ECSE 324 Lab 4 Report

Yuhang Zhang 260787441

Pengnan Fan 260768510

### 

## 

## 

## Content

#### VGA

#### KEYBOARD

#### 

**VGA**

The objective of this part of the lab was to write VGA\_draw\_point\_ASM and the VGA\_clearbuff\_ASM to clear all the valid memory locations in the character buffer and pixel buffer.

Besides, the VGA\_write\_char\_ASM, the VGA\_write byte\_ASM, and VGA\_draw\_point\_ASM should be also implemented that will write the hexadecimal representation of the value passed in the third argument to the screen, passed its result in the third argument to the screen at the (x,y) coordinates given in the first two arguments.

To do this, two loops were used, iterating through the x and y-axis of the screen. For the clear char subroutine, we were to iterate within 320 and 240.Every pixel or char of the has a unique address where to store the corresponding data. For the pixel buffer addresses

The address is assembled by using a base address and the x and y coordinates.

For the clear char subroutine, the y coordinate is shifted to the left by 7 bits and “ORRed” with the x coordinate then added to the base address. For the clear pixel, the y coordinate is shifted left by 10 bits and the x coordinate is shifted left by 1 bit. This part of the lab was fairly simple and straightforward, and both clear pixels and clear char were extremely similar to each other. No challenges were faced when writing this code.

For the VGA\_write\_char\_ASM and VGA\_write\_byte\_ASM. we check range, in this case, is a resolution of 80 \* 60 characters. We shift values and used the function ORR to store at the base address. No particular challenge is faced during the entire lab.

**Keyboard**

The second part of this lab asks us to create a simple application that uses the PS/2 keyboard and VGA monitor. The application should read raw data from the keyboard and display it to the screen if it is valid. Only the VGA write byte ASM subroutine is needed from the VGA driver, and the input byte is simply the data read from the keyboard. We are asked to implement a subroutine to check the VALID bit in the PS/2 Data register and if it is valid, the data will be stored at the address in the char pointer argument. Finally, the subroutine will simply return 1.

To implement the keyboard, we utilized FPGA registers, The memory location for the keyboard is on 0xFF200100. Thus, we created a variable KB pointer indicating the memory address of 0xFF200100. Moreover, we pushed 3 registers on to the board and therefore to trigger to manage and read the PS2 data by checking the R1, that has been previously stored with the content of KB, If there’s input, we write the data at the corresponding x,y coordinate and if x reaches the end, we change to the next line. No particular challenge is faced during this part of the lab.