Lab 4: Keypad Interfacing

Nico Whiteford

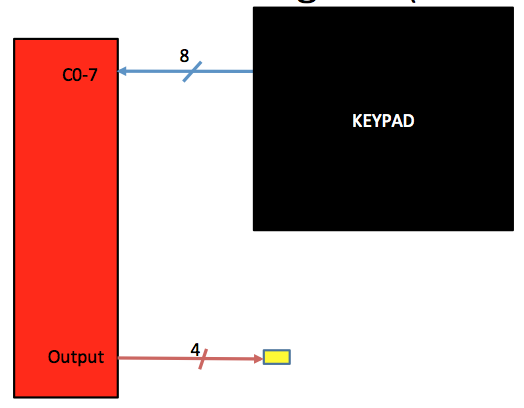
ECE 3720 – 007

10/9/11

Lab Partners: Steve Kasir, Josh Apter

*Abstract* – The purpose of this lab was to wire and code a circuit with a keypad in which when a specific button is pushed on the keypad the microcontroller will light up the corresponding number in binary on the LEDs. This lab should give us ideas on how pull up resistors work, now to use the CNPUEx register correctly and how to interface with a keypad.

*Procedure –* to start off this lab a crude block diagram provided by the TA was observed as shown below:



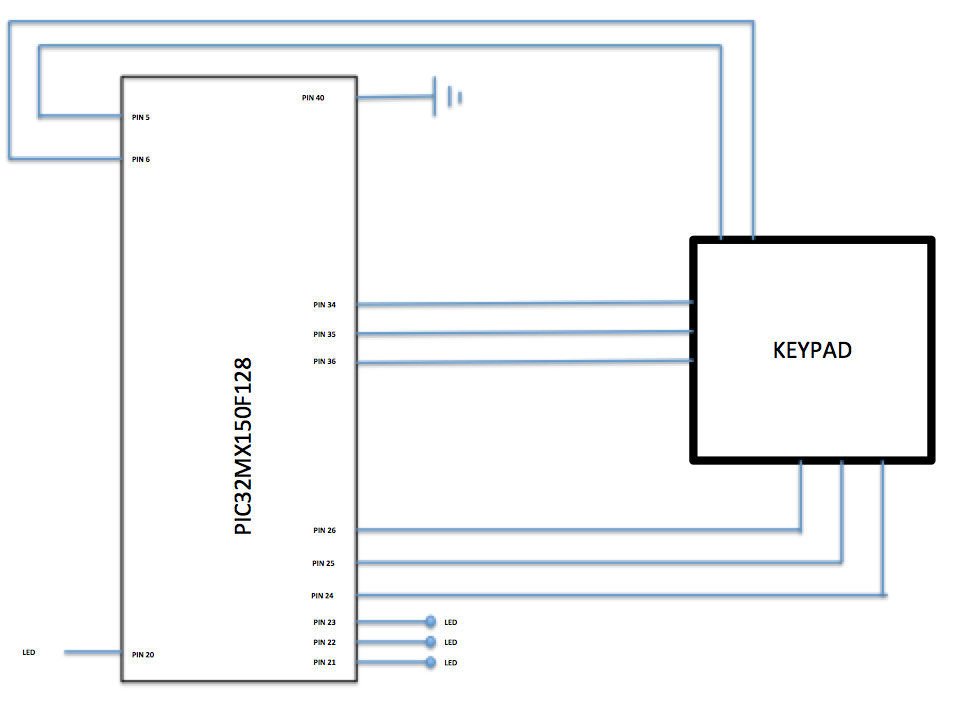
* Test LED’s to see which one work
* Wire the keypad pins to the corresponding pins on the microcontroller (keypad pin 0 was the far right pin and keypad pin 7 was far left pin)
* Ensure to ground pin 40 and 39
* Connect pins 20-23 to the corresponding LED
* Write program for 4 inputs and 8 outputs and execute

*Results –* the results acquired were favorable, when the keypad was pressed the corresponding binary representation appeared in the form of LED lights on the NI ELVIS.

*Discussion* – the actual circuit was not too complicated to wire. We just had to make sure that the correct pins in the microcontroller were connected to the corresponding pins in the keypad. Since this was relatively straightforward there was no issue here. Once the correct pins were grounded the code was written. The motif of the code was to create an array to represent the keypad and a corresponding mask array. The pull up component of the code required a line to set it on and a line to represent the actual pull up enabler. The rest of the set up component set the keypad outputs to zero, keypad inputs to 1, and the outputs to the corresponding LEDs (also 0 so that they were cleared until keypad was pressed). The loop utilized a for loop which anded the mask array and a constant hex variable in order to set the outputs to high. Then the loop checked to see if the pressed keypad button matched the binary bit. If it matched it output the corresponding binary representation. Initially we had an issue with the code because we set the high inputs to the wrong pins because we forgot that the pins were set in reverse. Once the code was fixed the desired output was acquired. Since the wires weren’t very good we had to hold down on the wires to get all of the LEDs to respond when the keypad was pressed.

*Conclusion –* in conclusion, our group was able to properly wire and code a keypad circuit that gave a corresponding binary representation of the value pressed on the keypad. We were able to better understand the ability of a pull-up enabler and how to use loops to match corresponding cells in an array.

*Circuit Schematic*



#include <plib.h>

void main(void) {

char out;

int i;

unsigned char mask[16] =

{

//0 1 2 3

0xEE, 0xDE, 0xBE, 0x7E, //0

0xED, 0xDD, 0xBD, 0x7D, //1

0xEB, 0xDB, 0xBB, 0x7B, //2

0xE7, 0xD7, 0xB7, 0x77 //3

};

unsigned char key[16] =

{// 0 1 2 3

1, 2, 3, 0xA, //0

4, 5, 6, 0xB, //1

7, 8, 9, 0xC, //2

14, 0, 15, 0xD //3

};

CNPUC = 0xF0; //pull up enabler pin

CNENC = 0xF0; //sets pin as on

//keypad output pins

TRISCbits.TRISC0 = 0;

TRISCbits.TRISC1 = 0;

TRISCbits.TRISC2 = 0;

TRISCbits.TRISC3 = 0;

//keypad input pins

TRISCbits.TRISC4 = 1;

TRISCbits.TRISC5 = 1;

TRISCbits.TRISC6 = 1;

TRISCbits.TRISC7 = 1;

//outputs to led

TRISBbits.TRISB0 = 0;

TRISBbits.TRISB1 = 0;

TRISBbits.TRISB2 = 0;

TRISBbits.TRISB3 = 0;

while(1){

for(out = 0x0, i = 0; i < 16; i++){

//sets outputs to high

LATC = mask[i] & 0x0F;

//checks for a match

if((PORTC & 0xF0) == (mask[i] & 0xF0)){

out = key[i];

}

else{

out = 0x0; //reset

}

LATB = out; //if condition is false output

}

}

}