Omar Khalil, Mariam Wael

Project Report

Design approach:

This design implements a Pong game on a VGA display using Verilog. The design makes use of several key modules to manage the game display, game state (paddles, ball), scoring, and input debouncing. The modules are interconnected to provide a smooth gameplay experience with proper synchronization, collision detection, score updates, and player interaction.

Modules:

top module

All other modules' operations are coordinated by the top module, which acts as the primary controller. It manages the clock, processes input, and makes sure the game's different parts function as a whole. It controls score updates, routes VGA signals, and initiates ball movement, for instance.

seven_seg

The score must be shown on a seven-segment display by this module. It ensures that the score is updated in real time by converting the binary score values into signals that govern each display segment.

clock_divider

Slower clock signals are produced from the main system clock by the clock_divider module. Timing tasks like ball movement, display updates, and game events require these slower clocks. It ensures that everything in the game happens at the right pace.

binary_counter

The binary counter module counts certain actions (like updates to the ball position) or generates pixel positions for the VGA display in order to keep track of game events. It can be set up to overflow or reset when needed, and it counts in binary.

bcd

In order to display the game's score on a seven-segment display, the binary format must be converted into binary coded decimal using this module. The conversion guarantees accurate display of multi-digit scores.

vga_controller

The VGA synchronization signals required for correct display on a monitor are produced by this module. It controls both vertical and horizontal sync signals and guarantees that all game elements including the paddles, ball, and score.

game_display
This module controls the actual rendering of game elements (paddles, ball, and score) on the VGA screen. It takes the current positions of game objects and outputs the corresponding VGA pixel data to display them on the screen.

text
This module handles the display of text messages, such as the game score or a "GAME OVER" message. It converts ASCII codes into pixel data for display on the screen, ensuring that the game's status messages appear in the correct location.

debouncer
This module filters out noise from button presses and generates clean signals for player input. Since mechanical switches can generate spurious signals (known as bouncing), this module ensures that only valid input events are passed to the game logic.
timer
Throughout the game, the timer module records the passing of time, allowing for time-based events or countdowns. In order to keep the game moving along at a steady pace, it can initiate actions such as initiating a new round or resetting the game after a point is scored.

score_counter
Both players' scores are tracked by the score_counter module. Every time a player scores a point, the score is updated; if necessary, the score is reset. To show the current score on the screen, this module collaborates with the seven_seg and bcd modules.

ball
The ball module manages how the ball moves and interacts with the walls and paddles. It checks for collisions, updates the ball's position, and adjusts its direction as necessary.

Challenges:

Debouncing Input: If not appropriately filtered, the noisy signals produced by mechanical switches used for player controls may result in unpredictable behavior. In order to guarantee smooth paddle movement, the debouncer module was crucial for cleaning up user input.

Making sure the score would stay visible during the game was one of the challenges. When rendering the paddles and ball, the game display logic must not overwrite the score area. The score area is supposed to be drawn first and updated without being overwritten by other game objects using rendering logic coordination; however, this is the one thing we failed to achieve.

VGA timing: It was hard to get the game to sync with the VGA display refresh cycle. It was necessary to carefully adjust the vga_sync module to make sure that the game objects were displayed at the appropriate moment throughout the screen refresh cycle.

Block Diagram:

This is an explanation of the components of the block diagram and how they interact:

1. Input: Takes user input of paddle movement through buttons of FPGA and sends this data to the `Paddle Movement` block.

2. Paddle Movement: Processes input data to determine the paddle's position and communicates with the `Graphics` block to update the paddle's visual position and with the `VGA Controller` for synchronization.

3. Ball Movement: Manages the ball's position and velocity, updates its movement based on collisions detected by the `Collision Sensor` block and sends updated position data to the `Graphics` and `VGA Controller` blocks.

4. Collision Sensor: Detects when the ball collides with the paddle, walls, or boundaries. It sends collision events to `Ball Movement` to adjust trajectory and provides data to `Score Num` if the ball passes a boundary (scoring a goal).

5. Graphics: Responsible for generating and managing the game visuals, including the paddle, ball, and background and receives data from `Paddle Movement`, `Ball Movement`, and the `Text Generator` and sends rendered visuals to the `RGB` block.

6. Text Generator: Generates on-screen text for the score and provides this data to the `Graphics` block.

7. Score Num: Tracks the player's score based on goals and sends updated score data to the `Text Generator` for display.

8. RGB: Converts the graphics data into a format suitable for display and acts as an intermediary between `Graphics` and the `Display`.

9. VGA Controller: Synchronizes the visual output to the display using VGA signals so it interfaces with `Graphics`, `Paddle Movement`, and `Ball Movement` to ensure proper timing and visual accuracy.

10. Clock: Provides the timing signal for the system and synchronizes `Ball Movement`, `VGA Controller` and other blocks to maintain consistent game speed and display refresh rate.

11. Display: Outputs the game visuals for the player to see.