EEL 4930 - System-on-Chip Design

Spring 2023

Homework 1

Submitted By

Hasan Al Shaikh (34715621)

Shuvagata Saha (58898380)

Mohammad Bin Monjil (50782093)

Date: Feb 10, 2023

Introduction:

The game of Pong is a classic arcade video game that has been enjoyed by players since its introduction in 1972. It is a simple two-dimensional game in which two players use paddles to hit a ball back and forth across the screen. In this report, we describe the implementation of a Pong game on the Basys3 FPGA (Field-Programmable Gate Array) board and display it on a VGA monitor. The Basys3 board is a low-cost, entry-level platform that provides a convenient way to learn and experiment with digital design using FPGAs. Our implementation takes advantage of the board's on-board devices, including push buttons, a VGA connector, to display the game on a VGA monitor.

Implementation:

The block diagram below shows the high-level block diagram of our implementation of the

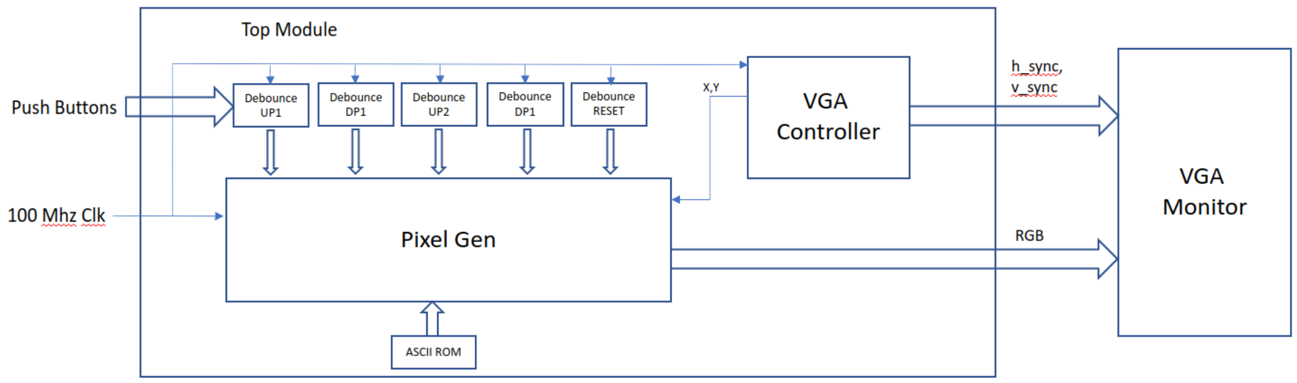

Fig 1: High-level block diagram

pong game. The top module encloses several modules that process inputs, create game logic and output the VGA signal to VGA monitor.

Pixel Gen:

The module "pixel_gen_2p" is a Verilog module for generating pixels for the display and implementing game logic, including ball movement, paddle movement, player scorekeeping, and game resetting. It takes inputs such as clock, reset, video_on, and various button presses, and outputs the RGB color value of each pixel. The module also has parameters that define the display boundaries and the size and movement of game elements such as the ball and paddles. Ball bounce logic is also implemented for top and bottom wall and the paddles. The score of each player is tracked and updated by detecting when the ball crosses the coordinates of the paddles. The game is over whenever a player scores 9 and the winner is shown on the screen. A separate ASCII ROM is used to display the scores and words. The triangular shape of the ball is implemented using a fixed ROM within this module.

VGA Controller:

This module generates video signals to display images on a computer monitor. The code defines the input signals (clock, reset), output signals (video_on, hsync, vsync, p_tick, x, y), and parameters (screen dimensions, display area, front porch, back porch, and retrace). The code also defines two counter registers (x,y) to keep track of the pixel positions. The video signals are generated based on the state of these counters and the VGA standards. The video_on signal is asserted only when the pixel counts are within the display area. The hsync and vsync signals are generated during the horizontal and vertical retrace, respectively. Our implementation has 800*525 pixels, and a 60 Hz refresh rate requires around update of 25MHz. We generated a 25 MHz signal from the 100MHz signal of The Basys 3. This signal (p_tick) is used for syncing the pixel counts, horizontal sync, vertical sync, and video on signals.

Debounce Module:

This module processes push button signals from the push buttons of the board and provides single outputs to avoid button bouncing. Five debounce modules were instantiated for five buttons (4 inputs and 1 reset) and outputs of these modules are fed into the Pixel Gen module.

ASCII ROM:

This module provides fixed values for alphanumeric display and is used by the Pixel Gen module for showing scores. It is also used to show the winner when the game is over.

Game Modes:

Our implementation features two modes: Game Play and Game Over. The game begins in Game Play mode as soon as the FPGA is turned on and the player starts playing (as seen in Figure 2).

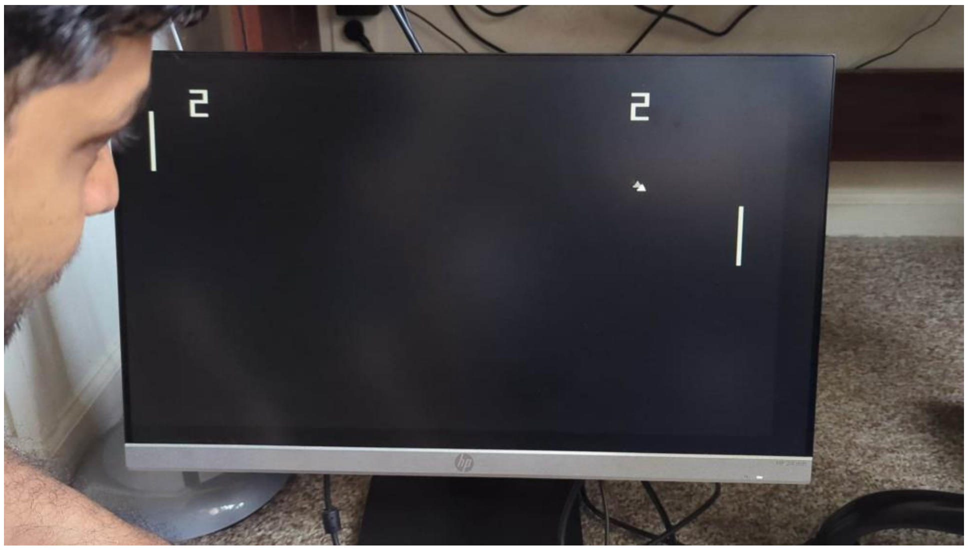

Figure 2: Pong Gameplay

The game switches to Game Over mode when either player reaches a score of 9 and displays the winner of the current game (as seen in Figure 3, where "P2 VIC" is displayed).

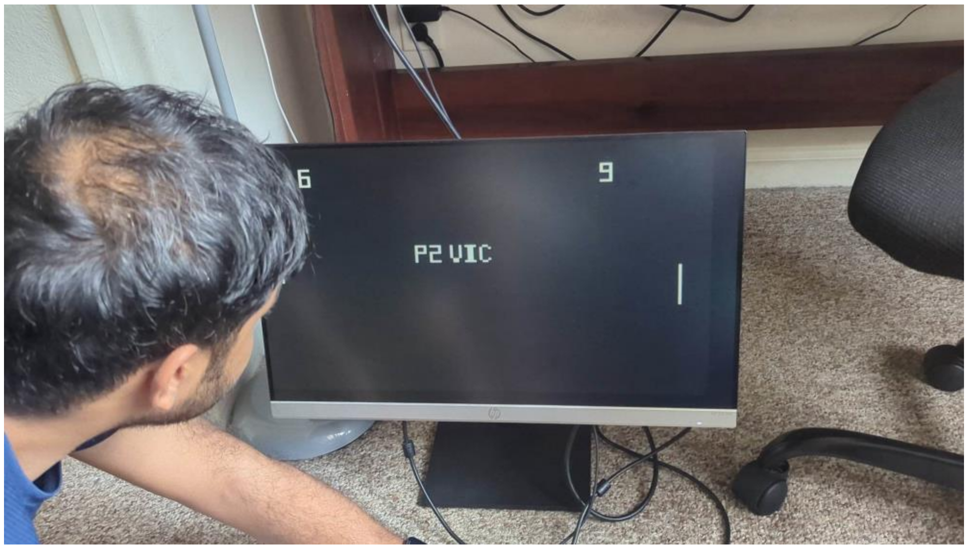

Figure 3: Game Over mode

Challenges in Implementation:

We encountered several obstacles during implementation, including:

Score Update Problem: The first challenge that we faced was with the player score update. The objective was to increase the player score by 1 each time the ball crossed the other player's paddle. However, what we observed was that the score was not updating as expected, but instead was increasing by a random value. This was due to a technical glitch in the code. To resolve this issue, we had to implement a flag for detecting ball crossing and incorporate it into the score update logic. This flag helped us accurately track the ball movement and correctly update the player score.

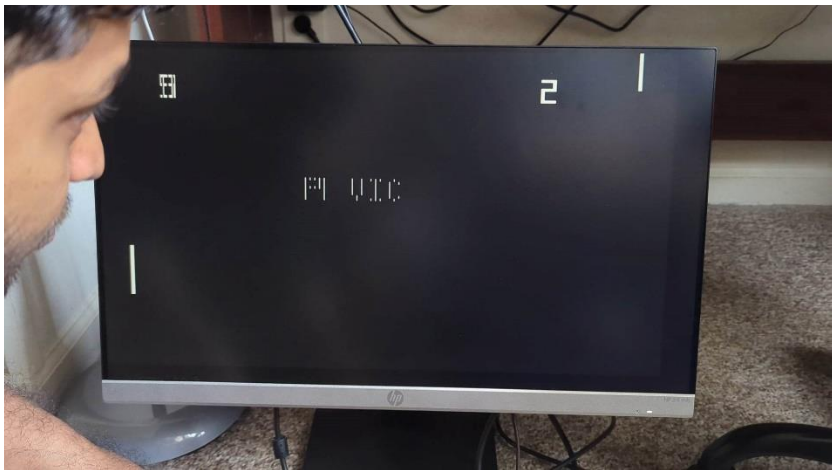

Fig 4: Glitches in Alphanumeric display

Glitch in Alphanumeric Display: Another challenge that we encountered was with the score display. The score was supposed to update whenever the ball crossed the boundary, but this was not happening as expected. The score would only update correctly when the ball returned to the display, but in the interim, the display showed garbage values. In addition, random values were appearing where the game over status was supposed to be shown in the Game Over mode. Both of these glitches can be seen in Figure 4. The cause of these problems is yet to be determined, but we plan to resolve them in our future work.

Contribution of members:

Hasan Al Shaikh – Worked on the movement of the paddles and the collision logic.
Shuvagata Saha – Worked on the scorekeeping logic and showing the score on the monitor.
Mohammad Bin Monjil – Completed the report, worked on the movement of the ball and integrating our design to the VGA controller.

Conclusion:

In conclusion, we successfully implemented the classic arcade video game Pong on the Basys3 FPGA board and displayed it on a VGA monitor. Our implementation utilized the on-board devices of the board, including push buttons, to provide a fun and interactive gaming experience. The process of implementing Pong on the Basys3 board provided a valuable learning experience, allowing us to gain hands-on experience with digital design using FPGAs.