CAMOSUN COLLEGE

ELECTRONICS DEPTARTMENT

ECET 165 - LAB 7

Matrix Keyboard

Create a E165L07XXkeypad project using the MPLAB development system where XX is your intials. Your source code shall be well documented and saved in the file E165L07XXkeypadTest.c
Create:
keypad18f.c
keypad18f.h

Objective

Write a program that will scan a matrix keypad and decode which switch is pressed. The output will be displayed on the LCD display.

Write the C code to interface the 16 key keypad to your development board. Your software should map the keys for the digits (1234567890) a ***clear*** key and a ***return*** key. You may use the remaining keys as you wish. The ASCII value for the key should be displayed on the lcd display. Note that at the end of the line, wrap the keypresses to the next line. Pushing clear will clear the whole display and postion the cursor at the top left postion.

Pressing the return key will move the cursor to the next line. If it is already on the second line, the cursor should move to the top line. Clear the new line and place the cursor start of that line.

Add a 30 ms delay to the begining of the keypad function to debouce the key. Use the built in functions for your delay.

“E165L07AHkeypadTest.c”

```
/*****************************************************************
*************
 * Lab 7 - Matrix Keyboard
 * ECET165 Embedded Micro-controllers
 * E165L07AHkeypadTest.c
 * CREATED 28 Feb 2023
 * UPDATED ***
 * v1.0
 * BY Aaron Huinink
 * Tests functionality of the keypad library

*****************************************************************
************/
// ============================ INCLUDES/DEFINES
==================================//
#include <xc.h>
#include "keypad18f.h"
#include "C:\Users\a_hui\OneDrive - Camosun
College\term2\ecet165_embedded_mc\labs\lab6\lcd18f.h"

#include "config_keypad.h"

// ============================ MAIN
==================================//
void main(void){

    // ============================ SETUP
==================================//

    // set up port c for debugging
    TRISC = 0x0; // portc output

    ANSELC = 0x0; // portc digital

    WPUC = 0x00;

    LATC = 0x0;

    // position variable to keep track of cursor position
    unsigned char pos = 0x0;

    // initialize lcd
    LCDinit();



    // ============================ MAIN LOOP
==================================//

    while(1){
        // manage cursor position @ end of lines
        if((pos > 0x0F) & (pos < 0x40)){
            pos = 0x40;
            LCDgoto(pos);
        }
```

```
        if (pos > 0x4F){
            pos = 0x00;
            LCDgoto(pos);
        }

        // get char from keypad
        char key = keyScan();

        // manage key inputs
        switch (key){
            case '*':
                pos = LCDreturn(pos);
                break;

            case '#':
                LCD_CLEAR;
                pos=0x0;
                break;

            default:
                LCDprintc(key);
                pos++;
                break;
        }

        // for debugging
        LATC++;



        }
    }
```

“keypad18f.h”

```
/*****************************************************************
*************
 * Lab 7 - Matrix Keyboard
 * ECET165 Embedded Micro-controllers
 * keypad18f.h
 * CREATED 28 Feb 2023
 * UPDATED ***
 * v1.0
 * BY Aaron Huinink
 * Provides 4x4 matrix keypad functionality with a PIC18F uC.
 * Keypad is wired to port f.

*****************************************************************
************/

#ifndef KEYPAD18F_H
#define    KEYPAD18F_H

#ifdef     __cplusplus
extern "C" {
#endif

// ============================ INCLUDES/DEFINES
================================//

#include <xc.h>

#ifndef _XTAL_FREQ
#define _XTAL_FREQ 64000000
#endif

#define KEY_LAT LATF
#define KEY_PORT PORTF
#define KEY_ANSEL ANSELF
#define KEY_TRIS TRISF
#define KEY_WPU WPUF

#define KEY_PORTEN KEY_ANSEL = 0x00; KEY_TRIS = 0xF0; KEY_WPU =
0xF0; __delay_us(1);

// ============================ FUNCTION PROTOTYPES
========================== //

// ----- keyScan ----- //
/*
* Scans the column keys and returns the column and row values of the
pressed key
* ARGS: (void)
* RETURNS: [key<unsigned char> : (column | row)]
*/
extern char keyScan();


#ifdef     __cplusplus
}
```

```
#endif

#endif      /* KEYPAD18F_H */
```

“keypad18f.c”

```
/*****************************************************************
*************
 * Lab 7 - Matrix Keyboard
 * ECET165 Embedded Micro-controllers
 * keypad18f.c
 * CREATED 27 Feb 2023
 * UPDATED ***
 * v1.0
 * BY Aaron Huinink
 * Provides 4x4 matrix keypad functionality with a PIC18F uC.

*****************************************************************
************/

// ============================ INCLUDES/DEFINES
================================//

#include <xc.h>
#include "keypad18f.h"

// ============================ PRAGMA CONFIG
================================//

// ============================ FUNCTION DEFINITIONS
========================== //

// ----- keyScan ----- //
/*
* Scans the column keys and returns the column and row values of the
pressed key
* ARGS: (void)
* RETURNS: [key<unsigned char> : {column, row}]
*/
char keyScan(){

    unsigned char colshift = 0x00;  // shift variable for checking
columns
    unsigned char cols = 0x0;     // store row pin input
    unsigned char rowshift = 0x00; // shift variable for checking
rows

    // char lookup table
    char lookup[4][4] = {
        {'1', '2', '3', 'A'},
        {'4', '5', '6', 'B'},
        {'7', '8', '9', 'C'},
        {'*', '0', '#', 'D'}
    };

    KEY_PORTEN;  // enable the keypad port
    KEY_LAT = 0x00; // turn on column pins pins

    while(KEY_PORT == 0xF0); // read col pins and wait for a key
press
```

```
    __delay_ms(30); // debounce


    while(!(cols)){ // while there's no reading on the col pins
        if(rowshift > 0x3){ // ensure rowshift is in range
            rowshift = 0x0;
        }

        KEY_LAT = ~(0x08>>rowshift);  // cycle a 0 through the row
pins

        __delay_us(1);

        cols = (~(KEY_PORT)& 0xF0);  // read the 1s complement of
the column pins
        rowshift++;  // increment colshift by one to cycle through
next pin

    }

    rowshift--;  // decrement to remove additional column shift

    // wait for key release
    KEY_LAT = 0x00; // turn on row pins
    __delay_us(1);

    while(KEY_PORT != 0xF0); // wait for a 0xF reading on the col
pins

    KEY_LAT = 0xF0;  // turn off column pins
    __delay_ms(15); // debounce release

    // count the column pins to find the col number
    while(!(cols<<colshift & 0x80)){  // while MSB of column pins
not 1
        colshift++;                     // add one to the colshift
variable and shift again
    }

    return lookup[rowshift][colshift]; // return the keypad char
}
```