LUCRAREA 10

SISTEM DE AFIŞARE ALFANUMERICĂ PE AFIŞAJ LCD CONTROLAT CU MICROCONTROLER (2)
- descriere software-

```
//********************** PREZENTARE GENERALA *********************

//Proiectul este conceput pentru a putea afisa simboluri alfanumerice si/sau simboluri
//definite de utilizatori pe dipozitive de afisare LCD de tip matricial cu 5x7 puncte Seiko
//sau compatibile, in cazul de fata cu 2 linii si 24 de caractere/linie (tip L2432).
//Se utilizeaza o placa de dezvoltare HCS12 STARTER KIT (SofTec) cu microcontrolerul de 16
//biti MC9S12C32 (Motorola).
//Datele necesare afisarii si comenzilor sunt transmise modulului de afisare LCD prin
//intermediul portului B de la microcontroler.
//Pentru buna functionare a afisajului LCD mai sunt necesare 3 linii de control: RS (selectie
//registru), E (enable) si RW (Read/Write). Aceste linii de control sunt realizate cu ajutorul
//liniilor din portul A pentru BIT0, BIT1 si BIT2. Microcontrolerul permite controlul la nivel
//de bit a pinilor portului A.
//Pentru a simplifica scrierea programului si a asigura rezerve de dezvoltare pentru alte
//modalitati de afisare, se utilizeaza doua fisiere header:
//      control_LCD.h - contine toate codurile de control ale LCD necesare
//                      configurarii unui mod specific de functionare
//      caractere_speciale_LCD.h - contine codurile ASCII ale unor caractere
//                      relativ des utilizate (cum ar fi litere din afabetul
//                      grec, simboluri matematice) si care nu se regasesc
//                      in editorul de text folosit pentru a scrie programul
//                      sursa in C. De asemenea sunt date si codificarile
//                      matricei 5x7 pentru simboluri definte de utilizator
//                      cum ar fi de exemplu diactitice. Aceste simboluri
//                      definite de utilizator vor fi codificate si
//                     inregistrate in memoria CGRAM a afisajului LCD
//Programul principal va utiliza din fisierele header numai acele elemente necesare
//proiectului. Fisierele header se gasesc in directorul "Libraries" ale proiectului.
//Denumirile functiilor folosite (desi uneori sunt lungi) sunt alese astfel incat intelegerea
//programului sa fie foarte usoara. Pentru a intelege functionarea diferitelor functii si
//portiuni din software sunt introduse comentarii consistente.


//*****************************************************************
//************************** SOFTWARE *****************************
//*****************************************************************

#include <hidef.h>                    //definitii comune si macrouri
#include <mc9s12c32.h>                //informatii despre microcontroler
#include <caractere_speciale_LCD.h>   //coduri caractere speciale ASCII si matrici de puncte
                                      //caractere definite de utilizator
#include <control_LCD.h>              //cuvinte de comanda pentru display LCD

//-----------------------------------------------------------------
// codul ce urmeaza se plaseaza in zona NOT BANKED FLASH MEMORY
//-----------------------------------------------------------------

#pragma CODE_SEG __NEAR_SEG NON_BANKED

#pragma LINK_INFO DERIVATIVE "mc9s12c32"

//******************        definitii        *********************
#define RS_LCD   PORTA_BIT0   //bitul RS selectie registru LCD: ”L" - instructiune; "H" - data
#define E_LCD    PORTA_BIT2   //bitul "enable" pentru afisajul LCD
#define RW_LCD   PORTA_BIT1   //bitul RW selectie LCD: "L" - scrie registru LCD; "H" – citeste
                              //registru LCD
#define DATA_LCD PORTB        //in portul B se scriu datele pentru afisajul LCD

//*** OBSERVATIE *** In cazul in care se folosesc alte porturi, linii de port sau locatii de
//RAM extern pentru transferul spre display a datelor si comenzilor, se vor modifica aceste
//definitii si se va adapta conditiilor functia "initilizari_porturi"


//**************  declarare tipuri variabile  *******************
unsigned char linie,pozitie;  //linie   =linia 1 - 2 (pe care se scrie) a LCD-ului
                              //pozitie =pozitie 1-24 pe care se pozitioneaza cursorul
```

```
unsigned char caracter;       //caracter care va fi scris in DDRAM

unsigned int us;              //us reprezinta aproximativ nr de microsecunde in functia
                              //intarziere

unsigned char a1, a2, i1, s1, t1, L1, L2, L3, L4; //denumirile acestor variabile se aleg
                              //pentru a descrie caracterul diacritic sau grafic definit de
                              //utilizator
                              //in cazul de fata este vorba de diacriticile "a~", "a^", "i^",
                              //"s,","t," si de simbolurile grafice"   |","  ||"," ||",‟ |||";

unsigned char diacritic;      //variabila utilizata pentru lucrul cu simboluri definite (in
                              //principal diacritice, dar nu numai)

unsigned char diacritic_afisat; //precizeaza ce diacritic - conform codificarii de mai sus -
                              //se afiseaza

unsigned char pozitie_DDRAM;  //precizeaza pozitia ocupata de un diacritic sau simbol grafic
                              //in memoria DDRAM; pe baza acestei pozitii se face apelul si
                              //scrierea in DDRAM pentru afisare;


//************    declarare prototipuri functii    ***************
void initializare_LCD(void);  //functia de initializare afisaj LCD

void transmite_comanda_LCD(unsigned char comanda);
              //functia de transmitere a comenzii catre registrul de control LCD

void pozitionare_cursor(unsigned char linie, unsigned char pozitie);
              //functia de pozitionare a cursorul la linia si pozitia dorita

void trimite_sir_caractere_LCD(char *mesaj);
              //se transmite un sir de caractere ce se inscriu in memoria DDRAM

void scrie_DDRAM(unsigned char caracter);
              //scriere in registrul DDRAM al afisajului

void intarziere(unsigned int us);
              //functia intarziere programabila (us reprezinta aproximativ nr de
              //microsecunde)

void initializare_porturi_LCD(void);
              //initializarea porturilor folosite in aplicatie pentru comanda si transferul
              //de date LCD

void afisare_caracter_special_ASCII(unsigned char caracter_special, unsigned char linia,
unsigned char pozitia);
              //permite afisarea pe o pozitie precizata de utilizator prin numarul liniei
              //si al locului pe linie a caracterelor ASCII recunoscute de display

void scrie_diacritice_in_CGRAM(char matrice[8],unsigned char pozitie_DDRAM, unsigned char
*diacritic);   //functia asigura incarcarea in memoria CGRAM a displayului a matricilor de
              //puncte (format 5x8) pentru diacritice si caractere grafice definite de
              //utilizator; matricile cu imaginea caracterelor se gasesc sau se incarca in
              //prealabil in fisierul header "caractere_speciale_LCD.h"

void initializare_diacritice(void);
              //functia permite incarcarea memoriei CGRAM cu pana la 8 caractere diacritice
              //sau grafice definite de utilizator in softul de initializare a sistemului;
              //caracterele utilizate sunt memorate sub forma unor matrici de puncte (5x8) in
              //fiserul header "caractere_speciale_LCD";
              //daca este necesara utilizarea unui simbol diferit de cele din header,
              //utilizatorul isi poate construi matricea de definitie, ii da un nume adecvat
              //si o incarca in headerul "caractere_speciale_LCD.h";

void afisare_diacritice(unsigned char diacritic, unsigned char linia, unsigned char pozitia);
              //permite afisarea pe display a unui diacritic sau caracter grafic definit de
              //utilizator, cu precizarea numarului liniei si a pozitiei pe linie; caracterul
              //respectiv trebuie sa fie in prealabil definit prin matricea de puncte (5x8)
              //incarcata in memoria CGRAM; de asemenea caracterul trebuie activat pentru
              //utilizare prin introducerea  in functia "initializare_diacritice"

void afisare_mesaj(char L1[25], char L2[25]);
              //permite scrierea unui mesaj care sa umple complet cele doua linii ale
              //afisajului; cele doua siruri L1 si L2 reprezinta sirurile de caractere de pe
              //cele doua linii; se introduc direct, conform normelor din C, folosind " ' "
              //ce delimiteaza sirurile;
```

```
void afisare_un_caracter_LCD(char caracter_ASCII, unsigned char linie, unsigned char pozitie);
               //permite afisarea unui singur caracter ASCII recunoscut de editorul de text
               //folosit pentru scrierea programului in limbaj C;
               //se precizeaza caracterul dorit folosind " ' " (de exemplu '?'), numarul
               //liniei pe care se doreste sa se faca afisarea si pozitia pe linie, precizata
               //printr-un numar corespunzator



// ***************************************************************
// ***************************************************************
// ********************  functii auxilare  ***********************
// ***************************************************************
// ***************************************************************

//---------------------------------------------------------------
 void initializare_porturi_LCD(void)
//---------------------------------------------------------------
//*** OBSERVATIE *** Aceasta functie este demonstrativa. In aplicatii specifice se pot utiliza
//alte porturi, linii de port sau locatii de RAM extern pentru transferul comenzilor si
//dateleor catre display. In aceste cazuri se vor face modificarile necesare, atat in functie
//cat si in definitii.

 {  //initializare port B
    //portul B va fi folosit pentru transferul de date pe 8 biti pentru afisajul LCD

    PORTB=0x00;       //sterge port B
    DDRB=0xFF;        //configureaza PORTB ca iesire

               //Initializare portA
        //portul A va fi folosit pe linii de bit pentru controlul afisajului LCD - 3 pini
        //linii de control, conform definitiilor

    PORTA=0x00;      //sterge port A
    DDRA=0xFF;       //configureaza PORTA ca iesire

    }


// ***************************************************************
// ***************************************************************
// ***************      functii afisaj LCD      ******************
// ***************************************************************
// ***************************************************************


// XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
// XXXXXXXXXXXXXXXXXXXXX  initializare LCD  XXXXXXXXXXXXXXXXXXXXX
// XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

//---------------------------------------------------------------
void initializare_LCD(void)
//---------------------------------------------------------------
//*** OBSERVATIE *** Aceasta functie de intializare a displayului are un mare grad de
//generalizare, fiind valabila pentru foarte multe aplicatii (de exemplu in sisteme de
//masurare). Totusi, daca se doresc  si alte modalitati de lucru, se vor introduce comenzile
//necesare. Codurile ce pot fi folosite se gasesc in headerul "control_LCD.h". Mai jos este
//precizat modul de lucru al afisajului in conditiile precizate de functia "initializare_LCD"
//curenta.

//initializare afisaj LCD - conform prescriptiilor din catalogul SEIKO
//afisjul LCD va functiona conform initializarii descrise mai jos astfel:
// - mod de functionare pe 8 biti
// - initial afisajul si cursorul sunt stinse
// - registrele de date sunt sterse si cursorul este adus pe pozitia initiala (capat)
// - cursorul incrementeaza fara deplasarea afisajului
// - se activeaza afisarea si cursorul; cursorul nu este vizibil (se poate face)

{
  intarziere(15000);  //temporizare de 15000us inainte de activarea comenzilor

    transmite_comanda_LCD(FUNCTIONARE_PE_8_BITI_2_LINII_MATRICE_5x7);
       //cuvant de control 38H - mod de functionare pe 8 biti

intarziere(4000);   //operatiunea se repeta de inca 3 ori, cu temporizari
```

```
    transmite_comanda_LCD(FUNCTIONARE_PE_8_BITI_2_LINII_MATRICE_5x7);
       //reinscriere dupa temporizare de 4000us

  intarziere(100);     //temporizare de 100us

    transmite_comanda_LCD(FUNCTIONARE_PE_8_BITI_2_LINII_MATRICE_5x7);
                       //reinscriere cuvant de control

  intarziere(100);     //temporizare de 100us

    transmite_comanda_LCD(FUNCTIONARE_PE_8_BITI_2_LINII_MATRICE_5x7);
                       //reinscriere cuvant de control;
                       //sfarsit comanda mod 8 biti

  intarziere(100);     //temporizare 100us inainte de urmatorul tronson de comenzi

    transmite_comanda_LCD(STINGE_AFISAJ_SI_CURSOR);
                       //08h = comanda pentru stingere afisaj si cursor

  intarziere(100);     //temporizare 100us inainte de urmatorul tronson de comnenzi

    transmite_comanda_LCD(STERGE_AFISAJ);
                       //01h = comanda pentru stergere afisaj si aducere cursor pe pozitia
                       //initiala

  intarziere(100)      //temporizare 100us inainte de urmatorul tronson de comenzi

    transmite_comanda_LCD(ADUCERE_CURSOR_PE_POZITIA_INITIALA);
                       //02h = comanda pentru aducere cursor pe pozitia initiala si e absolut
                       //necesara dupa una din instructiunile: Cursor Home sau Display Clear

  intarziere(100);     //temporizare 100us inainte de urmatorul tronson de comenzi

    transmite_comanda_LCD(DEPLASARE_CURSOR_DREAPTA_DISPLAY_FIX);
                       //06h = comanda pentru incrementare cursor fara depalsarea afisajului

  intarziere(4000);    //temporizare 4000us inainte de urmatorul tronson de comenzi

    transmite_comanda_LCD(AFISAJ_ON);
                       //0EH(ca sa apara cursor) Afisajul si cursorul sunt ON,iar cursorul nu
                       //va clipi

      intarziere(1000);//temporizare de 1000us la sfarsitul initializarii afisajului

}


// XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
// XXXXXXXXXXXXXXXXXX  functii LCD de baza  XXXXXXXXXXXXXXXXXXXXX
// XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

//Aceste functii sunt valabile pentru orice aplicatie, avand caracter de functii de baza.
//Pentru buna lor functionare in aplicatii ce folosesc alte porturi, linii de port sau locatii
//RAM externe, trebuie sa se aiba grija sa se faca corect modificarile necesare in definitii
//si functia "initializare_porturi_LCD"

//---------------------------------------------------------------
void transmite_comanda_LCD(unsigned char comanda)
//---------------------------------------------------------------
//functia de transmitere a comenzii catre registrul de control LCD

{
  RS_LCD = 0;          //se selecteaza transfer "instructiune"
  RW_LCD = 0;          //se selecteaza "scriere"

    intarziere(2);
  E_LCD = 1;           //se activeaza circuitul LCD ("enable")
  DATA_LCD = comanda;  //se pune pe liniile magistralei de date LCD octetul "comanda"

    intarziere(2);
  E_LCD = 0;           //se transfera informatia de pe magistrala 'date' in circuitul LCD

    intarziere(2);

  DATA_LCD = 0x00;     //se sterg liniile magistralei 'date' LCD

}
```

```
//---------------------------------------------------------------
void scrie_DDRAM(unsigned char caracter)
//---------------------------------------------------------------
//scriere in registrul DDRAM al afisajului

{
  intarziere(100);     //cand se scriu siruri de caractere, intre doua scrieri trebuie sa fie
                       //o pauza de minim 100 us
  RS_LCD = 1;          //se selecteaza transfer "data"
  RW_LCD = 0;          //se selecteaza "scriere"

    intarziere(2);     //se cere o intarziere de 2 us
  E_LCD = 1;           //se activeaza circuitul LCD ("enable")
  DATA_LCD = caracter; //se pune pe liniile magistralei de date LCD octetul "caracter"

    intarziere(2);     //se cere o intarziere de 2 us
  E_LCD = 0;           //se transfera informatia de pe magistrala 'date' in circuitul LCD
  DATA_LCD = 0x00;     //se sterg liniile magistralei 'date' LCD
  RS_LCD = 0;          //se sterge linia de comanda RS

 }


//---------------------------------------------------------------
void pozitionare_cursor(unsigned char linie, unsigned char pozitie)
//---------------------------------------------------------------
//functia de pozitionare a cursorul la linia si pozitia dorita

{
  if(pozitie>24)       //se limiteaza pozitia maxima 24
    pozitie=24;
  if(linie==1)
     transmite_comanda_LCD(0x80+pozitie-1);
                       //se muta la inceputul liniei 1 (0x80)+pozitia dorita
  else
  if(linie==2)         //daca cursorul este pozitionat pe linia 2
     transmite_comanda_LCD(0xC0+pozitie-1);
                       //se muta la inceputul liniei 2 (0xC0)+pozitia dorita
}



//---------------------------------------------------------------
void trimite_sir_caractere_LCD(char *mesaj)
//---------------------------------------------------------------
//se transmite un sir de caractere ce se inscriu in memoria DDRAM

{
  volatile unsigned char temporar;
  while (mesaj[0] != '\0'     //atat timp cat nu este sfarsitul sirului
  {
    temporar = mesaj[0];      //se citeste un caracter din sir
    scrie_DDRAM(temporar);    //se scrie caracterul in DDRAM send character
    mesaj ++;                 //se trece la urmatorul caracter din sir

  }
}


//---------------------------------------------------------------
void scrie_diacritice_in_CGRAM(char matrice[8], unsigned char pozitie_DDRAM,
                               unsigned char *diacritic)
//---------------------------------------------------------------
//aceasta functie permite scrierea in memoria CGRAM a unui caracter diacritic definit de
//utilizator (de unde si numele functiei), dar si a altor simboluri grafice definite de
//utilizator; in general este vorba de caractere ce nu se regasesc in setul de caractere ASCII
//incarcare in memoria CGROM (fixa) a generatorului de caractere; simbolurile folosite de
//aceasta functie sunt "construite" de utilizator, in conformitate cu prescriptiile
//producatorului displayului; in headerul "caractere_speciale_LCD.h" sunt incluse codificarile
//pentru diacriticile din limba romana: "(a~) = (denumita in header "a1"), (a^) = (a2), (s,) =
//(s1), (t,) = (t1), (i^) = (i1), precum si simboluri grafice de forma unor dreptunghiuri
//negre de dimensiunile (in matricea de puncte) 1x7 (denumit "L1"), 2x7 (L2), 3x7 (L3) si 4x7
//(L4), aliniate la dreapta, care impreuna cu simbolurile codificate ACSII 'space' (0x0)
//(denumita "ALB" in headerul "caractere_speciale_LCD.h" si 'negru plin' (5x7) = (NEGRU) se
//folosesc la realizarea afisarii de tip bargraph; daca se doresc si alte simboluri, se vor
```

```
//defini in headerul "caractere_speciale_LCD.h" si se vor face cuvenitele comentarii de
//identificare in continuarea acestei zone; pentru identificarea caracterelor si a matricilor
//de puncte corespunzatoare s-a folosit urmatoarea regula: se atribuie o denumire simbolica
//caracterului respectiv (de exemplu "a1" pentru diacriticul "a~"); se construieste matricea
//de puncte necesara afisarii si se denumeste cu denumirea simbolica anterioara precedata de
//litera "m" (matrice), "ma1" pentru exemplul ales; cu aceasta denumire se introduce in
//headerul "caractere_speciale_LCD" matricea corespunzatoare; functia are urmatorii parametri:
//"char matrice[8]" ce corespunde matricei memorate in fisierul header caractere_speciale_LCD"
//caracterului dorit, "pozitie_DDRAM" este pozitia (intre 0 si 7) pe care se inscrie in DDRAM
//caracterul dorit si "diacritic" ce se refera la denumirea simbolica aleasa pentru caracterul
//respectiv; in timpul desfasurarii programului, un diacritic neutilizat pe afisajul curent
//poate fi inlocuit cu un altul prin suprascriere pe aceeasi pozitie de scriere in DDRAM

{
  char i;

  *diacritic = pozitie_DDRAM;


      for (i=0; i<8; i++)
      {
        transmite_comanda_LCD(0x40+(pozitie_DDRAM*8)+i);
        intarziere(100);
        scrie_DDRAM(matrice[i]);
        intarziere(100);
      }
}


//---------------------------------------------------------------------
void intarziere(unsigned int us)
//---------------------------------------------------------------------
//functia intarziere programabila
//variabila "us" va exprima aproximativ numarul dorit de microsecunde

{
  unsigned int i;      //variabila ce incrementeaza pana la "us"
  for(i=0;i<(3*us);i++);
                       //o bucla de incrementare dureaza 1000 ns (i=1) - 1 us
}


// XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
// XXXXXXXXXXXX  initializari specifice aplicatiei  XXXXXXXXXXXXXX
// XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

//---------------------------------------------------------------------
void initializare_diacritice(void)
//---------------------------------------------------------------------
//se introduc cel mult 8 diacritice si simboluri grafice speciale; denumirile se gasesc in
//headerul "caractere_speciale_LCD.h" si se introduc in ordine aleatoare, functie de
//necesitatile aplicatiei; daca pe timpul desfasurarii programului sunt necesare mai mult de 8
//diacritice sau caractere grafice, situatia se va gestiona in mod corespunzator prin
//programare prin suprascrierea diacriticelor ce nu sunt utilizate in afisajul curent; la
//revenirea in afisajul anterior se vor reintroduce diacriticele necesare.
//*** ATENTIE! *** Niciodata nu se pot afisa simultan pe display mai mult de 8 tipuri de
//caractere diacritice sau grafice definite de utilizator (inscrise in memoria CGRAM a
//displayului)!

//*** OBSERVATIE *** Functia de fata este doar un exemplu necesar programului (demonstrativ)
//curent. Pentru o aplicatie specifica se vor introduce de catre utilizator acele caractere
//diacritice sau grafice ncesare.

{
    scrie_diacritice_in_CGRAM(ma1,0,&a1);  //litera "a~"
    scrie_diacritice_in_CGRAM(ma2,1,&a2);  //litera "a^"
    scrie_diacritice_in_CGRAM(mi1,2,&i1);  //litera "i^"
    scrie_diacritice_in_CGRAM(ms1,3,&s1);  //litera "s,"
    scrie_diacritice_in_CGRAM(mt1,4,&t1);  //litera "t,"
    scrie_diacritice_in_CGRAM(mL1,5,&L1);  //simbol grafic "   |"
    scrie_diacritice_in_CGRAM(mL2,6,&L2);  //simbol grafic "  ||"
    scrie_diacritice_in_CGRAM(mL3,7,&L3);  //simbol grafic " |||"
}
```

```
// XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
// XXXXXXXXXXXXXXX  functii afisaj LCD in main  XXXXXXXXXXXXXXXXX
// XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

//Aceste functii sunt de un nivel inalt si faciliteaza scrierea rapida a afisajului cu toate
//categoriile de caractere: carcactere ASCII recunoscute de editorul cu care se scrie
//programul in limbaj C, mesaje cu astfel de caractere caractere ASCII speciale nerecunoscute
//de editorul de text, dar care se gasesc in memoria genratoare de caractere (CGROM) a
//displayului si cracatere speciale, definite de utilizator folosind memoria CGRAM, cum ar fi
//diacriticele si caracterele grafice. Programul demonstrativ descris in functia "main" arata
//cat de simplu, rapid si eficient se pot folosi aceste functii.


//-----------------------------------------------------------------
void afisare_un_caracter_LCD(char caracter_ASCII, unsigned char linia,
                             unsigned char pozitia)
//-----------------------------------------------------------------
//functia permite scrierea unui singur caracter pe o pozitie precizata a unei linii a
//displayului;
//caracterele transmise sunt din setul de baza ASCII recunoscute de editorul de text folosit
//la scrierea programului in limabj C;

{

    pozitionare_cursor(linia, pozitia);
                      //se pozitioneaza cursorul pe pozitia dorita pentru afisarea
                      //caracterului special ASCII
    scrie_DDRAM(caracter_ASCII);
                      //se scrie caracterul ASCII dorit

}

//-----------------------------------------------------------------
void afisare_mesaj(char L1[25], char L2[25])
//-----------------------------------------------------------------
//functia de afisare mesaj - necesara in gestionarea informatiilor se scriu mesaje pe cele
//doua linii ale afisajului (in main);
//dimensiunea sirului de caractere -25- corespunde aici celor 24 de caractere pe linie permise
//de afisajul L2432 cu care s-a dezvoltat proiectul;
//se poate dezvolta si o varianta care sa permita introducerea numarului de caractere pe o
//line ca parametru.

{
    pozitionare_cursor(1,1);
    trimite_sir_caractere_LCD(L1);
    pozitionare_cursor(2,1);
    trimite_sir_caractere_LCD(L2);
}


//-----------------------------------------------------------------
void afisare_caracter_special_ASCII(unsigned char caracter_special,
        unsigned char linia, unsigned char pozitia)
//-----------------------------------------------------------------
//functia afisare caractere speciale codificate ASCII;
//permite afisarea unor caractere speciale (litere grecesti, simboluri matematice, simbolul
//grade "°", etc. ce sunt codificate ASCII si se gasesc in memoria generatorului de caracatere
//CGROM a displayului) folosind codurile ASCII corespunzatoare.

{
    pozitionare_cursor(linia, pozitia);
                      //se pozitioneaza cursorul pe pozitia dorita pentru afisarea
                      //caracterului special ASCII
    scrie_DDRAM(caracter_special);
                      //se scrie caracterul ASCII dorit extras din CGROM
}

//-----------------------------------------------------------------
void afisare_diacritice(unsigned char diacritic_afisat,
         unsigned char linia, unsigned char pozitia)
//-----------------------------------------------------------------
//functia permite afisarea de caractere speciale definite de utilizator;
//permite afisarea unor caractere speciale (cractere diacritice - de unde si denumirea
//functiei,caractere grafice definite de utilizator, etc. ce NU sunt codificate ASCII);
//pentru aceasta, caracterele definite de utilizator trebuie mai intai incarcate in memoria
//generatorului de caracatere CGRAM folosind functia "initializare_diacritice" ce contine un
//numar de apeluri corespunzator al functiei "scrie_diacritice_in_CGRAM";
```

```
//functia are ca parametri: caracterul special ce se doreste a fi afisat notat 'diacritic' (se
//utilizeaza denumirea corespunzatoare simbolului asa cum figureaza in headerul
//caractere_speciale_LCD.h"), 'linia' si 'pozitia'- linia si pozitia afisajului pe care se
//doreste afisarea respectivului caracter;
//practic aceste simboluri speciale se introduc unul cate unul (nu avem inca construita o
//functie care sa permita introducerea de siruri de caractere speciale - idee buna pentru
//dezvoltari ulterioare!).

{
    pozitionare_cursor(linia, pozitia);
                               //se pozitioneaza cursorul pe pozitia dorita pentru afisarea
                               //caracterului diacritic

    scrie_DDRAM(diacritic_afisat);
                               //se scrie caracterul diacritic dorit extras din CGRAM
}


// ***************************************************************
// ***************************************************************
// ******************    programul principal   ******************
// ***************************************************************
// ***************************************************************

//---------------------------------------------------------------
void main(void)
//---------------------------------------------------------------
//acest program este demonstrativ si foloseste toate functiile de scriere de nivel inalt
//realizate in program

{
        //*** initializari sistem si display ***

    initializare_porturi_LCD();//initializare porturi necesare controlului afisajului LCD
    initializare_LCD();        //initializare afisaj LCD
    initializare_diacritice(); //se incarca in CGRAM diacriticele dorite


        //*** se afiseaza un mesaj pe intreg afisajul ***

    afisare_mesaj(" **SCRIERE DIACRITICE** ",
                  "iata!                   ");


        //*** se scrie caracterul diacritic "a~" in locul
        //ultimei litere "a" din cuvantul "iata" ***

    afisare_diacritice(a1,2,4);  //scrie "a~" pe linia 2, pozitia 4


        //*** se scriu diacritice pe diferite pozitii ale afisajului ***

    afisare_diacritice(L1,2,6);  //scrie "   |" pe linia 2, pozitia 6
    afisare_diacritice(L2,2,7);  //scrie "  ||" pe linia 2, pozitia 7
    afisare_diacritice(L3,2,8);  //scrie " |||" pe linia 2, pozitia 8

    afisare_diacritice(a1,2,12);  //scrie "a~" pe linia 2, pozitia 12
    afisare_diacritice(a2,2,13);  //scrie "a^" pe linia 2, pozitia 13
    afisare_diacritice(i1,2,14);  //scrie "i^" pe linia 2, pozitia 14
    afisare_diacritice(s1,2,15);  //scrie "s," pe linia 2, pozitia 15

    afisare_diacritice(t1,2,16);  //scrie "t," pe linia 2, pozitia 16
    afisare_diacritice(L3,2,17);  //scrie " |||" pe linia 2, pozitia 17
    afisare_diacritice(L2,2,18);  //scrie "  ||" pe linia 2, pozitia 18
    afisare_diacritice(L1,2,19);  //scrie "   |" pe linia 2, pozitia 19


        //*** se scrie un caracter ASCII din setul de baza ***

    afisare_un_caracter_LCD('?',2,11);  //scrie "?" pe linia 2, pozitia 11


        //*** se scriu caractere speciale ASCII aflate in memoria CGROM a displayului ***

    afisare_caracter_special_ASCII(NEGRU,2,9);    //linia 2, pozitia 9
    afisare_caracter_special_ASCII(INFINIT,2,20); //linia 2, pozitia 20
    afisare_caracter_special_ASCII(ALPHA,2,21);   //linia 2, pozitia 21
```

```
    afisare_caracter_special_ASCII(BETA,2,22);    //linia 2, pozitia 22
    afisare_caracter_special_ASCII(RADICAL,2,23); //linia 2, pozitia 23
    afisare_caracter_special_ASCII(MINUS1,2,24);  //linia 2, pozitia 24


        //*** bucla fara sfarsit ***

    for(;;)
    {
      asm nop;
    }

}
```

Fişierul header “control_LCD.h”

```
//Se utilizeaza la afisaje LCD matriciale tip 5x7, cu 1, 2 sau 4 linii - Seiko sau compatibile

//************************************************************
//************************************************************
//**
//**     Filename  : control_LCD.h
//**     Display   : LCD tip matricial 5x7 - Seiko sau compatibile
//**     FileFormat: V1.00
//**     DataSheet : Liquid Crystal Displays Application - Standard Character
//**                 Modules
//**                 Seiko Instruments GmbH 1998,AN No.SIG-CHMO9805A
//**     Date/Time : 22.07.2007, 11:49
//**     Abstract  : Acest header defineste toate cuvintele de comanda
//**                 permise de registrul de control al dispozitivelor
//**                 de afisare LCD de tip matricial 5x7 Seiko sau
//**                 compatibile.
//**                 De asemenea sunt descrise efectul comenzilor
//**                 si modul de codificare, precum si starea
//**                 liniilor de control.
//**                 Se precizeaza timpul necesar executarii diferitelor
//**                 instructiuni.
//**
//**     (c) Copyright AEMC Laboratories
//**     Facultatea de Electronica, Telecomunicatii si Tehnologia Informatiei
//**     Bulevardul Carol I nr 11
//**     700506 Iasi
//**     ROMANIA
//**     http      : www.etc.tuiasi.ro
//**     mail      : dimitriu@etc.tuiasi.ro
//**
//**     File-Format-Revisions:
//**      - 22.07.2007, V1.00 :
//**                - cuvinte de control pentru afisaje LCD de tip
//**                  matricial 5x7 Seiko sau compatibile
//**
//************************************************************
//************************************************************


//acest header permite configurarea displayului LCD


char locatie_CGRAM[8]={0x40,0x48,0x50,0x58,0x60,0x68,0x70,0x78};

char locatie_citire[8]={0x00,0x01,0x02,0x03,0x04,0x05,0x06,0x07};



//************************************************************
//***************** functia stergere afisaj ******************
//************************************************************

//sterge afisajul si aduce cursorul pe pozitia initiala (home) la adresa 0;
//la toate adresele DDRAM se scrie codul "spatiu" 20h, iar in contorul de adrese (AC) se
//inscrie adresa DDRAM 0, iar daca a fost mutat, displayul revine pe pozitia initiala:
//cursorul revine pe capatul din stanga al liniei 1, cu exceptia M4024;
//in cazul M4024, daca cursorul sau clipirea sunt pe linia 3 sau 4, atunci acesta revine la
//capatul din stanga al liniei 3;
```

```
//dupa executarea instructiunii stergere afisaj, atunci se seteaza bitul I/D = 1
//(incrementare) in functiile mod de intrare.

#define STERGE_AFISAJ                               0x01
        //01h = comanda pentru stergere afisaj si aducere cursor pe pozitia initiala;
        //necesita pentru executie 1.64 ms


//************************************************************
//******** functia aducere cursor pe pozitia initiala *********
//************************************************************

//aduce cursorul pe pozitia initiala (home);
//in contorul de adrese (AC) se seteaza adresa DDRAM 0;
//afisajul deplasat revine pe pozitia initiala (home) si continutul DDRAM nu se schimba;
//daca cursorul sau clipirea este ON, atunci el revine pe capatul din stanga al liniei 1, cu
//exceptia M4024;
//in cazul M4024, daca cursorul sau clipirea sunt pe linia 3 sau 4, atunci acesta revine la
//capatul din stanga al liniei 3.


//  RS  R/W      DB7 DB6 DB5 DB4  DB3 DB2 DB1 DB0
// | 0 | 0 |    | 0 | 0 | 0 | 0 || 0 | 0 | 1 | * |  * = indiferent

#define ADUCERE_CURSOR_PE_POZITIA_INITIALA        0x02
        //0x03 - are acelasi efect!
        //02h = comanda pentru aducere cursor pe pozitia initiala;
        //continutul DDRAM nu se schimba;
        //necesita pentru executie 1.64 ms


//************************************************************
//*************** functii setare mod intrare *****************
//************************************************************

//fixeaza directia deplasarii cursorului si daca informatia afisata pe display se deplaseaza
//atunci cand datele sunt scrise sau citite

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

//I/D: adresa DDRAM este incrementata sau decrementata cu 1 cand
//     un caracter cod este scris in sau citit din DDRAM;
//     acest lucru este valabil pentru scriere in citire din CGRAM.

//     Când I/D = 1, adresa este incrementata cu o unitate si cursorul
//     sau clipirea se deplaseaza spre dreapta.
//     Când I/D = 0, adresa este decrementata cu o unitate si cursorul
//     sau clipirea se deplaseaza spre stânga.


//S:   daca S = 1, intregul display este shiftat spre dreapta sau
//     stanga pentru scriere in DDRAM;
//     pozitia cursorului nu se schimba, numai displayul se muta;
//     nu se executa mutare display pentru citire din DDRAM.

//     cand S = 1 si I/D = 1, displayul se muta cu un digit spre stanga
//     dupa ce data a fost scrisa in DDRAM;
//     cand S = 1 si I/D = 0, displayul se muta cu un digit spre dreapta
//     dupa ce data a fost scrisa in DDRAM;
//     daca S = 0 displayul nu se muta.

#define DEPLASARE_CURSOR_STANGA_DISPLAY_FIX       0x04
        //04h = comanda pentru deplasare cursor catre stanga,
        //fara deplasarea informatiei pe display;
        //necesita pentru executie 40 us

#define DEPLASARE_CURSOR_SI_DISPLAY_STANGA        0x05
        //05h = comanda pentru deplasare cursor spre stanga
        //cu deplasarea spre stanga a informatiei pe display;
        //necesita pentru executie 40 us

#define DEPLASARE_CURSOR_DREAPTA_DISPLAY_FIX      0x06
        //06h comanda pentru deplasare cursor spre dreapta
        //deplasarea informatiei pe display;
        //necesita pentru executie 40 us
```

```
#define DEPLASARE_CURSOR_SI_DISPLAY_DREAPTA       0x07
        //07h = comanda pentru deplasare cursor spre dreapta
        //cu deplasarea spre dreapta a informatiei de pe display
        //necesita pentru executie 40 us



//************************************************************
//************** functii control afisaj ON/OFF ****************
//************************************************************

//asigura comutarea ON/OFF a intregului afisaj, a cursorului si controleaza clipirea
//cursorului

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

//D: cand D = 1, displayul este ON;
//   cand D = 0, displayul este OFF;
//   daca se foloseste D = 0, datele afisate raman in DDRAM;
//   ca urmare datele pot fi afisate din nou fixand D = 1.
//C: cand C = 1, cursorul este afisat;
//   cand C = 0, cursorul nu este afisat;
//   cursorul este afisat pe linia de puncte de sub fonturi.
//B: cand B = 1, caracterul de pe pozitia cursorului incepe
//   sa clipeasca;
//   cand B = 0, caracterul nu clipeste.
//   pentru a clipi, toate punctele negre si caracterul sunt
//   comutate la fiecare circa 0,4 secunde pentru o frecventa
//   a oscilatorului de 250 kHz;
//   cursorul si clipirea pot fi active in acelasi timp.

//    C = 1 (afisare cursor)         B = 1 (clipire)
//
//     ***                            ***         *****
//    *                              *            *****
//    *                              *            *****
//    * ***                          * ***  <-->  *****
//    *   *                          *   *        *****
//    *   *                          *   *        *****
//     ****                           ****        *****
//    ***** <-- cursor               *****        *****
//
//    5x7 puncte                         5x7 puncte
//

#define STINGE_AFISAJ_SI_CURSOR                   0x08
               //08h = comanda pentru stingere afisaj si cursor;
               //display = off; cursor = off; clipire = off;
               //necesita pentru executie 40 us

#define CLIPIRE                                   0x09
               //09h = comanda clipire; cursorul este dezactivat
               //display = off; cursor off; clipire = on;
               //necesita pentru executie 40 us

#define AFISARE_CURSOR_FARA_CLIPIRE               0x0A
               //0Ah = comanda afisare cursor; nu avem clipire;
               //display = off; cursor = on; clipire = off;
               //necesita pentru executie 40 us

#define AFISARE_CURSOR_CU_CLIPIRE                 0x0B
               //0Bh = comanda afisare cursor; cursorul clipeste
               //display = off; cursor = on; clipire = on;
               //necesita pentru executie 40 us

#define AFISAJ_ON                                 0x0C
               //0Ch = afisajul este activat;
               //cursorul este dezactivat;
               //nu avem clipire;
               //display = on; cursor = off; clipire = off;
               //cursorul nu apare
               //necesita pentru executie 40 us

#define AFISAJ_SI_CURSOR_ON_FARA_CLIPIRE          0x0E
               //0Eh = afisajul si cursorul sunt ON; nu avem clipire;
               //display = on; cursor = on; clipire = off;
```

```
               //cursorul apare, dar nu clipeste
               //necesita pentru executie 40 us

#define AFISAJ_SI_CURSOR_ON_CU_CLIPIRE           0x0F
               //0Fh = afisajul si cursorul sunt ON, iar cursorul va clipi;
               //cursorul apare si clipeste
               //necesita pentru executie 40 us

//************************************************************
//***************** mutare display/cursor *********************
//************************************************************

//aceste functii asigura deplasarea cursorului si mutarea afisajului fara modificarea
//continutului memoriei DDRAM;
//pozitia cursorului corespunde cu continutul contorului de adrese (AC);
//aceasta instructiune este folositoare pentru corectarea sau refacerea continutului
//afisajului, deoarece pozitia cursorului sau a displayului se poate modifica fara a scrie sau
//citi datele afisate;
//pentru un display cu doua linii, cursorul este deplasat de pe pozitia digitului 40 (adresa
//DDRAM 27) a liniei 1 pe pozitia 1 a liniei 2;
//informatiile afisate pe liniile 1 si 2 sunt mutate in timp ce comanda "mutare display"
//deplaseaza orizontal continutul fiecarei linii;
//totusi continutul liniei 1 nu este deplasat pe linia 2 si nici continutul liniei 2 nu este
//deplasat pe linia 1;


//*** NOTA *** M1641 functioneaza intern ca un display de 8 caractere x 2 linii, L1614 ca
//dispaly cu 32 caractere x 2 linii, L2014 ca display cu 40 caractere x 2 linii si M4024 ca
//doua displayuri cu 40 caractere x 2 linii. vezi capitolul "locatii de adresa" din
//documentatie

//  RS  R/W      DB7 DB6 DB5 DB4  DB3 DB2 DB1 DB0
// | 0 | 0 |    | 0 | 0 | 0 | 1 ||S/C|R/L| * | * |   * = indiferent

//  S/C  R/L                  Functionare
// ------------------------------------------------------------------------
//   0    0  pozitia cursorului este mutata spre stanga (contorul de adrese
//           este decrementat cu 1)
//   0    1  pozitia cursorului este mutata spre dreapta (contorul de adrese
//           este incrementat cu 1)
//   1    0  intregul display se muta spre stanga impreuna cu cursorul
//   1    1  intregul display se muta spre dreapta impreuna cu cursorul

#define MUTA_CURSOR_STANGA                       0x10
               //10h = cursorul se muta spre stanga;
               //au acelasi efect si codurile 0x11, 0x12 si 0x13
               //necesita pentru executie 40 us

#define MUTA_CURSOR_DREAPTA                      0x14
               //14h = cursorul se muta spre dreapta;
               //au acelasi efect si codurile 0x15, 0x16 si 0x17
               //necesita pentru executie 40 us

#define MUTA_DISPLAY_STANGA                      0x18
               //18h = se muta displayul si cursorul spre stanga;
               //au acelasi efect si codurile 0x19, 0x1A si 0x1B
               //necesita pentru executie 40 us

#define MUTA_DISPLAY_DREAPTA                     0x1C
               //1Ch = se muta displayul si cursorul spre stanga;
               //au acelasi efect si codurile 0x1D, 0x1E si 0x1F
               //necesita pentru executie 40 us

//************************************************************
//******************** setare functionare ********************
//************************************************************
//"setare functionare" fixeaza lungimea de date a interfetei (4 sau 8 biti paralel), numarul
//de linii ale displayului si tipul de font al caracterului (5x7 sau 5x10);

//  RS  R/W      DB7 DB6 DB5 DB4  DB3 DB2 DB1 DB0
// | 0 | 0 |    | 0 | 0 | 1 | DL|| N | F | * | * |   * = indiferent

//DL: lungimea datei in interfata
//    cand DL = 1, lungimea datei este fixata la opt biti (BD7 la DB0)
//    cand DL = 0, lungimea datei este fixata la patru biti (DB7 la DB4);
//    cu o interfata de 4 biti, mai intai se transfera cei 4 biti
//    superiori, apoi cei 4 biti inferiori;
```

```
//N:  cand N = 1, serviciul este 1/16;
//    cand N = 0, serviciul este 1/8 sau 1/11;
//F:  fonturile pentru caractere
//    cand F = 1, fontul pentru caractere este fixat la matrice de 5x10
//    puncte
//    cand F = 0, fontul pentru caractere este fixat la matrice de 5x7 puncte
//    daca N este fixat la valoarea 1, F devine "indiferent"

//  ---------------------------------------------------------------------------
// | N | F | numar linii afisate| font caracter | serviciu |   modul LCD      |
//  ---------------------------------------------------------------------------
// | 0 | 0 |          1         |      5x7      |   1/8    |       ---        |
//  ---------------------------------------------------------------------------
// | 0 | 1 |          1         |     5x10      |   1/11   |       ---        |
//  ---------------------------------------------------------------------------
// | 1 | * |          2         |      5x7      |   1/16   | 1641,M1632,L1642 |
// |   |   |                    |               |          | L1652,L1614,L2012|
// |   |   |                    |               |          | L2022,L2014,L2432|
// |   |   |                    |               |          | L4042,M4024      |
//  ---------------------------------------------------------------------------

//o instructiune de setare functie trebuie sa fie executata inainte de
//toate instructiunile, cu exceptia citire flag ocupat / adresa.

#define FUNCTIONARE_PE_4_BITI_1_LINIE_MATRICE_5x7     0x20
               //20h = comanda pentru mod de functionare pe 4 biti;
               //display 1 linie; duty 1/8; matrice 5x7
               //necesita pentru executie 40 us

#define FUNCTIONARE_PE_4_BITI_1_LINIE_MATRICE_5x10    0x24
               //24h = comanda pentru mod de functionare pe 4 biti
               //display 1 linie; duty 1/11; matrice 5x10
               //necesita pentru executie 40 us

#define FUNCTIONARE_PE_4_BITI_2_LINII_MATRICE_5x7     0x28
               //0x2C are acelasi efect
               //28h = comanda pentru mod de functionare pe 4 biti
               //display 2 linii; duty 1/16; matrice 5x7
               //necesita pentru executie 40 us

#define FUNCTIONARE_PE_4_BITI_2_LINII_MATRICE_5x7   0x2C
               //2Ch = comanda pentru mod de functionare pe 4 biti
               //dispaly 2 linii; duty 1/16; matrice 5x7
               //necesita pentru executie 40 us

#define FUNCTIONARE_PE_8_BITI_1_LINIE_MATRICE_5x7     0x30
               //30h = comanda pentru mod de functionare pe 8 biti
               //display 1 linie; duty 1/8; matrice 5x7
               //necesita pentru executie 40 us

#define FUNCTIONARE_PE_8_BITI_1_LINIE_MATRICE_5x10    0x34
               //34h = comanda pentru mod de functionare pe 8 biti
               //display 1 linie; duty 1/11; matrice 5x10
               //necesita pentru executie 40 us

#define FUNCTIONARE_PE_8_BITI_2_LINII_MATRICE_5x7     0x38
               //0x3C are acelasi efect
               //38h = comanda pentru mod de functionare pe 8 biti
               //display 2 linii; duty 1/16; matrice 5x7
               //necesita pentru executie 40 us

//#define FUNCTIONARE_PE_8_BITI_2_LINII_MATRICE_5x7   0x3C
               //3Ch = comanda pentru mod de functionare pe 8 biti
               //dispaly 2 linii; duty 1/16; matrice 5x7
               //necesita pentru executie 40 us



//************************************************************
//******************** fixare adresa CGRAM ********************
//************************************************************

//fixarea adresei CGRAM este exprimata binar prin AAAAAA si se transfera contorului de adrese
//(AC);
//data scrisa sau citita de procesor este in sau de la CGRAM;
```

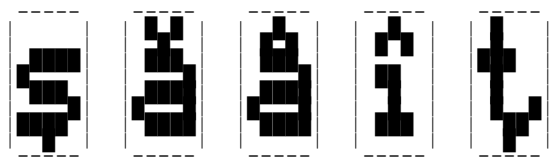
```
//  RS  R/W      DB7 DB6 DB5 DB4  DB3 DB2 DB1 DB0
// | 0 | 0 |    | 0 | 1 | A | A || A | A | A | A |
//                        |                   |
//                        |                   |->bit de semnificatie minima
//                        |
//                        |-> bit de semnificatie maxima


//prima adresa CGRAM este 40h, cand AAAAAA = 000000


#define FIXARE_ADRESA_INFERIOARA_CGRAM                0x40
               //40h = comanda de fixare a adresei inferioare CGRAM



//************************************************************
//******************** fixare adresa DDRAM ********************
//************************************************************

//fixarea adresei DDRAM este exprimata binar prin AAAAAAA si se
//transfera contorului de adrese (AC);
//data scrisa sau citita de procesor este in sau de la DDRAM;
//cand N = 0 (display cu 1 linie), adresele sunt de la 00h la 40h;
//cand N = 1 (display cu 2 linii: M1632, L1642, L1652, L2012,
//L2022, L2432, L4042), adresele folosite pentru linia 1 (AAAAAA)
//sunt de la 00h la 27h, iar cele pentru linia 2 sunt de la 40h la 67h;
//pentru M1641, L1614, L2024 si M2024, vezi sectiunea "Locatii de
//adresa" din documentatie.

//  RS  R/W      DB7 DB6 DB5 DB4  DB3 DB2 DB1 DB0
// | 0 | 0 |    | 1 | A | A | A || A | A | A | A |
//                    |                       |
//                    |                       |->bit de semnificatie minima
//                    |
//                    |-> bit de semnificatie maxima

//prima adresa DDRAM este 80h, cand AAAAAAA = 0000000

#define FIXARE_ADRESA_INFERIOARA_DDRAM                0x80
               //80h = comanda de fixare a adresei inferioare DDRAM


//************************************************************
//***************** citire flag ocupat/adresa *****************
//************************************************************

//  RS  R/W      DB7 DB6 DB5 DB4  DB3 DB2 DB1 DB0
// | 0 | 1 |    | BF| 1 | A | A || A | A | A | A |
//                        |                   |
//                        |                   |->bit de semnificatie minima
//                        |
//                        |-> bit de semnificatie maxima

//se citeste flagul "ocupat", indicand daca modulul lucreaza intr-o
//operatiune interna din cauza instructiunii anterioare;
//cand BF = 1, modulul executa o operatiune interna si instructiunea
//urmatoare nu poate fi acceptata pana cand BF nu devine 0;
//daca BF = 0, instructiunea urmatoare poate fi acceptata;
//ca urmare, asigurati-va ca BF = 0 inainte de a scrie o noua instructiune;
//valoarea binara AAAAAA a contorului de adrese (AC) este citita in
//acelasi moment ca si flagul "ocupat";
//adresele contorului de adrese sunt folosite atat pentru CGRAM cat si
//pentru DDRAM, iar intructiunea de fixare adresa de dinaintea executarii
//acestei instructiuni va determina daca adresa este de la CGRAM sau DDRAM;



//************************************************************
//*************** scrie date in CGRAM sau DDRAM ***************
//************************************************************

//se scrie o data binara de 8 biti DDDDDDDD in CGRAM sau DDRAM;
//adresa CGRAM sau DDRAM fixata inaintea acestei instructiuni
//selecteaza zona de RAM specifica;
//dupa operatiunea de scriere, adresa este incrementata sau
//decrementata in mod automat, dupa cum s-a fixat modul de intrare;
```

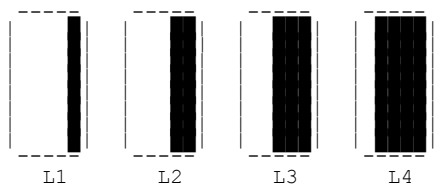
```
//in acest fel, modul de intrare setat determina daca displayul
//se shifteaza sau nu dupa operatiune de scriere;



//  RS  R/W      DB7 DB6 DB5 DB4  DB3 DB2 DB1 DB0
// | 1 | 0 |    | D | D | D | D || D | D | D | D |
//                |                             |
//                |                             |->bit de semnificatie minima
//                |
//                |-> bit de semnificatie maxima

//************************************************************
//************* citeste date din CGRAM sau DDRAM **************
//************************************************************

//se citeste o data binara de 8 biti DDDDDDDD din CGRAM sau DDRAM;
//adresa CGRAM sau DDRAM fixata inaintea acestei instructiuni selecteaza zona de RAM
//specifica;
//daca nu s-a executat nicio instructiune de fixare a adresei inaintea unei instructiuni de
//citire, prima data citita nu este valabila;
//data este in mod normal citita la timpul al doilea daca se executa citiri consecutive;
//pentru DDRAM, daca s-a executat o instructiune de deplasare cursor chiar inainte de citrea
//DDRAM, nu mai este necesar sa se execute o instructiune de fixare adresa, deoarece
//instructiunea deplasare cursor realizeaza acest lucru;
//dupa o operatiune de citire, adresa este in mod automat incrementata sau decrementata,
//conform cu modul de intrare fixat, dar displayul nu este shiftat conform modului de intrare
//ales

//*** NOTA ***
//contorul de adrese (AC) este in mod automat incrementat sau decrementat cu 1 conform cu
//modul de intrare selectat dupa ce se executa o instructiune de scriere in CGRAM sau DDRAM;
//daca o instructiune de citire se executa imediat dupa o astfel de instructiune, informatia
//din RAM specificata de contorul de arese (AC) nu este extrasa;
//data corecta este extrasa in urmatoarele conditii:

//** se executa o instructiune de fixare a adresei imediat inainte
//   de o instructiune de citire;
//** pentru DDRAM, o instructiune de deplasare a cursorului se
//   executa imediat inainte de o instructiune de citire;
//** a doua instructiune sau urmatoarele se executa succesiv
//   cu o instructiune de citire;

//  RS  R/W      DB7 DB6 DB5 DB4  DB3 DB2 DB1 DB0
// | 1 | 1 |    | D | D | D | D || D | D | D | D |
//                |                             |
//                |                             |->bit de semnificatie minima
//                |
//                |-> bit de semnificatie maxima
```

Fişierul header “caractere_speciale_LCD.h”

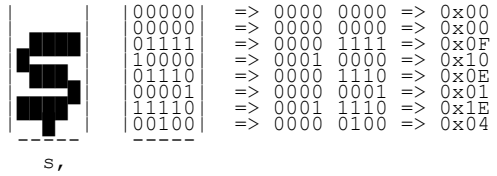
```
// Se utilizeaza la afisaje LCD matriciale tip 5x7 - Seiko sau compatibile

//************************************************************
//************************************************************
//**
//**     Filename  : caractere_speciale_LCD.h
//**     Display   : LCD tip matricial 5x7 - Seiko sau compatibile
//**     FileFormat: V1.00
//**     DataSheet : Liquid Crystal Displays Application - Standard Character
//**                 Modules
//**                 Seiko Instruments GmbH 1998,AN No.SIG-CHMO9805A
//**     Date/Time : 22.07.2007, 11:15
//**     Abstract  : Acest header permite utilizarea caracterelor speciale
//**                 ASCII (litere grecesti, simboluri matematice) care nu
//**                 sunt recunoscute de editorul folosit pentru scrierea
//**                 fisierelor in limbaj C/C++, a diacriticilor pentru limba
//**                 romana (litere mici), precum si a unor caractere
//**                 definite de utilizator,in acest caz dreptunghiuri de
//**                 4x7,3x7, 2x7 si 1x7 utilizate pentru afisaj de tip
//**                 bargraph
//**
//**     (c) Copyright AEMC Laboratories
//**     Facultatea de Electronica si Telecomunicatii
```

```
//**     Bulevardul Carol I nr 11
//**     700506 Iasi
//**     ROMANIA
//**     http      : www.etc.tuiasi.ro
//**     mail      : dimitriu@etc.tuiasi.ro
//**
//**     File-Format-Revisions:
//**      - 22.07.2007, V1.00 :
//**               - caractere grecesti si simboluri matematice
//**                 din setul ASCII - dar nepreluate de editor;
//**                 se gasesc in CG ROM
//**               - caractere diacritice in limba romana -
//**                 se incarca in CG RAM
//**               - caractere casuta neagra de latimi diferite -
//**                 se folosesc pentru afisaj tip bargraph;
//**                 se incarca in CG RAM
//**
//*************************************************************
//*************************************************************


//*************************************************************
//******** caractere grecesti si simboluri matematice *********
//******** din setul ASCII - dar nepreluate de editor *********
//***************** se gasesc in CG ROM ***********************
//*************************************************************

char GRADE   = 0xDF;  //codul ASCII pentru simbolul "°" (grade)
char MICRO   = 0xE4;  //codul ASCII pentru simbolul "µ" (micro)
char ALPHA   = 0xE0;  //codul ASCII pentru simbolul "α" (alfa)
char BETA    = 0xE2;  //codul ASCII pentru simbolul "ß" (beta)
char EPSILON = 0xE3;  //codul ASCII pentru simbolul "ε" (epsilon)
char THETA   = 0xF2;  //codul ASCII pentru simbolul "θ " (theta)
char MIU     = 0xE4;  //codul ASCII pentru simbolul "µ" (miu)
char OMEGA   = 0xF4;  //codul ASCII pentru simbolul "Ω" (omega)
char SIGMAm  = 0xE5;  //codul ASCII pentru simbolul "σ" (sigma mic)
char RO      = 0xE6;  //codul ASCII pentru simbolul "ρ" (ro)
char SIGMAM  = 0xF6;  //codul ASCII pentru simbolul "Σ" (sigma mare)
char PI      = 0xF7;  //codul ASCII pentru simbolul "π" (pi)
char RADICAL = 0xE8;  //codul ASCII pentru simbolul "radical"
char MINUS1  = 0xE9;  //codul ASCII pentru simbolul "-1" (putere -1)
char INFINIT = 0xF3;  //codul ASCII pentru simbolul "infinit"
char NEGRU   = 0xFF;  //codul ASCII pentru simbolul "celula neagra"
char ALB     = 0xFE;  //codul ASCII pentru simbolul "celula alba"


//*************************************************************
//*********** caractere diacritice in limba romana ************
//***************** se incarca in CG RAM **********************
//*************************************************************

char ms1[8]={0x00,0x00,0x0F,0x10,0x0E,0x01,0x1E,0x04}; //litera "s,"
char ma1[8]={0x0A,0x04,0x0E,0x01,0x0F,0x11,0x0F,0x00}; //litera "a~"
char ma2[8]={0x04,0x0A,0x0E,0x01,0x0F,0x11,0x0F,0x00}; //litera "a^"
char mi1[8]={0x04,0x0A,0x00,0x0C,0x04,0x04,0x0E,0x00}; //litera "i^"
char mt1[8]={0x08,0x08,0x1C,0x08,0x08,0x09,0x06,0x04}; //litera "t,"


//*************************************************************
//******** caractere casuta neagra de latimi diferite *********
//********** se folosesc pentru afisaj tip bargraph ***********
//***************** se incarca in CG RAM **********************
//*************************************************************

char mL1[8]={0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01}; //caracter "   |"
char mL2[8]={0x03,0x03,0x03,0x03,0x03,0x03,0x03,0x03}; //caracter "  ||"
char mL3[8]={0x07,0x07,0x07,0x07,0x07,0x07,0x07,0x07}; //caracter " |||"
char mL4[8]={0x0F,0x0F,0x0F,0x0F,0x0F,0x0F,0x0F,0x0F}; //caracter "||||"



//*************************************************************
//*********** caractere utilizator - forma grafica ************
//***************** se incarca in CG RAM **********************
//*************************************************************

/*
```





Exemplu de codificare:



se observa ca pentru completarea cuvintelor de 8 biti s-au completat la stanga pe fiecare linie cate trei biti cu valoarea “0”

*/