EEL 5930 – System-on-Chip Design - Spring 2023

Homework 1: Pong Game Implementation in Basys 3 FPGA

Group Members: Nurun Nahar Mondol, Md Kawser Bepary, Dipayan Saha

Objective:

The objective of this homework is to implement a hardware design of the Pong Game on Basys 3 FPGA. The hardware design involves the use of the I/O peripherals (i.e., VGA port), input signals, developing the game logic, and generating the VGA output for the monitor.

Pong Game Description:

The FPGA pong game is controlled with push buttons and switches. The display of the game is shown in figure.1. There are two players in the game, and each has a gaming pad they can move up and down (Gaming pad_p1 and Gaming pad_2). The triangular ball would move in a random direction and the players must hit the ball by moving their dedicated pad up and down using two push buttons. If a certain player (let's assume player 1) misses the ball, the ball would go out of the screen and the opposite player (player 2) would win a point (Socre_p2 will increase by 1). The point will increase until a player gets to 9 points. Once a player reached to score of 9, he/she wins, and the game is over. The game can be started again with the reset image switch (SW1).

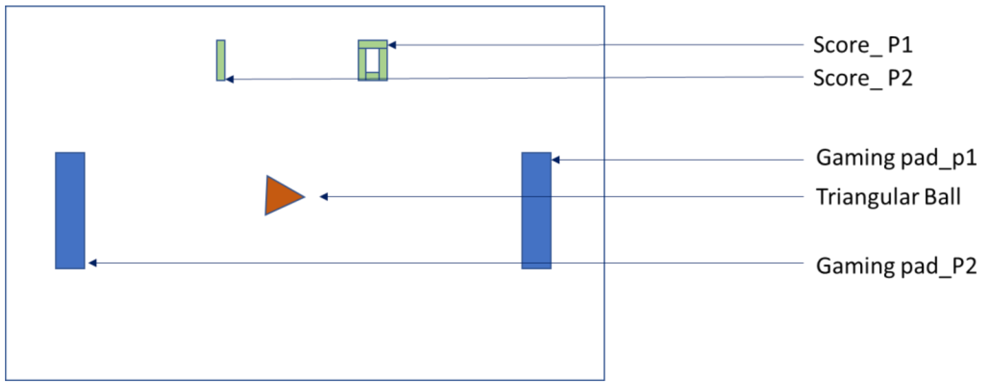

Figure 1. Display of the Pong Game

Tools and Hardware Used:

- Xilinx Vivado
- Digilent Basys 3 FPGA board
- VGA cable
- USB-UART Cable to program the board
- HP 24-mh monitor (FHD – 1920x1080) with VGA input port

Block Diagram:

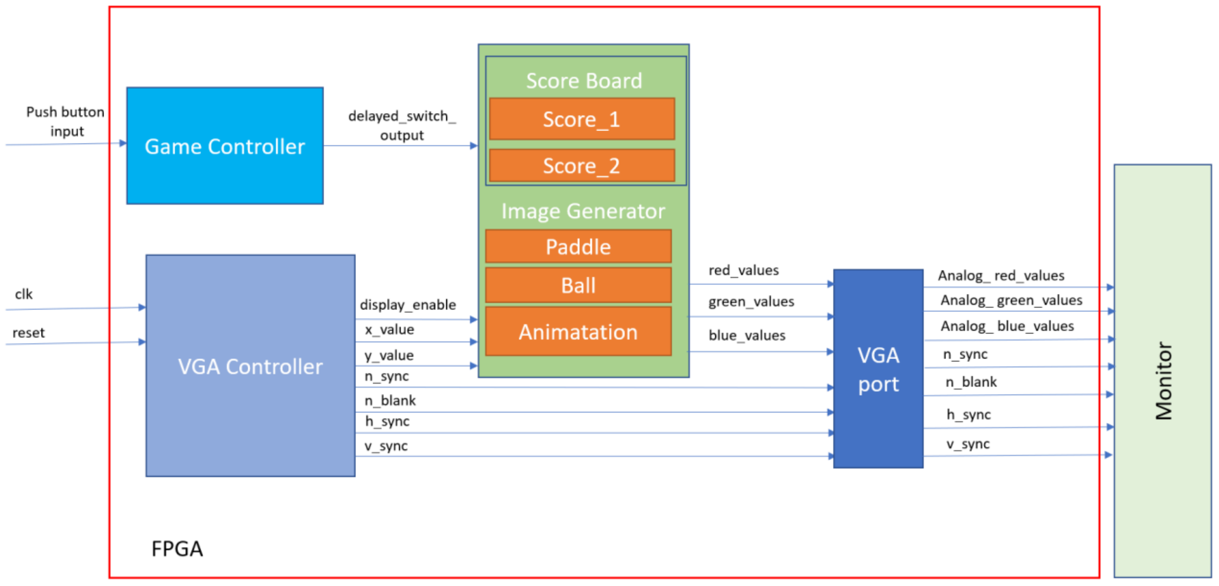

Figure 2. Overview of the hardware implementation

Hardware Implementation:
We implemented our pong game design with a mix of VHDL and Verilog languages. We modified the reference VGA controller (VHDL) for our display specification and wrote the rest of the modules in Verilog. A block diagram of the hardware implementation is shown in Figure 2. The implementation of this project includes several blocks. A detailed description of these blocks is given below:

VGA Controller: The main function of the VGA (Video Graphics Array) controller is to ensure graphics output to the display. It generates the horizontal and vertical pixel coordinates, sync pulses and necessary timing signals to interface with the monitor. The VGA controller takes the clock and reset signals as input. According to the datasheet, the parameters of the VGA controller have been configured to maintain the proper pulse width of different signals: horizontal synchronization, horizontal front, and back porch, horizontal display, vertical synchronization, vertical front, and back porch, vertical display, etc.

We used the following parameters in the VGA controller for a display window of 1280x960 resolution at a 60Hz refresh rate:

Resolution = 1280x960 pixels
Pixel clock = 102.1 MHz (We used the Basys 3 system clock of 100MHz)
Refresh Rate = 60 Hz
Horizontal: Display = 1280, Front Porch = 80, Sync Pulse = 136, Back Porch =216
Vertical: Display = 960, Front Porch = 1, Sync Pulse = 3, Back Porch =30

Out of the generated signals `display_enable' signal and horizontal and vertical pixel coordinates are passed to the image generator block, and horizontal synchronization, vertical synchronization, `n_blank', and `n_sync' signals are fed directly to the VGA port.

Game Controller: The main purpose of the game controller is to control the movement of two gaming paddles from outside using the push button switches of the board. The game controller takes the clock and push button signals as input. The push button signals are delayed with a debouncer to generate a stable switch output. We have used two push buttons for each player to control the paddle movement in up and down directions. Player 1 (Left paddle) uses the BTNL (up_L) and BTND (down_L), while player 2 (right paddle) uses the BTNU (up_R) and BTNR (down_R) to control the paddle movements. When a switch becomes active, the corresponding output signal becomes 1. These outputs of the switches are sent to the image generator block to move the paddle by a certain pixel value. The height, width, and velocity of the paddle are configurable by changing the parameters. We have used paddle height = 200 pixels, width = 20 pixels, and velocity (each push) = 50 pixels.

Image Generator: The image generator takes the output from the game controller and VGA controller and generates RGB values for the display. This block is responsible for ball-paddle creation, animation, and score generation. Along with the clock and reset signal, image generator takes all the switch output from the game controller, (x,y) coordinates of the pixel and display enable signal from the VGA controller. The image generator uses 16 addresses of the ROM to generate a triangle-shaped ball. The length, width of the paddles, size, initial position, and velocity of the ball are defined by parameters.

After creating the ball and paddles, we developed the logic for the ball movement. Initially, the ball moves in a fixed direction with parameterized velocity in both x- and y-axis. The pads are moved up and down with the push buttons. As soon as the ball hits the pad, the ball is moved in the opposite direction. The movement of the ball is updated only once in a frame using a 60 Hz refresh tick for proper animation. Consequently, we developed a logic block to keep track of the scores of two players. When a player misses the ball, the score of the opposite player increases by 1. The scores calculated in this block are displayed using the ASCII character code and ROM addresses defined for the digits 0-9. Another round of the game is started with two registers `next_round1' and `next_round2', after a miss by any player. The game is stopped with a register named `game_stop', when a player reaches a score of 9. The game can then only be restarted with the input reset image switch.

VGA Port: The RGB pixel values corresponding to each pixel are fed to the VGA port from the image generator along with the sync pulses. It uses a digital-to-analog (DAC) converter to convert these digital signals to analog and display the relevant pixel information on the monitor.