Chiao Lu 204848946

Shen Teng 104758168

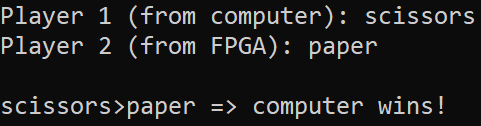
Yuhai Li 104844760

Project 3 Report

# Introduction

Throughout this quarter, we have synthesized various digital designs onto the FPGA board. In this lab, we synthesize a processor (called “a MicroBlaze”) onto the FPGA, which will allow us to execute C code on our Virtex5 boards. This will dramatically improve development time while still retaining some of the advantages of an FPGA.

# Results/Screenshot



As we can see from the screenshot above, player 1 (computer) is scissors and player 2 (FGPA) is paper, and the computer wins, as expected.

# Code Snippets

## Part 2a

This is the essential code of our main function:

|  |
| --- |
| a = getInt(); b = getInt(); printf("%d\r\n",a\*b); int prod=a\*b; if((prod)-100>0){  print("turn on led\r\n");  turnOnLED(); } |

What happens above is simple. We call function getInt() twice to read user’s input from the keyboard. After that we check if the product of the two input number is greater than 100. If so, we turn on the LED light by calling turnOnLed().

Here is the function definition of turnOnLed().

|  |
| --- |
| int turnOnLED() {  int Status = XGpio\_Initialize( & GpioOutput, XPAR\_LEDS\_8BIT\_DEVICE\_ID);  if (Status != XST\_SUCCESS) {  return XST\_FAILURE;  }  XGpio\_SetDataDirection( & GpioOutput, 1, 0x0);  Xuint32 mydata = 3;  XGpio\_DiscreteWrite( & GpioOutput, 1, mydata);  return 0;  } |

First, we initialize GpioOutput by calling XGpio\_SetDataDirection(). Then, we light up the LED by calling XGpio\_DiscreteWrite().

## Part 2b

Reading input from keyboard is just the same as Part 2a. Reading from the keypad is a bit more complicated. To check whether a button is pressed, we need to first set the column where the button belongs to to low, then scan each row. The row which the pressed button belongs to will return an active low.

|  |
| --- |
| XGpio\_SetDataDirection(&GpioOutput, 1, 14<<4); XGpio\_DiscreteWrite(&GpioOutput,1,14<<4); XGpio\_SetDataDirection(&GpioInput, 1, 0xF); p1=XGpio\_DiscreteRead(&GpioInput,1); |

The code above shows how we read input from the keyboard (take column 1 as example). In the above code, we set column 1 to low and read each row. We can then check which bits in p1 is 0 to locate the pressed button.

# Problems Encountered

* When getting the user’s input from the serial line, we cannot just use scanf() directly. Instead, we need to use getchar() to get the user input one character at a time.
* When reading from the keypad, we need to read the schematics on the board and use the UCF file to correctly obtain the user input. We also needed the library xparameters.h and the labels on the board.
* We kept getting errors when trying to export the design to SDK because of timing error. We fix this error by disabling the time error in preference.

# Conclusion

By doing this lab, we gain some experience about how to input/output FPGA GPIO pins using Microblaze C code. This helps us prepare for the final project.