ELE306 Spring 2006 Project

The TRONG game

The game, the specifications and the rules are described in the attached document.

Project Reports:

- The official course outcomes are:
  1. Critically evaluate and compare the results from different alternatives in each laboratory assignment (l).
  2. Ability to evaluate the trade-offs in digital design selections and to make decisions based on various considerations using modern EDA tools. (e,k,l)
  3. Ability to design, simulate, synthesize and verify complex digital systems with synthesis tools. (b)
  4. Ability to design specifications for a digital system and then design a digital system that meets the specifications. (c)
  5. Perform six laboratory exercises and assignments covering a wide range of potential applications including human interface, control, communication, etc. (c,k)
- In consistent with the assessment of these outcomes, your final project report must contain the followings:
  1. Show how you identify, formulate and solve engineering problems. For example, list the assumptions, rationales of your thinking and the eventual designs. The algorithm (in flowchart format) is a great way to demonstrate how the problem is formulated and solved.
  2. Show how you use the techniques, skills and modern engineering tools you've learned in this class. For example, the VHDL listing, the ASM design methodology, and the hierarchical design using the existing design entities.
  3. Show your ability to communicate effectively. This will be assessed separately in your written report and your presentation. Please prepare your presentation in powerpoint and file should be sent to me before the presentation. The written report should adhere to good technical writing practice.

- **Presentation and demonstration of your design will be held May 4, 11:30-2:30PM. The written project report is due at the time of presentation.**
- **The presentation should be prepared with PowerPoint or the equivalent. The presentation file should be emailed to the instructor prior to the presentation day.**
- **Optionally, you may prepare a project webpage for your project and use it in your presentation. This project webpage will be linked to the course webpage for future reference.**
- **Please report any scheduling conflict ASAP.**

Tim Parys
ELE 306
March 23, 2006

Final Project Specification - TRON

I. Background

Tron is a movie, produced by Disney in 1982, which was notable for its pioneering use of early computer graphics. Although primitive by today's standards, Tron managed to include around thirty minutes of rendered graphics, as well as merged scenes including both real actors and virtual scenes. The reason for this seemingly odd combination is that during the movie, one of the actors is digitized and pulled into the computer system, where programs are sentient entities. There, the human and many of the programs are forced to compete in games until they are eliminated and deleted. Most notable of these games is a sequence using virtual vehicles called light cycles, devices which are infinitely thin, leave impassable trails, and travel at incredible speeds. The participants then race around a large square arena attempting to force their competitors to crash into a wall or a trail left behind by one of the cycles.

II. Project Description

Your goal will be to implement the light cycle game, to the best of your ability, on the Altera Flex 10k FPGA board, using VHDL and reference code from previous labs and exercises. Since we will not have the luxury of virtual modeling and rendering, the game will be displayed as two-dimensional graphics on the provided VGA output. Two players will be allowed to compete at a time, and their lightcycles will leave trails which will become obstacles. For the sake of playability, each player should not die after a single impact, but should have a maximum of 5 chances to play. Additionally, each player should be allowed their own keyboard, which implies that projects should be able to play against each other via serial communication, and send enough data to play on a remote screen.

III. VGA Implementation

This implementation of TRON, will not require the use of fine graphics (for the sake of simplicity), but should be capable of displaying a game screen 40 columns wide by 30 rows high. The VGA signal will still be operating at a full 640 by 480 resolution, although as in previous labs, the perceived resolution will be much less. For reference, the last four bits of the current VGA row and column can be ignored to cut the resolution by a factor of 16.

The final game arena will be 27 rows high by 38 columns wide. This is to allow a border around the edges, one block wide, as well as leave one row empty at the top of the screen. This empty row is for displaying of the number of lives of each player, and a visual indication that a player has crashed, if it is not illustrated elsewhere.

Since this project will need knowledge of not only where the players are on the board, but also where they have been, this project will require the use of VGA memory. Given that the game board will be 40 blocks by 30 blocks, it can be represented with separate coordinates as a 64 by 32 memory, or by $2^{11} = 2048$ different locations. Additionally, at each location, there must be a determination of whether it contains a wall, an empty space, or either player. Hence the minimal representation for a single block can be two bits to represent the four possibilities. This leads to a total of 4096 memory bits, or 512 bytes. While students may use as many bits as they need to represent the game board, they should realize that this minimalistic approach consumes approximately one quarter of the onboard memory elements. While useful and simple to expand, the onboard RAM will quickly consume available FPGA resources if enlarged.

IV. Player Control and Gameplay

To make things as fair as possible, each player is spaced in the middle of the game arena vertically, ten spaces away from the horizontal edges of their respective sides. The row-column coordinates for the two players should be set at (16, 10) and (16, 29).

Once the game has begun, each player moves at a rate of four blocks per second in one of the four major directions (up, down, left, right). Neither player may stop until the end of the round. Both players move simultaneously, and play continues until one of the players moves to a location which is occupied already by either a wall or the trail of either player. At that point, the round ends, visual indication is given that a player has crashed, and the the game resets after a pause. However, in the case that a player has crashed a full five times, the game finishes completely, and stays in the state indicating that a player has lost.

The first player will be controlled by a keyboard attached to the attached PS/2 port. The second player will be playing on a remote FPGA, and respective control signals sent via RS232 serial communication.

V. Serial Communication Protocal

TBA

VI. Suggested Project Milestones

- Display the proper gameboard with the players in the appropriate places using a simple rom initialization file (one is provided at `http://www.ele.uri.edu/~tparys/ele306/`)

- Create a circuit to continuously and reliably update the vga screen, while observing VGA RAM syncronization issues (very important)

- Add sufficient logic to read keyboard input, and move one player continuously

- Make that player stop once they move to a location that was not empty

- Add a second player using a second set of keyboard keys

- Allow game to reset the board after a pause when a player crashes (will require a sequence to reset the contents of RAM)

- Enforce a maximum of five lives for both players

- Change second player control to an input from a serial port

VII. Report for Project

For your final report, you must include the following:

- A clear description of your approach and of the logic you implemented/changed to accomplish the various parts of the project

- A listing of all VHDL code modified

- A brief simulation for each major entity of your project, as well as a analysis saying how the simulation illustrates correct operation (modules reused from previous labs such as `vga_sync` and `keyboard` need not be presented)