#define F\_CPU 16000000UL

#include <avr/io.h>

#include <util/delay.h>

#include <stdio.h>

#include <stdlib.h>

#define ADC\_Pin PA0 //Defining the analogue input port

#define B5 PB5 //relay driver input

#define B6 PB6

#define B7 PB7

#define LCD\_PORT PORTC

#define LCD\_DDR DDRC

#define LCD\_PIN PINC

#define RS PC2

#define EN PC3

uint8\_t customChar[] = { //Enumerated character

0x00,

0x1B,

0x15,

0x11,

0x0A,

0x04,

0x00,

0x00

};

void LCD\_cmd(unsigned char cmd){

LCD\_PORT=(LCD\_PORT & 0x0F)|(cmd & 0xF0);

LCD\_PORT &=~(1<<RS);

LCD\_PORT |=(1<<EN);

\_delay\_us(100);

LCD\_PORT &=~(1<<EN);

\_delay\_us(100);

LCD\_PORT=(LCD\_PORT & 0x0F)|(cmd<<4);

LCD\_PORT |=(1<<EN);

\_delay\_us(100);

LCD\_PORT &=~(1<<EN);

}

void LCD\_data(unsigned char data){

LCD\_PORT=(LCD\_PORT & 0x0F)|(data & 0xF0);

LCD\_PORT |=(1<<RS);

LCD\_PORT |=(1<<EN);

\_delay\_us(100);

LCD\_PORT &=~(1<<EN);

\_delay\_us(100);

LCD\_PORT=(LCD\_PORT & 0x0F)|(data<<4);

LCD\_PORT |=(1<<EN);

\_delay\_us(100);

LCD\_PORT &=~(1<<EN);

}

void LCD\_init(){

LCD\_DDR=0xFC;

LCD\_PORT &=~(1<<EN);

\_delay\_ms(2);

LCD\_cmd(0x33);

\_delay\_us(100);

LCD\_cmd(0x32);

\_delay\_us(100);

LCD\_cmd(0x28);

\_delay\_us(100);

LCD\_cmd(0x0C);

\_delay\_us(100);

LCD\_cmd(0x01);

\_delay\_ms(2);

LCD\_cmd(0x06);

\_delay\_us(100);

}

void LCD\_gotoXY(unsigned char x, unsigned char y){

unsigned char firstCharAdr[]={0x80,0xC0,0x94,0xD4};

LCD\_cmd(firstCharAdr[y-1]+x-1);

\_delay\_us(100);

}

void LCD\_string(char \*str){

unsigned char i=0;

while(str[i]!=0){

LCD\_data(str[i]);

i++;

}

}

void LCD\_customChar (unsigned char loc, unsigned char \*msg)

{

unsigned char i;

if(loc<8)

{

LCD\_cmd(0x40 + (loc\*8)); /\* Command 0x40 and onwards forces the device to point CGRAM address \*/

for(i=0;i<8;i++)

{ /\* Write 8 byte for generation of 1 character \*/

LCD\_data(msg[i]);

}

}

}

void ADC\_init()

{

ADCSRA |=((1<<ADEN)|(ADPS2)|(1<<ADPS1)); /\* ADC Prescaling =64 \*/ //free running mode

//ADCSRA|=(1<<ADEN)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0); // Enable ADC also set Prescaler as 128

ADMUX |=(1<<REFS0); /\* Reference voltage on VCC \*/

}

uint16\_t ADC\_read(uint8\_t channel)

{

ADMUX=((0xF0 & ADMUX) | channel);//enabling 8bit channel

ADCSRA |=(1<<ADSC);//start conversion

while(!(ADCSRA&(1<<ADIF)));//wait for adif conversion complete return

//ADCSRA|=(1<<ADIF);//clear adif when conversion is complete

return(ADC);

}

int main(void)//main entry to the program

{

DDRA &=~(1<<ADC\_Pin);//set PORTA0 as input

ADC\_init();

int ADC\_Value=0 ; // define an integer to save adc read value

LCD\_init();

LCD\_customChar(0,customChar);

while(1)

{

LCD\_gotoXY(1,1);

LCD\_string("IOT system");

\_delay\_ms(10);

LCD\_gotoXY(1,2);

LCD\_string("Fault Detection");

\_delay\_ms(100);

LCD\_cmd(0X01);

LCD\_gotoXY(1,1);LCD\_string("RED");

LCD\_gotoXY(6,1);LCD\_string("YELLOW");

LCD\_gotoXY(13,1);LCD\_string("BLUE");

\_delay\_ms(10);

//LCD\_data(0);

PORTB|=(1<<B7);//Output

PORTB&=~(1<<B6);

PORTB&=~(1<<B5);

\_delay\_ms(100);

ADC\_Value = ADC\_read(ADC\_Pin); //save adc read value in integer

//RED PHASE

if(ADC\_Value>=1000)

{

LCD\_gotoXY(1,2);

LCD\_string("N0FT");

}

else if ((ADC\_Value>=890)&&(ADC\_Value<=920))

{

LCD\_gotoXY(1,2);

LCD\_string("2FL");

}

else if ((ADC\_Value>=870)&&(ADC\_Value<=880))

{

LCD\_gotoXY(1,2);

LCD\_string("4FL");

}

else if ((ADC\_Value>=800)&&(ADC\_Value<=825))

{

LCD\_gotoXY(1,2);

LCD\_string("6FL");

}

else if ((ADC\_Value>=670)&&(ADC\_Value<=688))

{

LCD\_gotoXY(1,2);

LCD\_string("8FL");

}

\_delay\_ms(100);

PORTB&=~(1<<B7);

PORTB|=(1<<B6);//Output

PORTB&=~(1<<B5);

\_delay\_ms(100);

ADC\_Value = ADC\_read(ADC\_Pin); //save adc read value in integer

//YELLOW PHASE

if(ADC\_Value>=1000)

{

LCD\_gotoXY(7,2);

LCD\_string("NOFT");

}

else if ((ADC\_Value>=890)&&(ADC\_Value<=920))

{

LCD\_gotoXY(7,2);

LCD\_string("2FL");

}

else if ((ADC\_Value>=870)&&(ADC\_Value<=880))

{

LCD\_gotoXY(7,2);

LCD\_string("4FL");

}

else if ((ADC\_Value>=800)&&(ADC\_Value<=825))

{

LCD\_gotoXY(7,2);

LCD\_string("6FL");

}

else if ((ADC\_Value>=670)&&(ADC\_Value<=688))

{

LCD\_gotoXY(7,2);

LCD\_string("6FL");

}

\_delay\_ms(100);

PORTB&=~(1<<B7);

PORTB&=~(1<<B6);

PORTB|=(1<<B5);//Output

\_delay\_ms(100);

ADC\_Value = ADC\_read(ADC\_Pin); //save adc read value in integer

//BLUE PHASE

if(ADC\_Value>=1000)

{

LCD\_gotoXY(13,2);

LCD\_string("NOFT");

}

else if ((ADC\_Value>=890)&&(ADC\_Value<=920))

{

LCD\_gotoXY(13,2);

LCD\_string("2FL");

}

else if ((ADC\_Value>=870)&&(ADC\_Value<=880))

{

LCD\_gotoXY(13,2);

LCD\_string("4FL");

}

else if ((ADC\_Value>=800)&&(ADC\_Value<=825))

{

LCD\_gotoXY(13,2);

LCD\_string("6FL");

}

else if ((ADC\_Value>=670)&&(ADC\_Value<=688))

{

LCD\_gotoXY(13,2);

LCD\_string("8FL");

}

\_delay\_ms(100);

LCD\_cmd(0X01);

\_delay\_ms(100);

}

return 0;

}