Electric Groove Fish

Project 2

EGR 426

Collin Barnhardt

Dr. Parikh

4/6/21

Basic Operation

A video game clone of a Wario Ware micro-game was implemented using VHDL, Xilinx design suite, the BASYS3 development board, and a VGA monitor. The game works as follows, in the initial state the game is in a hold state until the left button on the BASYS3 is pressed. Once pressed the lightning bolt sprite begins moving left while the button is held. This triggers a 5 second timer that determines whether the game is won or lost. The player navigated the pseudorandomly generated line. If the player reached the city portion of the background before the timer counts to 0, it is considered a win and the link splash screen and modified city sprite is displayed for 2 seconds. Otherwise, if the player does not reach the city the game is reset and a new line is generated, no splash screen is displayed. This is considered a loss. There is a known bug where a line is not generated, if the button is pressed and the timer times out a path will be generated.

VHDL Component Descriptions

blk_mem_gen_0:

blk_mem_gen_0 is a BRAM block generated from the IP catalog as ROM. It is constantly enabled, has a single address port, and output port. The module is initialized with the background data. This module is unique in that it is palletized to preserve memory on the BASYS3 and only works with the decoder module. It is 2 bits wide. The depth is calculated by calculating the number of pixels. The .coe file was generated from a GIMP exported header file and a simple C program that writes to a text file. This process was used to the other .coe files as well.

blk_mem_gen_1:

Contains the ROM for the fish sprite. This data is full width at 12 bits. It is similarly configured to blk_mem_gen_0.

blk_mem_gen_2:

Contains the ROM for the link splash screen. This is 12-bit data.

blk_mem_gen_5:

Contains the ROM for the win condition city display. This is 12-bit data.

blk_mem_gen_4:

Contains the ROM for the bolt sprite. This is 12-bit data.

backGroundDecoder:

This contains the logic for decoding the palletized background data and assigns the 2-bit numbers to the corresponding 12-bit number for the color it represents.

clk_wiz_0:

Generates the 25 MHz clock for the vga control module.

vga_controller_640_60:

This is the provided VGA controller from class.

debounce:

Basic debounce module that ensures the button presses are debounced before being interpreted by the rest of the modules.

boltControl:

This module is quite complicated. It takes in the pseudorandomly generated points that are converted to pixel locations in vertToLines and constrains the motion of the lightning bolt via a state machine that rejects button presses that would deviate from the path of the lines. When the end of the line is reached a 'linked' signal is driven high to indicate that the bolt is at the city. This signal is anded with the output from the sprite control device so that the link splash screen is only enabled when this condition is met.

lineCoordinateGenerator:

This module is also not intuitive. In order to generate the line psuedorandomly it was thought of as a 5x5 grid. The rules are that no more than two vertices can occupy the same row or column on the 5x5 grid. This module utilizes a state machine that reads values from LFSR. The state machine will generate each point in order, if a guessed point violates the rules it will guess again. This continues until 7 vertices are chosen. When seven vertices are chosen the lineReady signal is driven high.

oneToFiveShiftReg:

This LFSR is an 8 bit shift register that assigns an integer value between one and five to the output of the LFSR. This number is fed to the lineCoordinateGenerator state machine.

gameStateMachine:

This state machine handles the timing of the game cycle and provides reset and enable signals to the other state machines based on which part of the game loop the game is in.

vertToLines:

This is another complicated module that takes the 5x5 grid outputs from the lineCoordinateGenerator and converts them to pixel locations on the screen. Then it draws the lines that are defined by the vertices.

Draw_Priority_Control:

This module takes in enable signals for each sprite and assigns a drawing order to the various sprites. This 'layers' the sprites. Note that the win condition enable signals lightAND and linkAND are ANDed signals from the Sprite_Control_Device and the game state machines. This is so they don't display unless the win condition is met.

Sprite_Control_Device:

This module indexes the BRAM modules. The height and width of the sprites are specified via a generic and the location of the upper left corner of the sprite is an input signal. The sprites have a key color, that when detected drives the enable output signal low. The enable signal is driven high when it is calculated from vcount and hcount that the sprite should be drawn.

FullScreen.coe:

The coefficient file for the background.

bolt.coe:

The coefficient file for the lightning bolt.

groovFish.coe:

The coefficient file for the fish sprite.

lightCity.coe:

The coefficient file for the win condition city.

linkSplash.coe:

The coefficient file for the link splash text.

Schematic