ECE 350L – Spring 2016

Final Project Proposal

Ankit Kayastha (ak308) & Sivaneshwaran Loganathan (sl290)
Lab Section 16, Friday 10:05am to 1:05pm
April 10, 2016

We understand and have adhered to all the tenets of the Duke Community Standard in completing every part of this assignment. We understand that a violation of any part of the Standard on any part of this assignment can result in failure of this assignment, failure of this course, and/or suspension from Duke University.

Contents

List of Figures

1 Overall Description of Project

Our goal of this project is to create a 2-D style game similar to the one shown in Figure 1 of the Appendix section. The game is called Bug on a Wire. The context of the game is that it is based on the perspective of the bug, where the objective is to avoid the birds and travel as far along the wires as long as possible. As time goes on, the amount and frequency of birds increase, so the game grows more and more difficult. Once the bug is eaten by a bird, the game is over and the player loses. The bug can avoid the birds by moving left and right between the wires.

For this project, we are planning on implementing the basic features of the game, including the movement of the bug and placement of the birds. We are hoping that it will be a fun 2-D game that can use our proessor completely.

2 IO of Project

1. **Input - Keyboard**
   In order to move the bug across the wire, we would first need to detect the user's input through the keyboard. The user will either hit the right or left key on the keyboard to get the bug to move among the different wires. A keyboard was chosen over just using the switches and buttons on the Altera DE2 board because of the flexibility the keyboard offers relative to just using switches or buttons on the DE2 board.

2. **Output - VGA**
   To display the current game as shown in Figure 1, a VGA interface is going to be used. A VGA interface would allow us to display colors and images on a monitor display; to do this, we are going to attempt to partition a subsection of our data memory that is dedicated to the VGA display. A VGA interface was chosen over a LED matrix to allow for support of multiple colors and a better resolution as a VGA interface allows for a 640x480 resolution. Also a VGA display will bring a bit more flexibility in terms of what we can display.

3. **Output - 7-Segment Display**
   The 7-segment display on the DE2 board will be used to display the time (in seconds) that the user has played the game up to that point. Once the user gets eaten by a bird, the time will reset because that represents the time the bug survived. Because the 7-segment display is already on the DE2 board, we figured it would be easiest to display the time on there since it is meant by nature to display numbers.

3 Modifications to Processor

1. **Integrate Multiplication-Division Unit**
   The multiplication and division unit will allow some calculations in the processor to be done faster instead of just using many addition operations in a for loop. We also do have to integrate our multiplication and division units as a part of the final test of the processor to ensure complete functionality.

2. **Add the input interface for keyboard**
   The keyboard has to be integrated with the processor to allow for input to be read and acted upon by the program. This could be a seperate operation where in the execute stage, the processor would read in a key from the keyboard. Since the processor would be running around 50Mhz, the key will definitely be detected. The keycode will then be saved in a register and branch methods will be used to decide the operations that should follow the press of the key.

3. **Add the output interface for the monitor**
   The monitor will be used to display the gameplay. This will be done by having the VGA module continuously read out a partition of the data memory and display those pixels at that specific location. In order to display images on the screen, we would preload the images as binary strings through the .data section on the assembly file. We would then load that binary string into a register and store it into the data memory partitioned area to display it to the user.

4. **Add the output interface for the 7 segment display**
   The 7 segment display will be used to display the time that has elapsed during the game. We will have a register dedicated to calculating the time and based on our clock frequency, and this time will be displayed to the user through the display. The 7 segment display is on the DE2 board and we would establish an interface that takes in a number as a series of binary digits and then displays it as a decimal digit on the 7 segment display.

4 Description of Code

The code will be written using the ECE 350 ISA and assembled using the assembulator provided. The code is generally a big while loop until the winning condition or losing condition is triggered. The general structure of the operations in the while loop is as follows.

1. Detecting and Processing the keyboard Input
2. Detect collisions for the objects
3. Update the position of the player
4. Check the winning and losing conditions on whether they are satisfied
5. Create new Obstacles
6. Render the Images on the screen

The above operations are all explained in greater detail below.

4.0.1 Detecting and Processing the keyboard Input

The keycode is detected using the PS2 Keyboard module and then stored into one of the registers. Then, to determine what action to take based on the key pressed, several comparisons will have to made using branch operations, and these branches will lead to different subroutines being run based on the user input.

4.0.2 Detect collisions for the objects

In order for the game to end, collision detection has to be fully functional. For instance, the main collision detection that will be done is between the bird and the bug (or player). To do this within the code, there will be a check to see if the coordinates of the player is the same as the coordinate of any of the birds at that time, and if so, the game will be over. Thus, there will also be some branch instructions to determine what to do next based on whether or not the coordinates are equal.

4.0.3 Update the position of the player

The position of the player will by updated based on keyboard input as well as based on time. In the main loop, the player will continue to move forward (by adding some fixed value to some register that holds the player's position), and then if there is keyboard input, some calculations will have to be made to adjust the player's position in those scenarios.

4.0.4 Check the winning and losing conditions on whether they are satisfied

Once again, branch instructions will be used to check the winning and losing conditions. As mentioned before, the coordinates of the birds and the player will be checked against each other to see if the player has run into a bird, and therefore, lost the game.

4.0.5 Create new Obstacles

This is to be determined still based on the difficulty of the implementation and how the project goes. Essentially, there are two options: either the obstance (bird) can be generated every few seconds by reading from the data section of the code. The other way is to have a second player pariticpate in the game and determine when to place the birds (and where). As the project goes on, we will decide which implementation we think will work best.

4.0.6 Render the Images on the screen

In order to render images on the screen, we would represent all of the graphics in the data section of the assembly code. As we need a specific graphic to be loaded on the screen, the program would load that graphic from the data memory and then store it back into the data memory at an address partitioned specifically for the VGA interface.

5 Timeline of Project

5.0.1 First Week - Basic Implementation

Regarding the timeline of the project, we plan on getting a basic implementation completed within one week. This basic implementation will include getting the functionality of the bug moving between the wires and getting the display set up to display the graphical interface for a simple game. By this point, the user should be able to move left and right in the game, and a bird (or more than one) should be able to (hopefully) show up on the display.

5.0.2 Second Week - Continuation of Implementation

In the second week, we want to be able to implement the additional features such as generating the obstacles and being able to display the time on the 7 segment display. During this time, we would also debug some of the basic implementation features to ensure that they are fully working,

A Figures

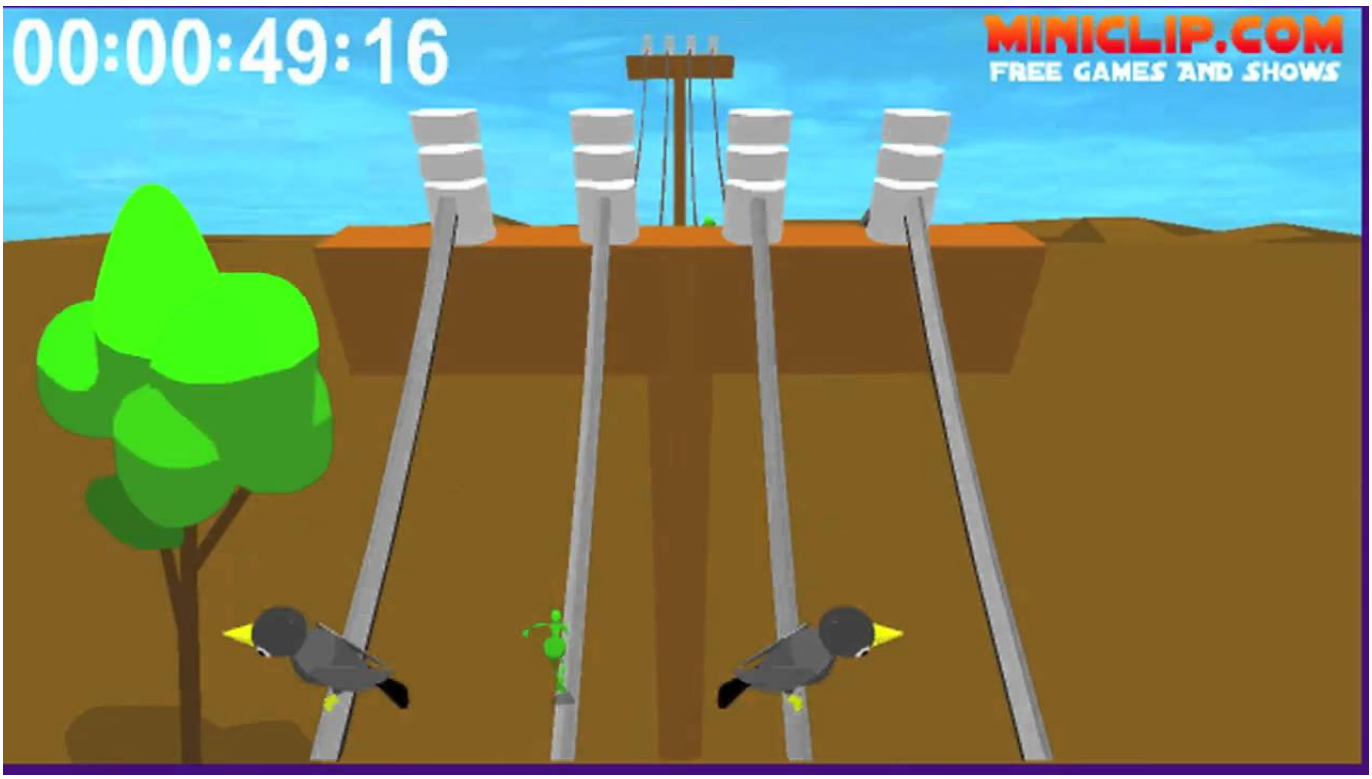

Figure 1: Gameplay of Bug on a Wire