Christopher Brant
C19816588
Due on 10/5/2017

ECE 3270 Microcontroller Interfacing Lab
Lab 4: Keypad Interfacing

Abstract

The purpose of this lab is to assemble a basic keypad circuit utilizing a keypad and our PIC32MX150F128D microcontroller to controller 4 LEDs so that they light up with the corresponding binary value of 0-15 depending on which button on the keypad is pressed.

Introduction

In this experiment, we used a 4x4 keypad to control a binary numerical output on displayed on 4 LEDs, which would show 0-15 depending on the button on the keypad that gets pressed. The circuit is wired as in Figure 1, and the code is written exactly as in Figure 2, and those components allow this circuit to function as desired. The keypad was wired with rows as outputs and columns as inputs so that when a button is pressed, you a 0 will be seen for the correct column and row value. Pull up resistors connected to the column inputs are necessary as well.

Experimental Procedures

- First the microcontroller is wired to both ground at DIP pins 39 and 40 and power at DIP pin 1, as well as power is wired from into DIP pin 3.
- Then the circuit is wired to have the connections seen ion Figure 1, where the keypad pins 0-3 connect to PortC bits 1-4, and keypad pins 4-7 connect to PortC bits 0 and 5-7.
- As well, we utilize pull up resistors for the pins that are designated as the columns, and those bit values are set in the code in Figure 2.
- Lastly, the code is written, and the code for this lab is included as Figure 2, and then the code is compiled, downloaded to the microcontroller, and then programmed onto it to run continuously.

Results

It can be observed from this lab that the resources available here at Clemson University are sub-par for this keypad interfacing lab, as well as the difficulty of making a keypad correctly work without a designated schematic is very high. It was also observed that the pull up resistors are important for a keypad, yet we utilized the CNPUx register to set the pull up status of the necessary pins instead of directly wiring in pull up resistors. No tabulated values were collected or are necessary.

Discussion

It was observed that the code necessary for reading in a single button press is far simpler than a multi-press read, yet it is important to understand how the code must be structured to run through the row values and check for columns to have values of 0 coming into the microcontroller, and that determines the key value set. It was observed that the basic keypad design is very straightforward, yet can be extremely tricky to make work in situations where there are unknown hardware faults.

Christopher Brant
C19816588
Due on 10/5/2017

Conclusions

Conclusions from this experiment are that creating a circuit to showcase the features of a keypad can be very tricky, yet the principles are very simple and understandable, as long as the correct masks and key values are used for your given keypad. As well as this, an explicit schematic of the given keypad is very important as it was observed that it was extremely difficult to solve the issues of this lab without one. Lastly, the process of using an array of keys and their array values to determine key presses is no longer a novel idea, yet a very efficient technique and seems like a technique that is likely still widely used today.

Figures and Tables

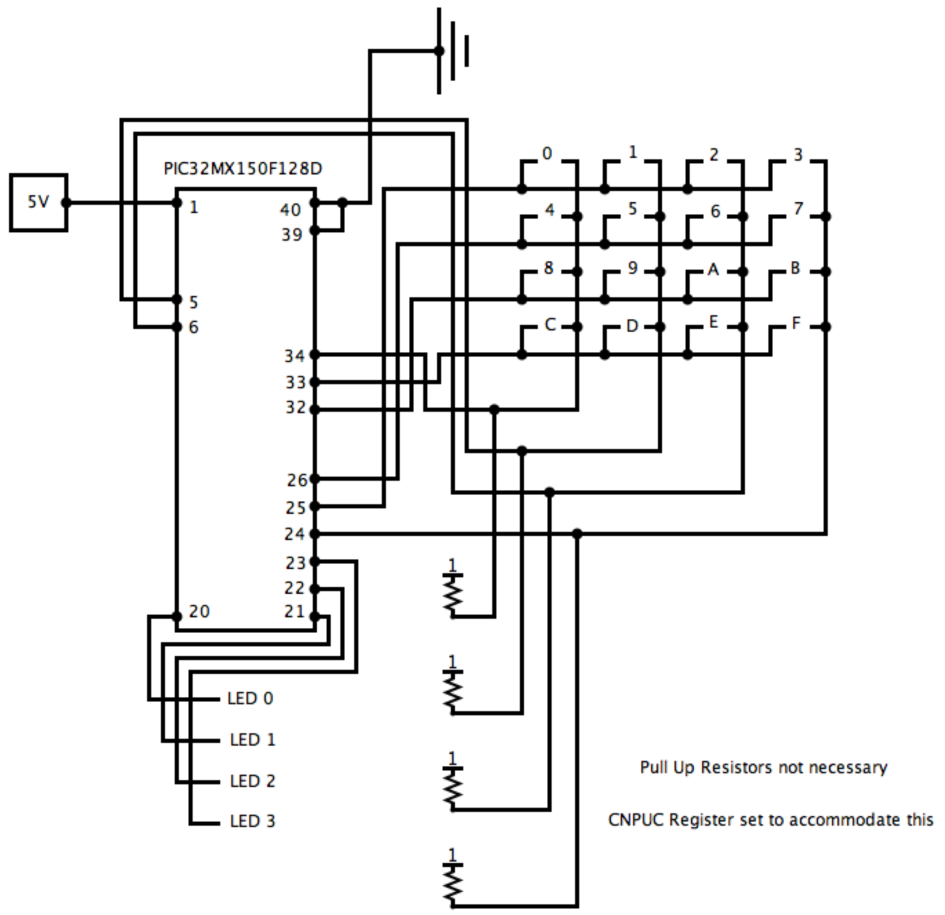
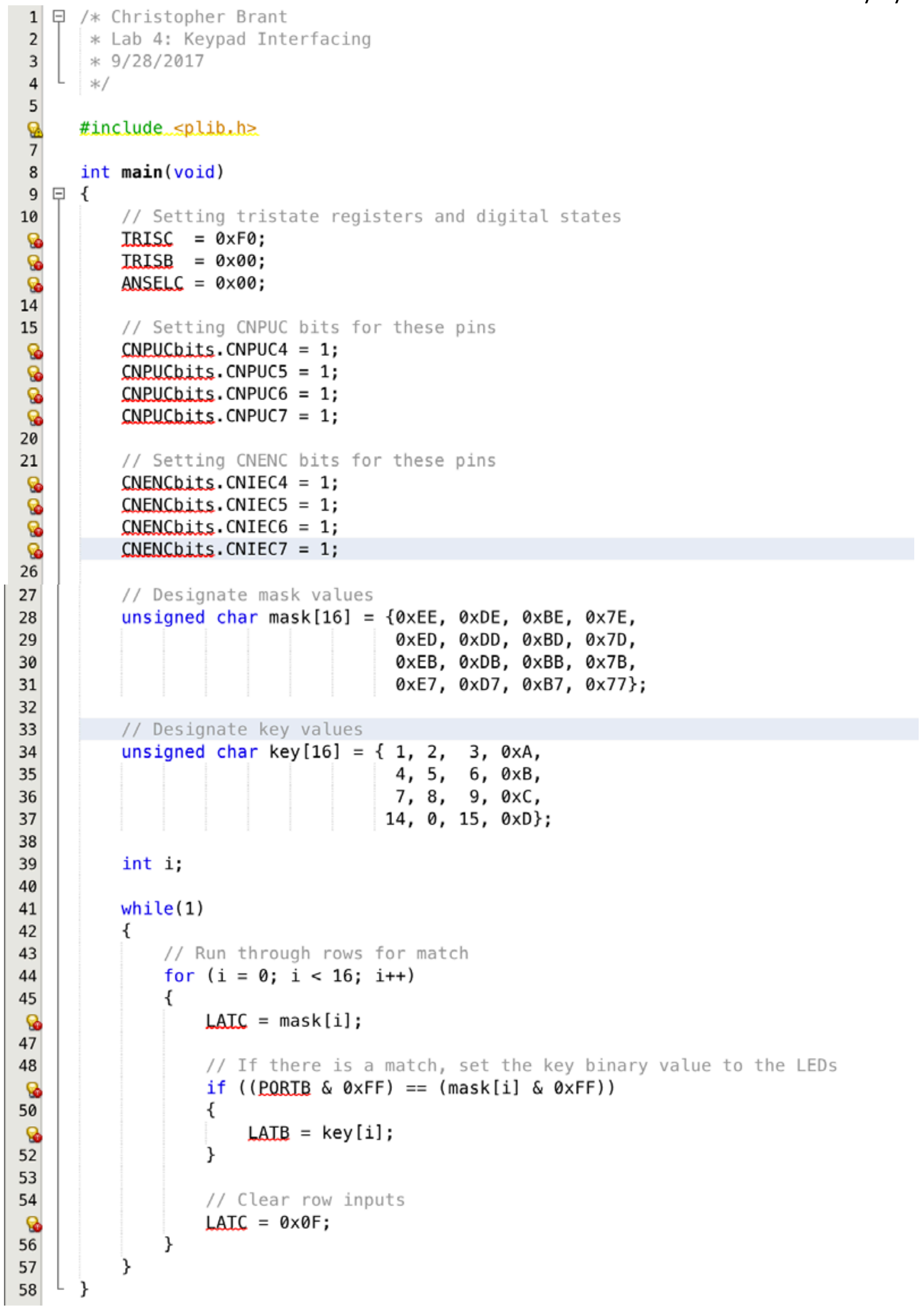


Figure 1: Wiring schematic for Keypad with PIC32MX150F128D microcontroller

Christopher Brant
C19816588
Due on 10/5/2017



```c
/* Christopher Brant
 * Lab 4: Keypad Interfacing
 * 9/28/2017
 */

#include <plib.h>

int main(void)
{
    // Setting tristate registers and digital states
    TRISC  = 0xF0;
    TRISB  = 0x00;
    ANSELC = 0x00;

    // Setting CNPUC bits for these pins
    CNPUCbits.CNPUC4 = 1;
    CNPUCbits.CNPUC5 = 1;
    CNPUCbits.CNPUC6 = 1;
    CNPUCbits.CNPUC7 = 1;

    // Setting CNENC bits for these pins
    CNENCbits.CNIEC4 = 1;
    CNENCbits.CNIEC5 = 1;
    CNENCbits.CNIEC6 = 1;
    CNENCbits.CNIEC7 = 1;

    // Designate mask values
    unsigned char mask[16] = {0xEE, 0xDE, 0xBE, 0x7E,
                              0xED, 0xDD, 0xBD, 0x7D,
                              0xEB, 0xDB, 0xBB, 0x7B,
                              0xE7, 0xD7, 0xB7, 0x77};

    // Designate key values
    unsigned char key[16] = { 1, 2,  3, 0xA,
                              4, 5,  6, 0xB,
                              7, 8,  9, 0xC,
                             14, 0, 15, 0xD};

    int i;

    while(1)
    {
        // Run through rows for match
        for (i = 0; i < 16; i++)
        {
            LATC = mask[i];

            // If there is a match, set the key binary value to the LEDs
            if ((PORTB & 0xFF) == (mask[i] & 0xFF))
            {
                LATB = key[i];
            }

            // Clear row inputs
            LATC = 0x0F;
        }
    }
}
```

Figure 2: Code for Single Button Press Reading Keypad