CS/EE 120B Custom Laboratory Project Report

Cyber Raiders 2024

Zexuan Wang

6/11/2024

Introduction/Project Overview:

In this game, Cyber Raiders 2024 the player controls a fighter jet via a SNES controller. The controller's left and right bumpers will allow the user to move their jet horizontally (Left/Right) while the A button will be used for shooting projects and the down button will be used for a bullet time effect. The objective of this game is to score as many points as possible before your fuel runs out. To score points the user must maneuver their jet and shoot enemy obstacles. For each destroyed obstacle, the user's score will increase by 1. The player will not lose if an object flies past them, they will simply lose a point. The player can also collect more fuel by flying over fueling stations while in the air. In addition, there will be bullet time for players to use every 10 seconds, and will last 5 seconds. While in bullet time, the player slows down the game by 50% while being able to move and shoot at normal speed.

The fuel, bullet ready, and bullet time will be represented via LED lights. To start the game the player will press start on the controller, before the start is pressed the game will display enemies flying past the play like an old arcade game. After the start is pressed, the game will start and the player can now shoot enemies and play. During playtime, fuel will run out and will be represented to the player via a row of LEDs. When the LEDs go dark the game ends outputting the player score and "press start to restart".

Link to demo video https://youtu.be/a301NQ2I5ck

|
|
|
|
|
|
|
|
|
V

Build Upons:

Hardware:

1. Using DIYmalls 128x128 4-wire SPIST7735S LCD
   -Used to display game, no issues getting the build upon to work, also made custom sprites, and handled collision, and movement.(SUCCESS)

2. Using SNES controller as control inputs
   -Used to control the player, A is to shoot, left and right bumper moves left and right, start is to start/restart, and down is for bullet time. No issues getting this build upon to work.(SUCCESS)

Software:

1. Dynamic game state: Added, bullet time powerup allowing players to press the down button slowing the obstacle movement speed by 50% but the user can move at normal speeds for 5 seconds. In addition, there is a live-time fuel system, which requires the player to collect fuel canisters, or else the game ends due to lack of fuel. The baseline includes the random spawning obstacles and the firing of projectiles from the user. No issues getting this software complexity to work. (SUCCESS)

Basic Functionality/User Interface:

1. This game will be a single-player player game as the user will control a jet to destroy enemy obstacles before their fuel runs out.

2. This jet will be able to move left and right via the SNES controller's left and right bumpers. In addition, the player will be able to fire projectiles to destroy enemy obstacles via the A button. The down button will be used to activate bullet time.

3. Obstacles will spawn and move vertically down the screen. Each destroyed obstacle will award the player one point, while obstacles that do not get destroyed will deduct a point(does not go below zero). After an enemy is shot their sprite will turn different representing that they are shot, they will then fly past the user. During the game, the user will not see their score as a real piolet would not care about their stats during a mission. The user score will be displayed when the game ends.

4. In addition to said obstacles, fuel cans will spawn on the screen allowing the user to refuel their jet continuing the game. If the user runs out of fuel then the game ends up displaying the score. During the game, 5 yellow LEDs will be used to display the user's

current fuel level. To show a decrease in fuel level one LED will turn off every 10 seconds.

5. In addition to said fuel cans, bullet time will be available to the user every 10 seconds. The user can press the down for bullet time slowing the game down by 50% while being able to move and shoot at normal speed for 5 seconds. During the game, an LED will display if bullet time is available. If the LED is on the user may activate the bullet time. On top of this one red LED will display if a bullet is ready to be fired or not.

6. Since I love CyberPunk 2077 hence the name Cyber Raiders 2024, there will be no happy endings. This means there is no win condition. The only way the game ends is if the user runs out of fuel, the player can use their imagination on what happened. After the player runs out of fuel, the score will display.

7. While the game is not currently playing the player can press start to start the game, there will be no pausing since a real pilot can not pause in real life. Before a user presses start, enemies will fly by the player, the player will be able to move but not shoot, this is to display an old arcade game feeling.

8. The user can press reset to restart the game if the score screen is on.

Hardware Components:

- **Computing:**
    1. ATmega328

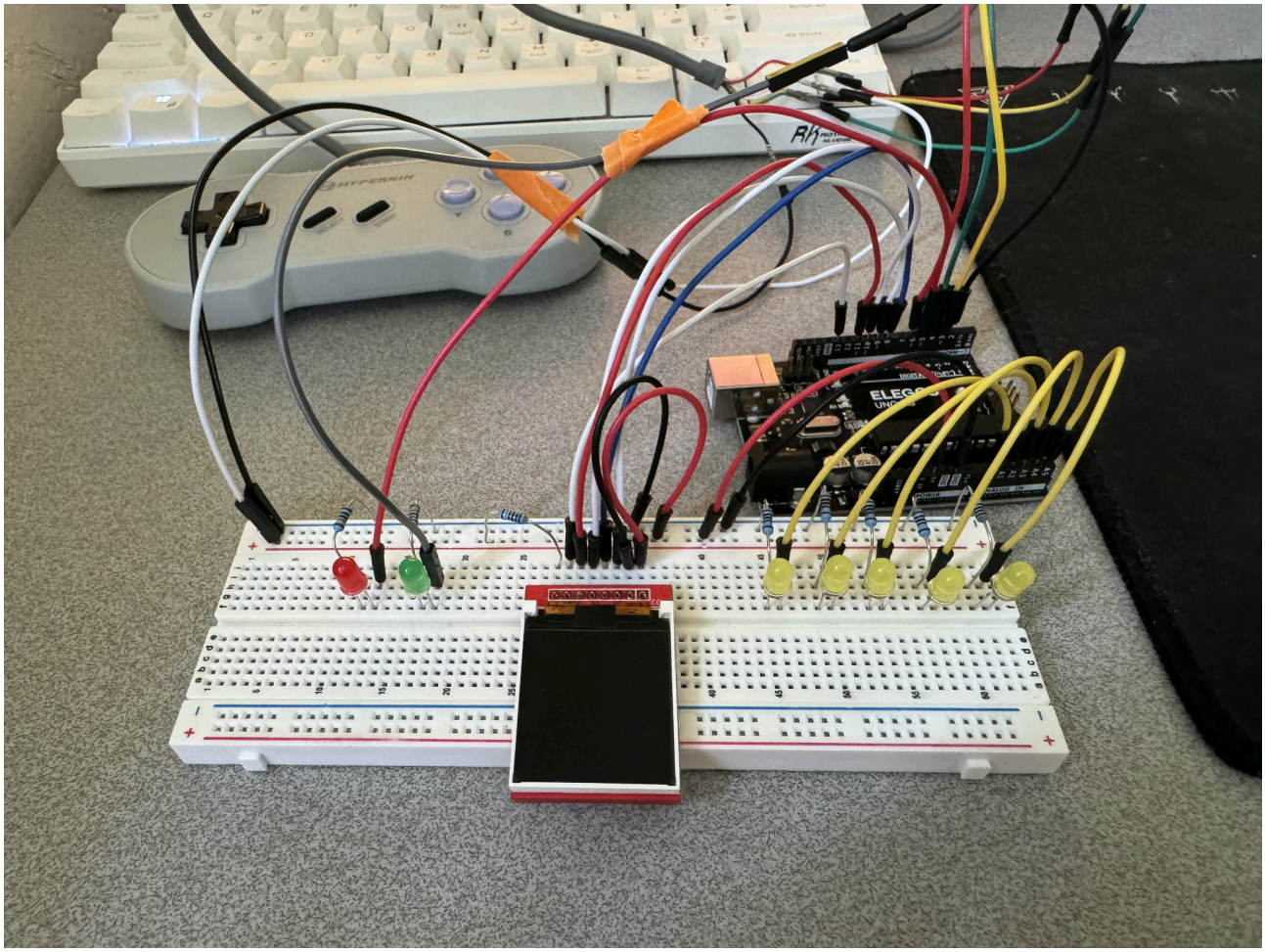
- **Inputs:**
    1. SNES controller
        -Movement
        -Shooting and bullet time
        -Start and Restart the game

- **Outputs:**
    1. DIYmalls 128x128 4-wire SPIST7735S LCD screen
        -Used to display game
        -After the game, it will display the score
       (Score was displayed on Serial Monitor, this was a convenience thing that I got clearance for from Brisk, Marios, and RB)
    2. LED’s
        -During the game, it will display the fuel level
        -Will display bullet and bullet time availability

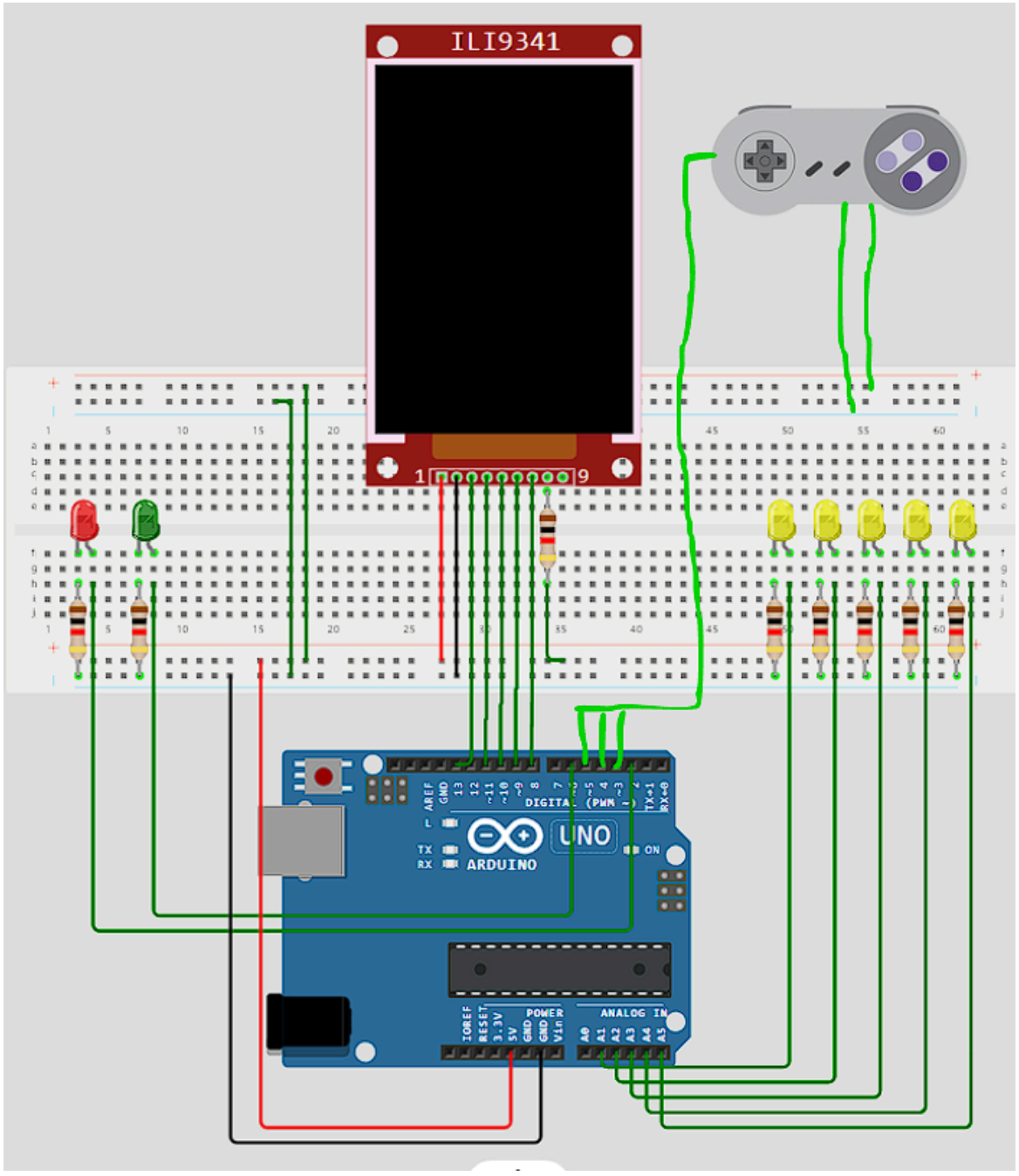
Wiring Diagram:

- **Pin Assignment:**
    1. PORTD[3-5] Will be used for the SNES Controller
    2. PORTB[0-5] Will be used for LCD screen
       -5(SCK),4(MISO),3(MOSI/SDA),2(SS/CS),1(A0),0(Reset)
    3. PORTC[1-5] Will be used for the LED's display fuel level
    4. PORTD[2,6] Will be used for the LED's bullet ready/bullet time

- **Picture:**



- **Diagram:**
  (on the diagram website they did not have the accurate screen but the port assignment states the ports for the SPIST7735S screen)



Additional Libraries:

1. No additional libraries were used besides cstdlib for the access of the rand function. All other libraries/include files were provided by the TAs.

Challenges/Difficulties:

Technical:

1. There were no real challenges or difficulties. The only small mishap was getting the SNES controller to work, but this was fixed after realizing I used PORTD rather than PIND for GetBit.
2. The hardest parts were probably getting the screen and controller to work but the RA's were very very helpful with that.

Bugs/Incomplete features:

3. There were no real bugs or incomplete aspects of this project, everything worked as intended game-wise.

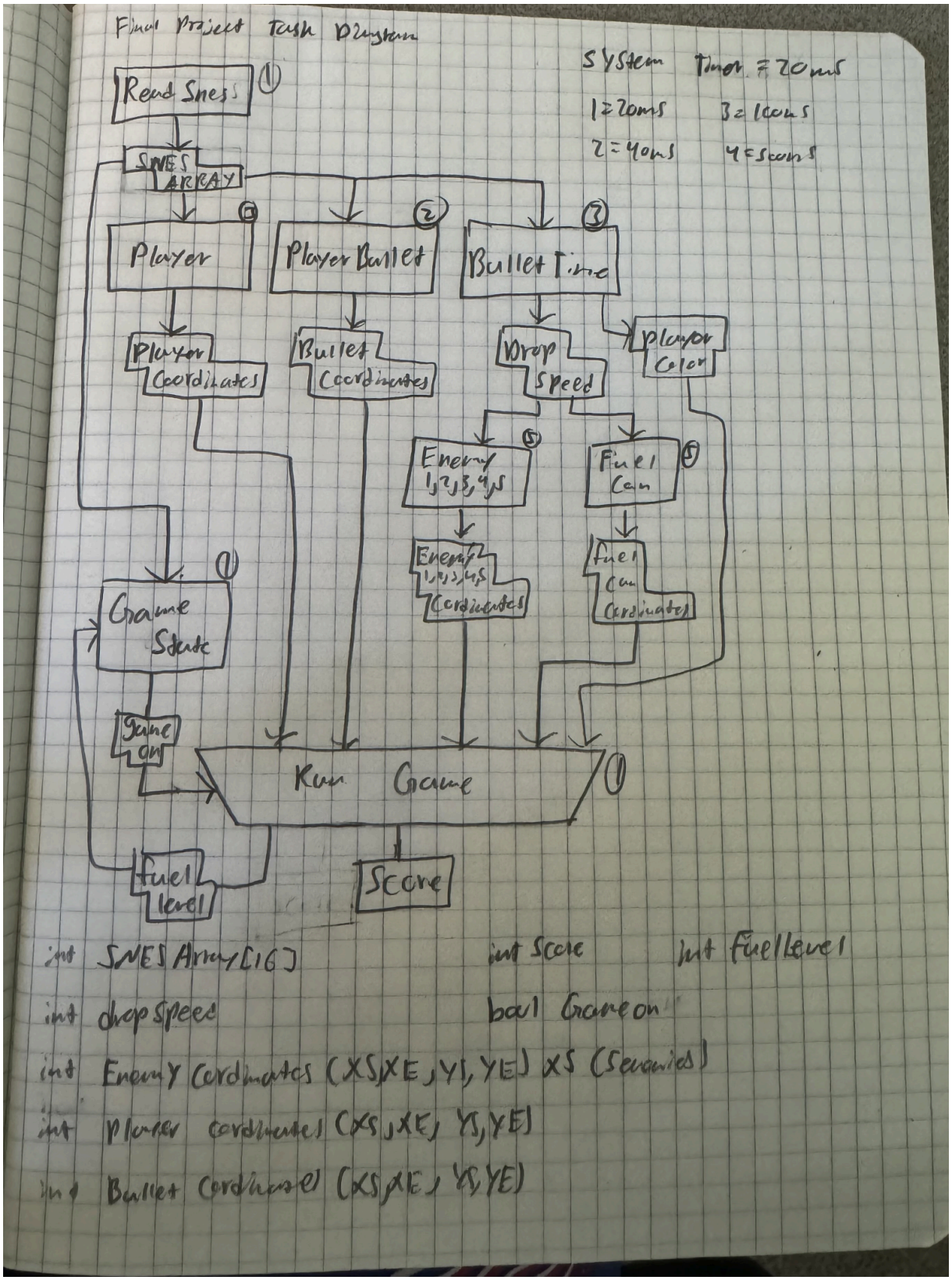
Extra Features:

1. If I had 2-3 more weeks I would have liked to implement sound with the passive buzzer and a hit marker sound with the active buzzer

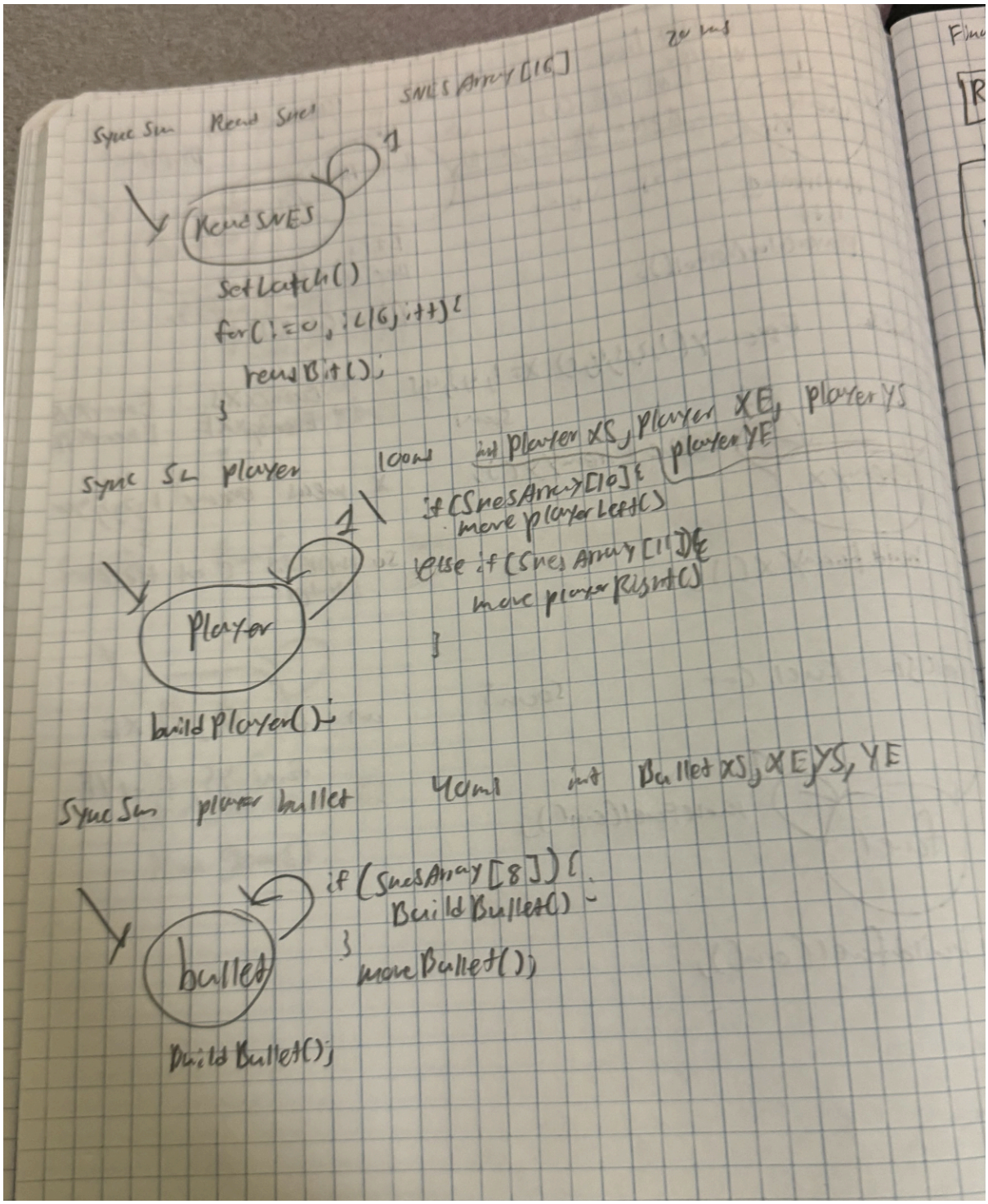
Experience:

1. I had an amazing experience doing this project. I'm happy that I made something that I can put on my resume. In addition, I'm glad that I was able to get this done with the vision staying the same the whole way out. I would say this was really successful and there was no aspects that I disliked.

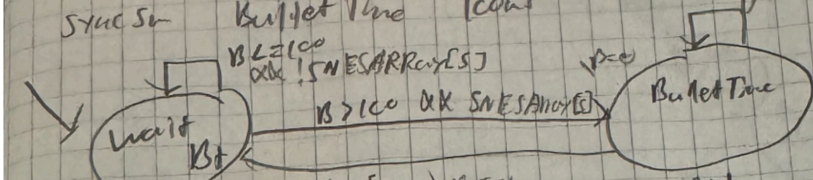
Task Diagram:



SynchSM Diagram:



SyncSM Bullet Time 1cons B0: 50 int DropSpeed



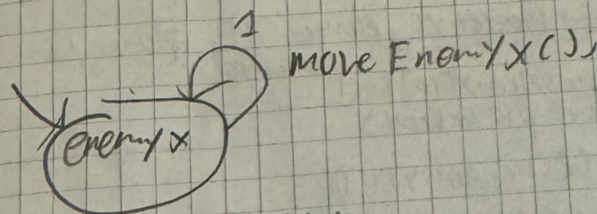
DropSpeed=2;
B++
PlayerColorNormal();

B++;
DropTime=1;
PlayerColorBT();

SyncSM Enemy(1,2,3,4,5) X=1,2,3,4,5 int EnemyXXS EnemyXYS
500ms EnemyXXE EnemyXYE

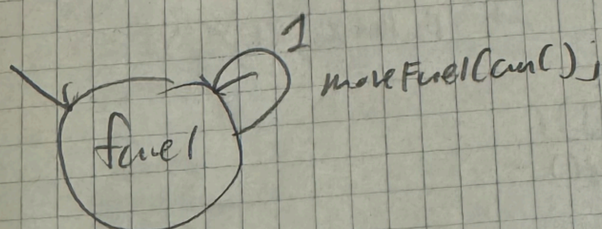
X means enemy 1,2,3,4, or 5

so there is 5 of these states



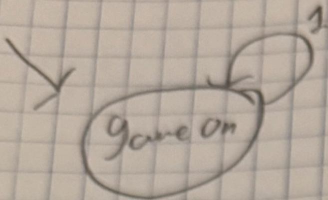
buildEnemyX();

SyncSM Fuel Can 500ms int fuelXS, fuelXE
fuelYS, fuelYE



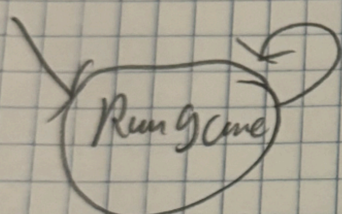
buildfuelCan();

SyncSm gameOn 20 ms bool gameOn




```
if(fuelTimer...){
    if(ShesArray[i]){
        gameOn = 1;
        DisplayOn()
    }
}
```

SyncSm RunGame 20ms int fuelTimer




```
detectCollision()
RunGame()
```