T-Rex M152A Final Project

Jonathan Liau, Zijian Zhao

March 18, 2020

Contents

1 Introduction

This project aims to implement Google Chrome's TRex mini game to FPGA board with the Verilog programming language. The goal is to mimick the game mechanism, graphics, and game rules, etc, as close as possible compared to the original version. We will utilize only three buttons on the board as the raw user input, and output to a 640x480 60FPS display through the VGA port. Due to the limitation of port numbers on Nexys3, we might encounter problems that are caused by the actual hardware, therefore, some unnecessary features or relatively insignificant modules might be cut off in order to decrease the total hardware resource used by the program.

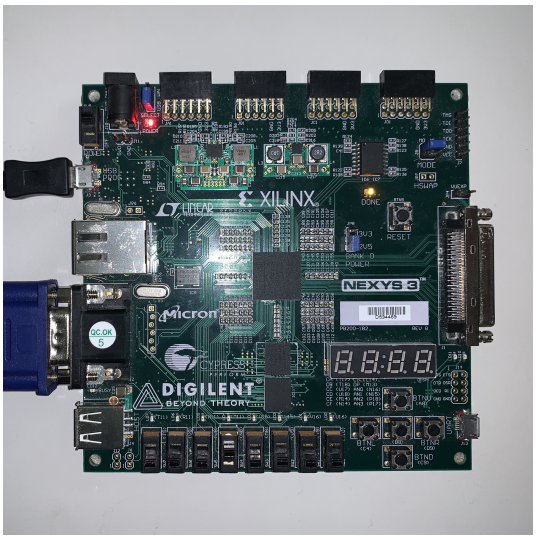

Figure 1: Nexys 3 Spartan-6 FPGA

Design Requirements

- Buttons:
  - jump : Control the jumping movement of the Trex
  - duck : Makes the Trex duck down
  - reset : Reset to Initial-state and zeroed the score count
- Background Image Processing: A selection between two pre-processed background image, which switchs between two states.
- Trex Image Processing: A selection between six pre-processed Trex image, which switchs between six states.
  - Default, Left, Right, duckRight, duckLeft, Dead
- Obstacle Image Processing: Small Size Cactus
- Score Board Processing:
  - Image Processed with drawNumber.v

  - Seven Segment Display with ScoreBoardDelegate.v

- State Controls:
  - BackGroundDelegate.v : Control the state of the correct horizon image for background display.
  - Obstacledelegate.v : Determine the cactus's displaying location in each posedge.
  - ScoreBoarddelegate.v : Control the digital seven segment counting display in the background.
  - TRexDelegate.v : Determine the movement of the Chrome Trex
    - Controls: Jump, Duck
  - gameDelegate.v : Manage the game states
    - States: Initial-state, In-game-state, Dead-state

- Drawing Pixels:
  - VGA Module : Settings for the 640x480 VGA display with proper synchronization and proch border processing.
  - Drawing Modules:
    - drawBackGround.v
    - drawDecor.v
    - drawDino.v
    - drawNumber.v
    - drawObstacle.v

- Main : TRextop.v: Top Module that connects all sub modules and produces the desire display on screen

2 Modules

2.1 Top Module

This is the top module of the entire project. The module takes raw inputs from the Nexys3 FPGA board including four buttons and generates proper output signals to VGA and some led indicators.

2.2 Game Finite State Machine

Game management finite state machine. There are totally three states: 00, 01 and 10 representing the initial state, in game state, dead state respectively.

- Initial state: Objects are rest to their initial position, TRex is reset to the default pattern, and score is cleared to zero.

- In game state: Objects start to move at a specified game speed, score starts to increment. TRex has normal animation (refer to chrome://dino)

- Dead state: Movements are paused, score increment is paused, and TRex is set to dead pattern.

Collision Detection

Once the top module begins to run, each corresponding modules will return a x y coordinate that will determine the color assigned for each pixel. Once the Trexdelegate and Obstacledelegate module assigned grey color to the same pixel, then a collision has occurred.

2.3 TRex Delegate

The TRex delegate module controls everything about the TRex. The module has local variable storing the information including dinosaur origin's X and Y coordinates, its width and height. In addition to calling two other modules dinoFSM and drawDino which are in charge of TRex pattern selection and vga graphic drawing, the module also takes care of TRex's jump movement. By implementing actual acceleration formula in physics.

Jumping Movement

We gave a gravitational acceleration of 1 and an initial veocity of -18 to local parameters. Every frame clock, we want the velocity increment by g and TRex Y displacement by V. (dt is 1 as the calculation is done in every clock pulse)

Helper Modules

- DinoFSM: Determined the movement states of Dino
- drawDino: in shade coordinates of the Dino Shape determine by current x y

  0000 : duckRight
  0001 : left
  0010 : dead
  0011 : default
  0100 : right
  0101 : duckLeft

2.4 Obstacle Delegate

This module manages all cactus obstacles, which randomly selects between the three types of cactus. However, due the limitation of the physical capability of the FPGA board, only the default cactus will be display and used during the actual game implementation.

Movement Management Scope

Similar to TRexDelegate, this module contains a instantaneous X Y coordinate that is corresponding to the default cactus, where X is initiated with the maximum width of the screen (640) and since the cactus only moved in the horizontal direction the Y coordinate is set to GroundY by default. During the execution of the game, the X coordinate will get decrements by one at each posedge moveClk, while Y coordinate remained in constant.

2.5 Background Delegate

This module is mainly in control with the selection of background display between horizon1 and horizon2 in asset file. The idea of the image display, resembles the continuous scrollable in the Crankie Box, where horizon1 and horizon2 properly display right after one another in loops.

Management Implementation Scope

The 2400 pixel wide background image is divided into two parts, horizon1 and horizon2, and put into in to the Crankie Box. Similar to obstacle module, the module used the moveClk pulse to decrements X coordinate of the background position by one at every posedge, in order for the background to move in the same pace.

2.6 Scoreboard Delegate

The scoreboard delegate module has a score counter that increments by one at posedge scoreClk. Just like seven segment display, the four digits of the score are Binary Coded Decimals and their value are directly used as number pattern selectors by the drawNumber module, which is being called four times that correspond to the four digit counting score.

Score Counting in Different States

- 00 Initial State : Cleared and Zeroed when game start or restart
- 01 Dead State : Scoreboard display the total score in count and remain stagnant
- 10 In Game State : Normal counting with proper carrying

2.7 Clocks

In the implementation of the whole Chrome Dino Game, two helper clock modules are in used, they are vgaClk.v and a multi-functional Clock Divider ClockDivider.v

- vgaClk:
  - Input: clk (100MHz Master Clock)
  - Output: pixc_lk (25MHz clock signal)
- ClockDivider.v:
  - format: module ClockDivider#(parameter velocity = 1)
  - where the parameter velocity represents the clock pulse frequency in Hertz, that is requesting the desired pulses per second

3 Simulation documentation

Test Description	Requirement Tested	Result
Displaying The game	VGA connection	Passed
Score Board	VGA connection	Passed
	Seven Segment Display	Passed
	Counting in initial state	Passed
	Counting in Dead state	Passed
	Counting after Dead state	Failed
Buttons	Jump	Passed
	Duck	Passed
	rst	Passed
Collision Detection	Trex intersecting cactus	Passed
Trex Delegate	VGA connection	Passed
	Trex Jumping with Gravity	Passed
	Trex ducking	Passed
	Trex left duck	Passed
	Trex right duck	Passed
	Trex Dead	Passed
Background Delegate	Horizon1 in display	Passed
	Horizon2 in display	Passed
	Crankie Box	Passed
Obstacle Delegate	VGA connection	Passed
	Default cactus	Passed
	type 2 cactus	Failed
	type 3 cactus	Failed

4 Bugs and Fix

- Inferred Latch: Missing Default Case
  - Remedy: By adding the default case, VGA display the game properly
- Overflow: Many time the over flow messed up the result
  - Remedy: setting proper size with global variable by doing math and check it constantly
- Negative Sign: Our jump control and background got into great issues when negative number was applied in the equation.
  - Remedy: Use Singed wire and reg
- Blocking/non-block assignment: one of the bug that we can't catch instantly is misused of Blocking assignment where it should actually be non-Blocking. This bug cause invisible cactus and corruption in background display
  - Remedy: change to non-Blocking assignment

- Class equipment:
  - Remedy: new board plz and a computer with HDMI connector, it is 2020 !!

5 Conclusion

5.1 Summary

The Verilog VGA project with FPGA board comes from the inspiration of the existing trex game developed by Chrome. The original goal of this project is to make a recreation of the original game, however due to the "limitation of the physical capability of the school provide FPGA board", our simple dream seems to be too much to ask. By making a heart breaking decision, the project has reduced the cactus from three type to one and we must also eliminate the clouds and the birds that can possibly make the game much more interesting and challenging. The project can be categorized into four main modules Delegates, Graphic, StateControl and Clocks by task.

- Delegates : each object has their own Delegates module that manage the overall movement and states
- Graphic : display the corresponding color processed by the top modules from the return values of each Delegates module
- StateControl : Determine the states of Dino, Background, and the score board
  - Dino: Dead - Jump - Duck - LeftDuck - RightDuck - default
  - Background: horizon1 - horizon2
  - Score board : Count - Clear - Stagnate
- Clocks :
  - VGA Clock: 25MHz clock signal
  - ClockDiv: multi-functional clock that can handle request for a desire pulses per second signal

5.2 Difficulty

- The board is outdated that cannot fulfill the requirement of our complete project design.
- All tests are limited to office hours due to the dependency of Xilinx board.
- Office Hour is too limited
- VGA Learning Curve

5.3 Recommendation

- Change The FPGA Board and Get Computer that has HDMI connector