Final Project:

NIOS-II VGA System

By:

Mark Hanna	A2110650
Hasan Mroueh	A2111411
Ziad El Cheikh Ibrahim	A2213444
Abderrahmen Ghader	A2112126

Presented to:

Dr Ghattas AKKAD

A worksheet for CPEN309:
Embedded Controllers Lab

09 – 05 – 2023

Introduction:

In this project, NIOS-II VGA system, we will add video display capabilities to our system using the DE1-SoC Video Graphic Array (VGA) port. Furthermore, external switches and pushbuttons will be used for color control, implementing an “Etch a Sketch” game, in addition to bonus features discussed in the design section.

Objective/Goal:

The aim of this project is to get familiar with the DE1-SoC’s peripherals and employ the VGA system, in addition to program different methods to perform different operations considering the user’s input which will be used to create an Etch a Sketch. This experiment also aims to make use of the NIOS-II external pushbuttons’ Interrupts to draw pixels on the VGA display. Implementing this experiment requires the use of the Altera type definitions, the Altera Avalon PIO macros, and bit-masking.

Tasks:

The tasks for this project are:

1. Program the FPGA board using the updated NIOS II SDRAM Design.
2. Set up and test the VGA hardware.
3. Create an Etch a Sketch by implementing the below functions:

- Start by drawing a pixel in the center of the VGA screen.
- Each time the KEY pushbuttons (one for x, one for y) are pressed, draw another pixel according to the following rules:
    i. For SW(0) = 0 put the new pixel to the left of the previous one
    ii. For SW(0) = 1 put the new pixel to the right of the previous one
    iii. For SW(1) = 0 put the new pixel below the previous one
    iv. For SW(1) = 1 put the new pixel above the previous one
    v. Bonus discussed in the Design section.
- Provide a mechanism for clearing the screen and starting over.
- Add color selection functionality to your design. Use at least one switch input for each color (R, G, & B), more if you want color options.

Design:

The button interrupt and switches were used heavily in order to get familiar with the VGA system. The handle_button_interrupts function, which is implicitly called when a button is pressed, includes all the special features implemented in our project. Macros from “altera_avalon_pio_regs.h”, variables from “alt_types.h”, and memory locations from “systems.h”, “altera_avalon_timer_regs.h”, “sys/alt_irq.h”, and “altera_up_avalon_video_pixel_buffer_dma.h” were implemented in order to draw pixels and clear the screen, in addition to various other features.

- Main features/options: First, the user has to choose between 4 options using switches 6 (SW5) and 7 (SW6):
  1- Free drawing: The whole screen is cleared, a box is drawn in the middle of the screen, and the user starts drawing 2 by 2 boxes next to the initial box. The boundaries in this mode are the borders of the screen (320x240).
  2- Draw in Pac-Man1: The whole screen is cleared, and a Pac-Man shape is drawn on the screen where the user can draw inside of it. The boundaries in this mode are the borders of the Pac-Man’s shape (the 0s in the array).
  3- Display Pac-Man2: The whole screen is cleared, and a Pac-Man shape (different than the one in the previous mode) is drawn on the screen. This mode is a display-only; so, no drawing is allowed in it so no boundaries are set.
  4- Display Mario: The whole screen is cleared, and a Mario shape is drawn on the screen. This mode is also display-only; therefore, no drawing is allowed.
- Colors: The user is given 8 different color options: Black, Red, Green, Blue, Orange, Yellow, Brown, and Pink. The color can be selected using the switches 8 (SW7), 9 (SW8), and 10 (SW9).
- Directions: The user can move horizontally (left & right) using switch 1 (SW0) and pushbutton 1 (KEY0), and vertically (up & down) using switch 2 (SW1) and pushbutton 2 (KEY1). The user is also given the ability to move diagonally in all 4 directions using pushbutton 3 (KEY2) after setting the desired direction using SW0 and SW1 and enabling diagonal movement using switch 3 (SW2). Diagonal movement will disable pushbuttons 1 and 2.
- Borders and Restrictions: 2 variables were created to easily take care of the borders: freeDrawing and DrawEnabled. When freeDrawing is enabled, the user drawing is limited by the borders of the screen, which means that we are in the free drawing

mode. When freeDrawing is disabled, this means that the borders are set to the 0's of the Pac-Man1 drawing array. When drawEnabled is enabled, the user is allowed to draw boxes. Otherwise, we are in one of the 2 display-only modes, so no drawing is allowed. When moving horizontally, we have to make sure that the x0 coordinate of top left corner of the box should be larger than 0 and the x1 coordinate of the bottom right corner of the box should be smaller than 319. Therefore, before drawing a box, we need to check that it is going to be within the boundaries set, which is done using the following formulas: $(x0 – 4) > 0$ and $(x1 + 4) < 319$ (we are using 4 here instead of 2 because each box is of width 2 plus some additional spacing between boxes). The same concept is applied when setting the vertical boundaries, but the comparison is done with respect to 0 and 239. The diagonal borders are a mixture of vertical and horizontal ones. In order to check the borders of the Pac-Man1 shape, two variables r and c come into play, which are used to keep track of the user's position in the drawn shape's array after drawing. As long as there is a 1 in the position (index, r, c) of the array, the user can draw, otherwise, drawing is not permitted. The values of r and c are changed with every box drawn inside the shape based on the direction specified. When free drawing is enabled, the values of r and c are ignored, since their role is limited to setting the borders of the Pac-Man1 shape drawn.

- Clearing and Restrictions: The clearing pushbutton PB4 (KEY3) is not only used to clear the screen, but it is also used as an enter button to switch between displays. The screen can be cleared with no restriction if the first 2 modes, which are the free drawing the Pac-Man1 drawing. But when it comes to the display-only modes (Pac-Man2 and Mario shapes), the screen will not be cleared unless a different mode is selected. This restriction is applied using the previousDrawing variable, which is used to compare the user's mode input with the previous one.
- Drawing: In order to draw the first, second, and third shapes, there is a for loop that loops through the array of the shape (the shapes were initialized in a 3D array, where each element is a 2D array corresponding to a specific shape) and draws a box where a 0 is found (because 0s define the outline of the shape to be drawn).

Conclusion:

In conclusion, this experiment allowed us to make use of the NIOS-II VGA, switches, and pushbuttons to develop an interactive “Etch a Sketch” game. The program methods we implemented in our code worked as intended and we were able to employ the appropriate peripherals and functions as described in the Tasks section above. Overall, this experiment provided an excellent opportunity to develop a practical understanding of microcontroller programming and the FPGA board’s capabilities.