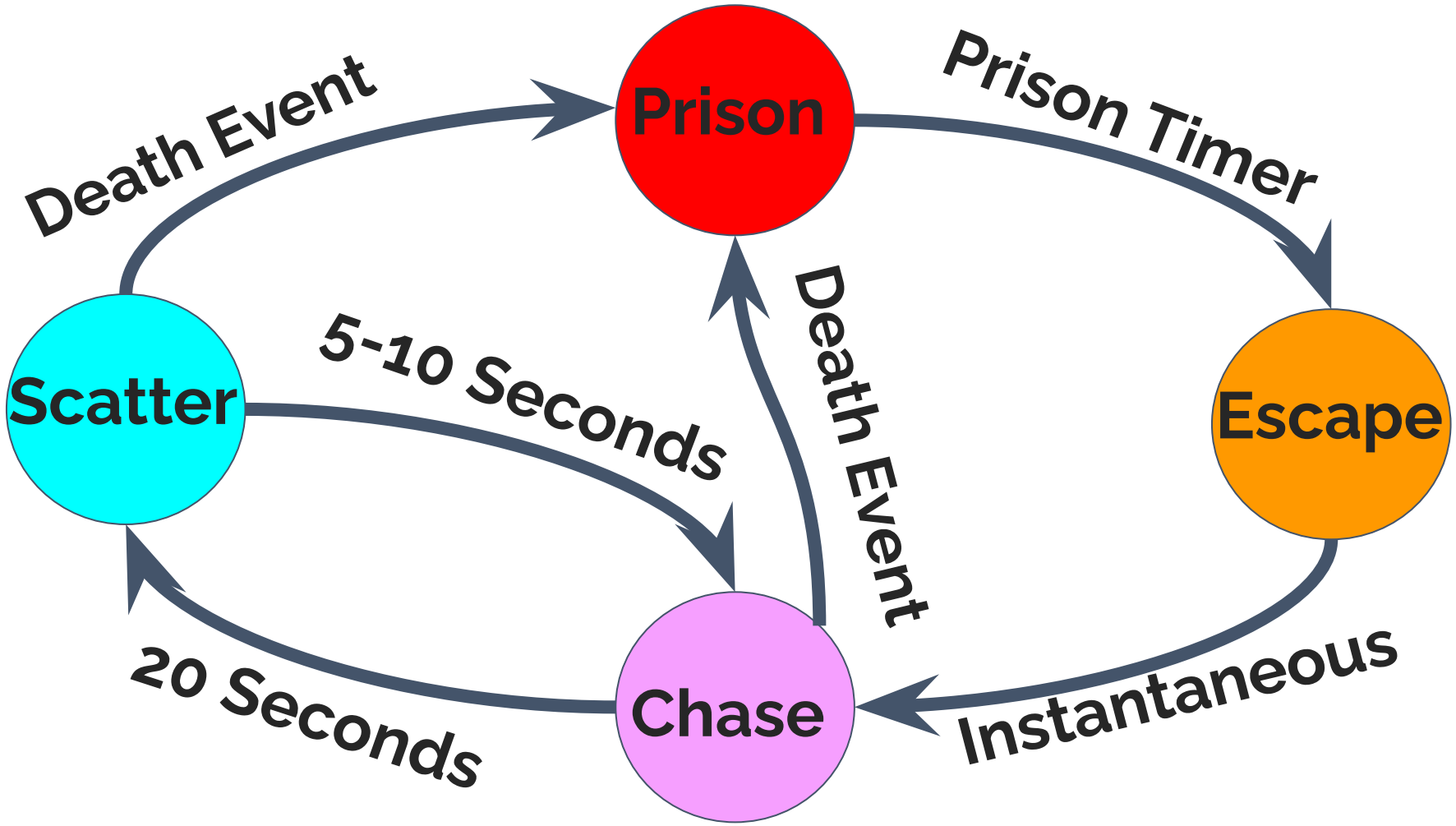
Blinky follows the inverted set course of Inky around the map.

Clyde



Clyde hunts Pac-Mans y coordinates first, essentially being the inverse of Pinky.

Ghost State Machine



Map Logic

- Collisions are detected by comparing (X,Y) coordinates of game objects (walls, players, and pellets).
- Collision events determine game actions such as stoppage of movement, death, score increase.