


SPACE INVADERS WITH A TWIST

Tuan Phan and Edwin Africano

Our Mission...

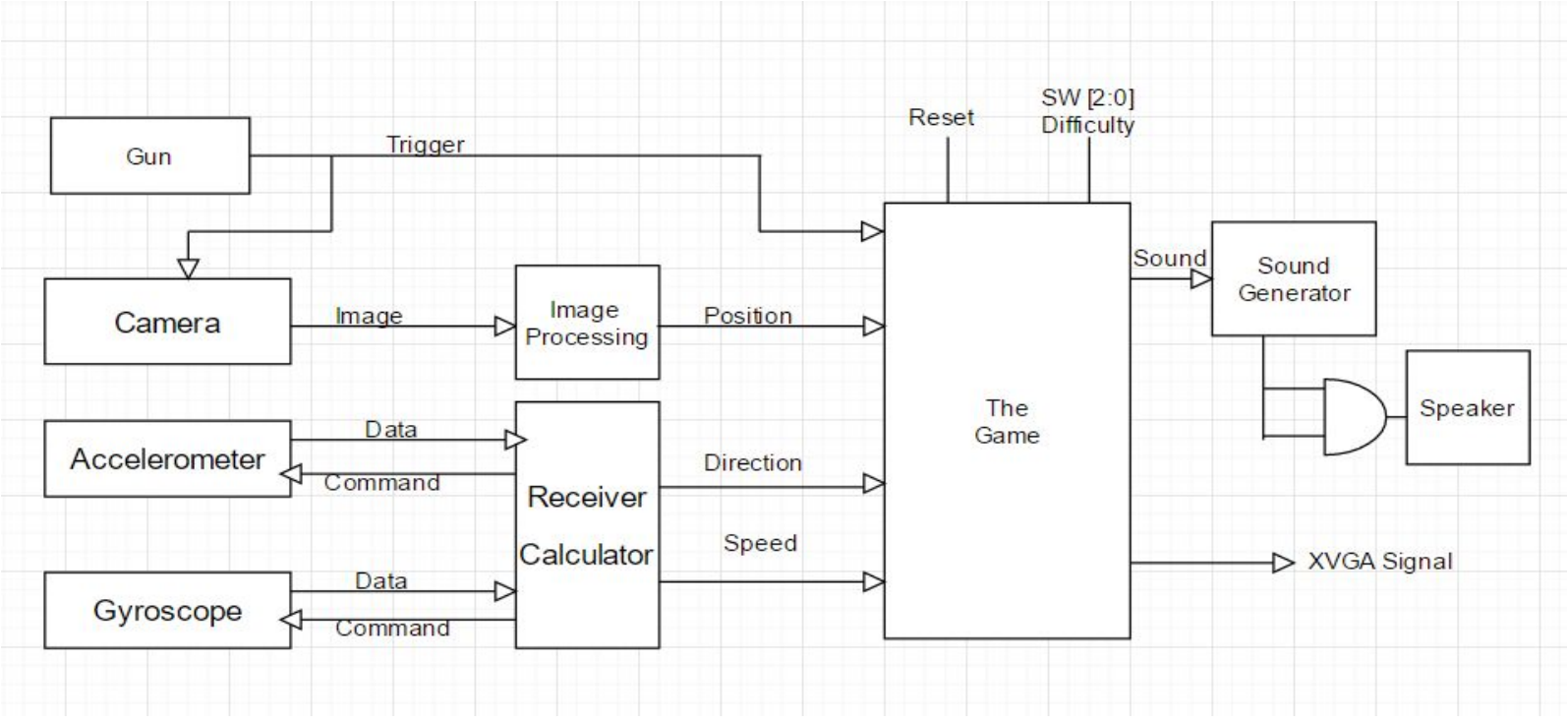
...is to create a game in the spirit of the classic arcade games Space Invaders and Asteroids.

We are adding to the mix a shooting mechanic ala Duck Hunt

We want to make it interesting to control so we are using Gyroscopes and Accelerometers!

Above all we want it to be fun!!!!!!!

Block Diagram



Control

The player's ship will be able to move in the x and y direction of the screen

The player will be able to accelerate the ship

Accelerometer and Gyroscope IMU to accomplish this task

A/G data runs through a module that calculates and feeds direction and speed to the game

Challenge: To be able to do this wirelessly

Next-Gen Zapper

Laser mounted on gun-shaped controller with a trigger

Trigger fires laser on screen and signals the camera to capture an image

Image is then post processed and location of “shot” conveyed to game module

FSM changes state and checks for hit or miss on trigger

Challenge: ensure that only one shot is registered per trigger pull and prevent user abuse by holding trigger

Sound Generator

Produces the sound effects for the game

Sounds depend on the state of the game: a shot fired, collision, a gain in score, etc.

Implementing using an AND gate to drive the external speaker.

Challenge: adding more complex sound effects and 8-bit music!

The Game Overview

Most Complex Module

Generates all video signals for game

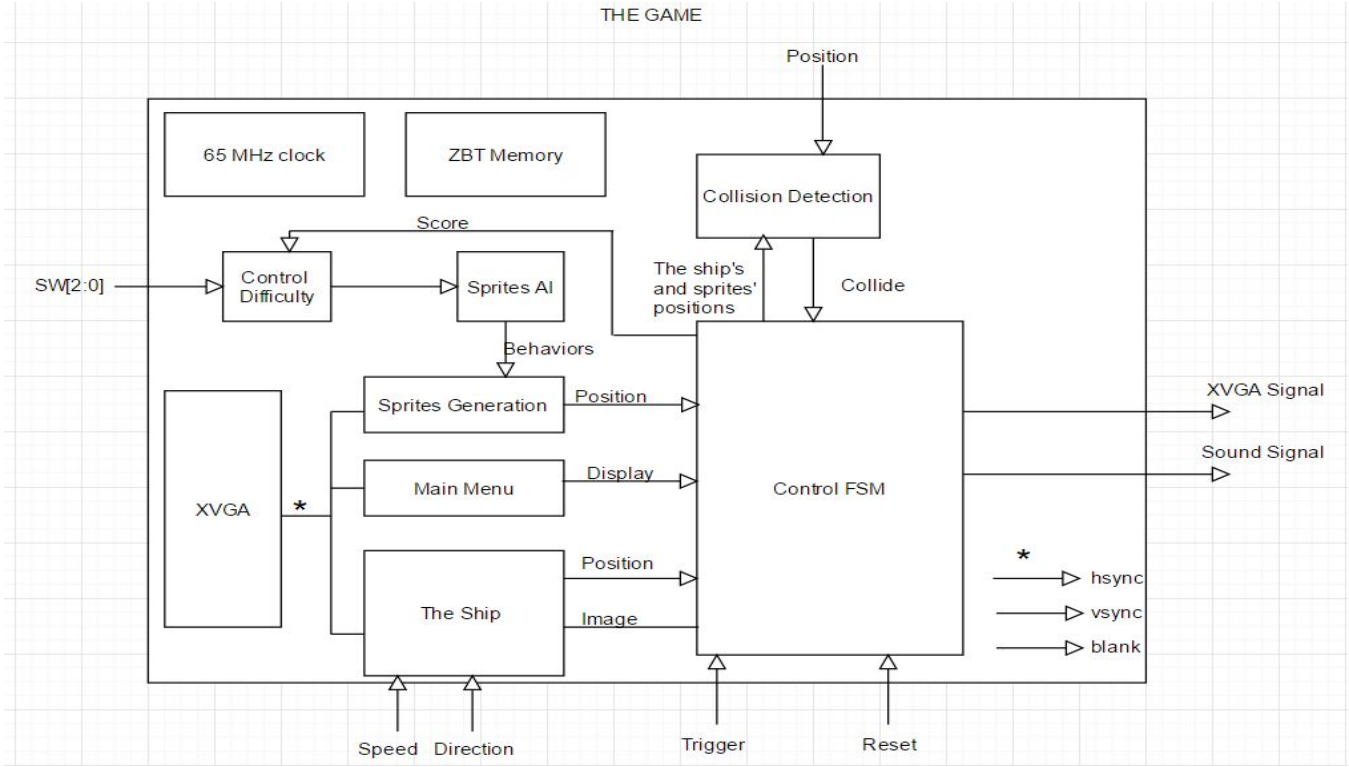
Updates state given user input and current state

Difficulty scales with score/user input

Provides user interface

Manages enemies and collision detection

The Game Block Diagram



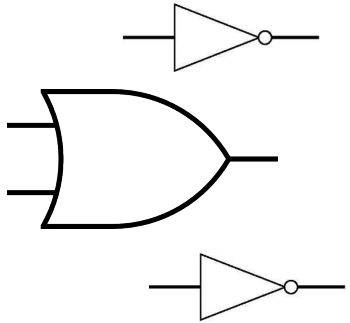
Sprites Generation

Instantiates all the enemies and muxes their display signals depending on the state of the game including collisions (lab 3 style)

Updates their position given by the Enemy Behavior module

Enemy sprites will be digital components: Or, NAND, INV, etc.

Stretch Goal: Enemy Boss



Ship Module

Instantiates the ship model and generates video signal for it

Updates position and speed of ship given user inputs (gyroscopes and accelerometer)

Design is still to be determined

Menu

Module that generates the initial user interface video signal

FSM muxes this signal with the Ship and Enemies signals (lab 3)

This module also updates to create the “Mission Failed” screen on game end

Stretch Goal: Add more functionality to the game that can be added to the game menu, i.e levels, difficulty, etc

Control FSM

The heart of the game

Takes updated video signals from menu, ship, and sprites and generates final output VGA signal

Holds four states: Main Menu, Mission Over, In-game, Shot Fired

In charge of updating the sprites module with hit information, ends game on a collision (ship-enemy, ship-projectile). Collision Detection.

Keeps track of the score

Enemy Behavior and Difficulty

Difficulty is set as a function of input switch and score and passed over to the Enemy Behavior module

Enemy Behavior holds the patterns that the enemies should follow given a difficulty. It also sets the number of enemies present on screen

Challenge: Make AI that responds to the ship position instead of following preset paths with increasing speed

Timeline

-End of Week Nov. 14: The control and shooting mechanism finished. We also able expect to display enemies from the ZBT memories.

-End of Week Nov. 21: The ship being control and integrate peripherals modules with the game. Static path enemies moving and shooting. Basic audio. Collision detection.

-End of Week Nov. 28: Behavioral enemies, wireless control. Music in game

-End of Week Dec. 5: Boss fight. Extra menu functionality

-End of Week Dec. 12: Check off