DLD PROJECT REPORT

MOTIONREX

GROUP MEMBERS:

SAIRA TALHA

WAJID HUSSAIN

RAAHIM FAISAL

ABSTRACT:

The game "MotionRex" is a unique single player version of the popular Chrome extension T-rex Runner.This game works by utilizing IR sensor(P103 T020) for the motion of Dino. Upon the signal of IR sensor by user the dino jumps over the obstacle which is cacti. There is also an FPGA button which converts the dino to duck position.

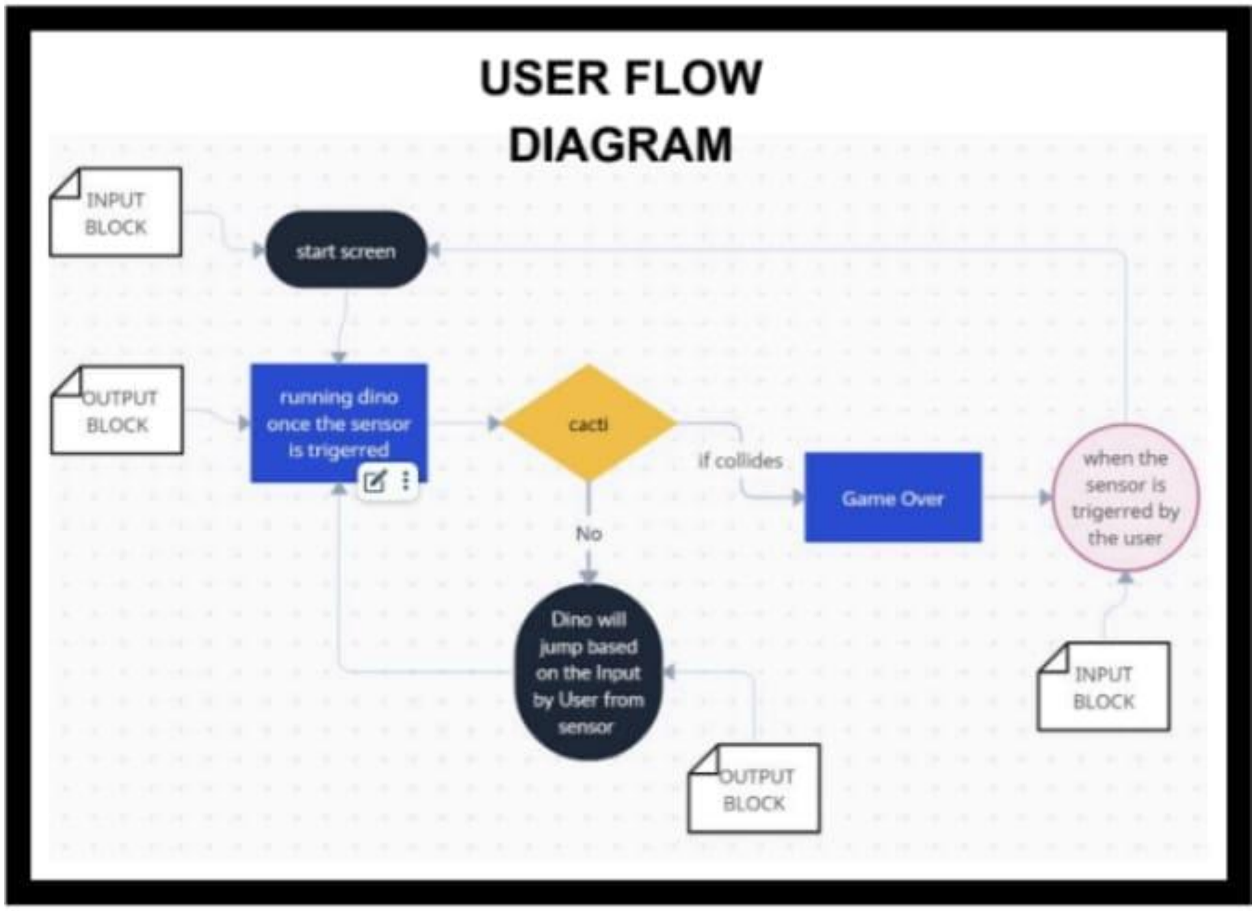
Table Of Contents

Introduction:

The game involves a running dino that can jump

Block Diagram:

The game can be created by utilizing three primary blocks which are input block, control block and output block. The game is based on 5 states which are idle state, run state , moveup state , move down state and dead state.


USER FLOW DIAGRAM
INPUT BLOCK
start screen
OUTPUT BLOCK
running dino once the sensor is trigerred
cacti
if collides
Game Over
when the sensor is trigerred by the user
No
Dino will jump based on the Input by User from sensor
INPUT BLOCK
OUTPUT BLOCK

Input Block:

The input block serves as the interface between external peripherals and the FPGA (Field Programmable Gate Array) on the BASYS3 board. It processes signals from various sources, including user inputs and system clocks, to control the behavior of the game implemented on the FPGA.

The input block consists of input taken through IR sensor (P103 T020). When the sensor signal is 1 by the user the dino jumps. If the timings of sensor signal is correct the dino would jump over the obstacle and if not the dino would collide in the obstacle and the "You Lose" screen would appear.

Pin Configuration: Connected to PACKAGE_PIN G2 on the FPGA BASYS3 board.

Input Block code:

```
module ir_sensor (
   input wire clk, reset,
   input wire ir_in, // Connect this to the output of the IR sensor
   input wire v17_pin, // Connect this to the V17 pin of the FPGA BASYS 3 board.
```

Control Block:

The control block consists of FSM module which has 5 states: - idle state, run state, moveup state, move down state and dead state.

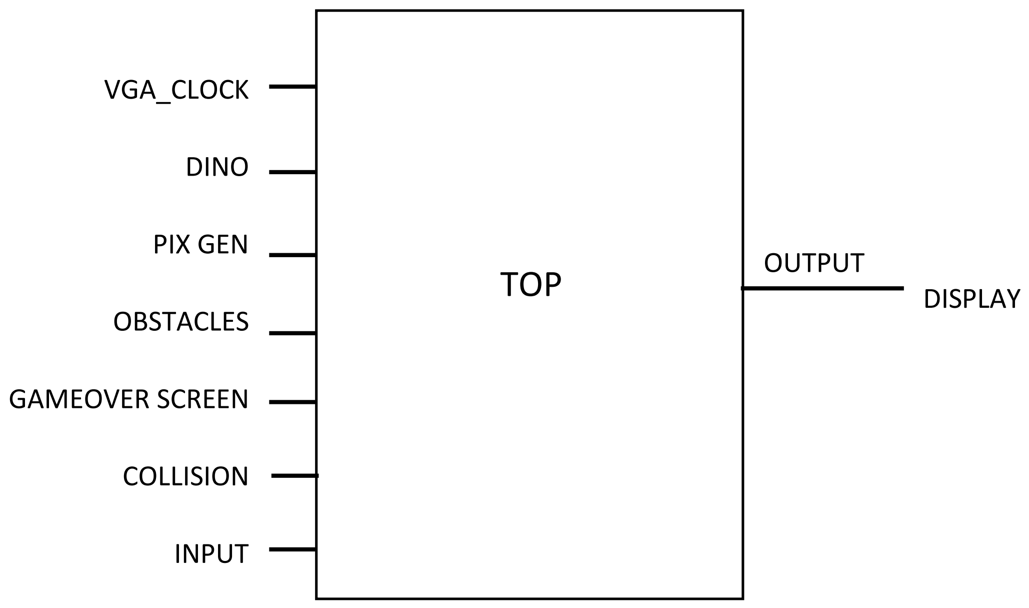
IDLE STATE: in this state only the start screen appears. Once the signal from sensor is 1 from the user the screen changes from start screen to game screen.

Run State: the run state allows the dino to keep on running on the desert.

Moveup State: once the user inputs the signal from the sensor the dino jumps to a maximum height after which it changes its state to Movedown State.

Movedown State: in this state the dino returns back to ground and starts running again.

Dead State: This is the last state, which occurs when the dino collides with the obstacle (cacti). Once there is collision the "you Lose" screen appears.



OUTPUT BLOCK

The output block is responsible for generating the output signals required to display graphics on a VGA monitor. These graphics include images of a dinosaur, cacti, and a background. As the game progresses, the dinosaur will move towards the left side of the screen while the cacti will generate automatically.

output Signals:

Clock and Synchronization:

The output block utilizes a clock divider to generate a 25 MHz clock (clk_25M) synchronized with the VGA timing requirements.

It generates horizontal and vertical synchronization signals (H-signal and V-signal) based on the horizontal and vertical counters (Hcount and Vcount).

Hcount:

Description: Hcount is a 10-bit input signal responsible for horizontal synchronization and determining the pixel's x-location on the screen.

Purpose: It ensures proper synchronization of horizontal display and helps in locating the position of each pixel along the x-axis.

Pin Configuration: Hcount is connected to PACKAGE_PIN P19 on the BASYS3 FPGA board.

Vcount:

Description: Vcount is a 10-bit input signal responsible for vertical synchronization and determining the pixel's y-location on the screen.

Purpose: It ensures proper synchronization of vertical display and helps in locating the position of each pixel along the y-axis.

Pin Configuration: Vcount is connected to PACKAGE_PIN R19 on the BASYS3 FPGA board.

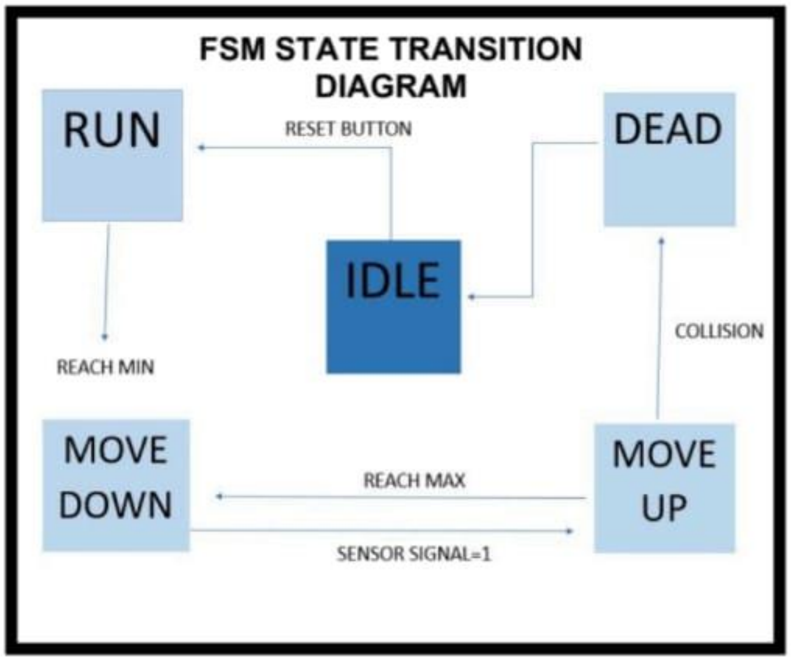
Pixel Generator:

The pixel generator block (pixel_gen) receives the synchronized clock (clk_25M), jump

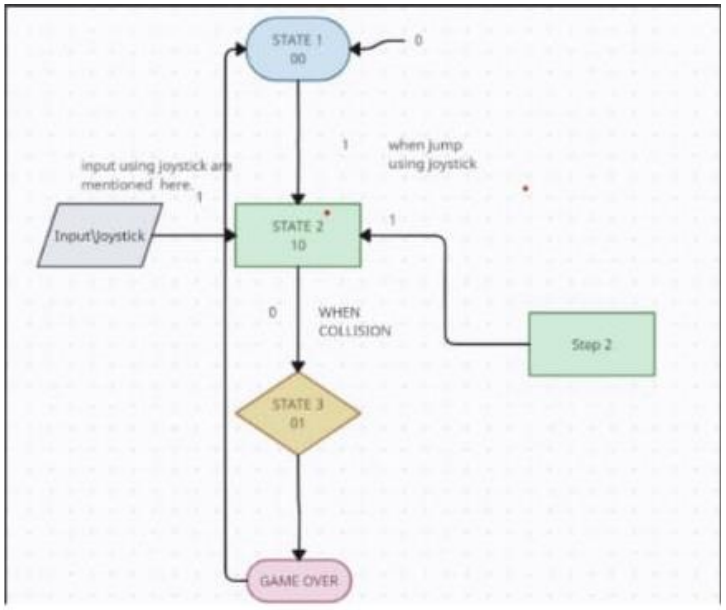
Finite State Machine:

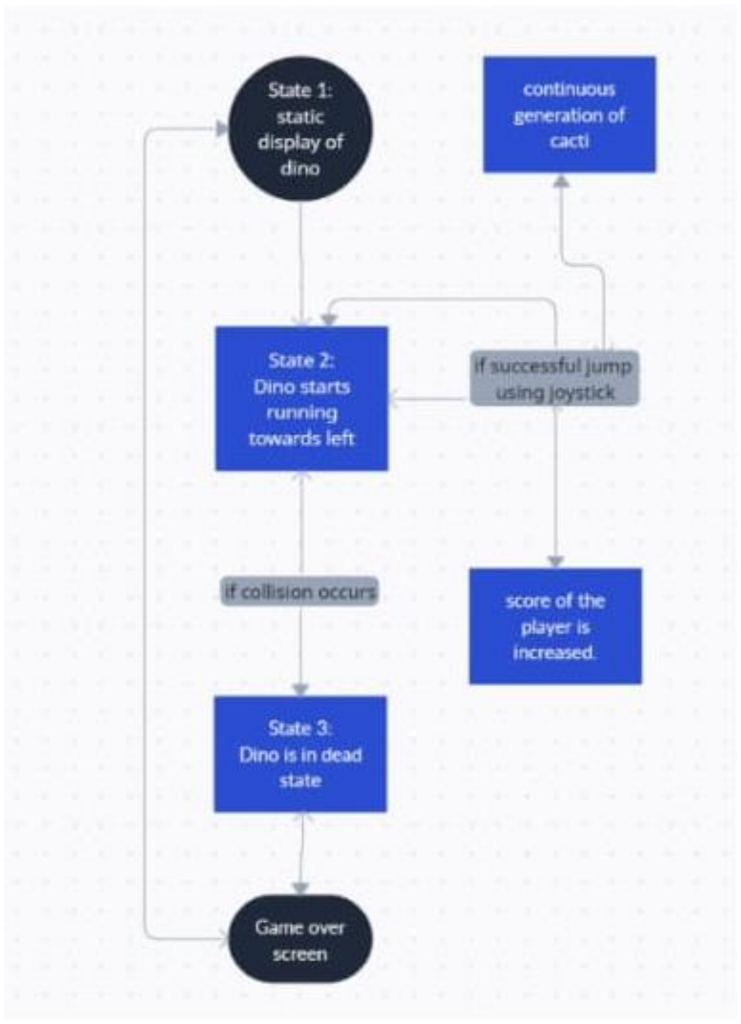
In the process of streamlining the Verilog module, I decided to refine the Finite State Machine (FSM) by breaking down its logic into a separate function called fsm_transition. This function serves as a centre where it takes in the current state, mover position, dinosaur position, collision status, and user input, and then transforms into the next state, along with updated mover and dinosaur positions. By integrating the FSM logic in this manner, the code becomes more easier to understand . Moreover, this approach fosters a more structured and efficient development process by isolating the FSM-related operations from other parts of the module, thus promoting better testing and debugging practices. So, through FSM factoring, the code achieves a balance of clarity and effectiveness, ensuring a smooth foundation for further development endeavors

FSM TRANSITION DIAGRAM:

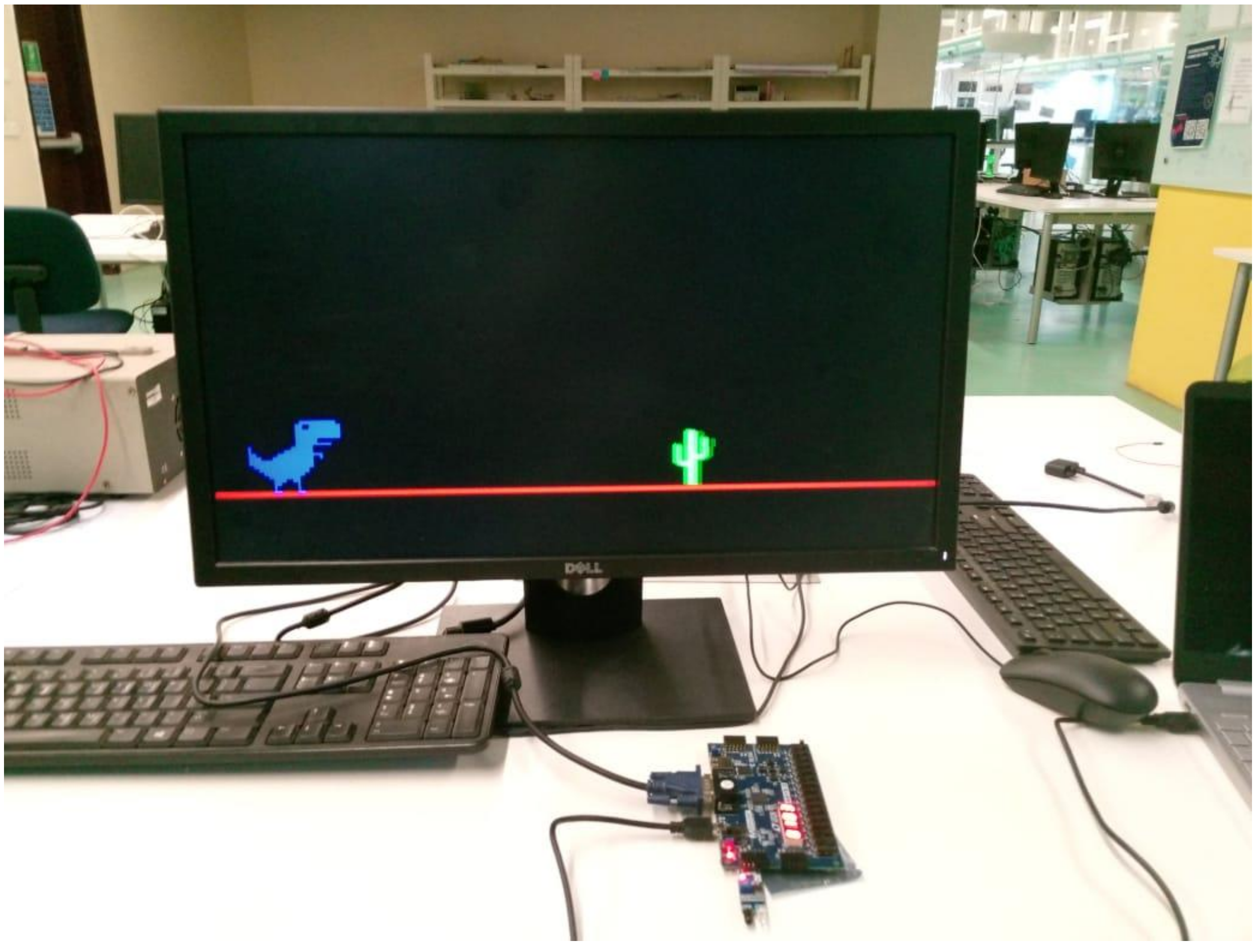


FSM DIAGRAMS:




State 1: static display of dino
continuous generation of cacti
State 2: Dino starts running towards left
if successful jump using joystick
if collision occurs
score of the player is increased.
State 3: Dino is in dead state
Game over screen

RESULT:



Challenges Faced:

While designing our game ("Motion Rex") we faced alot of challenges some of which are stated below:

- It was very difficult to bring the dino back to ground automatically. When the dino was receiving the input it moves to moveup state and stayed there it was not coming back to ground.
- The second main issue was of IR sensor integration as we were having difficulty in figuring out the range of sensor and correct sensor code.

Conclusion:

- In conclusion, the user flow diagram provides a comprehensive overview of the MotionRex game's operation and functionality, detailing the interaction between input, control, and output blocks within the FPGA implementation.
- The Input Block serves as the interface for user inputs and system clocks, enabling player actions such as jumping and ducking through integrated buttons or joystick devices. Meanwhile, the Control Block manages the game's state transitions, defining three distinct states: initialization, running, and dead, based on input signals and collision detection. This finite state machine guides the flow of the game, ensuring proper progression and response to player actions.On the output side, the Output Block generates signals necessary for displaying graphics on a VGA monitor, including horizontal and vertical synchronization, as well as color signals for each pixel. The Pixel Generator within this block dynamically adjusts pixel colors based on the game state and user inputs, creating animations and visual effects to enhance gameplay.
- With the integration of input and output blocks, along with the control block, forms the backbone of the game's functionality. , these components create a dynamic and interactive gaming experience. The game FSM module serves as the central control system, managing game states and progression. Additionally, modules like Dino, clock divider, and counters contribute to character animation, timing synchronization, and numerical display