



UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

Proiect Microcontrollere
Termostat de cameră

Documentație

Realizat: Zabunov Veaceslav Daniil

Grupa 2133/2

An III EA

CUPRINS

1.	Alegerea senzorului analogic de temperatură	2
1.1	Ce este un senzor analogic de temperatură?	2
1.2	LMT70 Precision Analog Temperature Sensor.....	2
1.3	LM35	4
1.4	LMT84.....	4
1.5	LM45	5
2	Conectarea senzorului la microcontroller.....	6
3	LCD LM016L.....	11
3.1	Modul de comunicare	13
3.2	Display data RAM – DDRAM	13
3.3	CGROM – Character Generator ROM.....	14
4	Afișarea temperaturii pe LCD LM016L.....	20
4.1	Microcontroller-ul 8051 si LCD LM016L	21
4.2	Modul de gândire al afișărilor.....	23
5	Elementul de execuție, afișarea setării cu butoane.....	30
	BIBLIOGRAFIE	35

1. Alegerea senzorului analogic de temperatură

Definiția unui senzor: un senzor este un dispozitiv tehnic ce poate reacționa calitativ sau cantitativ, la anumite proprietăți fizice sau chimice ale mediului în care se află.

Deci senzorul este un dispozitiv care măsoară o mărime fizică (masă, presiune, temperatură, umiditate etc.) și o transformă într-un semnal care poate fi citit de către un observator printr-un instrument sau poate fi prelucrat.

Senzorii de temperatură sunt clasificați, în general ca fiind:

- **Termoelectrici sau termocupluri:** aceștia se bazează pe efectul Seebeck. Acest efect se referă la apariția unei forte termoelectrice într-un circuit în care se află două joncțiuni din metale sau aliaje de metale. Valoarea forței depinde de diferența dintre temperaturile joncțiunilor și de tipul acestora. Sudura caldă este joncțiunea de măsurare, iar capetele libere ale conductoarelor creează punctul de referință.
- **Rezistivi:** aceștia necesită alimentare, acestia se bazează pe modificarea rezistenței senzorului, proporțional cu temperatura pe care o au. Cel mai des folosit material în construcția acestor senzori este platina, dar există senzori confectionați din nichel sau cupru.
- **Termistori:** senzori bazați pe termorezistoare, cu coeficienți de temperatură ridicăți.

Criteriile de alegere ale unui senzor sunt, în mare:

- Gama de temperatură
- Liniaritatea
- Sensibilitatea
- Timpul de răspuns
- Stabilitatea
- Acuratețea
- Durabilitatea

1.1 Ce este un senzor analogic de temperatură?

Un senzor analogic de temperatură emite o valoare analogică proporțională cu temperatura mediului ambient.

Pentru această aplicație am ales categoria de senzori bazați pe tehnologie CMOS, pentru acuratețea și liniaritatea pe care ne-o oferă.

1.2 LMT70 Precision Analog Temperature Sensor

Este un senzor de precizie înaltă, de putere mică realizat în tehnologie CMOS. Este cel mai des întâlnit în domeniul IoT, unde este nevoie de un senzor cu cost redus, precizie înaltă și putere mică.

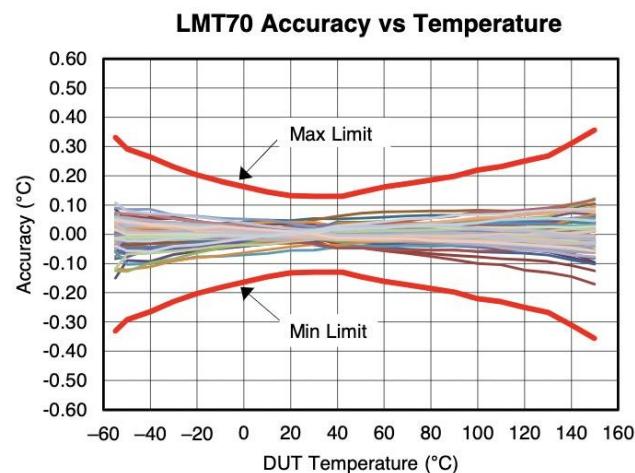
Parametri:

- Accuracy:
 - $\pm 0.05^\circ\text{C}$ (typ) or $\pm 0.13^\circ\text{C}$ (max) from 20°C to 42°C
 - $\pm 0.2^\circ\text{C}$ (max) from -20°C to 90°C
 - $\pm 0.23^\circ\text{C}$ (max) from 90°C to 110°C
 - $\pm 0.36^\circ\text{C}$ (max) from -55°C to 150°C
- Wide Temperature Range: -55°C to 150°C
- Matching of Two Adjacent LMT70A on Tape and Reel: 0.1°C (max) at 30°C
- Very Linear Analog Temperature Sensor with Output Enable Pin
- NTC Output Slope: $-5.19 \text{ mV}/^\circ\text{C}$
- Output On/Off Switch with $R_{DS\text{ on}} < 80 \Omega$
- Wide Power Supply Range: 2.0 V to 5.5 V
- Low Power Supply Current: $9.2 \mu\text{A}$ (typ) $12 \mu\text{A}$ (max)
- Ultra Small 0.88 mm by 0.88 mm 4-bump WLCSP (DSBGA) Package

Principiul de funcționare al unui senzor în tehnologie CMOS: Acești senzori măsoară temperatura comparând o tensiune dependentă de temperatură, cu o tensiune insensibilă la temperatură. Aceste tensiuni sunt obținute folosindu-ne de caracteristici de temperatură bine definite ale unui tranzistor PNP BJT, și anume:

- **Caracteristica CTAT** (Complementary to absolute temperature) al tensiunii V_{BE} .
- **Caracteristica PTAT** (Proportional to absolute temperature) a diferenței de tensiune dintre 2 jonctiuni baza-emitor. (ΔV_{be}).

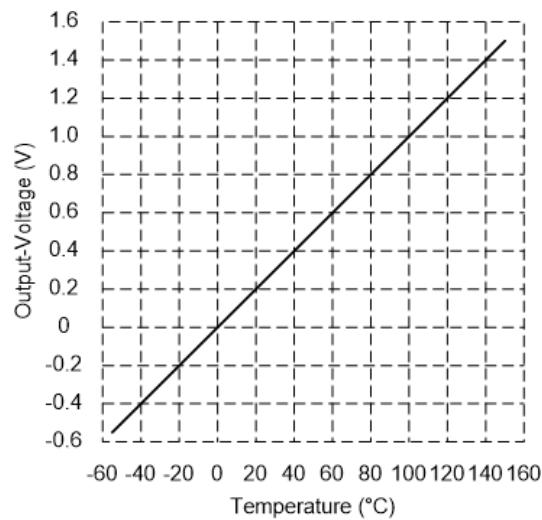
Acuratetea lui LMT70:



1.3 LM35

Principalele avantaje ale acestui senzor sunt: liniaritatea, care este foarte bună, costul redus **10RON**, tensiunea de ieșire variază cu $10\text{mV}/\text{grad C}$. Acuratețea acestui senzor este de 0.5C , la 25 de grade Celsius.

$$\text{Ecuatia de ieșire: } \text{VOUT} = 10 \text{ mV}/^\circ\text{C} \times T$$



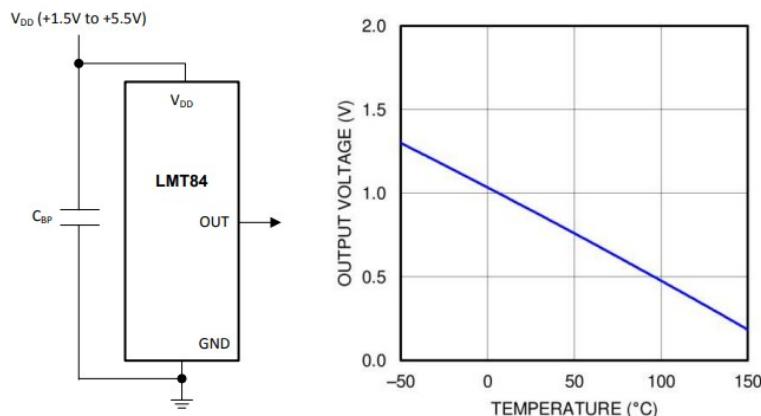
CARACTERISTICA DE IESIRE PENTRU LM35

1.4 LMT84

Parametri:

- Very Accurate: $\pm 0.4^\circ\text{C}$ Typical
- Low 1.5-V Operation
- Average Sensor Gain of $-5.5 \text{ mV}/^\circ\text{C}$
- Low $5.4\text{-}\mu\text{A}$ Quiescent Current
- Wide Temperature Range: -50°C to 150°C
- Output is Short-Circuit Protected
- Push-Pull Output With $\pm 50\text{-}\mu\text{A}$ Drive Capability
- Footprint Compatible With the Industry-Standard LM20/19 and LM35 Temperature Sensors
- Cost-Effective Alternative to Thermistors

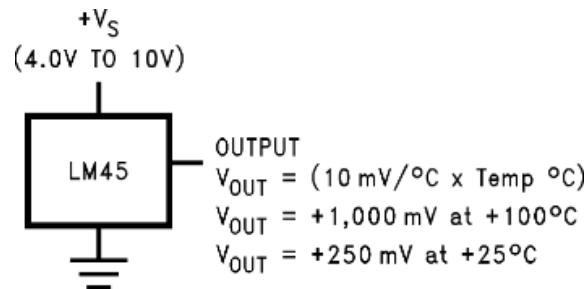
Output Voltage vs Temperature



Copyright © 2016, Texas Instruments Incorporated
Caracteristica de ieșire pentru LMT84

1.5 LM45

Este un senzor asemănător cu LM35, însă acesta nu are o precizie aşa de bună. Precizia acestuia este de $+$ - 3 grade Celsius. Ecuația tensiunii de ieșire este aceeași ca și pentru LM35.

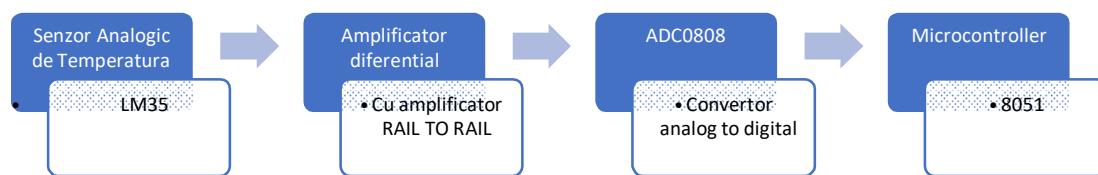


Concluzie

- Aleg LM35 deoarece:
- Preț redus – 10 RON aproximativ.
- Consum mic de curent și tensiune, implicit putere consumată mică.
- Acuratețe de ± 0.5 .
- Liniaritate foarte bună.
- Dimensiuni reduse.

2 Conectarea senzorului la microcontroller

Schema Bloc a Circuitului



Pentru această aplicație am ales un amplificator operațional de tip rail-to-rail, pentru a ne putea întinde pe aproximativ întreg domeniul de tensiune, și care poate fi alimentat de la 5V.

Am ales un amplificator **LMV34MA**.

6.1 Absolute Maximum Ratings

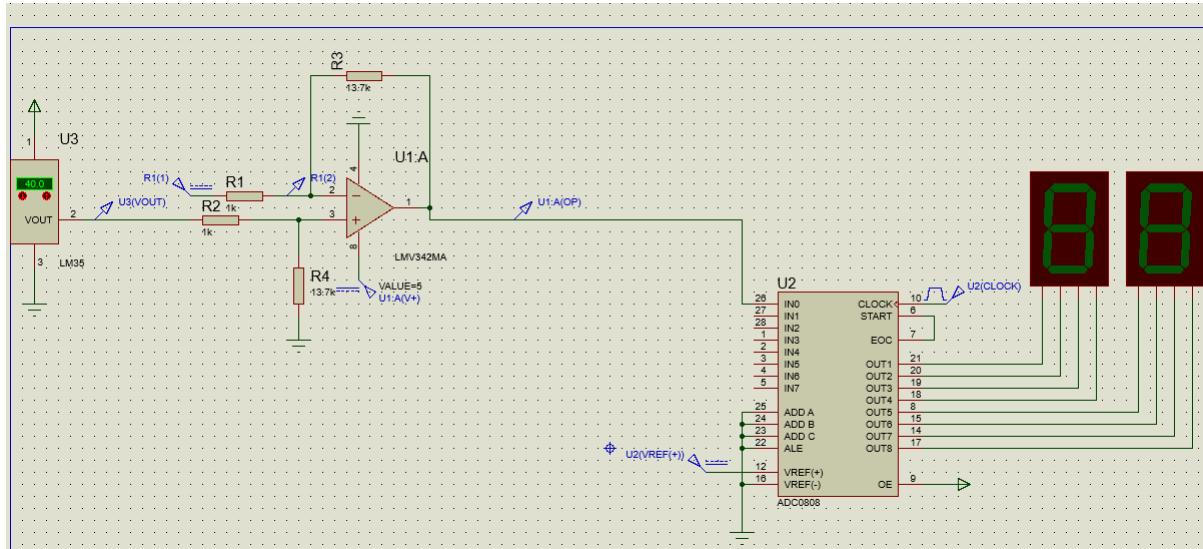
over operating free-air temperature range (unless otherwise noted)⁽¹⁾⁽²⁾

		MIN	MAX	UNIT
Differential input voltage	±Supply voltage			
Supply voltage ($V^+ - V^-$)		6		V
Output short circuit to V^+	See ⁽³⁾			
Output short circuit to V^-	See ⁽⁴⁾			
Lead temperature	Infrared or convection reflow (20 s)		235	°C
	Wave soldering (10 s)		260	
Junction temperature, T_J ⁽⁵⁾			150	°C
Storage temperature, T_{sig}		-65	150	°C

Acesta se poate alimenta la 5V, și este de tip **rail to rail**.

2.1 Breviar de calcul

Schema completă a circuitului se poate observa în figura de mai jos:



Pentru realizarea rezistenței de 13.7K se poate folosi o rezistență de 14k ohm din seria E48.

Rezistența de 1K se poate alege tot din seria E48. (luăm în calcul și toleranțele).

Deoarece senzorul LM35 variază cu aproximativ 10mV/°C, ADC0808 nu poate detecta schimbarea la fiecare grad.

$$V_{lsb} = \frac{VFS}{2^8} = \frac{5V}{256} = 19.53\text{ mV}$$

$$\frac{\Delta V}{\Delta T} = \frac{5V}{370mV} = \frac{13.51mV}{^{\circ}\text{C}} < V_{lsb}$$

La 3 grade dorim să fie afișată valoare de 00 pe cele BCD-uri. Ecuația de variație a tensiunii de ieșire a senzorului în funcție de temperatură este:

$$V_{out} = \frac{10mV}{^{\circ}\text{C}} * T$$

Unde T este temperatura în grade Celsius. Deci, la 3 grade o să avem aproximativ 30 mV. Noi la ieșire dorim o tensiune de aproximativ 0 volți deci scădem aproximativ 30mV. V1=30mV.

$$V_{out\,amp} = \frac{R3}{R1} * (V2 - V1)$$

La 40 de grade Celsius, dorim să avem aproximativ 5V la ieșirea diferențialului. Calculăm amplificarea necesară. La 40 de grade Celsius avem 400mV aproximativ la ieșirea din senzor.

$$Ampl = \frac{5V}{400mV} = 12.5$$

Pentru siguranță o să folosim o amplificare mai mare, chiar dacă intră în limitarea tensiunii de ieșire a amplificatorului.

La 3 grade Celsius:

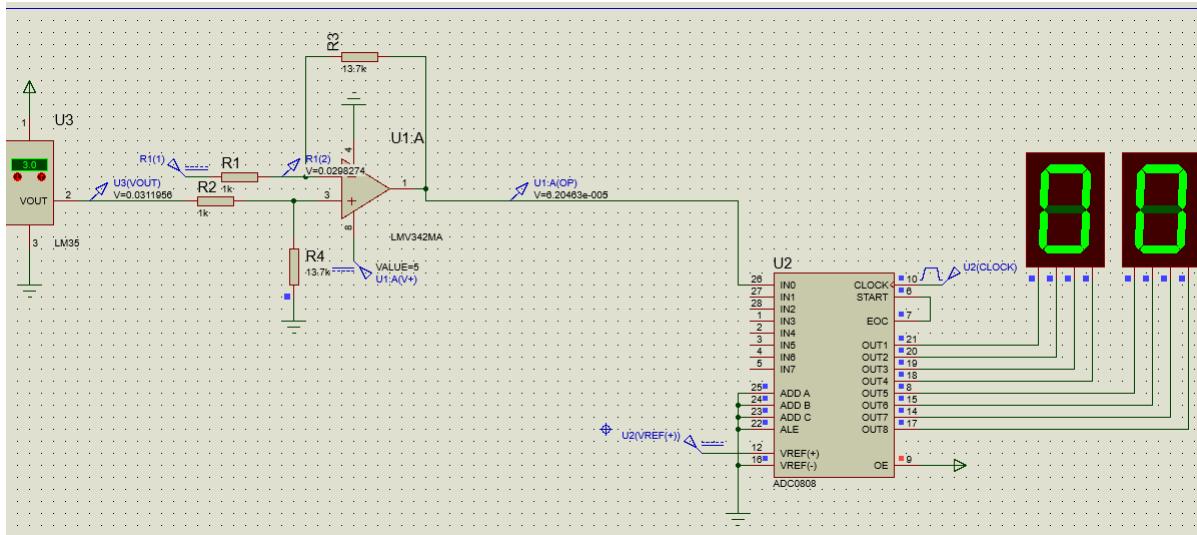
$$V_{out} = 13.5 * (\sim 30mV - \sim 30mV) = \sim 0V$$

La 40 de grade Celsius:

$$V_{out} = 13.5 * (\sim 400mV - \sim 30mV) = 4.995V$$

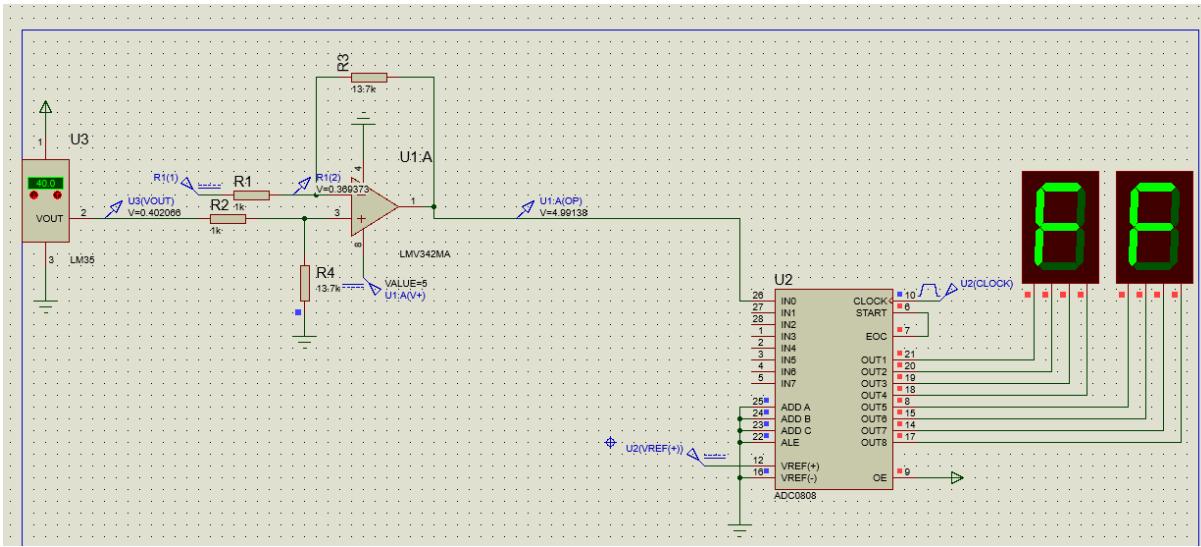
2.2 Rezultatele Simulării

La 3 grade Celsius pe cele 2 BCD 7 segmente obținem valoarea 0 0.



Simulare pentru 3 grade Celsius

La 40 de grade Celsius, obținem tensiunea maximă pe care ne-o poate oferi amplificatorul operațional. 4.991 V.



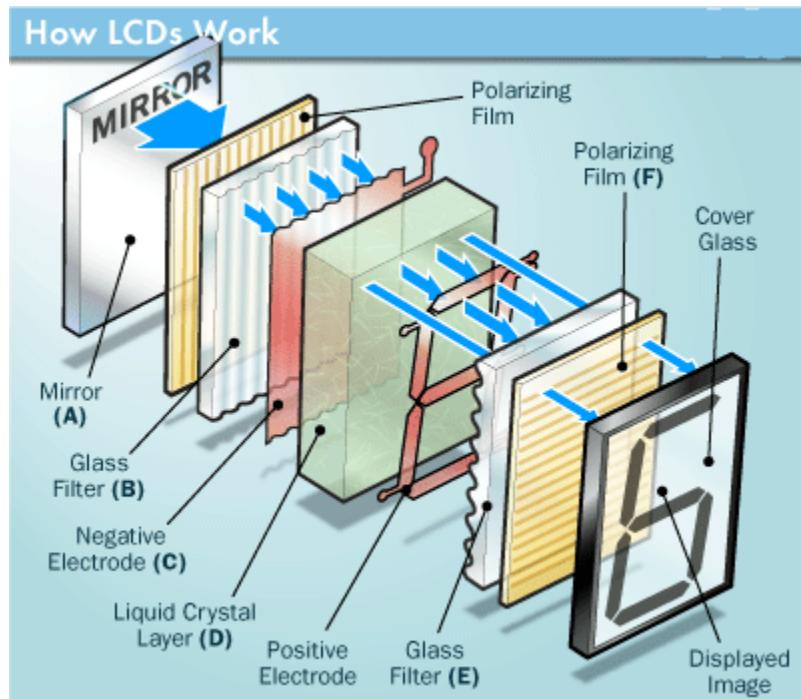
Simulare pentru 40 de grade Celsius

3 LCD LM016L- Ce este un LCD si cum functioneaza?

LCD (Liquid Crystal Display) este un tip de afişaj cu ecran plat care utilizează cristale lichide în forma sa principală de funcționare. LED-urile au multiple utilizări pentru consumatori și întreprinderi, deoarece pot fi găsite în mod obișnuit în telefoane inteligente, televizoare, și panouri de instrumente.

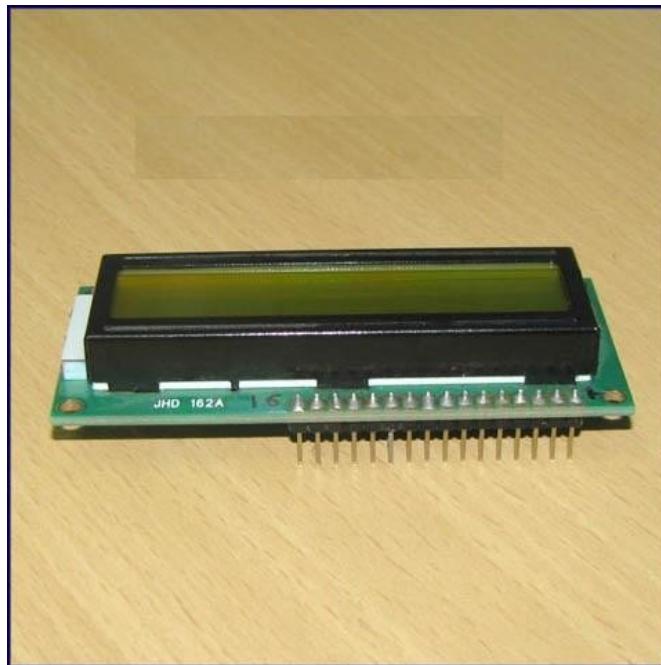
Ele funcționează folosind cristale lichide pentru a produce o imagine. Cristalele lichide sunt încorporate în ecranul de afişare și există o formă de lumină de fundal folosită pentru a le ilumina. Afişajul cu cristale lichide real este format din mai multe straturi, inclusiv un filtru polarizat și electrozi.

O celulă cu cristale lichide este alcătuită dintr-un strat subțire (aproximativ 10 μ m) de cristal lichid prinț între două foi de sticlă cu electrozi transparenti depuși pe fețele lor interioare. Cu ambele foi de sticlă transparente, celula este cunoscută ca celulă de tip transmisiv. Când o sticlă este transparentă, iar cealaltă are un strat reflectorizant, celula se numește tip reflectorizant. Ecranul LCD nu produce nicio iluminare proprie. De fapt, depinde în întregime de iluminarea care cade asupra ei dintr-o sursă externă pentru efectul său vizual.



LCD-urile color sunt cele care pot afișa imagini în culori. Pentru ca acest lucru să fie posibil, trebuie să existe trei sub-pixeli cu filtre de culoare roșu, verde și albastru pentru a crea fiecare pixel de culoare. Pentru combinarea acestor sub-pixeli, aceste LCD-uri trebuie

conectate la un număr mare de tranzistori. Dacă apare vreo problemă cu acești tranzistori, va cauza un pixel mort.



LM016L



Pinout

3.1 Modul de comunicare

Chipul de control al LCD-ului este standardizat, și acesta este HD44780U, făcut de Hitachi Semiconductors. Acest standard face usoara comunicarea datelor transmise de un microcontroller, în cazul nostru 8051, către LCD LM016L.

Liniile de control sunt EN, RS, RW.

- a) EN: vine de la Enable, aceasta linie de control îi comunica LCD-ului ca îi transmitem date. Ca să transmitem date LCD-ului, programul nostru trebuie să se asigure că aceasta linie este LOW, în timp ce celălalte 2 linii sunt HIGH ca să scriem date pe BUS. Cand celălalte linii sunt pe modul ready, punem enable pe HIGH și așteptăm un timp minim care se găsește în datasheet-ul LCD-ului.
- b) RS: Register Select. Cand acesta este setat pe LOW (0), datele care intră sunt tratate ca și comenzi sau instrucțiuni speciale (de exemplu clear screen, poziționare cursorului). Cand RS este 1, datele sunt transmise ca și niste mesaje care trebuie afisate pe LCD.
- c) RW: Read/Write. Cand RW este pe LOW, informația de pe busul de date este scrisă pe LCD. Cand RW e pe HIGH (1) programul citeste informația de la LCD. Doar Get LCD Status este o comandă de citire.
- d) Data bus: este de 2 tipuri: 4 linii sau 8 linii (în funcție de modul de operare ales de utilizator).

Busy flag – este un indicator care ne arată dacă LCD-ul poate să primească date, sau nu (adică este ocupat cu alte lucruri). Busy Flag-ul este reprezentat de bitul 7 cand rs e 0 și rw e 1.

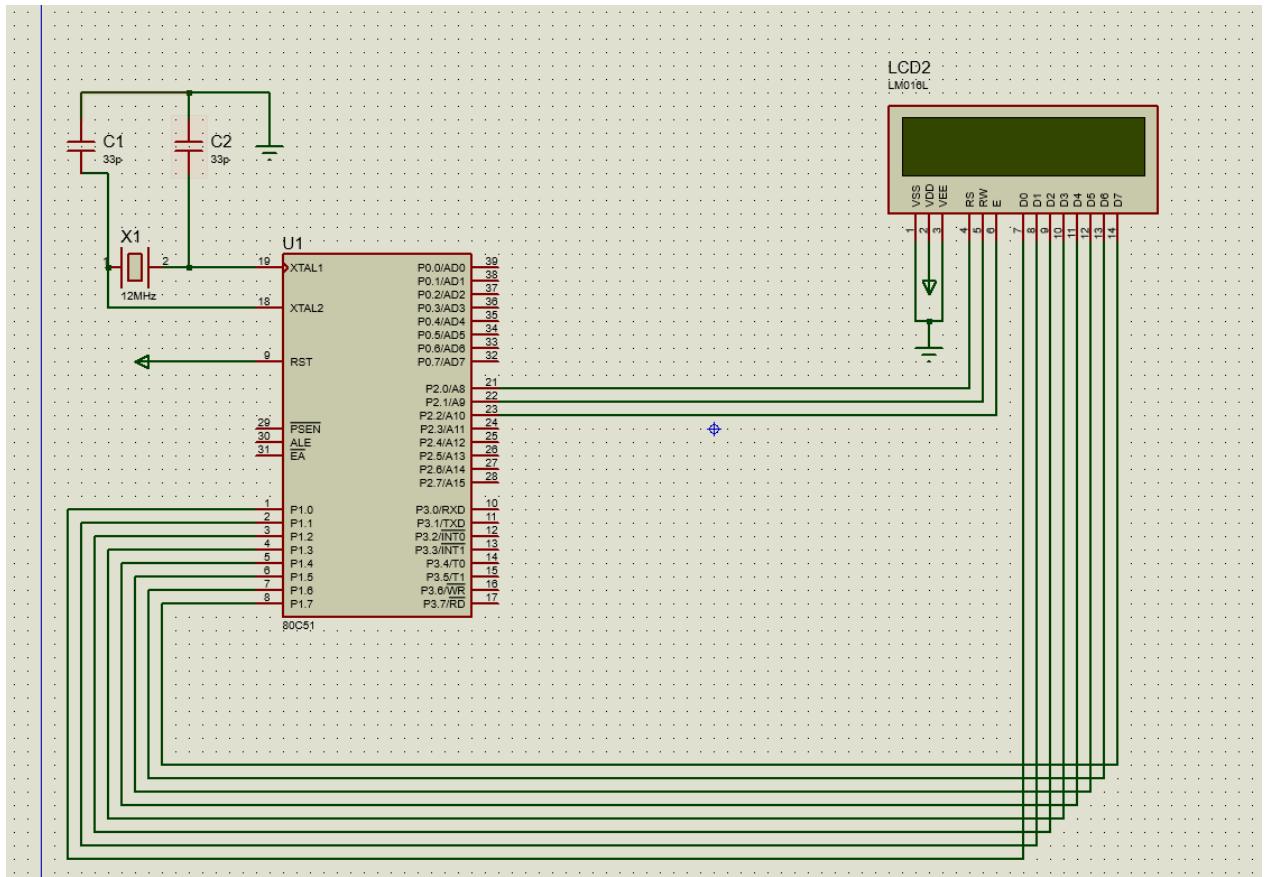
3.2 Display data RAM – DDRAM

Datele de afișare RAM (DDRAM) stochează datele de afișare reprezentate în coduri de caractere de 8 biți. Capacitatea sa extinsă este de 80×8 biți sau 80 de caractere. Zona din RAM de date de afișare (DDRAM) care nu este utilizată pentru afișare poate fi utilizată ca RAM de date generale. Deci, orice trimitem pe DDRAM este de fapt afișat pe LCD. Pentru LCD-uri precum 1×16 , sunt vizibile doar 16 caractere, astăzi că orice scrieți după 16 caractere este scris în DDRAM, dar nu este vizibil pentru utilizator.

3.3 CGROM – Character Generator ROM

ROM-ul generator de caractere generează modele de caractere de 5 x 8 puncte sau 5 x 10 puncte din coduri de caractere de 8 biți. Poate genera 208 modele de caractere de 5 x 8 puncte și 32 modele de caractere de 5 x 10 puncte. Modelele de caractere definite de utilizator sunt disponibile și prin ROM-ul programat cu mască.

Circuitul cu 8051:



Codul Sursa pentru 8051 în assembly:

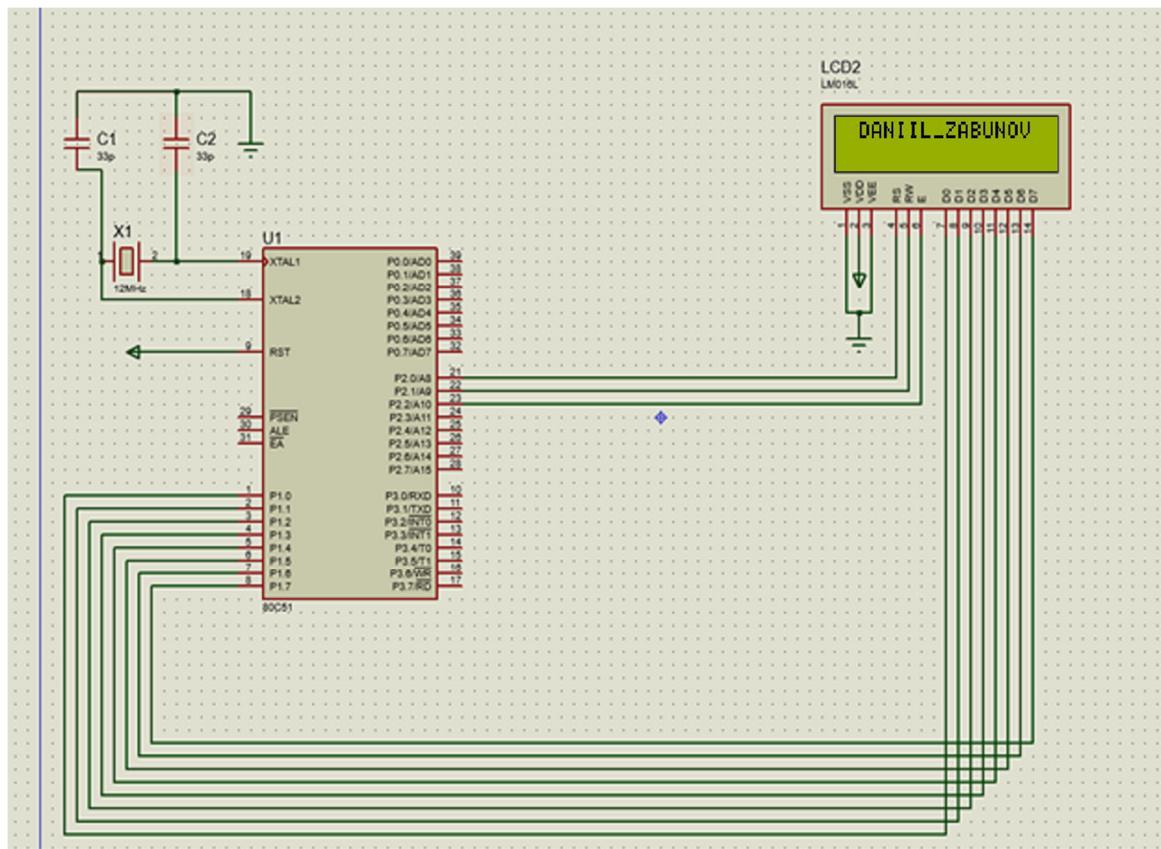
```
1 org 0h
2 mov a, #38h
3 acall command
4 mov a, #0eh
5 acall command
6 mov a, #01h
7 acall command
8 mov a, #06h
9 acall command
10 mov a, #80h
11 acall command
12     mov a, #'D'
13     acall data_display
14     mov a, #'A'
15     acall data_display
16     mov a, #'N'
17     acall data_display
18     mov a, #'I'
19     acall data_display
20     mov a, #'I'
21     acall data_display
22     mov a, #'L'
23     acall data_display
24     mov a, #'_'
25     acall data_display
26     mov a, #'Z'
27     acall data_display
28     mov a, #'A'
29     acall data_display
30     mov a, #'B'
31     acall data_display
32     mov a, #'U'
33     acall data display
34     mov a, #'N']
35     acall data_display
36     mov a, #'O'
37     acall data_display
38     mov a, #'V'
39     acall data_display
```

```

HERE: SJMP HERE ;STAY HERE
COMMAND:
    ACALL READY ;IS LCD READY
    MOV P1, A
    CLR P2.0 ;RS=0 PENTRU COMENZI
    CLR P2.1 ; RW = 0 PENTRU A SCRIE IN LCD
    SETB P2.2 ; E=1 FOR H-TO-L PULSE
    NOP
    NOP
    CLR P2.2 ;E=0, LATCH IN
    RET
DATA DISPLAY:
    ACALL READY ; VERIFICA DACA LCD UL ESTE PREGATIT
    MOV P1, A
    SETB P2.0
    CLR P2.1
    SETB P2.2
    NOP
    NOP
    CLR P2.2
    RET
READY:
    CLR P2.2
    SETB P1.7
    CLR P2.0
    SETB P2.1
BACK:
    CLR P2.2
    SETB P2.2
    JB P1.7, BACK ;STAY UNTIL BUSY FLAG = 0
    CLR P2.2
    RET
END

```

Acest cod verifica si Busy Flag-ul. Pentru verificarea acestuia folosim comanda **GET LCD STATUS**. Aceasta metoda returnează 2 biti de informație. Aceasta informație se găsește în bitul 7 de date. Dacă acest bit este ,1' atunci LCD este ocupat și nu poate prelua alte date. Dacă Bit 7 =0 atunci LCD-ul poate prelua date, adică putem să ii dam în continuare comenzi.



3.4 CODUL IN C51:

```
#include <reg51.h>
sfr ldata = 0x90; //P1=LCD data pins
sbit rs = P3^2;
sbit rw = P3^3;
sbit en = P3^4;
sbit busy = P1^7;
void display (unsigned char code *msgchar);
void lcdcmd(unsigned char value);
void lcddata(unsigned char value);
void MSDelay (unsigned int itime);
void lcdready();

code unsigned char ml [] = {"this a new test"};

void main(){
    lcdcmd(0x38);
    lcdcmd(0x0E);
    lcdcmd(0x01);
    lcdcmd(0x06);
    lcdcmd(0x86); //line 1, position 6
    display(&ml);
    //lcddata(ml[0]);
    //lcddata(ml[1]);
    //lcddata(ml[2]);
}

void lcdcmd(unsigned char value){
    lcdready(); //check the LCD busy flag
    ldata = value; //put the value on the pins
    rs = 0;
    rw = 0;
    en = 1; //strobe the enable pin
    MSDelay(1);
    en = 0;
    return;
}

void lcddata(unsigned char value){
    lcdready(); //check the LCD busy flag
    ldata = value; //put the value on the pins
    rs = 1;
    rw = 0;
    en = 1; //strobe the enable pin
    MSDelay(1);
    en = 0;
    //return;
}

void lcdready()
{
    busy = 1; //make the busy pin at input
    rs = 0;
    rw = 1;
    while(busy==1){ //wait here for busy flag
        en = 0; //strobe the enable pin
        MSDelay(1);
        en = 1;
    }
}
```

```

void MSDelay (unsigned int itime){
    unsigned int i, j;
    for(i=0;i<itime;i++)
        for(j=0;j<1275;j++);
}

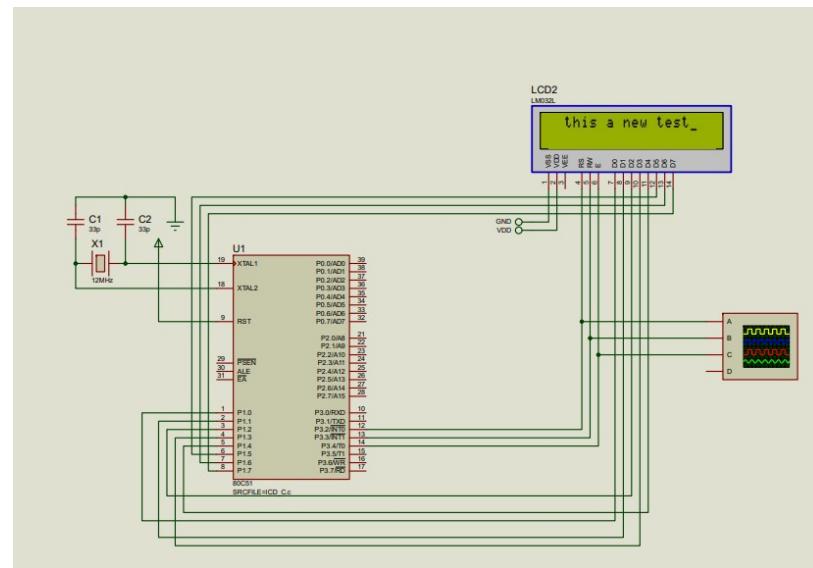
/*void display (void) {
    unsigned char i =0;
    while (m1[i]!=0) {
        //for (i=0;4; i++)

        lcddata (m1[i]);
        i++ ;
    }
} */

void display (unsigned char code *msgchar) {

    while (*msgchar != 0){
        lcddata (*msgchar);
        msgchar++;
    }
}

```



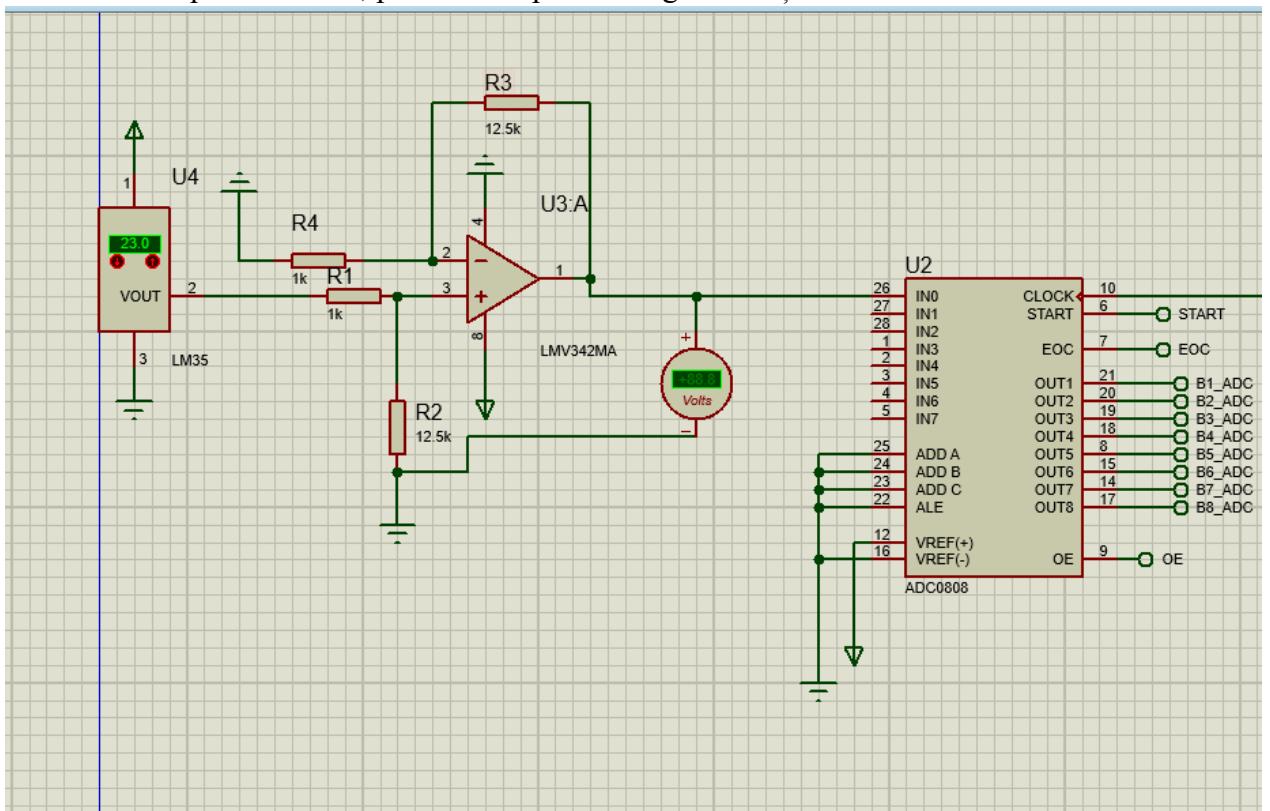
Schema pentru codul în C51 din Cursul de Microcontrollere

4 Afisarea temperaturii pe LCD LM016L

4.1 Componente

In schema noastra avem nevoie de urmatoarele componente: un senzor analog de temperatura, in cazul nostru LM35, un convertor analog-digital, pe 8 biti, ADC0808, un microcontroller din familia 8051, si un LCD LM016L.

Pentru a mari domeniul de variație a senzorului, în funcție cu temperatura, deoarece ADC-ul are VLSB de aproximativ 20mV, o să folosim un circuit de amplificare diferențială, cu factorul de amplificare 12.5, pentru a acoperi întreaga rezoluție a ADC-ului.



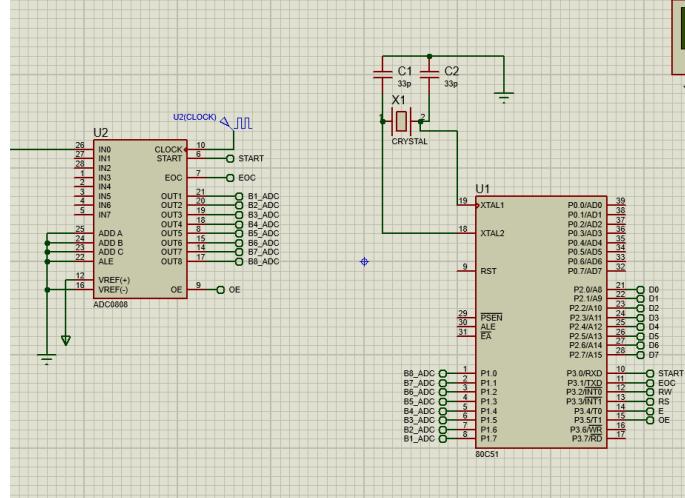
4.2 ADC0808

Pentru a face conversia din analog în digital, folosim acest convertor analog-digital, ce conține un CAN cu registru de aproximări succesive. Pentru a porni conversia avem nevoie de un impuls high-to-low pe bitul de start, output enable sa fie 1, iar conversia este terminata atunci când EOC (end of conversion) este pe HIGH.

Codul in C51 pentru conversia făcuta de ADC este următorul:

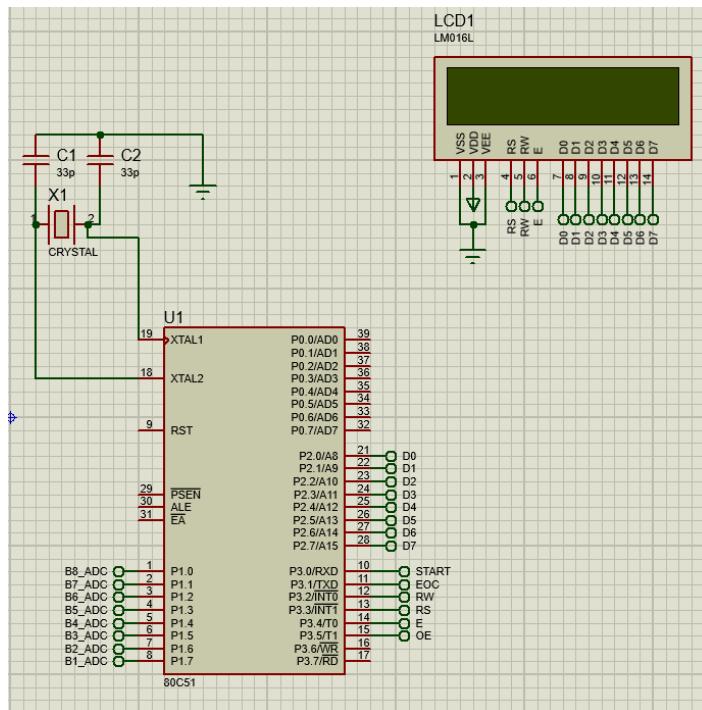
```
void conversie(){
    startc =1; //pornim conversia
    delay(3); //3ms delay
    startc =0; //start = ca sa facem un semnal dreptunghiular (impuls)
```

```
//asteptam eoc
while(!eoc){
    oe=1;
}
}
```



Microcontroller-ul ales a fost Philips 80C51, datorita temperaturilor la care functioneaza (0°C ; $+70^{\circ}\text{C}$). Aceasta este de tip low voltage si functioneaza de la o tensiune de intrare (2.7 V–5.5 V).

Aceste 2 componente sunt conectate intre ele în felul următor:



Consideram portul P2 ca fiind port de ieșire (initializat cu 0x00) și P1 port de intrare (initializat cu 0xFF). Pe intrările XTAL1 și XTAL2, punem un cristal cu frecvență de 11.0592MHz, ca și clock extern, pentru a putea porni clock-ul intern al lui 8051.

Portul 2 trimite datele convertite de către microcontroller la LCD ca acesta să poată fi comandat/să se poată scrie date pe el.

Portul 1 primește numărul de la ADC.

Pe pinul 3.0 și 3.1 punem comanda de start și EOC pentru conversia din ADC. Pe pinul 3.2, 3.3 și 3.4 punem cele 3 comenzi pentru LCD. Adică RS, RW, și EN.

```
void initializareLCD() {
    comandaLCD(0x38); //5x7 matrix
    comandaLCD(0x0C); //display on cursor off
    comandaLCD(0x80); //prima linie, prima pozitie
    comandaLCD(0x0E); |
}
```

Comenzile pentru initializare LCD-ului.

```
void delay(unsigned int itime) {
    unsigned int i,j;
    for(i=0;i<itime;i++)
        for(j=0;j<1275;j++);
}
void display(char *p) {
    for(*p=0;*p!='\0';*p++) {
        scrieLCD(*p);
    }
}
```

Funcțiile pentru delay, și afișarea unui sir de caractere (string).

```

1 void comandaLCD(unsigned char date) {
2     //COMANDAM LCD-UL
3     //checkbusy();
4     ldata = date;
5     rs =0;
6     rw=0;
7     en =1;
8     delay(3);
9     en=0;
0     return;
1 }
2 void scrieLCD(unsigned char date){ //scriem mesaj pe LCD
3     //checkbusy();
4     ldata = date;
5     rs=1;
6     rw=0;
7     en=1;
8     delay(3);
9     en=0;
0 }

```

Funcțiile pentru comanda și scriere pe LCD.

4.2 Modul de găndire al afișărilor

```

3 }
4 }
5 void LCD_DATA_ADC(unsigned char nr) //functia primeste ca parametru
6 //valoarea citita de la adc
7 {
8     int n, k=0;
9     int nr_2 = 2*nr/12.5 -0.5;
0     while(nr_2>0)
1     { //daca numarul citit, val , este mai mare decat 0
2         TAB[k]=nr_2%10;
3         nr_2=nr_2/10;
4         k++;
5     }
6     k--;
7     /*Vom parcurge vectorul invers pentru a afisa valoarea*/
8     comandaLCD(0x06);
9     for (n=k;n>=0;n--) {
0     /*Vom aduna 30h pentru a converti valoarea in cod ASCII*/
1     scrieLCD(TAB[n]+0x30);
2
3     }
4     scrieLCD(0xDF);
5     display("C");
6 }
7

```

Functia pentru afisarea temperaturii

Declaram o variabila globală TAB în care se vor stoca numerele prelucrate. Aceasta funcție primește ca parametru valoarea citita de la ADC. Cat timp numărul este mai mare decât 0 se

executa codul din while. De exemplu daca avem numărul 1C, in TAB[0] o sa avem 0x0C iar numarul devine 0x01. Incrementam contorul k. La următoarea iteratie nr>0 deci se executa din nou while. TAB[1] o sa devina 1, iar numărul o sa devina 0, deci se va ieși din while. In TAB avem momentan stocate următoarele date:

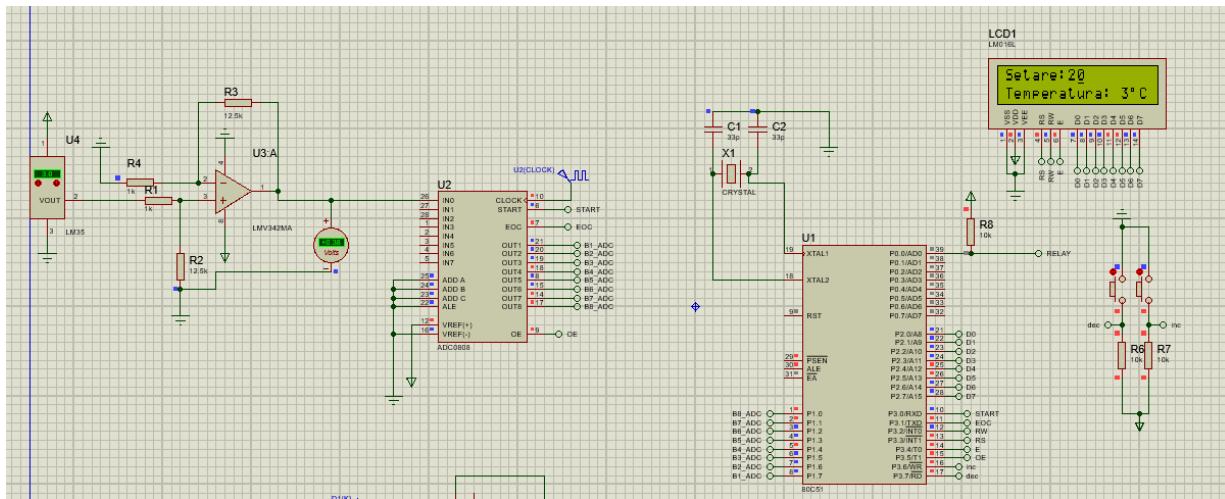
Tab[0]	Tab[1]
0x0C	0x01

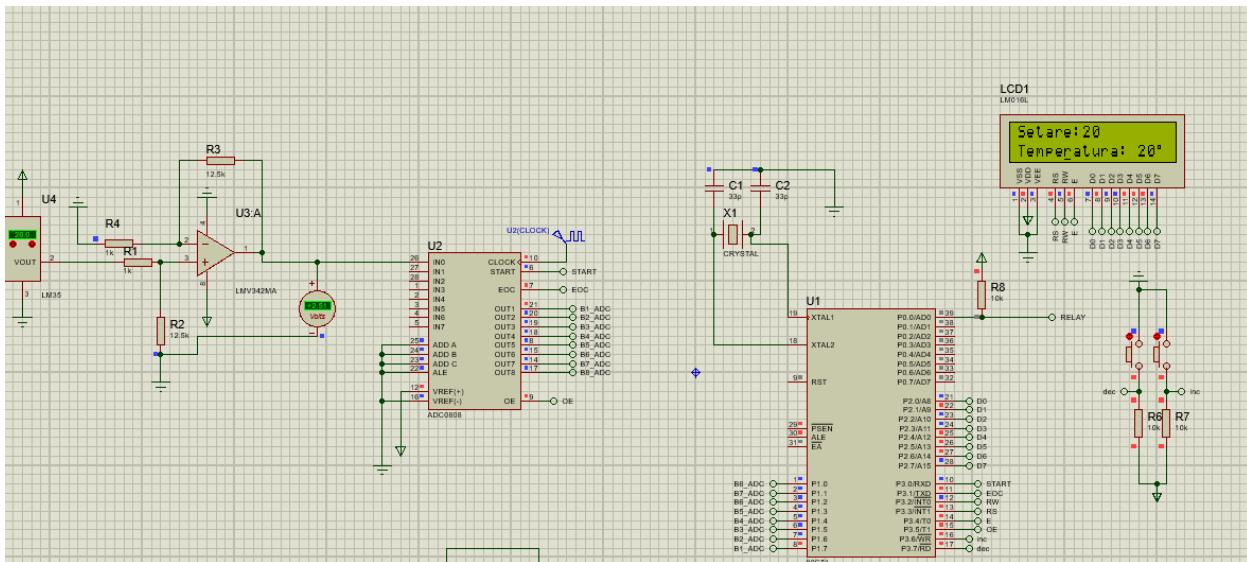
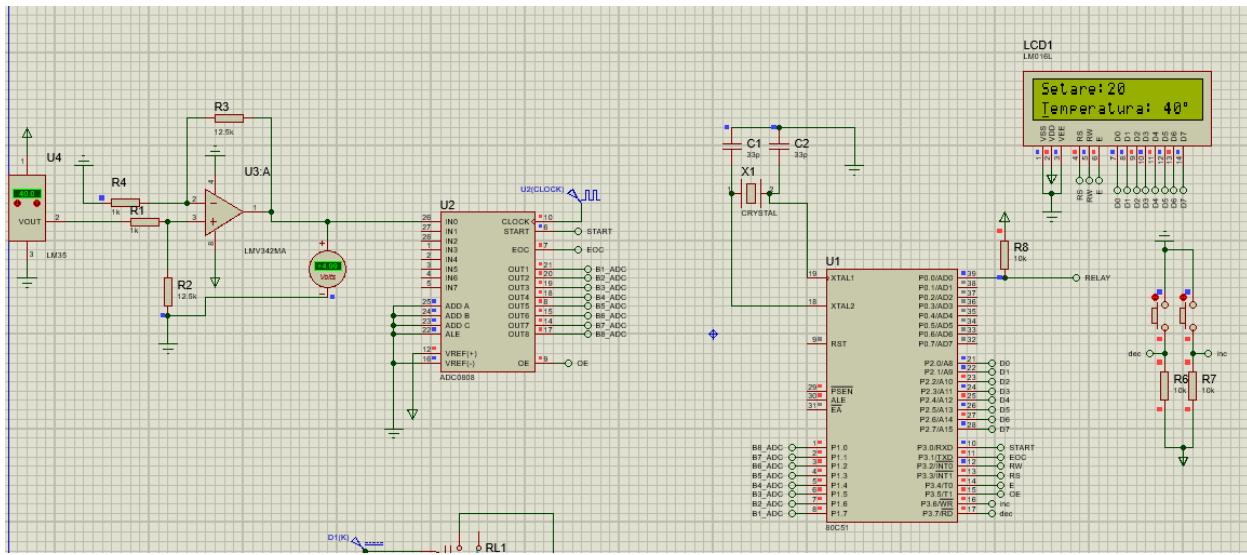
Cu următorul FOR vom parcurge invers tabloul, pentru a afisa pe LCD temperatura.

Initial n=1, deci scriem pe LCD 1+0x30 (ASCII), apoi n devine 0 si scriem C+0x30.

Apelam functiile de scriere pe LCD. 0xDF este caracterul pentru grad in ASCII.

Display(,,C") pentru a afisa Celsius.





```
#include <reg51.h>
```

```
sfr ldata = 0xA0;
```

sbit rw = P3^2

sbit rs = P3^3;

```
sbit en = P3^4;
```

sbit startc = P3^0;

sbit eos = P3^1;

```
sbit oe = P3^5;
```

```

sbit pinClock=P3^6;

void comandaLCD(unsigned char date);

//void checkbusy(); // nu merge

void delay(unsigned int itime);

void display(char *p);

void initializareLCD();

void conversie();

void scrieLCD(unsigned char date);

void LCD_DATA_ADC(unsigned char nr);

//void convert_display(unsigned char value);

signed int TAB[8];

void main(){

    //COMENZI DE INITIALIZARE

    unsigned char valoare_citita;

    P2 = 0x00;      //initializam P1 ca port de intrare de la ADC
    P1 = 0xFF;      //initializam P2 ca port de iesire spre LCD
    initializareLCD();

    //conversie valori din ADC
    oe=1;
    delay(2);// lasam 2 ms
    display("Temperatura: ");

    while(1){

        conversie(); //conversia valorilor analogice la digitale
        delay(5);
}

```

```

        comandaLCD(0xC0);

    comandaLCD(0x06);

        valoare_citita = P1;

    LCD_DATA_ADC(valoare_citita);

}

```

```

void initializareLCD(){

    comandaLCD(0x38); //5x7 matrix

    comandaLCD(0x0C); //display on cursor off

    comandaLCD(0x80); //prima linie, prima pozitie

    comandaLCD(0x0E);

}

```

```

void conversie(){

    startc =1; //pornim conversia

    delay(3); //1ms delay

    startc =0; //start = ca sa facem un semnal dreptunghiular (impuls)

    //asteptam eoc

    while(!eoc){

        oe=1;

    }

}

```

```

void comandaLCD(unsigned char date){

    //COMANDAM LCD-UL

    //checkbusy();

    ldata = date;

```

```

rs =0;
rw=0;
en=1;
delay(3);
en=0;
return;
}

void scrieLCD(unsigned char date){ //scriem mesaj pe LCD
//checkbusy();
ldata = date;
rs=1;
rw=0;
en=1;
delay(3);
en=0;
}

//void checkbusy(){
//    busy =1;
//    rs = 0;
//    rw=1;
//    while(busy==1){
//        en = 0;
//        delay(50);
//        en=1;
//    }
//}

void delay(unsigned int itime){
    unsigned int i,j;
}

```

```

for(i=0;i<itime;i++)
    for(j=0;j<1275;j++);
}

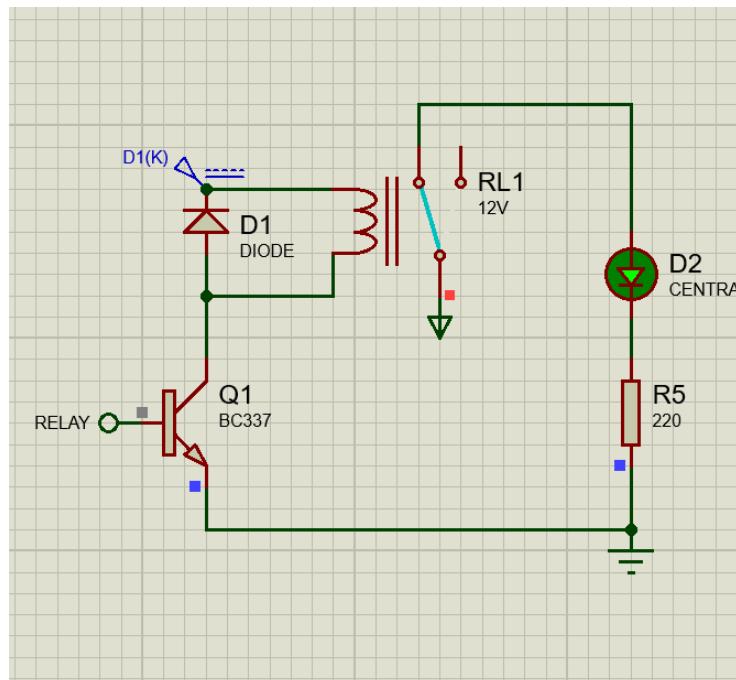
void display(char *p){
    for(*p=0;*p!='\0';*p++){
        scrieLCD(*p);
    }
}

void LCD_DATA_ADC(unsigned char nr) //functia primeste ca parametru
//valoarea citita de la adc
{
    int n, k=0;
    int nr_2 = 2*nr/12.5 -0.5;
    while(nr_2>0)
    { //daca numarul citit, val , este mai mare decat 0
        TAB[k]=nr_2%10;
        nr_2=nr_2/10;
        k++;
    }
    k--;
    /*Vom parurge vectorul invers pentru a afisa valoarea*/
    comandaLCD(0x06);
    for (n=k;n>=0;n--) {
        /*Vom aduna 30h pentru a converti valoarea in cod ASCII*/
        scrieLCD(TAB[n]+0x30);
    }
    scrieLCD(0xDF);
    display("C");
}

```

5 Elementul de execuție, afișarea setării cu butoane

Definiție: Releul este o componentă electronică, un dispozitiv, care produce anumite modificări (cum ar fi închiderea și deschiderea unui circuit) pe baza unui parametru care variază (precum tensiunea electrică aplicată), permitând controlarea unui curent de intensitate mare cu ajutorul unui curent de intensitate mică.



Circuitul din Proteus pentru oprirea centralei.

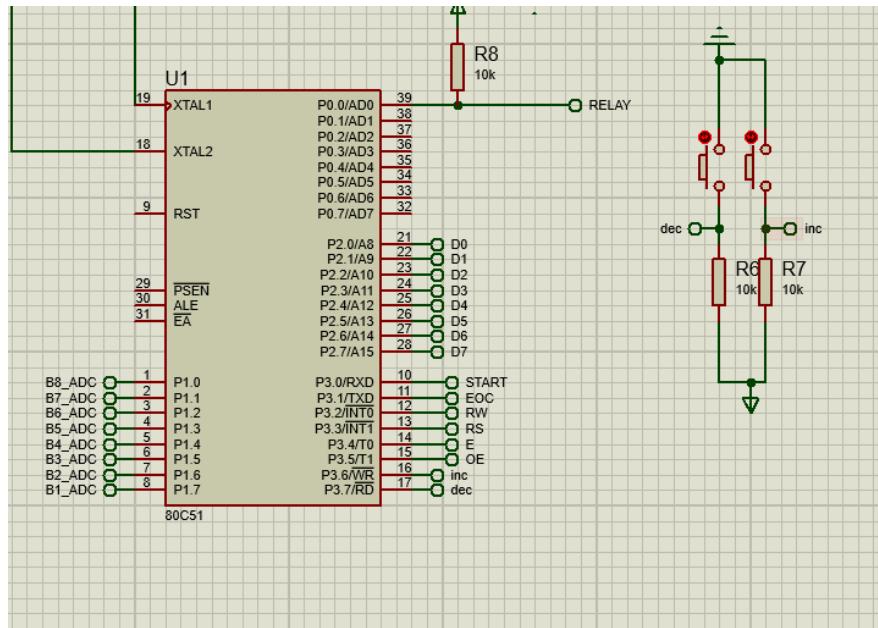
Modul de funcționare al circuitului:

Când în baza tranzistorului bipolar aplicăm starea 1 logic, acesta va permite trecerea curentului prin bobină, astfel se creează un câmp magnetic care va atrage releul în poziția ON, astfel centrala sugerată prin LED-ul de culoare verde, va porni. Când se aplica 0 logic, tranzistorul se închide, și dispare câmpul magnetic.

Dioda de fugă are rolul de a proteja tranzistorul când se face comutarea din ON în OFF a releeului.

Setarea temperaturii

Inițial vom avea temperatura setată la 20 grade C. Cu ajutorul celor 2 butoane vom trimite un bit de 0 către microcontroller, dacă acestea sunt apăsatate, și în rest ele vor avea starea 1 logic.



Circuitul de pornire centrală, și butoanele pentru setări.

```

void checkSetare(void) {
    if(!increment) {
        setare++;
        scrieLCD(setare/10 +0x30);
        scrieLCD(setare%10 +0x30);
    }
    else if(!decrement) {
        setare--;
        scrieLCD(setare/10 +0x30);
        scrieLCD(setare%10 +0x30);
    } else
    {
        scrieLCD(setare/10 +0x30);
        scrieLCD(setare%10 +0x30);
    }
}

```

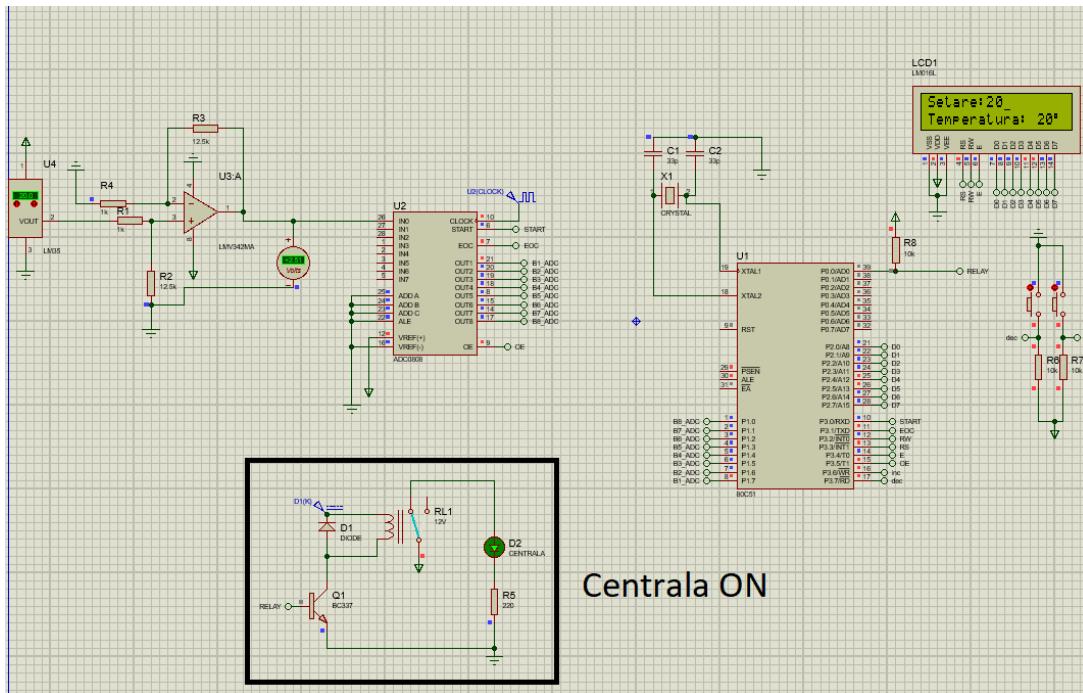
Codul în C51 pentru setarea temperaturii.

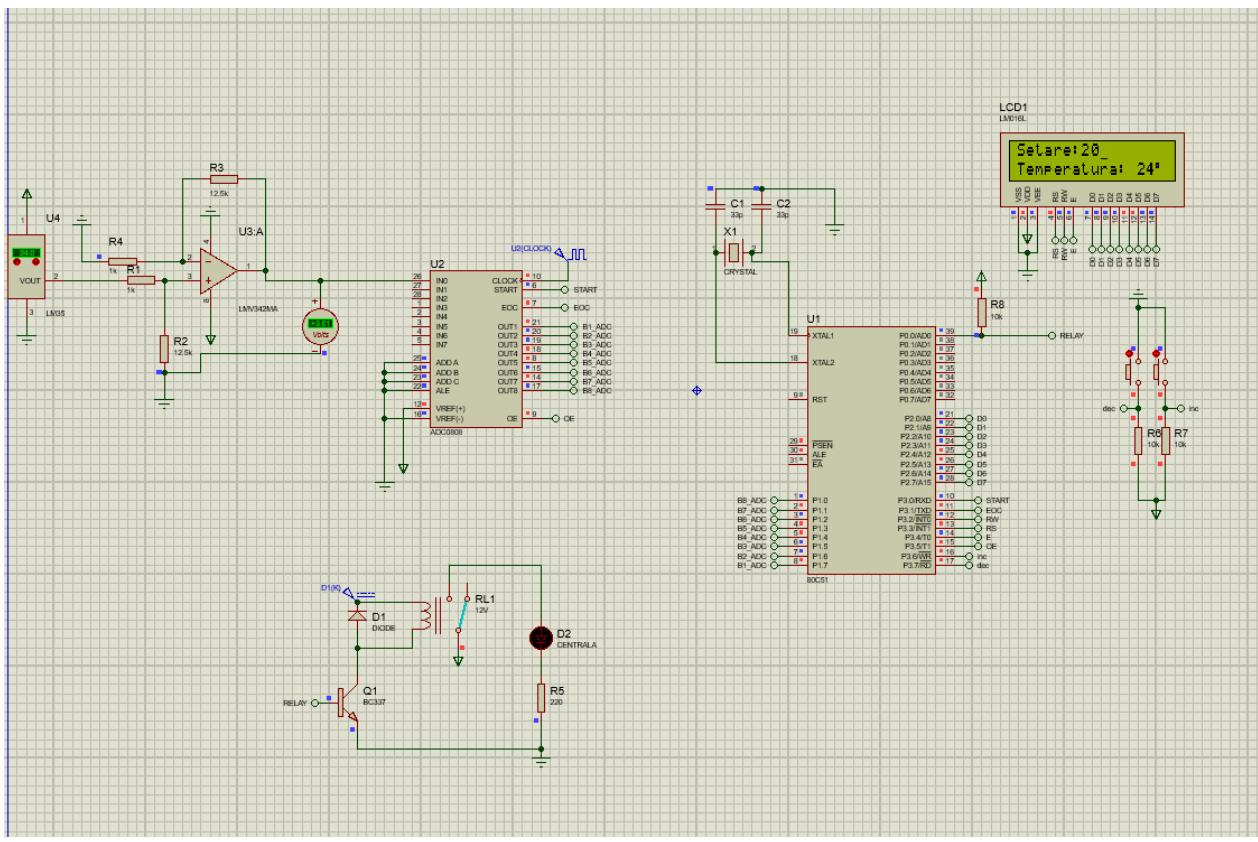
Deoarece când apăsăm butonul se trimite pe microcontroller pe porturile 3.6 si 3.7 un bit de 0, vom verifica cu mai multe if-uri dacă butonul este apăsat sau nu la apelarea funcției. Dacă este apăsat buton atunci vom incrementa setarea, și vom afișa noua setare, altfel vom decrementa setarea și vom afișa din nou setarea decrementată.

Verificăm dacă setarea este mai mare decât temperatura senzorului. Dacă este mai mare, atunci vom porni centrala. Dacă nu, o vom opri.

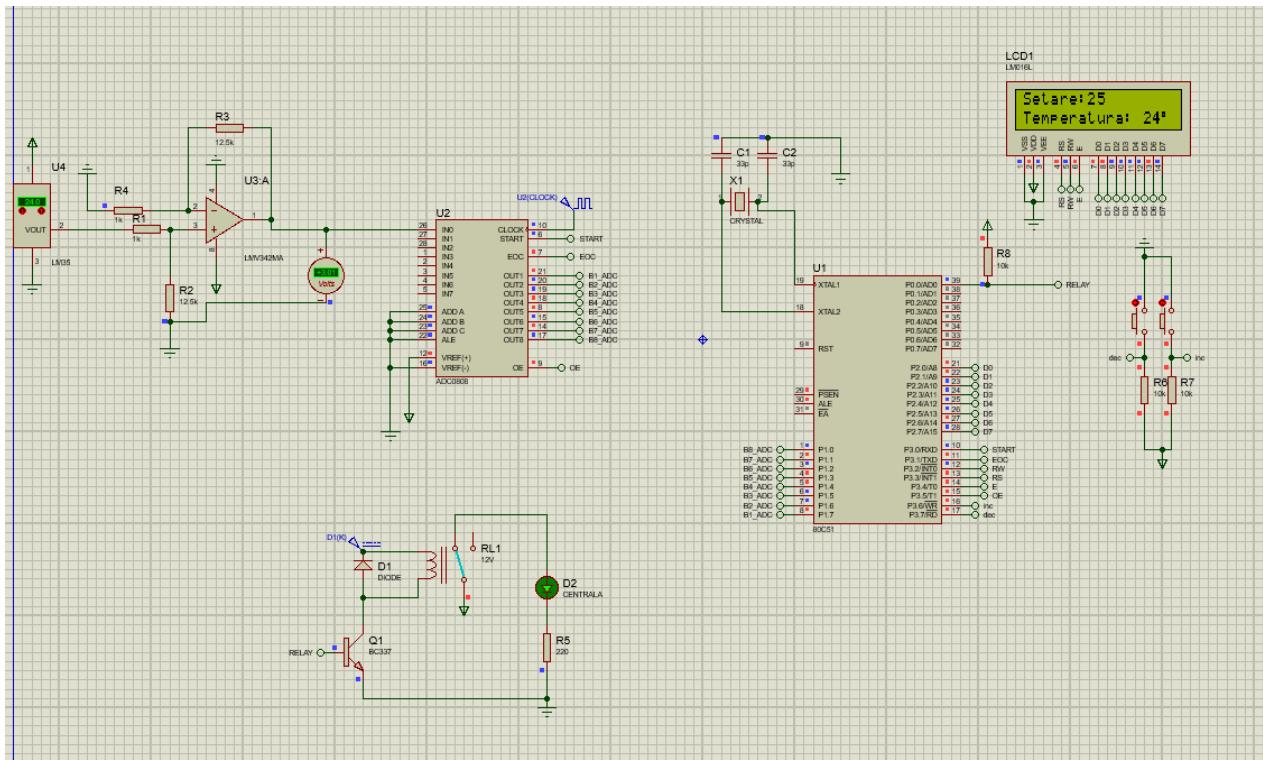
```
void LCD_DATA_ADC(unsigned char nr) //functia primeste ca parametru
//valoarea citita de la adc
{
    int n, k=0;
    int nr_2 = 2*nr/12.5;
    if(nr_2>setare){
        releu =0;
    } else{
        releu=1;
    }
}
```

Testare prin simulare în Proteus:





Temperatură 24, setare 20 Centrala OFF.



Setare 25, temperatură 24 Centrala ON

```

int Setare_25;
void main(){
    //COMENZI DE INITIALIZARE
    unsigned char valoare_citita;
    P2 = 0x00;           //initializam P1 ca port de intrare de la ADC
    P1 = 0xFF;           //initializam P2 ca port de iesire spre LCD
    initializareLCD();

    //conversie valori din ADC
    oe=1;
    delay(2); // lasam 2 ms

    while(1){
        display("Setare:");
        checkSetare();
        conversie(); //conversia valorilor analogice la digitale
        delay(5);
        comandaLCD(0xC0);
        comandaLCD(0x06);
        display("Temperatura: ");
        delay(1);
        valoare_citita = P1;
        LCD_DATA_ADC(valoare_citita);
        comandaLCD(0x80);
        comandaLCD(0x06);

    }
}

```

Programul MAIN

BIBLIOGRAFIE

- <https://repository.tudelft.nl/islandora/object/uuid:d452e5d3-355e-4a4c-809e-522989db4640/datastream/OBJ/download>
-
- <https://www.ti.com/lit/ds/symlink/lmv341-n.pdf?HQS=dis-mous-null-mousermode-dsf-pf-null->
- <https://www.ti.com/lit/ds/symlink/lmv341-n.pdf?HQS=dis-mous-null-mousermode-dsf-pf-null->
- <https://embedcenter.wordpress.com/ece-study-centre/display-module/lcd-16x2-lm016/>
- Curs LCD
- <https://www.electronicshub.org/interfacing-16x2-lcd-8051/>
- <https://www.electronicwings.com/8051/lcd16x2-interfacing-in-8-bit-with-8051>