/\*

\* KEY\_LCD3mar.c

\*

\* Created: 03-04-2023 11:54:53

\* Author : DSP LAB

\*/

/\*

4x4 Keypad Interfacing with ATmega16/32

http://www.electronicwings.com

\*/

#include <avr/io.h>

#include <util/delay.h>

#define KEY\_PRT PORTA

#define KEY\_DDR DDRA

#define KEY\_PIN PINA

#define LCD\_Dir DDRB /\* Define LCD data port direction \*/

#define LCD\_Port PORTB /\* Define LCD data port \*/

#define RS PB0 /\* Define Register Select pin \*/

#define EN PB1 /\* Define Enable signal pin \*/

unsigned char keypad[4][4] = { {'1','2','3','4'},

{'5','6','7','8'},

{'9','A','B','C'},

{'D','E','F','0'}};

unsigned char colloc, rowloc;

void LCD\_Init (void) /\* LCD Initialize function \*/

{

LCD\_Dir = 0xFF; /\* Make LCD port direction as o/p \*/

*\_delay\_ms*(20); /\* LCD Power ON delay always >15ms \*/

LCD\_Command(0x33);

LCD\_Command(0x32); /\* Send for 4 bit initialization of LCD \*/

LCD\_Command(0x28); /\* 2 line, 5\*7 matrix in 4-bit mode \*/

LCD\_Command(0x0c); /\* Display on cursor off \*/

LCD\_Command(0x06); /\* Increment cursor (shift cursor to right) \*/

LCD\_Command(0x01); /\* Clear display screen \*/

}

void LCD\_Command( unsigned char cmnd )

{

LCD\_Port = (LCD\_Port & 0x0F) | (cmnd & 0xF0);/\* Sending upper nibble \*/

LCD\_Port &= ~ (1<<RS); /\* RS=0, command reg. \*/

LCD\_Port |= (1<<EN); /\* Enable pulse \*/

*\_delay\_us*(1);

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

*\_delay\_us*(20);

LCD\_Port = (LCD\_Port & 0x0F) | (cmnd << 4);/\* Sending lower nibble \*/

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

*\_delay\_us*(1);

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

*\_delay\_ms*(2);

}

void LCD\_Data( unsigned char data )

{

LCD\_Port = (LCD\_Port & 0x0F) | (data & 0xF0);/\* Sending upper nibble \*/

LCD\_Port |= (1<<RS); /\* RS=1, data reg. \*/

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

*\_delay\_us*(1);

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

*\_delay\_us*(20);

LCD\_Port = (LCD\_Port & 0x0F) | (data << 4); /\* Sending lower nibble \*/

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

*\_delay\_us*(1);

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

*\_delay\_ms*(2);

}

char keyfind()

{

while(1)

{

KEY\_DDR = 0xF0; /\* set port direction as input-output \*/

KEY\_PRT = 0xFF;

do

{

KEY\_PRT &= 0x0F; /\* mask PORT for column read only \*/

asm("NOP");

colloc = (KEY\_PIN & 0x0F); /\* read status of column \*/

}while(colloc != 0x0F);

do

{

do

{

*\_delay\_ms*(20); /\* 20ms key debounce time \*/

colloc = (KEY\_PIN & 0x0F); /\* read status of column \*/

}while(colloc == 0x0F); /\* check for any key press \*/

*\_delay\_ms* (40); /\* 20 ms key debounce time \*/

colloc = (KEY\_PIN & 0x0F);

}while(colloc == 0x0F);

/\* now check for rows \*/

KEY\_PRT = 0xEF; /\* check for pressed key in 1st row \*/

asm("NOP");

colloc = (KEY\_PIN & 0x0F);

if(colloc != 0x0F)

{

rowloc = 0;

break;

}

KEY\_PRT = 0xDF; /\* check for pressed key in 2nd row \*/

asm("NOP");

colloc = (KEY\_PIN & 0x0F);

if(colloc != 0x0F)

{

rowloc = 1;

break;

}

KEY\_PRT = 0xBF; /\* check for pressed key in 3rd row \*/

asm("NOP");

colloc = (KEY\_PIN & 0x0F);

if(colloc != 0x0F)

{

rowloc = 2;

break;

}

KEY\_PRT = 0x7F; /\* check for pressed key in 4th row \*/

asm("NOP");

colloc = (KEY\_PIN & 0x0F);

if(colloc != 0x0F)

{

rowloc = 3;

break;

}

}

if(colloc == 0x0E)

return(keypad[rowloc][0]);

else if(colloc == 0x0D)

return(keypad[rowloc][1]);

else if(colloc == 0x0B)

return(keypad[rowloc][2]);

else

return(keypad[rowloc][3]);

}

int main(void)

{

LCD\_Init();

while(1)

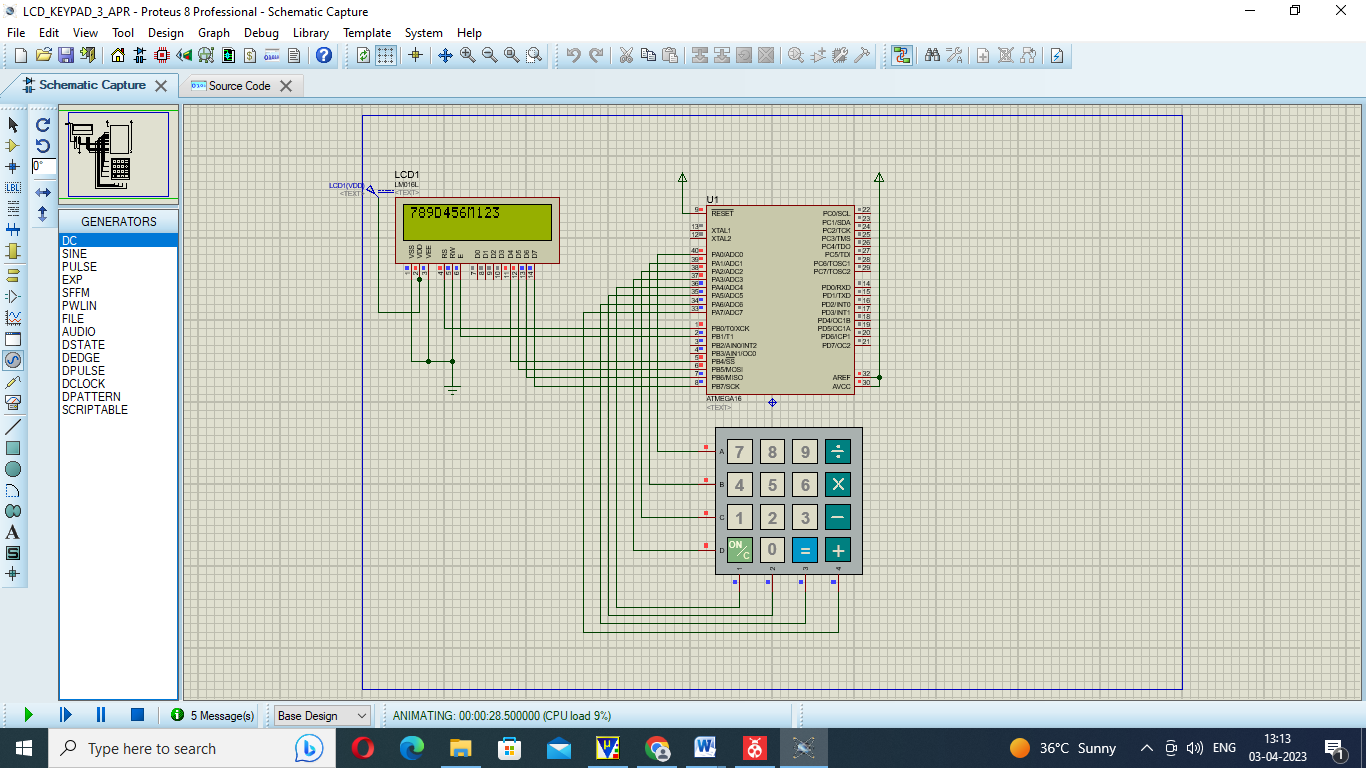
{

// LCD\_Command(0xc0);

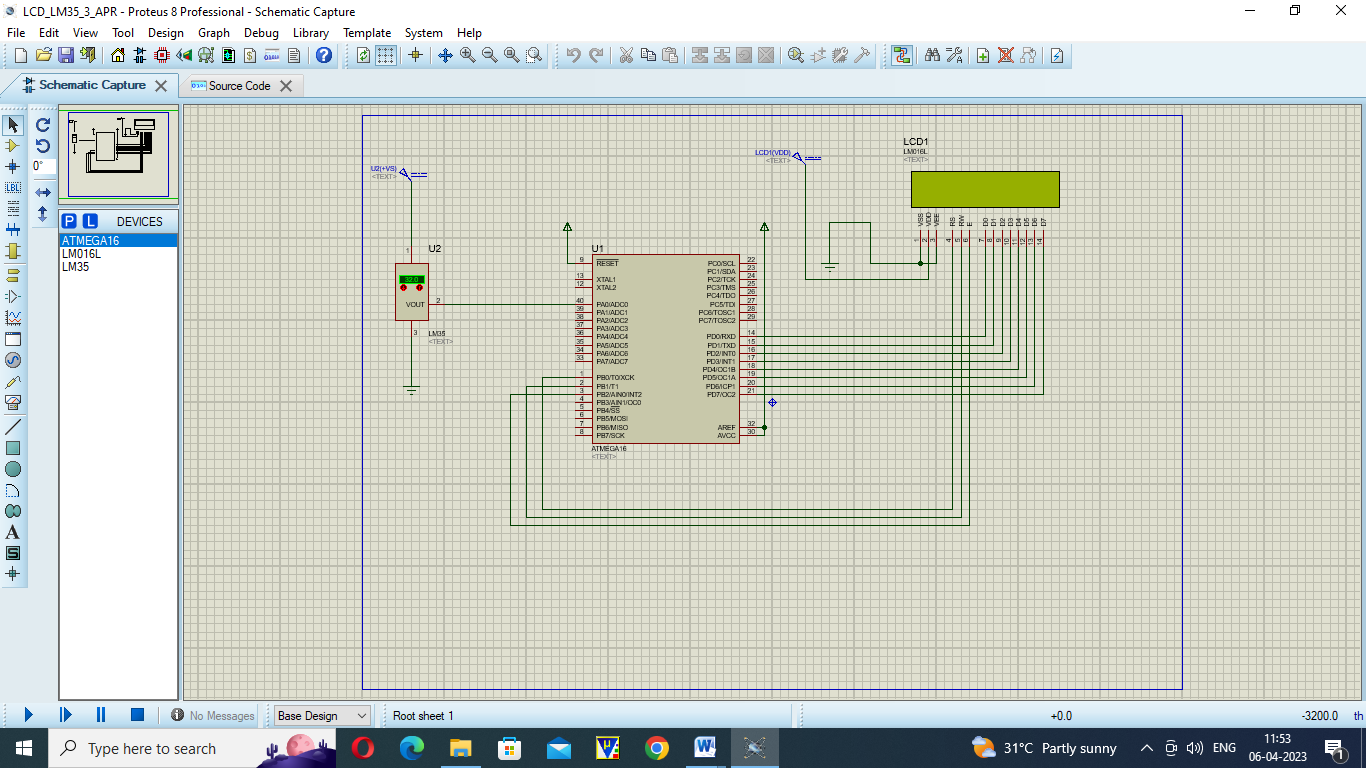
LCD\_Data(keyfind()); /\* Display which key is pressed \*/

}

}



KAY\_LCD3MAR – FILE (ATMEL STUDIO )



/\*

\* LCD\_LM35\_3\_APR.c

LM35 Interfacing with ATmega16/32

\*

\* Created: 03-04-2023 13:43:21

\* Author : DSP LAB

\*/

#include <avr/io.h>

#include <util/delay.h>

#include <string.h>

#include <stdio.h>

#define LCD\_DATA PORTD // LCD data port

#define ctrl PORTB

#define rs PB0 // register select signal

#define rw PB1 // read/write signal

#define en PB2 // enable signal

void LCD\_init(void);

void LCD\_cmd(unsigned char cmd);

void LCD\_write(unsigned char data);

void ADC\_Init(){

DDRA&= ~(1<0); /\* Make PA0 an input for ADC \*/

ADCSRA = 0x87; /\* Enable ADC, with freq/128 or ck/128 \*/

ADMUX = 0xE0; /\* 2.56 V Vref and ADC0 signle ended data will be left justified \*/

}

void LCD\_init(void)

{

LCD\_cmd(0x38); // initialization of 16X2 LCD in 8bit mode

*\_delay\_ms*(1);

LCD\_cmd(0x01); // clear LCD

*\_delay\_ms*(1);

LCD\_cmd(0x0E); // cursor ON

*\_delay\_ms*(1);

LCD\_cmd(0x80); // ---8 go to first line and --0 is for 0th position

*\_delay\_ms*(1);

return;

}

void LCD\_cmd(unsigned char cmd)

{

LCD\_DATA=cmd;

ctrl =(0<<rs)|(0<<rw)|(1<<en); // RS and RW as LOW and EN as HIGH

*\_delay\_ms*(1);

ctrl =(0<<rs)|(0<<rw)|(0<<en); // RS, RW , LOW and EN as LOW

*\_delay\_ms*(50);

return;

}

void LCD\_write(unsigned char data)

{

LCD\_DATA= data;

ctrl = (1<<rs)|(0<<rw)|(1<<en); // RW as LOW and RS, EN as HIGH

*\_delay\_ms*(1);

ctrl = (1<<rs)|(0<<rw)|(0<<en); // EN and RW as LOW and RS HIGH

*\_delay\_ms*(50); // delay to get things executed

return ;

}

int main()

{

unsigned char Vout, x,d1,d2;

//float Vout;

DDRD=0xff; // making LCD\_DATA port as output port

DDRB=0x07; // making signal as out put

LCD\_init(); // initialization of LCD

*\_delay\_ms*(5); // delay of 50 milli seconds

LCD\_write('T'); // call a function to write A on LCD

LCD\_write('e'); // call a function to write A on LCD

LCD\_write('m'); // call a function to write A on LCD

LCD\_write('p'); // call a function to write A on LCD

LCD\_write('='); // call a function to write A on LCD

ADC\_Init(); /\* initialize ADC\*/

while(1)

{

ADCSRA |=(1<<ADSC);

while((ADCSRA&(1<<ADIF))==0); // finidhes conversion

Vout=ADCH;

// Vout=(ADC\_Read(0)\*(5/1024));

//Vout1=Vout\*100;

x=Vout/10; // suppose x=32 then x=32/10

d1=x; // d1=3

d2=(Vout)%10; // d2=32%10 hence d2=2

//celsius = (ADC\_Read(0)\*4.88);

//celsius = (celsius/10.00);

LCD\_write(d1+0x30);

LCD\_write(d2+0x30);

*\_delay\_ms*(1000);

}

}

Exp-7 Serial\_comm

#define *F\_CPU* 12000000UL

#include <avr/io.h>

#include <util/delay.h>

#include <stdlib.h>

#include <stdio.h>

unsigned char data;

//Function for character receiving on Rx pin

//BAUD = Baud rate (in bits per second, bps)

//fosc = System Oscillator clock frequency

//UBRR = Contents of the UBRRH and UBRRL Registers, (0 - 4095)

// system frequency =12Mhz

// Set Baud Rate = 9600 then,

//BAUD=9600

//fosc= 12Mhz

//Then according to above given formula:

//UBRR = (12000000/(16x9600)) - 1

// UBRR = 77.125 which is quite near to 77.WHICH IS EQUV=0X4D

//UBRRH=0x00;

//UBRRL=0x4d;

unsigned char uart\_receive ()

{

while((UCSRA&(1<<RXC)) == 0);

// wait untill 8-bit of a character receive

return UDR;

}

//Functions transmit data

void uart\_transmit (unsigned char data)

{

while (!( UCSRA & (1<<UDRE)));

// wait for the register to free from data

UDR = data;

// load data in the register

}

//Main function

int main()

{

//UART initialization

UBRRH=0x00;

UBRRL=0x4d; // To set the Buad rate to 9600.. get value through above formula

UCSRB=(1<<RXCIE) | (1<<RXEN) | (1<<TXEN);

UCSRC=(1<<URSEL) | (1<<UCSZ1) | (1<<UCSZ0) ;

while(1){

data=uart\_receive(); // MC will wait here infinitly to get a 8-bit character

//the functions below will print the every character three times

uart\_transmit(data);

uart\_transmit(data);

uart\_transmit(data);

}

}