```
;mod1.a51
;pentru modul1

;numara, trimite, receptioneaza pe 485 un caracter si
afiseaza
;in ascii si in cod hexa acest caracter

;pentru test la 485, se pune afisaj lcd 1x16

$DEBUG
$PAGELENGTH (65535)
$PAGEWIDTH (110)
$XREF

CODESEG     SEGMENT CODE      ;definesc zona de
program (cod)
STIVA SEGMENT     DATA        ;definesc zona de stiva

      RSEG  STIVA
      DS    32          ;rezerv spatiu pt. stiva 32
octeti

;-------program principal--------------------------

      RSEG  CODESEG
      USING 0           ;folosesc bank 0 de registrii

        ORG     00H             ;adresa de inceput a
prog la placa
        LJMP    INCEPUT

        ORG     03H             ;External interrupt 0
        RETI

        ORG     0BH             ;Timer 0 interrupt
        RETI

        ORG     13H             ;External interrupt 1
        RETI

        ORG     1BH             ;Timer 1 interrupt
        RETI

        ORG     23H             ;Serial interrupt
        RETI
;-----------------------------------------------
INCEPUT:
      MOV   SP,#STIVA-1 ;initializare stiva
```

```
        CALL    INIHARD         ;initializare hard
placa
      CALL  INIAFI            ;initializare afisaj
LCD
      CALL  INISER            ;initializare seriala

;-------------------------------------------------
      CALL  DELAY_500MS
      CALL  BEEP
      CALL  DELAY_500MS
      CALL  TESTAFI           ;verific prezenta
afisajului LCD
      JZ    BINE
RAU:
;     CALL  BEEP        ;in caz de eroare intra in
bucla infinita
      CALL  DELAY_500MS
      JMP   RAU
BINE:
;-------------------------------------------------
      CALL  CLRAFI
      MOV   DPTR,#TEXT7 ;afiseaza ok
      CALL  AFITEXT
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  CLRAFI
      CALL  DELAY_500MS
;-------------------------------------------------
      MOV   C,T1        ;verific bateria
      JC    BINE1
RAU1:
      MOV   DPTR,#TEXT4 ;eroare baterie
      JMP   BATEND
BINE1:
      MOV   DPTR,#TEXT3 ;baterie ok
BATEND:
      CALL  AFITEXT
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  CLRAFI
      CALL  DELAY_500MS
;-------------------------------------------------
```

```
      MOV   C,INT0            ;verific semnalul PFAIL
      JC    BINE2
RAU2:
      MOV   DPTR,#TEXT5 ;pfail low
      JMP   PFEND
BINE2:
      MOV   DPTR,#TEXT8 ;pfail high
PFEND:
      CALL  AFITEXT
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  CLRAFI
      CALL  DELAY_500MS
;-------------------------------------------------

      MOV   DPTR,#0FFFFH      ;verific memoria RAM
32k (testare distructiva)
MEMO1:
      INC   DPTR
      MOV   A,#00H            ;scriu 00
      MOVX  @DPTR,A
      MOVX  A,@DPTR
      JNZ   RAU3
      MOV   A,DPH
      CLR   C
      SUBB  A,#7FH
      JNZ   MEMO1
      MOV   A,DPL
      CLR   C
      SUBB  A,#0FFH
      JNZ   MEMO1

      MOV   DPTR,#0FFFFH
MEMO2:
      INC   DPTR
      MOV   A,#0FFH           ;scriu FF
      MOVX  @DPTR,A
      MOVX  A,@DPTR
      INC   A
      JNZ   RAU3
      MOV   A,DPH
      CLR   C
      SUBB  A,#7FH
      JNZ   MEMO2
      MOV   A,DPL
      CLR   C
```

```
      SUBB  A,#0FFH
      JNZ   MEMO2
BINE3:
      MOV   DPTR,#TEXT1 ;RAM ok
      JMP   MEMO3
RAU3:
      MOV   DPTR,#TEXT2 ;RAM eroare
MEMO3:
      CALL  AFITEXT
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  DELAY_500MS
      CALL  CLRAFI
      CALL  DELAY_500MS

;-------------------------------------------------

      MOV   R2,#2       ;afiseaza : si *
      CALL  CURSOR
      MOV   R2,#3AH
      CALL  WRITEAFID

      MOV   R2,#5
      CALL  CURSOR
      MOV   R2,#3AH
      CALL  WRITEAFID

      MOV   R2,#9
      CALL  CURSOR
      MOV   R2,#2AH
      CALL  WRITEAFID

;-------------------------------------------------
PROG:

      CALL  SWPC
      MOV   R3,#0
BUCLA1:

      CALL  DELAY_500MS
      CALL  DELAY_500MS

      MOV   A,R3
      SETB  P1.7
      CALL  TXCHAR
```

```
        CALL    DELAY_2MS
        CALL    DELAY_2MS
        CALL    DELAY_2MS
        CALL    DELAY_2MS
        CALL    DELAY_2MS
        CALL    DELAY_2MS
        CALL    DELAY_2MS
        CALL    DELAY_2MS
        CALL    DELAY_2MS


        CLR     P1.7
        CALL    RXCHAR

        MOV     R4,A
        MOV     R2,#1
        CALL    CURSOR
        MOV     A,R4
        MOV     R2,A
        CALL    WRITEAFID
        MOV     R2,#3
        CALL    CURSOR
        MOV     A,R4
        MOV     R2,A
        CALL    HEXAS1
        MOV     R2,A
        CALL    WRITEAFID
        MOV     R2,B
        CALL    WRITEAFID

        INC     R3

        JMP     BUCLA1

PROGEND:
;-------------------------------------------------


TEXT0:      DB    00H
TEXT1:      DB    '32k RAM OK',00H
TEXT2:      DB    '32k RAM EROARE',00H
TEXT3:      DB    'BATERIE OK',00H
TEXT4:      DB    'BATERIE EROARE',00H
TEXT5:      DB    'PFAIL LOW',00H
TEXT6:      DB    'RAM AFI EROARE',00H
TEXT7:      DB    'AFISAJ OK',00H
TEXT8:      DB    'PFAIL HIGH',00H

TEXT10:     DB    '****************************',00H
```

```
TEXT11:     DB    'Unitate centrala MODUL1 ver
1.0',00H
TEXT12:     DB    '*****************************',00H
TEXT13:     DB    'Introduceti o litera',00H

;-------------------------------------------------
;******************** subroutine ******************
;-------------------------------------------------

HEXAS:                          ;conversie hex <10H (0-
F) in digit ascii
                          ;intra A si iese A
      CLR   C
      SUBB  A,#0AH
      JNC   HAS1
      ADD   A,#3AH
      RET
HAS1:
      ADD   A,#41H
      RET
;-------------------------------------------------

HEXAS1:                         ;converteste un hex
(00-FF) in doi
                          ;octeti ascii tiparibili
                          ;intra R2 si iese A si B
      MOV   A,R2
      ANL   A,#0FH
      CALL  HEXAS
      MOV   B,A
      MOV   A,R2
      SWAP  A
      ANL   A,#0FH
      CALL  HEXAS
      RET

;-------------------------------------------------

RXCHAR:                         ;receptie caracter de
la seriala
                          ;fara intrerupere (in A)
      JNB   RI,RXCHAR
      MOV   A,SBUF
      CLR   RI
      RET
;-----------------------------------------
;-----------------------------------------
TXCHAR:                         ;trimite caracter din A
pe seriala
      JNB   TI,TXCHAR
```

```
      CLR   TI
      MOV   SBUF,A
      RET
;-----------------------------------------
TXTEXT:                         ;trimite un text din
mem prog
                          ;(max 256 caractere) pe
seriala cu adr de
                          ;inceput in DPTR, txt trebuie
sa se termine
                          ;cu 00H. Textul e urmat de
CRLF
                          ;afecteaza A,R3,
      MOV   R3,#0
TXX1:
      MOV   A,R3
      MOVC  A,@A+DPTR
      JZ    TXX2
      CALL  TXCHAR
      INC   R3
      JMP   TXX1
TXX2:
      MOV   A,#0DH
      CALL  TXCHAR
      MOV   A,#0AH
      CALL  TXCHAR

      RET

;-----------------------------------------

AFITEXT:                  ;afiseaza un text cu adr de
inceput in DPTR
                          ;textul trebuie sa se termine
cu 00h
                          ;afecteaza A,R1,R2,R3
      MOV   R3,#0
AFIT1:
      MOV   A,R3
      MOV   R2,A
      CALL  CURSOR
      MOV   A,R3
      MOVC  A,@A+DPTR
      JZ    AFIT2
      MOV   R2,A
      CALL  WRITEAFID
      INC   R3
      JMP   AFIT1
AFIT2:
      RET
```

```
;-----------------------------------------

READCEAS:               ;citeste un registru al
ceasului RTC
                        ;intra R1=00h - unitati
secunde 0-9
                        ;       01h - zeci secunde 0-
5
                        ;       02h - unitati minute
0-9
                        ;       03h - zeci minute 0-5
                        ;       04h - unitati ore 0-9
                        ;       05h - zeci ore 0-2
                        ;       06h - unitati ziua 0-
9
                        ;       07h - zeci ziua 0-3
                        ;       08h - unitati luna 0-
9
                        ;       09h - zeci luna 0-1
                        ;       0Ah - unitati anul 0-
9
                        ;       0Bh - zeci anul 0-9
                        ;       0Ch - ziua din
saptamana 0-6
                        ;iese B valoarea ASCII citita
a registrului
                        ;afecteaza A
      SETB  P1.0
IRECE:
      MOVX  A,@R1
      MOV   B,A
      MOVX  A,@R1
      CLR   C
      SUBB  A,B
      JNZ   IRECE
      CLR   P1.0
      ANL   B,#00001111B
      ORL   B,#00110000B
      RET
;-----------------------------------------

WRITECEAS:              ;scrie un registru al
ceasului RTC
                        ;intra R1 - aceeasi
semnificatie ca la read
                        ;intra R2 - valoarea
registrului ASCII
                        ;afecteaza A
      SETB  P1.0
      MOV   A,R2
```

```
      MOVX  @R1,A
      CLR   P1.0
      CALL  READCEAS
      MOV   A,B
      CLR   C
      SUBB  A,R2
      JNZ   WRITECEAS
      RET
;-----------------------------------------

SWPC:                              ;pune seriala pe
directia calculatorului
      SETB  P1.3
      SETB  P1.4
        RET
;-------------------------------------------------

SWPRINT:                           ;pune seriala pe
directia imprimantei
      SETB  P1.3
      CLR   P1.4
        RET
;-------------------------------------------------

SWMF:                    ;pune seriala pe directia
memfis
      CLR   P1.3
      SETB  P1.4
      RET
;-------------------------------------------------

INIHARD:                           ;initializare hard
placa MODUL1
                         ;implicit este pe SWPC
                         ;
      CLR   P1.0         ;acces ram ext
      CLR   P1.7         ;485 pe receptie
      RET
;-----------------------------------------

INISER:                          ;initializare seriala
9600
      MOV   TMOD,#20H
      MOV   TH1,#0FDH
      SETB  TR1
      MOV   SCON,#52H

      RET
```

```
;-------------------------------------------------

BEEP:                              ;scoate un beep la
buzer
                          ;afecteaza R6,R7
      CLR   P1.6
      CALL  DELAY_2ms
      SETB  P1.6
      CALL  DELAY_2ms
      CLR   P1.6
      CALL  DELAY_2ms
      SETB  P1.6
      CALL  DELAY_2ms
      CLR   P1.6
      CALL  DELAY_2ms
      SETB  P1.6
        RET
;-------------------------------------------------

INIAFI:                            ;initializare afisaj
intern 1x16
                          ;afecteaza A,R1,R6,R7
                          ;se ruleaza la pornire
alimentare

      setb  p1.0
      mov   r1,#10h
      mov   a,#38h
      movx  @r1,a
      call  delay_2ms
      call  delay_2ms

      movx  @r1,a
      call  delay_100us

      movx  @r1,a
      call  delay_100us

      movx  @r1,a
      call  delay_100us

      mov   a,#06h
      movx  @r1,a
      call  delay_100us

      mov   a,#0ch
      movx  @r1,a
      call  delay_100us

      mov   a,#01h
```

```
      movx  @r1,a
      call  delay_2ms

      mov   a,#80h
      movx  @r1,a
      call  delay_100us
      clr   p1.0

        RET
;-------------------------------------------------

CLRAFI:                         ;sterge afisajul
      MOV   R2,#01H
      CALL  WRITEAFIC
      CALL  DELAY_2MS
      RET
;-------------------------------------------------

READAFIC:               ;citeste starea afisajului
(busy flag=ACC.7)
                        ;iese A (ACC.7)
                        ;afecteaza R1
      SETB  P1.0
      MOV   R1,#12H
      MOVX  A,@R1
      CLR   P1.0
      RET
;-------------------------------------------------

WRITEAFIC:              ;scrie o comanda din R2 la
afisaj
                        ;intra R2 comanda
                        ;afecteaza A,R1
      CALL  READAFIC
      MOV   C,ACC.7
      JC    WRITEAFIC
      SETB  P1.0
      MOV   R1,#10H
      MOV   A,R2
      MOVX  @R1,A
      CLR   P1.0
      RET
;-------------------------------------------------

WRITEAFID:              ;trimite un caracter (data)
din R2 la afisaj
                        ;intra R2 = caracterul ascii
de afisat
                        ;afecteaza A,R1
      CALL  READAFIC
```

```
      MOV   C,ACC.7
      JC    WRITEAFID
      SETB  P1.0
      MOV   R1,#11H
      MOV   A,R2
      MOVX  @R1,A
      CLR   P1.0
      RET
;-------------------------------------------------

READAFID:                ;citeste un caracter (data)
de la afisaj in A
                         ;iese A = codul ascii citit
                         ;afecteaza R1
      CALL  READAFIC
      MOV   C,ACC.7
      JC    READAFID
      SETB  P1.0
      MOV   R1,#13H
      MOVX  A,@R1
      CLR   P1.0
      RET
;-------------------------------------------------

CURSOR:                      ;muta cursorul
afisajului
                         ; la pozitia din R2 (0 - 15)
                         ;intra R2 = poz. cursor
                         ;afecteaza A,R1
      MOV   A,R2
      CLR   C
      SUBB  A,#8
      JNC   LINIA2
      MOV   A,#80H
      ADD   A,R2
      MOV   R2,A
      CALL  WRITEAFIC
      JMP   ENDCURSOR
LINIA2:
      MOV   A,#0B8H
      ADD   A,R2
      MOV   R2,A
      CALL  WRITEAFIC
ENDCURSOR:
      RET
;-------------------------------------------------

TESTAFI:                 ;scrie si citeste ceva in
memoria afisaj
```

```
                        ;daca totul e ok returneaza
zero in acumulator
                        ;iese A
                        ;afecteaza R1,R7

      SETB  P1.0

      MOV   A,#0D0H
      MOV   R1,#10H
      MOVX  @R1,A
      CALL  DELAY_100US

      MOV   A,#00H
      MOV   R1,#11H
      MOVX  @R1,A
      CALL  DELAY_100US

      MOV   A,#0D0H
      MOV   R1,#10H
      MOVX  @R1,A
      CALL  DELAY_100US

      MOV   R1,#13H
      MOVX  A,@R1
      CALL  DELAY_100US
      JNZ   ERTEST
;-------------------------------------
      MOV   A,#0D0H
      MOV   R1,#10H
      MOVX  @R1,A
      CALL  DELAY_100US

      MOV   A,#0FFH
      MOV   R1,#11H
      MOVX  @R1,A
      CALL  DELAY_100US

      MOV   A,#0D0H
      MOV   R1,#10H
      MOVX  @R1,A
      CALL  DELAY_100US

      MOV   R1,#13H
      MOVX  A,@R1
      CALL  DELAY_100US

      ADD   A,#1
ERTEST:
      CLR   P1.0
      RET
```

```
;-----------------------------------------------

DELAY_100us:                    ;asteapta 100us
      MOV   R7,#23              ;pt FQ=11,059MHz ,
Tosc=90,424ns
XXX1: NOP
      ;1ciclu_masina=12*Tosc=1,085us
      NOP               ;2 nop + 1 djnz = 4 cicluri
masina
      DJNZ  R7,XXX1
      ;delay=4*23*1ciclu_masina=99.8us
      RET
;-----------------------------------------------

DELAY_2ms:              ;asteapta 2ms
      MOV   R6,#19              ;incarca R6 cu valoarea
19
XXX2: ACALL DELAY_100us ;call absolut (short)
      DJNZ  R6,XXX2             ;decrement register and
jump if not zero
      RET
;-------------------------------------------------

DELAY_500ms:                    ;asteapta 0,5 secunde
      MOV   R5,#250             ;incarca R5 cu valoarea
250
XXX4: ACALL DELAY_2ms   ;call absolut (short)
      DJNZ  R5,XXX4             ;decrement register and
jump if not zero
      RET
;-----------------------------------------------
      END
```