Lab 6

Reyna, Ricardo

Section 75C9

07/16/2015

- Prelab Questions:

  1. If you were working on another microcontroller and you wanted to add an 8-bit LCD to it, what is the minimum amount of signals required from the microcontroller to get it working?
  A write enable, RS which uses a A0, and 8 data lines where the seventh bit is used for the busy flag.

  2. In this lab our reference range is ideally from 0V to 5V. If the range was form 0v to 2.0625V (a possible internal reference) and 12-bit unsigned mode was used, what is the resolution and what is the digital value for a voltage of 0.37V.
  When encoding the 12-bits we get 4096, we get 0.0005035400390625 of voltage sensitivity if we divide 0v to 2.0625 equally which is also the max resolution. 0.37V is represented as 0x2E4.

  3. Assume you wanted a voltage reference range from 1V to 3V, with a signed 12-bit ADC. What are the voltages if the ADC output is 0xA42 and 0xD37?
  -1.3V and 1.8V.

  4. What is the difference in conversion ranges between 12-bit unsigned and signed conversion modes? List both ranges.
  When unsigned we only get positive voltages and there's an offset to the 0, signed range allows for negatives.

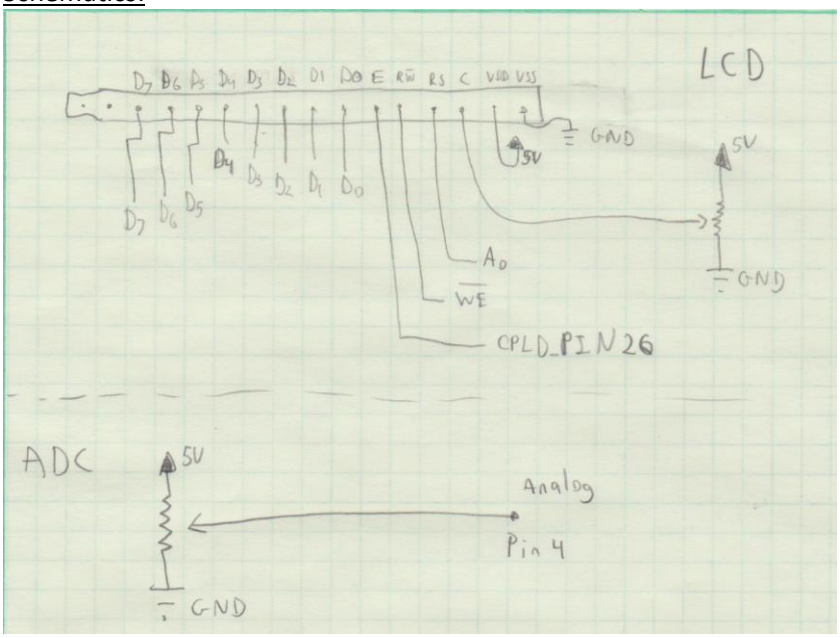
- Problems Encountered:

  The voltmeter isn't as accurate as I wanted it to be. Having the voltmeter read continuously, and going to the previous function after entering "pot mode".

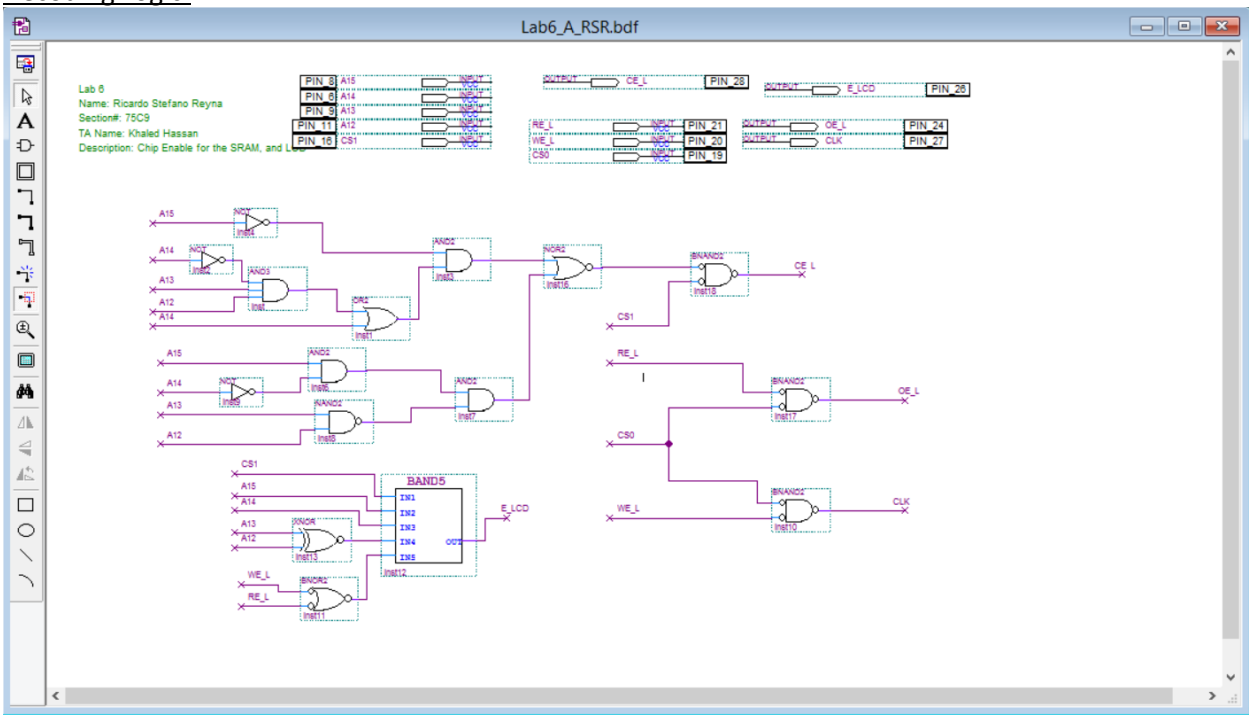
- Future Work/Applications:

  We can use the lcd to output characters, and we can use the ADC to connect other analog peripherals to the board.

- Schematics:



- Decoding Logic:



- Pseudo code:
  Part A:
  EBI_init
  LCD_init: Do command and check busy flag
  BF: until the 7th bit is low don't break the while loop
  print name

  PartB:
  Same inits as A
  show_volt: use a loop up table to get the values on the lcd.

  PartC:
  Same as part B
  Keypad: depending on the inputs and outputs return a char.
  Use switch case where depending on the keypad do one of the 6 functions.

- Program Code:
  Part A:

```
/*
 * Lab6_A_RSR.c
 *
 * Created: 7/13/2015 4:51:58 PM
 *  Author: stefano92
 Lab 6 part A
 Name: Ricardo Stefano Reyna
 Section#: 75C9
 TA: Khaled Hassan
 Description: This program is to check my name on the LCD.
 */


#include <avr/io.h>
#include "ebi_driver.h"
#define F_CPU 2000000
#define CS0_Start 0x4000
#define CS1_Start 0x1B0000
#define LCD_COM 0x1B1000
#define LCD_DAT 0x1B1001


void init_EBI();
void init_lcd();
void check_BF();
void out_string(char *str);

int main(void)
{
	char *name = "Stefano Reyna";
	init_EBI();
	init_lcd();

	out_string(name); //Print
	__far_mem_write(LCD_COM, 0x06);
```

```
	check_BF();

	return 0;
}

void init_EBI() {
	PORTH_DIR = 0x37;
	PORTH_OUT = 0x33;
	PORTK_DIR = 0xFF;

	EBI.CTRL = EBI_SRMODE_ALE1_gc | EBI_IFMODE_3PORT_gc;		// ALE1
multiplexing, 3 port configuration

    EBI.CS0.BASEADDRH = (uint8_t) (CS0_Start>>16) & 0xFF;
    EBI.CS0.BASEADDRL = (uint8_t) (CS0_Start>>8) & 0xFF;		// Set CS0 range to
0x004000 - 0x004FFF
    EBI.CS0.CTRLA = EBI_CS_MODE_SRAM_gc | EBI_CS_ASPACE_4KB_gc;		// SRAM mode, 4k
address space

    // BASEADDR is 16 bit (word) register. C interface allows you to set low and high
parts with 1
    // instruction instead of the previous two
    EBI.CS1.BASEADDR = (uint16_t) (CS1_Start>>8) & 0xFFFF;		// Set CS1 range to
0x1B0000 - 0x1BFFFF
    EBI.CS1.CTRLA = EBI_CS_MODE_SRAM_gc | EBI_CS_ASPACE_64KB_gc;
}


void init_lcd() {
	check_BF();
	__far_mem_write(LCD_COM,0x38);
	check_BF();
	__far_mem_write(LCD_COM,0x0F);
	check_BF();
	__far_mem_write(LCD_COM,0x01);
	check_BF();

}

void check_BF() {
	volatile uint8_t temp_val = 0;
	while (1) {
		temp_val = __far_mem_read(LCD_COM);
		if ((temp_val & 0x80) == 0x00) {
			break;
		}
	}
}

void out_string(char *str){
	int cntr = 0;
	//go through each char until null
	while(*str!= 0) {
		check_BF();
		//go to the next line when ends the line
		if(cntr == 16)
		{
			__far_mem_write(LCD_COM, 0xC0);
```

```
                check_BF();
            }
            check_BF();
            __far_mem_write(LCD_DAT,*str);
            str++;
            cntr++;
        }
        check_BF();
}
   PartB:
/*
 * Lab6_B_RSR.c
 *
 * Created: 7/14/2015 5:02:44 PM
 *  Author: stefano92
 Lab 6 part B
 Name: Ricardo Stefano Reyna
 Section#: 75C9
 TA: Khaled Hassan
 Description: This program is to check the voltage of th potentiometer and display it in
the lcd.
 */


#include <avr/io.h>
#include "ebi_driver.h"
#define F_CPU 2000000
#define CS0_Start 0x4000
#define CS1_Start 0x1B0000
#define LCD_COM 0x1B1000
#define LCD_DAT 0x1B1001

char* LUT[]={"0.00 V ", "0.01 V ", "0.02 V ", "0.03 V ", "0.04 V ", "0.05 V ", "0.06 V ",
"0.07 V ", "0.08 V ", "0.09 V ", "0.10 V ", "0.11 V ", "0.12 V ", "0.13 V ", "0.14 V ",
"0.15 V ", "0.16 V ", "0.17 V ", "0.18 V ", "0.19 V ", "0.20 V ", "0.21 V ", "0.22 V ",
"0.23 V ", "0.24 V ", "0.25 V ", "0.26 V ", "0.27 V ", "0.28 V ", "0.29 V ", "0.30 V ",
"0.31 V ", "0.32 V ", "0.33 V ", "0.34 V ", "0.35 V ", "0.36 V ", "0.37 V ", "0.38 V ",
"0.39 V ", "0.40 V ", "0.41 V ", "0.42 V ", "0.43 V ", "0.44 V ", "0.45 V ", "0.46 V ",
"0.47 V ", "0.48 V ", "0.49 V ", "0.50 V ", "0.51 V ", "0.52 V ", "0.53 V ", "0.54 V ",
"0.55 V ", "0.56 V ", "0.57 V ", "0.58 V ", "0.59 V ", "0.60 V ", "0.61 V ", "0.62 V ",
"0.62 V ", "0.63 V ", "0.64 V ", "0.65 V ", "0.66 V ", "0.67 V ", "0.68 V ", "0.69 V ",
"0.70 V ", "0.71 V ", "0.72 V ", "0.73 V ", "0.74 V ", "0.75 V ", "0.76 V ", "0.77 V ",
"0.78 V ", "0.79 V ", "0.80 V ", "0.81 V ", "0.82 V ", "0.83 V ", "0.84 V ", "0.85 V ",
"0.86 V ", "0.87 V ", "0.88 V ", "0.89 V ", "0.90 V ", "0.91 V ", "0.92 V ", "0.93 V ",
"0.94 V ", "0.95 V ", "0.96 V ", "0.97 V ", "0.98 V ", "0.99 V ", "1.00 V ", "1.01 V ",
"1.02 V ", "1.03 V ", "1.04 V ", "1.05 V ", "1.06 V ", "1.07 V ", "1.08 V ", "1.09 V ",
"1.10 V ", "1.11 V ", "1.12 V ", "1.13 V ", "1.14 V ", "1.15 V ", "1.16 V ", "1.17 V ",
"1.18 V ", "1.19 V ", "1.20 V ", "1.21 V ", "1.22 V ", "1.23 V ", "1.24 V ", "1.25 V ",
"1.26 V ", "1.27 V ", "1.28 V ", "1.29 V ", "1.30 V ", "1.31 V ", "1.32 V ", "1.33 V ",
"1.34 V ", "1.35 V ", "1.36 V ", "1.37 V ", "1.38 V ", "1.39 V ", "1.40 V ", "1.41 V ",
"1.42 V ", "1.43 V ", "1.44 V ", "1.45 V ", "1.46 V ", "1.47 V ", "1.48 V ", "1.49 V ",
"1.50 V ", "1.51 V ", "1.52 V ", "1.53 V ", "1.54 V ", "1.55 V ", "1.56 V ", "1.57 V ",
"1.58 V ", "1.59 V ", "1.60 V ", "1.61 V ", "1.62 V ", "1.63 V ", "1.64 V ", "1.65 V ",
"1.66 V ", "1.67 V ", "1.68 V ", "1.69 V ", "1.70 V ", "1.71 V ", "1.72 V ", "1.73 V ",
"1.74 V ", "1.75 V ", "1.76 V ", "1.77 V ", "1.78 V ", "1.79 V ", "1.80 V ", "1.81 V ",
"1.82 V ", "1.83 V ", "1.84 V ", "1.85 V ", "1.86 V ", "1.87 V ", "1.88 V ", "1.88 V ",
"1.89 V ", "1.90 V ", "1.91 V ", "1.92 V ", "1.93 V ", "1.94 V ", "1.95 V ", "1.96 V ",
"1.97 V ", "1.98 V ", "1.99 V ", "2.00 V ", "2.01 V ", "2.02 V ", "2.03 V ", "2.04 V ",
```

```
"2.05 V ", "2.06 V ", "2.07 V ", "2.08 V ", "2.09 V ", "2.10 V ", "2.11 V ", "2.12 V ",
"2.13 V ", "2.14 V ", "2.15 V ", "2.16 V ", "2.17 V ", "2.18 V ", "2.19 V ", "2.20 V ",
"2.21 V ", "2.22 V ", "2.23 V ", "2.24 V ", "2.25 V ", "2.26 V ", "2.27 V ", "2.28 V ",
"2.29 V ", "2.30 V ", "2.31 V ", "2.32 V ", "2.33 V ", "2.34 V ", "2.35 V ", "2.36 V ",
"2.37 V ", "2.38 V ", "2.39 V ", "2.40 V ", "2.41 V ", "2.42 V ", "2.43 V ", "2.44 V ",
"2.45 V ", "2.46 V ", "2.47 V ", "2.48 V ", "2.49 V ", "2.50 V ", "2.51 V ", "2.52 V ",
"2.53 V ", "2.54 V ", "2.55 V ", "2.56 V ", "2.57 V ", "2.58 V ", "2.59 V ", "2.60 V ",
"2.61 V ", "2.62 V ", "2.63 V ", "2.64 V ", "2.65 V ", "2.66 V ", "2.67 V ", "2.68 V ",
"2.69 V ", "2.70 V ", "2.71 V ", "2.72 V ", "2.73 V ", "2.74 V ", "2.75 V ", "2.76 V ",
"2.77 V ", "2.78 V ", "2.79 V ", "2.80 V ", "2.81 V ", "2.82 V ", "2.83 V ", "2.84 V ",
"2.85 V ", "2.86 V ", "2.87 V ", "2.88 V ", "2.89 V ", "2.90 V ", "2.91 V ", "2.92 V ",
"2.93 V ", "2.94 V ", "2.95 V ", "2.96 V ", "2.97 V ", "2.98 V ", "2.99 V ", "3.00 V ",
"3.01 V ", "3.02 V ", "3.03 V ", "3.04 V ", "3.05 V ", "3.06 V ", "3.07 V ", "3.08 V ",
"3.09 V ", "3.10 V ", "3.11 V ", "3.12 V ", "3.12 V ", "3.13 V ", "3.14 V ", "3.15 V ",
"3.16 V ", "3.17 V ", "3.18 V ", "3.19 V ", "3.20 V ", "3.21 V ", "3.22 V ", "3.23 V ",
"3.24 V ", "3.25 V ", "3.26 V ", "3.27 V ", "3.28 V ", "3.29 V ", "3.30 V ", "3.31 V ",
"3.32 V ", "3.33 V ", "3.34 V ", "3.35 V ", "3.36 V ", "3.37 V ", "3.38 V ", "3.39 V ",
"3.40 V ", "3.41 V ", "3.42 V ", "3.43 V ", "3.44 V ", "3.45 V ", "3.46 V ", "3.47 V ",
"3.48 V ", "3.49 V ", "3.50 V ", "3.51 V ", "3.52 V ", "3.53 V ", "3.54 V ", "3.55 V ",
"3.56 V ", "3.57 V ",
"3.58 V ", "3.59 V ", "3.60 V ", "3.61 V ", "3.62 V ", "3.63 V ", "3.64 V ", "3.65 V ",
"3.66 V ", "3.67 V ", "3.68 V ", "3.69 V ", "3.70 V ", "3.71 V ", "3.72 V ", "3.73 V ",
"3.74 V ", "3.75 V ", "3.76 V ", "3.77 V ", "3.78 V ", "3.79 V ", "3.80 V ", "3.81 V ",
"3.82 V ", "3.83 V ", "3.84 V ", "3.85 V ", "3.86 V ", "3.87 V ", "3.88 V ", "3.89 V ",
"3.90 V ", "3.91 V ", "3.92 V ", "3.93 V ", "3.94 V ", "3.95 V ", "3.96 V ", "3.97 V ",
"3.98 V ", "3.99 V ", "4.00 V ", "4.01 V ", "4.02 V ", "4.03 V ", "4.04 V ", "4.05 V ",
"4.06 V ", "4.07 V ", "4.08 V ", "4.09 V ", "4.10 V ", "4.11 V ", "4.12 V ", "4.13 V ",
"4.14 V ", "4.15 V ", "4.16 V ", "4.17 V ", "4.18 V ", "4.19 V ", "4.20 V ", "4.21 V ",
"4.22 V ", "4.23 V ", "4.24 V ", "4.25 V ", "4.26 V ", "4.27 V ", "4.28 V ", "4.29 V ",
"4.30 V ", "4.31 V ", "4.32 V ", "4.33 V ", "4.34 V ", "4.35 V ", "4.36 V ", "4.37 V ",
"4.38 V ", "4.38 V ", "4.39 V ", "4.40 V ", "4.41 V ", "4.42 V ", "4.43 V ", "4.44 V ",
"4.45 V ", "4.46 V ", "4.47 V ", "4.48 V ", "4.49 V ", "4.50 V ", "4.51 V ", "4.52 V ",
"4.53 V ", "4.54 V ", "4.55 V ", "4.56 V ", "4.57 V ", "4.58 V ", "4.59 V ", "4.60 V ",
"4.61 V ", "4.62 V ", "4.63 V ", "4.64 V ", "4.65 V ", "4.66 V ", "4.67 V ", "4.68 V ",
"4.69 V ", "4.70 V ", "4.71 V ", "4.72 V ", "4.73 V ", "4.74 V ", "4.75 V ", "4.76 V ",
"4.77 V ", "4.78 V ", "4.79 V ", "4.80 V ", "4.81 V ", "4.82 V ", "4.83 V ", "4.84 V ",
"4.85 V ", "4.86 V ", "4.87 V ", "4.88 V ", "4.89 V ", "4.90 V ", "4.91 V ", "4.92 V ",
"4.93 V ", "4.94 V ", "4.95 V ", "4.96 V ", "4.97 V ", "4.98 V ", "4.99 V ", "5.00 V "};
static char* LUT_INDEX[]= {"(0x01) ","(0x02) ","(0x03) ","(0x04) ","(0x05) ","(0x06)
","(0x07) ","(0x08) ","(0x09) ","(0x0A) ","(0x0B) ","(0x0C) ","(0x0D) ","(0x0E) ","(0x0F)
","(0x10) ","(0x11) ","(0x12) ","(0x13) ","(0x14) ","(0x15) ","(0x16) ","(0x17) ","(0x18)
","(0x19) ","(0x1A) ","(0x1B) ","(0x1C) ","(0x1D) ","(0x1E) ","(0x1F) ","(0x20) ","(0x21)
","(0x22) ","(0x23) ","(0x24) ","(0x25) ","(0x26) ","(0x27) ","(0x28) ","(0x29) ","(0x2A)
","(0x2B) ","(0x2C) ","(0x2D) ","(0x2E) ","(0x2F) ","(0x30) ","(0x31) ","(0x32) ","(0x33)
","(0x34) ","(0x35) ","(0x36) ","(0x37) ","(0x38) ","(0x39) ","(0x3A) ","(0x3B) ","(0x3C)
","(0x3D) ","(0x3E) ","(0x3F) ","(0x40) ","(0x41) ","(0x42) ","(0x43) ","(0x44) ","(0x45)
","(0x46) ","(0x47) ","(0x48) ","(0x49) ","(0x4A) ","(0x4B) ","(0x4C) ","(0x4D) ","(0x4E)
","(0x4F) ","(0x50) ","(0x51) ","(0x52) ","(0x53) ","(0x54) ","(0x55) ","(0x56) ","(0x57)
","(0x58) ","(0x59) ","(0x5A) ","(0x5B) ","(0x5C) ","(0x5D) ","(0x5E) ","(0x5F) ","(0x60)
","(0x61) ","(0x62) ","(0x63) ","(0x64) ","(0x65) ","(0x66) ","(0x67) ","(0x68) ","(0x69)
","(0x6A) ","(0x6B) ","(0x6C) ","(0x6D) ","(0x6E) ","(0x6F) ","(0x70) ","(0x71) ","(0x72)
","(0x73) ","(0x74) ","(0x75) ","(0x76) ","(0x77) ","(0x78) ","(0x79) ","(0x7A) ","(0x7B)
","(0x7C) ","(0x7D) ","(0x7E) ","(0x7F) ","(0x80) ","(0x81) ","(0x82) ","(0x83) ","(0x84)
","(0x85) ","(0x86) ","(0x87) ","(0x88) ","(0x89) ","(0x8A) ","(0x8B) ","(0x8C) ","(0x8D)
","(0x8E) ","(0x8F) ","(0x90) ","(0x91) ","(0x92) ","(0x93) ","(0x94) ","(0x95) ","(0x96)
","(0x97) ","(0x98) ","(0x99) ","(0x9A) ","(0x9B) ","(0x9C) ","(0x9D) ","(0x9E) ","(0x9F)
","(0xA0) ","(0xA1) ","(0xA2) ","(0xA3) ","(0xA4) ","(0xA5) ","(0xA6) ","(0xA7) ","(0xA8)
","(0xA9) ","(0xAA) ","(0xAB) ","(0xAC) ","(0xAD) ","(0xAE) ","(0xAF) ","(0xB0) ","(0xB1)
```

```
","(0xB2) ","(0xB3) ","(0xB4) ","(0xB5) ","(0xB6) ","(0xB7) ","(0xB8) ","(0xB9) ","(0xBA)
","(0xBB) ","(0xBC) ","(0xBD) ","(0xBE) ","(0xBF) ","(0xC0) ","(0xC1) ","(0xC2) ","(0xC3)
","(0xC4) ","(0xC5) ","(0xC6) ","(0xC7) ","(0xC8) ","(0xC9) ","(0xCA) ","(0xCB) ","(0xCC)
","(0xCD) ","(0xCE) ","(0xCF) ","(0xD0) ","(0xD1) ","(0xD2) ","(0xD3) ","(0xD4) ","(0xD5)
","(0xD6) ","(0xD7) ","(0xD8) ","(0xD9) ","(0xDA) ","(0xDB) ","(0xDC) ","(0xDD) ","(0xDE)
","(0xDF) ","(0xE0) ","(0xE1) ","(0xE2) ","(0xE3) ","(0xE4) ","(0xE5) ","(0xE6) ","(0xE7)
","(0xE8) ","(0xE9) ","(0xEA) ","(0xEB) ","(0xEC) ","(0xED) ","(0xEE) ","(0xEF) ","(0xF0)
","(0xF1) ","(0xF2) ","(0xF3) ","(0xF4) ","(0xF5) ","(0xF6) ","(0xF7) ","(0xF8) ","(0xF9)
","(0xFA) ","(0xFB) ","(0xFC) "};


void init_EBI();
void init_lcd();
void init_AD();
void check_BF();
void out_string(char *str);
void show_V();
void delay();

int main(void)
{
    init_EBI();
       init_lcd();
       init_AD();

       while(1) {
              show_V();
              delay();
       }

       return 0;
}

void init_EBI() {
       PORTH_DIR = 0x37;
       PORTH_OUT = 0x33;
       PORTK_DIR = 0xFF;

       EBI.CTRL = EBI_SRMODE_ALE1_gc | EBI_IFMODE_3PORT_gc;            // ALE1
multiplexing, 3 port configuration

    EBI.CS0.BASEADDRH = (uint8_t) (CS0_Start>>16) & 0xFF;
    EBI.CS0.BASEADDRL = (uint8_t) (CS0_Start>>8) & 0xFF;            // Set CS0 range to
0x004000 - 0x004FFF
    EBI.CS0.CTRLA = EBI_CS_MODE_SRAM_gc | EBI_CS_ASPACE_4KB_gc;          // SRAM mode, 4k
address space

    // BASEADDR is 16 bit (word) register. C interface allows you to set low and high
parts with 1
    // instruction instead of the previous two
    EBI.CS1.BASEADDR = (uint16_t) (CS1_Start>>8) & 0xFFFF;          // Set CS1 range to
0x1B0000 - 0x1BFFFF
    EBI.CS1.CTRLA = EBI_CS_MODE_SRAM_gc | EBI_CS_ASPACE_64KB_gc;
}


void init_lcd() {
       check_BF();
```

```
	__far_mem_write(LCD_COM,0x38);
	check_BF();
	__far_mem_write(LCD_COM,0x0F);
	check_BF();
	__far_mem_write(LCD_COM,0x01);
	check_BF();

}

void init_AD() {
	ADCB_CTRLA = 0x01; //channel 0 enabled and enable ADC
	ADCB_CTRLB = 0x0C; //Free-run and u8-bit
	ADCB_REFCTRL = 0x30; //AREFB
	ADCB_PRESCALER = 0x07; //512
	ADCB_CH0_CTRL = 0x81; //Take reading from channel 0
	PORTB_DIR = 0x01;
	ADCB_CH0_MUXCTRL = 0x20; //PIN4
}

void check_BF() {
	volatile uint8_t temp_val = 0;
	while(1) {
		temp_val = __far_mem_read(LCD_COM);
		if((temp_val & 0x80) == 0x00) {
			break;
		}
	}
}

void out_string(char *str){
	int cntr = 0;
	//go through each char until null
	while(*str!= 0) {
		check_BF();
		//go to the next line when ends the line
		if(cntr == 16)
		{
			__far_mem_write(LCD_COM, 0xC0);
			check_BF();
		}
		check_BF();
		__far_mem_write(LCD_DAT,*str);
		str++;
		cntr++;
	}
	check_BF();
}

void show_V() {
	uint16_t inVoltage = ADCB_CH0_RES;
	uint16_t inINDEX = ADCB_CH0_RES;
	check_BF();
	out_string(LUT[2*inVoltage]);
	check_BF();
	out_string(LUT_INDEX[inINDEX]);
	check_BF();
	__far_mem_write(LCD_COM, 0x02);
	check_BF();
```

```
}

void delay() {
	volatile uint32_t ticks; //Volatile prevents compiler optimization
	for(ticks=0;ticks<=10000;ticks++); //convinient delay
}
   PartC:
/*
 * Lab6_C_RSR.c
 *
 * Created: 7/15/2015 4:14:49 PM
 *  Author: stefano92
 Lab 6 part C
 Name: Ricardo Stefano Reyna
 Section#: 75C9
 TA: Khaled Hassan
 Description: This program depending on the keypad will select a function to perform.
 */


#include <avr/io.h>
#include "ebi_driver.h"
#define F_CPU 2000000
#define CS0_Start 0x4000
#define CS1_Start 0x1B0000
#define LCD_COM 0x1B1000
#define LCD_DAT 0x1B1001

char* LUT[]={"0.00 V ", "0.01 V ", "0.02 V ", "0.03 V ", "0.04 V ", "0.05 V ", "0.06 V ",
"0.07 V ", "0.08 V ", "0.09 V ", "0.10 V ", "0.11 V ", "0.12 V ", "0.13 V ", "0.14 V ",
"0.15 V ", "0.16 V ", "0.17 V ", "0.18 V ", "0.19 V ", "0.20 V ", "0.21 V ", "0.22 V ",
"0.23 V ", "0.24 V ", "0.25 V ", "0.26 V ", "0.27 V ", "0.28 V ", "0.29 V ", "0.30 V ",
"0.31 V ", "0.32 V ", "0.33 V ", "0.34 V ", "0.35 V ", "0.36 V ", "0.37 V ", "0.38 V ",
"0.39 V ", "0.40 V ", "0.41 V ", "0.42 V ", "0.43 V ", "0.44 V ", "0.45 V ", "0.46 V ",
"0.47 V ", "0.48 V ", "0.49 V ", "0.50 V ", "0.51 V ", "0.52 V ", "0.53 V ", "0.54 V ",
"0.55 V ", "0.56 V ", "0.57 V ", "0.58 V ", "0.59 V ", "0.60 V ", "0.61 V ", "0.62 V ",
"0.62 V ", "0.63 V ", "0.64 V ", "0.65 V ", "0.66 V ", "0.67 V ", "0.68 V ", "0.69 V ",
"0.70 V ", "0.71 V ", "0.72 V ", "0.73 V ", "0.74 V ", "0.75 V ", "0.76 V ", "0.77 V ",
"0.78 V ", "0.79 V ", "0.80 V ", "0.81 V ", "0.82 V ", "0.83 V ", "0.84 V ", "0.85 V ",
"0.86 V ", "0.87 V ", "0.88 V ", "0.89 V ", "0.90 V ", "0.91 V ", "0.92 V ", "0.93 V ",
"0.94 V ", "0.95 V ", "0.96 V ", "0.97 V ", "0.98 V ", "0.99 V ", "1.00 V ", "1.01 V ",
"1.02 V ", "1.03 V ", "1.04 V ", "1.05 V ", "1.06 V ", "1.07 V ", "1.08 V ", "1.09 V ",
"1.10 V ", "1.11 V ", "1.12 V ", "1.13 V ", "1.14 V ", "1.15 V ", "1.16 V ", "1.17 V ",
"1.18 V ", "1.19 V ", "1.20 V ", "1.21 V ", "1.22 V ", "1.23 V ", "1.24 V ", "1.25 V ",
"1.26 V ", "1.27 V ", "1.28 V ", "1.29 V ", "1.30 V ", "1.31 V ", "1.32 V ", "1.33 V ",
"1.34 V ", "1.35 V ", "1.36 V ", "1.37 V ", "1.38 V ", "1.39 V ", "1.40 V ", "1.41 V ",
"1.42 V ", "1.43 V ", "1.44 V ", "1.45 V ", "1.46 V ", "1.47 V ", "1.48 V ", "1.49 V ",
"1.50 V ", "1.51 V ", "1.52 V ", "1.53 V ", "1.54 V ", "1.55 V ", "1.56 V ", "1.57 V ",
"1.58 V ", "1.59 V ", "1.60 V ", "1.61 V ", "1.62 V ", "1.63 V ", "1.64 V ", "1.65 V ",
"1.66 V ", "1.67 V ", "1.68 V ", "1.69 V ", "1.70 V ", "1.71 V ", "1.72 V ", "1.73 V ",
"1.74 V ", "1.75 V ", "1.76 V ", "1.77 V ", "1.78 V ", "1.79 V ", "1.80 V ", "1.81 V ",
"1.82 V ", "1.83 V ", "1.84 V ", "1.85 V ", "1.86 V ", "1.87 V ", "1.88 V ", "1.88 V ",
"1.89 V ", "1.90 V ", "1.91 V ", "1.92 V ", "1.93 V ", "1.94 V ", "1.95 V ", "1.96 V ",
"1.97 V ", "1.98 V ", "1.99 V ", "2.00 V ", "2.01 V ", "2.02 V ", "2.03 V ", "2.04 V ",
"2.05 V ", "2.06 V ", "2.07 V ", "2.08 V ", "2.09 V ", "2.10 V ", "2.11 V ", "2.12 V ",
"2.13 V ", "2.14 V ", "2.15 V ", "2.16 V ", "2.17 V ", "2.18 V ", "2.19 V ", "2.20 V ",
"2.21 V ", "2.22 V ", "2.23 V ", "2.24 V ", "2.25 V ", "2.26 V ", "2.27 V ", "2.28 V ",
"2.29 V ", "2.30 V ", "2.31 V ", "2.32 V ", "2.33 V ", "2.34 V ", "2.35 V ", "2.36 V ",
```

```
"2.37 V ", "2.38 V ", "2.39 V ", "2.40 V ", "2.41 V ", "2.42 V ", "2.43 V ", "2.44 V ",
"2.45 V ", "2.46 V ", "2.47 V ", "2.48 V ", "2.49 V ", "2.50 V ", "2.51 V ", "2.52 V ",
"2.53 V ", "2.54 V ", "2.55 V ", "2.56 V ", "2.57 V ", "2.58 V ", "2.59 V ", "2.60 V ",
"2.61 V ", "2.62 V ", "2.63 V ", "2.64 V ", "2.65 V ", "2.66 V ", "2.67 V ", "2.68 V ",
"2.69 V ", "2.70 V ", "2.71 V ", "2.72 V ", "2.73 V ", "2.74 V ", "2.75 V ", "2.76 V ",
"2.77 V ", "2.78 V ", "2.79 V ", "2.80 V ", "2.81 V ", "2.82 V ", "2.83 V ", "2.84 V ",
"2.85 V ", "2.86 V ", "2.87 V ", "2.88 V ", "2.89 V ", "2.90 V ", "2.91 V ", "2.92 V ",
"2.93 V ", "2.94 V ", "2.95 V ", "2.96 V ", "2.97 V ", "2.98 V ", "2.99 V ", "3.00 V ",
"3.01 V ", "3.02 V ", "3.03 V ", "3.04 V ", "3.05 V ", "3.06 V ", "3.07 V ", "3.08 V ",
"3.09 V ", "3.10 V ", "3.11 V ", "3.12 V ", "3.12 V ", "3.13 V ", "3.14 V ", "3.15 V ",
"3.16 V ", "3.17 V ", "3.18 V ", "3.19 V ", "3.20 V ", "3.21 V ", "3.22 V ", "3.23 V ",
"3.24 V ", "3.25 V ", "3.26 V ", "3.27 V ", "3.28 V ", "3.29 V ", "3.30 V ", "3.31 V ",
"3.32 V ", "3.33 V ", "3.34 V ", "3.35 V ", "3.36 V ", "3.37 V ", "3.38 V ", "3.39 V ",
"3.40 V ", "3.41 V ", "3.42 V ", "3.43 V ", "3.44 V ", "3.45 V ", "3.46 V ", "3.47 V ",
"3.48 V ", "3.49 V ", "3.50 V ", "3.51 V ", "3.52 V ", "3.53 V ", "3.54 V ", "3.55 V ",
"3.56 V ", "3.57 V ",
"3.58 V ", "3.59 V ", "3.60 V ", "3.61 V ", "3.62 V ", "3.63 V ", "3.64 V ", "3.65 V ",
"3.66 V ", "3.67 V ", "3.68 V ", "3.69 V ", "3.70 V ", "3.71 V ", "3.72 V ", "3.73 V ",
"3.74 V ", "3.75 V ", "3.76 V ", "3.77 V ", "3.78 V ", "3.79 V ", "3.80 V ", "3.81 V ",
"3.82 V ", "3.83 V ", "3.84 V ", "3.85 V ", "3.86 V ", "3.87 V ", "3.88 V ", "3.89 V ",
"3.90 V ", "3.91 V ", "3.92 V ", "3.93 V ", "3.94 V ", "3.95 V ", "3.96 V ", "3.97 V ",
"3.98 V ", "3.99 V ", "4.00 V ", "4.01 V ", "4.02 V ", "4.03 V ", "4.04 V ", "4.05 V ",
"4.06 V ", "4.07 V ", "4.08 V ", "4.09 V ", "4.10 V ", "4.11 V ", "4.12 V ", "4.13 V ",
"4.14 V ", "4.15 V ", "4.16 V ", "4.17 V ", "4.18 V ", "4.19 V ", "4.20 V ", "4.21 V ",
"4.22 V ", "4.23 V ", "4.24 V ", "4.25 V ", "4.26 V ", "4.27 V ", "4.28 V ", "4.29 V ",
"4.30 V ", "4.31 V ", "4.32 V ", "4.33 V ", "4.34 V ", "4.35 V ", "4.36 V ", "4.37 V ",
"4.38 V ", "4.38 V ", "4.39 V ", "4.40 V ", "4.41 V ", "4.42 V ", "4.43 V ", "4.44 V ",
"4.45 V ", "4.46 V ", "4.47 V ", "4.48 V ", "4.49 V ", "4.50 V ", "4.51 V ", "4.52 V ",
"4.53 V ", "4.54 V ", "4.55 V ", "4.56 V ", "4.57 V ", "4.58 V ", "4.59 V ", "4.60 V ",
"4.61 V ", "4.62 V ", "4.63 V ", "4.64 V ", "4.65 V ", "4.66 V ", "4.67 V ", "4.68 V ",
"4.69 V ", "4.70 V ", "4.71 V ", "4.72 V ", "4.73 V ", "4.74 V ", "4.75 V ", "4.76 V ",
"4.77 V ", "4.78 V ", "4.79 V ", "4.80 V ", "4.81 V ", "4.82 V ", "4.83 V ", "4.84 V ",
"4.85 V ", "4.86 V ", "4.87 V ", "4.88 V ", "4.89 V ", "4.90 V ", "4.91 V ", "4.92 V ",
"4.93 V ", "4.94 V ", "4.95 V ", "4.96 V ", "4.97 V ", "4.98 V ", "4.99 V ", "5.00 V "};
static char* LUT_INDEX[]= {"(0x01) ","(0x02) ","(0x03) ","(0x04) ","(0x05) ","(0x06)
","(0x07) ","(0x08) ","(0x09) ","(0x0A) ","(0x0B) ","(0x0C) ","(0x0D) ","(0x0E) ","(0x0F)
","(0x10) ","(0x11) ","(0x12) ","(0x13) ","(0x14) ","(0x15) ","(0x16) ","(0x17) ","(0x18)
","(0x19) ","(0x1A) ","(0x1B) ","(0x1C) ","(0x1D) ","(0x1E) ","(0x1F) ","(0x20) ","(0x21)
","(0x22) ","(0x23) ","(0x24) ","(0x25) ","(0x26) ","(0x27) ","(0x28) ","(0x29) ","(0x2A)
","(0x2B) ","(0x2C) ","(0x2D) ","(0x2E) ","(0x2F) ","(0x30) ","(0x31) ","(0x32) ","(0x33)
","(0x34) ","(0x35) ","(0x36) ","(0x37) ","(0x38) ","(0x39) ","(0x3A) ","(0x3B) ","(0x3C)
","(0x3D) ","(0x3E) ","(0x3F) ","(0x40) ","(0x41) ","(0x42) ","(0x43) ","(0x44) ","(0x45)
","(0x46) ","(0x47) ","(0x48) ","(0x49) ","(0x4A) ","(0x4B) ","(0x4C) ","(0x4D) ","(0x4E)
","(0x4F) ","(0x50) ","(0x51) ","(0x52) ","(0x53) ","(0x54) ","(0x55) ","(0x56) ","(0x57)
","(0x58) ","(0x59) ","(0x5A) ","(0x5B) ","(0x5C) ","(0x5D) ","(0x5E) ","(0x5F) ","(0x60)
","(0x61) ","(0x62) ","(0x63) ","(0x64) ","(0x65) ","(0x66) ","(0x67) ","(0x68) ","(0x69)
","(0x6A) ","(0x6B) ","(0x6C) ","(0x6D) ","(0x6E) ","(0x6F) ","(0x70) ","(0x71) ","(0x72)
","(0x73) ","(0x74) ","(0x75) ","(0x76) ","(0x77) ","(0x78) ","(0x79) ","(0x7A) ","(0x7B)
","(0x7C) ","(0x7D) ","(0x7E) ","(0x7F) ","(0x80) ","(0x81) ","(0x82) ","(0x83) ","(0x84)
","(0x85) ","(0x86) ","(0x87) ","(0x88) ","(0x89) ","(0x8A) ","(0x8B) ","(0x8C) ","(0x8D)
","(0x8E) ","(0x8F) ","(0x90) ","(0x91) ","(0x92) ","(0x93) ","(0x94) ","(0x95) ","(0x96)
","(0x97) ","(0x98) ","(0x99) ","(0x9A) ","(0x9B) ","(0x9C) ","(0x9D) ","(0x9E) ","(0x9F)
","(0xA0) ","(0xA1) ","(0xA2) ","(0xA3) ","(0xA4) ","(0xA5) ","(0xA6) ","(0xA7) ","(0xA8)
","(0xA9) ","(0xAA) ","(0xAB) ","(0xAC) ","(0xAD) ","(0xAE) ","(0xAF) ","(0xB0) ","(0xB1)
","(0xB2) ","(0xB3) ","(0xB4) ","(0xB5) ","(0xB6) ","(0xB7) ","(0xB8) ","(0xB9) ","(0xBA)
","(0xBB) ","(0xBC) ","(0xBD) ","(0xBE) ","(0xBF) ","(0xC0) ","(0xC1) ","(0xC2) ","(0xC3)
","(0xC4) ","(0xC5) ","(0xC6) ","(0xC7) ","(0xC8) ","(0xC9) ","(0xCA) ","(0xCB) ","(0xCC)
","(0xCD) ","(0xCE) ","(0xCF) ","(0xD0) ","(0xD1) ","(0xD2) ","(0xD3) ","(0xD4) ","(0xD5)
```

```
","(0xD6) ","(0xD7) ","(0xD8) ","(0xD9) ","(0xDA) ","(0xDB) ","(0xDC) ","(0xDD) ","(0xDE)
","(0xDF) ","(0xE0) ","(0xE1) ","(0xE2) ","(0xE3) ","(0xE4) ","(0xE5) ","(0xE6) ","(0xE7)
","(0xE8) ","(0xE9) ","(0xEA) ","(0xEB) ","(0xEC) ","(0xED) ","(0xEE) ","(0xEF) ","(0xF0)
","(0xF1) ","(0xF2) ","(0xF3) ","(0xF4) ","(0xF5) ","(0xF6) ","(0xF7) ","(0xF8) ","(0xF9)
","(0xFA) ","(0xFB) ","(0xFC) "};

void init_EBI();
void init_lcd();
void init_AD();
void check_BF();
void out_string(char *str);
void show_V();
void k_init();
char get_key();
void delay();
void extra_credit();

int main(void)
{
    init_EBI();
       init_lcd();
       init_AD();
       k_init();

       char *ec = "Stefano in jap:";
       char *name =  "Stefano Reyna";
       char *schartz = "May the Schwartzbe with you!";
       char ck = '1';
       char lk = '3';
       int powah = 1;
       char ck2 = '7';
       while (1) {
              if (ck == '&') {
                     ck = lk;
              }
              do {
                     ck = get_key();
                     if (ck == '#') {
                            break;
                     }
              } while (ck == lk);
              switch (ck) {
                     case '1': //function 1
                            __far_mem_write(LCD_COM, 0x01);
                            check_BF();
                            out_string(name);
                            __far_mem_write(LCD_COM, 0x06);
                            check_BF();
                            lk = '1';
                            break;
                     case '3': //funtion 2
                            __far_mem_write(LCD_COM, 0x01);
                            check_BF();
                            lk = '3';
                            break;
                     case '5': //function 3
                            __far_mem_write(LCD_COM, 0x01);
                            check_BF();
```

```
                        out_string(schartz);
                        __far_mem_write(LCD_COM, 0x06);
                        check_BF();
                        lk = '5';
                        break;
                case '7': //function 4
                        __far_mem_write(LCD_COM, 0x01);
                        check_BF();
                        do {
                                ck2 = get_key();
                                show_V();
                                delay();
                        } while (ck2 == '&' || ck2 != '7');
                        break;
                case '#': //function 5
                        if (powah == 1) {
                                __far_mem_write(LCD_COM, 0x08);
                                check_BF();
                                delay();
                                powah = 0;
                                break;
                        }
                        __far_mem_write(LCD_COM, 0x0F);
                        check_BF();
                        delay();
                        powah = 1;
                        break;
                case '@': //function 6
                        __far_mem_write(LCD_COM, 0x01);
                        check_BF();
                        out_string(ec);
                        __far_mem_write(LCD_COM, 0xC0);
                        extra_credit();
                        __far_mem_write(LCD_COM, 0x06);
                        check_BF();
                        lk = '@';
                        break;
                default:
                        break;
                }
        }
        return 0;
}

void init_EBI() {
        PORTH_DIR = 0x37;
        PORTH_OUT = 0x33;
        PORTK_DIR = 0xFF;

        EBI.CTRL = EBI_SRMODE_ALE1_gc | EBI_IFMODE_3PORT_gc;            // ALE1
multiplexing, 3 port configuration

        EBI.CS0.BASEADDRH = (uint8_t) (CS0_Start>>16) & 0xFF;
        EBI.CS0.BASEADDRL = (uint8_t) (CS0_Start>>8) & 0xFF;            // Set CS0 range
to 0x004000 - 0x004FFF
        EBI.CS0.CTRLA = EBI_CS_MODE_SRAM_gc | EBI_CS_ASPACE_4KB_gc;       // SRAM mode, 4k
address space
```

```
	// BASEADDR is 16 bit (word) register. C interface allows you to set low and high
parts with 1
	// instruction instead of the previous two
	EBI.CS1.BASEADDR = (uint16_t) (CS1_Start>>8) & 0xFFFF;		// Set CS1 range
to 0x1B0000 - 0x1BFFFF
	EBI.CS1.CTRLA = EBI_CS_MODE_SRAM_gc | EBI_CS_ASPACE_64KB_gc;
}


void init_lcd() {
	check_BF();
	__far_mem_write(LCD_COM,0x38);
	check_BF();
	__far_mem_write(LCD_COM,0x0F);
	check_BF();
	__far_mem_write(LCD_COM,0x01);
	check_BF();

}

void init_AD() {
	ADCB_CTRLA = 0x01; //channel 0 enabled and enable ADC
	ADCB_CTRLB = 0x0C; //Free-run and u8-bit
	ADCB_REFCTRL = 0x30; //AREFB
	ADCB_PRESCALER = 0x07; //512
	ADCB_CH0_CTRL = 0x81; //Take reading from channel 0
	PORTB_DIR = 0x01;
	ADCB_CH0_MUXCTRL = 0x20; //PIN4
}

void k_init(){
	PORTF_PIN4CTRL = 0x18;
	PORTF_PIN5CTRL = 0x18;
	PORTF_PIN6CTRL = 0x18;
	PORTF_PIN7CTRL = 0x18;
	PORTF_DIRSET = 0x0F; //Low bits are outputs
	PORTF_DIRCLR = 0xF0; //High bits are inputs
}

void check_BF() {
	volatile uint8_t temp_val = 0;
	while(1) {
		temp_val = __far_mem_read(LCD_COM);
		if((temp_val & 0x80) == 0x00) {
			break;
		}
	}
}

void out_string(char *str){
	int cntr = 0;
	//go through each char until null
	while(*str!= 0) {
		check_BF();
		//go to the next line when ends the line
		if(cntr == 16)
		{
			__far_mem_write(LCD_COM, 0xC0);
```

```
                        check_BF();
                }
                check_BF();
                __far_mem_write(LCD_DAT,*str);
                str++;
                cntr++;
        }
        check_BF();
}

void extra_credit() {
        //My name in japanese
        __far_mem_write(LCD_DAT, 0xBD);
        check_BF();
        __far_mem_write(LCD_DAT, 0xC3);
        check_BF();
        __far_mem_write(LCD_DAT, 0xCC);
        check_BF();
        __far_mem_write(LCD_DAT, 0xB1);
        check_BF();
        __far_mem_write(LCD_DAT, 0xC9);
        check_BF();
        __far_mem_write(LCD_COM, 0x06);
        check_BF();
}

void show_V() {
        uint16_t inVoltage = ADCB_CH0_RES;
        uint16_t inINDEX = ADCB_CH0_RES;
        if (inVoltage > 250 && inINDEX > 250) {
                inVoltage = 250;
                inINDEX = 250;
        }
        check_BF();
        out_string(LUT[2*inVoltage]);
        check_BF();
        out_string(LUT_INDEX[inINDEX]);
        check_BF();
        __far_mem_write(LCD_COM, 0x02);
        check_BF();
}

char get_key(){
        uint8_t key;
        PORTF_OUT = 0x0E;
        asm("nop");
        key = PORTF_IN;
        if(key == 0xEE){
                return '1';
        }
        if(key == 0xDE){
                return '5'; //Change to 4 later
        }
        if(key == 0xBE){
                return '7';
        }
        if(key == 0x7E){
                return '#'; //Change to * later
```

```
        }

        PORTF_OUT = 0x0D;
        asm("nop");
        key = PORTF_IN;
        if(key == 0xED){
                return '3'; //Change to 2 later
        }
        if(key == 0xDD){
                return '5';
        }
        if(key == 0xBD){
                return '@'; //change to 8 later
        }
        if(key == 0x7D){
                return '1'; //Change to 0 later
        }
        PORTF_OUT = 0x0B;
        asm("nop");
        key = PORTF_IN;
        if(key == 0xEB){
                return '3';
        }
        if(key == 0xDB){
                return '7'; //Change to 6 later
        }
        if(key == 0xBB){
                return '@'; //Change to 9 later
        }
        if(key == 0x7B){
                return '#';
        }
        PORTF_OUT = 0x07;
        asm("nop");
        key = PORTF_IN;
        if(key == 0xE7){
                return '@'; //Change to A later
        }
        if(key == 0xD7){
                return '@'; //Change to B later
        }
        if(key == 0xB7){
                return '@'; //Change to C later
        }
        if(key == 0x77){
                return '@'; //Change to D later
        }

        return '&';
}

void delay() {
        volatile uint32_t ticks; //Volatile prevents compiler optimization
        for(ticks=0;ticks<=10000;ticks++); //convenient delay
}
```

- Appendix: