Box Breaker Game Project

EE319H Introduction to Embedded System Honors

By Daniel Yang

Executive Summary

- The game *Box Breaker* is played on a TI microcontroller interfaced with input and output hardware

- The software implementation is done on Keil 5 IDE that includes sprite design, level design, and main file programming

The game is the final project of Introduction to Embedded System Honors (EE319H) that must fulfill the minimum requirements

Hardware Requirements	• Two buttons • Slide potentiometer with ADC • Speaker with DAC • Liquid crystal display (LCD)
Software Requirements	• Two interrupt service routines (ISR) • Three sprites • Score display • Two languages

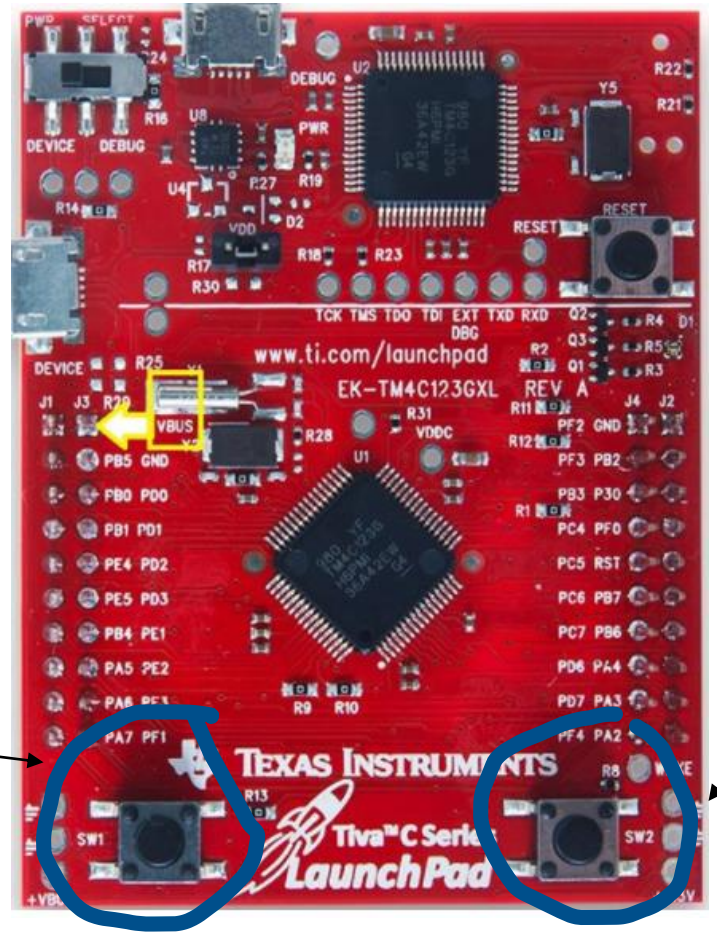
Box Breaker is like *Brick Breaker*, but instead of the ball hitting the brick, the cookie is hitting the box

- English or Spanish
- 3 levels
- 4 lives
- Each box can take 4 hit
- Pink Box contains a powerup

TM4C123GXL Microcontroller is the brain of hardware design

- All Hardware components are connected to the board
- The board receives the data from input hardware
- The software then process that data
- Finally, the data is sent to the output hardware

Two built-in buttons are used to change language and use power-up

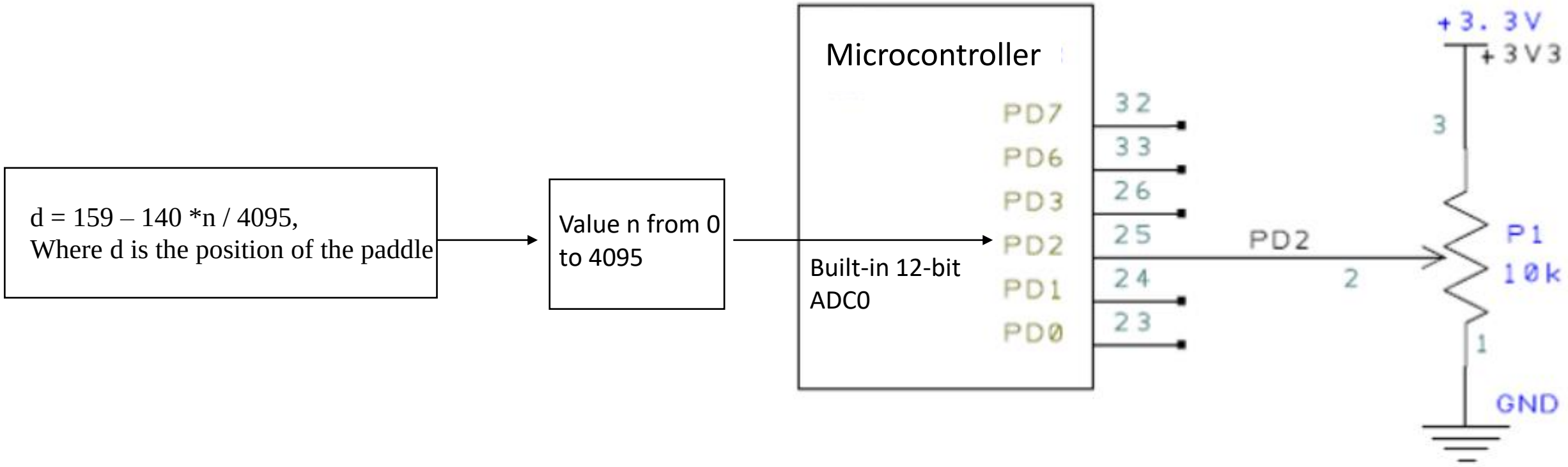


PORT F0

PORT F4

- Edge-triggered on the falling edge

The slide potentiometer is an adjustable 10kΩ resistor that controls the paddle position



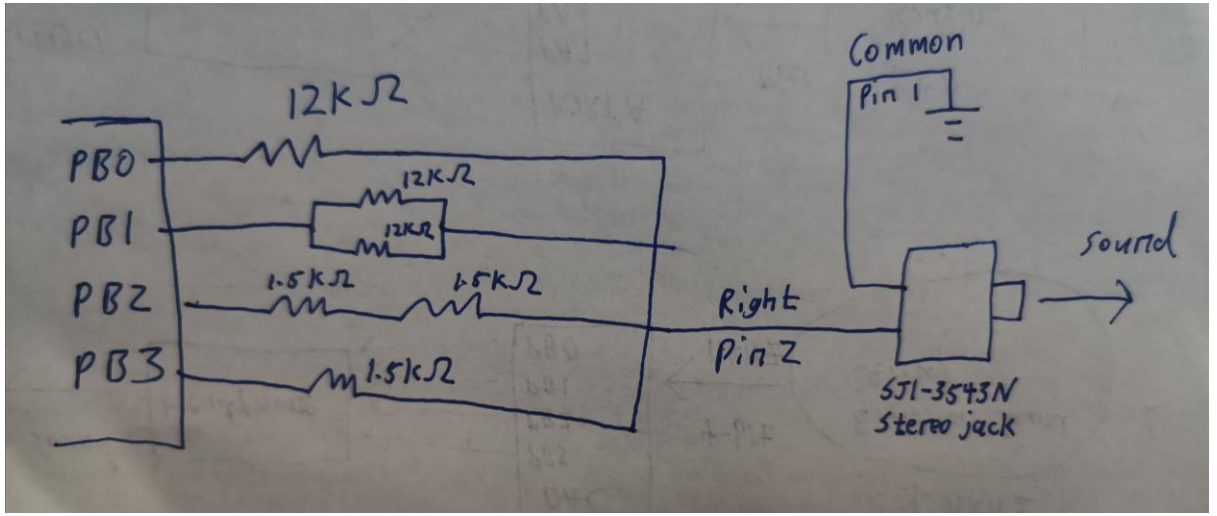
Two output devices are interfaced with the microcontroller, an LCD and a speaker

The 128x160 ST7745 LCD is used to display the game

```
// pin 10 Backlight     +3.3 V
// pin 9  MISO          unconnected
// pin 8  SCK           PA2 (SSI0Clk)
// pin 7  MOSI          PA5 (SSI0Tx)
// pin 6  TFT_CS        PA3 (SSI0Fss)
// pin 5  CARD_CS       unconnected
// pin 4  D/C           PA6 (GPIO)
// pin 3  RESET         PA7 (GPIO)
// pin 2  VCC           +3.3 V
// pin 1  Gnd           ground
```

Program 7.1. Interface connections for the Sitronix ST7735.

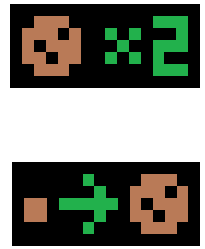
The 4-bit binary-weighted DAC converts PB0-3 value to current for the speaker to play sound



C++ Classes are written to code each sprite

Cookie Class

- Move
- Draw
- Power-up



Box Class

- Lose-hp
- Draw

Paddle Class

- Move
- Draw

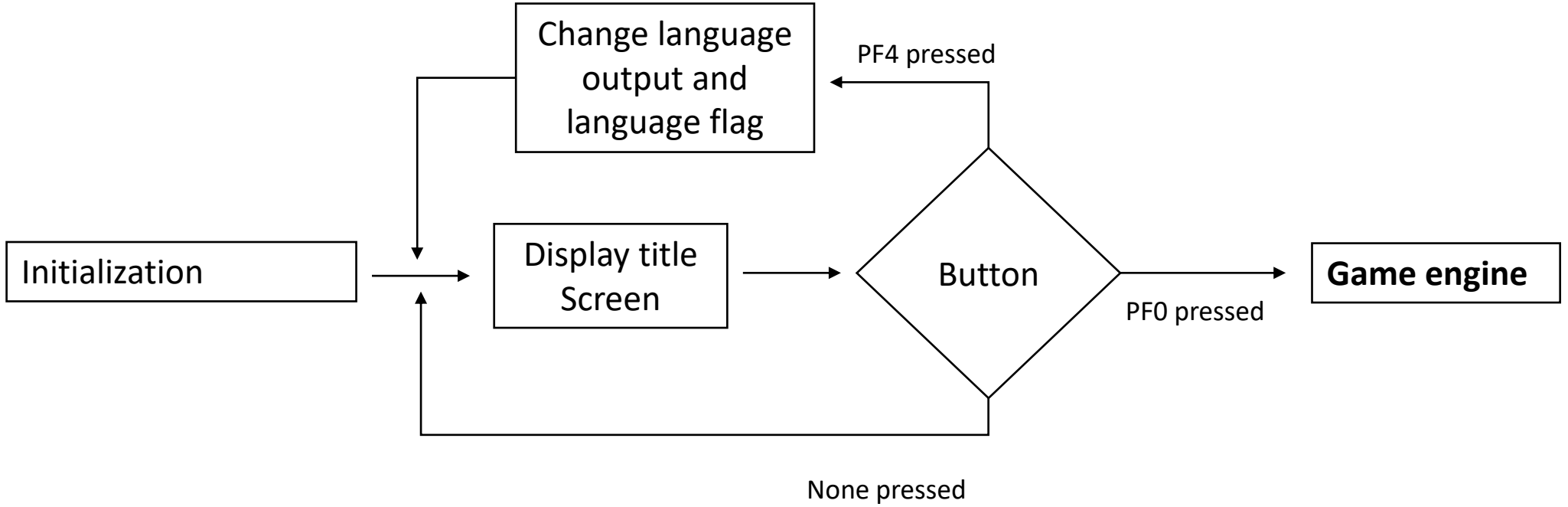
One function is written for each level to initialize a unique layout of boxes

- Each Level can have a maximum of 5 rows and 9 columns of boxes
- Box[5][9] 2D array is declared to represent each box sprite
- The `if statement` check is used to initialize certain Box Objects
- Uninitialized Box Objects will not become sprites

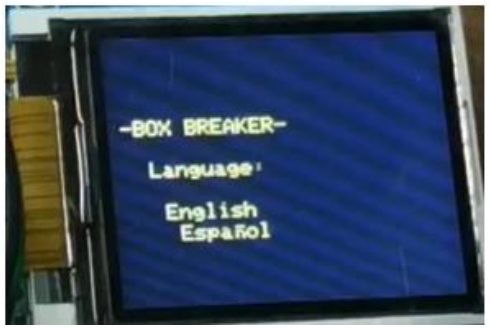
```
for (every row)
    for (every col)
        if (…)
            initialize Box[row][col]
```

The design of each level

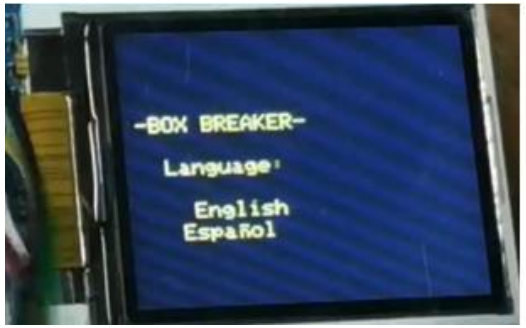
The game begins by prompting user to select a language



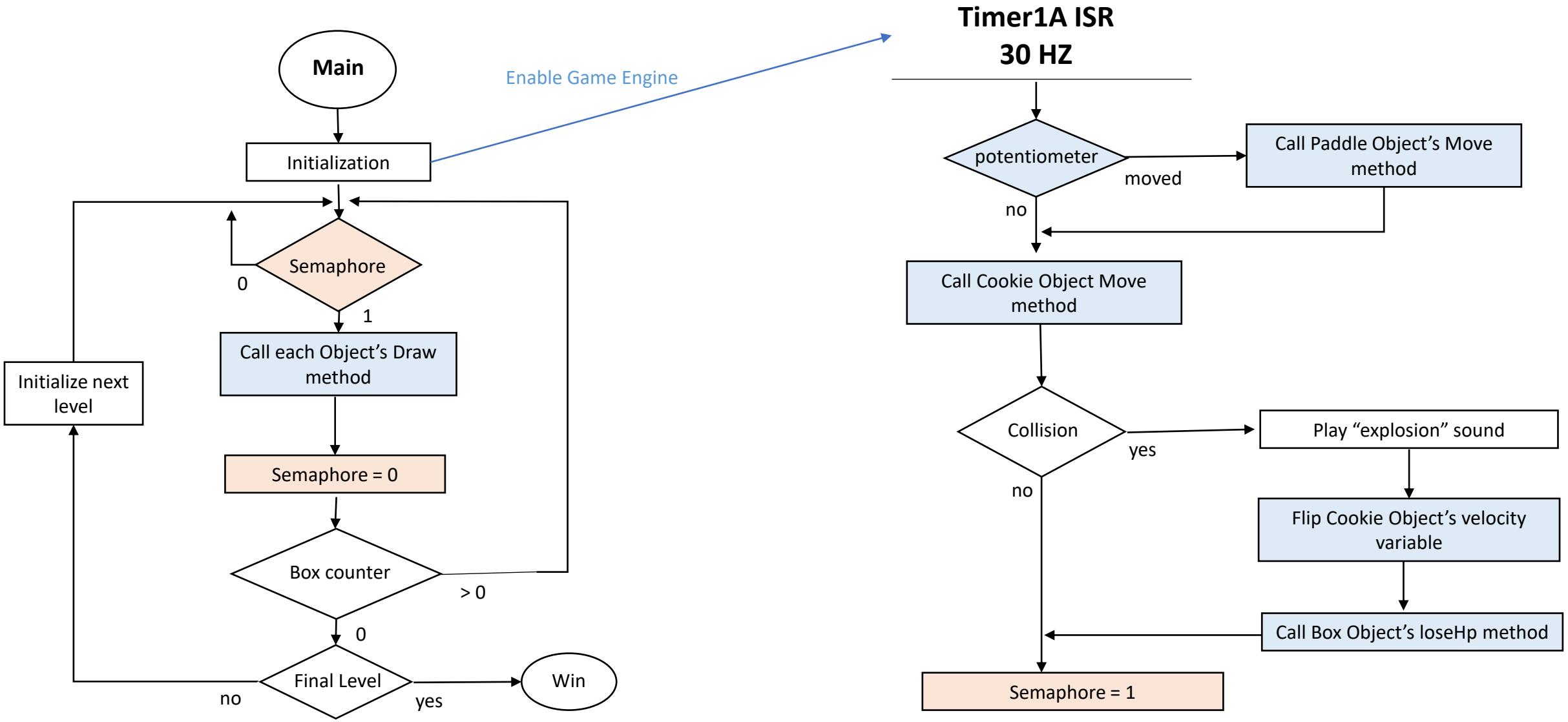
English is selected



Spanish is selected



Game Engine updates each sprite's movement, collision, and display every cycle



Conclusion

- I briefly explained the hardware interface with the microcontroller and software design
- More levels and power-ups could be designed to make the game more interesting
- A difficulty system could also be implemented