SkiFree

Erik Stafl and Eric Grebing
6.111 Final Project Presentation
November 14, 2006
Overview/Inspiration

- Inspired by the classic Windows 3.1 skiing game
- Small skier tries to navigate a downhill path to victory while avoiding trees, hills, and the Abominable Snowman
- Skier will be controlled by a camera tracking the movements of an orange block in user's hand
- Gameplay will be similar to that of the original Windows game
Basic Game Play

- Skier travels downhill and tries to avoid obstacles such as trees and ramps
- Timer module displays time and velocity in the top right corner of the screen
- Skier's velocity increases the longer he goes downhill without crashing
- When skier reaches the bottom and passes through the “Finish” gates, game pauses and is restarted with a button press
Additional Features (if time allows)

- Navigation of a slalom course (skier goes around markers or faces time penalties)
- Jumping with use of ramps and modification of control
- Audio (sound effects and game music)
- Attack of the Abominable Snowman given certain events
Landscape

- Includes sprites such as different types of trees, hills, and signs for direction
- The landscape module tracks the x and y coordinates of each object
- The objects in the landscape move up while the skier is relatively static - simulates downhill motion
- Passes object coordinates to the collision module to compare position of landscape objects to that position of skier
Sprites

- Generated by screen shots from the original Windows game
- .coe file generated through a Matlab script to convert from bitmap
- Each image is stored in an individual ROM
High Level Block Diagram

- Video Input
- Analyzer
- Game Logic
- Frame Buffer
- Sprite ROMs
- Compositor
- Final XVGA Output

Flow of Data:
- Pixel Data (YCrCb) from Video Input to Frame Buffer
- Control Signals from Analyzer to Game Logic
- Address from Game Logic to Sprite ROMs
- Pixel Data (RGB) from Sprite ROMs to Compositor
- Pixel Data (RGB) from Compositor to Final XVGA Output
Game Logic Block Diagram

Skier Video Module
- skier_orientation [2:0]
- skier_jump
- skier_address [10:0]
- skier_x [9:0]
- skier_y [9:0]
- skier_params [7:0]

Collision Detection
- crash_mode

Landscape Video Module
- landscape_image_x_1 [9:0]
- landscape_image_y_1 [9:0]
- landscape_params [7:0]

Skier ROMs
- rom_data [23:0]

Landscape ROMs
- image_address [10:0]
- landscape_pixel [23:0]
- rom_data [23:0]

Game Video Compositor
- time_pixel [23:0]
- game_pixel [23:0]

Timer
- 1_hz
- 10_hz
- 100_hz
- address [7:0]
- rom_data

Time Display
- time_pixel [23:0]

Character ROMs
- rom_data
Video Logic Block Diagram

**NTSC Decode**
- `tv_in_ycrcb[19:10]`
- `f`
- `v`
- `h`
- `yrcrb[29:0]`

**30bit to 18bit Compress**
- `yrcrb_small[17:0]`

**NTSC to ZBT**
- `ntsc_addr[18:0]`
- `nts_data[35:0]`

**ZBT Driver**
- `(RAM0)`

**Control Signal Generator**
- `skier_orientation[2:0]`
- `skier_jump`

**Center of Mass Detector**
- `center_x_1 [10:0]`
- `center_y_1 [9:0]`
- `center_x_2 [10:0]`
- `center_y_2 [9:0]`
- `hcoun[10:0]`
- `vcount[9:0]`
- `desired_color`

**Desired Color Selector**
- `yrcrb_small[17:0]`

**Vram Read**
- `vram_addr[18:0]`
- `read_data[35:0]`

**RGB Converter**
- `rgb[23:0]`

**Compressor**
- `rgb[23:0]`

**Final Compositor**
- `xvga_out, hs, vs, blank`

**XVGA Signal Generator Module**
- `game_pixel[23:0]`
- `htcount[10:0]`
- `vcount[9:0]`

**Game Logic**
- `skier_jump`
Project Timeline

- **11/22 Basic Functionality**
  - All Sprites loaded into ROMs and drawing correctly
  - Collision Detection Working
  - Timer and Timer Display Functional
  - Video Color Detection, Center of Mass Detection

- **12/1 Additional Features in Place**
  - Video Controller Functioning
  - Integration with Signals from the controller
  - Jumping and Slalom Working

- **12/6 Final Touches**
  - Finish Adding More Features to Game (monsters, audio)
  - Make sure all aspects of the Game are fully functional
  - Final Video Compositor to Allow for video input overlay

- **12/10 System Bug Free**
  - Rigorous Debugging and Testing
  - Play time!