6.111 DIGITAL DESIGN
SUPER FPGA BROS

Douglas Albert
Kevin Marengo
Overview

- Project Description
  - Objective
  - Descriptive Overview
- Technical Description
- Project Timeline
- Q & A
Project Description

- Overview
  - Input is provided by user gestures, initially upper body to be later expanded to legs as well
  - Game world is provided by predefined levels and include obstacles and environmental hazards

- Potential Additions
  - Scoring & Time Tracking
  - 2 player competitive “ghost” mode
  - Dynamic representation of player character
High Level Block Diagram

Camera Capture & Gesture Recognition

Game Logic

Video Output

Audio Output

Gesture Information

Game Events
Video Capture & Gesture Recognition

- Camera captures the player’s movements and they are stored in the ZBT
- Each color’s position is averaged to find the center of mass
  - Position used to determine direction and action
  - Velocity of arm patch used to determine walk vs. run
- The actions are output to the Game Logic
Gesture Recognition
Game Cartridge/ Main Logic

- Implement a 2D Game “engine” in hardware
- Abstract away controls and audio output
  - Camera capture and gesture module passes input signals here
  - Event signals triggered in the engine can trigger audio outputs
Game Cartridge/ Main Logic

GAME LOGIC MODULE

Frame Buffer
- pixel_status
- pixel_status1
- write_enable
- blob_row
- blob_column

Blob Processor
- tile_index
- tile_row
tile_column
- blob_index
- write_enable
blob_ram_data

Tile ROM Storage
- tile_index
- tile_row
tile_column
- pixel_data

Level Creator
- current_bg
- bg_image
- level.row
- level.column
- new_item
- blob.loc

Player Character
- player_x
- player.x
(stickman)
- player_status

LevelROM (Multiple)
- level.row
- level.column

Blob Memory
- blob_index
- write_enable
blob_ram_data

Blob FSM
Game Engine

- Level Creator
  - Writes current level layout to Frame Buffer

- Tile ROM
  - 16x16 tiles to create graphics with

- Level ROM
  - 15 x 256 x
  - Levels are made up of tiles

- Blob RAM
  - Holds information about actors on screen: enemies, items, etc.

- FSM & Processor
  - Collision detection & enemy movement behavior
In-Game UI Mockup
Graphics

- Base entity is 16x16
  - Mario is 16x16
  - Big Mario 16x32
- Store sprites in 16x16 chunks
- Use a framebuffer for glitchless output
The theme music is loaded into the FPGA FLASH memory:
- Song loops, starting when the game starts and ending when the player dies or completes a level

Action sounds like jumping are stored in a BRAM:
- Game events from the Video Output and Game Logic Output trigger these action sounds

Theme music and action sounds are combined in the mixer and output as sound via the AC97.
Planning is complete, now to implement

Major Milestones
- Rudimentary Game Logic & Functionality
- Graphical Overhaul and Gesture Control
- Audio Overhaul and Scoring Functionality

If we have time
- 2 player competitive race
- Additional levels & Items
## November 2009

<table>
<thead>
<tr>
<th>SUNDAY</th>
<th>MONDAY</th>
<th>TUESDAY</th>
<th>WEDNESDAY</th>
<th>THURSDAY</th>
<th>FRIDAY</th>
<th>SATURDAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
</tr>
<tr>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
</tr>
<tr>
<td>29</td>
<td>30</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Monday, 16th**: Design Presentations
- **Tuesday, 17th**: Implement Basic Game Logic
- **Tuesday, 17th**: Video Output
- **Wednesday, 18th**: Video Output
- **Thursday, 19th**: Camera Capture
- **Friday, 20th**: Camera Capture
- **Saturday, 21st**: Initial Debugging

- **Monday, 23rd**: Gesture Recognition
- **Tuesday, 24th**: Gesture Recognition
- **Wednesday, 25th**: Sprite Generation
- **Thursday, 26th**: Audio Output
- **Friday, 27th**: Audio Output
- **Saturday, 28th**: Audio Output

- **Monday, 30th**: Initial Debugging

- **Tuesday, 30th**: Initial Debugging
<table>
<thead>
<tr>
<th>SUNDAY</th>
<th>MONDAY</th>
<th>TUESDAY</th>
<th>WEDNESDAY</th>
<th>THURSDAY</th>
<th>FRIDAY</th>
<th>SATURDAY</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td></td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Initial Debugging</strong></td>
<td></td>
<td><strong>Implement Additional Functionality</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Project Checkoff</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
</tr>
<tr>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
</tr>
<tr>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
<td>31</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Questions and Discussion