

MICROCONTROLERE PROIECT

Facultatea: Electronică, Telecomunicații și Tehnologia Informației (ETTI UTCN)

Nume student: Lupu Miruna

Grupa: 2133

Specializarea: Electronică Aplicată

CUPRINS

1. TEMATICA PROIECTULUI	4
2. SCHEMA BLOC	5
3. SENZORI ANALOGICI	6
3.1. Senzorul capacitiv.....	6
3.2. Senzorul inductiv	7
3.3. Senzorul semiconductor.....	8
3.4. Sensorul cu termocuplu	9
3.5. Senzorul rezistiv.....	10
3.5.1. Senzorul rezistiv: Senzorul termorezistiv	11
3.5.2. Senzorul rezistiv: Senzorul tensometric	13
3.5.3. Senzorul rezistiv: Senzorul potențiometric.....	14
3.5.4. Senzorul rezistiv: Senzorul cu contacte	15
3.5.5. Senzorul rezistiv: Senzorul piezorezistiv	16
4. SENZORI	17
5. TABEL DE COMPARAȚIE PENTRU SENZORI.....	20
6. CIRCUITUL DE ADAPTARE.....	21
7. CONVERTORUL ANALOGIC-DIGITAL	23
8. SCHEMELE IN PROTEUS PENTRU 8°C SI 40°C	25
9. CONSIDERATII TEORETICE: LCD	27
10. LCD-UL ALES.....	29
11. CIRCUITUL IN PROTEUS PENTRU SCRIEREA NUMELUI SI A GRUPEI PE UN LCD.....	31
12. CE ESTE UN MICROCONTROLLER?	32
13. TIPURI DE MICROCONTROLERE.....	33
14. TABELE CU CARACTERISTICI ALE MICROCONTROLERELOR	37

15. MICROCONTROLERUL ALES.....	38
16. SCRIEREA UNUI CHARACTER SPECIAL	42
17. SCHEMA PENTRU CONVERSIA IN HEXAZECIMAL CU AFISARE PE LCD	43
18. CODURILE PENTRU CONVERSIE HEXA	44
19. TASTATURA.....	50
20. RELEUL	52
21. SCHEMA FINALA A TERMOSTATULUI DE CAMERA	54
22. CODURILE FINALE	56
23. SUPLIMENTAR	63
24. SCHEMA SUPLIMENTARA	64
25. COD SUPLIMENTAR	65
26. BIBLIOGRAFIE.....	73

1. TEMATICA PROIECTULUI

TERMOSTATUL DE CAMERĂ

Ce este un termostat de cameră?

Termostatul ambiental de cameră este un aparat care permite reglarea temperaturii furnizată de o centrală termică într-o încăpere.

De unde provine numele acestuia?

Numele dispozitivului vine din limba greacă unde "thermos" înseamnă încălzit, fierbinte, iar "statos" – a menține, a seta, a regla.

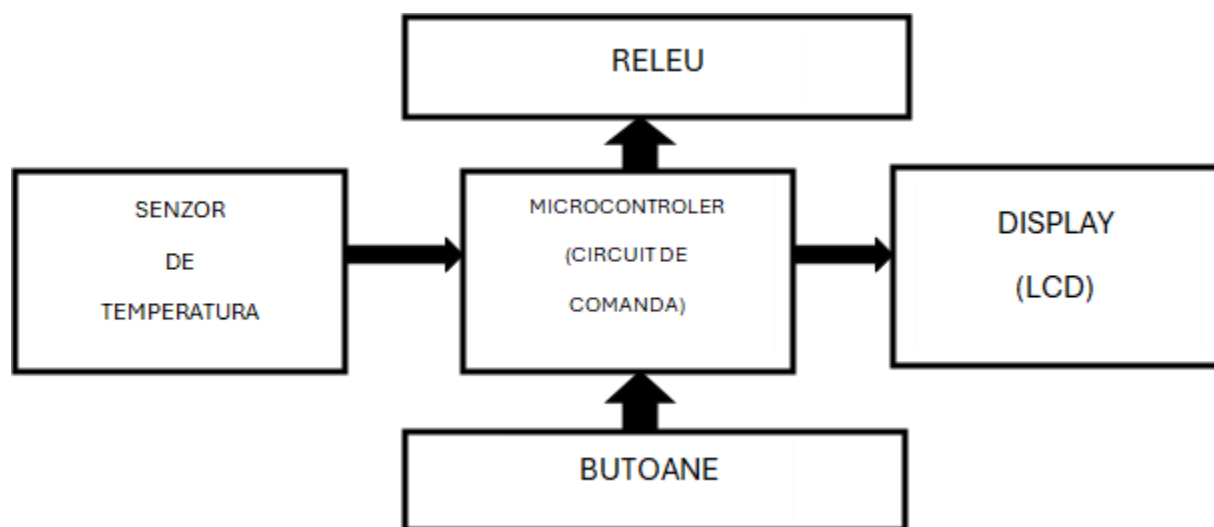
De câte feluri poate fi?

Termostatul ambiental poate fi mecanic sau electronic, cu afișaj digital, poate fi wireless sau cu fir, poate fi neprogramabil, programabil sau smart (recunoaște un tipar în programare și îl aplică).

Fiecare dintre aceste tipuri de termostat ambiental prezintă avantaje și dezavantaje, de exemplu, cel mecanic are un decalaj mai mare de temperatură, dar este mai ieftin, spre deosebire de cel electronic.

Termostatul ambiental cu fir se montează într-o cameră pilot, la o distanță de aproximativ 1,5 m de sol, departe de surse de căldură/frig sau curent.

2. SCHEMA BLOC



1.0. Schema bloc a unui termostat de camera

3. SENZORI ANALOGICI

3.1. Senzorul capacitiv

Acest tip de senzor măsoară modificările capacității unui condensator pentru a detecta prezența sau absența unui obiect sau pentru a măsura caracteristici, cum ar fi umiditatea sau nivelul de lichid.

Sunt utilizate în aplicații precum ecrane tactile și senzori de proximitate.

Senzorii capacitivi fac parte din grupa senzorilor parametrici și convertesc mărimea neelastică într-o variație de capacitate.

Prin definiție, capacitatea reprezintă raportul dintre cantitatea de sarcină electrică Q acumulată pe una dintre armăturile condensatorului și diferența de potențial U :

$$C = \frac{Q}{U} \quad 1.0$$



1.1. Senzorul capacitiv EL-XM24-308PM

3.2. Senzorul inductiv

Acești senzori detectează prezența sau absența unui obiect metalic prin măsurarea schimbării în inductanța unui circuit inductiv.

Sunt folosiți în aplicații de detectare a obiectelor în automate, sisteme de siguranță și în alte domenii.

Funcționarea senzorului inductiv se bazează pe variația inductanței unei bobine alimentate în curent alternativ.

Modificarea inductanței are loc datorită modificării circuitului magnetic prin deplasarea miezului bobinei sau a unei părți din miez.

Înfășurând N spire pe un miez magnetic, se obține o bobină a cărei inductanță este:

$$L = \frac{N^2}{R_m} ; [L]_{SI} = H \text{ (Henry)} \quad 1.1$$



1.2. Senzorul inductiv M12

3.3. Senzorul semiconductor

Acest tip de senzor utilizează proprietățile semiconductorilor pentru a măsura diverse caracteristici, cum ar fi temperatură, lumină, presiune sau gaz. Exemple includ senzorii de temperatură cu rezistori NTC sau PTC și senzorii de lumină cu fotodiodă sau fototranzistor.

Efectul Hall

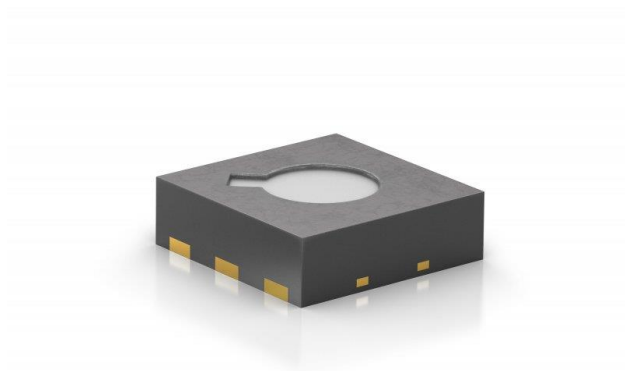
Un material, în general semiconductor, sub formă de plăcuțe, este parcurs de un curent I și supus unei inducții B , făcând un unghi θ cu curentul. Pe direcția perpendiculară pe inducție și curent va apărea o tensiune U_H de expresie:

$$U_H = k_H \cdot I \cdot B \cdot \sin\theta \quad 1.2$$

unde:

k_H – constanta Hall, ce depinde de material și de dimensiunile plăcuței;

θ – unghiul dintre inducție și curent.



1.3. Senzorul semiconductor din metal-oxid

3.4. Sensorul cu termocuplu

Un termocuplu este un senzor format din două fire din metale diferite care sunt sudate împreună la un capăt.

Prin măsurarea tensiunii generate la joncțiunea acestor metale în funcție de temperatura lor, termocuplele sunt utilizate pentru a măsura temperaturi într-o gamă largă de aplicații industriale și de laborator.

Aceasta este baza fizică a termocuplului:

$$\nabla V = S(T) \times \nabla T \quad 1.3$$

unde:

∇V este gradientul de tensiune;

∇T este gradientul de temperatură;

$S(T)$ este coeficientul Seebeck.



1.4. Senzorul de temperatura termocuplu GD1250-3

3.5. Senzorul rezistiv

Acest tip de senzor utilizează o schimbare în rezistența electrică pentru a măsura o anumită cantitate, cum ar fi temperatură, presiune sau umiditate.

Exemple includ termistorele, care își schimbă rezistența în funcție de temperatură, și senzorii de umiditate rezistivi.

Un senzor rezistiv este, din punct de vedere electric, un rezistor a cărui rezistență electrică este exprimată prin relația:

$$R = \rho \cdot \frac{l}{A} \quad 1.4$$

unde:

ρ – rezistivitatea materialului [$\Omega \cdot m$];

l – lungimea;

A – aria secțiunii transversale [m^2]

3.5.1. Senzorul rezistiv: Senzorul termorezistiv

Traductoarele termorezistive cuprind senzori a căror rezistență electrică depinde de temperatură.

Senzorii termorezistivi sunt realizați din metale pure și din materiale semiconductoare, bazându-se pe proprietățile materialelor conductoare și semiconductoare de a-și modifica rezistivitatea electrică la variația temperaturii.

Pentru metale, benzile de valență și de conducție se suprapun parțial, astfel că există întotdeauna electroni de conducție și metalul prezintă conductivitate.

În acest caz, rezistența senzorului depinde de temperatură.

Rezistivitatea metalelor crește odată cu creșterea temperaturii, adică prezintă un coeficient de temperatură pozitiv.

Pentru un interval restrâns de temperatură, coeficientul de temperatură α se consideră constant și rezistența traductorului este dată de relația:

$$R = R_0 \cdot (1 + \alpha \cdot \theta) \quad 1.5$$

unde:

R_0 – rezistența la 0°C;

α – coeficientul de temperatură;

θ – temperatura.

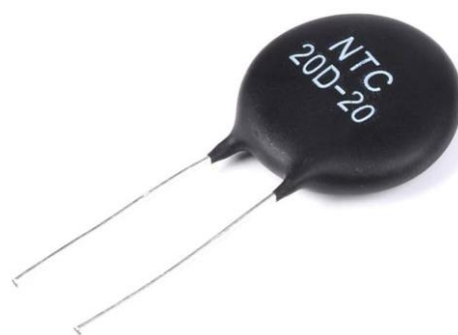
Termorezistențele sunt rezistențe executate din metale pure, care prezintă mari variații de rezistivitate cu temperatura, rezultând o caracteristică de conversie liniară, pe intervale largi de temperatură. Termorezistențele se utilizează la măsurarea temperaturii și în construcții speciale, la măsurarea vitezei gazelor, a debitului volumetric, a concentrației gazelor și a presiunilor scăzute.

Termistor PTC (Positive Temperature Coefficient): Contrar termistorilor NTC, termistorii PTC au o rezistență care crește odată cu creșterea temperaturii.

Sunt utilizați în aplicații precum protecția împotriva supraincălzirii în aparate electrocasnice și sisteme de încălzire. Termistorii PTC sunt de obicei instalați în serie cu un circuit și sunt folosiți pentru a proteja împotriva condițiilor de supracurent, cum ar fi resetarea siguranțelor.



1.5. Termistor PTC



1.6. Termistor NTC

Termistor NTC (Negative Temperature Coefficient): Acest tip de termistor are o rezistență care scade odată cu creșterea temperaturii.

Este folosit în aplicații precum sistemele de control al temperaturii, termometrele electronice și sistemele de monitorizare a temperaturii. Un NTC este utilizat în mod obișnuit ca senzor de temperatură sau în serie cu un circuit ca limitator de curent de intrare.

3.5.2. Senzorul rezistiv: Senzorul tensometric

Această categorie cuprinde senzorii rezistivi la care variația rezistenței electrice se produce prin variația lungimii conductorului, ca efect al alungirii sau al contracției. Dacă senzorul tensometric este fixat pe o porțiune dintr-o piesă care se deformează din cauza unei solicitări, el se va deforma la fel ca piesa.

După modul de realizare și de montare a senzorului rezistiv, se disting următoarele tipuri de traductoare:

- Traductoare tensometrice cu suport de hârtie
- Traductoare tensometrice simple
- Traductoare tensometrice rezistive cu folie
- Traductoare tensometrice rezistive cu semiconductor

Prin tratamente termice corespunzătoare, pentru traductoarele tensometrice metalice se poate modifica coeficientul de variație al rezistivității cu temperatura.



1.7. Senzorul tensometric NA1

3.5.3. Senzorul rezistiv: Senzorul potențiometric

Senzorii potențiometrici sunt constituiți dintr-o rezistență electrică dispusă pe un suport izolant, liniar sau circular, și un cursor a cărui poziție este determinată de mărimea neelectrică, introducându-se astfel în circuitul de măsurare o parte din rezistența senzorului, care este proporțională cu deplasarea cursorului.

Traductoarele rezistive de deplasare sunt constituite dintr-un senzor potențiometric, a cărui rezistență se modifică datorită unui cursor care se deplasează sub acțiunea mărimii de măsurat, deplasarea putând fi liniară sau circulară.

Deoarece traductorul potențiometric se execută prin bobinarea unui fir rezistiv pe un suport izolant, rezultă că variația rezistenței nu se produce în mod continuu, ci în trepte, corespunzător trecerii cursorului de pe o spirală pe alta.

Rezultă că valoarea rezistenței R este afectată de o eroare de discontinuitate. Valoarea sa minimă, care apare la sfârșitul cursei, se numește factor de treaptă.



1.8. Senzorul liniar potentiometric

3.5.4. Senzorul rezistiv: Senzorul cu contacte

Traductoarele rezistive cu contacte sunt traductoarele rezistive la care variația lungimii firului rezistiv se face în trepte, prin închiderea sau prin deschiderea unor contacte.

În acest scop, rezistența traductorului este divizată în mai multe porțiuni și prezintă posibilitatea închiderii sau deschiderii unor contacte de către mărimea mecanică de măsurat.

Sensibilitatea unui traductor rezistiv cu contacte se poate mări cu ajutorul unor transmisii cu pârghii. Limita sensibilității traductorului este determinată mai ales de distanța minimă dintre contacte, care este limitată de pericolul de străpungere și care depinde de tensiunea aplicată contactelor.

Senzorii rezistivi cu contacte sunt utilizați la realizarea traductoarelor destinate controlului dimensiunilor sau sortării pieselor pe intervale de valori.



1.9. Senzorul pentru contact usa

3.5.5. Senzorul rezistiv: Senzorul piezorezistiv

Efectul piezorezistiv constă în modificarea rezistivității unui material, dacă este supus unei presiuni exterioare, crescătoare din toate direcții.

Variația rezistenței cu presiunea se datorează deformării rețelei cristaline, produse de presiunea exterioară.

Cel mai utilizat material pentru senzori este manganinul, deoarece influența temperaturii este cea mai mică. Rezistența inițială este $R_0=100\Omega$.

Traductoarele piezorezistive sunt utilizate cu precădere pentru măsurarea presiunilor mari și foarte mari (peste 1000 atmosfere, ajungând până la 100.000 atmosfere).

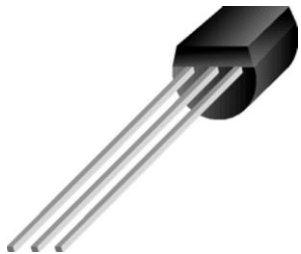


2.0. Senzorul de presiune electronica

4. SENZORI

I. Senzorul: LMT84LP

- masoara temperatura între -50 si 150 grade Celsius;
- tip interfata analogic;
- pret estimativ de 2 dolari;
- acuratete de 0.4 grade Celsius;
- montare through hole.



2.1. Senzorul LMT84LP



2.2. Senzorul AB13-130

II. Senzorul: AB13-130

- masoara temperatura între -30 si 85 grade Celsius;
- timp de raspuns de 0.1 s;
- consum redus de energie;
- pret estimativ de 2 dolari;
- acuratete de 0.5 grade Celsius;
- sensor specializat pentru termostat.

III. Senzorul: TMP36

- masoara temperatura intre -40 si 125 grade Celsius;
- acuratete de 0.5 grade Celsius;
- timp de raspuns de 0.33 s;
- consum redus de energie;
- pret estimativ de 2 dolari;
- tip interfata analogic.



2.3. Senzorul TMP36



2.4. Senzorul TC1046

IV. Senzorul: TC1046

- masoara temperatura intre -40 si 125 grade Celsius;
- acuratete de 2 grade Celsius;
- timp de raspuns de 1.5 s;
- consum redus de energie;
- pret estimativ de 2.2 dolari;
- tip interfata analogic.

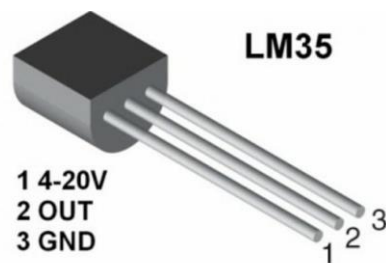
V. Senzorul: LM35

- masoara temperatura între -55 si 150 grade Celsius;
- tip interfata analogic;
- consum redus de energie;
- pret estimativ de 4.1 dolari;
- acuratete de 0.5 grade Celsius;
- potrivit pentru aplicatiile remote.

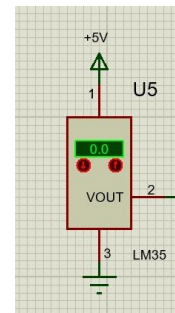
Astfel, putem spune pe scurt că senzorul **LM35** este un senzor analogic, însemnând că are ieșirea exprimată în tensiune.

Microcontrolerul știe să lucreze doar cu semnale digitale, deci va trebui creată legătura între ieșirea senzorului și intrarea microcontrolerului. Elementul care face legătura între senzor și microcontroler este convertorul analog-numeric, **ADC0808**, care face conversia semnalului analogic în semnal digital.

Deoarece convertorul ADC0808 are un domeniu al tensiunii de intrare diferit de cel dat de senzor, este nevoie de un circuit care să facă amplificarea semnalului, astfel încât acesta să funcționeze.



2.5. Senzorul LM35



5. TABEL DE COMPARAȚIE PENTRU SENZORI

Senzor	Producator	Cost [RON]	Temperatura °C °C	Rezolutii °C °C	Tens. Alimentare [V]
LMT84LP	Texas Instruments	7.13	-50÷150	±0.4%	1.5 (DC)
AB13-130	TOMIC	10.6	-30÷85	0,5%	250 (DC)
TMP36	ANALOG DEVICES	10.18	-20÷125	0,5%	1.8 (DC)
TC1046	MICROCHIP	11.12	-40÷125	±2%	2.7 - 4.4 (DC)
LM35	Texas Instruments	22,38	-55÷150	0,5%	4 - 20 (DC)

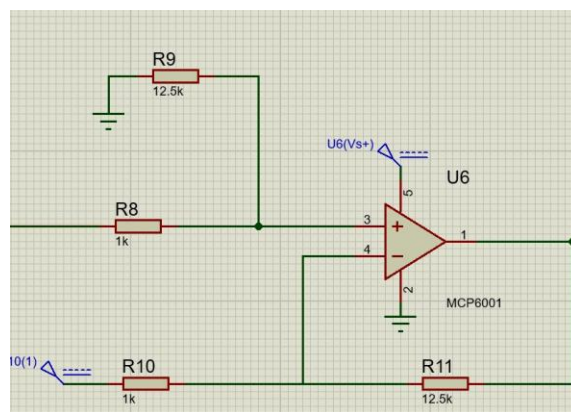
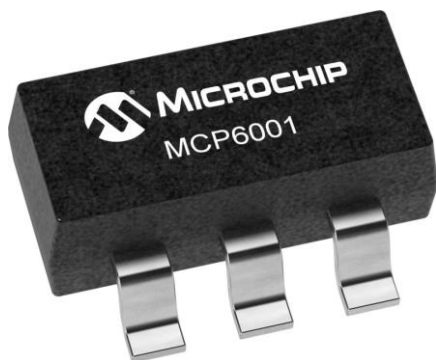
1.0. Tabelul 1 pentru senzori

6. CIRCUITUL DE ADAPTARE

Pentru adaptare (pentru a avea maxim 400 mV \rightarrow 40 °C la ieșirea senzorului), vom folosi un amplificator neinversor pentru a avea la ieșire 5V. Eu am ales să folosesc amplificatorul MCP6001.

Specificații:

- Temperatura de operare -65 și 150 °C;
- Tensiunea de alimentare $\pm 5V$;
- Slew-rate 2.3 V/ μs ;
- Preț scăzut (aproximativ 5 RON).



2.6. Amplificatorul MCP6001

Așadar, la modificarea temperaturii cu un $^{\circ}\text{C}$, tensiunea de ieșire se va modifica și ea cu 10 mV. Considerând domeniul de temperatură 8- 40°C , calculez domeniul tensiunii de ieșire:

$$\begin{array}{l}
 1^{\circ}\text{C} \dots \dots \dots 10\text{mV} \\
 8^{\circ}\text{C} \dots \dots \dots x \text{ mV} \Rightarrow x = \frac{8^{\circ}\text{C} \cdot 10\text{mV}}{1^{\circ}\text{C}} = 80 \text{ mV} \quad 1.6 \\
 1^{\circ}\text{C} \dots \dots \dots 10 \text{ mV} \\
 40^{\circ}\text{C} \dots \dots \dots y \text{ mV} \Rightarrow y = \frac{40^{\circ}\text{C} \cdot 10\text{mV}}{1^{\circ}\text{C}} = 400\text{mV} \quad 1.7
 \end{array}$$

Din aceste calcule, rezultă că domeniul de ieșire a senzorului este [80 mV; 400 mV]. Necesitatea acestui bloc de amplificare este reprezentată de faptul că este nevoie de adaptarea domeniului de ieșire al senzorului [80 mV; 400 mV] cu domeniul de intrare al CAN-ului [0.95V; 4.96V].

Vom continua prin calcularea restului elementelor din circuit:

$$\text{Amplificare: } A_V = \frac{5 \text{ V}}{400 \text{ mV}} = 12.5 = \frac{R_3}{R_4} \quad 1.8$$

$$V^+ = \frac{R_2}{R_1 + R_2} \cdot V_{\text{senzor}} \quad 1.9$$

$$V^- = \frac{R_4}{R_4 + R_3} \cdot V_{\text{out}} + \frac{R_3}{R_3 + R_4} \cdot V_{R_4} \quad 2.0$$

$$\text{Avem un amplificator operațional cu reacție negativă} \Rightarrow V^- = V^+ \quad 2.1$$

$$\text{Din (1.8), (2.1)} \Rightarrow \text{aleg } R_3 = R_2 = 15.5\text{k}\Omega \text{ si } R_1 = R_4 = 1\text{k}\Omega \quad 2.2$$

7. CONVERTORUL ANALOGIC-DIGITAL

Informațiile pe care le percepem din jurul nostru sunt analogice. Tensiunea de la ieșirea circuitului de amplificare generează un semnal cuprins între 0 V și 5 V (aproximativ), necesar funcționării corecte a ADC-ului. Deci, pentru ca informația să fie prelucrată de microcontroler, este nevoie de un convertor, care transformă semnalul analogic într-unul digital. Un astfel de dispozitiv îl reprezintă **ADC0808**, care prezintă în total 28 de pini dintre care 8 pentru 8 valori de biți la ieșire, 8 pini pentru intrări (IN0-IN7) și 3 pini pentru adrese.

$$V_{LSB} = \frac{V_{FS}}{2^8} = \frac{5V}{256} \cong 20mV \quad 2.3$$

Caracteristicile sale principale sunt:

- este ușor de interfațat cu toate microprocesoarele;
- rezoluție de 8 biți (256 nivele);
- este un convertor care folosește tehnica aproximărilor succesive;
- alimentare unipolară de 5V;
- prezintă un consum mic de putere: 15 mW;
- timpul de conversie este de 100 μs;
- frecvența semnalului de ceas este de 640 kHz.

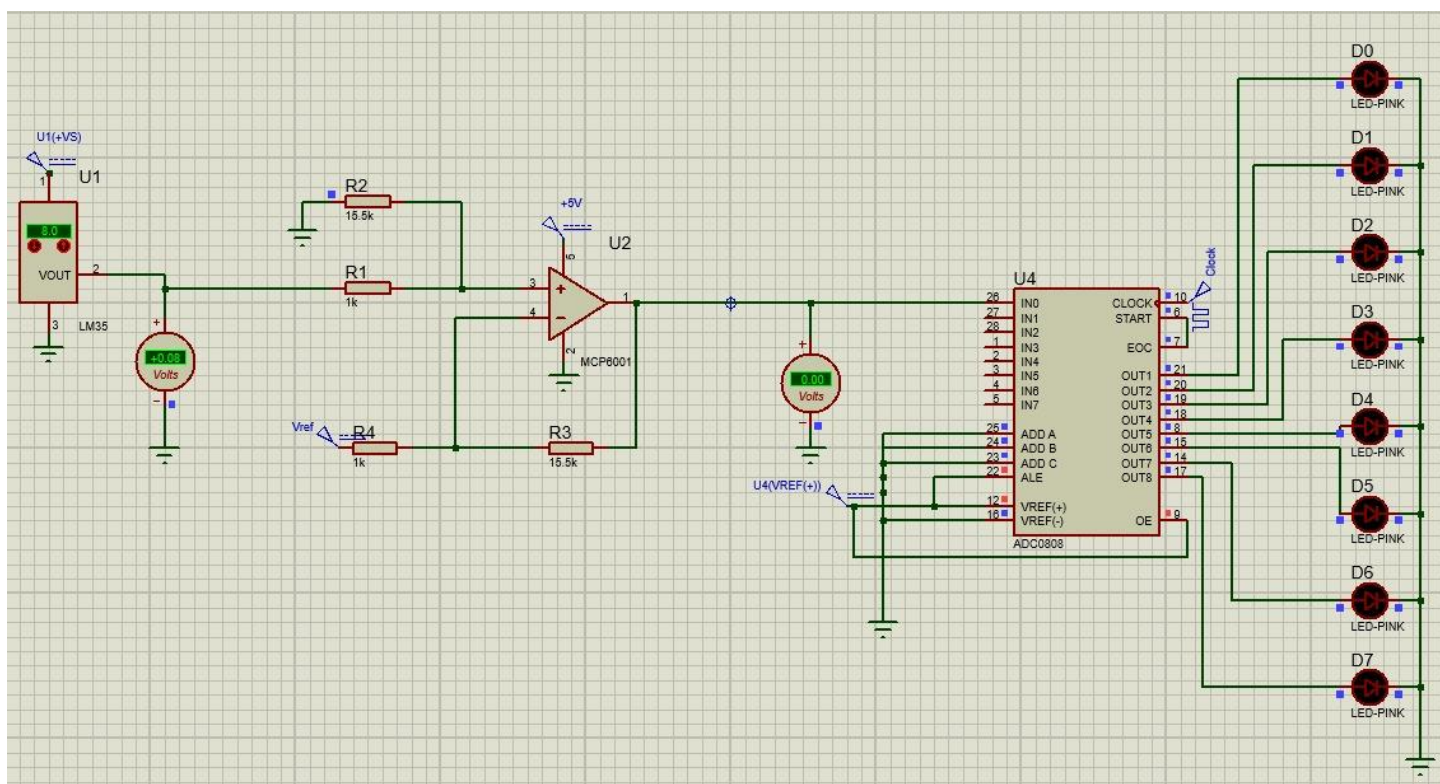
Semnalele analogice pot fi aplicate la pinii IN0 - IN7 ai circuitului. Pentru a selecta canalul de intrare se folosesc liniile de adresă – ADD A, ADD B, ADD C. În circuitul realizat pentru aplicația termostatlui de cameră am nevoie de o singură intrare pe care să vină semnalul analogic de la senzorul de temperatură. Am ales intrarea IN0 și urmărind în tabel valorile corespunzătoare pentru intrarea IN0 se obține: ADD A – low, ADD B – low, ADD C – low.

Configurarea pinilor:

<i>Pin</i>	<i>Denumire</i>	<i>Ce este?</i>
26	IN0	Intrările pentru selectarea canalului analogic. Aceștia determină canalul analogic care va fi convertit în modulul ADC.
27	IN1	
28	IN2	
1	IN3	
2	IN4	
3	IN5	
4	IN6	
5	IN7	
25	ADD A	Acești trei pini sunt utilizați pentru a selecta canalul analogic și alte opțiuni de configurare.
24	ADD B	
23	ADD C	
22	ALE (Adress Latch Enable)	Semnal de control care indică începutul procesului de selecție a canalului analogic (împreună cu ADD A, B și C se va forma adresa completă).
12	VREF (+)	Intrarea pentru tensiunea de referință pozitivă.
16	VREF (-)	Intrarea pentru tensiunea de referință negativă.
10	CLOCK	Semnalul de ceas care sincronizează operațiile.
6	START	Utilizat pentru a iniția procesul de conversie analog-digital.
7	EOC (End Of Conversion)	Indică faptul că procesul de conversie analog-digital s-a încheiat și că datele convertite sunt disponibile pentru citire.
21	OUT1	Iesirile digitale care furnizează datele convertite în format digital. Fiecare dintre aceste iesiri reprezintă un bit din rezultatul conversiei analog-digital.
20	OUT2	
19	OUT3	
18	OUT4	
8	OUT5	
15	OUT6	
14	OUT7	
17	OUT8	
9	OE (Output Enable)	Semnal de control utilizat pentru a activa/dezactiva iesirile.

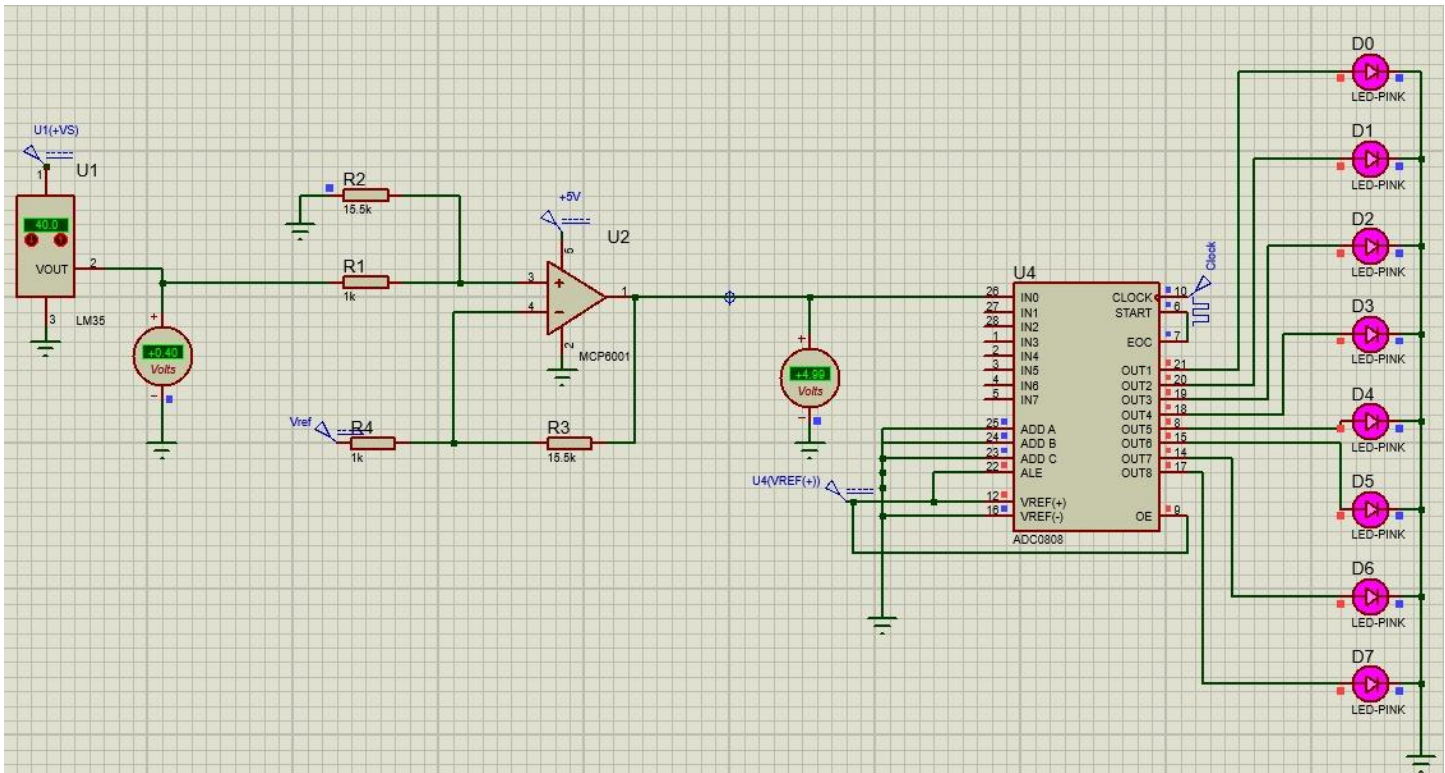
8. SCHEMELE IN PROTEUS PENTRU 8°C SI 40°C

- Pentru 8°C



2.7. Schema in Proteus pentru 8°C - la iesirea ADC-ului 00h

- Pentru 40°C



2.8. Schema in Proteus pentru 40°C - la iesirea ADC-ului FFh

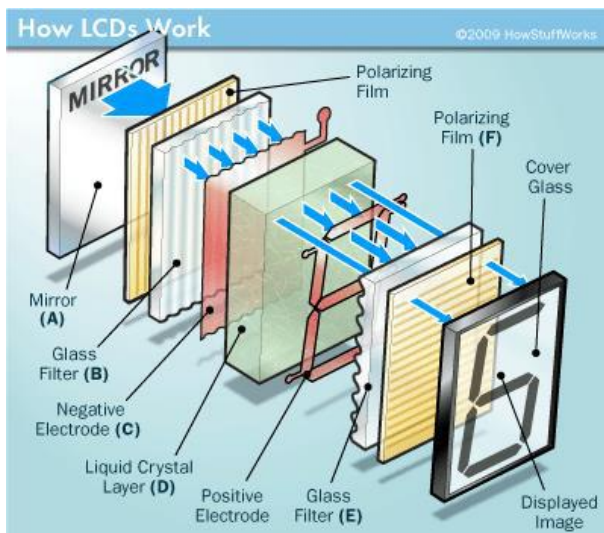
9. CONSIDERATII TEORETICE: LCD

Afișajele cu cristale lichide (LCD) sunt dispozitive de afișare pentru litere, cifre și imagini, fiind construit dintr-o matrice de celule lichide care devin opace când apare un curent sau câmp electric. Un afișaj LCD este format dintr-un ecran care este comandat de un microcontroler.

Panoul LCD nu produce lumină proprie, așa că ele necesită lumină exterioară pentru a produce o imagine vizibilă, sursa fiind în spatele panoului.

Un afișaj LCD este format din mai multe straturi interne care lucrează împreună pentru a produce imaginea afișată. Aceste straturi sunt:

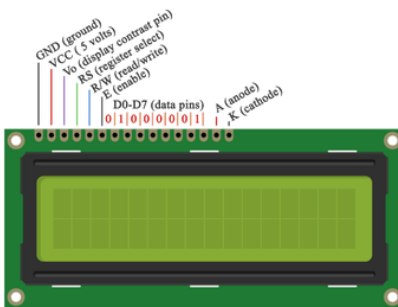
1. Substratul superior de sticlă: acesta este stratul superior al afișajului și este transparent pentru a permite trecerea luminii prin el.
2. Stratul de polarizare: acest strat este aplicat pe substratul superior de sticlă și ajută la direcționarea luminii într-o singură direcție.
3. Stratul de cristale lichide: acest strat este situat între cele două substraturi de sticlă și conține cristale lichide care se pot alinia sub influența unui câmp electric.
4. Electrozii de comandă: acești electrozi sunt amplasați pe ambele părți ale stratului de cristale lichide și sunt utilizați pentru a aplica un câmp electric la cristalele lichide.
5. Substratul inferior de sticlă: acesta este stratul inferior al afișajului și are rolul de a proteja restul componentelor interne ale afișajului.
6. Inca un strat aplicat pe substratul inferior de sticlă și ajută la îndrumarea luminii spre ochiul observatorului.



2.9. Cum lucrează un LCD

În plus față de aceste straturi principale, afișajele LCD pot conține și alte componente precum filtre de culoare, iluminare cu LED-uri, circuite electronice de control și altele.

Toate aceste componente lucrează împreună pentru a produce afișajul final.



3.0. Pinii unui LCD

10. LCD-UL ALES

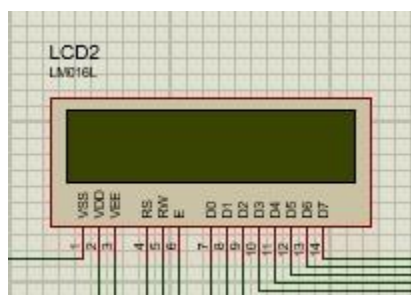
LCD-ul ales de mine pentru acest proiect este LM016L. Acesta are un display de 16x2, însemnând că informația va fi afișată pe 2 linii a câte 16 caractere fiecare linie.

Alimentarea se face la 5V și permite scrierea caracterelor alfanumerice principale.

CONFIGURATIA PINILOR

- *VDD* este alimentarea;
- *VSS/ VEE* este masa;
- *RS* (Register Select) este folosit pentru selectarea regiștrilor, în momentul în care este “low” se face introducerea de instrucțiuni, iar când este “high” se face introducerea de date;
- *R/W* (Read/Write) este pentru a selecta tipul operațiunii, adică de scriere pentru valoarea “low” și citire pentru valoarea “high”;
- *E* (Enable) permite scrierea sau citirea datelor, este activă pe front negativ;
- *DB₀-DB₇* reprezintă busurile de date, *DB₀-DB₃* este partea LOW, iar *DB₄-DB₇* partea HIGH.

Pentru a putea utiliza afișajul este nevoie să se introducă anumite comenzi.



3.1. LCD-ul LM016L

Am folosit acest tabel ASCII pentru scrierea pe LCD fără a fi nevoie să folosesc microcontrolerul:

<div> <div> <div> <div> <div>b₇</div> <div>b₆</div> <div>b₅</div> </div> <div> <div>b₄</div> <div>b₃</div> <div>b₂</div> <div>b₁</div> </div> </div> <div> <div>Bits</div> <div>↓</div> </div> </div> <div> <div>Column</div> <div>→</div> </div> <div> <div>Row</div> <div>↓</div> </div> </div>					0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	0	1	SOH	DC1	!	1	A	Q	a	q
0	0	0	1	0	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	\$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(8	H	X	h	x
1	0	0	1	9	HT	EM)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	
1	1	0	1	13	CR	GS	—	=	M]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	—	o	DEL

3.2. Tabelul ASCII pentru LCD

D7	D6	D5	D4	
0	0	1	1	EN
0	0	1	1	EN
0	0	1	1	EN
0	0	1	0	EN
0	0	1	0	EN
0	0	0	1	EN
0	0	0	0	EN
1	1	1	1	EN

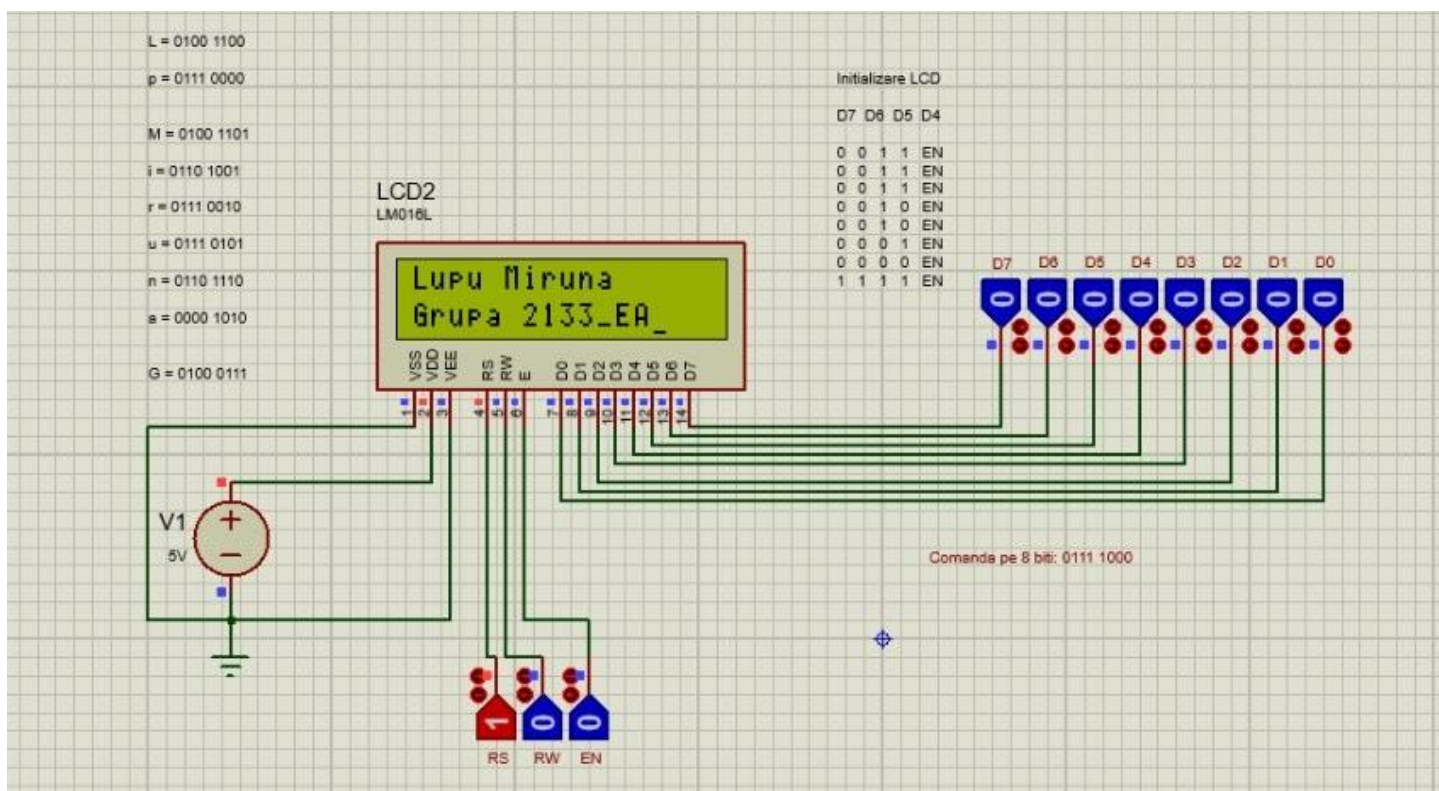
3.3. Comenzile de pornire ale LCD-ului

INTERNAL PIN CONNECTION

Pin No.	Symbol	Level	Function
1	V _{SS}	—	0V
2	V _{DD}	—	+5V
3	V _O	—	—
4	RS	H/L	L: Instruction code input H: Data input
5	R/W	H/L	H: Data read (LCD module→MPU) L: Data write (LCD module←MPU)
6	E	H, H→L	Enable signal
7	DB0	H/L	Data bus line Note (1), Note (2)
8	DB1	H/L	
9	DB2	H/L	
10	DB3	H/L	
11	DB4	H/L	
12	DB5	H/L	
13	DB6	H/L	
14	DB7	H/L	

3.4. Pinii LCD-ului ales

11. CIRCUITUL IN PROTEUS PENTRU SCRIEREA NUMELUI SI A GRUPEI PE UN LCD



3.5. Schema in Proteus pentru scrierea numelui si a grupei pe un LCD fara microcontroler

Comanda pentru prima linie : 80h

Comanda pentru a doua linie: C0h

12. CE ESTE UN MICROCONTROLLER?

Un controler este o structura electronica destinata controlului unui proces sau, mai general, a unei interactiuni caracteristice cu mediul exterior, fara sa fie necesara interventia operatorului uman.

O definitie, cu un sens foarte larg de cuprindere, ar fi aceea ca un microcontroller este un microcircuit care incorporeaza o unitate centrala (CPU) si o memorie. Impreuna cu resurse care-i permit interactiunea cu mediul exterior. Resursele integrate la nivelul microcircuitului ar trebui sa includa, cel putin, urmatoarele componente:

- unitatea centrala (CPU), cu un oscilator intern pentru ceasul de sistem;
- o memorie locala tip ROM/PROM/EPROM/FLASH si eventual una de tip RAM;
- un sistem de Intreruperi;
- I/O - intrari/iesiri numerice (de tip port paralel);
- port serial de tip asincron si/sau sincron, programabil;
- sistem de timere-temporizatoare/numaratoare programabile;

Printre multele domenii unde utilizarea lor este practic un standard industrial se pot menționa: în industria de automobile (controlul aprinderii/motorului, climatizare, diagnoză, sisteme de alarmă, etc.), în așa zisa electronică de consum (sisteme audio, televizoare, camere video și videocasetofoane, telefonie mobilă, GPS-uri, jocuri electronice etc.), în aparatura electrocasnică (mașini de spălat, frigidere, cuptoare cu microunde, aspiratoare), în controlul mediului și climatizare (sere, locuințe, hale industriale), în industria aerospațială, în mijloacele moderne de măsurare - instrumentație (aparate de măsură, senzori și traductoare inteligente), la realizarea de periferice pentru calculatoare, în medicină.

Intrările analogice vehiculează informații exprimabile prin funcții continue de timp. "Citirea" acestora de către microcontroller presupune prezența unor circuite capabile să prelucreze aceste informații, fie comparatoare analogice, fie convertoare analog-numerice, ale căror ieșiri sunt citite de către MC.



3.6. Microcontroller

13. TIPURI DE MICROCONTROLERE

Pe baza unui nucleu comun au fost definite familiile de microcontrolere; nucleul este constituit dintr-o unitate centrală, aceeași pentru toți membrii unei familii, și o serie de interfețe și periferice. Din punct de vedere al programatorului, toți membrii unei familii folosesc același set de instrucțiuni, permit aceleași moduri de adresare și folosesc aceleași registre. Diferența între membrii unei familii constă în primul rând în echiparea chip-ului cu memorie (tip de memorie și capacitatea memoriei). Alte diferențe pot fi găsite la frecvența de clock pentru unitatea centrală sau în interfețele on-chip și perifericele on-chip suplimentare față de cel mai simplu reprezentant al familiei. O diferență între membrii unei familii poate fi și modul în care sunt conectate semnalele la pin – respectiv tipul capsulei de prezentare a circuitului integrat. În cadrul acestui capitol, în continuare, sunt prezentate câteva familii de microcontroller-e cu sublinierea însușirilor caracteristice și considerând numele producătorului ca fiind unul din elementele reprezentative pentru o familie.

I. 8048 (Intel MCS-48)

A fost primul MC apărut pe piață, având o structură Harvard modificată, cu 64-256 octeți de RAM și este încă folosit în multe aplicații datorită prețului scăzut.



3.7. Microcontroler-ul 8048

II. 8051 (Intel MCS-51)

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. Arhitectura sa are spații de memorie separate pentru program și date. Poate adresa 64KBytes memorie de program, din care primii 4(8..32)KBytes locali (ROM). Poate adresa 64KBytes memorie de date externă, adresabilă doar indirect. Are 128 (256) octeți de RAM local, plus un număr de registre speciale pentru lucrul cu periferia locală. Are facilități de prelucrare la nivel de bit (un procesor boolean, adresare pe bit). Intel a dezvoltat și un “super 8051” numit generic 80151. Actualmente există zeci de variante produse de diverși fabricanți (Philips, Infineon, Atmel, Dallas, Temic, etc.) precum și cantități impresionante de soft comercial sau din categoria freeware/shareware.

Au apărut și dezvoltări ale acestei familii în sensul trecerii la o arhitectură similară (în mare), dar pe organizată pe 16 biți, cu performanțe îmbunătățite ca viteză de prelucrare: familia XA51 eXtended Arhitecture de la Philips și familia 80C251 (Intel). Din păcate aceste noi variante nu s-au bucurat nici pe departe de succesul „bătrânului” 8051.

Dispozitivele I/O au un spațiu propriu de adresare. 8051 dispune de un procesor boolean prin care se pot executa operații complexe la nivel de bit, iar în funcție de rezultate se pot face salturi. Pentru 8051 există foarte mult software, atât contra cost cât și gratuit.



3.8. Microcontroler-ul 8051

III. 80C186, 80C188 (Intel, AMD)

Derivate din clasicele 8086/88 prin includerea pe același microcircuit a 2 canale DMA, 2 numărătoare/timere, un sistem de întreruperi și un controler pentru DRAM. Marele avantaj al acestor cvasi(aproape) microcontrolere (ele nu au memorie integrată!) este legat de utilizarea ca mediu de dezvoltare a unor platforme de calcul tip IBM-PC, compatibile 80x86, cu tot softul aferent.



3.9. Microcontroler-ul 80C186

IV. **68HC05 (Freescale)**

Un microcontroler de 8 biți derivat din microprocesorul M6800 și care prezintă multe asemănări cu un alt microprocesor răspândit, la timpul său, 6502. Are un spațiu de memorie unic (64Kbytes) în care sunt plasate și registrele perifericelor (I/O, timere) cu un indicator de stivă (SP) hard pe 5biți (stivă de maxim 32 octeți !). Există variante cu memorie EEPROM, CAN, port serial, etc. Este unul din cele mai răspândite microcontrolere (comparabil cu 8051). Varianta evoluată a acestei familii este seria 68HC08 bazată pe o nouă unitate centrală de 8 biți numită CPU08, cu cea mai recentă dezvoltare sub forma seriei 68HCS08 destinată în mod special unor aplicații din industria automobilului.



4.0. Microcontroler-ul 68HC05

V. **Motorola 68HC11**

E preluat și de TOSHIBA si este un MC popular pe 8 biți de date și 16 biți de adresă, cu o arhitectură ca și 6805. 68HC11 are inclusă memorie EEPROM sau OTP, linii digitale I/O, numărătoare/temporizatoare, convertoare A/D, generatoare PWM, acumulator de impulsuri, canale seriale de comunicații sincrone și asincrone etc.



4.1. Microcontroler-ul 68HC11

14. TABELE CU CARACTERISTICI ALE MICROCONTROLERELOR

TABELE CU CARACTERISTICI A 5 MICROCONTROLERE DIN FAMILIA 8051

Micro-controler	Memorie Flash	Frecventa de ceas [MHz]	Interfata	Nr. I/O	Timere	Temp. [°C]	Alimen. [V]	Pret [USD]
AT89S52	8k x 8bit	24	UART	32	2	-55...125	4-5.5	1.50
AT89C2051	2k x 8bit	24	UART	15	2	-55...125	2.7-6	2.88
AT89C4051	4k x 8bit	24	UART	15	2	-55...125	2.7-6	1.17
AT80C51RD2	8k x 8bit	40	UART	32	3	-40...85	2.7-5.5	2.60
AT89C51RC23	32k x 8bit	40	UART, SPI	32	3	-40...85	2.7-5.5	11.30

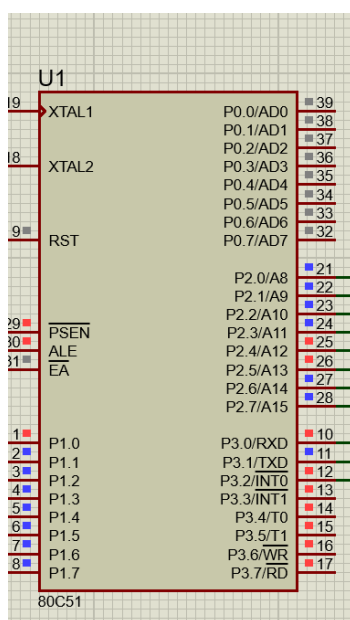
1.1. Tabelul 2

TABEL CU CARACTERISTICI A 5 MICROCONTROLERE

Micro-controler	Memorie	Frecventa de ceas [MHz]	Nr. I/O	Timere	Temp. [°C]	Alimen. [V]	Pret [USD]
8048	ROM	1	24	-	-40...85	2.5-5.0	7.95
68HC11	RAM	2/4/8	32	2	-40...85	4.5-5.5	4.99
80C186	ROM/RAM	8/10/12	45	min. 1	0...70	4.5-5.5	9.98
8051	ROM/RAM	12/24	32	2	-40...85	3-5.5	1.80
68HC05	ROM/RAM	1/2	8-28	-	-40...85	2.7-5.5	10.36

1.2. Tabelul 3

15. MICROCONTROLERUL ALES



4.2. Microcontrolerul 80C51

Microcontrolerul ales de mine este 8051. L-am ales deoarece poate adresa 64KBytes memorie de date externă, adresabilă doar indirect. Are 128 (256) octeți de RAM local, plus un număr de registre speciale pentru lucrul cu periferia locală. Are facilități de prelucrare la nivel de bit (un procesor boolean, adresare pe bit). Intel a dezvoltat și un “super 8051” numit generic 80151.

La mometul actual există zeci de variante produse de diverși fabricanți (Philips, Infineon, Atmel, Dallas, Temic, etc.) precum și cantități impresionante de soft comercial sau din categoria freeware/shareware.

8051 dispune de un procesor boolean prin care se pot executa operații complexe la nivel de bit, iar în funcție de rezultate se pot face salturi. Pentru 8051 există foarte mult software, atât contra cost cât și gratuit.

Configurația pinilor

Pinul 1 până la Pinul 8 (Portul 1) - Pinii 1 până la 8 sunt asigurați Portului 1 pentru operațiuni simple de intrare/ieșire (I/O). Ei pot fi configurați ca pini de intrare sau ieșire în funcție de controlul logic, adică dacă este aplicată logica zero (0) la portul I/O, acesta va acționa ca un pin de ieșire, iar dacă este aplicat logic unu (1), pinul va acționa ca un pin de intrare.

Pinul 9 (RST) - Pinul de resetare. Este un pin de intrare activ la nivel înalt. Prin urmare, dacă pinul RST este la nivel înalt pentru cel puțin 2 cicluri mașină, microcontrolerul va fi resetat, adică va închide și va termina toate activitățile.

Pinul 10 până la Pinul 17 (Portul 3) - Pinul 10 până la pinul 17 sunt pini ai portului 3, care sunt de asemenea denumiți P3.0 până la P3.7. Acești pini sunt similari cu cei ai portului 1 și pot fi utilizați ca pini de intrare sau ieșire universali. Acești pini sunt bidirecționali și au unele funcții suplimentare care sunt următoarele:

- **P3.0 (RXD):** Al 10-lea pin este RXD (pinul de recepție serială de date) care este pentru intrare serială. Prin acest semnal de intrare, microcontrolorul primește date pentru comunicare serială.
- **P3.1 (TXD):** Al 11-lea pin este TXD (pinul de transmitere serială de date) care este pinul de ieșire serială. Prin acest semnal de ieșire, microcontrolorul transmite date pentru comunicare serială.
- **P3.2 și P3.3 (INT0, INT1):** Al 12-lea și al 13-lea pin sunt pentru Întreruperea Hardware Externă 0 și, respectiv, Întreruperea 1. Când această întrerupere este activată (adică când este la nivel scăzut), 8051 este întrerupt în ceea ce face și sare la valoarea vectorului de întrerupere (0003H pentru INT0 și 0013H pentru INT1) și începe să execute Rutina de Servire a Întreruperii (ISR) de la acea locație vectorială.
- **P3.4 și P3.5 (T0 și T1):** Al 14-lea și al 15-lea pin sunt pentru Timerul 0 și Timerul 1 intrare externă. Acestea pot fi conectate cu un timer/contor pe 16 biți.
- **P3.6 (WR'):** Al 16-lea pin este pentru scrierea la memorie externă
- **P3.7 (RD'):** Al 17-lea pin este pentru citirea din memoria externă

Pinul 18 și Pinul 19 (XTAL2 și XTAL1) - Acești pini sunt conectați la un oscilator extern, care este în general un oscilator cu cristal de cuarț. Aceștia sunt folosiți pentru a furniza o frecvență de ceas externă de la 4MHz la 30MHz.

Pinul 21 până la Pinul 28 (Portul 2) - Pinul 21 până la pinul 28 sunt pini ai portului 2, de asemenea denumiți P2.0 până la P2.7. Atunci când este interfațată memorie externă suplimentară cu microcontrolerul 8051, pinii portului 2 acționează ca octeți de adresă de ordin superior. Acești pini sunt bidirecționali.

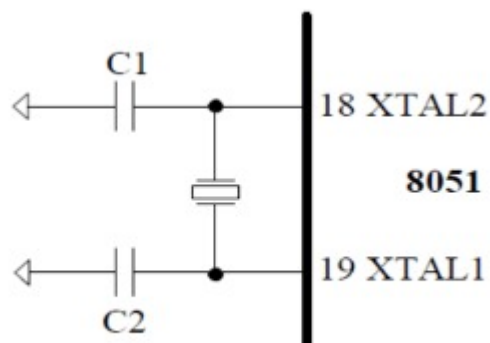
Pinul 29 (PSEN) - PSEN înseamnă Program Store Enable. Este un pin de ieșire, activ la nivel scăzut. Acesta este folosit pentru a citi memoria externă.

Pinul 30 (ALE/PROG) - ALE înseamnă Address Latch Enable. Este un pin de intrare, activ la nivel înalt. Acest pin este folosit pentru a distinge între cipuri de memorie atunci când sunt utilizate mai multe cipuri de memorie. De asemenea, este folosit pentru a demultiplexa semnalele de adresă și date disponibile la portul 0. În timpul programării flash, adică programarea EPROM-ului, acest pin acționează ca intrare de impuls de programare (PROG).

Pinul 31 (EA/VPP) - EA înseamnă intrare de acces extern. Este folosit pentru a activa/dezactiva interfațarea cu memoria externă.

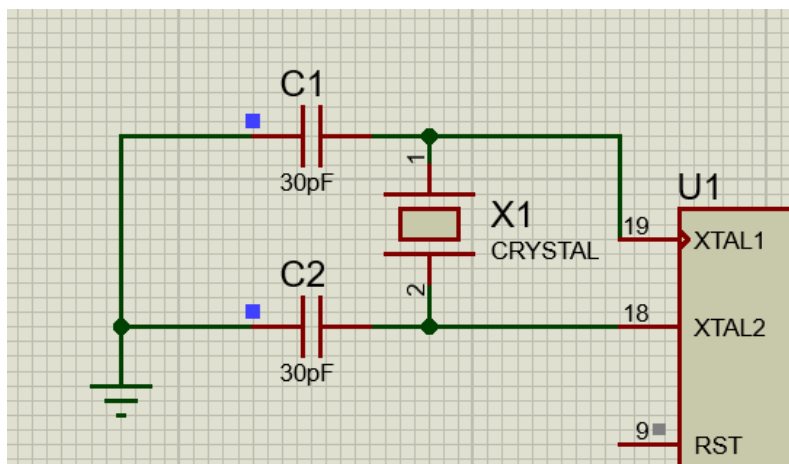
Pinul 32 până la Pinul 39 (Portul 0) - Pinul 32 până la pinul 39 sunt pini ai portului 0, de asemenea denumiți P0.0 până la P0.7. Aceștia sunt pini de intrare/ieșire bidirecționali. Ei nu au niciun pull-up intern. Prin urmare, sunt folosiți rezistoare de pull-up de 10 K Ω ca pull-up-uri externe. Portul 0 este de asemenea desemnat ca AD0-AD7 deoarece 8051 multiplexează adresa și datele prin portul 0 pentru a salva pini.

Oscilatorul cu cuarț



4.3. Oscilatorul cu cuarț

XTAL2 și XTAL1 - Acești pini sunt conectați la un oscilator extern, care este în general un oscilator cu cristal de cuarț. Aceștia sunt folosiți pentru a furniza o frecvență de ceas externă de la 4MHz la 30MHz. **RST**: Acest pin este folosit pentru a reseta microcontrolorul. Un impuls la nivel înalt pe acest pin resetează microcontrolorul la starea sa inițială, de aceea am folosit o rezistență de 10k pentru acest lucru.

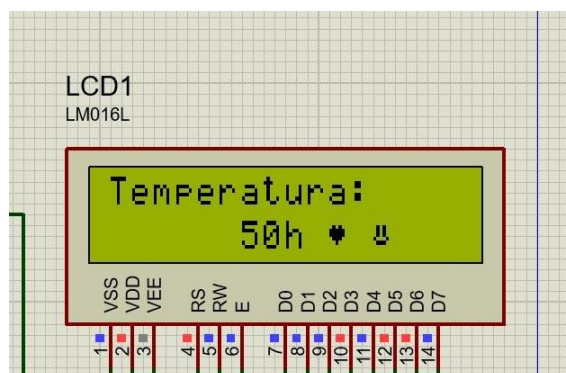


4.4. Pinii XTAL1 si XTAL2

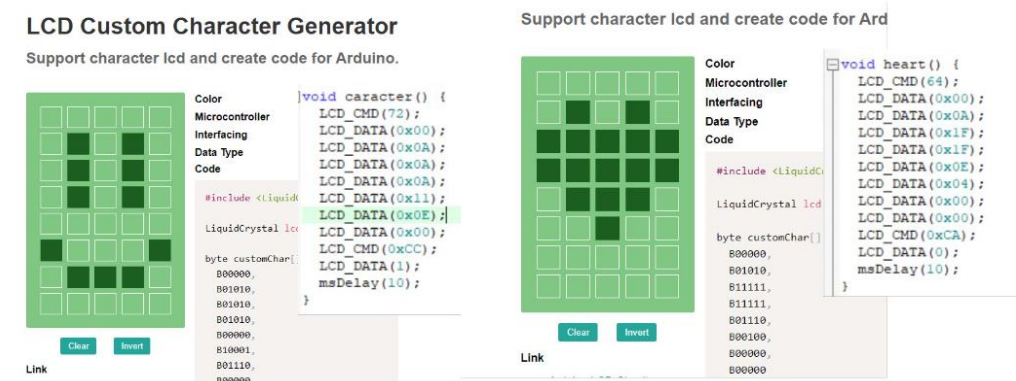
16. SCRIEREA UNUI CHARACTER SPECIAL

Pentru scrierea unui caracter special am luat informații de pe două site-uri:

- <https://www.engineersgarage.com/how-to-create-custom-characters-on-16x2-lcd-using-8051-microcontroller-at89c51-part-10-45/> pentru integrarea lui în codul meu;
- <https://maxpromer.github.io/LCD-Character-Creator/> pentru a crea un caracter special în Arduino.



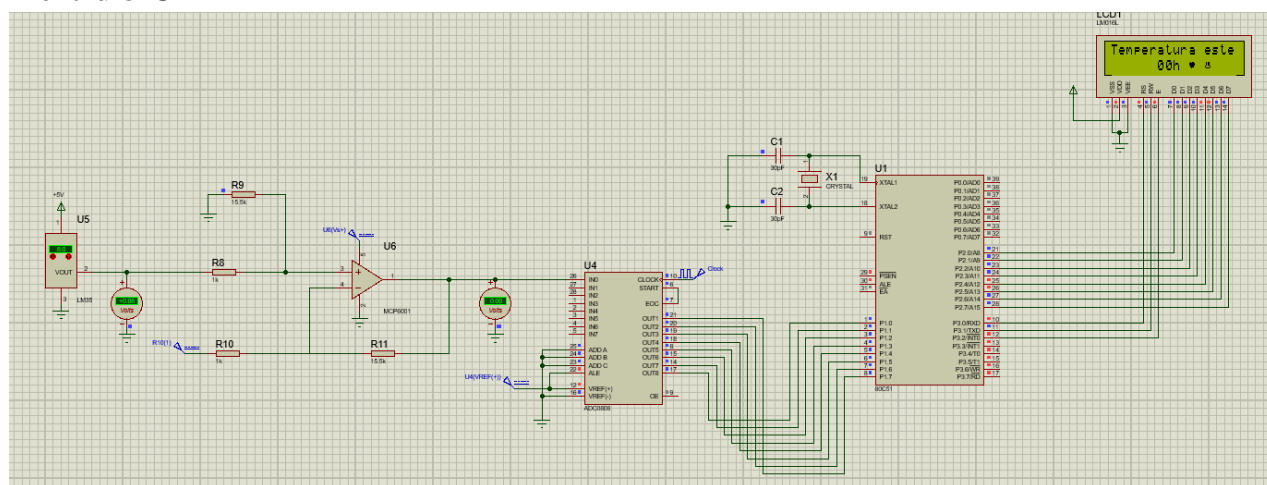
4.5. Afisarea pe LCD a valorii de pe senzor convertita in hexa



4.6. Codurile pentru scrierea caracterelor speciale

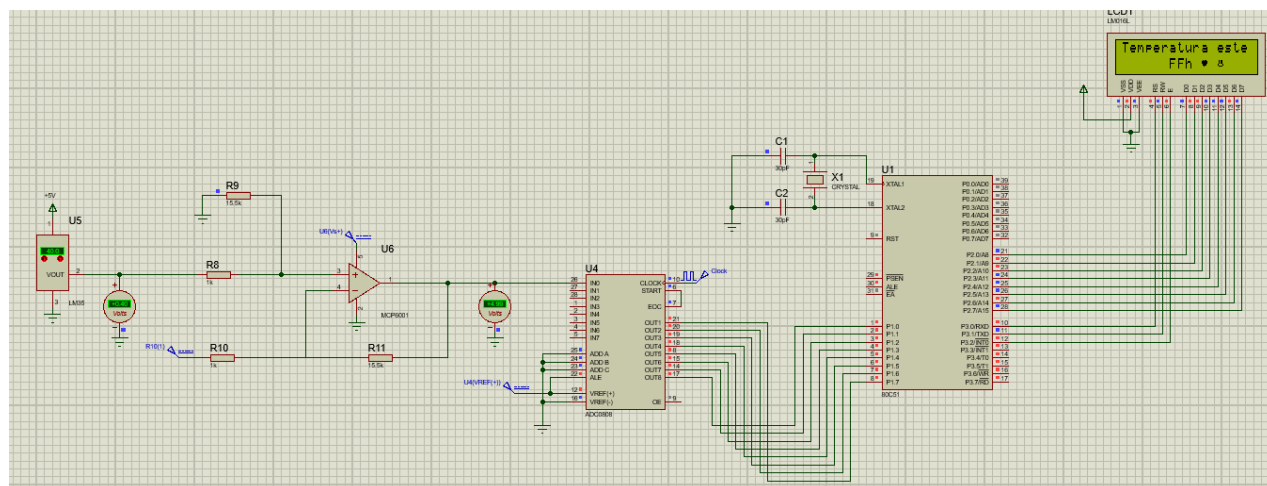
17. SCHEMA PENTRU CONVERSIA IN HEXAZECIMAL CU AFISARE PE LCD

- Pentru 8°C



4.7. Schema in Proteus pentru 8°C

- Pentru 40°C



4.8. Schema in Proteus pentru 40°C

18. CODURILE PENTRU CONVERSIE HEXA

- COD IN C:

<pre>#include <REG51.H> sfr LCD = 0xA0; sbit RS = P3^0; sbit RW = P3^1; sbit EN = P3^2; unsigned char adc_value = 0; // Variabila pentru valoarea ADC void LCD_CMD(unsigned char x); void LCD_DATA(unsigned char w); void LCD_INI(void); void Send_Data(unsigned char *Str); void msDelay(unsigned int); void ADC(); void timer1_delay_1ms(unsigned int delay); void display_adc_value_hex(unsigned char value); void heart(); void caracter();</pre> <p style="text-align: right;">---></p>	<pre>void main(void) { msDelay(200); // 0.2 secunde LCD_INI(); // Initializarea LCD LCD_CMD(0x80); Send_Data("Temperatura este:"); // Scrie mesaj pe LCD heart(); // Initialize the heart caracter caracter(); // Initialize the custom caracter while (1) { ADC(); display_adc_value_hex(adc_value); } } void ADC() { adc_value = P1; // Simulare citire valoare de la un ADC }</pre> <p style="text-align: right;">---></p>	<pre>void display_adc_value_hex(unsigned char value) { unsigned char hex_chars[3]; unsigned char high_nibble, low_nibble; // Convert value to hex string high_nibble = (value >> 4) & 0x0F; // Obținerea celor 4 biti superiori low_nibble = value & 0x0F; // Obținerea celor 4 biti inferiori if (high_nibble < 10) { hex_chars[0] = high_nibble + '0'; // Conversie in caracter pentru cifre 0- 9 } else { hex_chars[0] = high_nibble - 10 + 'A'; // Conversie in caracter pentru litere A-F } }</pre> <p style="text-align: right;">---></p>
--	--	---

<pre> if (low_nibble < 10) { hex_chars[1] = low_nibble + '0'; // Conversie in caracter pentru cifre 0-9 } else { hex_chars[1] = low_nibble - 10 + 'A'; // Conversie in caracter pentru litere A-F } hex_chars[2] = 'h'; // 'h' to indicate hex LCD_CMD(0xC6); // Pozitionarea temperaturii pe linia a doua // Display hex value on LCD LCD_DATA(hex_chars[0]); LCD_DATA(hex_chars[1]); LCD_DATA(hex_chars[2]); } void heart() { LCD_CMD(64); // Set CGRAM address for the heart character LCD_DATA(0x00); LCD_DATA(0x0A); LCD_DATA(0x1F); LCD_DATA(0x1F); LCD_DATA(0x0E); LCD_DATA(0x04); LCD_DATA(0x00); LCD_DATA(0x00); LCD_DATA(0x00); ---> </pre>	<pre> LCD_CMD(0xCA); // Set cursor position for heart character (C8h) LCD_DATA(0); // Display heart character msDelay(10); } void caracter(){ LCD_CMD(72); // Set CGRAM address for the custom character LCD_DATA(0x00); LCD_DATA(0x0A); LCD_DATA(0x0A); LCD_DATA(0x0A); LCD_DATA(0x11); LCD_DATA(0x0E); LCD_DATA(0x00); LCD_DATA(0x00); LCD_CMD(0xCC); // Set cursor position for custom character (CAh) LCD_DATA(1); // Display custom character msDelay(10); } void LCD_CMD(unsigned char x){ LCD = x; RS = 0; RW = 0; EN = 1; msDelay(5); EN = 0; } ---> </pre>	<pre> void LCD_DATA(unsigned char w){ LCD = w; RS = 1; RW = 0; EN = 1; msDelay(5); EN = 0; } void Send_Data(unsigned char *Str){ while (*Str) // Bucla pana cand s-au terminat datele LCD_DATA(*Str++); // Trimiterea datelor la LCD una cate una } void LCD_INI(void) { msDelay(250); // Call delay LCD_CMD(0x38); // Comanada pe 8 bits a LCD-ului LCD_CMD(0x0C); // Display ON LCD_CMD(0x01); // Clear LCD } void msDelay(unsigned int Time) { while (Time--) timer1_delay_1ms(1); // Delay de 1ms folosind Timer1 } ---> </pre>
--	---	--

```
void timer1_delay_1ms(unsigned int delay)
{
    unsigned int i;
    TMOD |= 0x10; // Timer1 in mode 1
    TH1 = 0xEE; // Initial value for 5ms delay at 11.0592 MHz
    TL1 = 0x00;
    TR1 = 1; // Start Timer1

    for (i = 0; i < delay; i++) {
        while (!TF1); // Wait until Timer1 overflow
        TF1 = 0; // Clear overflow flag
    }

    TR1 = 0; // Stop Timer1
}
```

• COD IN ASM:

<pre>org 0000h acall init_timer ; acall lcd acall heart acall caracter acall adc ; functie pentru initializarea timer-ului 1 init_timer: orl tmod, #10h ; setare timer 1 in modul 1 (16-bit timer) ret lcd: ; initializare lcd mov a, #38h ; afisare caractere pe o matrice de 5x7 acall instructiuni acall delay_timer mov a, #0eh ; pornire lcd acall instructiuni acall delay_timer mov a, #01h ; stergere lcd acall instructiuni acall delay_timer mov a, #80h ; pozitionare cursor pe prima linie la pozitia 0 acall instructiuni acall delay_timer</pre> <p style="text-align: right;">---></p>	<pre> mov a, #'T' acall date acall delay_timer mov a, #'e' acall date acall delay_timer mov a, #'m' acall date acall delay_timer mov a, #'p' acall date acall delay_timer mov a, #'e' acall date acall delay_timer mov a, #'r' acall date acall delay_timer mov a, #'a' acall date acall delay_timer mov a, #'t' acall date acall delay_timer</pre> <p style="text-align: right;">---></p>	<pre> mov a, #'u' acall date acall delay_timer mov a, #'r' acall date acall delay_timer mov a, #'a' acall date acall delay_timer mov a, #' ' acall date acall delay_timer mov a, #'e' acall date acall delay_timer mov a, #'s' acall date acall delay_timer mov a, #'t' acall date acall delay_timer mov a, #'e' acall date acall delay_timer ret</pre> <p style="text-align: right;">---></p>
--	---	--

<pre> instructiuni: ; trimite instructiuni la lcd mov p2, a clr p3.0 ; rs=0 pentru instructiuni clr p3.1 ; r/w=0 pentru scriere setb p3.2 ; e=1 se activeaza pinul e acall delay_timer clr p3.2 ; e=0 se dezactiveaza pinul e ret date: ; scrie date pe lcd mov p2, a setb p3.0 ; rs=1 pentru date clr p3.1 ; r/w=0 pentru scriere setb p3.2 ; e=1 se activeaza pinul e acall delay_timer clr p3.2 ; e=0 se dezactiveaza pinul e ret adc: mov p1, #0ffh ; set p1 as input port clr p2.7 setb p2.6 nop nop nop clr p2.6 </pre> <p style="text-align: right;">---></p>	<pre> wait_eoc: jb p2.4, wait_eoc ; wait for end of conversion setb p2.7 ; deselect adc mov a, p1 ; read adc value from p1 ; convert the adc value to hexadecimal mov b, a swap a ; swap nibbles anl a, #0fh ; isolate high nibble ; convert high nibble to hex character cjne a, #0ah, high_digit add a, #07h high_digit: add a, #30h mov r6, a ; save the converted high nibble ; convert low nibble to hex character mov a, b anl a, #0fh ; isolate low nibble cjne a, #0ah, low_digit add a, #07h </pre> <p style="text-align: right;">---></p>	<pre> low_digit: add a, #30h mov r5, a ; save the converted low nibble ; prepare to display on lcd mov a, #0c5h acall instructiuni ; display hex characters on lcd mov a, r6 acall date mov a, r5 acall date mov a, #'h' acall date ret heart: mov a, #40h acall instructiuni mov a, #00h acall date mov a, #0ah acall date mov a, #1fh acall date mov a, #1fh acall date mov a, #0eh acall date mov a, #04h acall date mov a, #00h acall date mov a, #00h acall date </pre> <p style="text-align: right;">---></p>
---	--	--

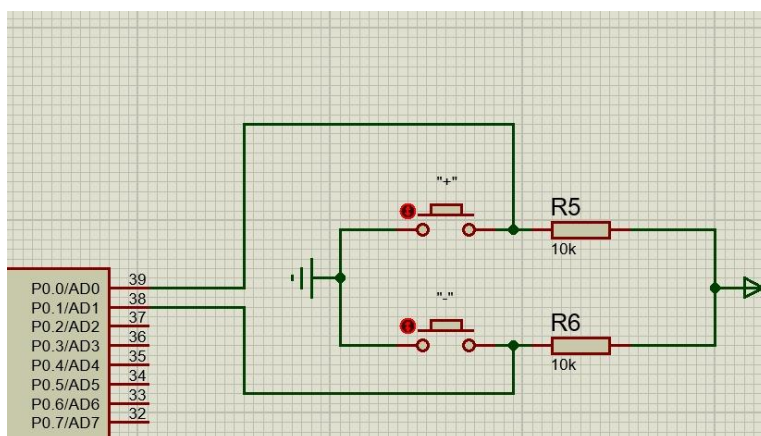
<pre> mov a, #0cah ; set cursor position for heart character (c8h) acall instructiuni mov a, #00h ; display heart character acall date acall delay_timer ret caracter: mov a, #48h acall instructiuni mov a, #00h acall date mov a, #0ah acall date mov a, #0ah acall date mov a, #0ah acall date mov a, #11h acall date mov a, #0eh acall date mov a, #00h acall date mov a, #00h acall date </pre> <p style="text-align: right;">---></p>	<pre> mov a, #0cch ; set cursor position for custom character (cah) acall instructiuni mov a, #01h ; display custom character acall date acall delay_timer ret ; functie de intarziere folosind timer 1 delay_timer: mov TH1, #0EEh ; Incarca TH1 pentru un overflow de 5ms (presupunand un cristal de 12MHz) mov TL1, #00h ; Incarca TL1 pentru un overflow de 5ms (presupunand un cristal de 12MHz) setb TR1 ; Porneste Timer 1 d1: jnb TF1, d1 ; Asteapta overflow clr TF1 ; Opreste Timer 1 clr TR1 ; Reseteaza flag-ul Timer 1 ret gata: end </pre>
---	---

19. TASTATURA

O tastatură este un dispozitiv de tip mașină de scris, care folosește un aranjament de butoane pentru a acționa ca și comutatoare electronice. Tastatura conține 2 butoane pentru a seta temperatura dorită:

- + (pentru a crește valoarea temperaturii dorite)
- - (pentru a scădea valoarea temperaturii dorite).

Conectarea acestor butoane fără rezistențele specificate nu ar fi fost corectă deoarece atunci când butonul nu este apăsat, intrarea se află într-o stare nedefinită (ca și cum ar fi lăsată în aer), ea nefiind conectată nici la masă, nici la alimentare. Starea de impedanță mărită nu poate fi citită de către circuitele interne ale microcontrolerului, un bit dintr-un registru poate să ia doar valorile 0 sau 1. Prin cele 2 butoane se controlează temperatura din cameră, activarea lor se face prin apăsare.



4.9. Schema in Proteus a tastaturii cu cele doua butoane

- Cod pentru tastatura in ASM:

```
; Tastatura
jnb P0.0, Buton1 ; Se verifica daca "+" este apasat
jnb P0.1, Buton2 ; Se verifica daca "-" este apasat
sjmp ADC
Buton1:
mov r3, #255
debouncing_1:
jb P0.0, Buton1
djnz r3, debouncing_1
inc R0
b1:
jnb P0.0, b1
sjmp ADC

Buton2:
mov r3, #255
debouncing_2:
jb P0.1, Buton2
djnz r3, debouncing_2
dec R0
b2:
jnb P0.1, b2
sjmp ADC
```

- Cod pentru tastatura in C:

```
void check_buttons() {
    if (BUTTON_PLUS == 0) { // Se verifica daca "+"
        este apasat
        msDelay(20); // Debouncing
        while (BUTTON_PLUS == 0); // Asteapta
        eliberarea butonului
        temp_ref++;
    }

    if (BUTTON_MINUS == 0) { // Se verifica daca "-"
        este apasat
        msDelay(20); // Debouncing
        while (BUTTON_MINUS == 0); // Asteapta
        eliberarea butonului
        temp_ref--;
    }
}
```

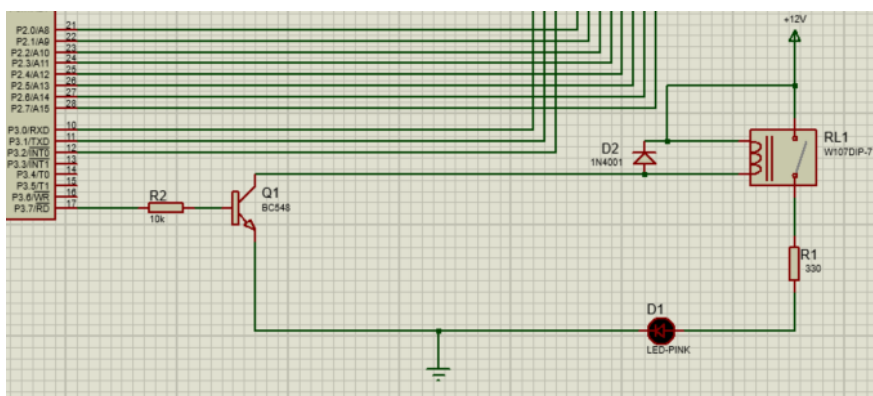
20. RELEUL

Releul este un dispozitiv electromecanic care transformă un semnal electric într-o mișcare mecanică. El este alcătuit dintr-o bobină din conductori izolați înfășurați pe un nucleu metalic și o armătură metalică cu unul sau mai multe contacte. În momentul în care o tensiune de alimentare este aplicată la bornele unei bobine, curentul circulă și va fi produs un câmp magnetic care mișcă armătura pentru a închide un set de contacte și/sau pentru a deschide un alt set.

Am utilizat un releu cu un singur contact ce comută în momentul în care P3.7 este activat. Astfel, tranzistorul de comandă se deschide și realizează legătura la masă a releului, moment în care contactul se va mișca și se va aprinde un led de culoare roz care indică faptul că temperatura din cameră este mai mare decât cea setată de utilizator.

Rezistența de 10K din baza tranzistorului limitează curentul dinspre microcontroler la o valoare solicitată de tranzistor. Releul ales este W107DIP-7, având nevoie de o alimentare de +12V.

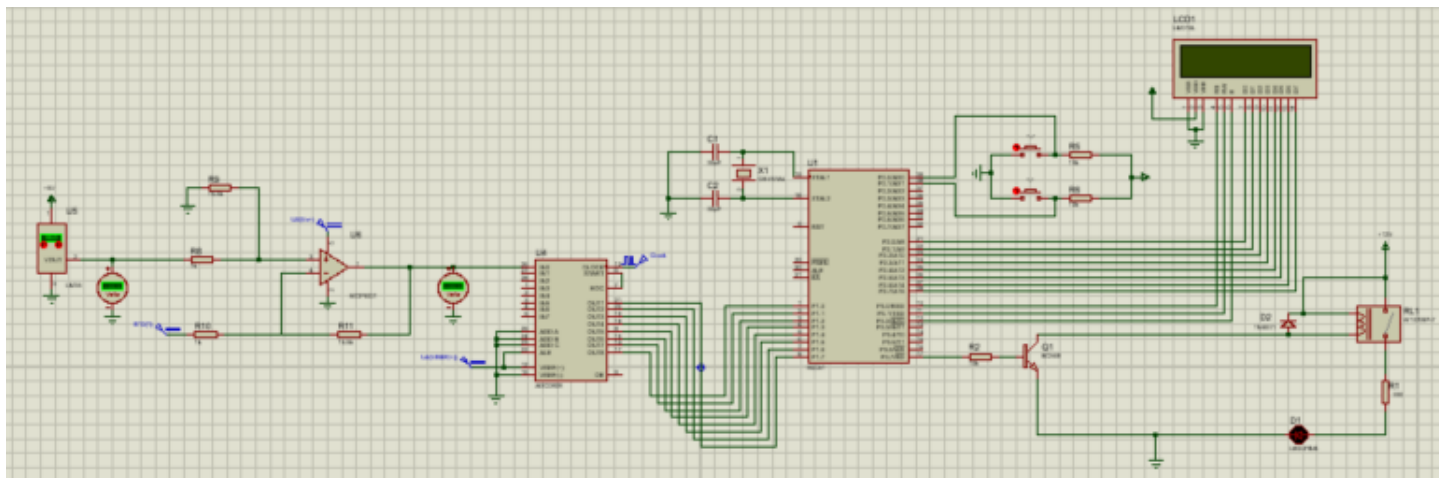
Releele sunt aparate destinate protecției împotriva suprasarcinilor de durată. Dacă temperatura prescrisă de noi este mai mare decât cea măsurată de senzor, atunci microcontrolerul va comanda releul prin intermediul driverului acestuia (tranzistorul 2N2222).



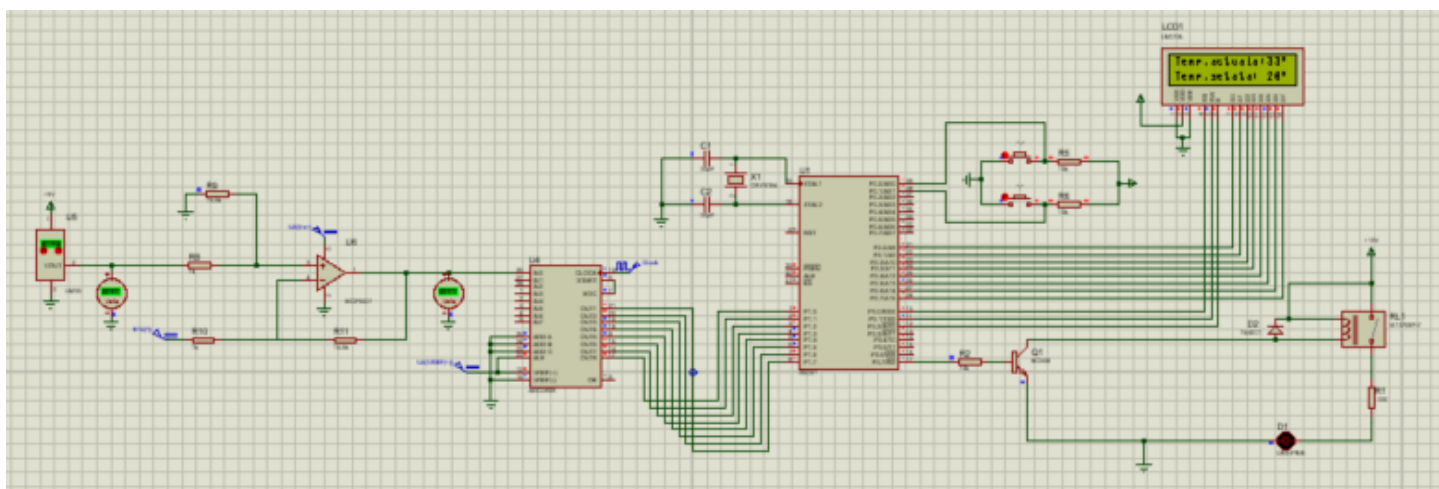
5.0. Scheme in Proteus a releului

<ul style="list-style-type: none"> • Cod pentru releu in ASM: <pre> ; Control releu mov a, P1 ;Valoarea actuala mov R6, P0 ;Valoarea setata subb a, R6 jc start ; Se verifica daca temperatura setata este mai mare decat cea actuala sjmp oprire start: clr P3.7 ; Pornire releu sjmp gata oprire: setb P3.7 ;Oprire releu </pre>	<ul style="list-style-type: none"> • Cod pentru releu in C: <pre> void ADC() { temp_actuala = P1; // Simulare citire temperatura actuala de la un ADC // Control releu if (temp_ref < temp_actuala) { // Se verifica daca temperatura setata este mai mare decat cea actuala RELAY = 0; // Pornire releu } else { RELAY = 1; // Oprire releu } } </pre>
--	--

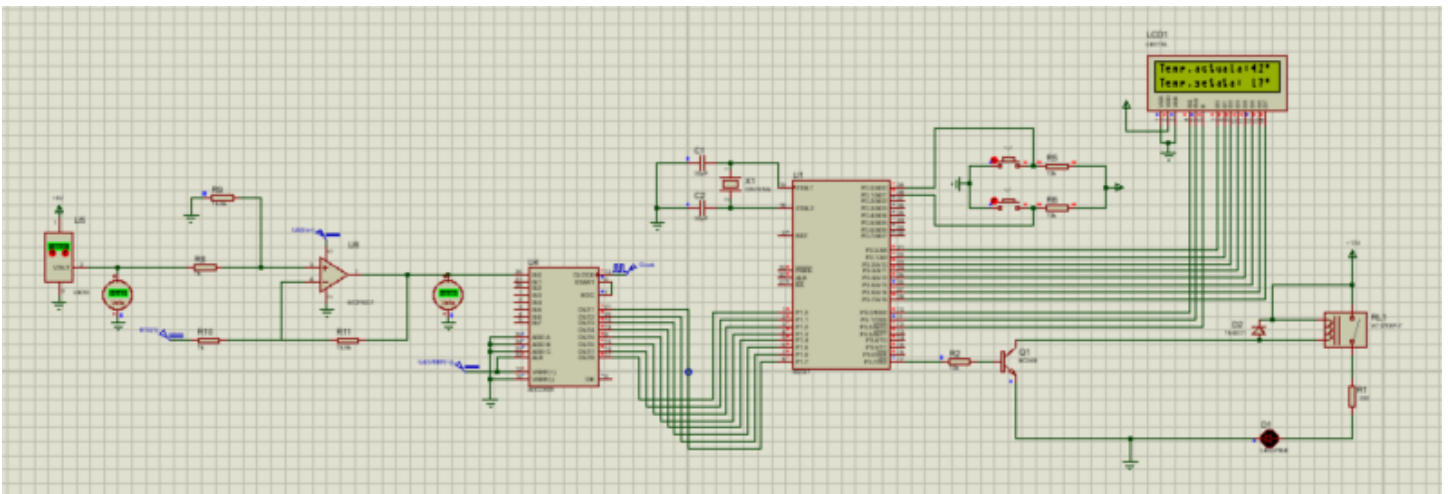
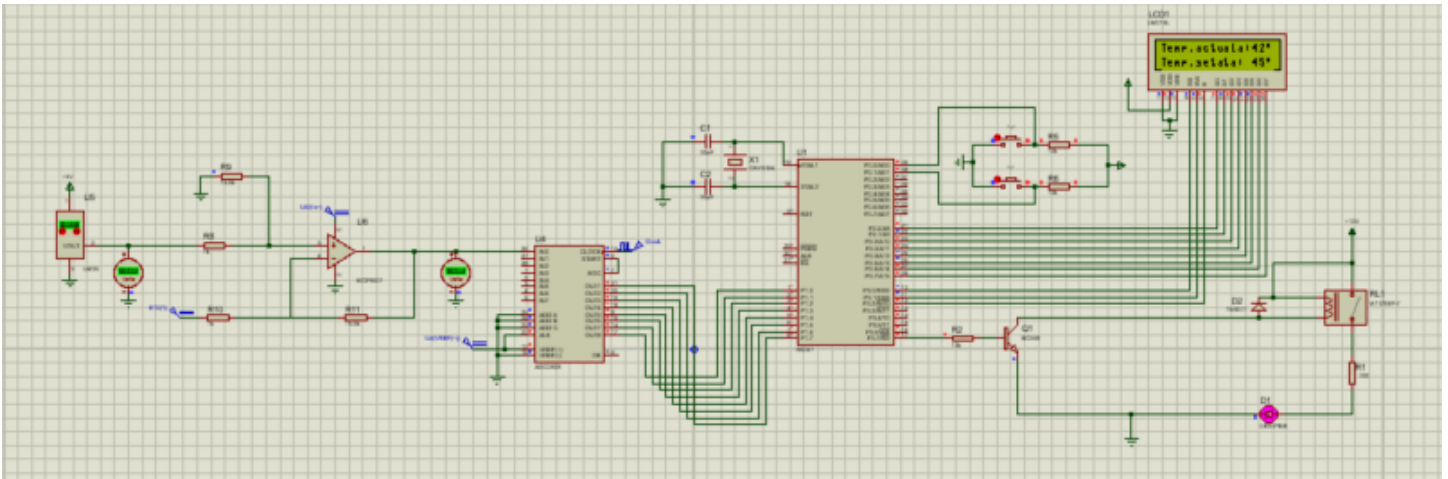
21. SCHEMA FINALA A TERMOSTATULUI DE CAMERA



5.1. Schema finala a termostatlui de camera in Proteus



5.2. Schema finala la temperatura de 33°C



22. CODURILE FINALE

- Cod in ASM:

<pre> org 0000h mov R0, #20 ; tensiunea de referinta din incapere setb P0.0 setb P0.1 ; setez butoanele initial ca fiind neapasate acall init_timer ; Initializare Timer acall LCD acall ADC ; Functie pentru initializarea Timer-ului 1 init_timer: orl TMOD, #10h ; Setare Timer 1 in modul 1 (16-bit timer) ret LCD: ; Initializare LCD mov a, #38h ; Afisare caractere pe o matrice de 5x7 acall instructiuni acall delay_timer mov a, #0Eh ; Pornire LCD acall instructiuni acall delay_timer mov a, #01h ; Stergere LCD acall instructiuni acall delay_timer ---</pre>	<pre> ; Afisare pe prima linie mesajul:"Temp. actuala" mov a, #80h ; pozitionare cursor pe prima linie la pozitia 0 acall instructiuni acall delay_timer mov a, #'T' acall date acall delay_timer mov a, #'e' acall date acall delay_timer mov a, #'m' acall date acall delay_timer mov a, #'p' acall date acall delay_timer mov a, #'.' acall date acall delay_timer mov a, #'a' acall date acall delay_timer ---</pre>
---	--

<pre> mov a, #'c' acall date acall delay_timer mov a, #'t' acall date acall delay_timer mov a, #'u' acall date acall delay_timer mov a, #'a' acall date acall delay_timer mov a, #'l' acall date acall delay_timer mov a, #'a' acall date acall delay_timer mov a, #':' acall date acall delay_timer ; Afisare pe a doua linie mesajul:"Temp.setata" mov a, #0C0h ; Pozitionare cursor pe linia a doua la pozitia 0 acall instructiuni acall delay_timer </pre> <p style="text-align: right;">---></p>	<pre> mov a, #'T' acall date acall delay_timer mov a, #'e' acall date acall delay_timer mov a, #'m' acall date acall delay_timer mov a, #'p' acall date acall delay_timer mov a, #',' acall date acall delay_timer mov a, #'s' acall date acall delay_timer mov a, #'e' acall date acall delay_timer mov a, #'t' acall date acall delay_timer mov a, #'a' acall date acall delay_timer </pre> <p style="text-align: right;">---></p>
--	---

<pre> mov a, #'t' acall date acall delay_timer mov a, #'a' acall date acall delay_timer mov a, #'.' acall date acall delay_timer ret instructiuni: ; Trimite instructiuni la LCD mov P2, a clr P3.0 ; RS=0 pentru instructiuni clr P3.1 ; R/W=0 pentru scriere setb P3.2 ; E=1 se activeaza pinul E acall delay_timer clr P3.2 ; E=0 se dezactiveaza pinul E Ret date: ; Scrie date pe LCD mov P2, a setb P3.0 ; RS=1 pentru date clr P3.1 ; R/W=0 pentru scriere setb P3.2 ; E=1 se activeaza pinul E acall delay_timer clr P3.2 ; E=0 se dezactiveaza pinul E Ret </pre> <p style="text-align: right;">---></p>	<pre> ADC: mov a, #8Dh ; Pozitionare cursor pe prima linie pozitia 13 acall instructiuni acall delay_timer mov r1, P1 ; Afisare temperatura setata mov a, r1 mov b, #6h ; 5/2^8 div ab mov b, #0Ah div ab push b add a, #30H acall date acall delay_timer pop b mov a, b add a, #30h acall date acall delay_timer ; Afisare temperatura actuala mov a, #0DFh ; simbolul de la grade acall date acall delay_timer mov a, #0CDh ; Pozitionare cursor pe a doua linie pozitia 13 acall instructiuni acall delay_timer </pre> <p style="text-align: right;">---></p>
--	---

<pre> mov a, R0 mov b, #0Ah div ab push b add a, #30h acall date acall delay_timer pop b mov a, b add a, #30h acall date acall delay_timer mov a, #0DFh acall date acall delay_timer ; Control releu mov a, P1 ;Valoarea actuala mov R6, P0 ;Valoarea setata subb a, R6 jc start ; Se verifica daca temperatura setata este mai mare decat cea actuala sjmp oprire start: clr P3.7 ; Pornire releu sjmp gata oprire: setb P3.7 ; Oprire releu ; Tastatura jnb P0.0, Buton1 ; Se verifica daca "+" este apasat jnb P0.1, Buton2 ; Se verifica daca "-" este apasat sjmp ADC </pre>	<pre> Buton1: mov r3, #255 debouncing_1: jnb P0.0, Buton1 djnz r3, debouncing_1 inc R0 b1: jnb P0.0, b1 sjmp ADC Buton2: mov r3, #255 debouncing_2: jnb P0.1, Buton2 djnz r3, debouncing_2 dec R0 b2: jnb P0.1, b2 sjmp ADC ; Functie de intarziere folosind Timer 1 delay_timer: mov TH1, #0EEh ; Incarca TH1 pentru un overflow de 5ms (presupunand un cristal de 12MHz) mov TL1, #00h ; Incarca TL1 pentru un overflow de 5ms (presupunand un cristal de 12MHz) setb TR1 ; Porneste Timer 1 d1: jnb TF1, d1 ; Asteapta overflow clr TF1 ; Opreste Timer 1 clr TR1 ; Reseteaza flag-ul Timer 1 ret gata: end </pre>
--	---

--->

● Cod in C:

<pre>#include <REG51.H> sfr LCD = 0xA0; sbit RS = P3^0; sbit RW = P3^1; sbit EN = P3^2; sbit RELAY = P3^7; sbit BUTTON_PLUS = P0^0; sbit BUTTON_MINUS = P0^1; unsigned char temp_ref = 20; // Tensiunea de referinta din incapere unsigned char temp_actuala = 0; // Variabila pentru temperatura actuala void LCD_CMD(unsigned char x); void LCD_DATA(unsigned char w); void LCD_INI(void); void Send_Data(unsigned char *Str); void msDelay(unsigned int); void display_actual_temp(unsigned char value); void display_set_temp(unsigned char value); void ADC(); void check_buttons(); void timer1_delay_1ms(unsigned int delay); void main(void) { P0 = 0x03; // Setez butoanele initial ca fiind neapasate msDelay(200); // 0.2 secunde LCD_INI(); // Initializarea LCD</pre> <p style="text-align: right;">---></p>	<pre>LCD_CMD(0x80); Send_Data("Temp.actuala:"); // Scribe mesaj pe LCD msDelay(200); // 0.2 secunde LCD_CMD(0xC0); Send_Data("Temp.setata:"); // Scribe mesaj pe LCD while (1) { check_buttons(); ADC(); display_actual_temp(temp_actuala); display_set_temp(temp_ref); } void ADC() { temp_actuala = P1; // Simulare citire temperatura actuala de la un ADC // Control releu if (temp_ref > temp_actuala) { // Se verifica daca temperatura setata este mai mare decat cea actuala RELAY = 0; // Pornire releu } else { RELAY = 1; // Opreire releu } } void check_buttons() { if (BUTTON_PLUS == 0) { // Se verifica daca "+" este apasat msDelay(20); // Debouncing while (BUTTON_PLUS == 0); // Asteapta eliberarea butonului temp_ref++; } }</pre> <p style="text-align: right;">---></p>
--	--

```

if (BUTTON_MINUS == 0) { // Se verifica daca "-" este
apasat
    msDelay(20); // Debouncing
    while (BUTTON_MINUS == 0); // Asteapta
    eliberarea butonului
    temp_ref--;
}
}
void display_actual_temp(unsigned char value)
{
    unsigned char tens, units;

    // Set cursor position for actual temperature
    LCD_CMD(0x80 + 13); // Prima linie
    value = P1;
    tens = (value / 0x06) / 10; // Obtinerea cifrei zecilor
    tens = tens + 0x30;
    units = (value / 0x06) % 10; // Obtinerea cifrei unitatilor
    units = units + 0x30;
    LCD_DATA(tens); // Afisarea primei cifre
    LCD_DATA(units); // Afisarea celei de-a doua cifre
    LCD_DATA(0xDF); // Afisarea simbolului gradului
}

void display_set_temp(unsigned char value)
{
    unsigned char tens, units;

    // Set cursor position for set temperature
    LCD_CMD(0xC0 + 13); // A doua linie

    tens = value / 10; // Obtinerea cifrei zecilor
    units = value % 10; // Obtinerea cifrei unitatilor

    LCD_DATA(tens + '0'); // Afisarea primei cifre
    LCD_DATA(units + '0'); // Afisarea celei de-a doua
cifre
    LCD_DATA(0xDF); // Afisarea simbolului gradului
}

```

--->

```

void LCD_CMD(unsigned char x)
{
    LCD = x;
    RS = 0;
    RW = 0;
    EN = 1;
    msDelay(5);
    EN = 0;
}

void LCD_DATA(unsigned char w)
{
    LCD = w;
    RS = 1;
    RW = 0;
    EN = 1;
    msDelay(5);
    EN = 0;
}

void Send_Data(unsigned char *Str)
{
    while(*Str) // Bucla pana cand s-au terminat datele
        LCD_DATA(*Str++); // Trimiterea datelor la LCD
una cate una
}

void LCD_INI(void)
{
    msDelay(250); // Call delay
    LCD_CMD(0x38); // Comanada pe 8 bits a LCD-ului
    LCD_CMD(0x0C); // Display ON
    LCD_CMD(0x01); // Clear LCD
}

void msDelay(unsigned int Time)
{
    while (Time--)
        timer1_delay_1ms(1); // Delay de 1ms folosind
Timer1
}

```

--->

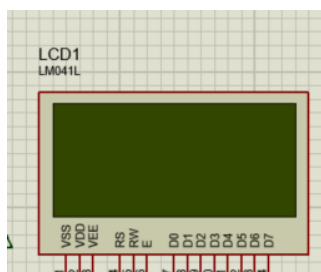
```
void timer1_delay_1ms(unsigned int delay)
{
    unsigned int i;
    TMOD |= 0x10; // Timer1 in mode 1
    TH1 = 0xEE; // Initial value for 5ms delay at 11.0592 MHz
    TL1 = 0x00;
    TR1 = 1; // Start Timer1

    for (i = 0; i < delay; i++) {
        while (!TF1); // Wait until Timer1 overflow
        TF1 = 0; // Clear overflow flag
    }

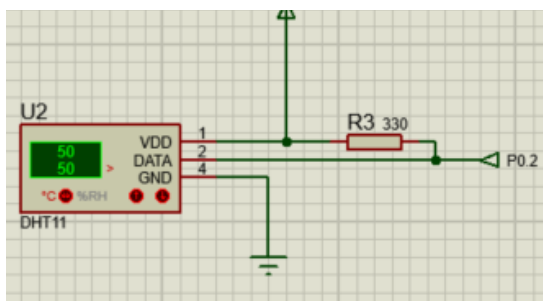
    TR1 = 0; // Stop Timer1
}
```

23. SUPLIMENTAR

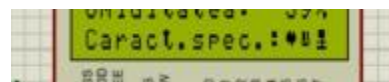
Suplimentar, pe lângă cele două caractere speciale făcute anterior în acest proiect, am mai creat încă unul și, totodată, am utilizat și un senzor de umiditate digital, care arată umiditatea în funcție de temperatura actuală. Pentru a putea ilustra toate cele necesare (temperatura actuală, temperatura setată, umiditatea și caracterele speciale) a trebuit să modific LCD-ul ales LM016L în LCD-ul LM041L care scrie textul pe patru linii, fiecare linie conținând 16 celule.



5.5. LCD-ul LM041L



5.6. Senzorul de umiditate DHT11



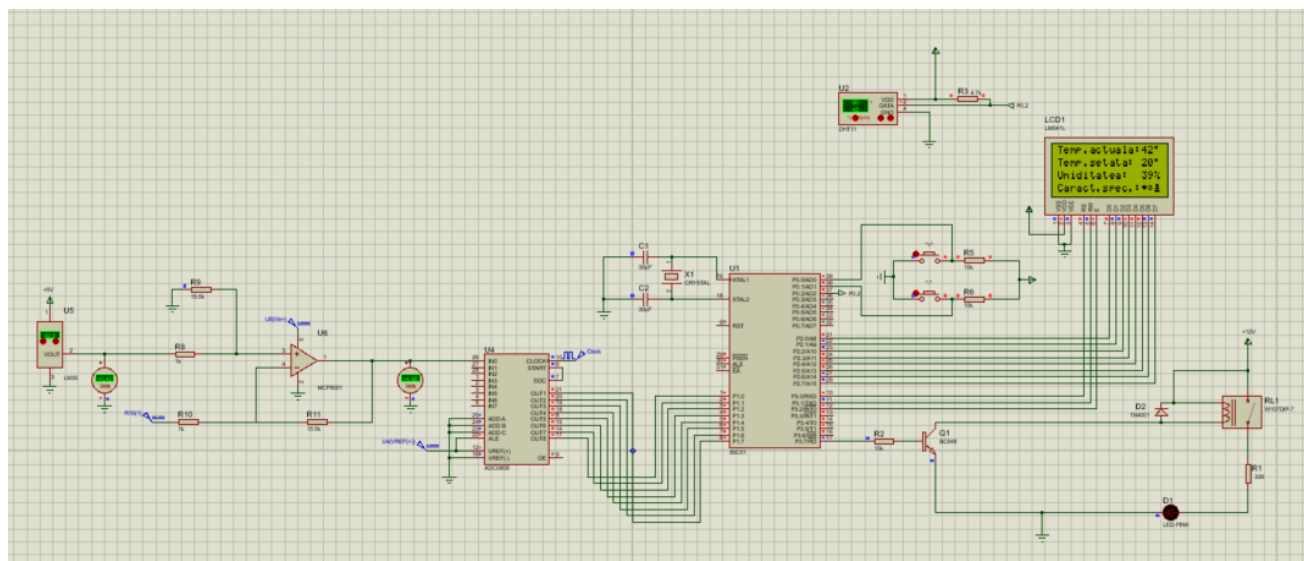
5.7. Noul caracter special: un omulet

Senzorul de umiditate DHT11

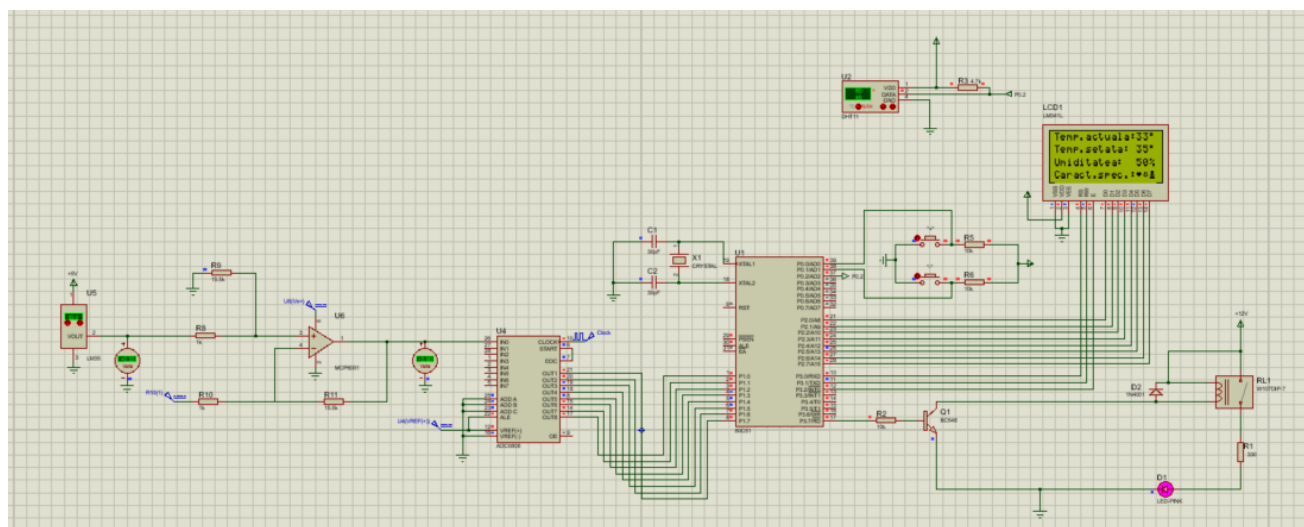
Este un senzor digital folosit frecvent pentru măsurarea umidității relative și a temperaturii. Este cunoscut pentru fiabilitatea și ușurința în utilizare. Iată câteva detalii despre DHT11:

- Interval de măsurare a umidității:
 - 20-90% RH (Umiditate Relativă)
 - Precizie: $\pm 5\%$ RH
- Interval de măsurare a temperaturii:
 - 0-50°C
 - Precizie: $\pm 2^\circ\text{C}$

24. SCHEMA SUPLIMENTARA



5.8. Schema in Proteus pentru temperatura de 42°C si temp. Setata de 20°C



5.9. Schema in Proteus pentru temperatura de 33°C si temp. Setata de 35°C

25. COD SUPLIMENTAR

• COD IN C:

<pre>#include <REG51.H> sfr LCD = 0xA0; sbit RS = P3^0; sbit RW = P3^1; sbit EN = P3^2; sbit RELAY = P3^7; sbit BUTTON_PLUS = P0^0; sbit BUTTON_MINUS = P0^1; unsigned char temp_ref = 20; // Tensiunea de referinta din incapere unsigned char temp_actuala = 0; // Variabila pentru temperatura actuala unsigned char humidity = 0; // Variabila pentru umiditate void LCD_CMD(unsigned char x); void LCD_DATA(unsigned char w); void LCD_INI(void); void Send_Data(unsigned char *Str); void msDelay(unsigned int); void display_actual_temp(unsigned char value); void display_set_temp(unsigned char value); void display_humidity(unsigned char value); --></pre>	<pre>void ADC(); void check_buttons(); void timer1_delay_1ms(unsigned int delay); void heart(); void caracter(); void omulet(); void main(void) { P0 = 0x03; // Setez butoanele initial ca fiind neapasate msDelay(200); // 0.2 secunde LCD_INI(); // Initializarea LCD LCD_CMD(0x80); Send_Data("Temp.actuala:"); // Scrie mesaj pe LCD msDelay(200); // 0.2 secunde LCD_CMD(0xC0); Send_Data("Temp.setata:"); // Scrie mesaj pe LCD msDelay(200); // 0.2 secunde LCD_CMD(0x90); Send_Data("Umiditatea:"); // Scrie mesaj pe LCD linia 3 msDelay(200); // 0.2 secunde LCD_CMD(0xD0); Send_Data("Caract.spec.:"); // Scrie mesaj pe LCD linia 4 msDelay(200); // 0.2 secunde --></pre>	<pre>heart(); caracter(); omulet(); while (1) { check_buttons(); ADC(); display_actual_temp(temp_actuala); display_set_temp(temp_ref); display_humidity(temp_actuala); } } void ADC() { temp_actuala = P1; // Simulare citire temperatura actuala de la un ADC humidity = P2; // Simulare citire umiditate de la un ADC // Control releu if (temp_ref > temp_actuala) { // Se verifica daca temperatura setata este mai mare decat cea actuala RELAY = 0; // Pornire releu (LED aprins) } else { RELAY = 1; // Opreire releu (LED stins) } } --></pre>
--	--	---

<pre> void check_buttons() { if (BUTTON_PLUS == 0) { // Se verifica daca "+" este apasat msDelay(20); // Debouncing while (BUTTON_PLUS == 0); // Asteapta eliberarea butonului temp_ref++; } if (BUTTON_MINUS == 0) { // Se verifica daca "-" este apasat msDelay(20); // Debouncing while (BUTTON_MINUS == 0); // Asteapta eliberarea butonului temp_ref--; } } void display_actual_temp(unsigned char value) { unsigned char tens, units; // Set cursor position for actual temperature LCD_CMD(0x8D); // Prima linie value = P1; tens = (value / 0x06) / 10; // Obtinerea cifrei zecilor tens = tens + 0x30; units = (value / 0x06) % 10; // Obtinerea cifrei unitatilor units = units + 0x30; LCD_DATA(tens); // Afisarea primei cifre LCD_DATA(units); // Afisarea cele de-a doua cifre LCD_DATA(0xDF); // Afisarea simbolului gradului } --> </pre>	<pre> void display_set_temp(unsigned char value) { unsigned char tens, units; // Set cursor position for set temperature LCD_CMD(0xCD); // A doua linie tens = value / 10; // Obtinerea cifrei zecilor units = value % 10; // Obtinerea cifrei unitatilor LCD_DATA(tens + '0'); // Afisarea primei cifre LCD_DATA(units + '0'); // Afisarea celei de-a doua cifre LCD_DATA(0xDF); // Afisarea simbolului gradului } void display_humidity(unsigned char temp_value) { unsigned char humidity_value; unsigned char tens, units; if (temp_value < 10) { humidity_value = 50 + 2 * (temp_value - 10); } else { humidity_value = 50 - (10 - temp_value); } if (humidity_value > 50) { humidity_value = 50; } else if (humidity_value < 0) { humidity_value = 0; } LCD_CMD(0x9D); } --> </pre>	<pre> tens = humidity_value / 10; // Obtinerea cifrei zecilor units = humidity_value % 10; // Obtinerea cifrei unitatilor LCD_DATA(tens + '0'); // Afisarea primei cifre LCD_DATA(units + '0'); // Afisarea celei de-a doua cifre LCD_DATA('%'); // Afisarea simbolului procent } void LCD_CMD(unsigned char x) { LCD = x; RS = 0; RW = 0; EN = 1; msDelay(5); EN = 0; } void LCD_DATA(unsigned char w) { LCD = w; RS = 1; RW = 0; EN = 1; msDelay(5); EN = 0; } void Send_Data(unsigned char *Str) { while(*Str) // Bucla pana cand s-au terminat datele LCD_DATA(*Str++); // Trimiterea datelor la LCD una cate una } --> </pre>
--	--	---

<pre> void LCD_INI(void){ msDelay(250); // Call delay LCD_CMD(0x38); // Comanada pe 8 bits a LCD-ului LCD_CMD(0x0C); // Display ON LCD_CMD(0x01); // Clear LCD LCD_CMD(0x06); // Entry mode set LCD_CMD(0x80); // Set cursor at beginning of first line } void msDelay(unsigned int Time){ while (Time--) timer1_delay_1ms(1); // Delay de 1ms folosind Timer1 } void timer1_delay_1ms(unsigned int delay){ unsigned int i; TMOD = 0x10; // Timer1 in mode 1 TH1 = 0xEE; // Initial value for 5ms delay at 11.0592 MHz TL1 = 0x00; TR1 = 1; // Start Timer1 for (i = 0; i < delay; i++) { while (!TF1); // Wait until Timer1 overflow TF1 = 0; // Clear overflow flag } --> </pre>	<pre> TR1 = 0; // Stop Timer1 } void heart(){ LCD_CMD(64); // Set CGRAM address for the heart character LCD_DATA(0x00); LCD_DATA(0x0A); LCD_DATA(0x1F); LCD_DATA(0x1F); LCD_DATA(0x0E); LCD_DATA(0x04); LCD_DATA(0x00); LCD_DATA(0x00); LCD_CMD(0xDD); // Set cursor position for heart character on line 4 LCD_DATA(0); // Display heart character msDelay(10); } void caracter() { LCD_CMD(72); // Set CGRAM address for the custom character LCD_DATA(0x00); LCD_DATA(0x0A); LCD_DATA(0x0A); LCD_DATA(0x0A); LCD_DATA(0x11); LCD_DATA(0x0E); LCD_DATA(0x00); LCD_DATA(0x00); LCD_DATA(0x00); --> </pre>	<pre> LCD_CMD(0xDE); // Set cursor position for custom character on line 4 LCD_DATA(1); // Display custom character msDelay(10); } void omulet(){ LCD_CMD(80); // Set CGRAM address for the custom character LCD_DATA(0x0E); LCD_DATA(0x0E); LCD_DATA(0x0E); LCD_DATA(0x04); LCD_DATA(0x1F); LCD_DATA(0x04); LCD_DATA(0x1F); LCD_DATA(0x11); LCD_CMD(0xDF); // Set cursor position for custom character on line 4 LCD_DATA(2); // Display custom character msDelay(10); } </pre>
---	---	---

- COD IN ASM**

<pre> org 0000h mov R0, #20 ; tensiunea de referinta din incapere setb P0.0 setb P0.1 ; setez butoanele initial ca fiind neapasate acall init_timer ; Initializare Timer acall LCD ACALL HEART ACALL CHARACTER ACALL OMULET acall ADC acall display_humidity sjmp \$; Infinite loop to prevent re-running the setup init_timer: orl TMOD, #10h ; Setare Timer 1 in modul 1 (16-bit timer) ret LCD: ; Initializare LCD mov a, #38h ; Afisare caractere pe o matrice de 5x7 acall instructiuni acall delay_timer mov a, #0Ch ; Pornire LCD acall instructiuni acall delay_timer --> </pre>	<pre> mov a, #01h ; Stergere LCD acall instructiuni acall delay_timer ; Afisare pe prima linie mesajul:"Temp. actuala" mov a, #80h ; pozitionare cursor pe prima linie la pozitia 0 acall instructiuni acall delay_timer mov a, #'T' acall date acall delay_timer mov a, #'e' acall date acall delay_timer mov a, #'m' acall date acall delay_timer mov a, #'p' acall date acall delay_timer mov a, #'.' acall date acall delay_timer mov a, #'a' acall date acall delay_timer --> </pre>	<pre> mov a, #'c' acall date acall delay_timer mov a, #'t' acall date acall delay_timer mov a, #'u' acall date acall delay_timer mov a, #'a' acall date acall delay_timer mov a, #'l' acall date acall delay_timer mov a, #'a' acall date acall delay_timer mov a, #'.' acall date acall delay_timer ; Afisare pe a doua linie mesajul:"Temp.setata" mov a, #0C0h ; Pozitionare cursor pe linia a doua la pozitia 0 acall instructiuni acall delay_timer --> </pre>
--	--	--

<p>mov a, #'T' acall date acall delay_timer</p> <p>mov a, #'e' acall date acall delay_timer</p> <p>mov a, #'m' acall date acall delay_timer</p> <p>mov a, #'p' acall date acall delay_timer</p> <p>mov a, #'.' acall date acall delay_timer mov a, #'s' acall date acall delay_timer</p> <p>mov a, #'e' acall date acall delay_timer</p> <p>mov a, #'t' acall date acall delay_timer</p> <p>mov a, #'a' acall date acall delay_timer--></p>	<p>mov a, #'t' acall date acall delay_timer</p> <p>mov a, #'a' acall date acall delay_timer</p> <p>mov a, #'.' acall date acall delay_timer</p> <p>mov a, #90h ; Pozitionare cursor pe linia a doua la pozitia 0 acall instructiuni acall delay_timer</p> <p>mov a, #'U' acall date acall delay_timer</p> <p>mov a, #'m' acall date acall delay_timer</p> <p>mov a, #'i' acall date acall delay_timer</p> <p>mov a, #'d' acall date acall delay_timer</p> <p>--></p>	<p>mov a, #'i' acall date acall delay_timer</p> <p>mov a, #'t' acall date acall delay_timer</p> <p>mov a, #'a' acall date acall delay_timer</p> <p>mov a, #'t' acall date acall delay_timer</p> <p>mov a, #'e' acall date acall delay_timer</p> <p>mov a, #'a' acall date acall delay_timer</p> <p>mov a, #'.' acall date acall delay_timer</p> <p>mov a, #0D0h acall instructiuni acall delay_timer</p> <p>--></p>
--	---	--

mov a, #'C' acall date acall delay_timer mov a, #'a' acall date acall delay_timer mov a, #'r' acall date acall delay_timer mov a, #'a' acall date acall delay_timer mov a, #'c' acall date acall delay_timer mov a, #'t' acall date acall delay_timer mov a, #'.' acall date acall delay_timer mov a, #'s' acall date acall delay_timer mov a, #'p' acall date acall delay_timer	mov a, #'e' acall date acall delay_timer mov a, #'c' acall date acall delay_timer mov a, #'.' acall date acall delay_timer mov a, #'.' acall date acall delay_timer instructiuni: ; Trimite instructiuni la LCD mov P2, a clr P3.0 ; RS=0 clr P3.1 ; R/W=0 setb P3.2 ; E=1 acall delay_timer clr P3.2 ; E=0 ret date: ; Scrie date pe LCD mov P2, a setb P3.0 ; RS=1 clr P3.1 ; R/W=0 setb P3.2 ; E=1 acall delay_timer clr P3.2 ; E=0 se dezactiveaza pinul E ret	ADC: mov a, #8Dh ; Pozitionare cursor pe prima linie pozitia 13 acall instructiuni acall delay_timer mov r1, P1 ; Afisare temperatura setata mov a, r1 mov b, #6h ; $5/2^8$ div ab mov b, #0Ah div ab push b add a, #30H acall date acall delay_timer pop b mov a, b add a, #30h acall date acall delay_timer ; Afisare temperatura actuala mov a, #0DFh ; simbolul de la grade acall date acall delay_timer mov a, #0CDh ; Pozitionare cursor pe a doua linie pozitia 13 acall instructiuni acall delay_timer
-->	-->	-->

<pre> mov a, R0 mov b, #0Ah div ab push b add a, #30h acall date acall delay_timer pop b mov a, b add a, #30h acall date acall delay_timer mov a, #0DFh acall date acall delay_timer display_humidity: ; Calculeaza umiditatea in functie de temperatura actuala mov a, P1 ; Citeste temperatura actuala mov b, #0Ah subb a, b jc calc_high_humidity mov a, P1 subb a, #0Ah add a, #32h sjmp finish_humidity </pre> <p style="text-align: right;">--></p>	<pre> calc_high_humidity: mov a, #32h subb a, P1 add a, #50h sjmp finish_humidity finish_humidity: ; Converteste valoarea umiditatii in zeci si unitati mov b, #0Ah div ab mov r6, a ; Salveaza zecile mov r7, b ; Salveaza unitatile ; Pozitionare cursor pe a treia linie la pozitia 13 mov a, #9Dh acall instructiuni acall delay_timer ; Afisare zeci mov a, r6 add a, #30h acall date acall delay_timer ; Afisare unitati mov a, r7 add a, #30h acall date acall delay_timer </pre> <p style="text-align: right;">--></p>	<pre> ; Afisare simbol procent mov a, #25h acall date acall delay_timer ret HEART: MOV A, #40H ACALL instructiuni MOV A, #00H ACALL date MOV A, #0AH ACALL date MOV A, #1FH ACALL date MOV A, #1FH ACALL date MOV A, #0EH ACALL date MOV A, #04H ACALL date MOV A, #00H ACALL date MOV A, #00H ACALL date MOV A, #0DDH ; Set cursor position for heart character (C8H) ACALL instructiuni MOV A, #00H ; Display heart character ACALL date ACALL delay_timer </pre> <p style="text-align: right;">--></p>
---	---	--

<p>CARACTER:</p> <pre> MOV A, #48H ACALL instructiuni MOV A, #00H ACALL date MOV A, #0AH ACALL date MOV A, #0AH ACALL date MOV A, #0AH ACALL date MOV A, #11H ACALL date MOV A, #0EH ACALL date MOV A, #00H ACALL date MOV A, #00H ACALL date MOV A, #0DEH ; Set cursor position for custom character (CAH) ACALL instructiuni MOV A, #01H ; Display custom character ACALL date ACALL delay_timer OMULET: MOV A, #50H ACALL instructiuni MOV A, #0EH ACALL date MOV A, #0EH ACALL date --> </pre>	<pre> MOV A, #0EH ACALL date MOV A, #04H ACALL date MOV A, #1FH ACALL date MOV A, #04H ACALL date MOV A, #1FH ACALL date MOV A, #11H ACALL date MOV A, #0DFH ; Set cursor position for custom character (CAH) ACALL instructiuni MOV A, #02H ; Display custom character ACALL date ACALL delay_timer RET ; Control releu mov a, P1 ;Valoarea actuala mov R6, P0 ;Valoarea setata subb a, R6 jc start ; Se verifica daca temperatura setata este mai mare decat cea actuala sjmp oprire start: clr P3.7 ; Pornire releu sjmp gata oprire: setb P3.7 ;Oprire releu --> </pre>	<pre> ; Tastatura jnb P0.0, Buton1 ; Se verifica daca "+" este apasat jnb P0.1, Buton2 ; Se verifica daca "-" este apasat ljmp ADC Buton1: mov r3, #255 debouncing_1: jb P0.0, Buton1 djnz r3, debouncing_1 inc R0 b1: jnb P0.0, b1 ljmp ADC Buton2: mov r3, #255 debouncing_2: jb P0.1, Buton2 djnz r3, debouncing_2 dec R0 b2: jnb P0.1, b2 ljmp ADC delay_timer: mov TH1, #0EEh ; 5ms mov TL1, #00h setb TR1 ; Porneste Timer 1 d1: jnb TF1, d1 ; Asteapta overflow clr TF1 ; Opreste Timer 1 clr TR1 ; Reseteaza flag-ul Timer 1 ret gata: end </pre>
---	---	---

26. BIBLIOGRAFIE

- [1] [Termostatul de camera](#)
- [2] [Termistor](#)
- [3] [Senzori capacitivi](#)
- [4] [Senzori rezistivi](#)
- [5] [Senzori inductivi](#)
- [6] [Principii termocuplu](#)
- [7] [Principii traductoare](#)
- [8] [Datasheet LMT84LP](#)
- [9] [Datasheet AB13-130](#)
- [10] [Datasheet TMP36](#)
- [11] [Datasheet LM35](#)
- [12] [Datasheet TC1046](#)
- [13] [Datasheet MCP6001](#)
- [14] [Datasheet ADC0808](#)
- [15] [Despre LCD](#)
- [16] [Caractere](#)
- [17] [Datasheet LCD LM016L](#)
- [18] [Alte surse](#)
- [19] [Crearea caracterelor pe LCD](#)
- [20] [Microcontrolerul 8051](#)
- [21] <https://chat.openai.com/>
- [22] [Caractere speciale pe LCD utilizand microcontrolerul 8051](#)
- [23] [Youtube1](#)
- [24] [Youtube2](#)
- [25] [Youtube3](#)
- [26] [Youtube4](#)
- [27] [Youtube5](#)
- [28] [Releul](#)
- [29] [Tastatura](#)
- [30] [Senzorul de umiditate DHT11](#)