

MICROCONTROLERE

TERMOSTAT DE CAMERA

Student:
Roșca David-Sorin

Coordonator:
Mirela Olteanu

CUPRINS

1. Tematica Proiectului	3
2. Schema Bloc	3
3. Senzori de Temperatura	4
3.1. Ce este temperatura?	4
3.2. Ce este un senzor de temperatură?	4
3.3. Tipuri de senzori de temperatură	5
3.3.1 Senzori rezistivi-RTD (resistance temperature detector)	5
3.3.2. Senzor de temperatură cu Termocopluri	6
3.3.3 Senzor de temperatură cu termistoare	7
3.3.4. Senzori Semiconductori	8
3.3.5. Alegerea senzorului de temperatură	9
4. Circuit de conditionare senzor de temperatură	10
4.1. Componente Utilizate	10
4.1.1. LM35	10
4.1.2 AD8541AS	10
4.1.3 ADC0808	11
4.2. Dimensionarea Componentelor	13
4.3. Simularea circuitului la 40°C în Proteus	14
4.4. Simularea circuitului la 8°C în Proteus	15
5. Display-ul	16
5.1. Ce este un display?	16
5.2. Tipuri de Display-uri	16
5.3. Alegerea tipului de afisaj electronic	19
5.4. Simulari si modul de functionare	21
5.4.1 Instructiuni	22
6. Alegerea Microcontrolerului și programarea LCD-ului în C și asm	25
6.1 Ce este un microcontroler?	25
6.2 Structura memoriei la microcontrolerul 8051	28
6.3 . Familii de microcontrolere	29
6.4 Pinii microcontrolerului si alegerea acestuia	34
6.5 Programarea microcontrolerului in ASM	36
6.5.1 Simularea in Proteus la 8 °C=>00h pe LCD	39

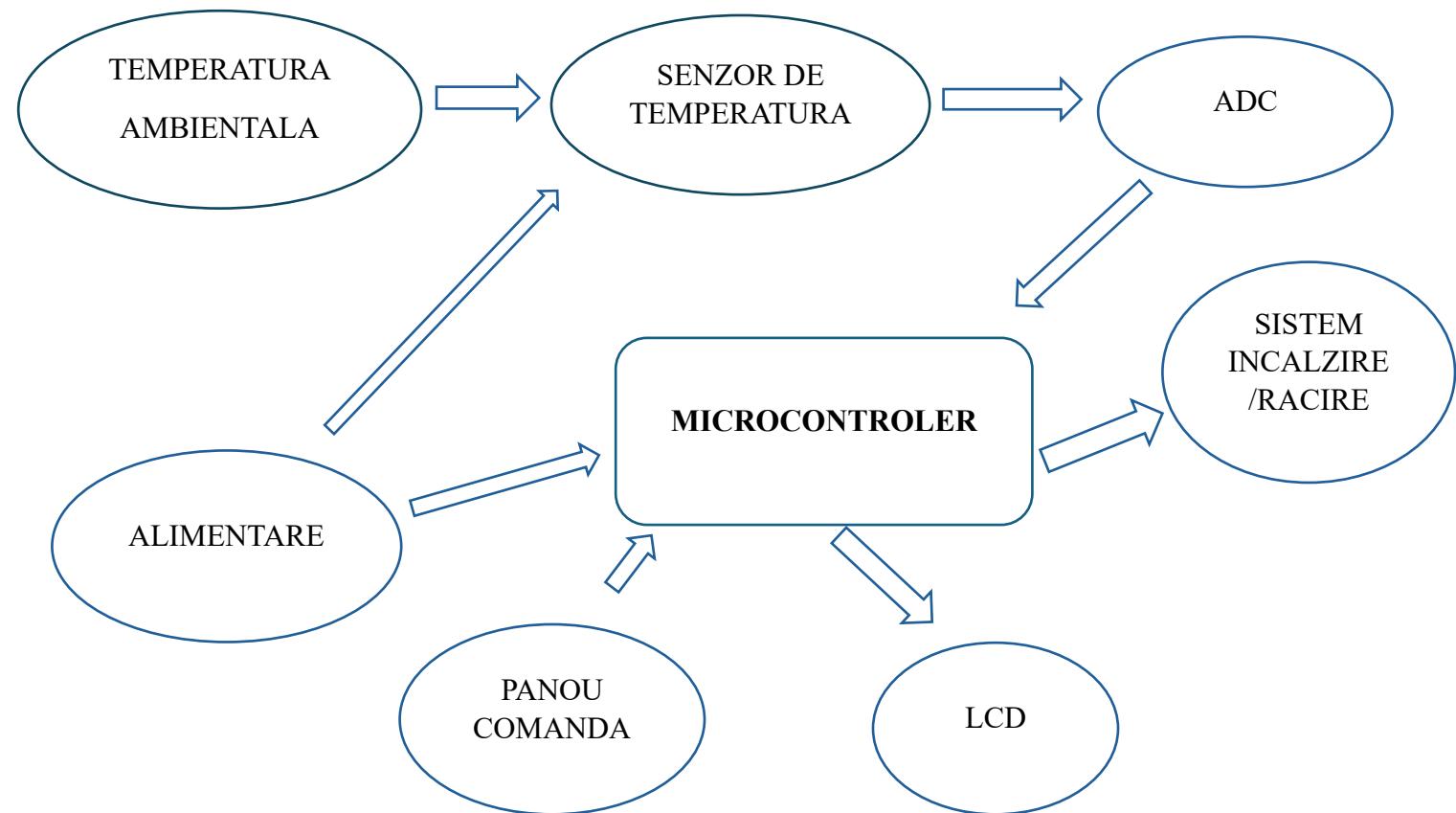
6.5.2 Simularea in Proteus la 40°C=>FFh pe LCD	39
6.6 Programarea microcontrolerului in C	40
6.6.1 Simularea in Proteus la 8 °C=>00h pe LCD.....	43
6.6.2 Simularea in Proteus la 40°C=>FFh pe LCD	43
7. Tastatura	44
7.1 Consideratii teoretice	44
7.2 Schema PROTEUS.....	44
8. Releul.....	45
8.1 Consideratii teoretice	45
8.2 Functionarea releului.....	45
8.3 Schema in Proteus	47
8.4 Simulare a releului in pozitia ON.....	48
8.5 Simulare a releului in pozitia OFF	48
9. Circuitul Final în Proteus	49
9.1 Simulare pentru centrala ON	50
9.2 Simulare pentru Centrala OFF	51
9.3 Programarea Microcontrolerului în Limbajul C: Afisarea Zecimală a Temperaturii	52
9.4 Programarea Microcontrolerului în Limbajul ASM: Afisarea Zecimală a Temperaturii	57
10. Bibliografie	66

1.Tematica Proiectului

Proiectul constituie dintr-un dispozitiv capabil să măsoare temperatura ambientală a unei încăperi și să mențină o temperatură constantă într-un anumit interval și să semnalizeze dacă aceasta depășește limitele impuse, deci tema proiectului este un **termostat de cameră**.

Termostatele de cameră sunt regulatori de temperatură ce comandă pornirea/oprirea sistemului de încălzire. Temperatura aerului din încăpere este sesizată de către termostat și când aceasta scade la o valoare sub cea reglată lasă instalația de încălzire să funcționeze, iar când temperatura aerului atinge valoarea setată stabilită închide echipamentul de încălzire pe care îl comandă. Funcționează pe principiul unui dispozitiv cu "buclă închisă", deoarece se dorește minimizarea erorilor dintre temperaturile dorite și cele măsurate. Numele dispozitivului vine din limba greacă unde thermos înseamnă încălzit, fierbinte iar statos – a menține, a seta, a regla.

2.Schema Bloc



3.Senzori de Temperatura

3.1.Ce este temperatura?

Temperatura este un parametru fundamental de stare care caracterizează starea termică a unui corp, mai exact, starea de echilibru termodinamic. Măsurarea temperaturii se bazează pe diferite fenomene și efecte fizice, în care, modificarea temperaturii determină modificări ale unor proprietăți sau caracteristici ale materialelor, ca: variația dimensiunilor geometrice, variația rezistenței electrice, apariția unei tensiuni electromotoare de-a lungul joncțiunii a două metale etc. Acuratețea procesului de măsurare a temperaturii este foarte importantă pentru cele mai multe aplicații de control a diferitelor procese tehnologice.

3.2.Ce este un senzor de temperatură?

Un senzor de temperatură este un dispozitiv care colectează date despre temperatură de la o sursă particulară și convertește datele într-o formă inteligibilă pentru un dispozitiv sau un observator.

Un senzor de temperatură este alcătuit din două tipuri fizice de bază:

- ✓ **cu contact** - Aceste tipuri de senzori de temperatură trebuie să fie în contact fizic cu obiectul detectat și să utilizeze conductiona pentru a monitoriza schimbările de temperatură. Ele pot fi folosite pentru a detecta solide, lichide sau gaze într-o gamă largă de temperatură.
- ✓ **non-contact** - Aceste tipuri de senzori de temperatură folosesc convecția și radiația pentru a monitoriza schimbările de temperatură. Acestea pot fi folosite pentru detectarea lichidelor și a gazelor care emit energie radiantă, pe măsură ce căldura crește și se răcesc în curenții de convecție.

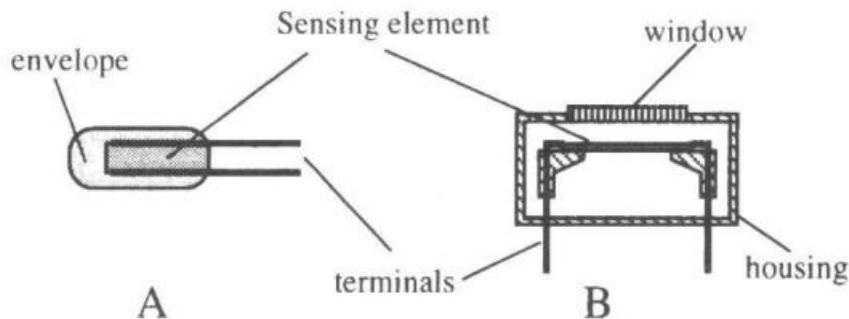


Figure 1. Arhitectura unui senzor de temperatură de tip: A contact, B noncontact

Exista și:

- ✓ **senzori analogici**, pentru care semnalul de ieșire este în permanență proporțional cu mărimea fizică de intrare;
- ✓ **senzori numerici (digitali)**, la care semnalul de ieșire poate lua numai un număr limitat de valori discrete, care permit cuantificarea semnalului fizic de intrare.

3.3. Tipuri de senzori de temperatură

3.3.1 Senzori rezistivi-RTD (resistance temperature detector)

Un senzor rezistiv de temperatură este un dispozitiv care poate fi utilizat pentru a măsura temperatura, prin măsurarea modificărilor rezistenței electrice ale unui material în funcție de temperatura la care se află acesta. De obicei, acești senzori sunt construiți dintr-un material cu coeficient termic de rezistență electrică pozitiv, cum ar fi platina sau nichelul.

Principiul de funcționare al unui senzor rezistiv de temperatură se bazează pe faptul că, odată cu creșterea temperaturii, atomii din material se mișcă mai rapid și se ciocnesc între ei, ceea ce crește rezistența electrică a materialului. În general, măsurarea rezistenței electrice se face prin utilizarea unui circuit electronic care aplică o tensiune cunoscută la senzor și apoi măsoară intensitatea curentului care curge prin acesta.

În ceea ce privește materialele de realizare, platina se apropie cel mai mult de caracteristicile unui termorezistor ideal: stabilitate pe termen lung, durabilitate, poate fi realizat cu puritate mare 99,99 %, inactivă chimic

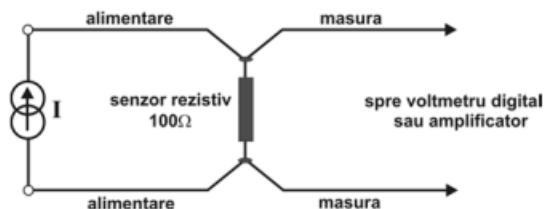


Fig.3.1:schema electrică

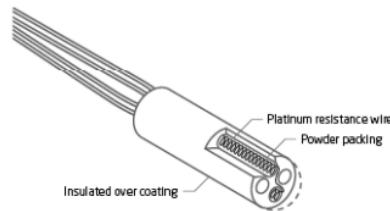


Fig.3.2: structura unui senzor de tip RTD

Tabel 3.3.1.1: Avantaje si Dezavantaje la RTD

Avantaje	Dezavantaje
-plajă mare de temperatură	-un răspuns lent în timp
-sensibilitate mai mare decât la termocupluri – termorezistoarele de Pt și Cu dă răspuns mai linear decât termocuplurile;	-încălzirea termorezistenței la trecerea unui curent de măsură afectează precizia de măsurare

3.3.2. Senzor de temperatură cu Termocupluri

Termocuplurile sunt senzori de temperatură care măsoară temperatura prin măsurarea tensiunii electrice generate de un cuplu termoelectric format din două metale diferite. Principiul de funcționare al termocuplurilor se bazează pe efectul Seebeck, care indică faptul că o diferență de temperatură între două metale diferite va produce o tensiune electrică între acestea.

Termocuplurile sunt alcătuite din două fire de metale diferite (de exemplu, cupru și constantan) conectate la două puncte de măsurare, denumite joncțiuni. Una dintre joncțiuni este expusă la temperatura de măsurat, iar cealaltă joncțiune este menținută la o temperatură constantă, de referință. Diferența de temperatură între cele două joncțiuni va genera o tensiune electrică în circuitul termocuplului.

Valorile tensiunii electrice generate de termocupluri sunt foarte mici (de obicei în ordinea milivoltilor), astfel încât acestea trebuie amplificate și convertite într-un semnal de temperatură utilizabil. Acest lucru se realizează de obicei cu ajutorul unui amplificator de termocuplu și a unui circuit de conversie analog-digital.

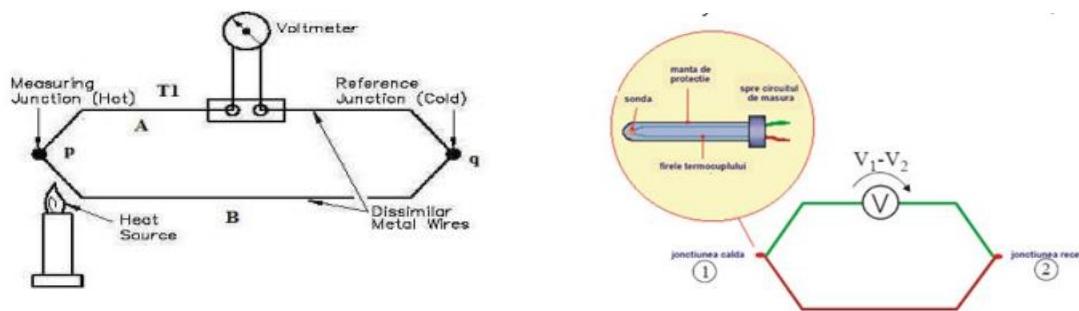


Fig. 3&4 : Funcționarea termocoplului

Tabel 3.3.2.1: Avantaje si Dezavantaje la Termocopluri

Avantaje	Dezavantaje
- precizie foarte mare, liniaritate buna	- valori mici ale tensiunii generate/grad Celsius
- repetabilitate in timp a măsurătorilor	-necessita contact fizic cu obiectul măsurat
- plajă foarte mare de temperatură (- 265°C pana la +2300 °C)	

3.3.3 Senzor de temperatura cu termistoare

Termistorii sunt senzori de temperatură care se bazează pe variația rezistenței electrice în funcție de temperatura la care sunt expoși. Există două tipuri principale de termistori: termistorii NTC (Negative Temperature Coefficient) și PTC (Positive Temperature Coefficient).

- ✓ Termistorii NTC au o rezistență electrică care scade odată cu creșterea temperaturii. Acest efect se datorează modificărilor în densitatea electronilor și a mobilității ionice în materialul din care este fabricat termistorul. În general, o creștere cu un grad Celsius a temperaturii poate duce la o scădere de 2-3% a rezistenței electrice a unui termistor NTC.
- ✓ Termistorii PTC au o rezistență electrică care crește odată cu creșterea temperaturii. Această proprietate se datorează modificărilor în structura cristalină a materialului și în mobilitatea ionilor. De obicei, o creștere cu un grad Celsius a temperaturii poate duce la o creștere de 2-3% a rezistenței electrice a unui termistor PTC.

Deoarece termistorii sunt senzori pasivi, aceștia au nevoie de o sursă externă de alimentare pentru a fi utilizat într-un circuit. Pentru a măsura temperatura cu ajutorul unui termistor, este necesar să se cunoască relația dintre variația rezistenței electrice și temperatura.

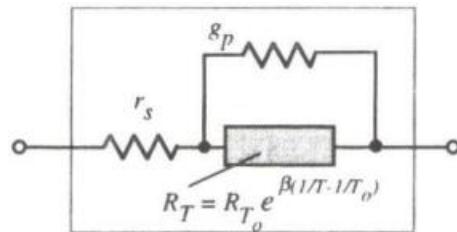


Fig. 5:Circuitul echivalent al unui termistor

Tabel 3.3.3.1: Avantaje si Dezavantaje la Termistoare

Avantaje	Dezavantaje
- termistorul este compact, durabil și mai puțin costisitor	- variația neliniara cu temperatura a rezistenței termistorilor
- bună repetabilitate și o rezoluție fină pe domenii mici de temperatură	

3.3.4. Senzori Semiconductori

Senzorii semiconductori sunt dispozitivele care sunt sub forma de circuit integrat. Acesteia sunt clasificati sub diferite tipuri: senzor de temperatură cu ieșire în curent, senzor de temperatură cu ieșire în tensiune, senzor de temperatură de siliciu cu ieșire a rezistenței, senzori de temperatură a diodei și senzor digital de temperatură de ieșire. Senzorii de temperatură semiconductori actuali oferă o liniaritate ridicată și o precizie ridicată pe un domeniu de operare de aproximativ -55°C până la $+150^{\circ}\text{C}$. Cu toate acestea, senzorii de temperatură AD590 și LM35 sunt cei mai populari senzori de temperatură.

3.3.5. Alegerea senzorului de temperatură

Pentru a selecta senzorul ideal aplicatiei noastre, trebuie să parcurgem și să înțelegem aplicațiile și caracteristicile senzorilor. Cu ajutorul lor, vom urmări următoarele aspecte:

- ✓ domeniul de temperatură în care trebuie să lucreze
- ✓ liniaritatea
- ✓ mediul în care senzorul trebuie să activeze pentru a alege un material corespunzător, care să reziste la toți factorii externi timpul de răspuns.

3.3.5.1 Tabel: Comparativ între senzori de temperatură

Proprietate	Termocupla	RTD	Termistor	Semiconductori
Domeniul de temperatură	-184°C - 2300°C	-270°C - 850°C	-75°C - 300°C	-55°C -150°C
Liniaritate	Destul de bună	Buna	Slabă	Ridicată
Sensibilitate	Scazuta	Medie	Foarte ridicată	Medie
Timpul de răspuns	Mediu-spre rapid	Mediu	Mediu pre rapid	Mediu
Stabilitate	Destul de bună	Buna	Slab	Foarte bună
Susceptibil de autoîncalzire?	Nu	Da, minimal	Da	Da, la 0.08°C
Durabilitate	Excețională	Buna	Slabă	Buna
Cost	Cel mai mic	Ridicat	Acceptabil	Mic

3.3.5.2 Tabel: Exemple de senzori de temperatură

Senzor:	Gama de masurare:	Precizie:	Analog/Digital:	Pret:
LM35	-55 °C la 150 °C	0.5 °C	±Analog	6.5 RON
LM135Z	-55 °C la 150 °C	1 °C	±Analog	3.5 RON
LMT85	-50 °C la 150 °C	0.4 °C	±Analog	4.5 RON
AC103J2F	-55°C la 125 °C	1°C	±Analog	4 RON
MAX6504UKN	-55°C la 150 °C	0.5°C	±Analog	6.66 RON
B57703M0103G000	-55°C la 175 °C	2°C	±Analog	20.35 RON
LMT70YFQT	-55°C la 150 °C	0.1°C	±Analog	6.80 RON

Luând în considerare caracteristicile celor 6 senzori prezentati, am ales **senzorul cu semiconductor LM35** deoarece are o precizie bună, un domeniu de temperatură potrivit realizării unui termostat de camera.

4. Circuit de conditionare senzor de temperatură

4.1. Componete Utilizate

4.1.1. LM35

Senzorul de temperatură pe care l-am ales este LM35. Este un senzor de temperatură de precizie, cu ieșire analogică de tensiune, proporțională liniar cu temperatura masurată în grade Celsius. Ieșirea are impedanță de 0.1 ohm și senzorul este calibrat implicit în grade Celsius.

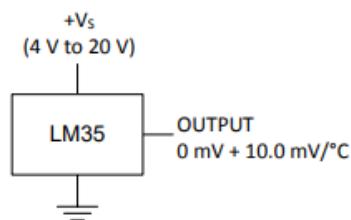


Fig.4.1.1 Senzorul de temperatură LM35

4.1.2 AD8541AS

Amplificatorul utilizat este un AD8541AS deoarece se încadrează în intervalul de temperatură al senzorului și are o tensiune de alimentare de 5V.

Caracteristici AD8541AS:

- Putere mică consumată
- Alimentare de la 2.7V la 5V și +/-5V pentru alimentarea diferențială
- Viteză mare
- Slew rate 30V/us
- Rail to rail input și output
- Frecvență de funcționare până la 80MHz
- Curent la ieșire 1.5mA

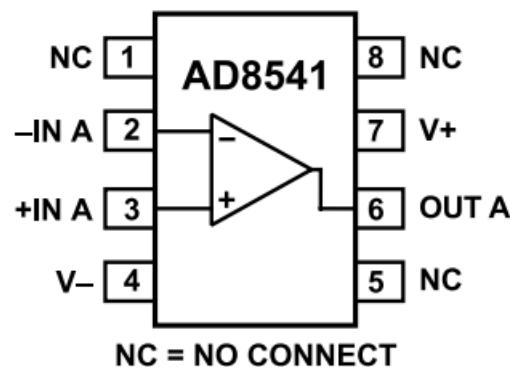


Fig.4.1.2 Amplificatorul AD8031

4.1.3 ADC0808

Convertorul analog – numeric utilizat este ADC0808, deoarece acesta operează pe 8 biți, necesită o tensiune de alimentare de 5V, are o eroare de 0.5LSB și 1.5LSB, timpul de conversie este de 100us și o putere de doar 15mW.

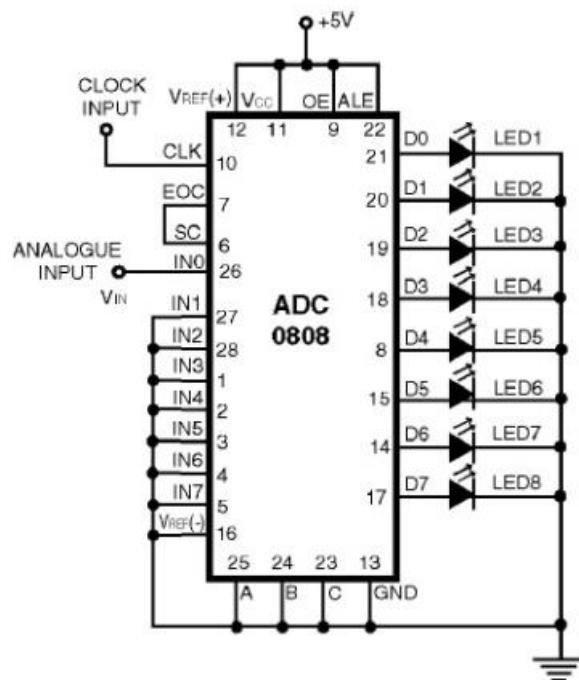


Fig.4.1.3 ADC0808

ADC0808 este un convertor A/D pe 8 biți ce are liniile de date D0-D7. Funcționează pe principiul aproximăției succesive. Are 8 canale de intrare analogice fiecare putând fi selectat prin liniile de adresare A, B, și C. În cazul prezentat intrarea IN0 este selectată prin punerea la masă a liniilor A, B, și C.

În mod obișnuit semnalele de control EOC (end of conversion), SC (start conversion), ALE (address latch enable) și OE (output enable) sunt prelucrate prin intermediul unui microprocesor. Însă, circuitul prezentat este realizat astfel încât să funcționeze continuu fără a folosi un microprocesor. Semnalele de control de intrare OE și ALE sunt active sus și sunt legate la +5Vcc. Semnalul de control de intare SC activ jos devenind 0 inițiază conversia pe frontul descrescător al impulsului, în timp ce semnalul de ieșire devine 1 după încheierea procesului de conversie. Ieșirea EOC este legată la intrarea SC, unde frontul descrescător al ieșirii EOC acționează ca intrare SC pentru a comanda începerea conversiei. Odată conversia începută, EOC devine 1.

La următorul impuls al ceasului EOC devine iarăși 0 și prin urmare SC poate comanda o nouă conversie. Astfel, convertorul furnizează continuu semnale de ieșire digitale pe 8 biți în concordanță cu valoarea instantanee a semnalului de intrare analogic. Valoarea maximă a tensiunii de intrare analogică nu trebuie să depășească nivelul de referință +5V.

Convertorul ADC0808 necesită un semnal de ceas cu o valoare tipică de 550kHz. Pentru a vizualiza semnalul de ieșire au fost folosite cele 8 LED-uri, fiecare fiind conectată la o linie de ieșire. Deoarece, ADC funcționează în mod continuu, acesta afișează semnalul de ieșire de îndată ce este aplicat

semnalul de pe intrare.

4.2. Dimensionarea Componentelor

Calculam A_v și tensiunea de referință:

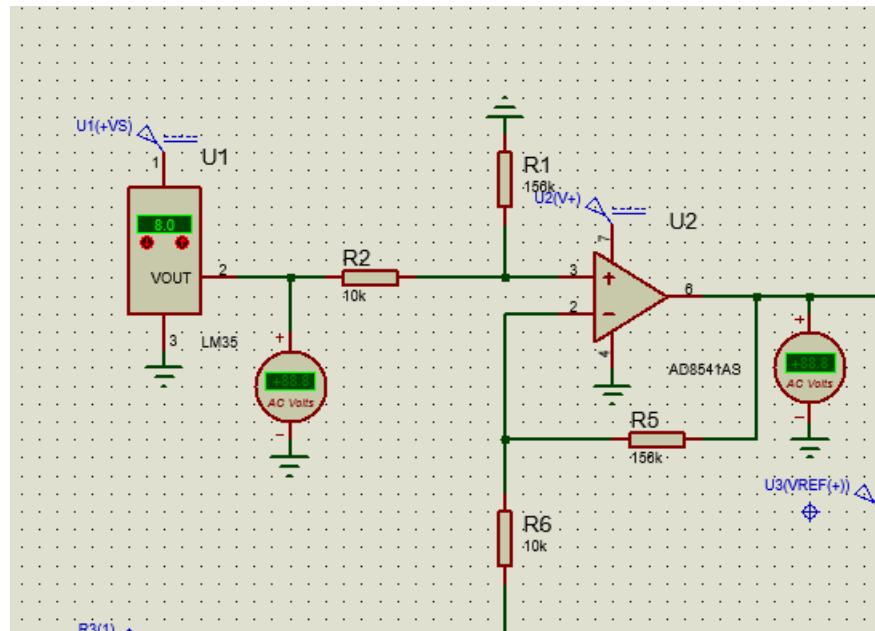


Fig.4.2.1 Amplificatorul diferențial

$$A_v = \frac{5V - 0V}{400mV - 80mV} = 15.6V \quad (1)$$

Valoarea tensiunii de referință: $(400mV - V_{ref}) * 15.6V = 5V \Rightarrow V_{ref} = 80mV$

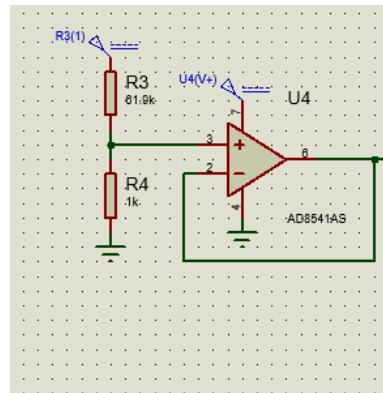


Fig.4.2.2 Divizorul Rezistiv

$$V_o = \frac{R4}{R3} (VA - VB) \quad (2)$$

$$Av = \frac{R4}{R3} \Rightarrow R4 = 15.6R3 \quad (3)$$

$$\text{Aleg } R3=1k \Rightarrow R4=15.6k \quad (4)$$

$$R1=R3=1k \quad (5)$$

$$R2=R4=15.6k \quad (6)$$

$$Vref = \frac{R8}{R8+R7} * VCC \quad (7)$$

$$80mV = \frac{R8}{R8+R7} * 5V \quad (8)$$

$$(R8+R7)*0.08=5R8 \quad (9)$$

$$0.08R8+0.08R7=5R8 \quad (10)$$

$$0.08R7=4.99R8 \Rightarrow R8=1k, R7=61.5k \quad (11)$$

4.3. Simularea circuitului la $40^{\circ}C$ în Proteus

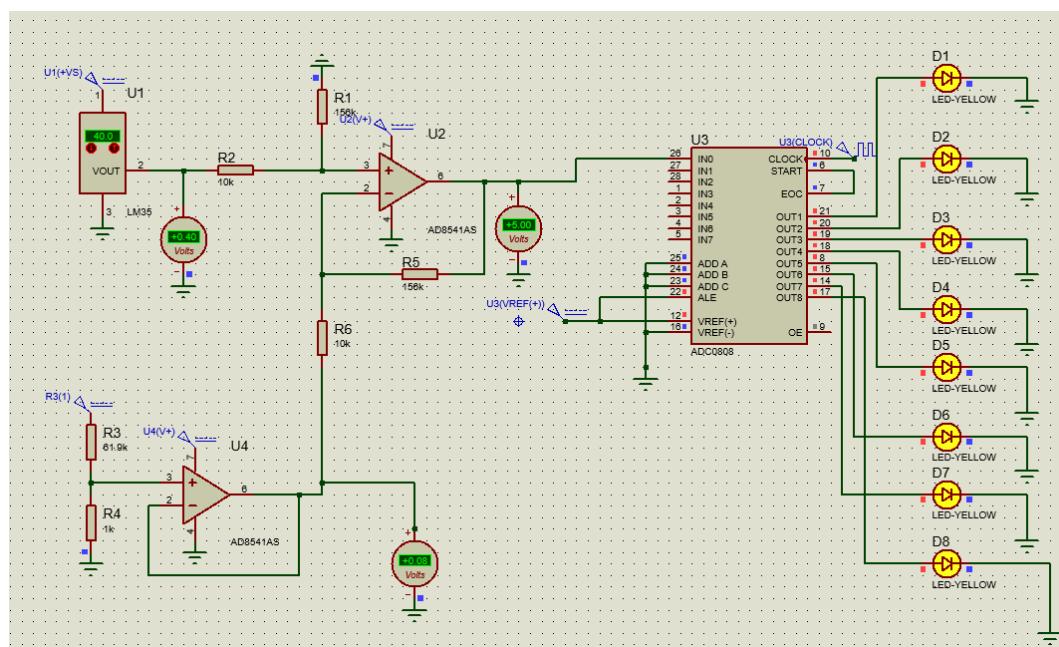


Fig. 4.3.1 Simulare la 40 de grade

Pentru temperatura de 40°C , valoarea digitală afisată de ADC va fi:

$$N=FFh=1111\ 1111b$$

4.4. Simularea circuitului la 8°C în Proteus

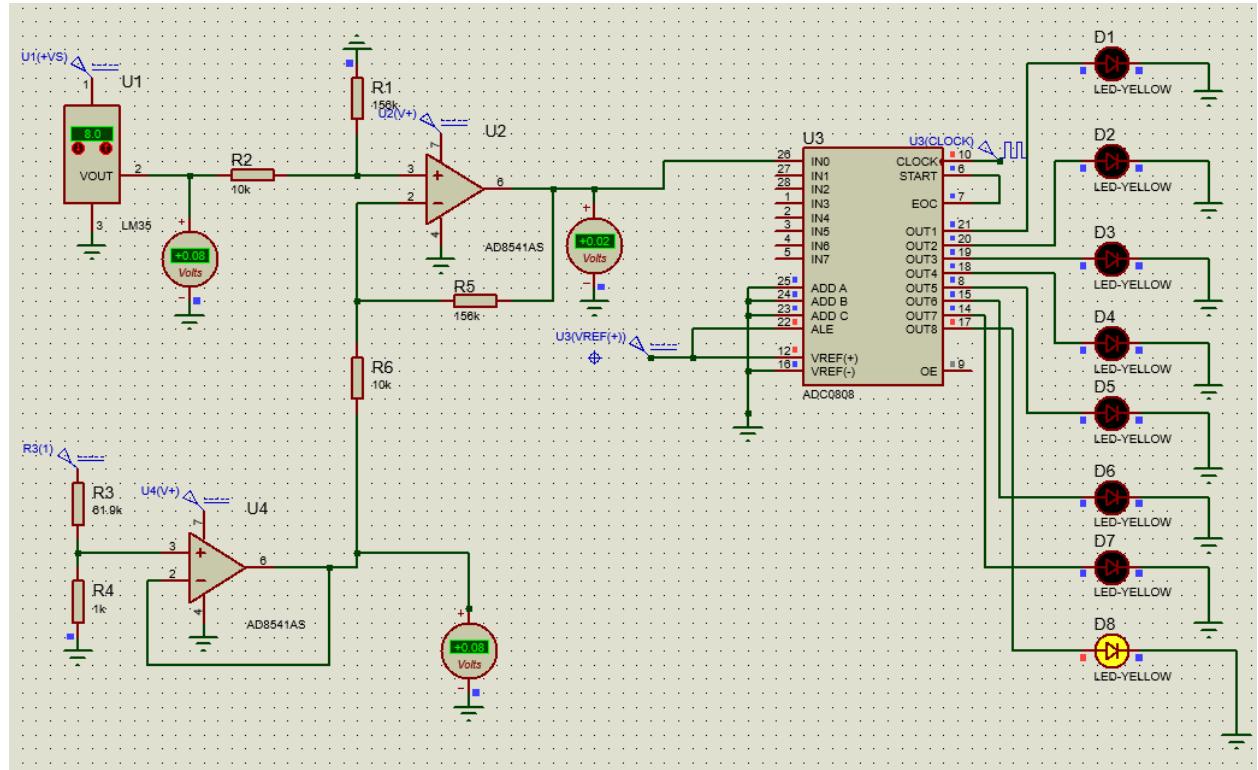


Fig.4.4.1 Simulare la 8 grade

Dupa cum se poate observa in figura de mai sus, pentru temperatura de 8°C , tensiunea de intrare este aproximativ 0V. Cu toate acestea, din cauza tensiunii de offset a amplificatorului AD8541AS, tensiunea de iesire va fi de 0.02V, ceea ce va conduce la aprinderea unuia dintre cele opt Led-uri galbene. Astfel, valoarea digitală a ADC pentru aceasta temperatură va fi:

$$N=01h=0000\ 0001b$$

5. Display-ul

5.1.Ce este un display?

Display-ul este un dispozitiv optoelectric folosit pentru a afișa litere, numere sau imagini, fără a înregistra în permanență, și este considerat un periferic al calculatorului.

5.2. Tipuri de Display-uri

✓ Afisajul electroluminiscent (EL)

Afișajul Electroluminiscent (EL) este un tip de tehnologie de afișare care utilizează un strat subțire de material luminescent pentru a emite lumină atunci când un curent electric este aplicat acestuia.

Acest tip de display este similar cu condensatoarele, diferența constituind-o stratul de fosfor utilizat în afișajele electroluminescente. Este construit din benzi plate, opace, de electrozi, care sunt paralele între ele și acoperite cu un material electroluminiscent (ex: fosforul), urmat, apoi, de alt strat de electrozi, care sunt perpendiculari pe stratul de jos.

Curentul electric excită atomii, lucru care duce la emisia unor radiații sub formă de lumină. Variația nivelului de excitație al atomilor schimbă culoarea afișată pe display. Aceste afișaje sunt monocromatice.

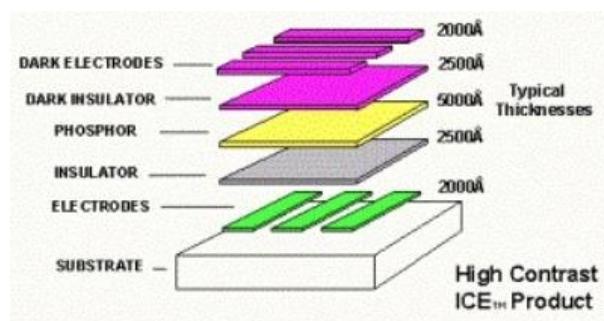


Fig.5.2.1 Afisaj electroluminiscent

Tabel 5.2.1: Avantaje si Dezavantaje la Afisajul EL

Avantaje	Dezavantaje
subțiri, compacte	cost ridicat
funcționare la joasă tensiune	eficiență redusă

✓ Afișajul cu cristale lichide (LCD)

Afișajul cu cristale lichide (LCD) este un tip de tehnologie de afișare folosit într-o gamă largă de dispozitive electronice, inclusiv în telefoanele mobile, televizoare, monitoare de computer și multe alte dispozitive. Acesta utilizează cristale lichide, care sunt substanțe cu proprietăți optice unice, pentru a crea imagini.

Principiul de bază al funcționării unui LCD constă în utilizarea unor cristale lichide care se pot schimba în funcție de tensiunea electrică aplicată acestora. Un LCD este format dintr-un strat de cristale lichide între două polarizoare. Fiecare cristal lichid este încadrat de doi electrozi care controlează polarizarea luminii care trece prin stratul de cristale lichide.

Atunci când un curent electric este aplicat electrozilor, cristalele lichide se aliniază, schimbând proprietățile optice ale stratului și permittând astfel luminii să treacă sau să fie blocată. În acest fel, se poate controla cantitatea de lumină care trece prin cristalele lichide, astfel încât să se creeze imagini.

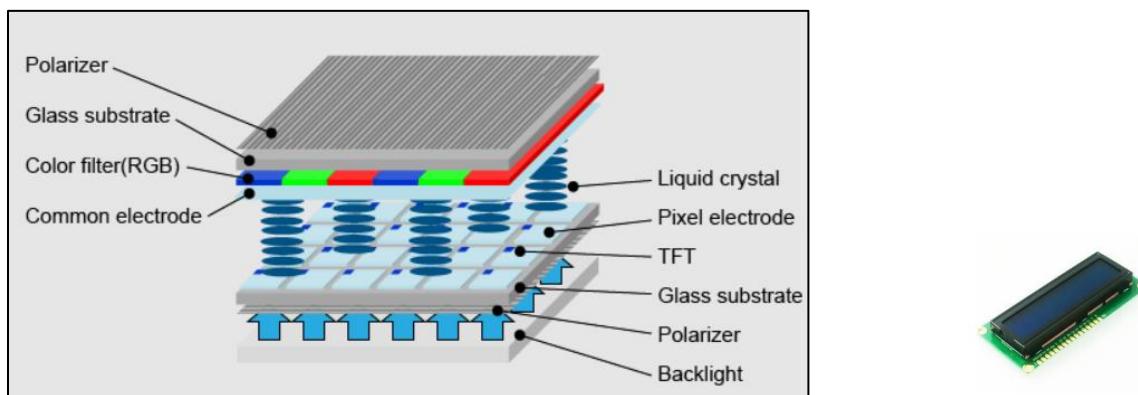


Fig.5.2.2 : Afisaj LCD

Tabel 5.2.2 Avantaje si Dezavantaje la LCD

Avantaje	Dezavantaje
Subtiri, compacte	Unghi de vizualizare limitat
Consum redus	Lumina de fundal poate să fie neuniformă

✓ Afişajul cu diode (LED)

Este un tip de afişaj ce utilizează diode emițătoare de lumină. Un afişaj LED este alcătuit din panouri LED. Diodele din componență sunt luminate de mișcarea electronilor dintr-un material semiconductor. LED-urile sunt construite pentru a elibera un număr mare de fotoni spre exterior, aflați într-un bec din plastic pentru a focaliza lumina, pentru a o proiecta într-o anumită direcție.

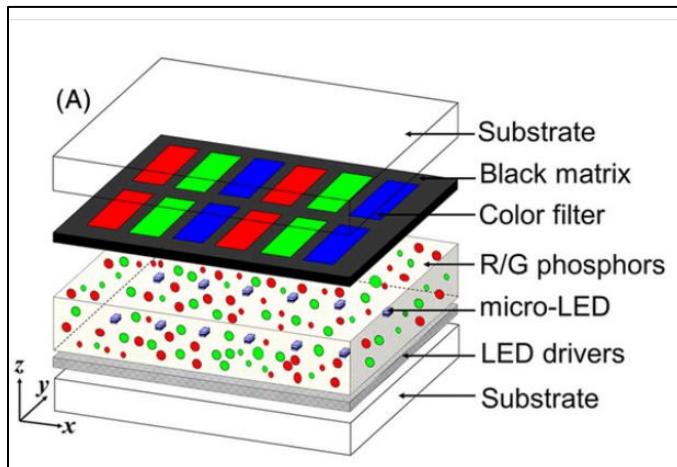


Fig.5.2.3: Afisaj LED



Tabel 5.2.3 Avantaje si Dezavantaje LED

Avantaje	Dezavantaje
contrast ridicat	cost ridicat
distribuție bună a luminii	își schimbă culoarea

✓ Afişajul cu plasmă

Panoul cu plasmă afișează text sau grafică folosind lumina de la un număr mare de celule plasmaticе minusculе. Fiecare pixel este alcătuit din trei subcelule, care emit lumini roșii, verzi și albastre pentru afișarea color. Plasma este un gaz ionizat care conține ioni și electroni cu curgere liberă. Plasma este creată prin aplicarea unei tensiuni foarte mari pe electroziile de pe părțile laterale ale camerei de gaz. Xenonul și neonul sunt cele mai frecvent utilizate gaze pentru producerea plasmei.

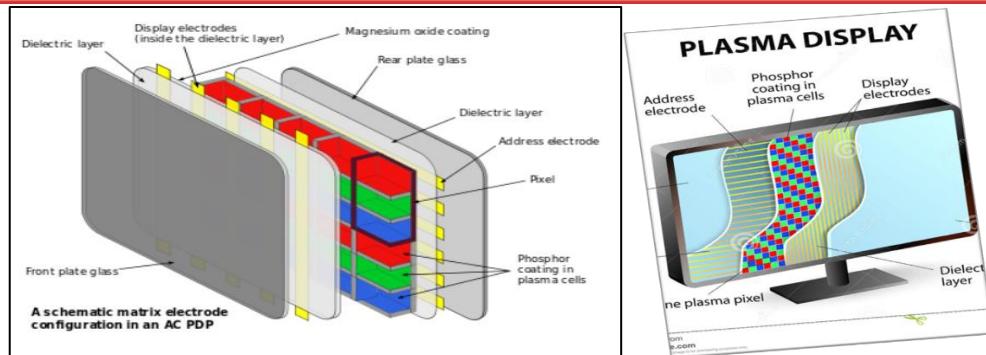


Fig.5.2.4 Afisaj cu plasma

Tabel 5.2.4 Avantaje si Dezavantaje plasma

Avantaje	Dezavantaje
contrast ridicat	cost ridicat
distribuție bună a luminii	își schimbă culoarea, contrastul

5. 3. Alegerea tipului de afisaj electronic

Pentru proiectul meu voi alege un afisaj electronic de tip LCD deoarece:

- ✓ Poate fi utilizat la altitudinii inalte
- ✓ Energie consumata redusa
- ✓ Greutate mica a ecranelor
- ✓ Durata crescuta de viata
- ✓ O luminozitate buna si o vizibilitate superioara sub lumina directa a soarelui

Criteriile dupa care ar trebui sa alegem un afisaj LCD sunt:

- ✓ Pretul
- ✓ Numarul de caractere/numarul de linii
- ✓ Energia consumata
- ✓ Durata de viata
- ✓ Tensiunea de alimentare

Tabel 5.3.1 Exemple de LCD-uri:

Denumire	Tip Afisaj	Nr de semne	Tensiunea de alimentare	Temperatura de functionare	Tipul Controllerului folosit	Pret
JHD162A	Alfanumeric	16x2	5V	0°C pana la 50°C	HD44780	21.21 LEI
WH1602B	Alfanumeric	16x2	5V	-20°C pana la 70°C	HD44780	72.56 LEI
ADM1602K	Alfanumeric	16x2	5V	-20°C pana la 70°C	HD44780	29.87 LEI
LCM1602C	Alfanumeric	16x2	5V	0°C pana la 50°C	HD44780	15.32 LEI
LM016L	Alfanumeric	16x2	5V	0°C pana la 50°C	HD44780	82.17 LEI

În urma documentării făcute, am ales să folosesc un afișaj LCD. Display-ul LM016L este unul dintre cele mai folosite de pe piață și este ideal pentru realizarea proiectului meu.

LCD LM016L este un display de 16x2. Acest lucru înseamnă că informația o să fie afișată pe 16 coloane și două linii. Acest display este alimentat de la o sursă de 5V și permite scrierea caracterelor alfanumerice principale. LM016L are în componența lui un cip operator standard HD44780 care primește date de la o sursă externă și comunică în mod direct cu LCD-ul.

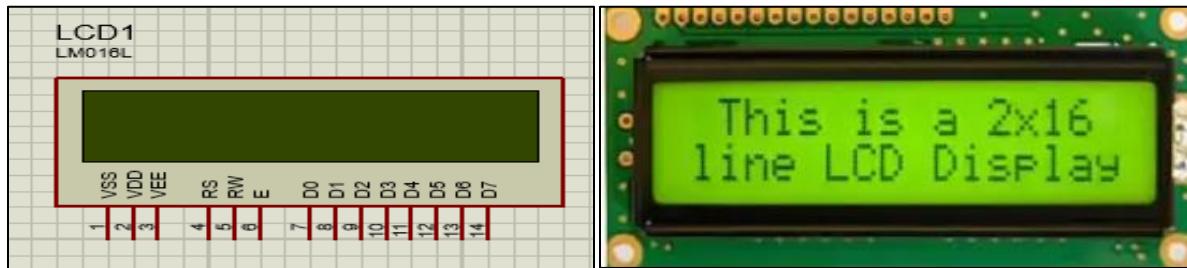


Fig.5.3.1 LCD LM016L

Primii 3 pini au scopul de a porni (alimenta) display-ul LCD și de a regla intensitatea luminoasă a acestuia:

- VSS(pinul nr.1) – reprezintă masa display-ului și se leagă la GND.
- VDD(pinul nr.2) – reprezintă alimentarea LCD-ului și este legat la o sursă de alimentare de +5V.
- VEE(pinul nr.3) – reprezintă locul de unde este ajustat contrastul, mai exact, pentru intensitate maximă, acesta trebuie să fie legat la GND, dar putem să modificăm acest lucru prin intermediul unui potențiometru de 2k.

- RS(pinul nr.4) – Register Select, este folosit pentru selectarea pinilor interni ai LCD-ului. In cazul in care RS='0' înseamnă că trimitem o instrucțiune (operație scriere), iar daca RS='1' înseamnă că trimitem data (operație scriere/citire).
- RW(pinul nr.5) – Read/Write reprezintă semnalul de intrare, adică pentru RW='0' o să avem operație de scriere, iar pentru RW='1' o să avem operație de citire.
- E(pinul nr.6) – Enable are rolul de a permite sau de a nu permite transmiterea de date. Cu alte cuvinte, activează operația de scriere sau de citire si este activ pe front negativ.
- DB0 ... DB3(pinul 7 – pinul 10) - reprezintă semnale de intrare/ ieșire – liniile reprezintă partea LOW a busului de date, care sunt folosite in tranzacțiile cu unitatea de procesare; aceste linii nu o să fie folosite în cazul în care se face interfațarea pe 4 biți cu un microprocesor.
- DB4 ... DB7(pinul 11 – pinul 14) – reprezintă semnale de intrare/ieșire – liniile reprezintă partea HIGH a busului de date, care sunt folosite in tranzacții cu unitatea de procesare.

5.4. Simulari si modul de functionare

Acest tip de display poate să afișeze pe 2 linii câte 16 caractere. Toate aceste date sunt memorate în memoria RAM și de acolo sunt afișate pe display. În cazul în care avem mai mult de 16 caractere, acestea o să fie memorate în memoria RAM, dar nu o să fie afișate pe ecran.

CGRAM (character generator RAM) : se pot genera noi caractere de către utilizator pe adresele 0x00 pana la 0x07, câte 8 caractere de 5x8 puncte si 4 caractere de 5x10 puncte.

CGROM(character generator RAM): se pot afișa simbolurile ASCII de câte 5x8 puncte sau 5x10 puncte pe câte 8 biți. Se pot afișa 208 caractere de câte 5x8 puncte si 32 de caractere de câte 5x10 puncte.

Fig.5.4.1 Codul Caracterelor

5.4.1 Instructiuni

Tabel 5.4.1 Clear Display:

Tabel 5.4.2 Display On/Off control – această instrucție setează starea afișajului, a cursorului și modul de afisare al acestuia.

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	1	D	C	B

Este necesar ca DB3 să fie “1”, iar în DB2(D), DB1(C), DB0(B) avem:

D	Functia
1	Display On.
0	Display Off.

C	Functia
1	Cursor On. Cursorul este afișat.
0	Cursor Off.

B	Functia
1	Caracterul pentru poziția cursorului clipește.
0	Caracterul nu va clipești.

Entry mode set – se setează direcția de mișcare a cursorului și mutarea poziției de afișare.

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	1	I/D	S

În DB2 este “1”, iar în DB1(I/D) și DB0(S), avem:

I/D	Functia
1	Incrementarea adresei:adresa DDRAM sau CGRAM este incrementată cu 1 când scriem/citim, și cursorul se mută la dreapta
0	Decrementarea adresei:adresa DDRAM sau CGRAM este decrementată cu 1 când scriem/citim, și cursorul se mută la stânga

S	Functia
1	Întregul display este mutat.Direcția de mutare este determinată de I/D.La stânga când I/D='1' și la dreapta pentru I/D='0'.Mutarea este operată doar pentru caractere
0	Afișajul nu este mutat.

Function Set – setează lățimea de date a interfeței(prin DL) și numărul de linii ale afișajului(prin N).

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	DL	N	*	E1	E0

În DB5 este scris “1”, iar în DB4, DB3, DB1, DB0 sunt scrise (DL), (N), (E1), (E0) – în orice ordine. (E1) și (E0) – setează numărul de caractere pentru afișaj.

DL	Funcția
1	Setează lungimea de date a interfeței la 8 biți.(DB7-DB0)
0	Setează lungimea de date a interfeței la 4 biți (DB7-DB4)

N	E1	E0	Linii display	Caractere/line
0	0	0	2	16
	0	1		24
	1	0		32
	1	1		40
1	*	0	4	20
	*	1		24

Pentru acest tip de testare am avut nevoie de 7 LOGICSTATES pentru biții D0-D7 al display-ului, pe care setăm codul ASCII al literii dorite și încă 3 LOGICSTATES pentru RS, RW și respectiv E.

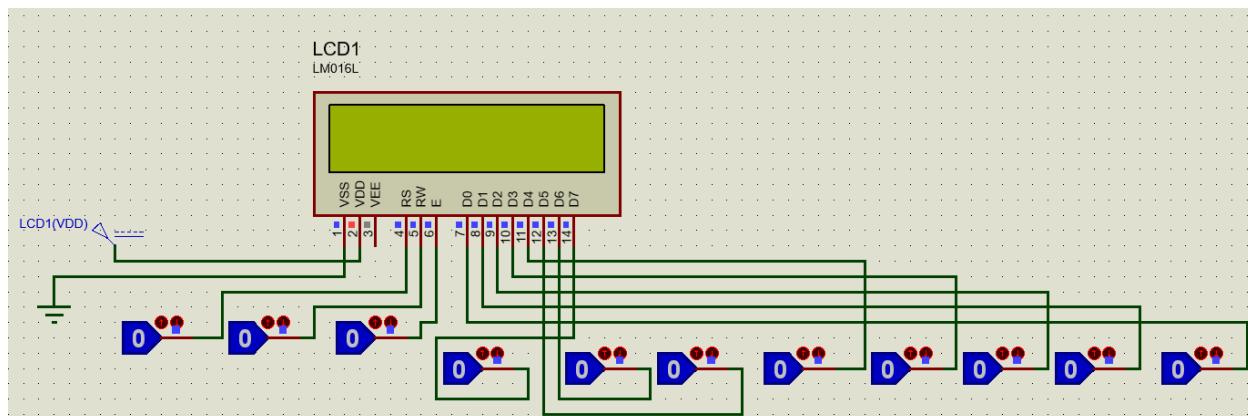


Fig.5.4.2 LCD Proteus

No	HEX Value	COMMAND TO LCD
1	0x01	Clear Display Screen
2	0x30	Function Set: 8-bit, 1 Line, 5x7 Dots
3	0x38	Function Set: 8-bit, 2 Line, 5x7 Dots
4	0x20	Function Set: 4-bit, 1 Line, 5x7 Dots
5	0x28	Function Set: 4-bit, 2 Line, 5x7 Dots
6	0x06	Entry Mode
7	0x08	Display off, Cursor off
8	0x0E	Display on, Cursor on
9	0x0C	Display on, Cursor off
10	0x0F	Display on, Cursor blinking
11	0x18	Shift entire display left
12	0x1C	Shift entire display right
13	0x10	Move cursor left by one character
14	0x14	Move cursor right by one character
15	0x80	Force cursor to beginning of 1st row
16	0xC0	Force cursor to beginning of 2nd row

Fig.5.4.3 Comenzi pentru LCD

Pentru inițializare s-au folosit secvențele (000-0000 1110), (000-0000 1111) și (100-0000 1111), fiecare secvență fiind urmată de activarea și dezactivarea bitului E(enable). Fiecare literă ce apare în LCD a fost scrisă conform codului ASCII, prin activarea bitului RS și a bitului ENABLE și a biților corespunzători valorii în binar a literelor.

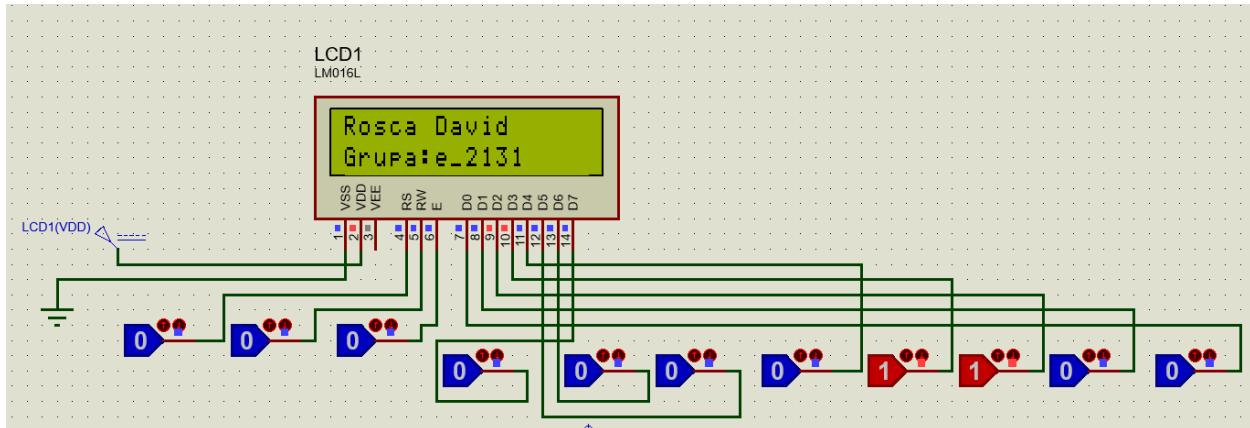


Fig.5.4.5 Simularea numelui și a grupei în Proteus cu ajutorul unui LM016L

6. Alegerea Microcontrolerului și programarea LCD-ului în C și asm

6.1 Ce este un microcontroler?

Un microcontroler este un circuit integrat compact conceput pentru a guverna o anumită operațiune într-un sistem încorporat. Un microcontroler tipic include un procesor, memorie și periferice de intrare/ieșire (I/O) pe un singur cip.

În 1981, Intel a introdus pe piață un microcontroler pe 8 biți numit 8051. Acesta avea 128 de octeți de memorie RAM, 4KB de memorie ROM internă, două timere, un port serial și patru porturi I/O, fiecare de câte 8 biți.

Microcontrolerul 8051 a devenit popular după ce Intel a permis altor firme producătoare de chip-uri să facă propriile variații de microcontroler de acest tip, cu condiția ca ele să rămână

compatibile între ele din punct de vedere al codului. Acest lucru înseamnă că un program scris pentru 8051 va funcționa pe oricare microcontroler din familie, indiferent de firma producătoare.

Astfel, au apărut multe versiuni ale microcontroler-ului 8051, fiind diferite din punct de vedere al vitezei de ceas, cantitatea de memorie RAM sau ROM, numărul de porturi și funcțiile acestora, etc.

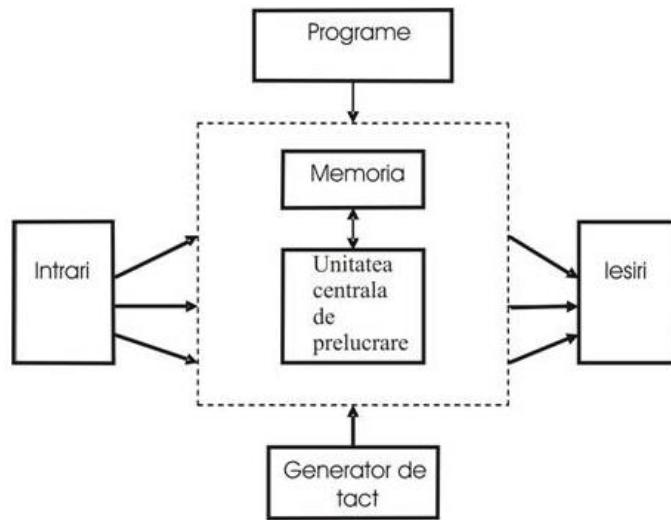


Fig. 6.1 Schema bloc specifică a unui microcontroller

Ca intrari se folosesc de regula semnale provenind de la comutatoarele individuale sau de la traductoare (de temperatură, de presiune, foto, traductoare specializate). Intrările pot fi *digitale* sau *analogice*.

- ✓ **Intrările digitale** vehiculează semnale discrete, informația citită fiind informația ce se esantionează la momentul citirii liniei respective.
- ✓ **Intrările analogice** vehiculează informații exprimabile prin funcții continue de timp. Citirea acestora de către microcontroller presupune prezenta unor circuite capabile să prelucreze aceste informații, fie comparatoare analogice, fie convertoare analog-numerice, ale caror ieșiri sunt citite de către MC.

Unitatea centrală de prelucrare este compusă din unitatea aritmetică și logică (UAL) și din unitatea de control.

- ✓ **Unitatea aritmetica si logica** este sectiunea responsabila cu efectuarea operatiilor aritmetice si logice asupra operanzilor ce ii sunt furnizati. Modul de implementare a operatiilor este transparent pentru utilizator; important pentru utilizatorul de MC este repertoriul operatiilor implementate pentru a aprecia posibilitatea implementarii optime a aplicatiei particulare de control. De asemenea este important timpul de executie al fiecarei operatii pentru a aprecia daca timpul necesar procesarii complete satisface cerintele de timp ale aplicatiei.
- ✓ **Unitatea de control** este responsabila cu decodificarea codului operatiei continut de codul unei instructiuni. Pe baza decodificarii unitatea de control elaboreaza semnale pentru comanda celoralte blocuri functionale pentru a finaliza executarea unei instructiuni. Modul de implementare al acestui bloc este de asemenea transparent utilizatorului.

Unitatea centrala de prelucrare contine un set de registre interne, similare unor locatii de memorie, folosite pentru memorarea unor date des apelate sau pentru programarea unor anumite functii. Diferitele familii de MC folosesc seturi diferite de registre. Exista insa cateva registre comune.

- ✓ **A** (Accumulator) - registrul acumulator - este folosit deseori pentru a stoca un operand si rezultatul operatiei aritmetice sau logice.
- ✓ **PC** (Program Counter) - regisztrul numarator de program - este regisztrul care stocheaza adresa urmatoarei instructiuni de executat. Dupa un RESET (initializarea MC), regisztrul PC se incarca dintr-o locatie de memorie numita vector de reset. Aceasta locatie contine adresa primei instructiuni de executat. PC este incrementat automat la executia unei instructiuni.
- ✓ **SP** (Stack Pointer) - regisztrul indicator de stiva - continutul acestui regisztr indica adresa curenta a stivei. Stiva reprezinta o zona de memorie accesibila rapid in care se depun temporar informatii importante in desfasurarea programului. Stiva este definita de obicei in RAM. Implementarea accesului presupune existenta unui regisztr de adresare (SP) si a mecanismului de memorare declansat de instructiuni specifice (instructiunile PUSH/POP).

6.2 Structura memoriei la microcontrolerul 8051

Memoria microcontrolerului 8051 este organizată în trei spații diferite: memoria de cod, memoria de date și memoria externă.

Memoria de cod poate fi internă (pentru 8051 de bază, ROM-ul fiind de 4KB), externă sau în totalitate externă, unele microcontrolere neavând memorie de cod incorporată în structură.

Este segmentul de memorie de tip read-only în care se stochează programul și constantele.

Memoria de date are 128 octeți de memorie RAM internă. Aceasta este împărțită în mai multe segmente. La adresele din această memorie pot fi accesati regiștri cu funcții speciale, bancurile de regiștri R, stiva, memoria adresabilă la nivel de bit. Variabilele folosite des în program vor fi stocate în segmentul de date datorită vitezei ridicate cu care poate fi accesată această memorie.

Microcontrolerul 8051 permite conectarea de memorie RAM externă la porturi, în caz că memoria internă nu este destulă. Memoria externă poate fi de maxim 64KB.

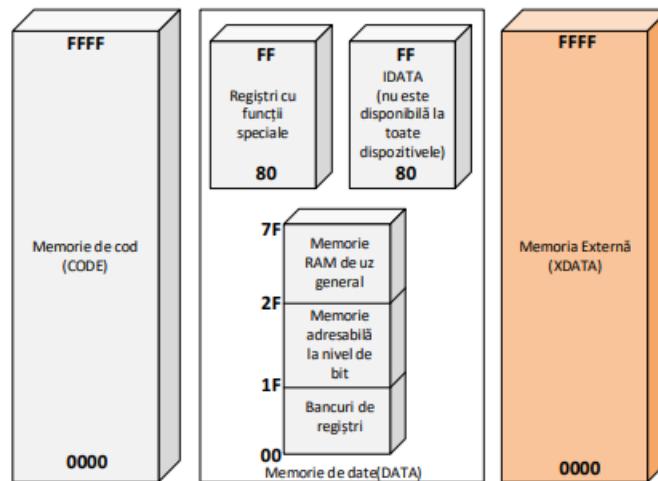


Fig.6.2 Organizarea memoriei la microcontrolerul 8051

6.3 . Familii de microcontrolere

Familiile cele mai cunoscute ale micricontrolerelor de 8 biți și 16biți sunt:

- 8048** (Intel MCS-48) – „bunicul” microcontrolerelor pe 8 biți, mai este încă folosit!



Fig.6.3 Microcontrolerul 8048

- 8051** (Intel MCS-51 și mulți alții: Atmel, Philips, Atmel, Dallas-Maxim, etc) - a doua generație de microcontrolere de 8 biți a firmei Intel care, deși apărută acum 20 de ani, încă ocupă un segment semnificativ de piață. Cu o arhitectură destul de ciudată, este suficient de puternic și ușor de programat.



Fig.6.4 Microcontrolerul 8051

- c. **80C196** (Intel MCS-96) - este un microcontroler pe 16 biți făcând parte din generația treia de microcontrolere a firmei Intel. Destinat inițial unor aplicații din industria de automobile, are o arhitectură Von Neumann, cu un spațiu de adresare de 64KBytes, o unitate de intrări/ieșiri numerice de mare viteză (destinată inițial controlului injecției la un motor cu ardere internă), ieșiri PWM, convertor analog numeric, timer watchdog.

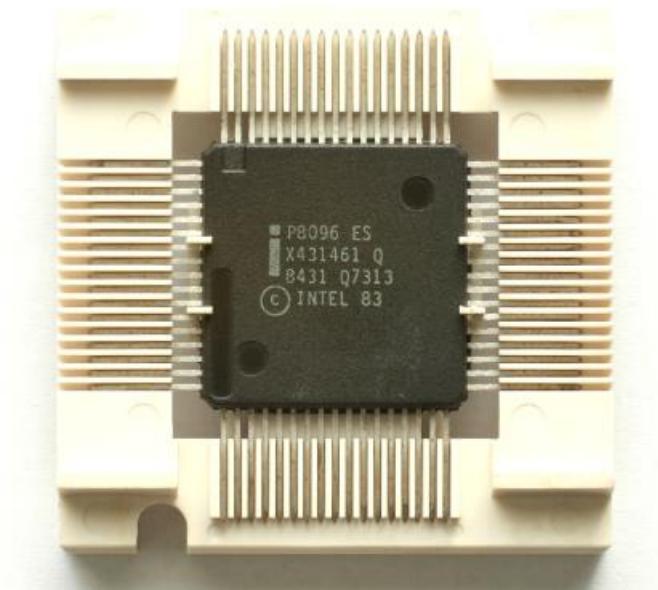


Fig.6.5 Microcontrolerul 80C196

- d. **68HC11, 68HC12, 68HC16** (Freescale) 68HC11 a fost unul din cele mai puternice microcontrolere pe 8 biți. Are un spațiu de adrese unic de 64K. Prezintă ca particularitate existența unui program de încărcare rezident (bootstrap loader în ROM intern) cu care, la reset, un segment din memoria RAM externă poate fi încărcat cu cod program prin intermediul portului serial.



Fig.6.6 . Microcontrolerul 68HC11

- e. **PIC** (Microchip) Primul microcontroler din această familie (PIC1650) a apărut acum mai bine de 20 de ani pe vremea când firma era proprietatea General Instruments. Este o familie de microcontrolere care, în ultimii ani, a cunoscut o dezvoltare explozivă. Sunt disponibile actualmente sub forma a 6 serii: PIC10, PIC12, PIC14, PIC16, PIC17 și PIC18. Au fost primele microcontrolere de 8 biți cu arhitectură RISC: PIC16C5x avea un set de doar 33 instrucțiuni (Intel 8048 avea 90). Arhitectura este de tip Harvard și, ca o particularitate, dimensiunea cuvântului pentru program este de 12, 14 sau 16 biți, cuvântul de date fiind tot de 8 biți. Există foarte multe variante pentru cele șase serii, unele din ele fiind caracterizate printr-un număr mic de conexiuni exterioare (pini) și în consecință dimensiuni mici, consum foarte mic, ideea de bază fiind costul redus.



Fig.6.7 Microcontrolerul PIC1650

- e. **Z8** (Zilog) Un derivat al microprocesorului Z80, reprezintă un composit al mai multor arhitecturi diferite. Nu este compatibil cu setul de instrucțiuni și nici cu perifericele standard Z80. Are trei spații de adrese: program, date și un masiv de registre.

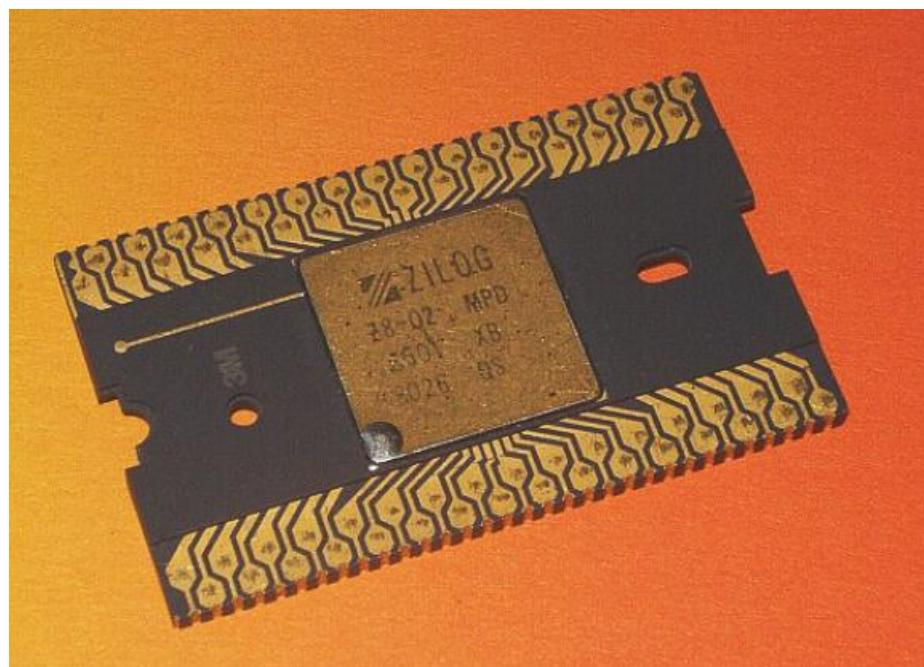


Fig. 6.8 Microcontrolerul Z8

- f. **80386EX** (Intel) Un 80386 destinat aplicațiilor de tip controler. Resurse locale: I/O seriale, timere/numărătoare, optimizarea consumului, controler de întreruperi, controler pentru 3/5 Microcontrolere și automate programabile Laborator 001 – Microcontrolerul RAM dinamic. Marele avantaj al unui astfel de microcontroler este că se poate utiliza ca platformă de dezvoltare un sistem de tip IBM PC împreună cu tot mediul de programare aferent.



Fig. 6.9 Microcontrolerul 80386EX

- g. **80C16x** (Infineon, ex Siemens) Unul din microcontrolerele de 16 biți foarte utilizat în Europa. Arhitectură deosebit de performantă a CPU, de tip RISC, are diverse variante, cu resurse complexe: 80C165, 80C166, 80C167, etc.

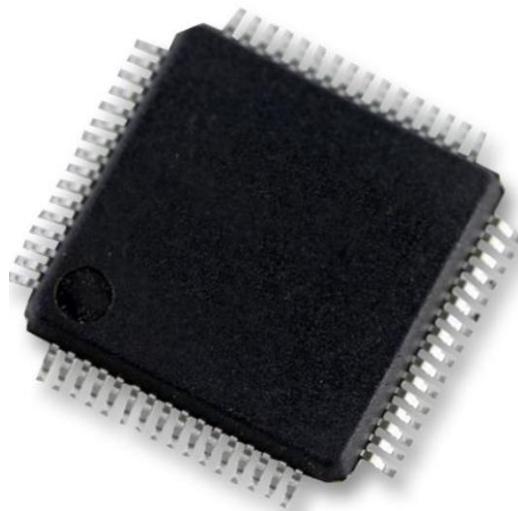


Fig. 6.10 Microcontrolerul 80c16x

6.4 Pinii microcontrolerului și alegerea acestuia

După cum se poate observa în imaginea de mai jos, 8051 are 2 pini de alimentare (VCC, GND), 2 pini pentru oscilator (XTAL), 4 pini de control (EA, ALE, PSEN,RST), și patru porturi de câte 8 biți fiecare (P0, P1, P2, P3).

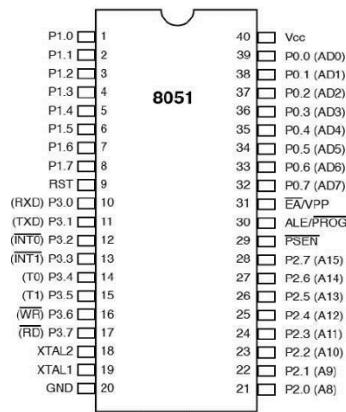


Figura 6.3 Pinii microcontrolerului 8051

Semnalele de control EA, ALE, PSEN sunt utilizate în principal pentru interfațarea cu memoria externă. RST resetează microcontrolerul. Cele patru porturi I/O, P0, P1, P2 și P3 sunt utilizate pentru a prelua sau transmite semnale digitale. Porturile P0 și P2 sunt folosite pentru conectarea memoriei RAM externe, portul P1 nu are funcții duale, iar portul P3 pe lângă funcția I/O, mai poate accesa regiștrii cu funcții speciale, pentru ca utilizatorul să poată folosi timerele interne, portul serial, intreruperile externe, etc.

Alte modele de microcontroler 8051 pot avea mai mulți sau mai puțini pini pentru porturi, fiecare implementând diferite funcții speciale. Spre exemplu, unele microcontrolere au incorporate circuite de conversie (CAN sau CNA), iar anumiți pini ai unor porturi sunt utilizati pe post de intrare sau ieșire.

În tabelul de mai jos se pot observa parametrii mai multor modele de microcontroler 8051.

	AT89S51	AT89C51	C8051F860	C8051F042	DS80C323
Tensiunea de alimentare	4V ~ 5.5V	4V ~ 6V	2.2V ~ 3.6V	2.7V ~ 3.6V	2.7V ~ 5.5V
Viteza de ceas	24MHz	24MHz	25MHz	25MHz	18MHz
Interfața de comunicare	UART/USART	UART/USART	I2C, SPI, UART/USART	CANbus, EBI/EMI, SMBus, SPI, UART/USART	EBI/EMI, SIO, UART/USART
Nr. de pini I/O	32	32	13	64	32
Dim. memorie de cod	4KB	4KB	8KB	64KB	-
Tipul memoriei de cod	Flash	Flash	Flash	Flash	ROMless
Dim. memorie RAM	128B	128B	512B	4.25B	256B
Convertor de date	-	-	A/D 12x12b	A/D 8x8b, 13x10b; D/A 2x10b, 2x12b	-

Dintre modelele de mai sus, pentru acest proiect am ales **microcontrolerul AT89S51**. Acest model este o variantă mai nouă a modelului AT89C51. Această aplicație nefiind una foarte complexă, nu necesită dimensiuni prea mari de memorie, astfel că cei 128B de RAM și 4KB de memorie de cod satisfac nevoile proiectului. AT89S51 are 4 porturi I/O, destule pentru conectarea cu CAN-ul și interfațarea LCD-ului. De asemenea, deși nu are CAN incorporat, folosim deja un CAN extern pentru conversia de semnal din analog în digital.

6.5 Programarea microcontrolerului în ASM

```

LCD_ASM_BUN.a51

1  ORG 0000h
2      SJMP Start ; Sari la eticheta Start
3
4 ; Definirea constantelor
5 dl          EQU 099h ; Durata de întârziere pentru funcția delay
6 rs          EQU P2.5 ; Pinul pentru selectarea registrelor LCD
7 rw          EQU P2.6 ; Pinul pentru citire/scriere pe LCD
8 en          EQU P2.7 ; Pinul de activare pentru LCD
9 ale         EQU P2.3 ; Pinul pentru activarea latch-ului de adresa pe LCD
10 oe          EQU P2.4 ; Pinul pentru activarea iesirii pe LCD
11 start_button EQU P2.1 ; Pinul pentru butonul de pornire
12 eoc          EQU P2.0 ; Pinul pentru sfârșitul conversiei
13 clk          EQU P2.2 ; Pinul de ceas
14 chc          EQU P0.7 ; Pinul pentru selectia caracterului C
15 chb          EQU P0.6 ; Pinul pentru selectia caracterului B
16 cha          EQU P0.5 ; Pinul pentru selectia caracterului A
17
18 Start:
19     ACALL lcdinit           ; Initializarea LCD-ului
20     MOV R1, #'T'            ; Încarcarea caracterului 'T' în registrul R1 pentru afisare
21     ACALL lcd_data         ; Afisarea caracterului 'T' pe LCD
22     MOV R1, #'E'            ; Încarcarea caracterului 'E' în registrul R1 pentru afisare
23     ACALL lcd_data         ; Afisarea caracterului 'E' pe LCD
24     MOV R1, #'M'            ; Încarcarea caracterului 'M' în registrul R1 pentru afisare
25     ACALL lcd_data         ; Afisarea caracterului 'M' pe LCD
26     MOV R1, #'P'            ; Încarcarea caracterului 'P' în registrul R1 pentru afisare
27     ACALL lcd_data         ; Afisarea caracterului 'P' pe LCD
28
29 cont:
30
31     CLR chc                ; Debifarea selectiei caracterului C
32     CLR chb                ; Debifarea selectiei caracterului B
33     CLR cha                ; Debifarea selectiei caracterului A
34     SETB ale                ; Setarea pinului de activare al latch-ului de adresa
35     SETB start_button       ; Setarea pinului pentru butonul de pornire
36     MOV R0, #1                ; Încarcarea valorii 1 în registrul R0
37     ACALL delay             ; Apelul functiei de întârziere
38     CLR ale                ; Debifarea pinului de activare al latch-ului de adresa
39     CLR start_button        ; Debifarea pinului pentru butonul de pornire
40     XXX:                   ; Eticheta XXX
41     JB eoc, XXX             ; Sari daca pinul pentru sfârșitul conversiei este ridicat
42     XXX1:                  ; Eticheta XXX1
43     JNB eoc, XXX1           ; Sari daca pinul pentru sfârșitul conversiei este coborât
44     SETB oe                 ; Setarea pinului de activare a iesirii pe LCD
45     MOV R1, #085h            ; Încarcarea valorii 085h în registrul R1 pentru comanda
46     ACALL lcd_command       ; Apelul functiei de comanda pentru LCD
47     mov a,pl                ; Încarcarea valorii de pe portul P1 în registrul accumulator
48     acall separate          ; Apelul functiei separate
49     mov a,r2                ; Încarcarea valorii din registrul R2 în registrul accumulator
50     acall convert           ; Apelul functiei de conversie
51     mov rl,a                ; Transferul valorii din registrul accumulator în registrul R1
52
53     acall lcd_data          ; Apelul functiei de trimitere a datelor catre LCD
54     mov a,r3                ; Încarcarea valorii din registrul R3 în registrul accumulator
55     acall convert           ; Apelul functiei de conversie
56     mov rl,a                ; Transferul valorii din registrul accumulator în registrul R1

```

```
LCD_ASM_BUN.a51

58 acall lcd_data ; Apelul functiei de trimitere a datelor catre LCD
59     MOV R1, #'h' ; Încarcarea simbolului 'h' in registrul R1 pentru afisare
60     ACALL lcd_data ; Afisarea simbolului 'h' pe LCD
61     CLR oe ; Debifarea pinului de activare a iesirii
62     CPL clk ; Complementarea pinului de ceas
63     SJMP cont ; Sari la eticheta cont pentru a relua bucla
64
65 lcdinit:
66     MOV R1, #038h ; Încarcarea valorii 038h in registrul R1 pentru comanda
67     ACALL lcd_command ; Apelul functiei de comanda pentru LCD
68     MOV R1, #0lh ; Încarcarea valorii 0lh in registrul R1 pentru comanda
69     ACALL lcd_command ; Apelul functiei de comanda pentru LCD
70     MOV R1, #00fh ; Încarcarea valorii 00fh in registrul R1 pentru comanda
71     ACALL lcd_command ; Apelul functiei de comanda pentru LCD
72     MOV R1, #006h ; Încarcarea valorii 006h in registrul R1 pentru comanda
73     ACALL lcd_command ; Apelul functiei de comanda pentru LCD
74     MOV R1, #0ch ; Încarcarea valorii 0ch in registrul R1 pentru comanda
75     ACALL lcd_command ; Apelul functiei de comanda pentru LCD
76     MOV R1, #080h ; Încarcarea valorii 080h in registrul R1 pentru comanda
77     ACALL lcd_command ; Apelul functiei de comanda pentru LCD
78     RET ; Returneaza la adresa de apel
79
80 lcd_command:
81     MOV P3, R1 ; Trimiterea valorii din registrul R1 pe portul P3
82     CLR rs ; Debifarea pinului pentru selectarea registrelor
83     CLR rw ; Debifarea pinului pentru citire/scriere
84     SETB en ; Setarea pinului de activare
85     ACALL delay ; Apelul functiei de intârziere
86     CLR en ; Debifarea pinului de activare
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
```

```
LCD_ASM_BUN.a51

85     ACALL delay ; Apelul functiei de intârziere
86     CLR en ; Debifarea pinului de activare
87     ACALL delay ; Apelul functiei de intârziere
88     RET ; Returneaza la adresa de apel
89
90 lcd_data:
91     MOV P3, R1 ; Trimiterea valorii din registrul R1 pe portul P3
92     SETB rs ; Setarea pinului pentru selectarea registrelor
93     CLR rw ; Debifarea pinului pentru citire/scriere
94     SETB en ; Setarea pinului de activare
95     ACALL delay ; Apelul functiei de intârziere
96     CLR en ; Debifarea pinului de activare
97     ACALL delay ; Apelul functiei de intârziere
98     RET ; Returneaza la adresa de apel
99
100 delay:
101     MOV R7, #dl ; Încarcarea valorii dl in registrul R7
102 Timer:
103     NOP ; Nicio operatie
104     NOP ; Nicio operatie
105     NOP ; Nicio operatie
106     NOP ; Nicio operatie
107     DJNZ R7, Timer ; Decrementarea lui R7 si sari la eticheta Timer daca nu este zero
108     NOP ; Nicio operatie
109     DJNZ R0, delay ; Decrementarea lui R0 si sari la eticheta delay daca nu este zero
110     RET ; Returneaza la adresa de apel
111
112 convert:
113     mov dptr,#0500h ; Încarcarea adresei de memorie 0500h in DPTR
```

```

LCD_ASM_BUN.a51

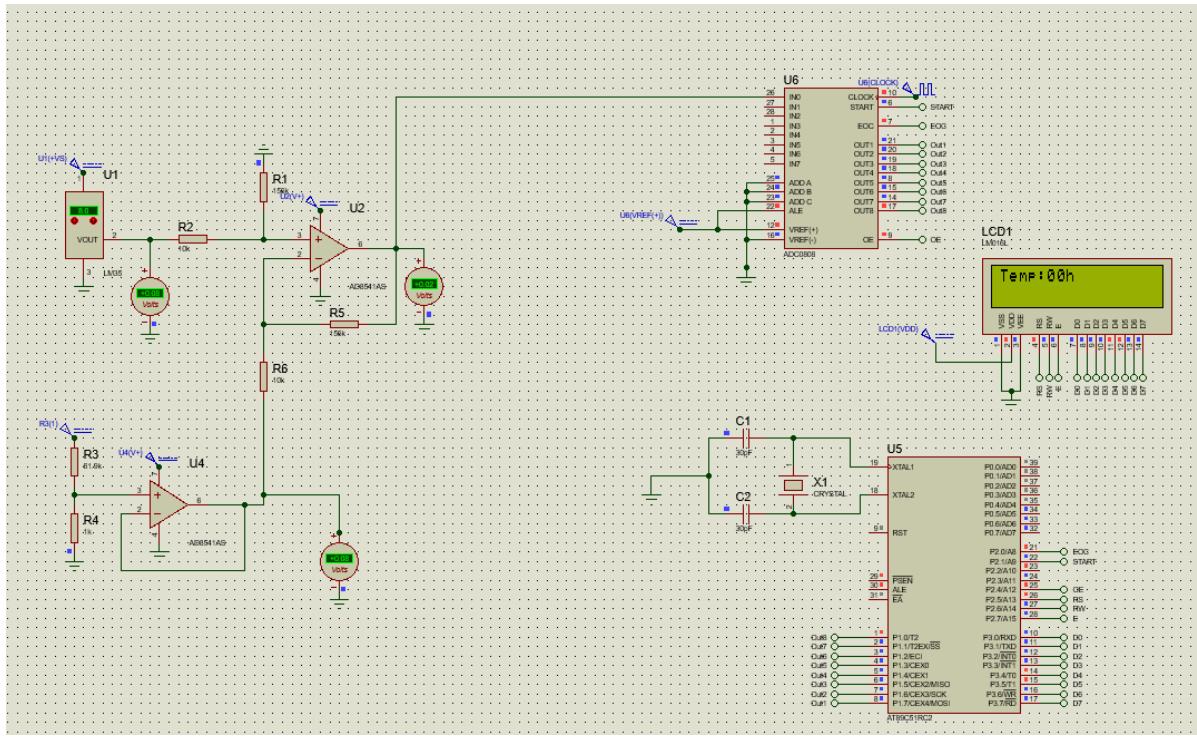
112 convert:
113 mov dptr,#0500h ; Încarcarea adresei de memorie 0500h în DPTR
114 movc a,@a+dptr ; Încarcarea conținutului de la adresa indicată de DPTR în registrul accumulator
115 ret ; Returnează la adresa de apel
116
117 separate:
118 push acc ; Pune valoarea registrului accumulator pe stiva
119 MSB:
120 clr cy ; Debifarea flagului de carry
121 rrc a ; Rotirea la dreapta cu transfer de carry
122 clr cy ; Debifarea flagului de carry
123 rrc a ; Rotirea la dreapta cu transfer de carry
124 clr cy ; Debifarea flagului de carry
125 rrc a ; Rotirea la dreapta cu transfer de carry
126 clr cy ; Debifarea flagului de carry
127 rrc a ; Rotirea la dreapta cu transfer de carry
128 clr cy ; Debifarea flagului de carry
129 mov r2,a ; Transferul valorii din registrul accumulator în registrul R2
130 pop acc ; Ia valoarea de pe stiva și o pune în registrul accumulator
131 LSB:
132 clr cy ; Debifarea flagului de carry
133 rlc a ; Rotirea la stânga cu transfer de carry
134 clr cy ; Debifarea flagului de carry
135 rlc a ; Rotirea la stânga cu transfer de carry
136 clr cy ; Debifarea flagului de carry
137 rlc a ; Rotirea la stânga cu transfer de carry
138 clr cy ; Debifarea flagului de carry
139 rlc a ; Rotirea la stânga cu transfer de carry
140 clr cy ; Debifarea flagului de carry

LCD_ASM_BUN.a51

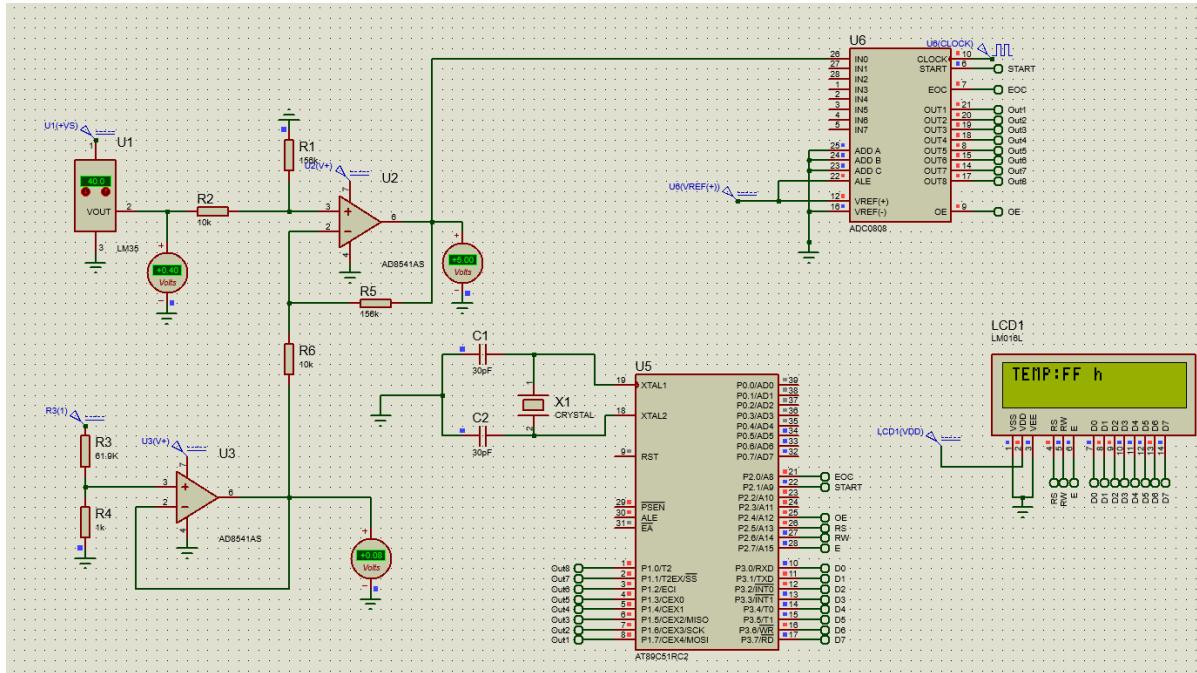
128 clr cy ; Debifarea flagului de carry
129 mov r2,a ; Transferul valorii din registrul accumulator în registrul R2
130 pop acc ; Ia valoarea de pe stiva și o pune în registrul accumulator
131 LSB:
132 clr cy ; Debifarea flagului de carry
133 rlc a ; Rotirea la stânga cu transfer de carry
134 clr cy ; Debifarea flagului de carry
135 rlc a ; Rotirea la stânga cu transfer de carry
136 clr cy ; Debifarea flagului de carry
137 rlc a ; Rotirea la stânga cu transfer de carry
138 clr cy ; Debifarea flagului de carry
139 rlc a ; Rotirea la stânga cu transfer de carry
140 clr cy ; Debifarea flagului de carry
141 rrc a ; Rotirea la dreapta cu transfer de carry
142 clr cy ; Debifarea flagului de carry
143 rrc a ; Rotirea la dreapta cu transfer de carry
144 clr cy ; Debifarea flagului de carry
145 rrc a ; Rotirea la dreapta cu transfer de carry
146 clr cy ; Debifarea flagului de carry
147 rrc a ; Rotirea la dreapta cu transfer de carry
148 clr cy ; Debifarea flagului de carry
149 mov r3,a ; Transferul valorii din registrul accumulator în registrul R3
150 ret ; Returnează la adresa de apel
151
152 org 500h ; Punct de start pentru zona de date
153 db 030h, 031h, 032h, 033h, 034h, 035h, 036h, 037h, 38h, 039h, 041h, 042h, 043h, 044h, 045h, 046h
154
155 END

```

6.5.1 Simularea in Proteus la 8 °C=>00h pe LCD



6.5.2 Simularea in Proteus la 40°C=>FFh pe LCD



6.6 Programarea microcontrolerului în C

```

LCD_C.c
1 #include<reg51.h>           // Include biblioteca pentru microcontrollerul 8051
2
3 #include<string.h>          // Include biblioteca pentru lucrul cu siruri de caractere
4
5 sbit RS = P2^5;             // Definirea pinului RS pentru LCD
6 sbit RW = P2^6;             // Definirea pinului RW pentru LCD
7 sbit EN = P2^7;             // Definirea pinului EN pentru LCD
8 //sbit ale=P2^3;            // Pinul pentru controlul aleator al LCD (nu este utilizat in cod)
9 sbit oe=P2^4;               // Pinul pentru controlul ie?irii LCD
10 sbit start=P2^1;            // Pinul pentru controlul pornirii LCD
11 sbit eoc=P2^0;              // Pinul pentru semnalizarea sfâr?itului conversiei
12 //sbit clk=P2^2;            // Pinul pentru controlul ceasului (nu este utilizat in cod)
13 sbit chc=P0^7;              // Pinii pentru selectarea canalelor de intrare ADC
14 sbit chb=P0^6;              // Declaratie variabila pentru stocarea datelor de la ADC
15 sbit cha=P0^5;              // Declaratie variabile pentru calculul rezultatului ADC
16
17 void delay(int t);         // Declaratia functiei pentru intârziere
18 void lcd_init(void);        // Declaratia functiei pentru initializarea LCD-ului
19 void lcd_command(char c);   // Declaratia functiei pentru trimiterea comenzilor catre LCD
20 void lcd_data(char d);      // Declaratia functiei pentru trimiterea datelor catre LCD
21 void str(char a[]);         // Declaratia functiei pentru afisarea sirurilor de caractere pe LCD
22 void print(char p);         // Declaratia functiei pentru afisarea numerelor pe LCD
23 char adc_data;              // Declaratia variabila pentru stocarea datelor de la ADC
24 long float k;               // Declaratie variabila pentru calculul rezultatului ADC
25 char y,z;                  // Declaratie variabile pentru afisarea rezultatului ADC pe LCD
26
27 void main()                // Functia principala a programului
28 {
29     lcd init();              // Initializarea LCD-ului

```



```

LCD_C.c
25 char y,z;                  // Declaratie variabile pentru afisarea rezultatului ADC pe LCD
26
27 void main()                // Functia principala a programului
28 {
29     lcd_init();              // Initializarea LCD-ului
30     lcd_command(0x01);       // Curatarea ecranului LCD
31
32     str("TEMP:");           // Afisarea sirului "TEMP:" pe LCD
33
34     lcd_data(0x10);          // Afisarea unui caracter special pe LCD
35     lcd_data(0x07);
36     lcd_data(0x08);
37     lcd_data(0x08);
38     lcd_data(0x08);
39     lcd_data(0x08);
40     lcd_data(0x07);
41     lcd_command(0x0b);       // Mutarea cursorului pe LCD
42
43     lcd_data(4);              // Afisarea valorii pe LCD
44
45     eoc=1;                   // Setarea pinului EOC ca intrare
46
47     oe=0;                    // Debifarea pinului OE
48     start=0;                 // Debifarea pinului START
49
50     TMOD=0x02;               // Configurarea modului timerului 0
51     TH0=0xc2;                 // Setarea valorii de incarcare pentru timerul 0
52     IE=0x82;                  // Activarea intreruperilor pentru timerul 0
53     TR0=1;                   // Pornirea timerului 0

```

```

LCD_C.c
52     IE=0x82;           // Activarea intreruperilor pentru timerul 0
53     TR0=1;             // Pornirea timerului 0
54
55     while(1)
56     {
57         chc=0;           // Selectarea canalului 0
58         chb=0;
59         cha=0;
60
61         start=1;          // Pornirea conversiei ADC
62         delay(1);
63         start=0;
64
65         while(eoc==1);    // Asteptarea sfârșitului conversiei
66         while(eoc==0);
67
68         oe=1;              // Activarea iesirii ADC
69         k=P1;               // Citirea datelor ADC
70         lcd_command(0x85); // Setarea cursorului LCD
71         print(k);           // Afisarea datelor pe LCD
72         oe=0;               // Debifarea iesirii ADC
73     }
74 }
75
76 void str(char a[]) // Functie pentru afisarea sirurilor de caractere pe LCD
77 {
78     int j;
79     for(j=0; a[j]!='\0'; j++)
80     {

```



```

LCD_C.c
76 void str(char a[]) // Functie pentru afisarea sirurilor de caractere pe LCD
77 {
78     int j;
79     for(j=0; a[j]!='\0'; j++)
80     {
81         lcd_data(a[j]);
82     }
83 }
84
85 void lcd_init(void) // Functie pentru initializarea LCD-ului
86 {
87     lcd_command(0x38);      // Initializarea LCD-ului: 8 biti, 2 linii, 5x8 dots
88     lcd_command(0x01);      // Curatarea ecranului LCD
89     lcd_command(0x0f);      // Afisarea cu cursorul cliptind
90     lcd_command(0x06);      // Modul de intrare
91     lcd_command(0x0c);      // Dezactivarea cursorului
92     lcd_command(0x80);      // Pozitionarea cursorului la inceputul primei linii
93 }
94
95 void lcd_command(char c) // Functie pentru trimiterea comenzilor catre LCD
96 {
97     P3=c;                  // Transmiterea comenzi catre LCD
98     RS=0;                  // Selectarea registrului de comanda
99     RW=0;                  // Setarea in modul de scriere
100    EN=1;                 // Activarea LCD-ului
101    delay(5);              // Întârziere
102    EN=0;                 // Dezactivarea LCD-ului
103    delay(5);              // Întârziere
104 }

```

```

LCD_C.c
103     delay(5);           // Întârziere
104 }
105
106 void lcd_data(char d) // Functie pentru trimitera datelor catre LCD
107 {
108     P3=d;                // Transmiterea datelor catre LCD
109     RS=1;                // Selectarea registrului de date
110     RW=0;                // Setarea in modul de scriere
111     EN=1;                // Activarea LCD-ului
112     delay(5);           // Întârziere
113     EN=0;                // Dezactivarea LCD-ului
114     delay(5);           // Întârziere
115 }
116
117 void delay(int t)      // Functie pentru întârzierea programului
118 {
119     int j;
120     for(j=0; j<t*1275; j++);
121 }
122
123 void print(char p)    // Functie pentru afisarea numerelor pe LCD
124 {
125     //y = p >> 4;
126     y = ((p >> 4)& 0x0F) < 10 ? ((p >> 4)& 0x0F) + '0' : ((p >> 4)& 0x0F) - 10 + 'A';
127     lcd_data(y);        // Afisarea zecilor pe LCD
128
129     z = p & 0x0f;
130     z = z < 10 ? z + '0' : z - 10 + 'A';
131     lcd_data(z);        // Afisarea unitatilor pe LCD

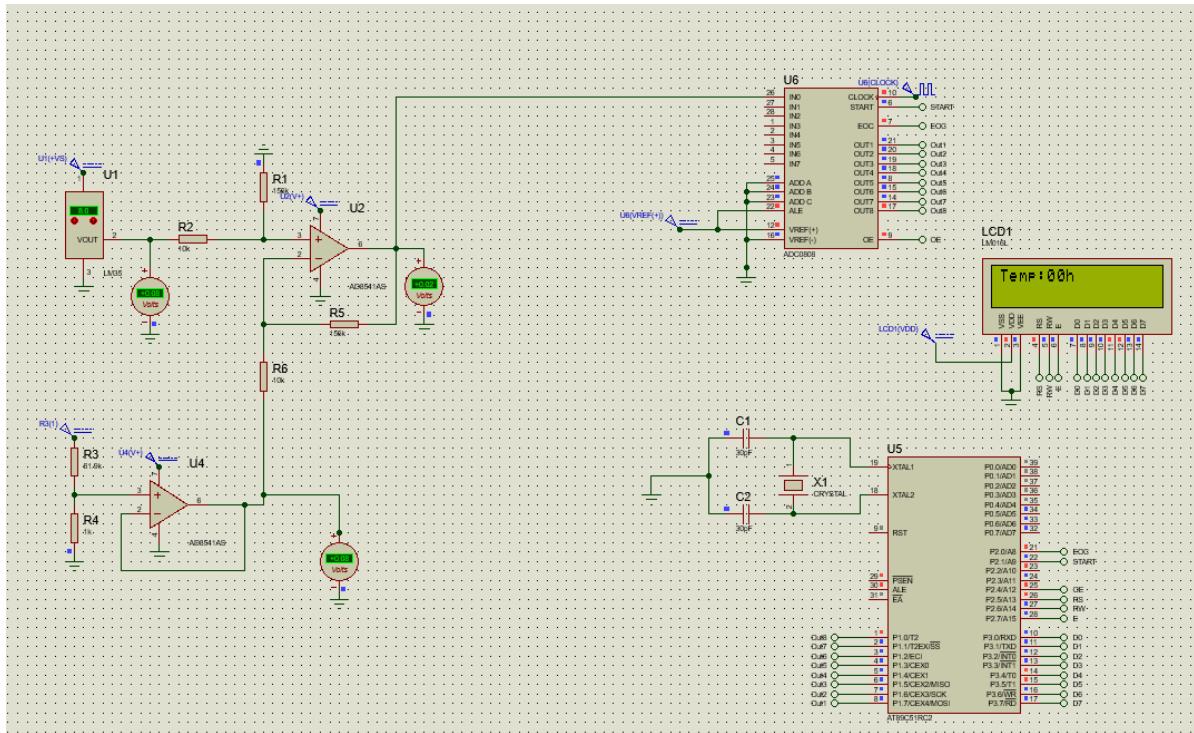
```

```

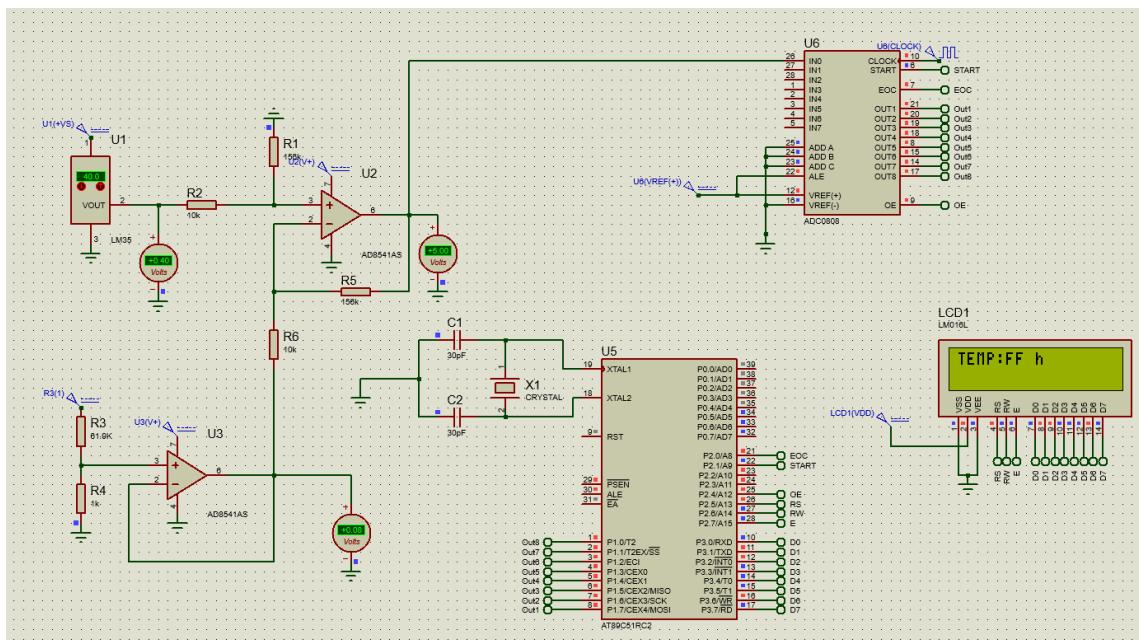
LCD_C.c
109     RS=1;                // Selectarea registrului de date
110     RW=0;                // Setarea in modul de scriere
111     EN=1;                // Activarea LCD-ului
112     delay(5);           // Întârziere
113     EN=0;                // Dezactivarea LCD-ului
114     delay(5);           // Întârziere
115 }
116
117 void delay(int t)      // Functie pentru întârzierea programului
118 {
119     int j;
120     for(j=0; j<t*1275; j++);
121 }
122
123 void print(char p)    // Functie pentru afisarea numerelor pe LCD
124 {
125     //y = p >> 4;
126     y = ((p >> 4)& 0x0F) < 10 ? ((p >> 4)& 0x0F) + '0' : ((p >> 4)& 0x0F) - 10 + 'A';
127     lcd_data(y);        // Afisarea zecilor pe LCD
128
129     z = p & 0x0f;
130     z = z < 10 ? z + '0' : z - 10 + 'A';
131     lcd_data(z);        // Afisarea unitatilor pe LCD
132
133     lcd_data(0x20);    // Afisarea unui spatiu pe LCD
134     lcd_data(0x68);    // Afisarea unui caracter specific pe LCD
135 }
136

```

6.6.1 Simularea in Proteus la 8 °C=>00h pe LCD



6.6.2 Simularea in Proteus la 40°C=>FFh pe LCD



7. Tastatura

7.1 Consideratii teoretice

Butonul este un tip de comutator utilizat pentru controlul unor aparate sau numai a unor funcții ale acestora. Butoanele sunt de diferite forme și dimensiuni și se găsesc în tot felul de dispozitive, în principal, în echipamentele electrice și electronice.

Butoanele sunt în general activate atunci când sunt apăsate. Butoanele permit închiderea circuitului electric pe durata acțiunării lor. Când se încetează cu apăsarea butonului, acesta revine în poziția inițială.

Am ales butoanele în locul tasturii matriceale deoarece, keypad-ul PHONE avand cei mai puțini pini (7 pini) dar totuși un numar mai mare decât numărul de pini al celor 5 butoane (5 pini)

7.2 Schema PROTEUS

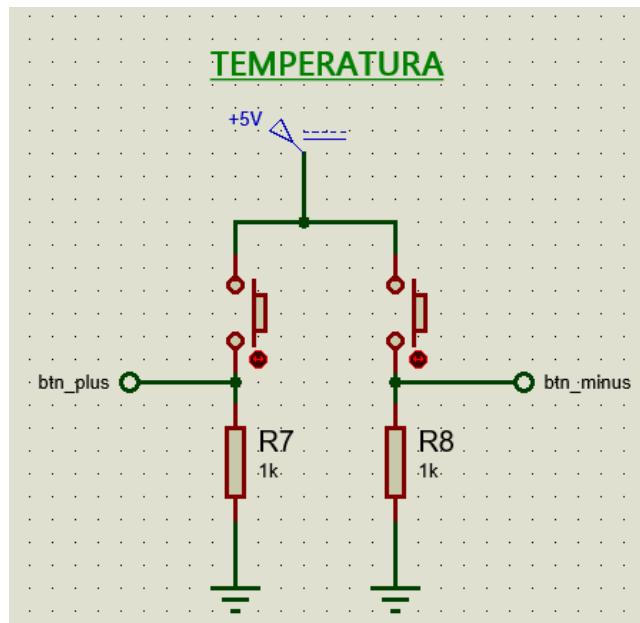


Fig.7.1 Butoanele

Pentru proiectul meu vreau sa folosesc 2 butoane:

- ✓ 1 buton btn_plus pentru cresterea (“+”) temperaturii.
- ✓ 2 buton btn_minus pentru scaderea (“-”) temperaturii.

8. Releul

8.1 Consideratii teoretice

Releele sunt componente extrem de versatile, care sunt la fel de eficient în circuitele complexe ca și în cele simple. Ele pot fi folosite în locul altor forme de comutatoare sau pot fi proiectate special pe baza unor amperajul necesar. Una dintre cele mai frecvente situații care necesită a-ți cere să releu atunci când o aplicație trebuie să treacă de la curent mare la curent redus (sau invers) în același circuit. Releele oferă amplificarea necesară pentru a converti un curent mic într-un mai mare.

Cu alte cuvinte, releul este un switch, un comutator actionat electric

8.2 Funcționarea releului

Funcționează pe principiul unei atracții electromagnetice. Când circuitul releului detectează curentul de defect, acesta energizează câmpul electromagnetic care produce câmpul magnetic temporar.

Acest câmp magnetic deplasează armătura releului pentru a deschide sau a închide conexiunile. Releul mic de putere are numai un singur contact, iar releul de putere mare are două contacte pentru deschiderea comutatorului.

Secțiunea interioară a releului este prezentată în figura de mai sus. Are un miez de fier care este înfășurat de o bobină de control. Sursa de alimentare este furnizată bobinei prin contactele sarcinii și comutatorul de comandă. Curentul care trece prin bobină produce câmpul magnetic din jurul acestuia.

Datorită acestui câmp magnetic, brațul superior al brațului magnetului atrage bratul inferior. Prin urmare, închideți circuitul, ceea ce face ca curentul să curgă prin sarcină. Dacă contactul este deja închis, atunci se mișcă opus și, prin urmare, deschideți contactele.

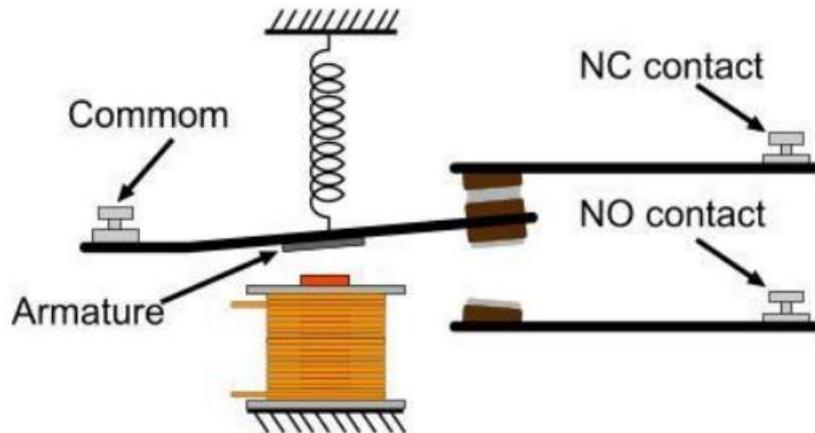


Figura 8.1 Funcționarea Releului

De ce componente avem nevoie pentru a comanda un releu?

- ✓ Rezistenta
- ✓ Un tranzistor
- ✓ Dioda de protectie
- ✓ Un releu
- ✓ Sursa de alimentare
- **Dioda** este utilizata pentru protectie si pentru a elimina spike-urile (varfurile) de tensiune din circuit ce apar in urma switch-ului din ON-OFF, respectiv OFF-ON.
- **Bobina** actioneaza ca un inductor, rezistand schimbarilor in curent .
- **Tranzistorul** functioneaza ca un amplificator, din moment ce baza acestuia primeste suficienta putere, tranzistorul va conduce de la emitor la colector si va alimenta releul.
- **Rezistenta** limiteaza curentul in baza tranzistorului pentru a nu se deteriora in cazul unui curent prea mare.

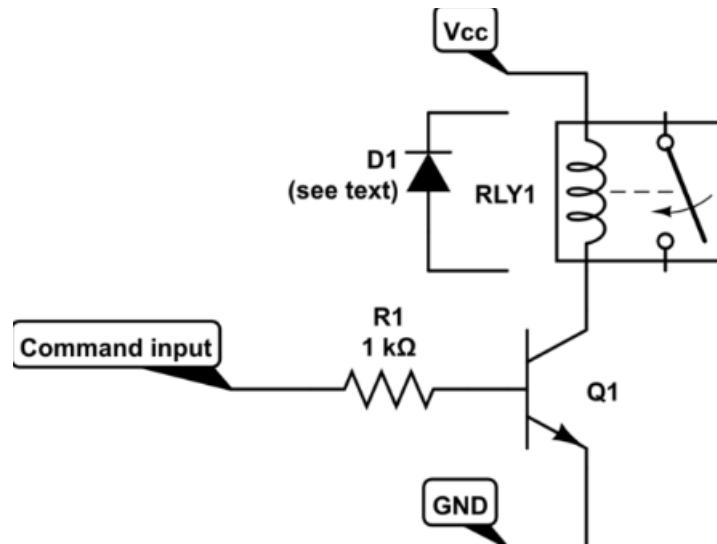


Figura 8.2 Circuitul de Comanda a unui releu

Pentru proiectul meu am ales releul **W17-DIP-7**.

- ✓ Tensiunea pe bobina: 5V
- ✓ Rezistenta bobinei: 750Ω

8.3 Schema in Proteus

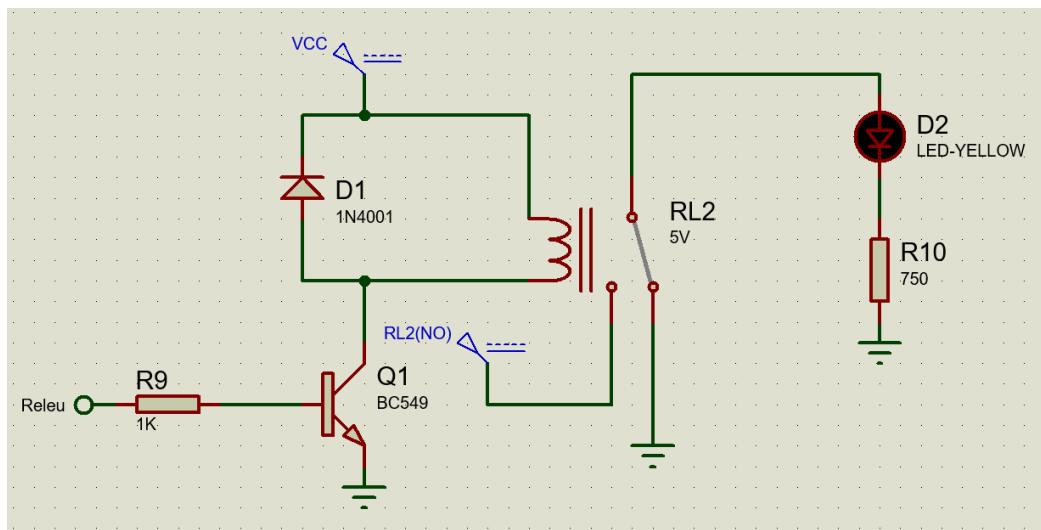


Figura 8.3 Schema Releului in Proteus

8.4 Simulare a releului in pozitia ON

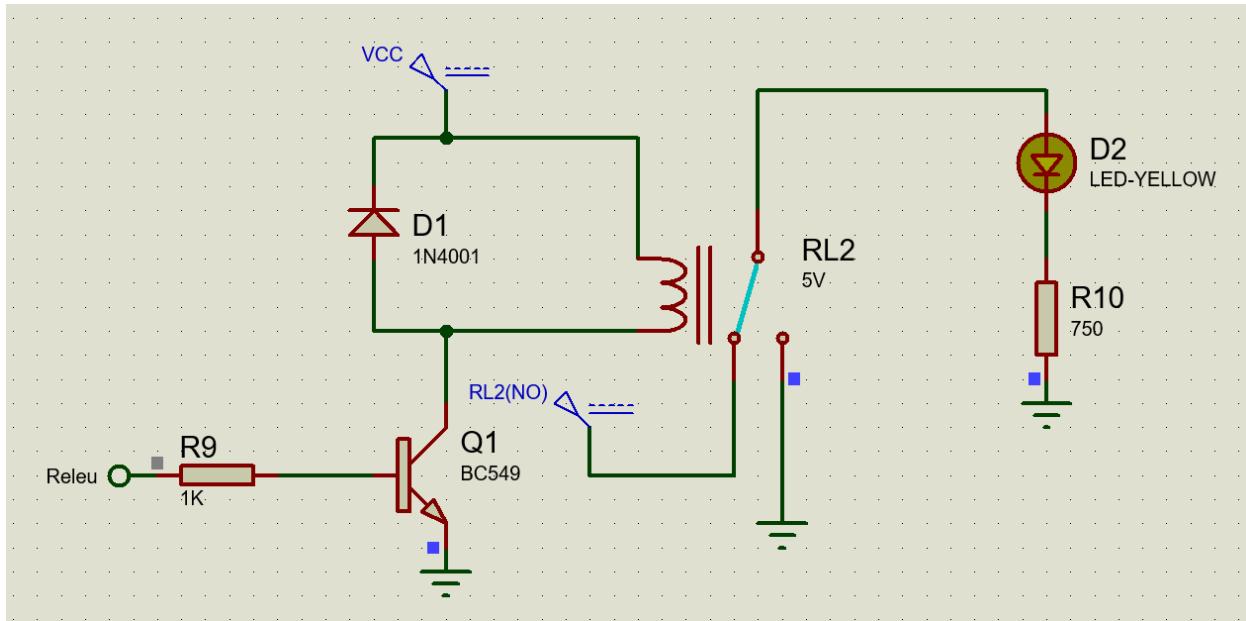


Figura 8.4 Releu ON

8.5 Simulare a releului in pozitia OFF

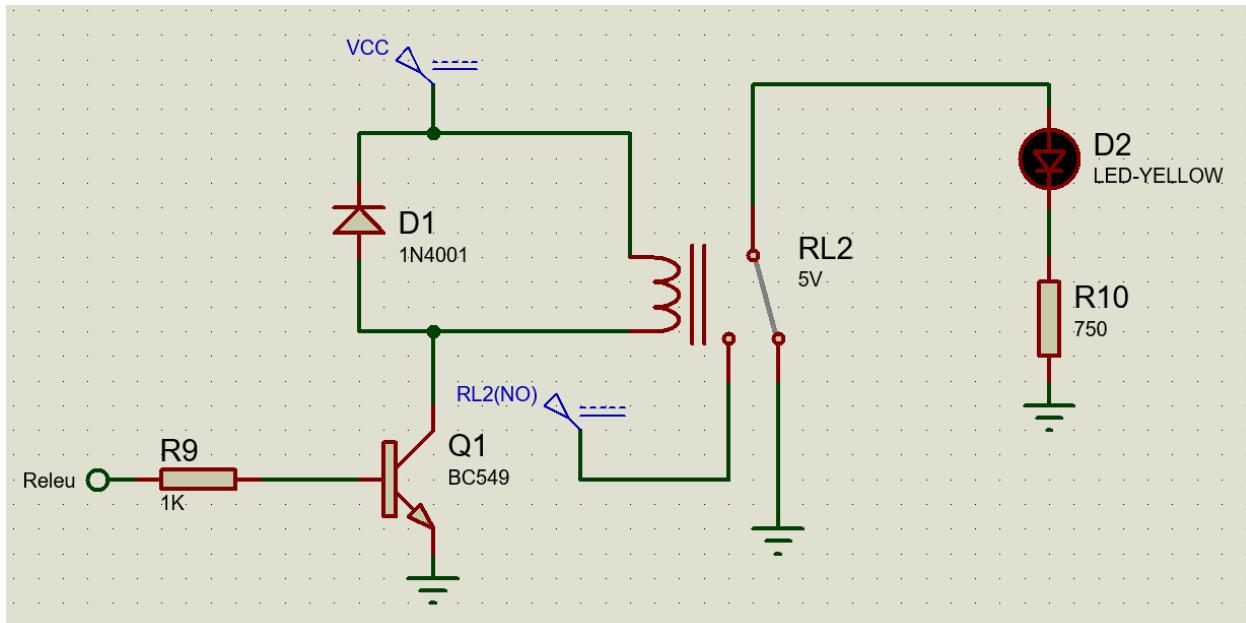


Figura 8.5 Releu OFF

9. Circuitul Final în Proteus

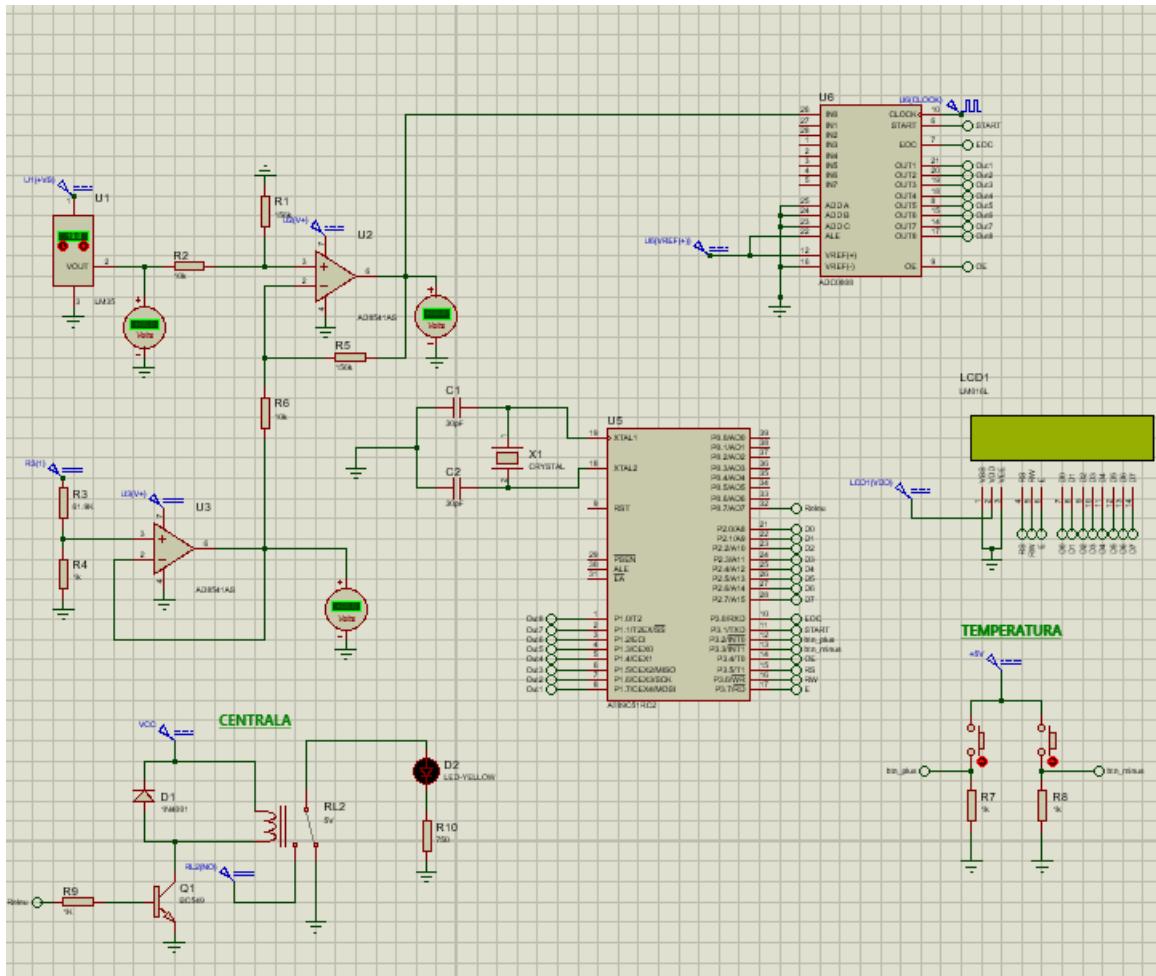
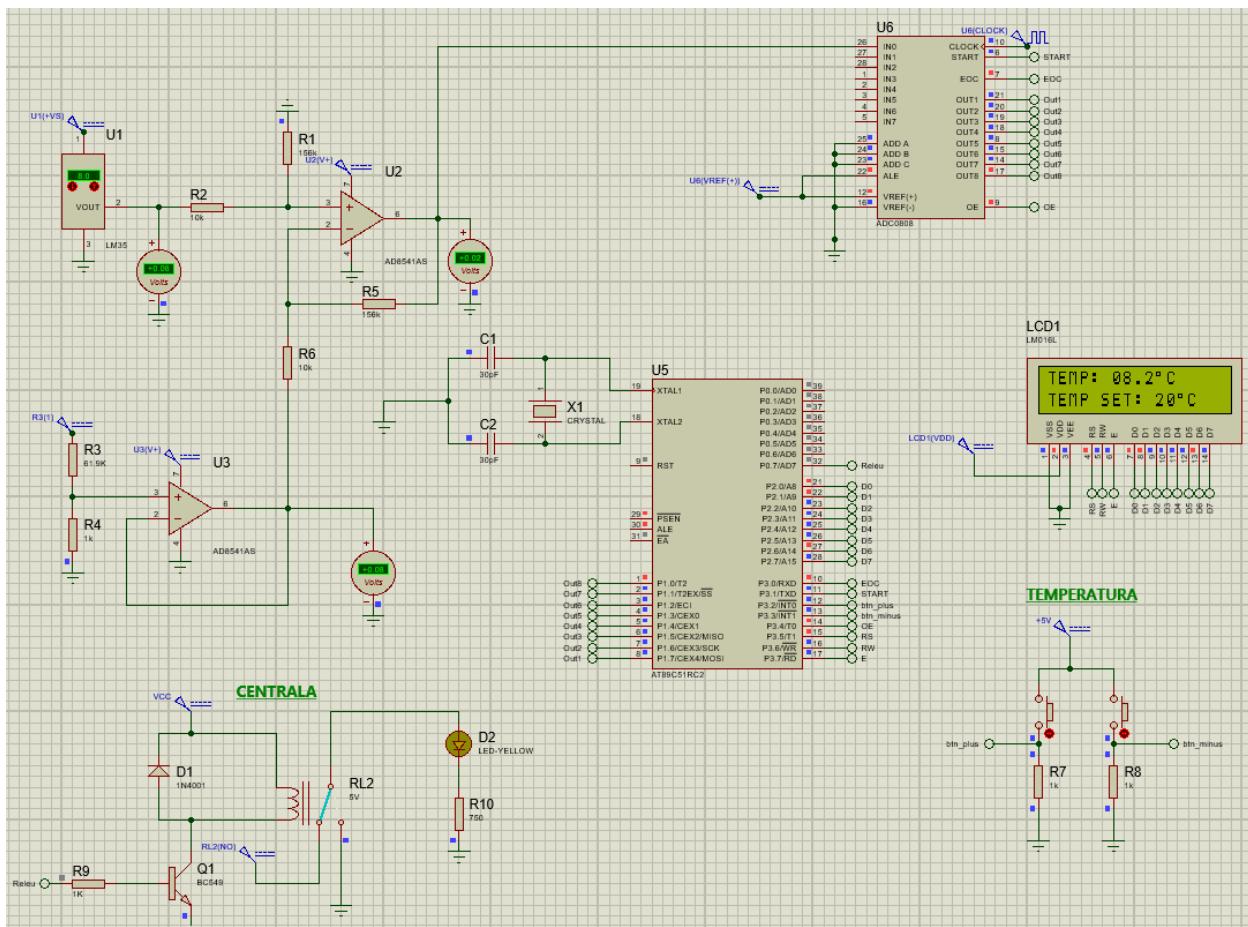


Figura 9.1 Schema Circuitului Final in Proteus

În schema finală a termostatului de cameră se regăsesc următoarele blocuri fundamentale: microcontrolerul, LCD-ul, circuitul de condiționare, butoane/tastatura (au rolul de a seta temperatura în cameră) și centrala (aceasta are rolul de a informa utilizatorul dacă tensiunea de la senzor a depășit tensiunea setată de utilizator și se oprește. Dacă temperatura setată este mai mare decât temperatura senzorului, centrala va rămâne pornită, indicată prin intermediul LED-ului galben aprins).

9.1 Simulare pentru centrala ON

Dacă temperatura măsurată de senzor este mai mică decât temperatura setată, se va aprinde LED-ul galben, indicând astfel că centrala este în funcțiune.



9.2 Simulare Centrala ON la 8°C

9.2 Simulare pentru Centrala OFF

În cazul în care temperatura măsurată de senzor depășește temperatura setată de utilizator, LED-ul galben se va stinge, indicând că centrala s-a oprit.

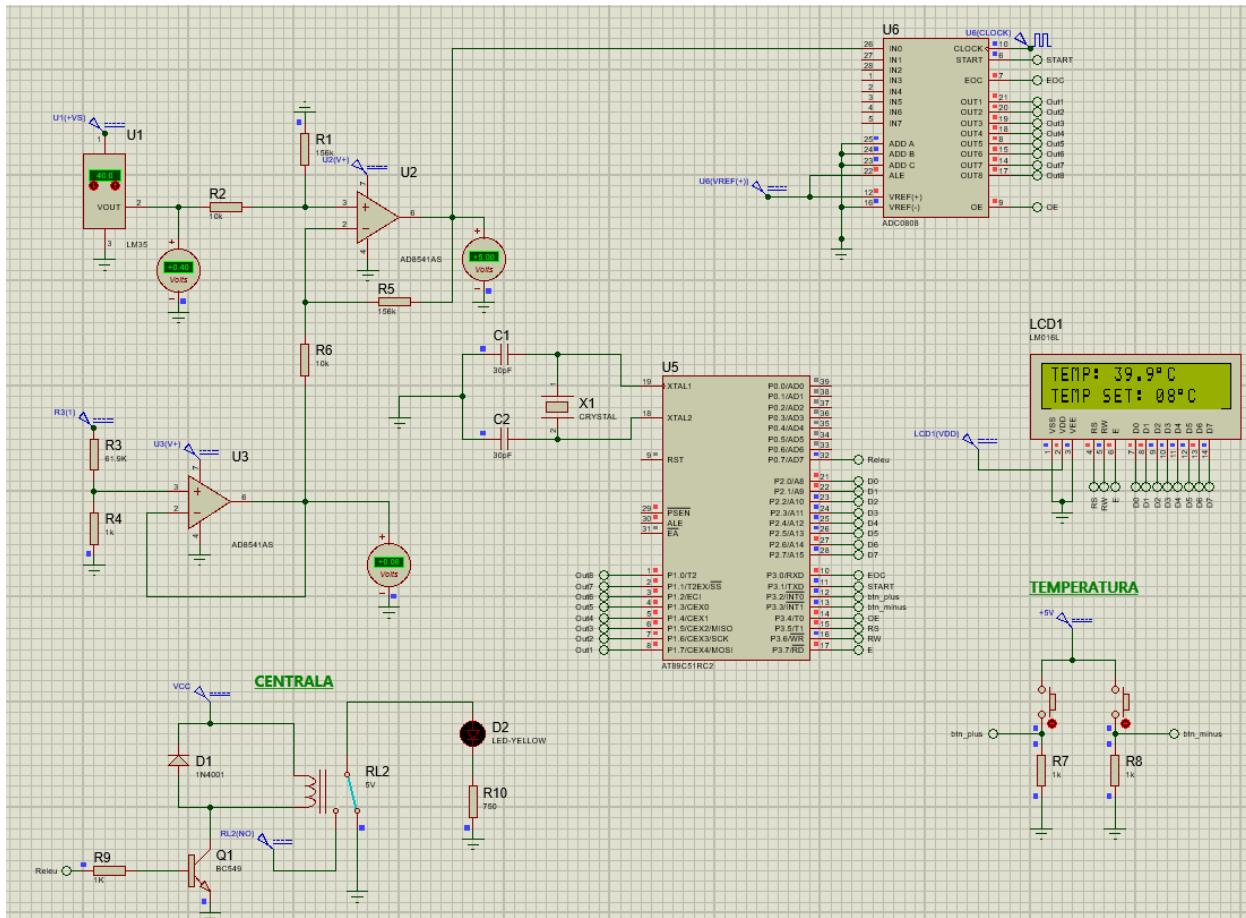
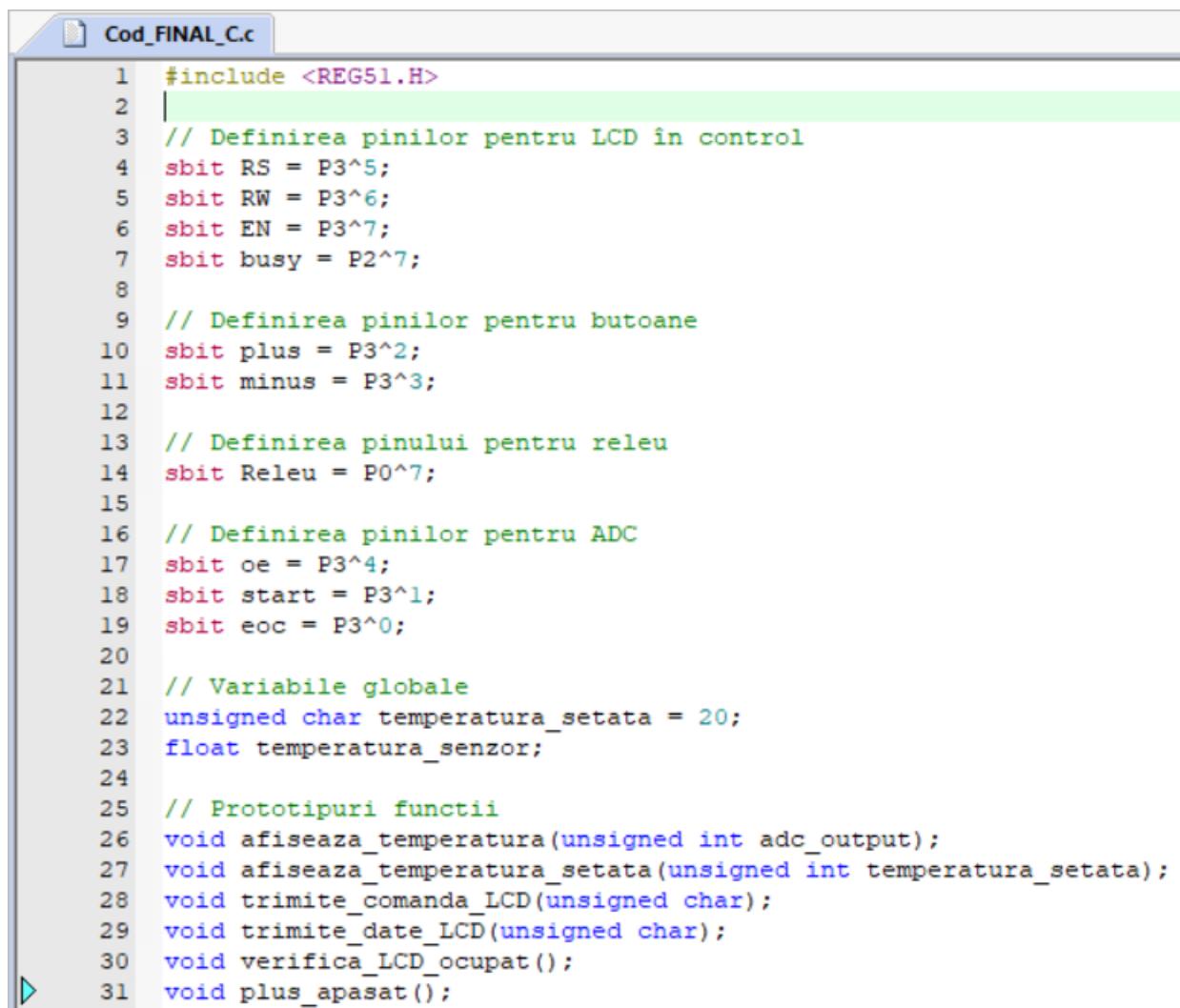


Figura 9.3 Simulare pentru Centrala OFF

9.3 Programarea Microcontrolerului în Limbajul C: Afisarea Zecimală a Temperaturii



```
Cod_FINAL_C.c
1 #include <REG51.H>
2
3 // Definirea pinilor pentru LCD in control
4 sbit RS = P3^5;
5 sbit RW = P3^6;
6 sbit EN = P3^7;
7 sbit busy = P2^7;
8
9 // Definirea pinilor pentru butoane
10 sbit plus = P3^2;
11 sbit minus = P3^3;
12
13 // Definirea pinului pentru releu
14 sbit Releu = P0^7;
15
16 // Definirea pinilor pentru ADC
17 sbit oe = P3^4;
18 sbit start = P3^1;
19 sbit eoc = P3^0;
20
21 // Variabile globale
22 unsigned char temperatura_setata = 20;
23 float temperatura_senzor;
24
25 // Prototipuri functii
26 void afiseaza_temperatura(unsigned int adc_output);
27 void afiseaza_temperatura_setata(unsigned int temperatura_setata);
28 void trimite_comanda_LCD(unsigned char);
29 void trimite_date_LCD(unsigned char);
30 void verifica_LCD_ocupat();
31 void plus_apasat();
```

```

Cod_FINAL_C.c
31 void plus_apasat();
32 void minus_apasat();
33 void intarziere(unsigned int);
34
35 // Variabila pentru stocarea datelor ADC
36 unsigned int adc_data;
37
38 void main() {
39     // Mesaje pentru afisare
40     unsigned char textTmas[] = "TEMP: ";
41     unsigned char textTdorita[] = "TEMP SET:";
42     unsigned char i;
43
44     eoc = 1; // Seteaza EOC ca intrare
45     oe = 0; // Seteaza OE ca iesire
46     start = 0;
47
48     P1 = 0xFF; // Seteaza P1 ca port de intrare pentru citirea ADC
49     Releu = 0;
50     plus = 0;
51     minus = 0;
52     IE = 0x85; // Activeaza intreruperile externe INT0 si INT1
53     ITO = 1; // Configureaza INT0 pentru front descendant
54     IT1 = 1; // Configureaza INT1 pentru front descendant
55     TMOD = 0x01; // Timer 0 in Modul 1 (16-bit)
56     TH0 = 0xFC;
57     TL0 = 0x66;
58
59     // Initializare LCD
60     intarziere(20); // Asteapta pentru a permite pornirea LCD-ului
61     trimite_comanda_LCD(0x38); // Initializare LCD: 2 linii, matrice 5x7

```



```

Cod_FINAL_C.c
61     trimite_comanda_LCD(0x38); // Initializare LCD: 2 linii, matrice 5x7
62     intarziere(5);
63     trimite_comanda_LCD(0x0C); // LCD ON, cursor OFF
64     intarziere(5);
65     trimite_comanda_LCD(0x01); // ?terge display-ul
66     intarziere(5);
67     trimite_comanda_LCD(0x02); // Returneaza cursorul la pozitia 0 de pe prima linie
68     intarziere(5);
69
70     // Afisare "Temp: "
71     for(i = 0; i < 6; i++) {
72         trimite_date_LCD(textTmas[i]);
73     }
74
75     // Afisare "Temp Set: "
76     trimite_comanda_LCD(0xC0); // Muta cursorul pe a doua linie
77     for(i = 0; i < 10; i++) {
78         trimite_date_LCD(textTdorita[i]);
79     }
80
81     while(1) {
82         // Initiere conversie ADC
83         start = 1;
84         intarziere(1);
85         start = 0;
86
87         // Asteapta finalizarea conversiei ADC
88         while(eoc == 1);
89         while(eoc == 0);
90
91         oe = 1; // Activeaza iesirea ADC
92         . . . . .

```

Cod_FINAL_C.c

```

88     while(eoc == 1);
89     while(eoc == 0);

90
91     oe = 1; // Activeaza iesirea ADC
92     adc_data = P1; // Citeste valoarea ADC
93
94     // Afisare temperatura senzorului
95     trimite_comanda_LCD(0x86);
96     afiseaza_temperatura(adc_data);
97     trimite_date_LCD(0xDF); // Simbol grad
98     trimite_date_LCD(0x43); // 'C'
99
100    // Afisare temperatura setata
101    trimite_comanda_LCD(0xCA);
102    afiseaza_temperatura_setata(temperatura_setata);
103    trimite_date_LCD(0xDF); // Simbol grad
104    trimite_date_LCD(0x43); // 'C'
105
106    // Controlul releului
107    if (temperatura_senzor >= temperatura_setata) {
108        Releu = 0; // Opriti releul
109    } else {
110        Releu = 1; // Porniti releul
111    }
112
113    intarziere(10); // Asteapta pentru a observa afisajul
114}
115}
116
117 // Trimite o comanda catre LCD
118 void trimite_comanda_LCD(unsigned char c) {

```

Cod_FINAL_C.c

```

118 void trimite_comanda_LCD(unsigned char c) {
119     verifica_LCD_ocupat();
120     P2 = c;
121     RS = 0;
122     RW = 0;
123     EN = 1;
124     intarziere(1);
125     EN = 0;
126 }
127
128 // Trimite date catre LCD
129 void trimite_date_LCD(unsigned char c) {
130     verifica_LCD_ocupat();
131     P2 = c;
132     RS = 1;
133     RW = 0;
134     EN = 1;
135     intarziere(1);
136     EN = 0;
137 }
138
139 // Verifica daca LCD-ul este ocupat
140 void verifica_LCD_ocupat() {
141     busy = 1;
142     RS = 0;
143     RW = 1;
144     P2 = 0xFF; // Seteaza P2 ca intrare pentru a citi flagul de ocupare
145
146     while(busy) {
147         EN = 0;
148         intarziere(1);

```

```

148     intarziree(1);
149     EN = 1;
150 }
151 P2 = 0x00; // Seteaza P2 ca ie?ire dupa citire
152 }
153 }
154 // Functie de intarziree
155 void intarziree(unsigned int timp) {
156     unsigned int i, j;
157     for(i = 0; i < timp; i++) {
158         for(j = 0; j < 1275; j++);
159     }
160 }
161 }
162 // Functie pentru afisarea temperaturii senzorului
163 void afiseaza_temperatura(unsigned int adc_output) {
164     char cifre_temperatura[3];
165
166     // Calcularea temperaturii pe baza valorii ADC
167     float interval_temperatura = 32.0; // Intervalul de temperatura (40 - 8)
168     float increment_temperatura = interval_temperatura / 256.0; // Increment de temperatura pentru fiecare unitate ADC
169
170     float temperatura = increment_temperatura * adc_output + 8.0; // Conversie la temperatura efectiva
171
172     // Extracteaza partea intreaga si zecimala a temperaturii
173     int parte_intreaga = (int)temperatura;
174     int parte_zecimala = (int)((temperatura - parte_intreaga) * 10 + 1); // Zecimale
175
176     // Convertirea cifrelor la ASCII
177     cifre_temperatura[2] = (char)(parte_intreaga / 10) + '0'; // Zeci
178     cifre_temperatura[1] = (char)(parte_intreaga % 10) + '0'; // Unitati
179     cifre_temperatura[0] = (char)(parte_zecimala + '0'); // Zecimala
180
181     // Afisare temperatura pe LCD
182     trimit_date_LCD(cifre_temperatura[2]);
183     trimit_date_LCD(cifre_temperatura[1]);
184     trimit_date_LCD(0x2E); // Punct zecimal
185     trimit_date_LCD(cifre_temperatura[0]);
186
187     temperatura_senzor = temperatura;
188 }
189
190 // Functie pentru afisarea temperaturii setate
191 void afiseaza_temperatura_setata(unsigned int temperatura_setata) {
192     char cifre_temperatura[2];
193
194     cifre_temperatura[1] = (char)(temperatura_setata / 10) + '0'; // Zeci
195     cifre_temperatura[0] = (char)(temperatura_setata % 10) + '0'; // Unitati
196
197     // Afisare temperatura setata pe LCD
198     trimit_date_LCD(cifre_temperatura[1]);
199     trimit_date_LCD(cifre_temperatura[0]);
200 }
201
202 // Functie pentru incrementarea temperaturii setate
203 void plus_apasat() interrupt 0 {
204     if (temperatura_setata < 40) {
205         temperatura_setata++;
206     }
207 }

```

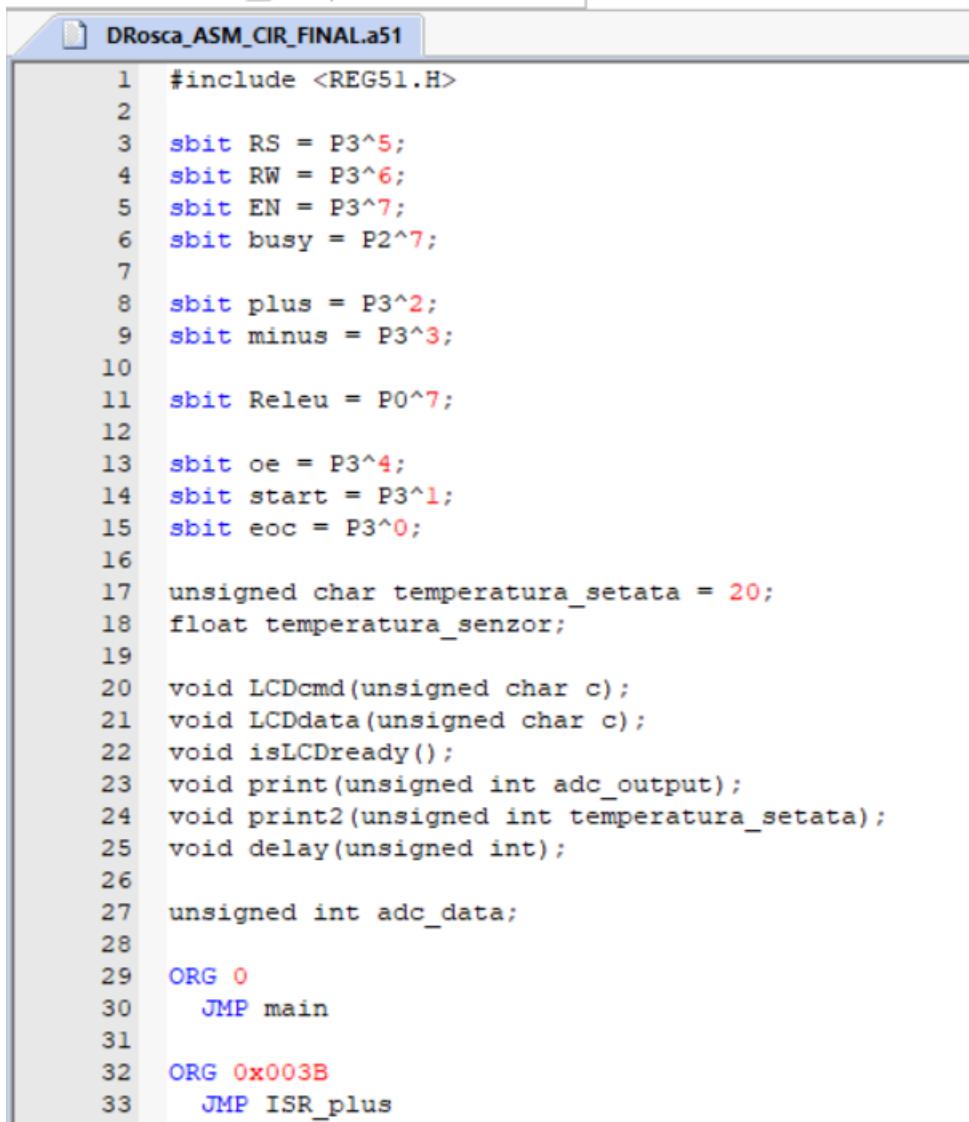
```

175
176     // Convertirea cifrelor la ASCII
177     cifre_temperatura[2] = (char)(parte_intreaga / 10) + '0'; // Zeci
178     cifre_temperatura[1] = (char)(parte_intreaga % 10) + '0'; // Unitati
179     cifre_temperatura[0] = (char)(parte_zecimala + '0'); // Zecimala
180
181     // Afisare temperatura pe LCD
182     trimit_date_LCD(cifre_temperatura[2]);
183     trimit_date_LCD(cifre_temperatura[1]);
184     trimit_date_LCD(0x2E); // Punct zecimal
185     trimit_date_LCD(cifre_temperatura[0]);
186
187     temperatura_senzor = temperatura;
188 }
189
190 // Functie pentru afisarea temperaturii setate
191 void afiseaza_temperatura_setata(unsigned int temperatura_setata) {
192     char cifre_temperatura[2];
193
194     cifre_temperatura[1] = (char)(temperatura_setata / 10) + '0'; // Zeci
195     cifre_temperatura[0] = (char)(temperatura_setata % 10) + '0'; // Unitati
196
197     // Afisare temperatura setata pe LCD
198     trimit_date_LCD(cifre_temperatura[1]);
199     trimit_date_LCD(cifre_temperatura[0]);
200 }
201
202 // Functie pentru incrementarea temperaturii setate
203 void plus_apasat() interrupt 0 {
204     if (temperatura_setata < 40) {
205         temperatura_setata++;
206     }
207 }

```

```
201
202 // Functie pentru incrementarea temperaturii setate
203 void plus_apasat() interrupt 0 {
204     if (temperatura_setata < 40) {
205         temperatura_setata++;
206     }
207 }
208
209 // Functie pentru decrementarea temperaturii setate
210 void minus_apasat() interrupt 2 {
211     if (temperatura_setata > 8) {
212         temperatura_setata--;
213     }
214 }
215
```

9.4 Programarea Microcontrolerului în Limbajul ASM: Afişarea Zecimală a Temperaturii



```
DRosca_ASM_CIR_FINAL.a51
1 #include <REG51.H>
2
3 sbit RS = P3^5;
4 sbit RW = P3^6;
5 sbit EN = P3^7;
6 sbit busy = P2^7;
7
8 sbit plus = P3^2;
9 sbit minus = P3^3;
10
11 sbit Releu = P0^7;
12
13 sbit oe = P3^4;
14 sbit start = P3^1;
15 sbit eoc = P3^0;
16
17 unsigned char temperatura_setata = 20;
18 float temperatura_senzor;
19
20 void LCDcmd(unsigned char c);
21 void LCDdata(unsigned char c);
22 void isLCDready();
23 void print(unsigned int adc_output);
24 void print2(unsigned int temperatura_setata);
25 void delay(unsigned int);
26
27 unsigned int adc_data;
28
29 ORG 0
30     JMP main
31
32 ORG 0x003B
33     JMP ISR_plus
```

DRosca_ASM_CIR_FINAL.a51

```
31
32     ORG 0x003B
33     JMP ISR_plus
34
35     ORG 0x0013
36     JMP ISR_minus
37
38     main:
39         MOV eoc, #1
40         MOV oe, #0
41         MOV start, #0
42         MOV IE, #0x85 ; enable external interrupts INTO and INT1
43         MOV IT0, #1 ; configure INTO for falling edge
44         MOV IT1, #1 ; configure INT1 for falling edge
45         MOV TMOD, #0x01 ; Timer 0 in Mode 1
46         MOV TH0, #0xFC
47         MOV TL0, #0x66
48
49         ; Init LCD
50         MOV R1, #20 ; delay(20)
51         ACALL delay
52         MOV R1, #0x38 ; LCDcmd(0x38)
53         ACALL LCDcmd
54         MOV R1, #5 ; delay(5)
55         ACALL delay
56         MOV R1, #0x0C ; LCDcmd(0x0C)
57         ACALL LCDcmd
58         MOV R1, #5 ; delay(5)
59         ACALL delay
60         MOV R1, #0x01 ; LCDcmd(0x01)
61         ACALL LCDcmd
62         MOV R1, #5 ; delay(5)
63         ACALL delay
```

```
DRosca_ASM_CIR_FINAL.a51

61    ACALL LCDcmd
62    MOV R1, #5 ; delay(5)
63    ACALL delay
64    MOV R1, #0x02 ; LCDcmd(0x02)
65    ACALL LCDcmd
66    MOV R1, #5 ; delay(5)
67    ACALL delay
68
69    ; Display "Temp: "
70    MOV R1, #6
71    MOV R2, #0 ; i = 0
72    display_temp1:
73    MOV A, textTmas[R2]
74    ACALL LCDdata
75    INC R2
76    DJNZ R1, display_temp1
77
78    ; Display "Temp Set: "
79    MOV R1, #10
80    MOV R2, #0 ; i = 0
81    MOV R3, #0xC0 ; LCDcmd(0xC0)
82    ACALL LCDcmd
83    display_temp2:
84    MOV A, textTdorita[R2]
85    ACALL LCDdata
86    INC R2
87    DJNZ R1, display_temp2
88
89    ; Main loop
90    main_loop:
91    MOV start, #1
92    MOV R1, #1 ; delay(1)
93    ACALL delay
```

```
DRosca_ASM_CIR_FINAL.a51

91    MOV start, #1
92    MOV R1, #1 ; delay(1)
93    ACALL delay
94    MOV start, #0
95
96    wait_conversion:
97        MOV A, eoc
98        CJNE A, #1, wait_conversion ; while(eoc == 1)
99        MOV A, eoc
100       CJNE A, #0, wait_conversion ; while(eoc == 0)
101
102       MOV oe, #1
103
104       MOV A, P1
105       MOV adc_data, A
106
107       ; Display sensor temperature
108       MOV R1, #0x86 ; LCDcmd(0x86)
109       ACALL LCDcmd
110       MOV A, adc_data
111       ACALL print
112       MOV A, #0xDF ; degree symbol
113       ACALL LCDdata
114       MOV A, #0x43 ; C
115       ACALL LCDdata
116
117       ; Display set temperature
118       MOV R1, #0xCA ; LCDcmd(0xCA)
119       ACALL LCDcmd
120       MOV R1, temperatura_setata
121       ACALL print2
122       MOV A, #0xDF ; degree symbol
123       ACALL LCDdata
```

```
DRosca_ASM_CIR_FINAL.a51

121    ACALL print2
122    MOV A, #0xDF ; degree symbol
123    ACALL LCDdata
124    MOV A, #0x43 ; C
125    ACALL LCDdata
126
127    ; Control the relay
128    MOV A, temperatura_senzor
129    CJMP A, temperatura_setata, turn_off_relay
130    SJMP continue_main_loop
131
132    turn_off_relay:
133    MOV Releu, #0 ; Turn off the relay
134    SJMP continue_main_loop
135
136    continue_main_loop:
137    MOV R1, #100 ; delay(1000)
138    ACALL delay
139    SJMP main_loop
140
141    ISR_plus:
142    MOV A, temperatura_setata
143    CJMP A, #40, no_increment
144    INC temperatura_setata
145    no_increment:
146    RETI
147
148    ISR_minus:
149    MOV A, temperatura_setata
150    CJMP A, #8, no_decrement
151    DEC temperatura_setata
152    no_decrement:
153    RETI
```

```
DRosca_ASM_CIR_FINAL.a51
151    DEC temperatura_setata
152    no_decrement:
153        RETI
154
155    LCDcmd:
156        ACALL isLCDready
157        MOV P2, R1
158        MOV RS, #0
159        MOV RW, #0
160        MOV EN, #1
161        MOV R1, #1 ; delay(1)
162        ACALL delay
163        MOV EN, #0
164        RET
165
166    LCDdata:
167        ACALL isLCDready
168        MOV P2, R1
169        MOV RS, #1
170        MOV RW, #0
171        MOV EN, #1
172        MOV R1, #1 ; delay(1)
173        ACALL delay
174        MOV EN, #0
175        RET
176
177    isLCDready:
178        MOV busy, #1
179        MOV RS, #0
180        MOV RW, #1
181        MOV P2, #0xFF ; Set P2 as input to read busy flag
182
183    check_busv:
```

```
DRosca_ASM_CIR_FINAL.a51

181     MOV P2, #0xFF ; Set P2 as input to read busy flag
182
183     check_busy:
184         MOV A, busy
185         CJNE A, #0, check_busy
186         MOV P2, #0x00 ; Set P2 as output after reading
187         RET
188
189     delay:
190         MOV R3, #0
191     delay_outer:
192         MOV R2, #0
193     delay_inner:
194         INC R2
195         DJNZ R2, delay_inner
196         INC R3
197         CJNE R3, R1, delay_outer
198         RET
199
200     print:
201         MOV R7, #0 ; integer_part
202         MOV R6, #0 ; mantissa_part
203         MOV R5, #32 ; temperature_range
204
205         MOV A, adc_output
206         MUL AB, R5
207         MOV R6, B ; mantissa_part
208         MOV A, R0
209         ADD A, #8
210         MOV R7, A ; integer_part
211
212         MOV R1, #10 ; R1 = 10
213         DIV AB, R1
```

```
DRosca_ASM_CIR_FINAL.a51

214     MOV A, B
215     ADD A, #48
216     ACALL LCDdata
217
218     print:
219     MOV R7, #0 ; integer_part
220     MOV R6, #0 ; mantissa_part
221     MOV R5, #32 ; temperature_range
222
223     MOV A, adc_output
224     MUL AB, R5
225     MOV R6, B ; mantissa_part
226     MOV A, R0
227     ADD A, #8
228     MOV R7, A ; integer_part
229
230     MOV R1, #10 ; R1 = 10
231     DIV AB, R1
232     MOV A, B
233     ADD A, #48
234     ACALL LCDdata
235
236     ; Display tens digit of integer part
237     MOV A, R7
238     DIV AB, R1
239     MOV A, B
240     ADD A, #48
241     ACALL LCDdata
242
243     ; Display decimal point
244     MOV A, #0x2E
245     ACALL LCDdata
246
```

```
DRosca_ASM_CIR_FINAL.a51
247      ; Display ones digit of integer part
248      MOV A, R7
249      MOV B, R1
250      MUL AB, B
251      SUBB A, #48
252      ACALL LCDdata
253
254      ; Set temperatura_senzor
255      MOV A, R7
256      MOV R6, #10
257      MUL AB, R6
258      MOV R0, A
259      MOV A, R6
260      MOV B, R1
261      DIV AB, R1
262      ADD A, B
263      MOV temperatura_senzor, A
264      RET
265
266      print2:
267      MOV R1, #10    ; R1 = 10
268
269      ; Display tens digit of temperatura_setata
270      MOV A, temperatura_setata
271      DIV AB, R1
272      MOV A, B
273      ADD A, #48
274      ACALL LCDdata
275
276      ; Display ones digit of temperatura_setata
277      MOV A, temperatura_setata
278      MOV B, R1
279      MUL AB, B
```

10. Bibliografie

1. https://www.analog.com/media/en/technical-documentation/datasheets/ad8541_8542_8544.pdf
2. https://www.ti.com/lit/eb/slyy161/slyy161.pdf?ts=1614791213989&ref_url=https%253A%252F%252Fwww.google.com%252F
3. <https://uk.farnell.com/maxim-integrated-products/ds18b20/temperature-sensor/dp/2515553?st=ds18b20+>
4. <http://www.farnell.com/datasheets/2345098.pdf>
5. <https://uk.farnell.com/maxim-integrated-products/ds18s20/digital-thermometer-18s20-3to/dp/2519401>
6. <https://uk.farnell.com/maxim-integrated-products/ds1822/temperature-sensor-2deg-c-to-92/dp/2515551?st=ds18>
7. [ASCII table - Table of ASCII codes, characters and symbols \(ascii-code.com\)](#)
8. [LCD Interfacing | LCD Interfacing with Microprocessor 8086 | 16 * 2 LM016L Proteus 8 | Urdu Hindi \(youtube.com\)](#)
9. <https://www.mobilissimo.ro/articole-telefoane/lcd-versus-amoled-care-sunt-diferentele-dintre-cele-doua-tipuri-de-display>
10. https://en.wikipedia.org/wiki/Plasma_display
11. <https://ardushop.ro/ro/43-display-uri-si-led-uri>
12. <https://nwww.interelectronix.com/ro/o-scurta-explicatie-diferentelor-de-afisare-oled-lcd-sau-amoled.html>
13. <https://misterfix.ro/blog/afisaj-cu-cristale-lichide-ce-este-lcd-si-cum-functioneaza/>
14. <https://itotem.ro/blog/display-stretched-video-wall-ecrane-led/>
15. [Crystal Oscillator and Quartz Crystal Oscillators \(electronics-tutorials.ws\)](#)
16. https://en.wikipedia.org/wiki/Intel_MCS-48#/media/File:KL_Intel_P8048H.jpg
17. https://en.wikipedia.org/wiki/MCS-51#/media/File:KL_Intel_P8051.jpg

18. [https://www.google.com/search?scasession=5d0811d5ae0715ef&scay=1&sxsrf=ACQVn0-PEH09HoqOQLUepfpdkh1DZpMpg:1713909158380&q=80C196+\(Intel+MCS-96\)+imagine&spell=1&sa=X&ved=2ahUKEwiL1LUqdmFAXXQh_0HHXPJCz8QBSgAegQIBxAC&biw=1536&bih=730&dpr=1.25#vhid=qBJxQG4rZ0_RNM&vssid=1](https://www.google.com/search?scasession=5d0811d5ae0715ef&scay=1&sxsrf=ACQVn0-PEH09HoqOQLUepfpdkh1DZpMpg:1713909158380&q=80C196+(Intel+MCS-96)+imagine&spell=1&sa=X&ved=2ahUKEwiL1LUqdmFAXXQh_0HHXPJCz8QBSgAegQIBxAC&biw=1536&bih=730&dpr=1.25#vhid=qBJxQG4rZ0_RNM&vssid=1)
19. https://en.wikipedia.org/wiki/Motorola_68HC11#/media/File:KL_Motorola_68HC11.jpg
20. https://www.google.com/search?scasession=5d0811d5ae0715ef&scay=1&sxsrf=ACQVn0-wNTP-VDzvgoQoFGCB2NFTSMO73g:1713909429571&q=PIC1650+image&uds=AMwkrPth2lwGf-lsu6J4oEsDB3N5D25R8rQHcJH4z9d1OsQVqIJph9tT3XsGg5YCCXP_IzgA5UyNEHbvqX7mkhxWV_IyTojVqNhAwBHL_3mFQ6tyhOme9K-X4j7WnlumPcRn1_DzWuEgfmboso_ENzgBjCBgTq-Nkx7cu8YfXxfOYsc6v_YGDioG1KCVCdwGFgSKEml7vdzUJZYduygTOdj1Ot1YGe3nzjdTeSO2vwd9LK18Wp0K9ObH9Rlf2fyLm-7qcbu_pJMxkQV-5GPGGD_2OiMsOnvZBZnkWmvUXjrwXZKY3TFDoI&udm=2&prmd=ivnbz&sa=X&ved=2ahUKEwiq5OeVqtmFAXWqh_0HHZDaBYQQtKgLegQICxAB&biw=1536&bih=730&dpr=1.25#vhid=EHutLrVtnaOUmM&vssid=mosaic
21. https://www.google.com/search?scasession=5d0811d5ae0715ef&scay=1&sxsrf=ACQVn098xrzZg7vXfStKabdQ5rKLABuAbg:1713909602851&q=z8+microcontroller&uds=AMwkrPubqdDjBmC7DhicLtpZCZYiITcpClTexmo70EdKNy7hPTYxp2DEyUJRlh5Mh1-fTfM35MutO74MMR0Jiistovhva3KtpGyGbb7eS0hjixExCvVulNfo6oPZkwLW_TfECgX1YcfLrDr-6oy_ehp1wOPXg1mbuUAqJ-u2lAWctg14k7tw4S0PqqUt-Ewitf5mdb5fzaUluTr2vVa3TIn7EM9jkVSrOMWsqtY8zmXikAuVTOniTrG4lSlhn1LUZs_9s7DS4gvHrCsTIMi4TOqc8j65Ku-nX7gH10DGPTvL93ybPADGSvnBNi5XnNZ9muUv3667uT&udm=2&prmd=sivnbz&sa=X&ved=2ahUKEwj_7bfoqtmFAXWuh_0HHVC1BGwQtKgLegQIDRAB&biw=1536&bih=730&dpr=1.25#vhid=uR5cO9iOKfwVVM&vssid=mosaic

-
22. Digital Thermometer using 8051 and ADC 0808 Interfacing With 8051
(embeddedprojects222.blogspot.com)
23. https://www.youtube.com/watch?v=rnDS9ZQrZw8&ab_channel=StarTechnology5G
24. https://www.youtube.com/watch?v=zTadMNxkxXM&ab_channel=StarTechnology5G
25. https://www.youtube.com/watch?v=Dy_uaqiRZBY&ab_channel=StarTechnology5G
26. https://www.youtube.com/watch?v=7VTYBKf9Cqk&ab_channel=StarTechnology5G
27. <https://www.electronics-lab.com/top-10-popular-microcontrollers-among-makers/>
28. <https://ro.farnell.com/microchip/at89c51cc03ca-rltum/mcu-8bit-8051-60mhz-vqfp-44/dp/2318801>
29. <https://www.flyrobo.in/at89c51-microcontroller-ic-ics-integrated-circuits-chips-core-electronics>
30. Indrumator laborator microcontrolere :
– Dorin Petreus | Enikő Szilágyi | Radu Etz | Toma Pătărău