GLYCH

Team: The FreqZ

Steven Brown, Travis Gray, Matthew Humphries, Mark Stacey

ECE 3710

University Of Utah

Fall 2014

**Application**

GLYCH is a four player third-person shooter game inside a maze. It is a game of elimination, meaning the last player remaining wins. The game begins by loading maze onto the screen and each of the four players being placed into one of the four corners of the screen. Each player will be a different color stick figure to allow the players to identify their character on the screen. When the game begins the players will navigate through the maze to eliminate other players by firing lasers. The rules of the game prohibit shooting through walls, and shooting in the north and south directions of the screen, allowing for only horizontal attacks. When two players are on the same level, and no wall separates them, they are poised for attack. When this occurs, if a player is hit by a laser, the player is eliminated and the character will no longer display on the screen. The game continues until only on player is left on the screen.

Audio effects will enhance the game by alerting a player when a laser had been fired, and termination effects being sounded when a player has been hit by a laser.

**Proposed Modifications to the ISA**

The main modification to be made to the provided ISA is to change the processor to a 32-bit processor. In this sense, we will ensure ample memory space with the possibility of accessing cellularRAM if necessary, and compress the amount of instructions required for our processor through the use of 32-bit registers.

**I/O**

The input and output required by this application will be:

1. VGA Display
2. 4 NES gaming controllers
3. Speakers

The VGA Display will be Glyph based to minimize memory consumption.

The NES gaming controllers will provide user the ability to interact with the application and will be used to play the game.

The sound will be provided through a music synthesizer written in Verilog, which can then be outputted to a speaker through a PMOD connector to the NEXYS board.

**Assembler Plan**

The assembler will be written in Python. The application will be written by using commas to separate both instructions, registers, immediate values, and comments, to allow for easy parsing.

**Team Responsibilities**

The project will be broken up into four parts with both lead roles and backup roles for each one as follows:

|  |  |  |
| --- | --- | --- |
| **Team Member** | **Lead Role** | **Backup Role** |
| Steven Brown | Application | Processor |
| Travis Gray | I/O | Assembler |
| Matthew Humphries | Processor | Application |
| Mark Stacey | Assembler | I/O |