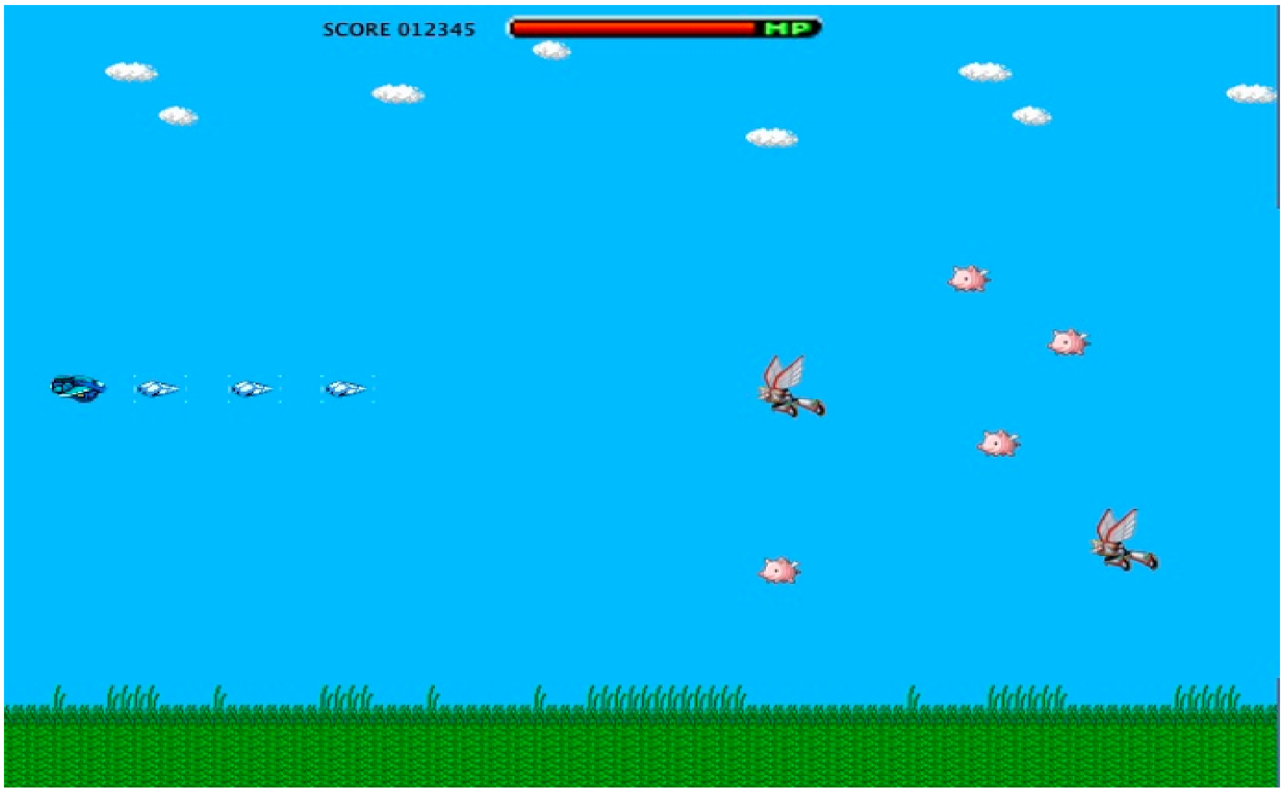
Eskimo @ Farm

Miguel Yanez, Prachi Shukla and Shruti Ramesh
{may2114, ps2829,sr3155}@columbia.edu



Overview:
Eskimo @ Farm is a single player, side scrolling, shoot em up game. It follows the adventures of an eskimo navigating off the complexities of farm life, fighting off different farm animals. Our design document comprises of the following :

1. Design Block Diagram with Descriptions
2. Game Logic as State Diagram
3. Graphics to be used
4. Player Input
5. Handling Audio
6. Milestones

1. **Design Block Diagram**

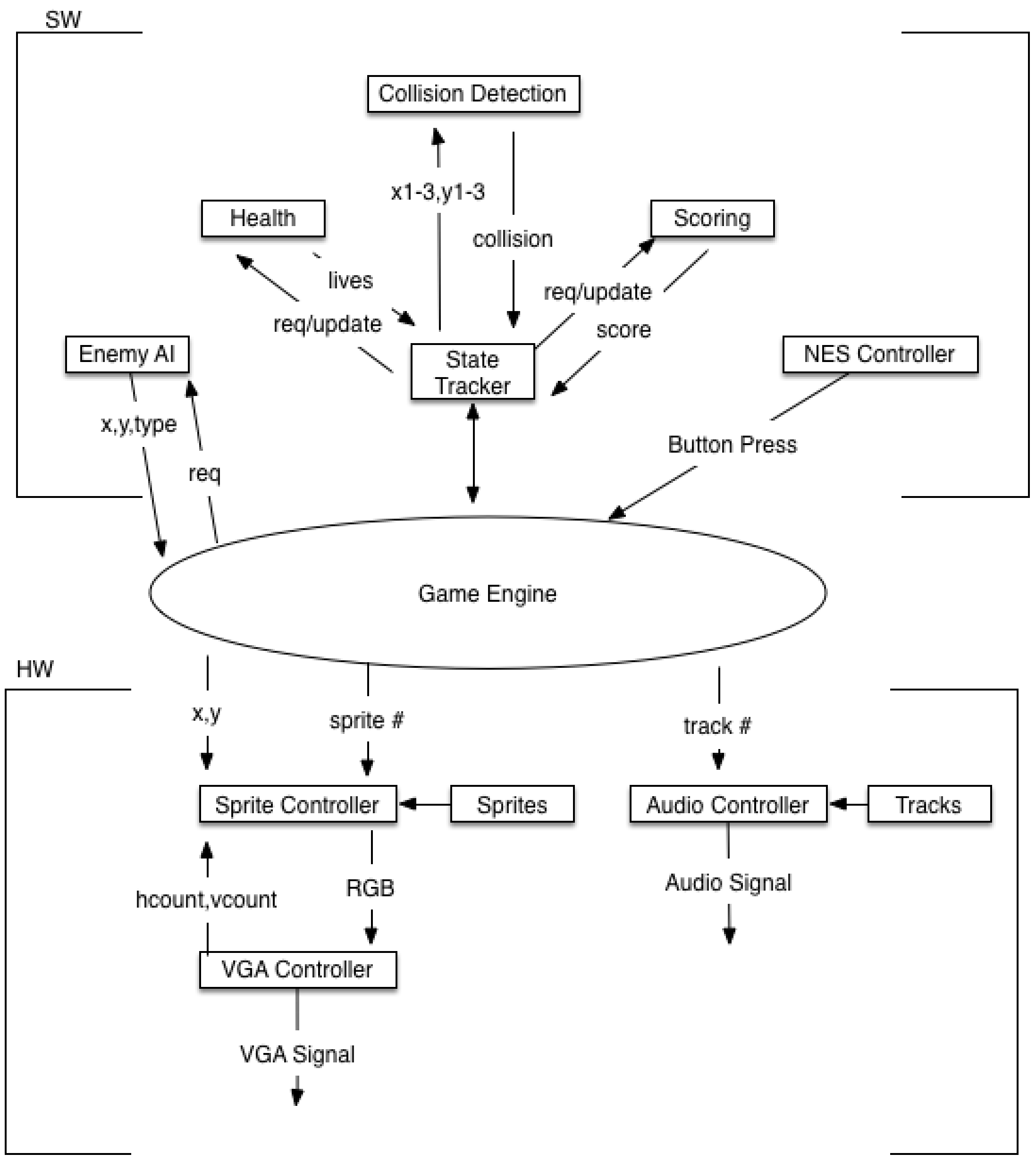


Figure 1: *Overview - Game Engine Components*

Block Descriptions:

SW Blocks:

1. **Game Engine :** The game logic is contained in this module. It takes in inputs from the player, Enemy AI module and passes on the coordinates of these Sprites to the Hardware i.e. Sprite Controller. In addition, it keeps track of the score of the eskimo, health of eskimo and nature of enemy.

2. **Enemy AI :** This module decides the coordinates at which the Enemy sprite needs to be drawn and also specifies the type of the Enemy i.e. Pig or Horse or.. The Game Engine sends a request for the x and y coordinates of the Enemy and this module responds with the values.

3. **State Tracker :** This module keeps a track of the following attributes during the game.It sends periodic updates to the Game Engine.

   1. **Health :** This module keeps a track of the health of the Eskimo. Typically the Eskimo starts out with 3 lives and loses a life everytime he collides with an enemy.The State Tracker either requests the current health of the Eskimo or sends an update to this module when there is an Eskimo - Enemy collision and this module updates the health count accordingly and sends back the updated health to state tracker.

   2. **Collision Detection :** There are two types of collisions possible. One when a bullet shot by eskimo collides with an enemy and the other when a enemy collides with the eskimo. The State Tracker sends the x and y coordinates of the bullet, enemy and eskimo and this module calculates and checks for collisions and sends a notification to the State Tracker in case of collision.

   3. **Scoring :** Every time a bullet shot by the Eskimo collides with an Enemy, the Eskimo scores a point. This module keeps a track of the score. The State Tracker module either requests for the current score or sends an update when there is an Enemy - Bullet collision.

4. **NES Controller :** This module deals with the inputs made by the player and conveys them to the Game Engine.

HW Blocks :

1. **Sprite Controller :** Based on the input from the Game Engine, the sprite controller draws the particular sprite at the coordinates passed. It draws the sprite at the hcount, vcount values passed from VGA controller.

2. **Sprite :** This module holds the sprites primarily for the Enemy [Pig, Horse, Duck, Sheep, Boss], Eskimo, Health, Win, Lose and backdrop.

3. **Audio Controller :** The Game Engine provides the type of track to be played depending on the state of the game. This module picks the track from the Tracks module and plays the audio file.

4. **Tracks :** This module hold the different sound tracks to be played in the game.

5. **VGA Controller:** This module provides the hcount and vcount values to the Sprite controller. It takes in the RGB values from the sprite controller and draws the sprites.

2. Game Logic

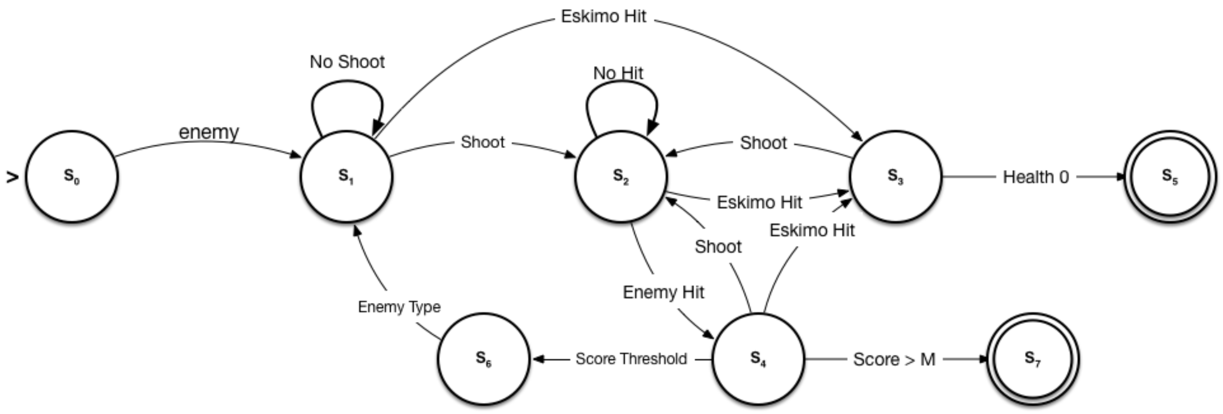


Figure 2: *Overview - Game Logic State Machine*

Attributes of Each State:

- Score
- Player state {co-ordinate, sprite state, health}
- [Enemy state {co-ordinate, type, sprite state}]
- [Bullet {co-ordinate}]
- Audio

State Diagram Table:

Current State	Description	Input	Next State
S_0 (Start State)	The game starts in this state. All the attributes are initialized to 1	Enemy arrives	S_1
S_1	Enemy state is updated	Player shoots	S_2
		Player does not shoot	S_1
S_2	Player is shooting	Enemy is hit	S_4
		Enemy is not hit	S_2
S_3	Eskimo gets hit	Eskimo collides with	S_2 When Health > 0

		enemy	S_5 When Health = 0
S_4	Enemy gets hit	Bullet hits enemy	S_2 Player shoots S_3 Eskimo gets hit S_6 New Enemy enters S_7 Final Enemy arrives
S_5	Lose State	Eskimo dies	-
S_6	New Enemy	New Enemy enters when score increases over threshold value	S_1 Enemy state updated
S_7	Win State	Eskimo wins	-

3. Graphics

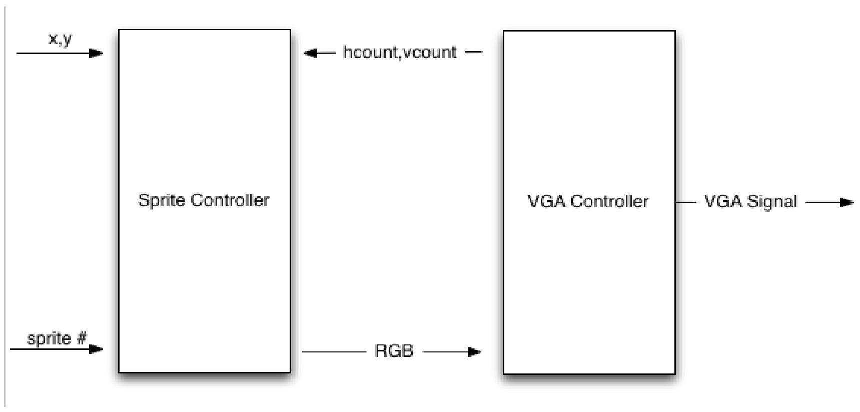


Figure 3: *Sprite Controller & VGA Controller Interaction*

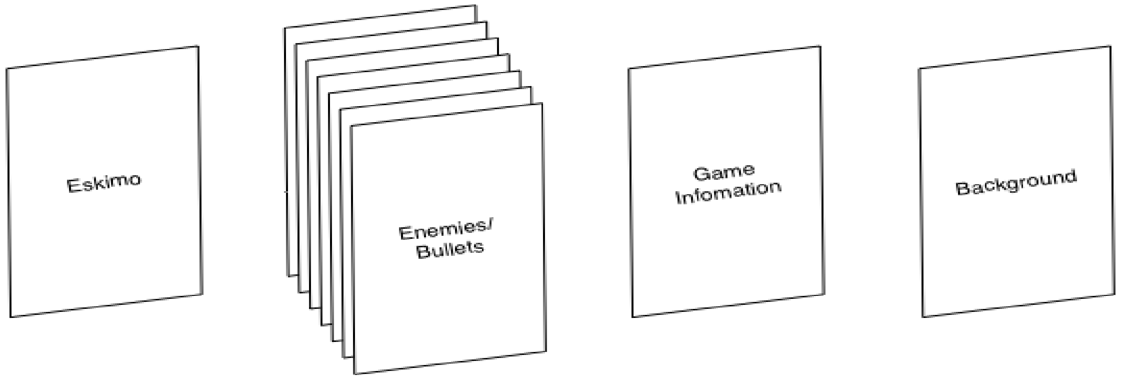


Figure 4: *Sprite Controller - Layers*

- The screen size is 640x480 pixels.
- The VGA Controller will output *hcount* and *vcount*.

- The Sprite Controller will take as input *hcount, vcount* and output the *RGB* value at that location.
- The Sprite Controller will prioritize which *RGB* values to output depending on the layers.
- The Sprite Controller will get input from the Game State Tracking component specifying the positions of the sprites as well as which sprite to output.

4. Player Input

Figure 5: *NES Controller*

Eskimo@Farm will use *libusb* to receive and decode button presses from a USB HID NES Controller.

5. Audio

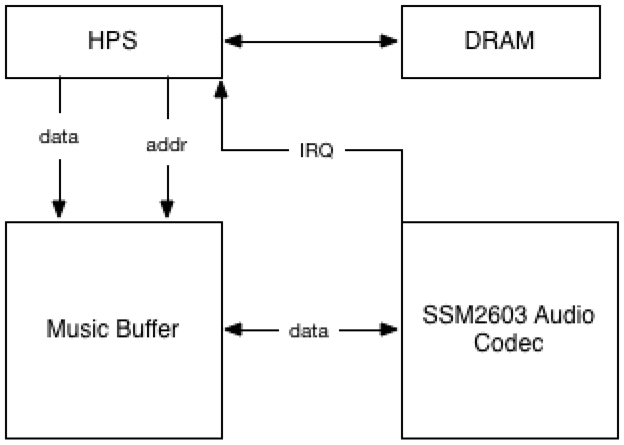


Figure 6: *Audio Overview*

The SoCKit board uses the Analog Devices SSM2603 audio codec. The protocol the audio codec uses for configuration is the Inter-Integrated Circuit (I^2C) protocol. The HPS will read audio files from DRAM and put them in the music buffer. The SSM2603 codec will subsequently retrieve the audio file and play it through the line out port.

6. Milestones

Milestone 1 (April 2nd):

- Choose Sprites and Audio files.
- Basic game implementation using SDL in SW.

Milestone 2 (April 14th):

- Sprite Controller in HW.
- VGA Controller in HW.
- Phase out SDL from game replacing with HW components.

Milestone 3 (April 28th):

- Audio Controller in HW.
- Enemy AI

Final Submission (May 14th):

- Wrap up
- Documentation
- Presentation