A4 Assignment

Group members:

Jacob Rowe U12325180

Jegert Merkaj U27189696

Michael Strickland U03010237

Xzavier McKenzie U05013794

Problem Description:

The objective of this assignment is to control a 12VDC motor with a PWM output signal using a keypad matrix, controlling the speed from 0 to 100% and displaying the results on an LCD display using the I2C protocol. The microcontroller takes in the number input by the matrix and translates it to what percent the duty cycle is of the PWM. This would control the number of rotations the motor does per second, controlling its speed.

Pseudocode:

Include Libraries.

Define Macros.

Initialize global variables.

Define an array that will map values from the matrix keypad.

Initialize function prototypes.

Void main(){

Initialize Port 3 that will be used to read form matrix keypad.

Disable high impedance mode.

Set Up LCD with predefined functions.

Set up Port 2 for PWM.

Configure TimerB and CC0 and CC1.

Enable interrupts.

Start an infinite loop that detects when a matrix key is pressed. {

When key press detected, get the corresponding char value from the matrix array defined before.

Check if the character is a number.

Convert the asci number to motor speed.

Send the asci the LCD screen.

If char is '*' update the CCR1 value with the motor speed.

}

Return 0;

}

Define previously declared functions and interrupts

Interrupt for CC0 should set P2OUT to 1 and for CC1 it should set P2OUT to 0.

C Code:

```
#include <msp430.h>
#include "LiquidCrystal_I2C.h"
#include <stdio.h>
#include <string.h>
#include <ctype.h>
// We are using P3 for the keypad. The columns are inputs and rows are outputs.
#define keyport P3OUT
// Column pins are P3.0, P3.1, P3.2, and P3.3.
#define COL1 (P3IN & 0x10) // Input pins for the columns.
#define COL2 (P3IN & 0x20)
#define COL3 (P3IN & 0x40)
#define COL4 (P3IN & 0x80)
unsigned char key_press; unsigned i, k, key = 0;
unsigned int motor_speed;
unsigned int length = 0;
// ' ' is empty, index 0 is dummy; does not return anything.
unsigned char Key_Val[] = {' ','1','2','3','A','4','5','6','B','7','8','9','C','*','0','#','D'};
```

```
// User-defined functions prototypes:
unsigned char get_key(void);
void DelayMs(unsigned int Ms);

void main(void) {

WDTCTL = WDTPW | WDTHOLD; // Stop WDT
unsigned int count = 0;    // Count set to 0 initially.
unsigned char Value = 0;

P3DIR = 0x0F;     // Output registers for the ROWS. Pins: P3.4, P3.5, P3.6, P3.7.

// Pull-up resistor initialization:
P3REN = 0xFF; // These pins will be associated with a resistor type. Should correlate with input pins
(columns).
P3OUT = 0xF0;           // Set P3 with pull-up resistor type

PM5CTL0 &= ~LOCKLPM5;      // Disable high-impedance mode.

// LCD function call and setups:
I2C_Init(0x27);  // 0x27 signifies the slave address (address of the LCD with the I/O expander).
LCD_Setup();

LCD_SetCursor(0, 0);       // Initial position for the cursor at row 1, column 1.




P2DIR |= BIT0;
P2OUT &= ~BIT0;
TB0CTL = TBSSEL__ACLK | MC__UP | TBCLR;

// SET THE MAXIMUM TIMER VALUE HERE !!!
TB0CCR0 = 100;
TB0CCTL0 = CCIE;
TB0CCTL0 &= ~(CCIFG);

TB0CCR1 = 0;
TB0CCTL1 = CCIE;
TB0CCTL1 &= ~(CCIFG);

__enable_interrupt();

// The user types one character or number at a time:
```

```
while(1) {

   while ((count = get_key()) == 0);  // It waits for a key input now. It returns 0 when no key is pressed.

   // It exits the while loop when a key is pressed.

   // An array that maps to what number the key is pressed.
   Value = Key_Val[count];
/*
This is reading a character from Key_Val at index count, and storing that character in value. The count is
whatever the value is returned from the user-defined function get_key. Whatever is returned by the get_key
function is figuring out what keys belong to which character or number in the array. In summary, it is an array
where element at index i is the character we want to associate with that number. Index i, in this case, is
count. So, this Key_Val[] is just an array that maps the return value of the get_key function to a character for
convenience. It is like a vessel that holds the key combos.
*/

   if (isdigit(Value)) {

      unsigned int current_value = (int)Value;
      if (length == 0) {
         LCD_ClearDisplay();
         DelayMs(50);
         LCD_SetCursor(0, 0);
         DelayMs(50);
      }



      motor_speed = (motor_speed * 10) + (current_value - 48);
      LCD_Send(current_value, Rs | LCD_BACKLIGHT);

      length++;
   }
   else if (Value == '*') {
      if (motor_speed != 0) motor_speed--;
      TB0CCR1 = motor_speed;
      motor_speed = 0;
      length = 0;
   }



}  // End of while(1).
```

```
} // End of main.

// User-defined functions:
unsigned char get_key(void)
{
// A number associated with the key they pressed.
k=1;  // k is just a number to increment by 4, so there will be a different return value for every key. Now we start initializing k as 1.

for(i = 0; i < 4; i++) {          // To assign different key presses. Send 0 to bit i.
   keyport = ((0x01 << i) ^ 0xff);  // Shift left is setting the nth bit to 1, then it inverts the bit to set it to 0 and other bits to 1.
/*
Every loop iteration is sending one zero and sending it gets back a zero on one of the pins.
There are four locations it needs to send a zero, and for each of those locations there are four
places it needs to check if it got a zero back.
 In summary: sending 0 to bit 1, 2, 3, 4 and see if getting 0 from different columns.
*/

   if(!COL1) {
      key = k+0;
      while(!COL1);
      DelayMs(50);            // Add a small delay of 0.05s. Every time we check a row, the button debouncing concept is utilized.
      return key;
   }

   if(!COL2) {
      key = k+1;
      while(!COL2);
      DelayMs(50);
      return key;
   }

   if(!COL3) {
      key = k+2;
      while(!COL3);
      DelayMs(50);
      return key;
   }

   if(!COL4) {
      key = k+3;
      while(!COL4);
```

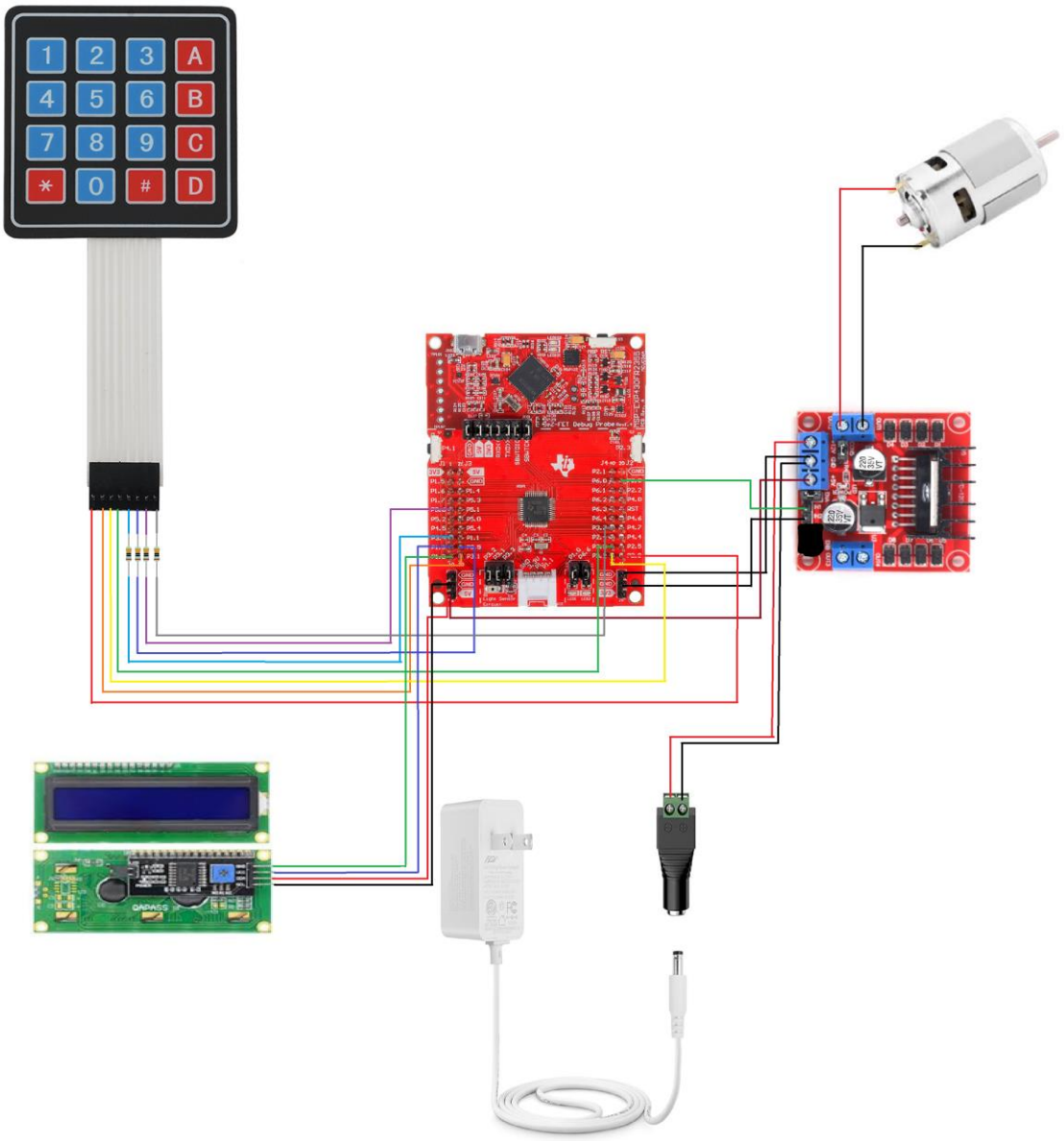
```
    DelayMs(50);
    return key;
  }

k+=4;                // This is k += 4 because we checked four values; we want a different return value for each possible key press.
keyport |= (0x01 << i);   // It is setting the bit it set back to 1. This is to stop sending a 0, in other words to stop checking that row.

} // End of for loop.

return 0; // Return the unsigned char to main.
} // End of UDF get_key.
void DelayMs(unsigned int Ms) { while(Ms) { __delay_cycles(1000); Ms--; } } // End of DelayMs user-defined function.
#pragma vector=TIMER0_B0_VECTOR __interrupt void TMR0() { P2OUT |= BIT0; TB0CCTL0 &= ~(CCIFG); }
#pragma vector=TIMER0_B1_VECTOR __interrupt void TMR1() { P2OUT &= ~BIT0; TB0CCTL1 &= ~(CCIFG); }
```

Wiring Diagram:



Video:

Additional Attachment on Canvas Submission