Megan Doherty – 3500734

Jacob Majcan – 3503476

ECE3221 – Lab 1

October 6, 2017

**ECE3221 Lab 1: Switches, LEDS, and the Hex Keypad**

**Introduction**

In Laboratory 1 for ECE3221, an exploration of using the NIOS-II processor along with the Altera Monitor Program in order to develop programs written binary machine and machine assembly languages. Laboratory 1 is the first part of two in using the hexadecimal keypad.

In this experiment, the processor is identified as having a 16-bit input port connected to switches and a 16-bit output port connected to LEDS. The deliverable of Laboratory 1 is t configure the input/output port using assembly and binary machine language to explore how the buttons on the keypad in lab may be interfaced to a computer.

A pre-assembled program is given and expected to be modified in order to reach the expected outcome of configuring the buttons on the keypad to interface like a computer.

**In-Lab Results**

**Single Step – FETCH AND EXECUTE**

The lab Manuel instructs on only executing the first line of the pre-assembled program. By clicking the single step action and executing the first line of code, ori r10, r0, 0x8850, the register r10 is changed. This is because the first instruction is loading the address of 0x8850 into register 10. R10 = 0x8850

The next line of code, ori r11, r0, 0x8880 is executed. This line in the program is loading the address 0x8880 into register 11. Therefore, r11 is changed after this execution and equals 0x8880. R11 = 0x8880

All switches, SW7…SW0, are set to some non-zero value. The following results occur after this modification.

The following line of code is now executed, ldhio r2, 0(r10). No LEDS have appeared on and there is no change in the registers that can be noted after the execution of this code in the program.

The following line of code is executed: sthio r2, 0(r11). The execution of this line results in all the LEDS being loaded and stored, meaning that they are on. There is no change in the registers at the moment.

Next, execute the line of code in the program that loops back to the top of the program (br top). Although it was not noted that any registers changed, the program counter underwent change. Pc = 8, and looped through the program again. The program counter increments itself until it goes back to its original value. By branching back to the top of the program, we loop through and re-execute the code.

Lastly in this section of the lab, it asks for the program to be executed at full speed instead of stepping through the code line by line as the questions above asked. It was noted that the program executed rapidly and the switches equalled 0 at the end of the program and the LEDS were off.

**Using the Hex Keypad**

**Part A:**

The resulting changes affected LEDS 3 through 0, the remaining LEDS on the board are off.

**Part B:**

After compiling and running changes, the LED at PORT PD is now enabled.

**Part C:**

After writing to the PD port control register, output pins are on 0 to 3 and input ins on bit 4 to 7. The resulting changes affecting the switches disabled an enabled row on the switches located on the keypad. Now pressing buttons on the keypad would allow certain LEDS to turn on if the switches are configured correctly.

**Part D:**

After modifying the program correctly, as the lab manual stated the 8-bits read from the keypad now go down to the 7-segment hex displays located on the NIOS II board. Answers are now visible in hexadecimal.

**Part E:**

The resistance of each pull-up resistor is 2.18 kOhms. When the resistor was touched, an LED came on. The importance of having a pullup resistor is when you have switches in series it keeps the current from flowing back into switches. Without a pullup resistor and a button is pressed, the current will flow directly into another wire and cause issues within your circuit.

**Part F:**

Upon inspection of the system it can be noted that the state of the switches depends on the buttons pressed in order to show differences between the rows. I.e show rows are disabled when clicking a specific button. It can be noted that pressing buttons down in a column displays different values.

Values achieved from pressing specific buttons can be noted below in Table 1.

|  |  |  |
| --- | --- | --- |
| Binary Scan Code | Hex Scan Code | Key Pressed |
| 11010111 | D7 | 0 |
| 11101110 | EE | 1 |
| 11011110 | DE | 2 |
| 10111110 | BE | 3 |
| 11101101 | ED | 4 |
| 11011101 | DD | 5 |
| 10111101 | BD | 6 |
| 11101011 | EB | 7 |
| 11011011 | DB | 8 |
| 10111011 | BB | 9 |
| 01111110 | 7E | A |
| 01111101 | 7D | B |
| 01111011 | 7B | C |
| 01110111 | 77 | D |
| 11100111 | E7 | E |
| 10110111 | B7 | F |

Table 1. Results of the hex scan codes and binary scan codes from pressing specific buttons on the hex key pad.

**Conclusion**

The aim of Laboratory 1 for ECE3221 was to explore the usage of the BIOS-II processor connected with the Altera Monitor program and the UNB designed keypad. A binary machine and assembly language program was successfully developed in order to view hexadecimal results on the hexadecimal display from the keypad. The expected outcome was reached of configuring the buttons on the keypad to interface like a computer.

**Please see code developed in lab below.**

/\*\*\*

CMPE3221 LAB#1 - MY FIRST NIOS-II PROGRAM

-----------------------------------------------

DATE: October 6, 2017 NAME: Megan Doherty, Jacob Majcan

-----------------------------------------------

This program reads a value from 16 switches

and writes the value to 16 LEDs

-----------------------------------------------

PORT MAP

0x00008850 = 16-bit input port = SWITCHES

0x00008880 = 16-bit output port = LEDS

-----------------------------------------------

\*\*\*/

.global \_start

\_start:

ori r13, r0, 0x8890 #r13 = address of LED 8

#-----------------------------

# -- ADDING HEX PAD --------

ori r12, r0, 0x8930 #r12 = address PD

ori r3, r0, 0x0F #r3 = iiii 0000

stbio r3, 4(r12) #Setup PD Control

# -- Adding HEX Display ---

ori r9, r0, 0x88B0 # r9 = address of the HEX Control

ori r8, r0, 0x88A0 # r8 = address of HEX Display

ori r10,r0,0x8850 # r10 = address of switches

ori r11,r0,0x8880 # r11 = address of red LEDs

READ\_SW: # Where the keypad will read the value from the switches

# Part 2a:

# -- Initialiazing LED40 ---

and r4, r4, r0 # clearing r4, r4 = 0x0

ori r4, r4, 0x0800 # r4 = 0x0800, this is the value for LED40

sthio r4, (r13) # writing r4 to LED40

and r5, r5, r0 # clearing r4, r4 = 0x0

ori r5, r5, 0x0103 # r5 = 0x0103, this is the value to

# turn on for HEX0 and HEX1

sthio r5, (r9) # writing r5 to HEX0 and HEX1

top:

ldwio r3, (r10) # r3 is value of the switches

# r10 is already the address of the switches

andi r3, r3, 0x0F # r3 = 4 switches 3...0

stbio r3, (r12) # Sending switches to the keypad rows

ldbio r3, (r12) # read keypad

andi r3, r3, 0xFF # isolate bits 0...7

stwio r3, (r11) # storing binary value into the leds

sthio r3, (r8) # Storing HEX Values into Hex Display

#ldhio r2,0(r10) # read the 16 switches

#sthio r2,0(r11) # write to the 16 LEDs

br top # repeat forever

# -----------------------------------------------