ECE 3720

Microcomputer Interfacing Laboratory

Section 005

Aaron Bruner

Date Performed: 02 / 04 / 2021

Lab 4

ABSTRACT:

A lab designed to demonstrate how to program the PIC32 microcontroller to handle keypad input and relate that keypad input to LED output. We will be using a mask technique to determine exactly what output should be displayed for every input. The purpose of the lab is to represent the number pressed on the keypad in binary using four LEDs. Similar to the previous labs, this will also enhance skills in reading documentations for new and existing parts.

**INTRODUCTION:**

The goal of lab 4 is to program our microcontroller to take whatever value was pressed on the keypad and relate that to a binary representation in StaticIO. We will be comparing the binary input with our mask to determine which key was pressed. Thus, the inputs (columns) are scanned and if one reads low, that means the button corresponding to the row and column is pressed. The result after pressing the key will remain on the LED display until the next key is pressed. The keypad diagram demonstrates using pull up resistors but for this lab we will be u sing pull down resistors. To perform our lab, we required a variety of materials including:

* A breadboard
* Wires
* Keypad
* PIC32MX150F128D
* Analog Discovery 2 (AD2)
* Digilent WaveForms
* MPLAB Snap Debugger
* MPLAB X IDE (Programming Software)

The AD2 is going to be a partial power source and our output LEDs. Instead of using actual LEDs we will use the WaveForms software and simulate the LED output. The partial power source will power the PIC32 chip and be a 3V reference source. However, it will not power the MPLAB Snap Debugger. It will be powered off of a Micro USB cord.

**EXPERIMENTAL PROCEDURES:**

We must assume that the individual reproducing this lab has already setup their breadboard in a manner that their PIC32 chip can be programmed using the MPLAB X IDE software.

To begin the lab, we will observe the circuit diagram figure 1 to show exactly how we need to wire our breadboard. The circuit shown is a rather simple one. We will be connecting the four LED outputs to pins 2, 3, 9 and 10 which correspond to A0-A3. Next, we will be connecting the right four pins of our keypad to pins 4-7 which correspond to B0-B3. These pins will be our input values to the keypad. Next, we will connect the left four pins of our keypad to pins 23-26 which corresponds to B12-B15. These pins will be our output values from the keypad.

The most complex part of this lab is the code. Wiring up the circuit as shown takes very little effort. To begin the overview of the code we first copy the mask and key\_mask values provided in the PowerPoint. These two arrays are how we are going to determine based on the input value from our B0-B3 pins we will determine which output should be displayed on the key\_mask. Next, we need to set all A and B pins to digital since analog would not benefit us in any way. As mentioned earlier, the keypad is intended to be used with pull up resistors but instead we’re going to be using pull down resistors for the keypad inputs. We simply enable the pull down resistors for the B pins using CNPDB = 0xF which enables the pull down resistors for B0-B3.

Next, we need to setup our pins for input and outputs. As mentioned in the PowerPoint, the B0-B3 pins will be our input pins and B12-B15 will be output pins. Thus, utilizing the TRISB operation we can set each pin to either input or output with a simple 1 or 0. It’s also worth noting that we could have used a simpler method of TRISB = 0xF which would have set B0-B3 to 1 and B4-B15 to 0. We don’t care about B4-B11 since we’re not using them. Lastly, we need to set our LED output pins to outputs with TRISAbits or like previously mentioned, TRISA = 0x0.

The while loop is the core of how this lab works. It begins by iterating through every option on the keypad checking to see if any of them are being pressed. As soon as one of them is pressed the internal if statement is triggered. As a result, we set the output (LEDs) to be whatever key was being pressed using the key\_mask option.

The last line of code does nothing but keep representing the last successful output operation. If three was pressed it will continue to display three in binary until the MC is powered off.

**RESULTS and DISCUSSION:**

Once the microcontroller was programmed it was clear that things were working when we started pressing values on the keypad and their binary representation was displayed on LEDs. Simply iterating through 0-15 we can see that every key on the keypad is functional and represents the correct output.

One problem during this experiment was the output was being instantly removed as soon as a key was pressed. This turned out to be an incorrect line of code which reset the output ever time the while loop would restart. After removing the line of code everything worked just as expected.

This week we demonstrated how to utilize a keypad and key masks. This lab is a good learning experience because a keypad is the primary way of entering integer values into computers. A great example of a keypad being used in the real work is a pocket calculator. It takes the input value and represents it on the screen when the key is pressed.

**CONCLUSION:**

To conclude, lab 4 taught us many useful techniques for using keypad inputs on the PIC32 and demonstrating these input values on LEDs. This lab also taught us more complex ways to analyze problems and how to solve them.

Diagram, schematic

Description automatically generated**FIGURES AND TABLES:**

**Figure 1: Wiring for lab 3 (Pin connections described in experimental procedures)**

**CODE:**

#include <xc.h>

void main(){

// Declare integers to be used in program

int output;

int i;

// Masking to connect our bianry inputs with the binary output

unsigned int mask[16] = { 0x8008, 0x8004, 0x8002, 0x8001,

0x4008, 0x4004, 0x4002, 0x4001,

0x2008, 0x2004, 0x2002, 0x2001,

0x1008, 0x1004, 0x1002, 0x1001};

unsigned char key\_mask[16] = {1, 2, 3, 10,

4, 5, 6, 11,

7, 8, 9, 12,

14, 0, 15, 13};

// Set all A & B pins to digital

ANSELA = 0;

ANSELB = 0;

// Pull down resistor for keypad input pins

CNPDB = 0xF; // 0xF = 1111

// Keypad Input Pins ( B0-B3 )

TRISBbits.TRISB0 = 1;

TRISBbits.TRISB1 = 1;

TRISBbits.TRISB2 = 1;

TRISBbits.TRISB3 = 1;

// Keypad Output Pins ( B12-B15 )

TRISBbits.TRISB12 = 0;

TRISBbits.TRISB13 = 0;

TRISBbits.TRISB14 = 0;

TRISBbits.TRISB15 = 0;

// LED Output Pins

TRISAbits.TRISA0 = 0;

TRISAbits.TRISA1 = 0;

TRISAbits.TRISA2 = 0;

TRISAbits.TRISA3 = 0;

while(1) {

for(i = 0; i < 16; i++) { // Checking every input on the keypad

// Set Keypad Output To High

LATB = mask[i] & 0xF000;

if((PORTB & 0xF) == (mask[i] & 0xF)) { // If our input equals the

// mask then set the

// output to the key\_mask

// index value

output = key\_mask[i];

}

LATA = output; // If we're not pressing a button just keep

// representing our old output value on the LEDs

}

}

}