Universitatea Tehnică a Moldovei
Facultatea Calculatoare, Informatică și Microelectronică
Catedra Microelectronica și Inginerie Biomedicala

Raport

Lucrare de Laborator nr.2

La Disciplina:Programarea Microprocesoarelor
Tema: Calculator digital

A efectuat: st. gr.ISBM-141 Idricean Dionisie_____

A verificat: prof.univ. Bragarenco Andrei _____

Chișinău 2017

1.Scopul lucrării

1.Configurarea keypad-ului.

2.Configurarea LCD-ului.

3.Interconexiunea dintre keypad și LCD.

2.Sarcina

Realizarea unei aplicații MCU,care conține un keypad 4x4 configurat manual.Keypadul realizează functia butoanelor de pe un calculator(0..9,+,-,*,/).

3.Noțiuni Teoretice

Afisorul alfanumeric in baza controllerului LCD HD44780, devenit standard pentru afisoarele de acest tip, este unul din cele mai utilizate dispozitive de iesire alfanumerice. Fiind limitat doar la reprezentarea textului monocrom este pe larg intalnit la copiatoare, fax, imprimante, si alte dispozitive de uz comun.

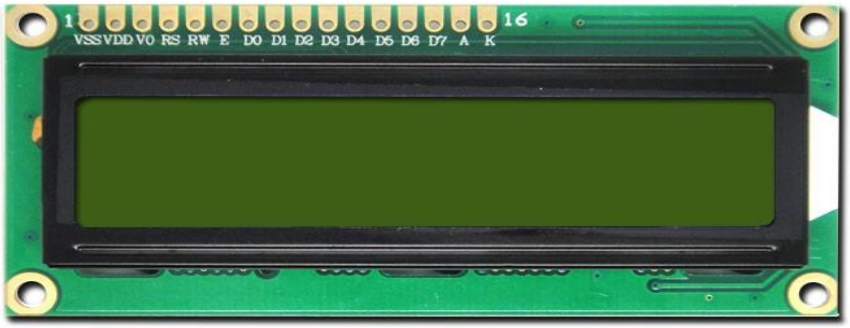

Figura 1: LCD HD44780

Afisorul are cateva configuratii standard: 8x1, 16x2, 20x2, 20x4, fiind capabil sa afiseze maxim 80 caractere. Pentru configuratii mai mari a afisorului, vizualizarea unui numar mai mare de 80 caractere, ca de exemplu afisorul 40x4, se va utiliza un chip aditional.
Cel mai des utilizat este afisorul cu configuratia 16x2, care poate fi usor gasit in magazinele de componente electronice. Este utilizat pe larg pentru constructia de prototipuri si in randurile amatorilor de aplicatii cu microcontrollere. Anume acest tip il vom lua ca exemplu pentru explicatiile lucrului cu afisorul alfanumeric.

Interfata afisorului prezentat aici este una paralela. Orice afisor interfatat paralel care se poate gasi in prezent va avea la baza un controller de tip Hitachi HD44780 sau unul compatibil cu acest cip.Ca regula afisorul are 14 pini de conexiune.

- D0 - D7 - Bus de date bidirectional.
- R/W - determina scrierea sau citirea pentru afisor
- RS - selectarea registrului pentru transfer. RS = 0 registrul de instructiuni este selectat, RS = 1 registrul de date este selectat. Cu acest bit se poate configura ca prin bus-ul de date sa se transfere o camanda sau un caracter.
- E - Bit de activare al LCD. Cand E = 0, LCD nu este activ respectiv semnalale de pe D, RW si RS vor fi ignorate, Cand E = 1, Afisorul este activat si va procesa datele de pe ceilalti pini de interfatare. De mentionat ca scriere datele sunt aplicate pe frontul descrescator al semnalului E., iar la citire, datele vor fi valide odata cu frontul crescator si se vor pastra pana la urmatorul front descrescator.
- Vo - Setarea contrastului pentru afisor.
- Vdd si Vss - pini de alimentare a Afisorului.

Tabelul de comenzi:

Instruction	**Code**									
	RS	**R/W**	**DB7**	**DB6**	**DB5**	**DB4**	**DB3**	**DB2**	**DB1**	**DB0**

Clear display	0	0	0	0	0	0	0	0	0	1
	Curata ecranul afisorului si returneaza cursorul la pozitia initiala. (addresa 0).									
Cursor home	0	0	0	0	0	0	0	0	1	*
	Returneaza cursorul la pozitia initiala (address 0). La fel si cu deplasamentul ferestrei de afisare. Continutul DDRAM ramane neschimbat.									
Entry mode set	0	0	0	0	0	0	0	1	I/D	S
	Seteaza directia deplasarii cursorului (I/D), activeaza deplasarea ferestrei de afisare (S). aceste setari sunt luate in consideratie in timpul operatiilor de citire/scriere.									
Display On/Off control	0	0	0	0	0	0	1	D	C	B
	Aprinde/Stinge ecranul (D), Aprinde/Stinge cursorul (C) si activeaza/deactiveaza licarirea caracterului la pozitia cursorului (B).									
Cursor/display shift	0	0	0	0	0	1	S/C	R/L	*	*
	Deplasare cursor/deplasare ecran (S/C) in directia de deplasare (R/L). Continutul DDRAM ramane neschimbat.									
Function set	0	0	0	0	1	DL	N	F	*	*
	Seteaza marimea interfetei (1-8bit/0-4bit) (DL), numarul de linii (N) si fontul (0 - 5x8 / 1 - 5x10) (F).									
Set CGRAM address	0	0	0	1	CGRAM address					
	Seteaza adresa de citire/scriere pentru CGRAM.									
Set DDRAM address	0	0	1	DDRAM address						
	Seteaza adresa de citire/scriere pentru DDRAM.									
Read busy-flag and address counter	0	1	BF	CGRAM / DDRAM address						
	Citeste indicatorul Busy-flag (BF) care indica ca dispoziticul este ocupat cu o operatie interna. totodata citeste contorul de adresa curenta a memorie CGRAM or DDRAM (in dependenta de care a fost selectata cu o operatie precedenta).									

Write to CGRAM or DDRAM	1	0	write data
	Inscrie date in CGRAM or DDRAM.		
Read from CGRAM or DDRAM	1	1	read data
	Citeste date din CGRAM or DDRAM.		

4.1.Schema-bloc:

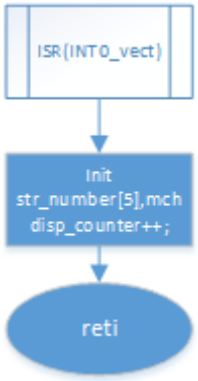

Figura 2 : Schema Bloc ISR-"Ext0_Int"

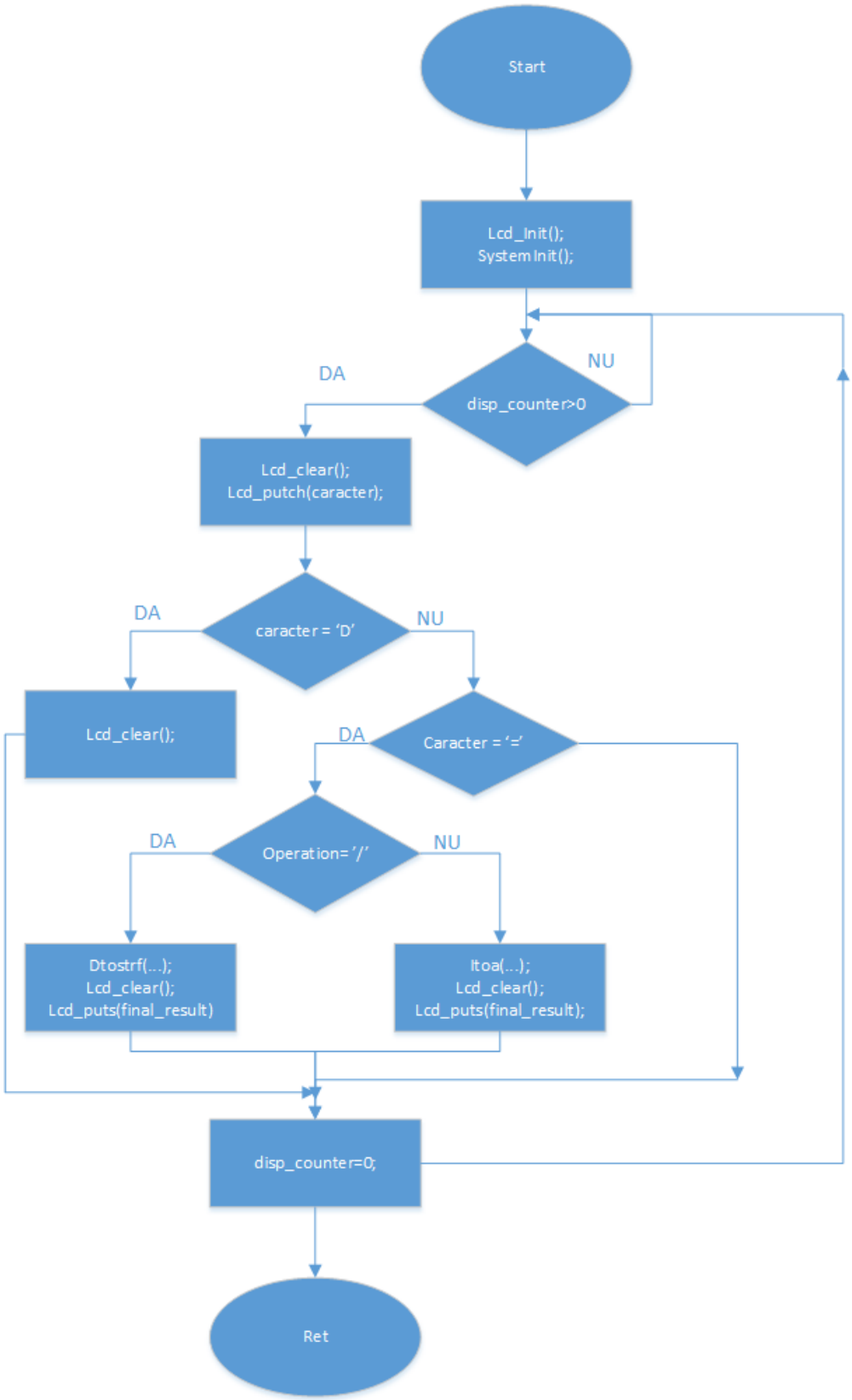

Figura 3 : Schema Bloc a functiei "main"

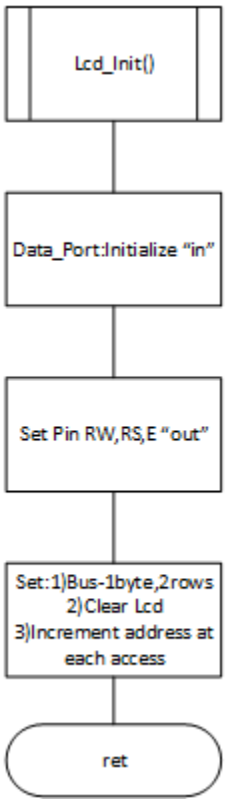

Figura 4 : Schema Bloc a funcției"Lcd_Init()"

4.2.Schema Flux de Date:

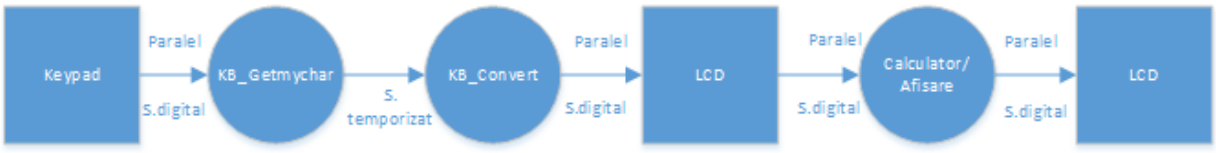

Figura 5 : Flux de Date

4.3.Schema electrică:

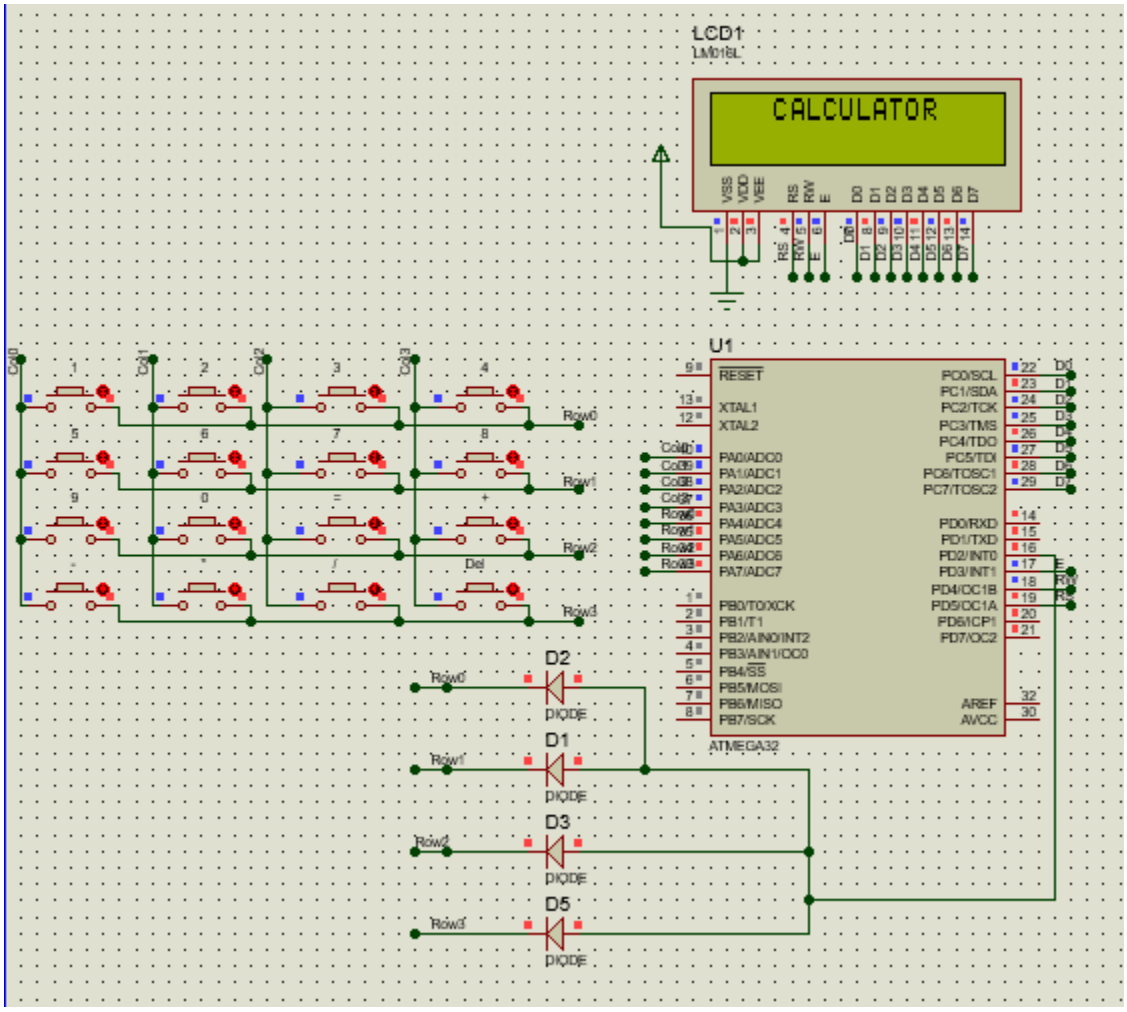

Figura 6 : Schema electrică în Proteus

5.Programul:

main.c

```
/*
 * L2.c
 *
 * Created: 2/21/2017 9:28:15 AM
 * Author : denis
 */
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include "lcd.h"
#include "calculator.h"
```

```
volatile char i=0;
unsigned char operation;
unsigned int my_number;
unsigned int my_number2;
volatile unsigned char mch,disp_counter;
volatile float div_result;
char final_result[10];

/*
Functia data creaza legatura dintre libraria
calculator si keypad
Input:
1)array de 5 caractere-in care se va salva
n-rul reprezentat in forma de caractere
2)un parameru de tip char-reprezinta tasta
apasata de utilizator
*/
void ProcessPushedButton(char str_number[5],char mch)
{
//daca utilizatorul apasa pe 'Del',atunci
//aducem i la starea initiala,str_number-resetam
       if(mch == 'D'){i=0; str_number[0] = '\0';}
       else
       {
              str_number[i++] = mch;//umplem tabloul cu caractere
//Daca utilizatorul apasa pe semn,atunci are loc salvarea
//semnului in variabila 'operation',din str_number se scoate semnul
              if(mch == '+' | mch == '-' | mch == '*' | mch == '/')
              {
                     operation = mch;
                     str_number[--i]='\0';//excludem semnul din tablou
                     my_number=atoi(str_number);//convertim in int n-rul
                     i=0; str_number[i]='\0';
              }
/*Daca a fost folosit '=',
prelucram operatia aritmetica*/
              else if(mch == '=')
              {
                     str_number[--i]='\0';//excludem egalul din tablou
                     my_number2=atoi(str_number);//convertim in int n-r2
                     i=0; str_number[i]='\0';
/*Identificarea operatiei cerute de utilizator*/
                     switch (operation)
                     {
                            case '+':
                            my_number = Suma(my_number,my_number2);
                            break;
                            case '-':
                            my_number = Scaderea(my_number,my_number2);
                            break;
                            case '*':
                            my_number = Inmultirea(my_number,my_number2);
                            break;
                            case '/':
                            div_result = Impartirea(my_number,my_number2);
                            break;
                     }
```

```
                }
        }

}

ISR(INT0_vect)
{
/*Afisarea pe LCD
in dependenta de butonul apasat*/
        char str_number[5];
        mch = KB_Getmychar();
        mch = KB_Convert(mch);

        ProcessPushedButton(str_number[5],mch);
//valoarea aceasta se incrementeaza atunci
//cind este indeplinita ISR,poate fi inlocuita cu bool
disp_counter++;
}
void SystemInit(void);
int main(void)
{
        SystemInit();
        while (1)
    {
                /*conditia pentru afisarea aceluiasi caracter o data*/
                if(disp_counter>0)
                {
                        Lcd_clear();
                        Lcd_putc(mch);
                        if(mch=='D') Lcd_clear();
                        else if(mch=='=')
                        {
                                Lcd_clear();
                                if(operation == '/')
                                {
                                        dtostrf(div_result,3,2,final_result);
                                        Lcd_clear();
                                        Lcd_puts(final_result);
                                }
                                else{
                                        itoa(my_number,final_result,10);
                                        Lcd_clear();
                                        Lcd_puts(final_result);
                                }
                        }

                disp_counter=0;
                }

    }
}

void SystemInit()
{
/*PORT INIT*/
        DDRD  = 0b0000000;
        PORTD = 0xff;
        DDRA=0x0f;//0..3-in;4..7-out
```

```
	PORTA=0xf0;
/*EXT0 INIT*/
	MCUCR |= (1<<ISC01);
	GICR |=(1<<INT0);
	GIFR |=(1<<INTF0);

	sei();
	Lcd_Init();
	Lcd_puts("   CALCULATOR");
}
```

calculator.c

```
/*
 * calculator.c
 *
 * Created: 3/5/2017 9:33:05 AM
 *  Author: denis
 */

unsigned int Suma(int a,int b)
{
	return (a+b);
}
unsigned int Scaderea(int a,int b)
{
	return (a-b);
}
unsigned int Inmultirea(int a,int b)
{
	return (a*b);
}
float Impartirea(float a,float b)
{
	return (a / b);
}
```

lcd.h

```
/*
 * lcd.h
 *
 * Created: 3/4/2017 2:14:49 PM
 *  Author: denis
 */


#ifndef LCD_H_
#define LCD_H_

#include <avr/io.h>
/*Define Data Pins,RW,RS and E Pins*/
#define DATA_DDR DDRA
#define DATA_PORT PORTA
#define DATA_PIN PINA
```

```
#define PIN_DATA0 0
#define PIN_DATA1 1
#define PIN_DATA2 2
#define PIN_DATA3 3
#define PIN_DATA4 4
#define PIN_DATA5 5
#define PIN_DATA6 6
#define PIN_DATA7 7
#define CONTROL_DDR DDRD
#define CONTROL_PORT PORTD
#define PIN_E  3
#define PIN_RW 4
#define PIN_RS 5



void Lcd_Init(void);
void Lcd_putc(unsigned char);
void Lcd_clear(void);
static void Read_Pins(unsigned char);
void Data_Port_In(void);
void Data_Port_Out(void);
#endif /* LCD_H_ */
```

lcd.c

```
#include "lcd.h"
#include <util/delay.h>

void Lcd_Init()
{
/*Configurarea PINILOR D0..D7->OUT,0*/
	DDRC     = 0xff;
	DATA_PORT = 0x00;
/*Configurarea Pinilor de Comanda RW,RS,E->OUT,0*/
	DDRD         |=  ((1 << PIN_E) | (1 << PIN_RS) | (1<<PIN_RW));
	CONTROL_PORT &= ~((1 << PIN_E) | (1 << PIN_RS) | (1<<PIN_RW));
/*Bus-ul de date pe 8 biti,afisare in 2 rinduri*/
	Read_Pins(0x38);
/*Curatirea ecranului*/
	 Lcd_clear();
/*Display ON*/
	 Read_Pins(0x0c);
/*Incrementarea adresei la fiecare acces, Ecranul nu se deplaseaza.*/
	Read_Pins(0x06);

/*
	CONTROL_PORT |= (1 << PIN_RS);
	CONTROL_PORT &= ~(1 << PIN_RW);
	Read_Pins(0x31);
*/

}

void Lcd_clear(void)
{
	CONTROL_PORT &= ~(1 << PIN_RS);
```

```
	CONTROL_PORT &= ~(1 << PIN_RW);
	Read_Pins(0x01);
}

static void Read_Pins(unsigned char byte)
{
		DATA_PORT = byte;
		CONTROL_PORT |=  (1 << PIN_E); _delay_ms(1);
		CONTROL_PORT &= ~(1 << PIN_E); _delay_ms(1);
}

void Lcd_putc(unsigned char character)
{
	CONTROL_PORT |= (1 << PIN_RS);
	CONTROL_PORT &= ~(1 << PIN_RW);
	Read_Pins(character);
}
void Lcd_puts(char *str)
{
	while(*str)
	{
		Lcd_putc(*str++);
	}
}
```

keypad.c

```
/*
 * mycharpad.c
 *
 * Created: 2/21/2017 9:33:31 AM
 *  Author: denis
 */
#include "keypad.h"
#include <avr/io.h>
unsigned char KB_Getmychar(void)
{
	DDRA=0x0f;//0..3-in;4..7-out
	PORTA=0xF0;
	unsigned char cols = PINA;
	DDRA=0xf0;
	PORTA=0x0f;
	unsigned char row = PINA;
	DDRA=0x0f;//0..3-in;4..7-out
	PORTA=0xF0;
	return (cols | row);

}
unsigned char KB_Convert(unsigned char mymychar)
{
	unsigned char mychar;
	 switch (mymychar) {
		case 0b11101110:
		mychar = '1';
		 break;
		 case 0b11101101:
		mychar = '2';
		 break;
```

```
                case 0b11101011:
                mychar = '3';
                break;
                case 0b11100111:
                mychar = '4';
                break;
                case 0b11011110:
                mychar = '5';
                break;
                case 0b11011101:
                mychar = '6';
                break;
                case 0b11011011:
                mychar = '7';
                break;
                case 0b11010111:
                mychar = '8';
                break;
                case 0b10111110:
                mychar = '9';
                break;
                case 0b10111101:
                mychar = '0';
                break;
                case 0b10111011:
                mychar = '=';
                break;
                case 0b10110111:
                mychar = '+';
                break;
                case 0b01111110:
                mychar = '-';
                break;
                case 0b01111101:
                mychar = '*';
                break;
                case 0b01111011:
                mychar = '/';
                break;
                default : mychar = 'D';
                break;
        }
         return mychar;
}
```

6.Soft-uri utilizate pentru sarcina dată:

- Proteus 8.5
- Atmel Studio 7.0
- Microsoft Visio

7.Concluzie

Prin executarea a mai multor încercări,am realizat programul pînă la capăt.Cea mai grea parte a lucrării a constituit elaborarea librăriei pentru LCD.Crearea manuală a librarii me-a permis să înțeleg mai bine structura și funcționarea LCD-ului,la fel și configurarea keypad-ului.Ambele structuri reprezintă interfețe eficiente pentru comunicarea cu utilizatorul.

8.Referințe

www.microlab.club

http://www.atmel.com/images/Atmel-0856-AVR-Instruction-Set-Manual.pdf

http://www.avrfreaks.net/