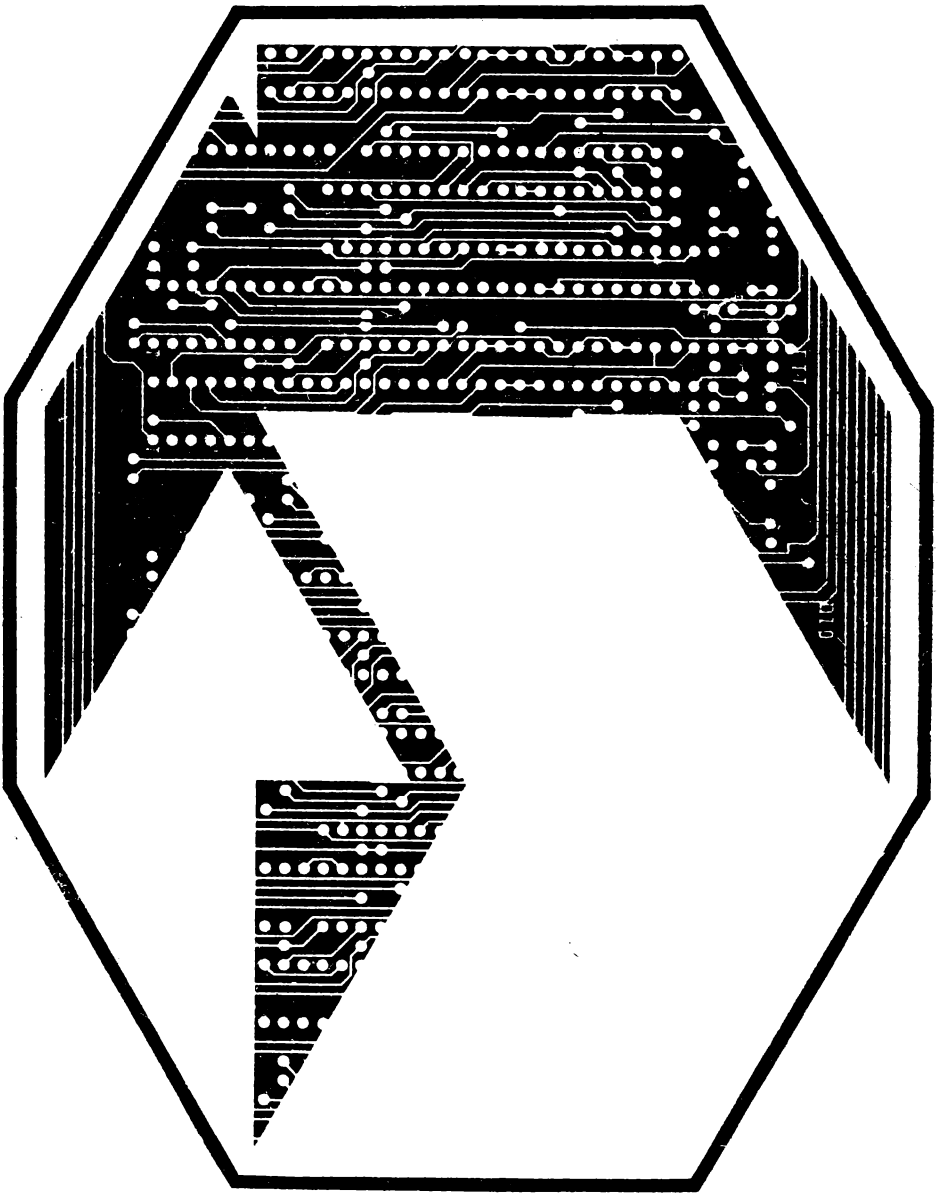
ice felix®
DATA PROCESSING SYSTEMS

HC 85

MANUAL TEHNIC

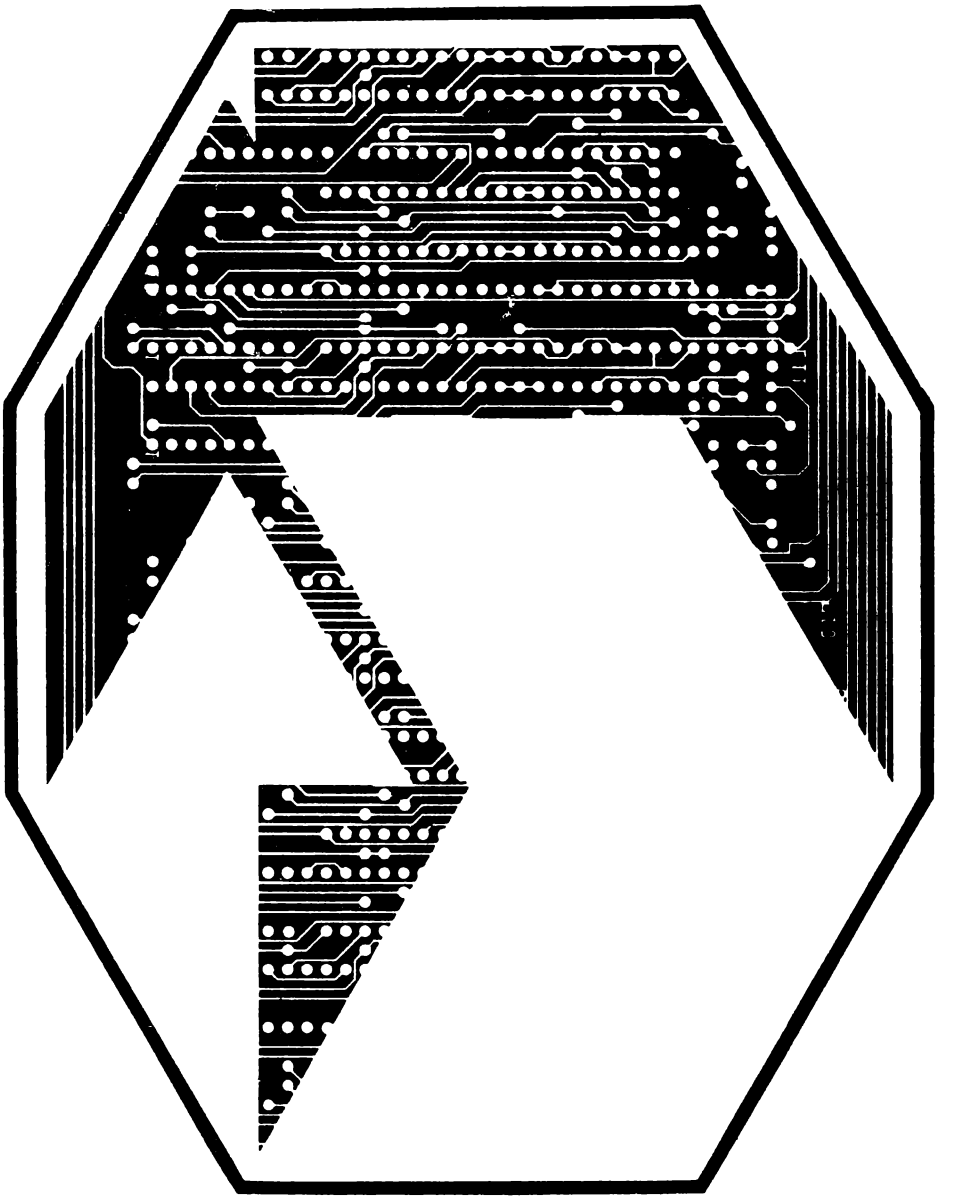


întreprinderea de calculatoare electronice

ice felix®
DATA PROCESSING SYSTEMS

HC 85

MANUAL TEHNIC



întreprinderea de calculatoare electronice

EVIDENȚA MODIFICĂRILOR		
Revizia	Data	Document modificator și locații afectate
A	Iun. '86	Prima editare
B	Feb.'87	Editarea a doua

Codul acestei publicatii este :

Comentarii sau observatii asupra acestei publicatii se pot trimite direct la :

INTREPRINDEREA DE CALCULATOARE ELECTRONICE

Compartimentul Pregatirea Documentatiei pentru Beneficiari

Str. Ing. GEORGE CONSTANTINESCU Nr. 2 - 78009 - Bucuresti 2

sau folosind fisa pentru comentarii aflata la sfirsitul acestei publicatii.

IMPRIMAT IN I.C.E.

CUPRINS

1. PREZENTARE GENERALA SI INSTALARE

1.1 PREZENTARE GENERALA

Microcalculatorul HC-85 este construit cu microprocesorul Z80A. Sistemul dispune de 64 Ko memorie, din care 16 Ko sint de tip EPROM si contin interpretorul BASIC iar 48 Ko sint de tip RAM. Ca dispozitiv de afisare este utilizat un televizor alb-negru sau color, iar ca memorie externa un casetofon audio obisnuit.

Manualul ajuta pe utilizator sa realizeze si sa ruleze programe scrise in limbajul BASIC sau in limbaj masina.

Calculatorul este prevazut cu conectori pentru legarea echipamentelor periferice. Amplasarea conectorilor este data in fig. 1.1 a sau 1.1 b in noua varianta de carcasa.

Conectorul de alimentare este folosit la alimentarea calculatorului de la alimentatorul de +9V. Puterea consumata de calculator este de aprox. 15 W.

Conectorul de televizor este folosit pentru interconectarea cu un televizor alb-negru sau color de tip PAL. Televizorul trebuie acordat pe canalul 10.

Conectorul video permite conectarea oricarui tip de monitor alb-negru sau color (PAL, RGB). Asignarea pinilor este data in capitolul 4.

Conectorii de joystick permit cuplarea a doua joystick-uri de tip Sinclair.

Conectorul pentru extensie (edge-conectorul) foloseste la legarea extensiilor (disc, interfata seriala, PROM, etc.).

Conectorul de casetofon este folosit pentru incarcarea programelor de pe caseta si salvarea lor pe caseta. Se recomanda folosirea casetelor de tip C-60. Casetofonul folosit poate fi orice tip de casetofon audio. Se poate folosi de asemenea orice tip de magnetofon.

Calculatorul HC-85 se livreaza impreuna cu urmatoarele:

- prezentul manual de utilizare
- alimentator + 9V DC
- cablu pentru televizor
- cablu pentru casetofon

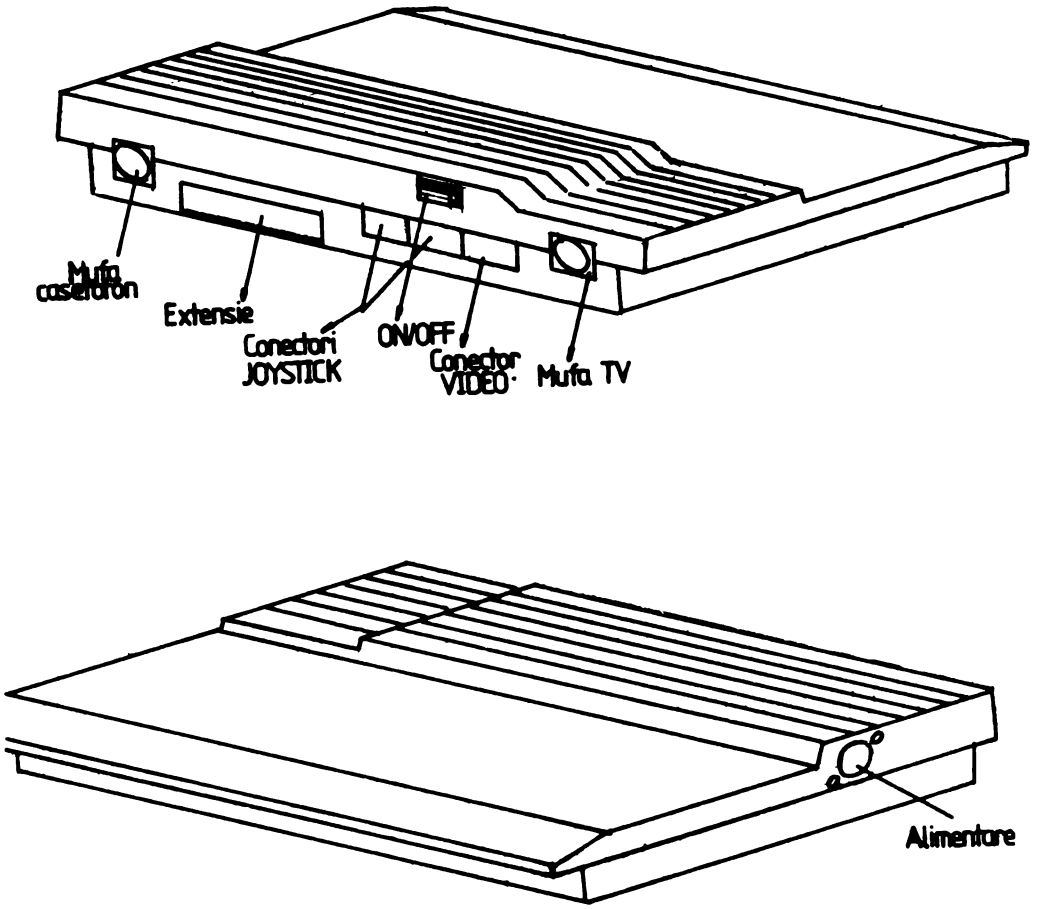

Mufa
casetofon
Extensie
Conectori
JOYSTICK
ON/OFF
Conector
VIDEO
Mufa TV
Alimentare

Figura 1.1 a

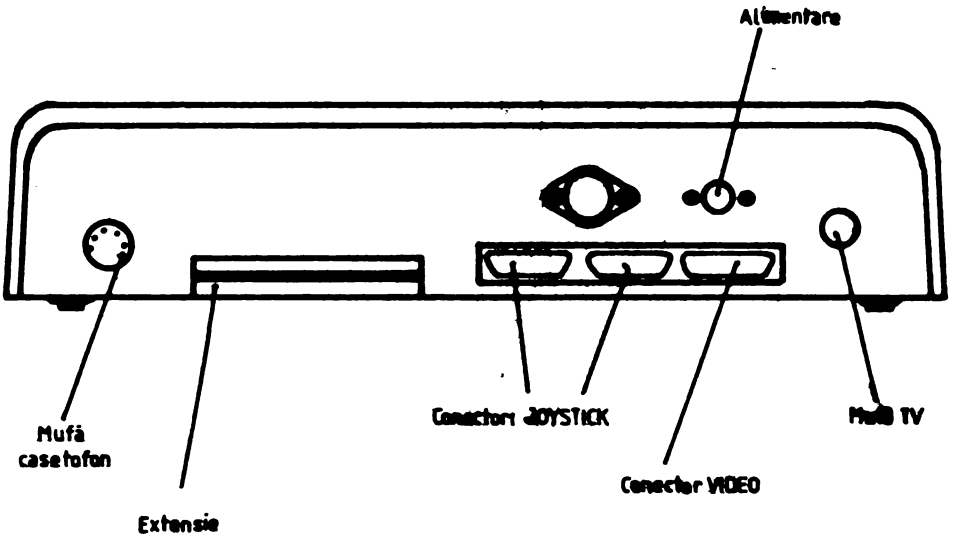

Alimentare
Mufă
casetofon
Extensie
Conector JOYSTICK
Conector VIDEO

Figura 1.1 b

1.2 INSTALARE

Calculatorul se alimenteaza prin intermediul alimentatorului de +9V de la reteaua de curent alternativ de 220V. Pentru evitarea pericolului de accidentare calculatorul trebuie sa fie alimentat numai de la prize prevazute cu legatura la pamint.

Punerea sub tensiune si oprirea calculatorului:

a. se introduce cablul de televizor atit in televizor cit si in calculator.

b. se acordeaza televizorul pe canalul 10.

c. se introduce cablul de casetofon (daca urmeaza a se folosi si casetofonul).

d. se introduc si eventualele extensii sau cabluri video.

e. se introduce alimentatorul in priza de 220V.

f. se introduce conectorul alimentatorului in conectorul de alimentare de +9V.

g. se regleaza televizorul din butonul de acord fin pina cind imaginea devine clara si stabila.

h. daca pe ecran nu apare mesajul

HC - 85 I.C.E. Felix

se apasa cu ajutorul unui creion pe intrerupatorul de RESET aflat in dreapta tastei 0 sub nivelul tastaturii; intrerupatorul de RESET se poate folosi ori de cite ori se doreste reinitializarea calculatorului.

i. in cazul televizoarelor color se regleaza culorile in asa fel incit sa avem litere negre pe fond alb.

j. pentru oprire operatiile a÷f vor fi executate in ordine inversa.

2. ELEMENTE DE PROGRAMARE SI EDITARE

2.1 UTILIZAREA TASTATURII

Tastatura calculatorului HC-85 este similara cu a unei masini de scris; literele si cifrele sint in aceleasi pozitii cu exceptia literelor Q, Z si M. Tastatura cuprinde simboluri simple (litere, numere, etc.) si compuse (cuvinte cheie, nume de functii, etc.) care sint introduse printr-o singura actionare si nu prin tastarea caracter cu caracter. Pentru a obtine toate functiile si comenzile, unele taste au pina la sase semnificatii diferite, selectionabile prin actionarea tastei corespunzatoare simultan cu una din tastele CAPS SHIFT sau SYMBOL SHIFT si in functie de modul de lucru al calculatorului.

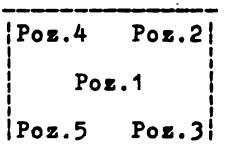


Fig. 2.1

HC-85 are cinci moduri de lucru :

1. Modul K (keyword-cuvint cheie) apare atunci cind se asteapta o comanda sau o linie de program (altceva decit date de intrare). Aceasta se intimpla la inceput de linie, imediat dupa un THEN, sau dupa caracterul ":" (ce separa instructiuni aflate pe aceasi linie). In modul K, daca nu sint utilizate shift-urile, tasta urmatoare va fi interpretata ca un cuvint cheie (pozitia 3 din figura 2.1 pentru taste alfanumerice) sau ca o cifra (pozitia 1 din fig. 2.1 pentru taste numerice).

2. Modul L (letters-litere) alterneaza cu modul K. Simbolul principal scris pe o tasta apare pe ecran prin simpla actionare a acesteia. In cazul unei litere, ea va apare ca litera mica. Atit in modul L cit si in modul K, actionarea simultana a lui SYMBOL SHIFT si a unei taste numerice va fi interpretata drept caracterul din pozitia 2 (vezi fig. 2.1). CAPS SHIFT cu o tasta numerica se va interpreta ca functia de control din pozitia 4 (vezi fig. 2.1). Actionarea unei taste literale simultan cu CAPS SHIFT in modul de lucru K nu are nici un efect, iar in modul L produce conversia literelor mici in litere mari.

3. Modul C (capitals-majuscule) este o varianta a modului L, in care scrierea se face cu majuscule. CAPS LOCK determina trecerea din modul L in modul C si invers.

4. **Modul E** (extended-extins) este utilizat pentru a obtine caractere noi, in special alte comenzi. Acest mod se obtine prin actionarea simultana a ambelor shift-uri cind pe ecran apare cursorul **E**. Iesirea din modul extins se face automat dupa actionarea tastei dorite . In acest mod apasarea unei taste literale genereaza caracterul sau simbolul de pe pozitia 4 (fig. 2.1) daca e apasata singura si caracterul sau simbolul de pe pozitia 5 daca e apasata impreuna cu una din tastele SHIFT. Apasarea unei taste numerice genereaza o comanda daca e apasata impreuna cu SYMBOL SHIFT si o secventa de control a culorii daca e apasata singura.

5. **Modul G** (graphics-grafic) se obtine prin actionarea tastelor CAPS SHIFT si 9. Anularea acestui mod de lucru se face actionind din nou tastele CAPS SHIFT si 9 sau numai tasta 9. 0 tasta numerica va da un mozaic grafic predefinit (in afara tastelor 0 si 9) si orice tasta literala in afara de V, W, X, Y si Z va genera un simbol grafic definit de utilizator.

Daca o tasta este apasata mai mult de 2-3 secunde, aceasta va incepe sa se repete.

Tastele apasate apar in partea de jos a ecranului, fiecare caracter fiind inserat pe locul cursorului. Cursorul poate fi mutat la stinga cu <-- (CAPS SHIFT si 5) sau la dreapta cu --> (CAPS SHIFT si 8). Caracterul din stinga cursorului poate fi sters cu **DELETE** (CAPS SHIFT si 0).

La inscrierea simbolurilor pe tastatura au fost folosite urmatoarele prescurtari:

RAND	in loc de	**RANDOMIZE**
BRGT	in loc de	**BRIGHT**
INV	in loc de	**INVERSE**
CR	in loc de	**ENTER**
CS	in loc de	**CAPS SHIFT**
SS	in loc de	**SYMBOL SHIFT**
SCR$	in loc de	**SCREEN$**
CONT	in loc de	**CONTINUE**

2.2 MODUL DE AFISARE

Ecranul afisare are 24 de linii, fiecare cu 32 de caractere.

Ecranul are doua parti.Partea de sus de 22 linii e folosita pentru listarea instructiunilor sau a rezultatelor programului. Cind aceasta parte este plina calculatorul face "scroll". Pentru a putea vedea toate liniile, calculatorul se opreste si apare mesajul **scroll ?**. Apasarea tastelor **N**, SPACE sau STOP va intrerupe programul si va afisa mesajul:

```
D    BREAK - CONT  repeats
```

Orice alta tasta determina calculatorul sa faca scroll. Partea de jos a ecranului este folosita pentru comenzi de intrare, linii de program, tiparirea datelor de intrare, cit si pentru mesaje.

2.3 PROGRAME, LINII DE PROGRAM SI EDITAREA PROGRAMELOR UTILIZIND EDIT SI SAGETILE, RUN, LIST, GO TO, CONTINUE, INPUT, NEW, REM, PRINT, STOP IN INPUT DATA, BREAK

Limbajul BASIC admite doua tipuri de instructiuni: numerotate si nenumerotate. Instructiunile nenumerotate sint executate imediat dupa apasarea tastei CR. Instructiunile numerotate sint stocate ca linii de program. Numerele de linie trebuie sa fie intregi, intre 1 si 9999. Listarea si executia unui program se fac prin ordonarea programului dupa numerele de linie. De aceea este indicat ca la scrierea unui program sa se pastreze spatii intre numerele a doua linii consecutive, dind astfel posibilitatea inserarii cu usurinta de linii noi. O linie de program poate contine una sau mai multe instructiuni; separarea instructiunilor dintr-o linie se face cu caracterul :.

Cursorul > indica linia curenta asupra careia se pot face modificari sau dupa care se pot insera alte linii. De obicei, cursorul se afla pe ultima linie introdusa, dar pozitia lui poate fi deplasata in sus sau in jos prin apasarea simultana a tastei CAPS SHIFT si a sagetilor.

In continuare vor fi prezentate exemple de programe in care sint trecute in revista citeva instructiuni BASIC, punindu-se accentul pe facilitatile de editare ale sistemului.

Exemplul 1. Sa se tipareasca suma a doua numere.
Dupa ce se vor introduce liniile (in ordinea mentionata):

```
20 PRINT a
10 LET a=10
```

se constata ca programul se tipareste pe ecran in permanenta ordonat dupa numarul de linie.

Pina acum s-a introdus primul numar. Pentru a-l introduce pe al doilea, se scrie linia:

```
15 LET b=15
```

Pentru tiparirea sumei, este necesar ca linia 20 sa aiba forma:

```
20 PRINT a+b
```

S-ar putea rescrie linia, dar este mai usor sa se faca uz de facilitatile EDIT. Pentru aceasta se coboara cursorul de la linia 15 la linia 20, actionind tasta ↓. In continuare se actioneaza tasta EDIT; in partea de jos a ecranului va apare o copie a liniei curente (in exemplul prezentat linia 20). Se actioneaza tasta --> pina cind cursorul > se deplaseaza la sfirsitul liniei si apoi se introduc +b (fara CR).

Ultima linie a ecranului va arata acum astfel:

```
20 PRINT a+b
```

Cu CR, vechea linie 20 va fi inlocuita cu cea noua.

Se executa acest program utilizind RUN si CR; ca urmare pe ecran va apare afisat rezultatul operatiei a+b. Apasind din nou RUN si CR programul este executat identic. Dupa terminarea executiei programului ramine inregistrata in memorie ultima valoare a fiecarei variabile din program. Ele pot fi vizualizate printr-o instructiune PRINT neetichetata. Aceasta operatie este utila la depanarea programului.

Pentru a sterge ultima linie a ecranului se utilizeaza EDIT. Se introduce o succesiune de caractere (fara CR) care vor fi sterse folosind una dintre metodele:

1. actionarea tastei DELETE pina cind linia este stearsa in intregime.
2. actionarea tastei EDIT; pe ultima linie a ecranului apare o copie a liniei curente. Cu CR acum, linia curenta ramine nemodificata, iar ultima linie a ecranului este stearsa.

Presupunem ca se introduce din greseala linia:

```
12 LET b=8
```

Ea va putea fi stearsa scriind:

```
12              (cu CR desigur).
```

Se observa ca a disparut cursorul programului. Daca se actioneaza ↑, cursorul va apare la linia 10, in timp ce daca se actioneaza ↓ va apare la linia 15. Se scrie:

```
12              (si CR)
```

Din nou cursorul programului va fi ascuns intre liniile 10 si 15. Actionind acum EDIT, linia 15 va apare in zona de editare. Cind cursorul programului este ascuns intre doua linii, EDIT aduce in josul ecranului linia care are numarul de linie imediat urmator.
Se scrie acum:

```
30              (si CR)
```

De aceasta data cursorul este ascuns dupa sfirsitul programului.
Cu comanda:

```
LIST 15
```

pe ecran se obtine:

```
15>LET b=15
20 PRINT a+b
```

Instructiunea LIST 15 produce listarea incepind cu linia 15 si pune cursorul programului la linia 15. Pentru un program foarte lung, LIST va fi o metoda mai utilizata de mutare a cursorului decit ↓ sau ↑.
Aceasta ilustreaza o alta utilitate a numerelor de linie; ele actioneaza ca nume ale liniilor de program astfel incit se pot face referiri la ele in acelasi mod in care se fac referiri la numele de variabile. LIST (neurmat de un numar) determina listarea de la inceputul programului.

O alta comanda este NEW. Efectul ei consta in stergerea programelor si variabilelor din memoria calculatorului.

Exemplul 2. Sa se scrie un program care transforma temperatura din grade Fahrenheit in grade Celsius.

```
10 REM conversia temperaturii
20 PRINT "grade F","grade C"
30 PRINT
40 INPUT "introduceti gradele F.   ",f
50 PRINT f,(f-32)*5/9
60 GO TO 40
```

Este necesar sa fie introdusa pe rind fiecare litera pentru a obtine "conversia temperaturii" in linia 10. In linia 60 se obtine GO TO actionind tasta G (desi contine spatiu, GO TO constituie un singur cuvint cheie).

Rulind programul, se va vedea pe ecran capul de tabel tiparit de linia 20. Linia 10 este ignorata de calculator, instructiunea **REM** introducind un comentariu in textul sursa. Comanda **INPUT** din linia 40 asteapta sa fie introdusa o valoare pentru variabila F; se introduce un numar si se actioneaza apoi CR. Calculatorul afiseaza rezultatul si nu se opreste din rulare, ci asteapta alt numar (datorita saltului din linia 60). Programul se poate opri prin actionarea tastei STOP in momentul in care pe ecran apare scris:

```
Introduceti gradele F.
```

Calculatorul intoarce mesajul

```
H    STOP in INPUT 40:1
```

care precizeaza de ce si unde s-a oprit din rulare (in prima instructiune din linia 40).

Pentru a continua programul se introduce **CONTINUE** si calculatorul va astepta alt numar. **CONTINUE** determina rularea programului de la linia de la care se oprise executia (linia 40).

Se rescrie linia 60 sub forma

```
60 GO TO 31
```

In executie, aceasta varianta se comporta identic cu varianta precedenta. Daca numarul liniei intr-o comanda **GO TO** se refera la o linie inexistenta, atunci se sare la linia imediat urmatoare numarului dat. Acest lucru este valabil si pentru comanda RUN (de fapt **RUN** are acelasi efect cu **RUN** 0).

Daca tiparim numere pina cind se umple ecranul, calculatorul va muta intreaga parte de sus a ecranului cu o linie pentru a face loc, pierzind astfel capul de tabel. Cind am terminat de tiparit, programul se poate opri cu **STOP** urmat de CR. Lista de instructiuni a programului se poate afisa dupa intrerupere apasind CR.

Se analizeaza instructiunea **PRINT** din linia 50. Virgula utilizata aici determina inceperea tiparirii fie in marginea din stinga, fie in mijlocul ecranului, in functie de ce urmeaza dupa virgula. In acest caz tiparirea temperaturii in grade Celsius are loc in mijlocul liniei.

Caracterul punct si virgula ";" determina tiparirea sirului urmator imediat dupa sirul precedent. Se poate vedea aceasta daca in linia 50 e inlocuit caracterul "," cu ";".

Alt semn de punctuatie ce poate fi utilizat in comenzi PRINT este apostroful "'". El determina saltul cursorului la inceputul liniei urmatoare si continuarea tiparirii din acel punct, ca si cum elementele despartite prin "'" ar fi fost sub incidenta unor comenzi PRINT succesive. Pentru ca instructiunea PRINT sa nu determine saltul la linia urmatoare este necesar ca PRINT-ul precedent sa se termine cu "," sau cu ";". Pentru exemplificare sa se substituie linia 50 pe rind cu liniile:

```
50 PRINT f,
50 PRINT f;
50 PRINT f
50 PRINT f'
```

Se constata ca varianta cu "," imparte totul in doua coloane, cea cu ";" scrie totul compact, cea fara semn de punctuatie si cea cu "'" scriu un numar pe o linie.

In memorie pot exista simultan mai multe programe cu conditia ca numerele de linie sa fie in intervale disjuncte.

Exemplul 3.

```
100 INPUT n$
110 PRINT "Salut ";n$ " !"
120 GO TO 100
```

Acesta este un program care poate coexista in memorie cu programul din exemplul 2 intrucit unul are numerele de linie in intervalul 0...60, iar celalalt in 100...120. Pentru lansarea in executie a programului din exemplul 3 se da comanda RUN 100. Executia unei comenzi RUN determina stergerea ecranului si a tuturor variabilelor, dupa aceasta executind sirul instructiunilor programului. Daca nu se doreste initializarea variabilelor si stergerea ecranului, se poate utiliza comanda GO TO 100. .

La executia programului din exemplul 3 se observa ca pe ecran apare "L" care indica faptul ca se doreste citirea unui sir de caractere. Sistemul admite ca o instructiune INPUT sa se comporte similar cu o instructiune de atribuire, dar numai pentru cazul citirii de variabile de tip sir de caractere. Pentru aceasta se sterg ghilimelele (utilizind <-- si DELETE) si se introduce numele unei variabile de acelasi tip. Introducerea unui nume de variabila determina cautarea valorii acelei variabile ce trebuia citita de la tastatura.

De exemplu daca la executia programului din exemplul 3 la prima solicitare de sir de caractere se introduce "ANA", valoarea variabilei n$ va deveni n$="ANA". La urmatoarea citire se introduce "MARIA". n$ devine n$="MARIA". La executia urmatoarei instructiuni INPUT se va introduce n$; in acest caz se cauta valoarea vechii variabile n$ si i se asociaza variabilei n$. Deci comanda se comporta similar cu LET n$=n$. Valoarea lui n$ in urma acestei instructiuni va fi n$="MARIA", deci instructiunea PRINT din linia 110 va tipari:

```
Salut MARIA !
```

Uneori din greseala se scrie un program ce ruleaza la infinit, cum este urmatorul:

```
200 GO TO 200
RUN 200
```

Pentru oprirea executiei se actioneaza **BREAK** (CAPS SHIFT si SPACE) si calculatorul raspunde cu mesajul:

L **BREAK into program, 200:1**

La sfirsitul fiecarei instructiuni programul verifica daca aceste taste sint actionate; daca da, este oprita rularea.

Tasta **BREAK** poate fi utilizata de asemenea cind sint conectate casetofonul sau imprimanta, in cazul cind calculatorul asteapta ca aceste periferice sa efectueze o comanda. Mesajul produs in acest caz este diferit:

D **BREAK - CONT repeats.**

Comanda **CONTINUE**, in cazul lucrului cu casetofonul sau imprimanta repeta instructiunea unde programul a fost oprit. Listingurile automate sint acelea care nu rezulta in urma unei comenzi **LIST**, ci au loc dupa introducerea unei linii noi. De retinut este faptul ca linia curenta (cea cu >) apare intotdeauna pe ecran si in mod normal in pozitie centrala. Calculatorul memoreaza numarul liniei curente si de asemenea al primei linii din partea de sus a ecranului.

Cind incearca sa listeze, primul lucru pe care-l face este sa compare prima linie de pe ecran cu linia curenta. Daca prima linie de pe ecran este mai mare decit linia curenta, atunci cursorul va apare pe prima linie a ecranului. Altfel listarea consta in tiparirea pe ecran in mod defilare a programului cuprins intre prima linie si linia curenta.

Oricum, mai intii se efectueaza un calcul aproximativ pentru a vedea cit timp ia listarea si daca acesta este prea lung, linia din virf se muta mai jos pentru a fi mai aproape de linia curenta. Acum, avind stabilita linia din virf, listarea poate incepe. Daca linia curenta a fost listata, listarea se opreste cind s-a ajuns la sfirsitul programului sau la partea de jos a ecranului.

3. LIMBAJUL BASIC

3.1 VARIABILE SI EXPRESII ARITMETICE

Cuprins: Sume de variabile, expresii, notatii
Operatii: +,-,*,/

Versiunea BASIC a calculatorului HC-85 admite pentru variabilele numerice nume formate din oricite caractere (litere sau cifre), care incep cu o litera. Printre caractere poate fi si blancul, care este insa ignorat. Prezenta lui face variabila mai usor de citit. Sistemul face filtrarea literelor mari, astfel incit atit litera mare cit si litera mica corespunzatoare sint interpretate la fel. Nu este indicata folosirea numelor foarte lungi deoarece sint greu de manipulat.

Variabilele speciale sint:

1. Variabilele folosite in instructiunile FOR, care trebuie sa fie reprezentate printr-o singura litera.
2. Variabilele de tip sir de caractere, al caror nume este format dintr-o litera urmata de "$".

Expresiile numerice pot fi reprezentate si printr-un numar zecimal urmat de un exponent.

Exemplul 1. Sa se tipareasca numerele:

```
PRINT 2.3e0
PRINT 2.34e1
```

si asa mai departe pina la

```
PRINT 2.34e15
```

Se observa ca dupa un timp calculatorul incepe sa foloseasca scrirea cu exponent deoarece nu se pot utiliza mai mult de 14 caractere consecutive pentru scrierea unui numar.

Se poate tipari in mod similar:

```
PRINT 2.34e-1
PRINT 2.34e-2
```

si asa mai departe. Comanda PRINT afiseaza numai 8 cifre semnificative.

Exemplul 2.

```
PRINT 4294967295,4294967295-429e7
```

Acest exemplu demonstreaza ca toate cifrele numarului 4294967295 sint memorate, desi nu toate pot fi tiparite pe ecran.

HC-85 utilizeaza scrierea numerelor in virgula mobila.

Numerele sint reprezentate cu precizie de aproximativ noua cifre si jumatate. Cel mai mare intreg ce poate fi reprezentat cu precizie in memorie este 2e32-1=4294967295.

Exemplul 3.

```
PRINT 1e10+1-1e10,1e10-1e10+1
```

Rezultatele afisate vor fi:

```
0     1
```

deoarece 1e10+1 si 1e10 au aceeasi reprezentare interna.

Operatiile aritmetice executate de calculator sint inmultirea, impartirea, adunarea si scaderea. Operatiile de inmultire "*" si impartire "/" au prioritate egala. De aceea, o expresie ce contine numai inmultiri si impartiri se executa de la stinga la dreapta. Adunarea si scaderea au de asemenea, prioritate egala dar mai mica decit a inmultirii si a impartirii.

Pentru a modifica ordinea de executie a operatiilor se folosesc parantezele.

3.2 SIRURI DE CARACTERE

Cuprins: Operatii cu siruri de caractere

Sirurile de caractere sint reprezentate prin secvente de caractere ASCII, incadrate intre ghilimele ("). Daca se doreste tiparirea in text a caracterului ghilimele, el trebuie sa fie dublat. Un sir de caractere poate fi atribuit ca valoare unei variabile sir sau poate fi tiparit cu o comanda **PRINT**.

Fiind dat un sir, un subsir al lui consta in citeva caractere consecutive continute in el, luate in secventa. De exemplu "string" este un subsir al lui "bigger string", insa "b string" nu este. Manipularea subsirurilor in BASIC se face cu:

```
s(n1 TO n2)
```

unde:

1. s este un sir de caractere sau o variabila sir.
2. n1,n2 sint numere intregi nenegative ce reprezinta ordinul caracterului de inceput, respectiv de sfirsit, din subsir. Daca n1>n2, rezultatul este sirul vid ("").

Daca nu se precizeaza inceputul si/sau sfirsitul subsirului se iau implicit 1, respectiv lungimea sirului.

Exemplul 1.

```
"abcdef"(2 TO 5)="bcde"
"abcdef"( TO 5)="abcdef"(1 TO 5)="abcde"
"abcdef"(2 TO )="abcdef"(2 TO 6)="bcdef"
"abcdef"( TO )="abcdef"(1 TO 6)="abcdef"
"abcdef"(3)="abcdef"(3 TO 3)="c"
"abcdef"(5 TO 7)  da mesaj de eroare deoarece sirul are
                  numai sase caractere
"abcdef"(8 TO 7)=""
"abcdef"(1 TO 0)=""
```

Exemplul 2.

```
10 LET a$="able was !"
20 FOR n=1 TO 10
30 PRINT a$(n TO 10), a$((11-n) TO 10)
40 NEXT n
50 STOP
```

Exemplul 3.

```
10 LET c$="acesta este un calculator HC-85"
20 LET c$(13 TO 25)="hc-85"
30 PRINT c$
```

Dupa executia programului pe ecran va apare mesajul:

```
Acesta este hc-85          HC-85
```

Daca intr-o atribuire sirul din dreapta contine mai putine caractere decit sint specificate in subsirul din stinga, atunci diferenta de lungime va fi completata cu blancuri. O astfel de asignare se numeste "procusteana".

3.3 TABLOURI

Cuprins: **Tablouri de numere si siruri**
DIM

In limbajul BASIC al calculatorului HC-85 se pot defini variabile de tip tablou cu oricite dimensiuni. Elementele tabloului pot fi numere reale, caz in care numele variabilei este reprezentat pintr-o singura litera, sau de tip sir de caractere, numele variabilei fiind format dintr-o litera urmata de $. Inainte de a utiliza un tablou, trebuie rezervat spatiu in calculator pentru el; aceasta se realizeaza utilizind instructiunea DIM, a carei forma este:

```
DIM m(n1,n2,..,nk)
```

unde:

1. m - este numele unei variabile de tip tablou.
2. n1,n2,..,nk - sint numerele maxime de componente corespunzatoare fiecarei dimensiuni a tabloului.

Printr-o comanda DIM poate fi definita numai o singura variabila de tip tablou. Aceasta instructiune are urmatorul efect:

1. rezerva spatiul necesar tabloului definit.
2. initializeaza elementele tabloului cu 0.
3. sterge orice tablou care are acelasi nume cu variabila definita prin instructiunea curenta.

Se mentioneaza ca pot coexista un tablou si o variabila simpla cu acelasi nume, fara sa apara confuzii.

Sirurile dintr-un tablou difera de sirurile simple prin aceea ca au lungime fixa si asignarea lor este procusteana. Un alt mod de interpretare al unui tablou de siruri de caractere este ca tablou de caractere simple cu numarul dimensiunilor majorat cu 1 fata de cazul precedent. Un tablou de siruri si o variabila sir simpla nu pot avea acelasi nume (spre deosebire de cazul variabilelor numerice).

Pentru a defini un tablou a$ de 5 siruri, trebuie stabilita mai intii lungimea sirului - spre exemplu 10 caractere.

Linia:

```
DIM a$(5,10)
```

defineste 5*10=50 caractere, dar fiecare rind poate fi interpretat ca un sir.

De exemplu a$(1) este format din:

```
a$(1,1) a$(1,2) ...  a$(1,10)
```

Daca sint utilizate doua dimensiuni, se obtine un singur caracter, dar daca este omisa a doua dimensiune, atunci se obtine un sir cu lungime fixa. Astfel a$(2,7) e al saptelea caracter in sirul a$(2); o alta notatie a aceluiasi element este a$(2)(7).

Ultimul indice poate avea si forma unui selector de subsir. De exemplu, daca a$(2)="12345667890", atunci

```
a$(2.4 TO 8)= a$(2)(4 TO 8)="45678"
```

Se pot defini variabile de tip tablou de siruri de caractere cu o singura dimensiune; in acest caz variabila se comporta ca o variabila simpla cu exceptia faptului ca are totdeauna lungime fixa iar asignarea ei este procusteana.

Exemplu

```
DIM a$(10)
```

3.4 INITIALIZAREA VARIABILELOR

Cuprins: READ, DATA, RESTORE

Introducerea constantelor intr-un program se face prin grupul de instructiuni **READ**, **DATA** si **RESTORE**. Forma generala a unei instructiuni **READ** este:

```
READ n1,n2,...
```

unde n1,n2,... este lista variabilelor care trebuiesc initializate, ele fiind separate prin virgula. Instructiunea **READ** lucreaza la fel cu instructiunea **INPUT**, exceptind faptul ca valorile variabilelor sint luate dintr-o instructiune **DATA**, nu de la terminal.

Fiecare instructiune **DATA** este o lista de expresii numerice sau de tip sir de caractere, separate prin virgula. Instructiunile **DATA** pot fi puse oriunde in program, ele comportindu-se ca o lista unica realizata prin concatenarea tuturor instructiunilor **DATA** din program (lista **DATA**).

Cind calculatorul citeste prima variabila cu **READ**, ei ii este asociata prima valoare din lista **DATA**, si asa mai departe. Daca se incearca citirea mai multor variabile decit numarul valorilor din lista **DATA**, atunci apare eroare.

Este posibil sa se faca salturi in lista **DATA**, utilizind instructiunea **RESTORE**. Forma instructinii este:

```
RESTORE n
```

Ea face ca instructiunea **READ** urmatoare sa citeasca datele de la o instructiune **DATA** aflata la linia "n" sau dupa aceasta. Daca "n" lipseste, se ia valoare implicita 1.

Exemplul 1.

```
10 READ a,b,c
20 PRINT a,b,c
30 DATA 10,20,30
40 STOP
```

Rezultatele programului vor fi:

```
10    20        (a=10, b=20)
30              (c=30)
```

Exemplul 2.

```
10 READ d$
20 PRINT "Data este:  ",d$
30 DATA "21 aprilie 1985"
```

Rezultatul acestui program este:

```
Data este:  21 aprilie 1985
```

Exemplul 3.

```
10 READ a,b
20 PRINT a,b
30 RESTORE 10
40 READ x,y,z
50 PRINT x,y,z
60 DATA 1,2,3
70 STOP
```

Rezultatele furnizate de acest program sint:

```
1     2         (a=1, b=2)
1     2         (x=1, y=2)
3               (z=3)
```

3.5 OPERATII LOGICE

Cuprins: =, <, >, <=, >=, <>
AND, OR, NOT

Operatiile aritmetice executate de calculator sint inmultirea, impartirea, adunarea si scaderea. Operatiile de adunare si scadere au prioritate egala dar mai mica decit a inmultirii si a impartirii.

Pentru sirurile de caractere s-a definit operatia de concatenare, notata cu "+".

Exemplul 1.

```
10 LET n$="Ionescu "
20 LET p$="Ana"
30 LET s$=n$+p$
40 PRINT s$
50 STOP
```

Programul prezentat va determina tiparirea pe ecran a textului:

```
Ionescu Ana
```

care reprezinta valoarea variabilei s$.

Relatiile de ordine in multimea numerelor sint relatiile de egalitate si de inegalitate apelabile folosind notatiile "=", "<", ">","<=", ">=", "<>".

In multimea sirurilor de caractere relatia de ordine folosita este ordonarea alfabetica, relatiile folosite fiind aceleasi ca la numere.

Pentru realizarea unor expresii complexe se pot utiliza si operatiile logice "OR", "AND" si "NOT" care admit operanzi de tip boolean. De exemplu instructiunea:

```
IF a$="DA" AND x>0 THEN PRINT x
```

tipareste valoarea numarului "x" daca sint indeplinite simultan cele 2 conditii.

Similar se pot realiza expresii cu "OR" daca se doreste identificarea situatiei in care cel putin una dintre conditii este indeplinita. Operatia "NOT" produce ca rezultat inversul valorii argumentului sau.

Operatiile "OR", "AND", "NOT" pot fi aplicate si unor argumente numerice. Functiile definite astfel sint:

1. x AND y ia valoarea
   x , daca y e nenul
   0 , daca y=0

2. x OR y ia valoarea
   1 , daca y e nenul
   x , daca y=0

3. NOT x ia valoarea
   0 , daca x e nenul
   1 , daca x=0

In continuare sint prezentate operatiile recunoscute de limbajul BASIC in ordinea crescatoare a prioritatilor:

1. "OR"
2. "AND"
3. "NOT"
4. relatiile conditionale
5. "+", "-"
6. "*", "/"

3.6 FUNCTII

Cuprins: ↑, PI, EXP, LN, SIN, COS, TAN, ASN, ACS, ATN, DEF, LEN, STR$, VAL, SGN, ABS, INT, SQR, FN

Functiile definite de calculator au prioritate mai mare decit operatiile. Daca in evaluarea unei expresii este necesara o alta ordine de executie a operatiilor si functiilor decit cea determinata de prioritatile lor, atunci se foloseso paranteze.

Functiile matematice definite in BASIC sint ridicarea la putere, functia exponentiala, functia logaritmica si functiile trigonometrice.

Functia ridicare la putere "↑" are prioritate mai mare decit inmultirea si impartirea. Ea necesita 2 operanzi dintre care primul este obligatoriu pozitiv. Intr-o insiruire de ridicari la putere, ordinea evaluarii este de la stinga la dreapta, ceea ce inseamna ca :

```
2↑3↑2=8↑2=64
```

Functia **EXP** defineste functia exponentiala:

```
EXP x=e↑x
```

unde e=2,71...

Functia **LN** calculeaza logaritmul natural al argumentului.

Ea poate fi utilizata la calculul unui logaritm in orice baza folosind formula:

```
LOGa x=LN x/ LN a
```

SIN, COS, TAN, ASN, ACS, ATN sint mnemonicele functiilor sinus, cosinus, tangenta, arcsinus, arccosinus si respectiv arctangenta.

Sistemul pune la dispozitia utilizatorului numarul "pi", ce poate fi apelat apasind tasta PI. Comanda **PRINT PI** tipareste valoarea numarului "pi".

Functiile descrise in continuare sint disponibile in modul de lucru extins. Actionarea simultana a tastelor **CAPS SHIFT** si **SYMBOL SHIFT** determina trecerea din modul "L" in modul "E".

Functia **LEN** da lungimea unui sir.

Exemplul 1.

```
PRINT LEN "majuscule"
```

va determina tiparirea numarului 9.

Functia **STR$**, converteste numere in siruri. Argumentul este un numar, iar rezultatul este sirul care ar apare pe ecran daca numarul ar fi afisat cu **PRINT**. Se observa ca numele functiei se sfirseste cu "$" pentru a arata ca rezultatul ei este un sir.

Exemplul 2.

```
LET a$=STR$ 1e2
```

Instructiunea de mai sus are acelasi efect cu:

```
LET a$="100"
```

Comanda

```
PRINT LEN STR$ 100.000
```

produce raspunsul 3, deoarece STR$ 100.000="100"

Functia **VAL** converteste siruri de caractere in numere.

```
VAL "3.5"=3.5
```

Daca se aplica functiile **STR$** si **VAL** asupra unui numar, totdeauna se va obtine numarul initial, pe cind daca se aplica **VAL** urmat de **STR$** asupra unui sir de caractere nu se obtine totdeauna sirul initial. Evaluarea functiei VAL se face in 2 pasi:

1. argumentul este evaluat ca sir.
2. ghilimelele sint indepartate si caracterele ramase sint evaluate ca numere.

Exemplul 3.

```
VAL "2*3"=6
VAL ("2"+"*3")=6
```

Alta functie similara lui **VAL** dar mai putin utilizata este **VAL$**. Si aceasta functie se evalueaza tot in 2 pasi; primul pas este la fel cu al functiei **VAL**, dar dupa inlaturarea ghilimelelor caracterele sint evaluate ca alt sir.

```
VAL$ """fructe"""="fructe"
```

Functia **SGN** aplicata asupra variabilei x are urmatoarea definitie:

```
1.  1,     daca x>0
2.  0,     daca x=0
3. -1,     daca x<0
```

Functia **ABS** produce valoarea absoluta a numarului pe care-l are ca argument.

```
ABS -3.2 = ABS 3.2 = 3.2
```

Functia **INT** furnizeaza partea intreaga a argumentului sau.

```
INT  3.9 =  3
INT -3.9 = -4
```

Functia **SQR** calculeaza radacina patrata a argumentului sau care este un numar pozitiv.

```
SQR 0.25 = 0.5
SQR -4          ==>genereaza mesaj de eroare
```

Sistemul permite definirea de functii utilizator. Numele posibile pentru acestea sint **FN** urmat de o litera (daca rezultatul e un numar); sau **FN** urmat de o litera si **$** (daca rezultatul e un sir). Obligatoriu argumentul trebuie sa fie inclus in paranteze. Definirea functiilor utilizator se face cu functia predefinita **DEF**. Definirea functiei de ridicare la patrat se poate face astfel:

```
DEF FN s(x)=x*x
```

Rotunjirea unui numar real la cel mai apropiat intreg poate fi facuta prin aplicarea functiei **INT** asupra argumentului marit cu 0.5:

```
20    DEF FN r(x)=INT(X+0.5)
```

Exemplul 5.

```
10 LET x=0:  LET y=0:  LET a=10
20 DEF FN p(x,y)=a+x*y
30 DEF FN q()=a+x*y
40 PRINT FN p(2,3), FN q()
```

Cind este evaluata FN p(2,3), "a" are valoarea 10, deoarece e variabila libera, x are valoarea 2 deoarece este primul argument si y are valoarea 3 deoarece este al doilea argument. Rezultatul este 10+2*3=16.

Cind este evaluata functia fara argumente FN q, a,x si y sint variabile libere si au valorile: 10, 0 respectiv 0. Raspunsul in acest caz este 10+0*0=10.

Schimbind linia 20 cu

```
20    DEF FN p(x,y)=FN q()
```

de aceasta data FN p(2,3) va avea valoarea 10.

O functie poate avea pina la 26 argumente numerice si in acelasi timp pina la 26 argumente de tip sir de caractere.

3.7 DECIZII

Cuprins: IF, THEN, STOP.

Instructiunea care realizeaza luarea deciziilor este de forma:

```
n  IF conditie THEN comenzi
```

unde

1. "n" este numarul liniei.
2. "comenzi" este o secventa de instructiuni care trebuie sa fie executata in cazul in care "conditia" este adevarata.
3. "conditie" este o relatie operationala care in urma evaluarii poate fi adevarata sau falsa. Daca conditia este adevarata, atunci se executa secventa de instructiuni scrisa dupa **THEN**. Altfel, programul executa instructiunile de pe linia urmatoare.

Cele mai simple conditii compara doua numere sau doua siruri de caractere. Ele pot testa daca doua numere sint egale sau daca unul este mai mare decit celalalt. Se poate testa si egalitatea a doua siruri de caractere, sau daca in ordinea alfabetica unul apare inaintea celuilalt.

Exemplu

```
10 REM Ghiciti numarul
20 INPUT a :  CLS
30 INPUT "Ghiciti numarul" , b
40 IF b=a THEN PRINT "Rezultat corect" :  STOP
50 IF b<A THEN PRINT "Prea mic!  Mai incearca  o
   data!"
60 IF b>a THEN PRINT "Prea mare!  Mai  incearca
   o data!"
70 GO TO 30
```

In acest program linia 40 compara variabilele a si b. Daca sint egale, programul este oprit cu comanda STOP. In partea de jos a ecranului apare mesajul

```
9    STOP statement, 40:3
```

care arata ca oprirea programului este cauzata de a treia instructiune din linia 40.

Linia 50 determina daca b este mai mic decit a, iar linia 60 opusul, adica daca b este mai mare decit a. Instructiunea CLS din linia 20 sterge ecranul si impiedica adversarul de joc sa vada ce numar s-a introdus.

3.8 ITERATII

Cuprins: FOR, NEXT, TO, STEP

In BASIC instructiunea de ciclare este FOR - NEXT. Forma generala a instructiunii FOR este:

```
FOR v=vi TO vf STEP p
corp ciclu
NEXT v
```

unde

1. "v" este o variabila contor specifica ciclului FOR - NEXT; ea trebuie sa aiba numele format dintr-o singura litera.
2. "vi" este valoarea cu care este initializat contorul ciclului.
3. "vf" este valoarea maxima la care poate ajunge "v"; deci "v" <= "vf" (s-a presupus ca "p" > 0).
4. "p" este marimea pasului; el reprezinta diferenta intre doua valori succesive ale contorului.
5. "corp ciclu" este secventa de instructiuni ce se repeta. "vi","vf" si "p" pot fi exprimate prin constante, variabile sau expresii de tip real.

In cazul in care "p" este negativ, regula de raminere in ciclu este "v" >= "vf".

Doua cicluri **FOR - NEXT** pot fi imbricate sau complet separate. Este gresita suprapunerea partiala a doua cicluri. De asemenea trebuie evitat saltul din exterior in interiorul unei bucle FOR - NEXT deoarece contorul nu poate fi initializat decit printr-o instructiune FOR. Pentru a fi siguri ca nu se fac salturi in interiorul unui ciclu se pot scrie toate instructiunile ciclului pe o singura linie (daca spatiul permite).

Exemplul 1.

```
10 FOR n=10 TO 1 STEP -1
20 PRINT n
30 NEXT n
```

Exemplul 2.

```
50 FOR m=0 TO 6
60 FOR n=0 TO m STEP 1/2
70 PRINT m;":";n;" ";
80 NEXT n
90 PRINT
100 NEXT m
```

Exemplul 3.

```
100 FOR m=0 TO 10: PRINT m: NEXT m
```

Exemplul 4.

```
FOR n=0 TO 1 STEP 0: INPUT a: PRINT a: NEXT n
```

Aceasta comanda determina repetarea la infinit a instructiunii **INPUT** in modul de lucru imediat (deci nu prin program). Daca apare o eroare, comanda **INPUT** se pierde si deci pentru continuarea citirii trebuie rescrisa intreaga linie.

3.9 SUBRUTINE

Cuprins: GOSUB, RETURN

Utilizarea subrutinelor este posibila prin utilizarea instructiunii **GO SUB** (go to subroutine-apel de subrutina) si **RETURN** (revenire din subrutina). Instructiunea GOSUB are forma:

```
GO SUB n
```

unde "n" este numarul primei linii din subrutina. Ea este asemanatoare instructiunii **GO TO n**, cu exceptia faptului ca in cazul instructiunii **GO SUB** este memorata adresa instructiunii, astfel incit dupa executarea subrutinei programul continua cu instructiunea urmatoare saltului la subrutina. Aceasta se realizeaza memorind numarul liniei si numarul instructiunii din linie (care impreuna formeaza adresa de revenire) intr-o stiva.

Instructiunea **RETURN** ia adresa din virful stivei **GO SUB** si merge la instructiunea care ii urmeaza.

In BASIC subrutinele sint recursive.

Exemplu

```
10 INPUT a:  CLS
20 INPUT "ghiciti numarul !",b
30 IF a=b THEN PRINT "corect !!!":  STOP
40 IF a<b THEN GO SUB 90
50 IF a>b THEN GO SUB 90
60 GO TO 20
90 PRINT "Mai incearca o data !"
100 RETURN
```

Instructiunea **GO TO** este foarte importanta deoarece sistemul semnaleaza eroarea daca, in executie, intilneste un **RETURN** care nu a fost precedat de un **GO SUB**.

3.10 GENERAREA NUMERELOR ALEATOARE

Cuprins; RND, RANDOMIZE

Generarea numerelor aleatoare se face cu functia predefinita **RND**. Ea nu este o functie complet aleatoare ci o functie periodica cu perioada suficient de mare (65535), astfel incit efectul de periodicitate poate fi neglijat. In cadrul unei perioade, numerele generate sint complet aleatoare. In anumite privinte, **RND** se comporta ca o functie fara argumente: efectueaza calcule si produce un rezultat. De fiecare data cind e utilizata, rezultatul sau este un numar aleator nou, cuprins intre 0 si 1 (uneori poate lua valoarea 0, dar niciodata 1). Daca se doreste ca numerele aleatoare sa fie intr-un anumit domeniu de valori se poate proceda ca in exemplele urmatoare:

```
5*RND          genereaza numere intre 0 si 5;
1.3+0.7*RND    produce numere intre 1.3 si 2;
1+INT(RND*6)   furnizeaza numere aleatoare  intregi
               intre 1 si 6.
```

Exemplu

```
10 REM  Program de simulare a aruncarii zarurilor
20 CLS
30 FOR n=1 TO 2
40 PRINT 1+INT(RND*6);" ";
50 NEXT n
60 INPUT a$: GO TO 20
```

Linia 60 face sa fie generata o pereche de numere aleatoare dupa fiecare apasare a tastei CR.

Functia **RANDOMIZE** e utilizata pentru a face ca **RND** sa porneasca dintr-un punct definit al secventei de numere; argumentul sau este un numar intre 1 si 65535 care reprezinta numarul de ordine al viitorului apel al functiei **RND**. Efectul instructiunii **RANDOMIZE** se poate vedea in programul urmator.

```
10 RANDOMIZE 1
20 FOR n=1 to 5 :PRINT RND :NEXT n
30 PRINT:GO TO 10
```

Dupa fiecare executie a instructiunii **RANDOMIZE 1**, **RND** va furniza o secventa de 5 numere ce incepe cu 0.0022735596, care este primul numar generat de functia **RND** (are numarul de ordine 1). **RANDOMIZE** poate fi folosit la testarea programelor ce contin functia **RND**, deoarece secventa numerelor aleatoare generate este mereu aceeasi.

RANDOMIZE, ca si **RANDOMIZE 0**, are efect diferit de **RANDOMIZE** urmat de un numar. Aceasta instructiune utilizeaza timpul trecut de la punerea in functiune a calculatorului. Programul:

```
10 RANDOMIZE
20 PRINT RND: GO TO 10
```

determina tiparirea aceluiasi numar. Deoarece timpul de lucru al calculatorului a crescut cu aceeasi cantitate la fiecare executie a lui **RANDOMIZE**, urmatorul **RND** furnizeaza aproximativ acelasi rezultat.

Pentru a se obtine o secventa aleatoare se inlocuieste GO TO 10 cu GO TO 20.

Exemplu

Programul determina frecventa de aparitie a "capului" si a "pajurei" la aruncarea unei monezi.

```
10 LET cap=0:LET pajura=0
20 LET moneda=INT(RND*2)
30 IF moneda=0 THEN LET cap=cap+1
40 IF moneda=1 THEN LET pajura=pajura+1
50 PRINT cap;  ",";pajura
60 IF pajura>0 THEN PRINT cap/pajura;
70 PRINT: GO TO 20
```

Daca timpul de rulare este suficient de mare, raportul cap/pajura devine aproximativ 1, deoarece numerele aleatoare generate sint uniform repartizate in intervalul 0,1.

3.11 SETUL DE CARACTERE

Cuprins: CODE, CHR$, POKE, PEEK, USR, BIN

Alfabetul utilizat de HC-85 cuprinde 256 caractere si fiecare are un cod intre 0 si 255. Caracterele pot fi simboluri simple sau cuvinte cheie ca **PRINT**, **STOP**, >, etc.

Pentru conversia intre coduri si caractere, limbajul poseda doua functii: **CODE** si **CHR$**. **CODE** se aplica unui sir si intoarce codul primului caracter al sirului (sau 0 daca sirul e vid).

CHR$ se aplica unui numar si produce caracterul ce are acel cod.

Setul de caractere este format din: caracterele ASCII, cuvinte cheie, caractere grafice definite de utilizator.

Un caracter se deseneaza pe o retea de 8*8 puncte, fiecarui punct corespunzindu-i un bit in memorie. Pentru programarea unui caracter definit de utilizator este necesara descrierea starii fiecarui punct al matricii prin care se reprezinta caracterul respectiv:

1. 0 corespunde unui punct alb
2. 1 corespunde unui punct negru

Pentru definirea caracterului se folosesc 8 instructiuni **BIN**. O instructiune **BIN** descrie o linie a caracterului, argumentul sau fiind format din 8 cifre binare.

Cele 8 numere sint memorate in 8 octeti care corespund aceluiasi caracter.

Instructiunea **USR** converteste un argument de tip sir in adresa din memorie a primului octet al caracterului definit de utilizator corespunzator argumentului. Argumentul trebuie sa fie un singur caracter; el poate fi graficul definit de utilizator sau litera corespunzatoare (majuscula sau minuscula).

POKE memoreaza un numar direct intr-o locatie de memorie, fara sa faca apel la mecanismele utilizate in mod obisnuit in BASIC. Opusul lui **POKE** este **PEEK**, care ne permite sa vizualizam continutul unei locatii de memorie, fara a-l modifica.

Pentru a defini caracterul grafic pi (care sa apara pe ecran la apasarea tastei P in mod grafic) se utilizeaza urmatoarea secventa de program:

```
10 FOR n=0 TO 7
20 INPUT  acum: POKE USR "p"+n, acum
30 NEXT n
```

Datele introduse vor fi (in ordinea prezentata):

```
BIN 00000000
BIN 00000000
BIN 00000010
BIN 00111100
BIN 01010100
BIN 00010100
BIN 00010100
BIN 00000000
```

In cele ce urmeaza se prezinta modul de obtinere a cuvintelor cheie. Caracterele 0,..,31 sint caractere de control al modului de lucru. De exemplu **CHR$6** realizeaza tabularea pe orizontala (efect similar unei virgule intr-o instructiune **PRINT**).

```
PRINT 1;  CHR$ 6;  2
```

are acelasi efect cu:

```
PRINT 1,2
```

si cu:

```
LET a$="1"+CHR$6+"2"
PRINT a$
```

CHR$8 determina mutarea cursorului inapoi cu o pozitie.

Exemplu

```
PRINT "1234";  CHR$8;  "5"
```

tipareste:

```
1235
```

CHR$13 muta cursorul la inceputul liniei urmatoare.

Utilizind codurile pentru caractere putem extinde conceptul de ordine alfabetica pentru a acoperi siruri ce contin orice caractere, nu numai litere, folosind in locul alfabetului uzual de 26 litere, alfabetul extins de 256 caractere (la codificarea caracterelor s-a avut in vedere ca ordinea crescatoare a codurilor atasate literelor sa coincida cu ordinea alfabetica).

Este prezentata mai departe o regula de gasire a ordinii in care se afla doua siruri. Mai intii se compara primele caractere. Daca sint diferite, unul dintre ele are codul mai mic decit celalalt si, deci, se poate decide care este ordinea alfabetica a sirurilor. Daca aceste coduri sint egale, se compara urmatoarele caractere.

Exemplu

```
  5 LET b=BIN 01111100:LET c=BIN 00111000:LET d=BIN
    00010000
 10 FOR n=1 TO 6:  READ p$:  REM 6 piese
 20 FOR f=0 TO 7:  REM citeste piesele in octeti
 30 READ a:  POKE USR p$+f,a
 40 NEXT f
 50 NEXT n
100 REM bishop
110 DATA "b", 0, 0, BIN 001001000, BIN 01000100
120 DATA BIN 01101100, c, b, 0
130 REM king
140 DATA "k", 0, d, c, d
150 DATA c, BIN 010001000, c, 0
160 REM rook
170 DATA "r", 0, BIN 01010100, b, c
180 DATA c,b,b,0
190 REM queen
200 DATA "q",0,BIN 01010100, BIN 00101000, d
210 DATA BIN 01101100, b, b, 0
220 REM pawn
230 DATA "p", b, 0, d, c
240 DATA c, d, b, 0
250 REM knight
260 DATA "n", 0, d, c, BIN 01111000
270 DATA BIN 00011000, c, b, 0
```

3.12 GRAFICE

Cuprins: PLOT, DRAW, CIRCLE, POINT

In acest capitol se prezinta trasarea desenelor cu HC-85. Partea utilizabila a ecranului are 22 de linii si 32 de coloane (22*32=704 pozitii de caractere). Fiecare pozitie de caracter e un patrat format din 8*8 puncte. Punctele se numesc pixeli (picture elements). Un pixel se specifica prin coordonatele sale. Coordonata "x" arata distanta fata de extrema stinga, iar coordonata "y" reprezinta distanta fata de baza ecranului. Coordonatele se scriu de obicei ca o pereche de numere, in paranteze. Astfel (0,0), (255,0), (0,175), (255,175) sint extremele stinga jos, dreapta jos, stinga sus, dreapta sus.

Instructiunea

```
PLOT        x,y
```

deseneaza punctul de coordonate x,y.

Programul:

```
10 PLOT INT (RND *256), INT(RND *175):INPUTa$:GO TO 10
```

scrie aleator un punct pe ecran de fiecare data cind se actioneaza **CR**. Programul urmator traseaza graficul functiei **SIN** pentru valori intre 0 si 2*pi.

```
10 FOR n=0 TO 255
20 PLOT n,88+80*SIN(n/128*pi)
30 NEXT n
```

Calculatorul deseneaza linii drepte, cercuri si portiuni de cerc utilizind instructiunile **DRAW** si **CIRCLE**. Cu

```
DRAW x,y
```

se poate trasa o linie dreapta. Linia incepe din punctul in care se afla cursorul ultimei instructiuni **PLOT**, **DRAW**, sau **CIRCLE**. Comenzile **RUN**, **CLEAR**, **CLS** si **NEW** il reseteaza, aducindu-l pe pozitia (0,0).

DRAW determina lungimea si directia liniei. De remarcat ca argumentele unei instructiuni **DRAW** pot fi si negative.

```
PLOT 0,100:  DRAW 80,-35
PLOT 90,150:  DRAW 80,-35
```

Calculatorul HC-85 are facilitati pentru a desena in culori. Urmatorul program demonstreaza acest lucru:

```
10 BORDER 0:  PAPER 0:  INK 7:  CLS:   REM  tot
   ecranul este negru
20 LET x1=0:  LET y1=0:  REM inceputul liniei
30 LET c=1:  REM prima culoare cu care  se  de-
   seneaza este albastru
40 LET x2=INT(RND*255):   LET  y2=INT(RND*176):
   REM capatul liniei este aleator
50 DRAW INK c;  x2-x1,y2-y1
60 LET x1=x2:  let y1=y2:  REM urmatoarea linie
   incepe de unde s-a terminat precedenta
70 LET c=c+1:  IF c=8 THEN LET c=1:   REM  alta
   culoare
80 GO TO 40
```

Comenzile **PAPER**, **INK**, **FLASH**, **BRIGHT**, **INVERSE**, **OVER** pot apare in instructiuni **PLOT** sau **DRAW** in acelasi fel in care apar in **PRINT** si **INPUT**.

Comanda **DRAW** permite si trasarea de portiuni de cercuri. Forma generala este:

```
DRAW x,y,a
```

unde x,y semnifica punctul final al liniei iar a este numarul de radiani corespunzator circumferintei. Cind a este pozitiv portiunea de cerc se traseaza in sens antiorar in timp ce, pentru a negativ se deseneaza in sens orar. Pentru a=pi se traseaza un semicerc, indiferent de valorile luate de x si y (raza este functie de punctul initial si de cel final):

```
10 PLOT 100,100: DRAW 50,50,pi
```

Trasarea cercurilor se face cu o comanda **CIRCLE** a carei forma este:

```
CIRCLE x,y,r
```

unde r este raza cercului iar (x,y) sint coordonatele centrului cercului. Ca si instructiunea **PLOT** si **DRAW**, si **CIRCLE** admite comenzi de modificare a culorii.

Functia **POINT** arata daca un pixel are asociata culoarea **INK** sau culoarea **PAPER**. Ea are doua argumente numerice care reprezinta coordonatele pixel-ului care trebuie sa fie inchis intre paranteze. Rezultatul este:

1. 0 - daca punctul are culoarea fundalului (paper).
2. 1 - daca are culoarea INK.

```
CLS:  PRINT POINT (0,0):  PLOT 0,0:   PRINT  POINT(0,0)
```

Se scrie

```
PAPER 7: INK 0
```

Intr-o instructiune **PLOT x,y, REVERSE** si **OVER** afecteaza doar pixel-ul desemnat, nu si restul pozitiilor din caracter. Deoarece aceste comenzi sint in mod normal dezactivate (0), pentru a le activa (1), trebuiesc incluse intr-o comanda **PLOT**.

Se poate face ca punctul (x,y) sa ia culoarea "ink" prin

```
PLOT x,y;
```

```
PLOT INVERSE 1;
```

face ca pixel-ul (x,y) sa ia culoarea fundalului;

```
PLOT OVER 1;  x,y
```

inverseaza culoarea pixel-ului specificat.

```
PLOT INVERSE 1;  OVER 1;  x,y
```

lasa pixel-ul nemodificat dar schimba pozitia de tiparire.

Alt exemplu de utilizare al instructiunii **OVER** este urmatorul:

-se umple ecranul scriind negru pe alb si apoi se tasteaza:

```
PLOT 0,0:  DRAW OVER 1,255,175
```

-se traseaza astfel o linie (cu intreruperi acolo unde traverseaza caracterele tiparite pe ecran).

-reexecutind comanda, linia trasata anterior o sa dispara.

Avantajul instructiunii **OVER** este ca permite sa se deseneze si apoi sa se stearga desenele fara a afecta ce se afla anterior pe ecran.

Utilizind programul

```
PLOT  0,0:  DRAW 255,175
PLOT 0,0:  DRAW INVERSE 1;  255,175
```

se constata ca aceasta comanda sterge si partile din caracterele tiparite anterior.

Daca se scrie o linie cu:

```
PLOT 0,0: DRAW OVER 1; 250,175
```

se constata ca ea nu va putea fi stearsa cu:

```
DRAW OVER 1;-250,-175
```

deoarece parcurgerea dreptei intr-un sens si in celalalt nu se face exact prin aceleasi puncte. O linie se sterge pe aceeasi directie si in acelasi sens in care a fost trasata.

Pentru a extinde gama de culori se amesteca doua culori de baza pe un singur patrat, folosind un caracter grafic definit de utilizator. Programul urmator defineste un caracter grafic echivalent unei table de sah.

```
1000 FOR n=0 TO 6 STEP 2
1010 POKE USR "a"+n, BIN 01010101: POKE USR
     "a"+n+1, BIN 10101010
1020 NEXT n
```

3.13 INSTRUCTIUNI DE INTRARE-IESIRE

Cuprins: PRINT,INPUT
Utilizarea separatorilor :, ;, TAB, AT, LINE, CLS

Expresiile folosite pentru a tipari valori cu instructiunea **PRINT** sint numite elementele instructiunii si sint separate intre ele cu virgula sau punct si virgula (separatori). Un element al instructiunii **PRINT** poate lipsi si in acest caz pot apare 2 virgule, una dupa alta.

Exista 2 elemente ale instructiunii **PRINT** care servesc la pozitionarea cursorului in vederea tiparirii. Acestea sint **AT** si **TAB**.

AT linie,coloana

deplaseaza cursorul (locul unde va fi tiparit urmatorul element) la linia si la coloana specificate. Liniile sint numerotate de la 0 la 21 (de sus in jos) si coloanele de la 0 la 31 (de la stinga la dreapta).

Exemplu

```
PRINT AT 11,16;"*"
```

imprima un asterisc in centrul ecranului. Instructiunea

TAB coloana

deplaseaza cursorul in coloana specificata. **TAB** determina deplasarea pe aceeasi linie pe care se gaseste cursorul, exceptind cazul cind pozitia de tiparire specificata se afla inaintea pozitiei de tiparire actuala; in aceasta situatie se face o deplasare la linia urmatoare.

Obs. : calculatorul considera coloanele din instructiunea **TAB** "modulo 32" (adica TAB 33 este echivalent cu TAB 1).

Exemplul de mai jos arata cum se poate tipari inceputul paginii 1 a unei carti:

```
PRINT TAB 30;1;TAB 12; "Index"; AT 3,1;
"Capitol"; TAB 24; "Pagina"
```

Un exemplu din care rezulta reducerea modulo 32 a numarului din instructiunea **TAB** este urmatorul:

```
10 FOR n=0 TO 20
20 PRINT TAB 8*n;n;
30 NEXT n
```

De retinut urmatoarele observatii:

1. Elementele de tiparire care urmeaza instructiunilor **TAB** sau **AT** sint de obicei terminate cu ";". Daca s-ar folosi "," sau nimic, cursorul, dupa ce este pozitionat, se deplaseaza.
2. Liniile 22 si 23 ale ecranului nu pot fi folosite pentru tiparire. Ele sint rezervate pentru comenzi, pentru citirea datelor, mesaje, etc.
3. Tiparind cu **AT** intr-o pozitie deja scrisa, ultima tiparire o anuleaza pe precedenta.

CLS sterge tot ecranul, functie care mai este realizata si de comenzile **CLEAR** si **RUN** (care mai executa si alte functii). Cind calculatorul, in timp ce tipareste, ajunge la ultima linie a ecranului, executa "scrolling" anulind prima linie.

Exemplu:

```
CLS: FOR n=1 TO 22: PRINT n: NEXT n
```

si apoi,

```
PRINT 99
```

de mai multe ori.

In timpul tiparirii, dupa ce calculatorul a umplut complet ecranul, se opreste scriind in partea de jos:

```
scroll ?
```

Se raspunde cu "y" sau "n".

Instructiunea **INPUT**

O linie de **INPUT** este compusa dintr-o serie de elemente si de separatori care au aceeasi functie ca într-o linie de **PRINT**. **INPUT** considera orice element care incepe cu o litera ca pe o variabila asignabila (careia urmeaza sa i se introduca valoarea de la tastatura). Instructiunea **INPUT** poate tipari si mesaje; pentru a tipari un sir de caractere este suficienta introducerea acestuia intre ghilimele. Daca contine si valori de variabile, mesajul se inchide intre paranteze.

Daca se doreste citirea unei variabile de tip sir de caractere, a$, pe ecran apare caracterul ghilimele. Daca aceasta variabila trebuie sa ia valoarea unei alte variabile de tip sir definita in program, b$, aceasta se face prin stergerea ghilimelelor si introducerea numelui variabilei (b$).

Toate elementele instructiunii **PRINT** care nu sint supuse acestor reguli pot fi elemente ale instructiunii **INPUT**.

Exemplu

```
LET virsta mea=INT( RND*100):  INPUT ("Eu am";
virsta mea; "ani." ); "citi ani ai ?"; virsta ta
```

Variabila "virsta mea" este continuta intre paranteze, deci valoarea sa se tipareste, in timp ce variabila "virsta ta" nu este intre paranteze, si deci valoarea sa se citeste de la tastatura.

O alta modalitate de citire a variabilelor sir consta in scrierea cuvintului cheie **LINE** dupa **INPUT** si inaintea variabilei sir de citit:

```
INPUT LINE a$
```

In acest caz calculatorul nu va tipari ghilimelele, care in mod normal sint tiparite cind se asteapta introducerea unei variabile sir, chiar daca se comporta ca si cum ar fi fost. Astfel, scriind carte ca variabila de intrare, a$ va lua valoarea "carte". Deoarece ghilimelele nu sint tiparite, nu este posibila introducerea altui sir. De notat ca **LINE** nu poate fi folosit pentru variabile numerice.

Caracterele de control **CHR$22** si **CHR$23** functioneaza aproape similar lui **AT** si **TAB**. Caracterul de control pentru **AT** este **CHR$22**. Primul caracter care il urmeaza specifica numarul de linie, iar al doilea numarul coloanei, astfel ca:

```
PRINT CHR$22 + CHR$1 + CHR$c;
```

este analog lui

```
PRINT AT 1,c;
```

CHR$1 si **CHR$c** (c=13) in mod normal au alta semnificatie, pe care insa si-o pierd cind urmeaza dupa **CHR$22**.

Caracterul de control echivalent lui **TAB** este **CHR$23** si cele doua caractere care-l urmeaza sint folosite pentru a indica un numar cuprins intre 0 si 65535 care specifica numarul de **TAB** ca si argumentul unei instructiuni **TAB**.

```
PRINT CHR$23 + CHR$a + CHR$b
```

este echivalent lui

```
PRINT TAB a + 256*b
```

Daca nu se doreste afisarea mesajului "scroll ?" la sfirsitul fiecarui ecran, se poate folosi:

```
POKE 23692,255
```

din cind in cind. Dupa aceasta linie calculatorul inhiba mesajul "scroll ?" pentru urmatoarele 255 linii.

3.14 CULORI

Cuprins: PAPER, INK, FLASH, INVERSE, OVER, BORDER, ATTR

Calculatorul HC-85 are facilitati color. El foloseste 8 culori (numerotate de la 0 la 7). Lista culorilor in ordinea in care sint pe tastele numerice este urmatoarea:

0 - negru
1 - albastru
2 - rosu
3 - purpuriu (magenta)
4 - verde
5 - albastru deschis
6 - galben
7 - alb

Intr-un televizor alb-negru aceste numere corespund unor tonuri de gri ordonate de la inchis spre deschis.

Orice caracter are asociate 2 culori: culoarea caracterului propriu-zis si culoarea fondului (vezi subcapitolul Setul de caractere). La pornirea calculatorului, sistemul lucreaza in alb-negru, cu caractere negre pe fond alb. Tiparirea poate fi facuta normal, dar exista si posibilitatea sa apara pe ecran pilpiind (flash). Pilpiirea se obtine inversind continuu culoarea caracterului cu culoarea fondului. Deoarece atributele de culoare si pilpiire sint asociate caracterelor (deci matricilor de 64 puncte), nu este posibil ca intr-un caracter sa fie mai mult de doua culori. Valorile acestor atribute pot fi modificate cu instructiunile **INK**, **PAPER** si **FLASH**. Forma acestor instructiuni este:

```
PAPER n
INK n
FLASH m
```

unde

1. n este un numar cuprins intre 0 si 7.
2. m este un numar binar (0 pentru inactiv si 1 pentru activ).

Pentru ilustrarea modului de folosire al instructiunilor prezentate se propune programul:

```
 20 FOR n=1 TO 10
 30 FOR c=0 TO 7
 40 PAPER c:  PRINT " ";:REM spatii colorate
 50 NEXT c:  NEXT n
 60 PAPER 7
 70 FOR c=0 TO 3
 80 INK c:  PRINT c;" ";
 90 NEXT c:  PAPER 0
100 FOR c=4 TO 7
110 INK c:  PRINT c;" ";
120 NEXT c
130 PAPER 7:  INK 0
```

In afara de aceste valori de argumente a caror semnificatie a fost deja prezentata, mai pot fi folosite valorile 8 si 9. 8 poate fi folosit ca argument pentru toate cele 4 comenzi si semnifica transparenta, fapt ce nu altereaza atributele pozitiei la tiparirea unui caracter. De exemplu:

```
PAPER 8
```

face ca la tiparirea unui caracter, culoarea fondului sa fie aceeasi cu a caracterului tiparit anterior. 9 poate fi folosit numai cu comenzile **PAPER** si **INK** si indica contrastul. Culoarea "cernelii" sau a "hirtiei" (fundalului), in functie de comanda utilizata, este facuta sa contrasteze cu cealalta, punind alb pe o culoare inchisa (negru, albastru, rosu, magenta) si negru pe o culoare deschisa (verde, bleu, galben, alb).

```
INK 9: FOR c=0 TO 7: PAPER c: PRINT c: NEXT c
```

Rulind programul

```
INK 9: PAPER 8: PRINT AT 0,0; FOR n=1 TO 1000:
PRINT n: NEXT n
```

dupa primul program din acest paragraf, culoarea cernelii este facuta mereu sa contrasteze cu vechea culoare pe care o avea fundalul in fiecare pozitie. Comanda

```
INVERSE 1
```

inverseaza fundalul cu cerneala pentru caracterul specificat.
Comanda

```
OVER 1
```

realizeaza supratiparirea. In mod obisnuit, cind ceva este scris intr-o pozitie de caracter, sterge complet ce era scris inainte; de data aceasta noul caracter va fi doar adaugat. Acest lucru este util in scrierea caracterelor compuse, cum ar fi literele cu accente. Trebuie utilizat in acest scop caracterul de control **CHR$8** pentru intoarcerea cu o pozitie.

Exista o alta posibilitate de a utiliza **INK**, **PAPER**, **FLASH**. Pot apare in **PRINT** urmate de ";" si fac exact acelasi lucru pe care l-ar face cind sint utilizate independent, exceptind faptul ca efectul lor este numai temporar.

Astfel daca se ruleaza:

```
PRINT PAPER 6; "x";: PRINT "y"
```

numai x va fi pe fond galben.

INK si celelalte comenzi nu afecteaza culorile partii de jos a ecranului. Aceasta foloseste culoarea marginii drept culoare a fundalului si codul 9 pentru a contrasta culoarea cernelii. Nu are posibilitatea de pilpiire si este cu luminozitate normala.

Marginea poate lua oricare din cele 8 culori (0-7) cu comanda

```
BORDER culoare
```

Se pot schimba culorile mesajului scris pe ecran cu comanda **INPUT**, inserind in aceasta comanda **INK**, **PAPER**, etc, ca si in cazul comenzii **PRINT**. Efectul lor este activ numai asupra comenzii urmatoare:

```
INPUT FLASH 1; INK 1; "text"; n
```

Comenzile pot fi schimbate utilizind caracterele de control ca si in cazul AT si TAB (vezi capitolul Instructiuni de intrare-iesire).

```
CHR$16 --> INK
CHR$17 --> PAPER
CHR$18 --> FLASH
CHR$20 --> INVERSE
CHR$21 --> OVER
```

Aceste caractere de control sint urmate de un caracter care indica culoarea prin intermediul codului sau. De exemplu:

```
PRINT CHR$16 + CHR$9 ;  ...
```

are acelasi efect cu:

```
PRINT INK 9;  ...
```

Functia **ATTR** are forma:

```
ATTR (linie,coloana)
```

Rezultatul este un numar care arata atributele pentru caracterul aflat la linia si coloana precizata. Numarul este suma a patru numere, conform schemei:

1. 128 - daca pozitia pilpiie , 0 daca este stabila
2. 64 - daca pozitia este stralucitoare , 0 daca este normala
3. 8*n - n = codul fundalului
4. m - m = codul cernelii

Exemplu: Pentru o pozitie pilpiitoare, normala, cu fundal galben si cerneala albastra se obtine:

```
128+0+8*6+1=177
```

3.15 MISCAREA

Cuprins: PAUSE, INKEY$, PEEK

Pentru a realiza o pauza in program in timpul careia nu se desfasoara nici o operatie se foloseste comanda:

```
PAUSE n
```

care opreste executia programului mentinind activ display-ul pe durata a n perioade de baleiaj ale ecranului (20 ms pentru fiecare ecran); n poate lua valoarea maxima 65535, careia ii corespunde o pauza de aproximativ 22 minute. Daca n=0 se opreste definitiv.

O pauza obtinuta in acest mod poate fi scurtata apasind orice tasta (cu exceptia lui **SPACE** si **CAPS SHIFT** care produce **BREAK**).

Programul urmator deseneaza cadranul unui ceas pe care se misca secundarul:

```
 10 REM Mai intii e desenat cadranul.
 20 FOR n=1 TO 12
 30 PRINT AT 10-10*COS( n/PI), 16+10*SIN( n/PI)
 40 NEXT n
 50 REM Se porneste ceasul.
 60 FOR t=0 TO 200000; :REM t e timpul in secunde
 70 LET a=t/30*PI:  REM a este unghiul secundaru-
    lui in radiani
 80 LET sx=80*SIN( a):  LET sy=80*COS( a)
200 PLOT 128,88:  DRAW OVER 1;  sx, sy:  REM  Se
    deseneaza secundarul
210 PAUSE 42
220 PLOT 128,88:  DRAW OVER 1;  sx, sy:  REM  Se
    sterge secundarul
230 NEXT t
```

Cu linia 210 se marcheaza trecerea unei secunde; s-a folosit n=42 si nu n=50 deoarece calculatorul foloseste un timp pentru scrierea liniilor ciclului **FOR - NEXT**; linia 210 opreste calculatorul doar pentru timpul care mai ramine.

O temporizare mai precisa se poate realiza citind continutul anumitor locatii de memorie cu **PEEK**. Expresia urmatoare:

```
(65536 *PEEK 23674+ 256*PEEK 23673+ PEEK 23672)/50
```

da numarul de secunde scurse de la aprinderea calculatorului pina la 3 zile si 21 ore, dupa care se reseteaza. Programul unui ceas mai precis este dat in continuare:

```
 10 REM Se deseneaza cadranul
 20 FOR n=1 TO 12
 30 PRINT AT 10-10*cos(n/6*pi),16+10*SIN(n/6*PI);n
 40 NEXT n
 50 DEF FNt()= INT(65536* PEEK 23674+ 256*  PEEK
    23673+ PEEK 23672)/50:  REM Numarul de secunde de
    la inceput
100 REM se porneste ceasul
110 LET t1=FNt()
120 LET a=t1/30*PI:  REM a este unghiul in radi-
    ani
130 LET sx=72* SIN a:  LET sy=72*COS a
140 PLOT 131,91:  DRAW OVER 1;  sx;  sy:  REM
    Se deseneaza secundarul
200 LET t=FNt()
210 IF t=t1 THEN GO TO 200
220 PLOT 131,91:  DRAW OVER 1;  sx;  sy:  REM Se
    sterge vechiul secundar
230 LET t1=t:  GO TO 120
```

Acest ceas se opreste temporar de cite ori se executa **BEEP** ori se utilizeaza imprimanta, casetofonul. Numerele **PEEK** 23674, **PEEK** 23673 si **PEEK** 23672 sint folosite pentru a numara in incremente de 20 ms. Fiecare variaza de la 0 la 255, dupa care se reincepe. Cel mai rapid se incrementeaza locatia 23672 (cu 1 la fiecare 20 ms); cind se trece de la 255 la 0, locatia 23673 se incrementeaza cu 1; analog pentru 23674. Presupunind ca cele 3 numere sint 0 (pentru **PEEK** 23674), 255 (pentru **PEEK** 23673) si 255 (pentru **PEEK** 23672), au trecut deci circa 21 minute de la pornirea calculatorului. Expresia devine:

(65536*0+256*255+255)/50=1310.7

Pentru a pozitiona ceasul pe ora 10 se procedeaza astfel:

10*60 *60 *50=1800000= 65536*27 +256*119 +64

si se memoreaza numerele 27, 119 si 64 cu:

POKE 23674,27: POKE 23673,119: POKE 23672,64

Functia **INKEY$**, fara argument, da caracterul apasat pe tasta in momentul apelarii sale. Cu programul urmator calculatorul devine o masina de scris:

```
10 IF INKEY$ > "" THEN GO TO 10
20 IF INKEY$ = "" THEN GO TO 20
30 PRINT INKEY$;
40 GO TO 10
```

Linia 10 asteapta sa se elibereze ultima tasta apasata: linia 20 asteapta apasarea uneia noi. Spre deosebire de **INPUT**, **INKEY$** nu asteapta apasarea lui CR sau a unei taste.

3.16 MEMORIA

Cuprins: CLEAR

Fiecarui octet ii este asociata o adresa care este un numar intre 0 si FFFFH.
Memoria este impartita in trei zone distincte:

1. 0 - 4000H zona ROM
   in aceasta zona se gaseste memoria ROM in care este inregistrat interpretorul BASIC.

2. 4000H - 7FFFH zona RAM video
   in aceasta zona se gaseste memoria video cit si o parte din memoria RAM de program.

3. 8000H - FFFFH zona RAM suplimentar
   aceasta zona nu este neaparat necesara. Ea este folosita pentru marirea capacitatii de memorie. Ea difera de zona video printr-un timp de acces mai mic.

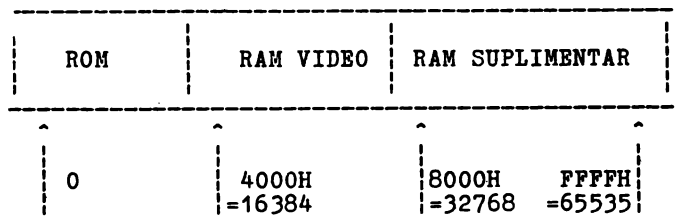


Fig. 3.1

Continutul memoriei poate fi vizualizat cu functia **PEEK** care are ca argument o adresa. Exemplul urmator vizualizeaza primii 21 octeti din memoria ROM si adresele lor:

```
10 PRINT "Adresa";  TAB 10;  "Octet"
20 FOR a=0 TO 20
30 PRINT a TAB 10;  PEEK a
40 NEXT a
```

Schimbarea continutului memoriei RAM se poate face cu instructiunea **POKE**, care are forma:

```
POKE adresa, continut nou
```

unde "adresa" si "continut nou" sint expresii numerice.

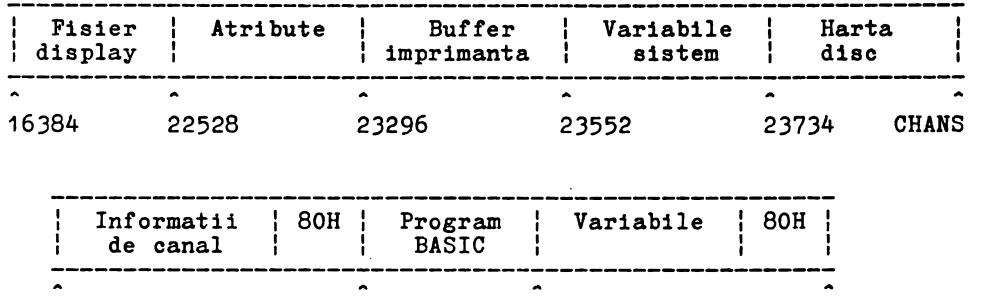
```
POKE 31000, 57
```

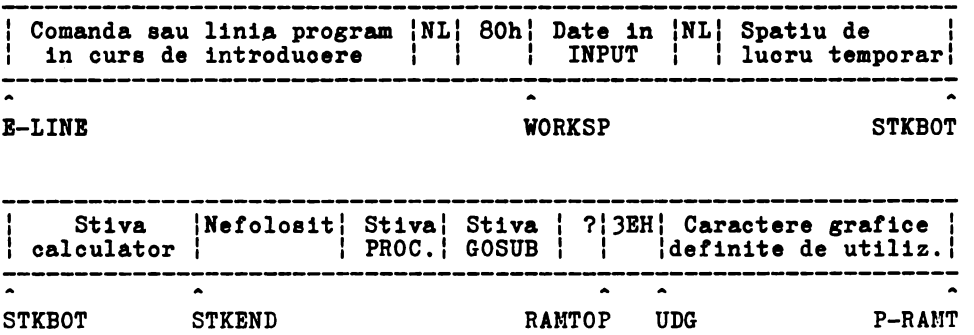
determina incarcarea valorii 57 la adresa 31000. Cu

```
PRINT PEEK 31000
```

se va tipari 57. "Continut nou" trebuie sa aiba valoarea intre -255 si 255. Daca e numar negativ, se aduna 256.

De importanta pentru utilizator este organizarea memoriei RAM. Memoria este impartita in zone specifice stocarii unui anumit gen de informatie. Zonele sint suficient de mari pentru ca informatia continuta actualmente sa poata fi reorganizata daca se insereaza ceva intr-un anumit punct (de exemplu prin adaugarea unei linii de program sau a unei variabile). La inserare, spatiul necesar este creat prin mutarea in sus a tot ce se afla deasupra. Daca se sterge informatie, atunci totul este mutat in jos.





Variabilele sistem (**PROG**, **CHANS**, **VARS**, **ELINE**, etc.) contin diferite informatii necesare pentru gestiunea interna a memoriei. Ele indica limitele pentru diverse zone de memorie. Ele nu sint variabile BASIC si deci nu pot fi recunoscute de calculator.

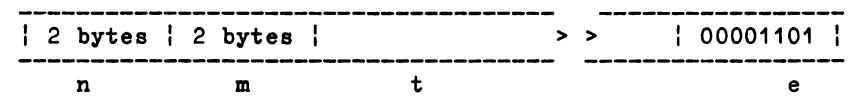
Fisierul display stocheaza imaginea televizorului. In loc de **PEEK** si **POKE**, pentru imaginea display-ului se pot utiliza **SCREEN$** si **PRINT AT** sau **PLOT** si **POINT**.

Atributele sint culorile, etc. pentru fiecare pozitie de caracter (se afla cu instructiunea **ATTR**). Ele sint stocate linie cu linie in ordinea dorita.

Buffer-ul imprimantei stocheaza caractcrele destinate imprimantei.

Informatiile de canal sint necesare cind se lucreaza cu dispozitive de intrare-iesire. Si lucrul cu tastatura necesita aceasta zona deoarece partea de jos a ecranului functioneaza ca un port de intrare, in timp ce restul ecranului se comporta ca un port de iesire.

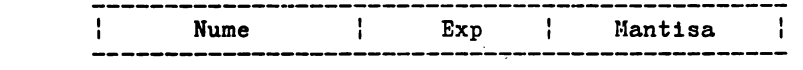
Orice linie de comanda are forma:



unde:

1. n - este numarul liniei curente
2. m - este lungimea textului + CR
3. t - este textul liniei
4. e - este codul caracterului CR

Modul de memorare al variabilelor numerice este:



unde:

1. Nume - este un numar de octeti egal cu numarul de caractere ce formeaza identificatorul variabilei
2. Exp - este un octet ce contine exponentul numarului
3. Mantisa - este un grup de 4 octeti ce contine mantisa numarului. Bitul cel mai semnificativ al primului octet este bitul de semn.

3.17 PRODUCEREA SUNETELOR

Cuprins: BEEP

Pentru producerea sunetelor, se foloseste instructiunea:

```
BEEP d,i
```

unde:

1. d - este o expresie numerica ce indica durata in secunde a sunetului respectiv
2. i - este o expresie numerica ce reprezinta inaltimea sunetului, masurat in semitonuri relativ la DO central.

Pentru a transcrie muzica este indicat sa se scrie pe marginea fiecarui spatiu si linie a portativului inaltimea corespunzatoare, tinind cont de armura cheii.

Exemplu:

```
10 PRINT "Frere Gustav"
20 BEEP 1,0:BEEP 1,2:BEEP .5,3:BEEP .5,2:BEEP 1,0
30 BEEP 1,0:BEEP 1,2:BEEP .5,3:BEEP .5,2:BEEP 1,0
40 BEEP 1,3:BEEP 1,5:BEEP 2,7
50 BEEP 1,3:BEEP 1,5:BEEP 2,7
60 BEEP .75,7:BEEP .25,8:BEEP .5,7:BEEP .5,5:BEEP .5,3:
   BEEP .5,2:BEEP1,0
70 BEEP .75,7:BEEP .25,8:BEEP .5,7:BEEP .5,5:BEEP .5,3:
   BEEP .5,2:BEEP1,0
80 BEEP 1,0:BEEP 1,-5:BEEP 2,0
90 BEEP 1,0:BEEP 1,-5:BEEP 2,0
```

Pentru alcatuirea programului s-a procedat dupa cum urmeaza:

1. s-au adaugat mai intii deasupra si dedesubt cite o linie de referinta
2. s-au numerotat liniile si spatiile, observind ca mi bemol din armura cheii afecteaza nu numai mi de sus (coborindu-l de la 16 la 15) cit si mi de jos (coborindu-l de la 4 la 3)

Pentru a schimba cheia partiturii, trebuie sa se adune la inaltimea fiecarei note o variabila (de exemplu "Cheie") careia trebuie sa i se atribuie valoarea adecvata inaintea executiei piesei.

Linia 20 a programului devine:

```
20 BEEP 1, Cheie 0:BEEP1
```

In acest exemplu variabila "Cheie" trebuie sa aiba valoarea 0 pentru DO minor, 2 pentru RE minor, 12 pentru DO minor in octava superioara, etc.

Cu acest sistem este posibila acordarea calculatorului cu un alt instrument, folosind valori zecimale pentru variabila "Cheie". De asemenea, este posibil sa se execute piese cu viteze diferite. In exemplul dat "o patrime" a fost programata sa dureze o secunda. Daca se introduce o variabila "PATRIME" analog cu "Cheie", linia 20 devine:

```
20    BEEP patrime, cheie+0:   BEEP patrime,
      cheie+2:BEEP patrime/2, cheie+3:BEEP patrime/2,
      cheie+2:BEEP patrime,cheie+0
```

In acest fel este posibila executia aceluiasi program in orice cheie, cu orice acordare.
Programul de mai jos:

```
FOR n=0 TO 1000:  BEEP 0.5 , n:  NEXT n
```

va produce note din ce in ce mai acute, pina la limita posibilitatilor calculatorului, cind acesta va tipari mesajul:

```
B    integer out of range
```

Tiparind n se obtine inaltimea notei celei mai acute care poate fi produsa. Procedeul poate fi repetat pentru notele joase.
Sunetele din gama medie sint cele mai potrivite pentru a fi redate.
Sunetele grave se aud ca niste pacanituri. Ele pot fi prelungite pentru a deveni mai naturale, cu comanda:

```
POKE 23609, m
```

cu m=0,...,255.

3.18 UTILIZAREA CODULUI MASINA

Cuprins: USR

Calculatorul HC-85 poate fi dotat cu un asamblor inregistrat pe caseta. Introducerea programului scris in limbaj masina (functie executata in general de asamblor) se face in general cu specificarea adresei de inceput (cel mai bine este ca aceasta adresa sa se afle intre zona BASIC si zona caracterelor grafice definite de utilizator).
La pornirea unui calculator HC-85 inceputul memoriei RAM, **RAMTOP** se afla la adresa 65366 (vezi fig. 3.2), dar se poate deplasa **RAMTOP** cu comanda **CLEAR 65266** obtinindu-se neutilizarea de catre sistem a 100 octeti incepind cu adresa 65267 (vezi fig. 3.3).

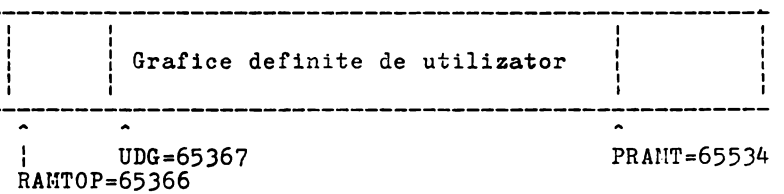


Fig. 3.2

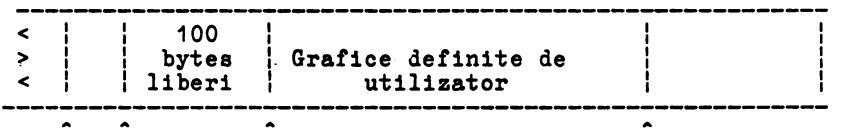


Fig 3.3

Pentru a insera codurile obiect in memorie, se poate utiliza un program de genul:

```
10 LET a=32500
20 READ n:  POKE a,n
30 LET a=a+1:  GO TO 20
40 DATA 1,99,0,201
```

care introduce programul:

```
LD bc,99
RET
```

transpus in cod masina ca:

1, 99, 0 (pentru LD bc,99) si 201 (pentru RET).

Cind se termina cei 4 octeti specificati, apare mesajul:

E Out of DATA

Rularea programului introdus in cod masina se face cu instructiunea:

USR adresa de inceput

In exemplul de mai sus, cu:

```
PRINT USR 32500
```

se tipareste valoarea 99 din perechea de registre BC.

Adresa de revenire in BASIC se memoreaza cu instructiunea Z80 **RET**. In rutinele scrise in limbaj masina nu se pot folosi registrele index IY si IX.

Calculatorul HC-85 are scoase in exterior magistralele de date, de adrese si de control prin intermediul unui conector de extensie.

Un program in limbaj masina poate fi memorat ca o informatie de tip byte; deci cu:

```
SAVE "nume" CODE 32500,4
```

se memoreaza programul exemplu.

Un program in limbaj de asamblare nu se poate lansa automat, odata incarcat; el poate fi insa lansat de un program in BASIC ca in exemplul:

```
10 LOAD "" CODE 32500,4
20 PRINT USR 32500
```

Dupa aceasta se executa:

```
SAVE "nume" LINE
```

si apoi

```
SAVE "nume" CODE 32500,4
```

Rebobinind caseta si scriind:

```
LOAD "nume"
```

se incarca si se executa programul BASIC care, la rindul sau, va apela programul in limbaj masina.

3.19 UTILIZAREA PORTURILOR INPUT, OUTPUT

Cuprins: IN ,OUT

Calculatorul HC-85 dispune de 65536 adrese de memorie de tip RAM si ROM organizate pe opt biti. El poate sa scrie cuvinte in memoria de tip RAM si poate sa citeasca cuvinte din memoriile de tip RAM si ROM. Analog sint 65536 porturi de INPUT si de OUTPUT. Aceste porturi sint folosite de procesor pentru a comunica cu exteriorul. Instructiunile sint:

```
IN adresa port
```

care preia bitul citit de la acel port;

```
OUT adresa port, valoare
```

inscrie valoarea in portul de adresa specificat. Exista un ansamblu de adrese de intrare care citeste tastatura si conectorul de casetofon. Tastatura este impartita in 8 semipagini de 5 taste fiecare. Lista porturilor utilizate este:

```
IN 65278 citeste semipagina CAPS SHIFT - v
```

Aceste adrese sint 254+256*(255-2↑n) cu n=0,...,7

Bitii d0,..,d4 sint asociati celor 5 taste din semipagina specificata. D6 este asociat conectorului de casetofon.

Portul de iesire cu adresa 254 controleaza difuzorul (D4), conectorul de casetofon (D3) si determina culoarea chenarului (D2, D1, D0). Portul de adresa 251 controleaza imprimanta in scriere si citire; la citire verifica daca imprimanta este gata sa imprime o noua linie si la scriere trimite linia care trebuie sa fie tiparita. Porturile de adrese 254, 247 si 239 sint folosite pentru echipamentele suplimentare (capitolul Alte periferice).

3.20 INREGISTRAREA PE CASETA

Cuprins: SAVE, VERIFY, LOAD, MERGE

Calculatorul HC-85 are posibilitatea sa inregistreze programe de pe banda magnetica cu orice casetofon.

Conectarea calculatorului la casetofon se face cu ajutorul unui cablu special.

Pentru a memora un program pe banda, acesta trebuie sa primeasca un nume compus din maximum 10 caractere, litere si/sau cifre. Comanda este:

Save "nume"

Calculatorul raspunde cu mesajul:

Start tape then press any key.

La terminarea inregistrarii apare mesajul:

0 OK.

Pentru verificare se regleaza volumul casetofonului la nivel mediu si se conecteaza cablul; se pozitioneaza banda in punctul in care a inceput inregistrarea. Comanda este:

VERIFY "nume"

In acest fel se verifica daca programul si variabilele inregistrate pe caseta sint identice cu cele din memoria calculatorului. Daca programul a fost inregistrat si chemat corect, pe ecran apare:

Program "nume"

(In timpul cautarii programului specificat, calculatorul tipareste numele tuturor programelor pe care le intilneste) si la sfirsit mesajul:

0 OK.

In cazul unei erori de inregistrare (eroare ce apare la VERIFY) se afiseaza mesajul:

R Tape loading error

si se incearca o noua inregistrare. Incarcarea unui program memorat pe caseta se face cu comanda:

LOAD "nume"

Aceasta comanda sterge vechiul program (si variabilele sale) din calculator inainte de a incarca unul nou.

LOAD ""

fara a fi urmat de un nume de program incarca primul program gasit pe caseta.

Comanda **MERGE** incarca un program inregistrat pe caseta in memoria calculatorului, dar spre deosebire de comanda **LOAD**, anuleaza din vechiul program, inaintea inceperii transferului doar acele linii si variabile cu numere, respectiv nume deja existente in programul ce urmeaza a fi incarcat. Daca instructiunile **VERIFY**, **LOAD** si **MERGE** sint urmate de un sir vid ca nume al fisierului cautat, calculatorul va lucra asupra primului program pe care il intilneste.

Este posibil sa se inregistreze un program pe caseta, astfel incit atunci cind este reincarcat in memorie, el se lanseaza automat de la o linie specificata. Instructiunea este:

SAVE sir LINE numar

si face ca programul incarcat cu **LOAD** (dar nu si cu **MERGE**) sa fie rulat automat de la linia specificata cu "numar". Daca nu este loc suficient in memorie, programul vechi si vechile variabile nu sint sterse si apare eroare:

Out of memory

In afara de programe si variabile se mai pot memora matrici si octeti. Pentru memorarea unei matrici se foloseste instructiunea:

SAVE sir DATA matrice()

unde:

1. sir - este numele de pe banda al matricii
2. matrice - specifica numele matricii care va fi memorata (numerica sau sir de caractere).

Exemple:

SAVE "test" DATA b()

In acest caz se cauta pe caseta o matrice cu numele "test". Cind o gaseste trimite mejajul :

Number array: test

Matricea gasita este comparata cu matricea B din memorie.

LOAD "test" DATA b()

Se cauta matricea pe banda si daca este memorie libera suficienta, anuleazea o eventuala matrice B preexistenta, si incarca noua matrice pe banda denumind-o B.

MERGE nu poate fi folosit la inregistrarea matricilor pe banda.

Memorarea tip octet este folosita pentru orice tip de data, fara vreo referire asupra utilizarii acestei date. Memorarea tip octet se face cu:

SAVE sir CODE primul octet, numarul de octeti

Acest mod de memorare copiaza o parte din memoria interna a calculatorului, asa cum este, pe banda.

Transferul in sens invers se face cu:

LOAD sir CODE adresa de inceput, lungime

Cind nu se specifica lungimea sirului de octeti, calculatorul va incarca toti octetii inregistrati pe caseta.

Exemplu:

Zona de memorie in care se pastreaza imaginea pentru display incepe la adrasa 16384 si are 6912 octeti. Comanda:

SAVE "imagine" CODE 16384,6912

copiaza imaginea de pe ecran in momentul executiei comenzii, pe banda, cu numele imagine.
CODE 16384,6912 este folosita frecvent; de aceea a fost abreviata sub forma:

SCREEN$

La memorarea imaginii video nu poate fi folosita comanda **VERIFY**.

3.21 IMPRIMANTA

Cuprins: LLIST, LPRINT, COPY

Comenzile **LPRINT** si **LLIST** sint identice cu **PRINT** si **LIST**, tiparind pe imprimanta, nu pe televizor.
Comanda **COPY** tipareste la imprimanta o copie a ecranului televizorului. **COPY** nu are efect in cazul listarilor automate (de cite ori se apasa CR).
Pentru a obtine un listing se poate folosi **LIST** urmat de **COPY** sau numai **LLIST**. Imprimanta poate fi oprita in timpul unei tipariri actionind **BREAK**.

3.22 VARIABILE DE SISTEM

Octetii din memorie de la adresa 23552 la adresa 23733 sint rezervati pentru operatii specifice ale sistemului. Ei pot fi cititi pentru a afla diferite lucruri despre sistem, iar citiva din ei pot fi si modificati. Acesti octeti se numesc variabile de sistem, si au cite un nume, dar nu trebuie confundati cu variabilele utilizate de BASIC. In cazul variabilelor formate din mai multi octeti, primul va fi octetul cel mai putin semnificativ. Variabilele de sistem sint date in lista de mai jos. Abrevierile din coloana 1 au urmatoarea semnificatie:

- X aceasta variabila nu poate fi modificata deoarece sistemul va functiona eronat
- N modificarea acestei variabile nu are un efect asupra functionarii normale a sistemului
- n numarul de octeti din variabila

Tip	Adresa	Nume	Continut
N8	23552	KSTATE	Folosita in citirea tastaturii
N1	23560	LAST K	Retine ultima tasta apasata
1	23561	REPDEL	Durata (in 1/50 sec) cit trebuie tinuta apasata o tasta pentru a se repeta
1	23562	REPPER	Timpul (in 1/50 sec) dupa care se repeta o tasta apasata
N2	23563	DEFADD	Adresa argumentelor functiilor definite de utilizator
N1	23565	K DATA	Al doilea octet pentru controlul culorii introdus de la tastatura
N2	23566	TVDATA	Controlul culorii, al lui AT si TAB pentru TV
X38	23568	STRMS	Adresa canalului atasat caii
2	23606	CHARS	Adresa generatorului de caractere minus 256
1	23608	RASP	Durata sunetului de eroare
1	23609	PIP	Durata sunetului la apasarea unei taste
1	23610	ERR NR	Codul de mesaj minus 1
X1	23611	FLAGS	Diferiti indicatori de control ai sistemului BASIC
X1	23612	TVFLAG	Indicatori asociati cu televizorul
X2	23613	ERR SP	Adresa elementului din stiva masinii utilizat ca adresa de intoarcere in caz de eroare
N2	23615	LIST SP	Adresa de intoarcere la listarile automate
N1	23617	MODE	Specifica cursorul (K,L,C,E,G)
2	23618	NEWPPC	Linia la care se sare
1	23620	NSPPC	Numarul instructiunii in linie la care se sare
2	23621	PPC	Numarul liniei pentru instructiunea in executie
1	23623	SUBPPC	Numarul instructiunii din linie in executie
1	23624	BORDCR	Culoarea border-ului
2	23625	E PPC	Numarul liniei curente
X2	23627	VARS	Adresa variabilelor
N2	23629	DEST	Adresa variabilelor asignate
X2	23631	CHANS	Adresa datelor de canal
X2	23633	CURCHL	Adresa informatiei curente folosita pentru intrare sau iesire
X2	23635	PROG	Adresa programului BASIC
X2	23637	NXTLIN	Adresa urmatoarei linii din program
X2	23639	DATADD	Adresa ultimului element din lista DATA
X2	23641	E LINE	Adresa comenzii introduse
2	23643	K CUR	Adresa cursorului
X2	23645	CH ADD	Adresa urmatorului caracter care urmeaza sa fie interpretat
2	23647	XPTR	Adresa caracterului dupa semnul intrebarii
X2	23649	WORKSP	Adresa spatiului de lucru temporar
X2	23651	STKBOT	Adresa inferioara a stivei calculator
X2	23653	STKEND	Adresa de inceput a spatiului liber
N1	23655	BREG	Registrul B al calculatorului
N2	23656	MEM	Adresa spatiului folosit pentru memoria calculatorului
1	23658	FLAGS2	Alti indicatori
X1	23659	DF SZ	Numarul liniilor din partea de jos a ecranului

2	23660	S TOP	Numarul liniei de sus a programului la listarea automata
2	23662	OLDPPC	Numarul liniei la care sare CONTINUE
1	23664	OSPCC	Numarul din linie la care sare CONTINUE
N1	23665	FLAGX	Diversi indicatori
N2	23666	STRLEN	Lungimea asignata sirului
N2	23668	T ADDR	Adresa urmatorului element din tabela sintaxa
2	23670	SEED	Variabila pentru RND
3	23672	FRAMES	Contorul de cadre
2	23675	UDG	Adresa primului grafic definit de utilizator
1	23677	COORDS	Coordonata x a ultimului punct plot-at
1	23678		Coordonata y a ultimului punct plot-at
1	23679	P POSN	Numarul pozitiei de scriere pe ecran
1	23680	PR CC	Octetul mai putin semnificativ al adresei pentru noua pozitie la care se imprima prin LPRINT
1	23681		Nefolosit
2	23682	ECHO E	Numarul coloanei si al liniei
2	23684	DF CC	Adresa de afisare pe ecran prin PRINT
2	23686	DFCCL	Acelasi lucru pentru partea de jos a ecranului
X1	23688	S POSN	Numarul coloanei pentru PRINT
X1	23689		Numarul liniei pentru PRINT
X2	23690	SPOSNL	Ca S POSN pentru partea de jos a ecranului
1	23692	SCR CT	Numara defilarile de ecran
1	23693	ATTR P	Culoarea curenta
1	23694	MASK P	Folosit pentru culori transparente
N1	23695	ATTR T	Culori temporare
N1	23696	MASK T	Ca MASK P dar temporar
1	23697	PFLAG	Alti indicatori
N30	23696	MEMBOT	Arie memorie calculator
2	23728		Nefolosit
2	23730	RAMTOP	Adresa ultimului din aria sistemului BASIC
2	23732	P-RAMT	Adresa ultimului octet de RAM

3.23 CANALE I/O SI CAI

Cuprins: INPUT#, PRINT#, OPEN#, CLOSE#, LIST#, INKEY$#

Pentru fiecare echipament periferic sau port I/O este asignata o linie de comunicatie numita canal. Fiecarui canal existent i se poate asocia o parte componenta software numita cale. Pentru a transmite informatii pe un canal oarecare este suficient sa transmitem informatiile pe calea asignata acestui canal.

Exemplu:

INPUT# s; 'lista variabile'

citeste date de la portul asignat caii s si le asociaza variabilelor din lista de variabile. Similar

PRINT# s; 'lista variabile'

trimite date catre portul asociat caii s.

Asignarea unei cai la un echipament I/O se face cu instructiunea **OPEN# s,c** unde:

s este numarul caii
c este un sir care specifica canalul

Instructiunea **OPEN#** realizeaza si initializarea echipamentului I/O. Unui canal i se pot asocia mai multe cai.

In configuratia de baza calculatorul HC - 85 recunoaste trei canale:

canalul K - claviatura
canalul S - ecran
canalul P - imprimanta

Canalele S si P sint canale pe care se poate doar scrie la echipamentul I/O.

Exemplu:

```
10 OPEN# 5,"K"
20 PRINT# 5,"hc 85"
30 GO TO 20
```

trimite date la iesirea caii 5 care este asociata prin instructiunea **OPEN#** partii de jos a ecranului.

Pentru a anula asignarea caii s la un canal se foloseste instructiunea **CLOSE# s**. Dupa instructiunea **CLOSE#** calea s poate fi asociata altui canal.

La initializarea sistemului se deschid automat caile 0-3, cu urmatoarea asignare:

calea 0 - canalul K
calea 1 - canalul K
calea 2 - canalul S
calea 3 - canalul P

Instructiunea **LIST# s,n** listeaza programul incepind cu linia n pe calea s.

Comanda **INKEY$# s** citeste un octet de pe calea s.

3.24 ALTE ECHIPAMENTE

Retea

Poate fi folosita o periferie de tip retea pentru conectarea mai multor calculatoare HC-85 intre ele.

Interfata seriala

Interfata standard RS-232 permite conectarea unui HC-85 cu alt calculator sau alte periferice inzestrate cu aceasta interfata. Utilizarea se realizeaza folosind cuvintele cheie **OPEN#, CLOSE#, MOVE, ERASE, CAT** si **FORMAT**.

4. FUNCTIONAREA CALCULATORULUI HC-85

4.1 SCHEMA BLOC

HC-85 este un microcalculator pe o singura placa construit in jurul microprocesorului pe 8 biti Z80A.

In figura 4.1 este prezentata schema bloc a microcalculatorului. Unitatea centrala de prelucrare Z80A formeaza blocul central notat 1. Memoria RAM este alcatuita din doua blocuri :

- un bloc de 16K RAM - memoria video si de program notat 4.
- un bloc de 32K RAM - extinde capacitatea memoriei la 48 K.

Microcalculatorul mai cuprinde :

- un bloc de memorie ROM de 16K notat 2, continind interpretorul BASIC.
- interfete pentru claviatura, casetofon, difuzor notate 8.
- interfata pentru televizor alb negru sau color.

Pe placa de baza se afla si convertorul pentru tensiunile de +12V, -5V, care se formeaza din tensiunea oferita de un alimentator extern.

Stabilizatorul de +5V se afla asezat pe un radiator montat pe sasiu.

Tastatura se afla pe o placa separata care se conecteaza la placa de baza printr-un cablu de 16 fire.

Modul de functionare al calculatorului se poate urmari pe schema bloc.

Unitatea centrala (1) avind la baza microprocesorul Z80A executa instrutiuni stocate in memorie. Interpretorul BASIC situat in ROM asigura posibilitatea lucrului in acest limbaj inca de la punerea sub tensiune.

Microprocesorul acceseaza prin intermediul magistralelor de date, adrese, control dispozitivele conectate la aceste magistrale, memoria si dispozitivele de intrare/iesire (I/O).

Blocul memorie de afisare si program (4) poate fi accesat atit de microprocesor cit si de sincrogenerator (5), care genereaza imaginea TV.

Blocul de control (6) asigura selectia pentru memoria RAM de afisare si program. Interfata video (7) converteste informatia din memoria de afisare in semnale seriale R, G, B, semnale care codificate PAL in blocul (9) si modulate permit afisarea informatiei pe un televizor alb-negru sau color.

Blocul de memorie de 32K (4) este accesat numai de microprocesor si lucreaza independent de sincrogenerator.

Blocul memoriei video si de program contine la inceputul lui o zona de 6912 octeti reprezentind memoria de ecran. Cind se afiseaza imaginea TV aceasta arie este accesata de sincrogenerator. CPU acceseaza si el aceasta zona de ecran ori de cite ori vrea sa modifice imaginea de pe ecran sau cind lucreaza cu programe memorate in blocul de memorie de 16K la adrese superioare spatiului de ecran.

Din acest motiv magistralele memoriei de afisare si program sint folosite atit de CPU cit si de sincrogenerator, acesta din urma avind prioritate. La accesarea primilor 16 Kocteti de RAM procesorul trebuie sa astepte terminarea ciclului curent de afisare. Magistralele CPU sint separate de magistralele sicrogeneratorului, astfel incit CPU lucreaza la viteza normala cind acceseaza alte resurse hard (ROM, RAM suplimentar, I/O).

In figura 4.1 se poate observa separarea magistralelor de adrese printr-o bariera rezistiva notata R si separarea magistralelor de date prin circuite separatoare notate B.

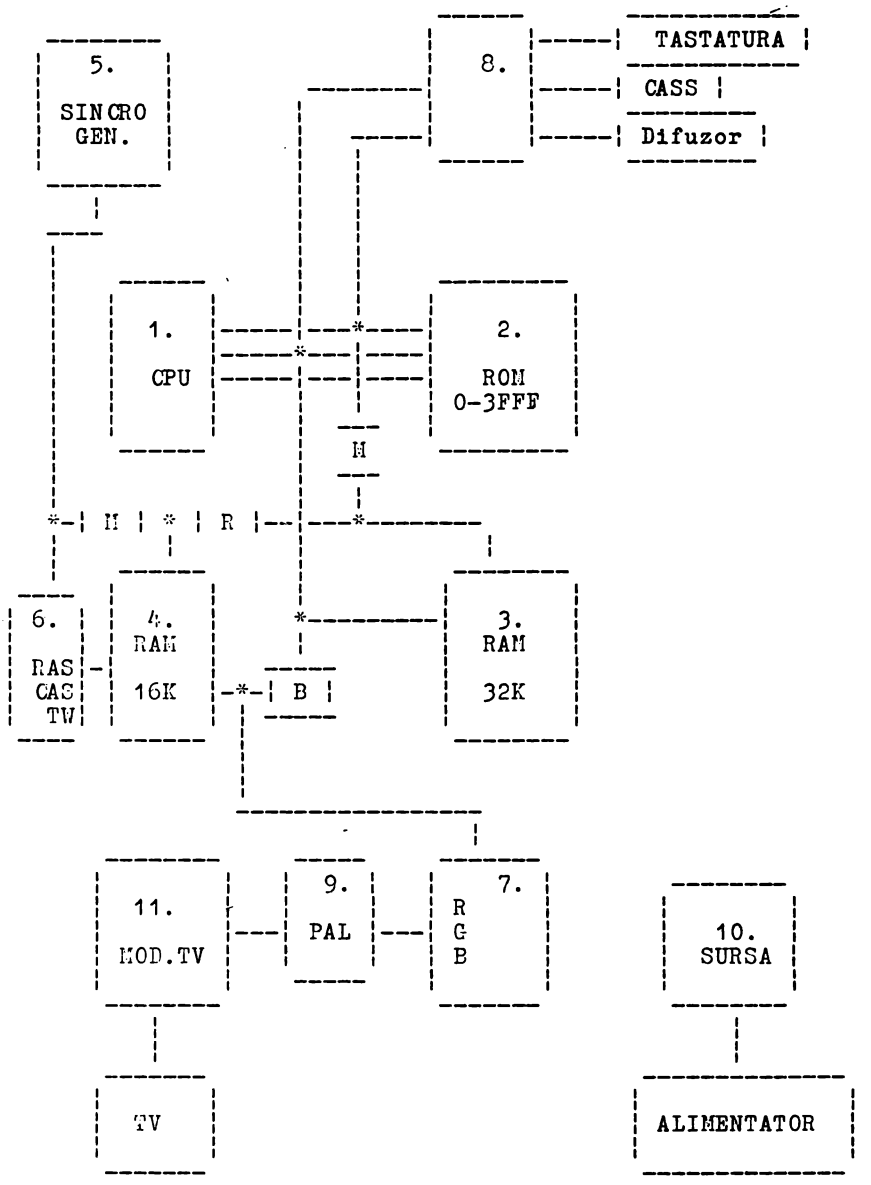


Fig. 4.1 Schema bloc a calculatorului HC-85

4.2 UNITATEA CENTRALA DE PRELUCRARE (vezi fila 1/10) CPU

Unitatea centrala de prelucrare este constituita din microprocesorul pe 8 biti Z80A.

Z80A este un circuit MOS-LSI in capsula 40 pini, cu 3 magistrale:

- magistrala de date (DATA BUS)
- magistrala de adresa (ADDRESS BUS)
- magistrala de comenzi (CONTROL BUS)

Magistrala de date D0 - D7 este o magistrala bidirectionala, 3 stari, utilizata pentru schimb de informatie cu memoria si circuitele de interfata I/O.

Z80A intra in categoria microprocesoarelor pe 8 biti, avind posibilitatea de a prelucra 8 biti de informatie simultan pe magistrala sa de date.

Magistrala de adrese de 16 biti, este utilizata pentru selectia memoriei sau a dispozitivelor de I/O pe durata schimburilor de informatie.

Avind 16 biti pentru magistrala de adrese Z80A poate adresa 64K de memorie si un spatiu aditional de 64K dedicat dispozitivelor de intrare-iesire.

Magistrala de comenzi ofera semnalele necesare pentru a asigura transferul datelor de la sau catre microprocesor.

Microprocesorul poate executa mai multe functii:

- citeste date din memorie
- scrie date in memorie
- citeste date de la echipament I/O
- scrie date la echipament I/O
- executa operatii aritmetice asupra datelor

Z80A executa un repertoriu de 158 tipuri de instructiuni. Ceasul microprocesorului este de 3.5 MHz.

Descrierea pinilor

A0-A15 - magistrala de adrese
- iesiri 3 stari, active pe 1 logic
- poate adresa pina la 64 K octeti memorie si echipamente I/O.
- in cazul I/O, 8 biti mai putin semnificativi de adresa sint folositi pentru selectia a pina la 256 dispozitive de intrare sau 256 dispozitive de iesire.
- in timpul ciclului de improspatare pentru memoria dinamica (refresh) 7 biti mai putin semnificativi contin adresa de improspatare.

D0-D7 - magistrala de date
- bidirectionala, intrari/iesiri 3 stari, active 1 logic

M1 - ciclu masina nr. 1
- iesire activa 0 logic
- indica ca microprocesorul primeste din memorie codul instructiunii
- M1 si IOREQ indica un ciclu de recunoastere intrerupere

MREQ - cerere de memorie
- iesire 3 stari activa 0 logic
- indica adresa valida pentru un ciclu de citire sau scriere din memorie

IOREQ - cerere de I/O
- iesire 3 stari, activa pe 0 logic
- indica adresa inferioara valida pentru operatii I/O

RD - citire
- iesire 3 stari, activa pe 0
- indica o operatie de citire din memorie sau de la echipament I/O

WR - scriere
- iesire 3 stari, activa pe 0
- indica date valide pe magistrala de date, care pot fi inscrise in memorie sau echipament I/O

RFSH - improspatare
- iesire activa 0
- indica adresa valida pentru improspatarea memoriilor dinamice

HALT - oprire CPU
- iesire activa pe 0 logic
- CPU intra dupa executia unei instructiuni HALT in starea HALT semnalizata prin activarea iesirii 18 si asteapta o intrerupere, executind in acest timp instructiuni NOP

WAIT - asteapta
- intrare, activa pe 0
- indica microprocesorului ca memoria sau echipamentul I/O nu sint gata pentru transferul datelor
- atit timp cit WAIT este activ CPU introduce stari de asteptare

INT - intrerupere
- intrare, activa pe 0
- cererea de intrerupere generata de la un dispozitiv I/O este recunoscuta la sfirsitul instructiunii curente daca intreruperile au fost activate prin program

NMI - intrerupere nemascabila
- intrare activa 0 logic
- intreruperea nemascabila are prioritate superioara lui INT si este totdeauna recunoscuta la sfirsitul instructiunii curente.
- NMI forteaza automat CPU sa porneasca de la locatia 0066(H)

RESET - intrare, activa pe 0
- initializeaza CPU
- in timpul RESET-ului magistralele de adrese si date trec in starea de mare impedanta, iar semnalele de control sint inactive

BUSRQ - cerere de magistrala
- intrare activa pe 0 logic
- cererea de magistrala are prioritate mai mare decit NMI si este recunoscuta la terminarea ciclului masina curent

- semnalul indica o cerere de magistrala si ca urmare toate magistralele CPU trec in stare de mare impedanta astfel incit sa poata fi utilizate de un alt dispozitiv

BUSAK - recunoastere cerere de magistrala
- iesire, activa 0 logic
- este utilizata pentru a indica dispozitivului care cere magistrala ca CPU a pus magistrala de date, adrese, comenzi in stare de mare impedanta si dispozitivul extern le poate utiliza

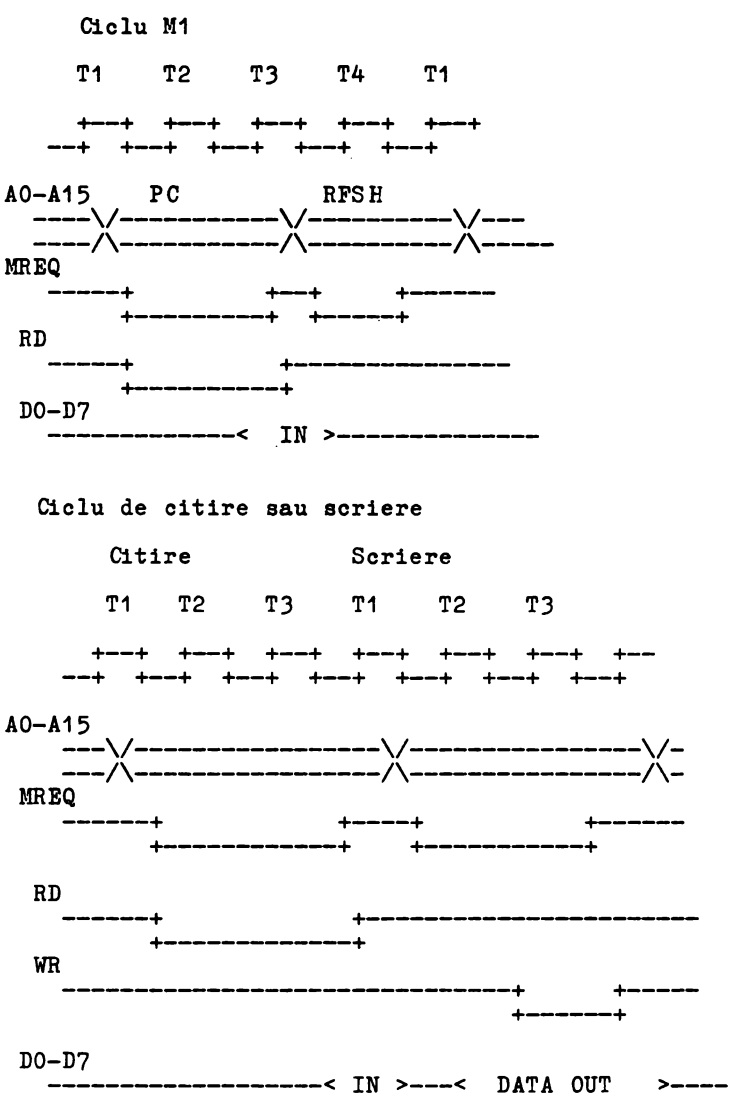


Fig. 4.2 Forme de unda pentru cicluri CPU

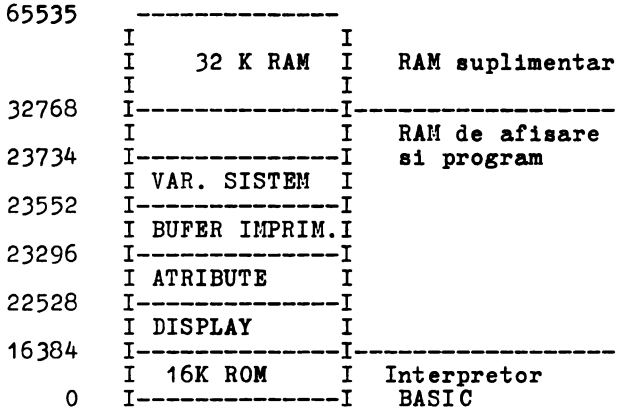
Harta memoriei

Spatiul de memorie adresabil de microprocesor este divizat in doua parti:

- memoria ROM (read only memory) cu o capacitate de 16K asezata intre adresele 0 si 16384 (3FFF hexa)
- memoria RAM (random access memory) ocupa restul de 48K din spatiul memoriei

Memoria RAM este divizata la rindul ei in doua blocuri distincte:

- memoria video si de program cu o capacitate de 16K asezata intre adresele 16384 si 32768
- memoria suplimentara cu o capacitate de 32K asezata intre adresele 32768 si 65535.



Memoria RAM de afisare si program contine o zona fixa la inceputul ei, formata din zona informatiei de afisare numita DISPLAY si zona atributelor video. Aceasta zona de memorie este accesata de sincrogenerator pentru afisarea imaginii TV, ea continind informatia corespunzatoare fiecarui punct de pe ecran. Urmeaza apoi zona BUFER IMPRIMANTA, zona care memoreaza o linie de 32 caractere pentru imprimanta.

Urmatoarele zone de memorie cuprind programe BASIC sau cod masina, variabile, stiva. Aceste zone au lungimi variabile in functie de programele si configuratia hard a sistemului, limitele lor curente aflindu-se la adresele specificate de variabilele de sistem.

4.3 MEMORIA ROM

Memoria ROM are o capacitate de 16K octeti. Ea este formata din 8 circuite EPROM 2716 (2Kx8). ROM-ul contine toata informatia in cod masina necesara pentru a implementa instructiuni, comenzi BASIC si subrutinele care manevreaza toate resursele hard ale calculatorului. De exemplu, exista o subrutina care scaneaza tastatura, o subrutina care programeaza bistabilul de "bell" facind sa sune difuzorul, o subrutina pentru citirea benzii magnetice, etc.

Semnalul ROMCS (vezi fila 2/10) merge la conectorul de extensie. Daca de pe extensie se forteaza acest semnal la 1 logic, ROM-ul existent pe placa este dezactivat.

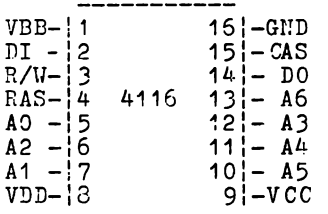
Cipurile EPROM 2716 dupa ce au fost programate pe arzator nu pierd informatia chiar daca se opreste tensiunea. Ele pot fi sterse cu raze ultraviolete si apoi reprogramate.

Memoria ROM este conectata direct pe magistralele CPU si lucreaza independent de restul memoriei.

4.4 MEMORIA VIDEO SI DE PROGRAM

Aceasta memorie (vezi fila 4/10) cu o capacitate de 16 Kocteti contine informatia necesara pentru a genera imaginea TV, variabilele de sistem cerute de BASIC, grafice definite de utilizator si programele BASIC ale utilizatorului.

Fiecare circuit de memorie 4116 poate contine 16K biti de informatie. Pentru adresare circuitul are 7 pini, deci adresa trebuie multiplexata pentru a adresa toate locatiile de memorie. La intrarile de adresa se prezinta 7 biti de adresa reprezentind adresa de rind, inscrisa in cip de semnalul RAS, apoi la intrarile de adresa se prezinta adresa de coloana formata din alti 7 biti si inscrisa de semnalul CAS.



Dupa ce a fost selectata in functie de felul in care este activat pinul 3 al cipului de memorie, dupa timpul de acces, memoria va scoate la iesire date valide (in cazul unui ciclu de citire) sau va inscrie date prezente pe magistrala (in cazul unui ciclu de scriere).

Memoria video suporta o dubla accesare:

- sincrogeneratorul o acceseaza la perioade fixe de timp pentru a citi informatia video si atributele de culoare
- CPU acceseaza memoria video pentru a schimba imaginea, atributele de culoare, variabilele de sistem sau pentru a stoca programe BASIC sau date

Dubla accesare conduce la o dubla multiplexare:

- multiplexarea adreselor de la microprocesor - se realizeaza prin circuitele D11 și E10 (tip 74LS157). Iesirile lor ajung pe intrarile memoriilor printr-o bariera rezistiva formata din rezistentele R79-R85.
- multiplexarea adreselor sincrogeneratorului - se realizeaza prin intermediul multiplexoarelor E5 si F5 (tip 74LS257). Iesirile lor sint direct conectate pe intrarile de adresa ale memoriei video, avind prioritate la accesul memoriei. Din aceasta cauza cind microprocesorul acceseaza memoria aceste multiplexoare sint deselectate cu ajutorul semnalului SEL.

Sincrogeneratorul accesind memoria video in mod pagina in afara de multiplexarea simpla linie/coloana, adresa de coloana este multiplexata pentru citirea octetului de informatie respectiv octetului de atribute video. Aceasta multiplexare se realizeaza de catre circuitul E4 (tip 74LS157).

Dubla accesare a memoriei video duce la conflict intre microprocesor si sincrogenerator. Daca conflictul s-ar rezolva prin oprirea afisarii pe durata accesului microprocesorului, calitatea imaginii ar avea de suferit. Se prefera oprirea ceasului de procesor ori de cite ori aceasta acceseaza memoria video pina la terminarea ciclului de afisare curent. Oprirea ceasului provoaca greutati pentru programele care ruleaza in memoria video si necesita bucle de temporizare de durata precisa. De aceea in aceste cazuri se recomanda folosirea memoriei suplimentare.

Functionarea selectiei memoriei 16K RAM (vezi pagina 6/10) se poate urmari pe formele de unda.

Ciclul de memorie RAM este impartit in doua:

- ciclu CPU
- ciclu afisare

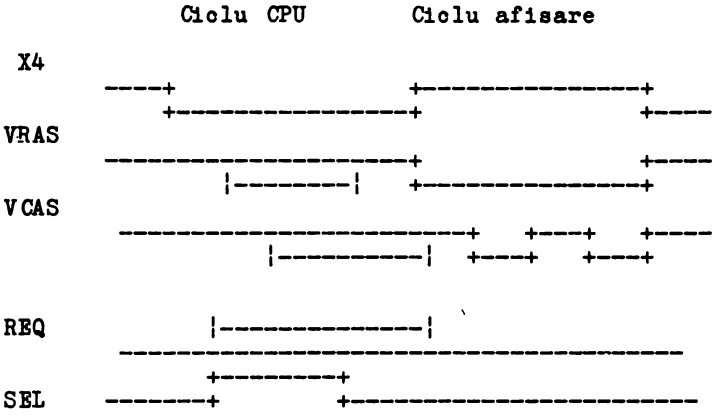
Ciclul CPU

Ori de cite ori CPU acceseaza memoria RAM de afisre si program (A15=0, A14=1) se decodifica semnalul ACC, care provoaca oprirea ceasului de procesor si repornirea lui pe perioada semnalului REQ, cind procesorului i se permite accesul la memorie.

Se activeaza semnalele VCAS si VRAS iar semnalul SEL dezactiveaza multiplexoarele de adresa de la sincrogenerator, CPU putind trimite sau primi un octet pe magistrala de date prin bufferul format de circuitele D8 si D9.

Ciclu afisare

Ori de cite ori X4 este pe unu, sincrogeneratorul acceseaza memoria RAM de afisare si program pentru a citi informatia din memorie corespunzatoare unui grup de 8 puncte de pe ecran. Pentru a specifica aceste puncte sint necesari doi octeti, un octet din zona DISPLAY si octetul corespunzator din zona atributelor video. Pentru aceasta se utilizeaza un ciclu de citire din memorie in mod pagina. Se activeaza semnalul VRAS iar apoi VCAS pulseaza de doua ori corespunzator citirii octetului de informatie si atribute.



Formele de unda pentru RAM video

4.5 MEMORIA SUPLIMENTARA

Memoria suplimentara are o capacitate de 32 Kocteti fiind formata din doua rinduri de cite 8 memorii 4116, E11 - E18 si F11 - F18.

Memoria suplimentara este direct legata pe magistrala de date CPU si prin intermediul multiplexoarelor D11 si E10 pe magistrala de adrese CPU.

Memoria suplimentara poate fi accesata de catre CPU chiar daca sincrogeneratorul lucreaza cu memoria video, magistralele lor fiind separate.

Semnalele de selectie SRAS, CAS1, CAS2, SWR, pentru memoria suplimentara deriva direct din semnalele de control ale procesorului (vezi fila 1) semnalul SWR este chiar WR trecut printr-o poarta repetoare C10.

La fel se obtine semnalul SRAS din MRQ. Se asigura astfel reimprospatarea automata a memoriei suplimentare pe durata fiecarui ciclu FETCH de catre CPU.

Pentru selectia CAS, semnalul RAS este intirziat si aplicat pe rindul de memorie corespunzator atunci cind se decodifica ciclul de citire sau scriere in memoria suplimentara.

Semnalul MUX care schimba adresa la multiplexoarele D11, E10 are ca sursa semnalul MREQ in cazul unui ciclu de memorie suplimentara si RASRQ in cazul in care microprocesorul adreseaza memoria video.

4.6 SINCROGENERATORUL

Sincrogeneratorul este alcatuit din numaratoare sincrone tip 74LS161, cu incarcare paralela si logica aferenta.

Pornind de la un oscilator cu cuart (vezi fila 6/10) de 14 MHz prin divizari inlantuite se obtin fazele de tact pentru microprocesor pentru adresarea memoriei video si semnalele de sincronizare linie si cadre.

Adresele pentru explorarea memoriei video au fost notate de la x1 la x8 pentru cele 256 de puncte pe linie si de la y0 la y7 pentru a adresa 192 linii utile pe cadru.

Punctele ecranului sint grupate in patratele 8x8, corespunzind unui caracter. Se pot afisa in acest mod 24 de rinduri/32 caractere. Din motive de economie de memorie atibutele de culoare sint definite la nivel de matrice 8x8 (caracter).

In interiorul fiecarei locatii de caracter un punct nu poate avea decit doua culori, numite "ink (cerneala)" si "paper (hirtie)".

Cele doua culori sint specificate cu un sigur octet, octetul de atribute.

Formatul octetului de atibute:

D7	D6	D5	D4	D3	D2	D1	D0
F	*	PAPER				INK	

F - atributul de flash (caracter clipitor) activ pe 1
PAPER - specifica una din 8 culori pentru fond prin bitii D3-D5
INK - specifica una din 8 culori pentru scris prin bitii D0-D2

Informatia asupra imaginii TV se afla stocata la inceputul memoriei RAM intr-o zona fixa impartita in doua :

- zona de informatie video, cuprinsa intre adresele 16384-22527
- zona de atribute video, cuprinsa intre adresele 23528-23295

Zona de informatie specifica pentru fiecare punct de ecran tipul sau. Daca bitul corespunzator este zero, punctul este de tip paper; daca bitul este 1, punctul va fi ink.

Zona de atribute specifica pentru fiecare zona de 8x8 puncte cele doua culori (ink/paper) si daca zona clipeste sau nu.

Interfata video (vezi pagina 7) produce semnale RGB pentru monitor color, folosind informatia din memoria video. La fiecare grup de 8 octeti de informatie video corespunde un octet de atribute care reprezinta :

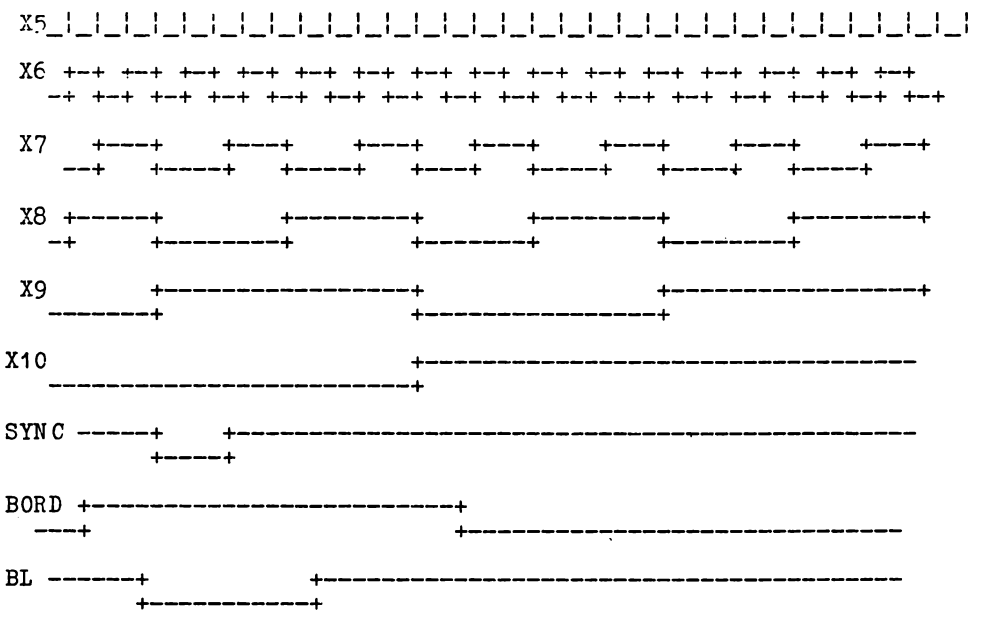
- 3 biti informatie de culoare pentru hirtie (PAPER)
- 3 biti informatie de culoare pentru cerneala (INK)
- un bit informatie pentru atributul de clipire (FLASH)

Informatia video este inscrisa in registrul de serializare pe 8 biti (tip 74LS165) iar cea de culoare in registrul de 8 biti D7 (tip 74LS374).

Multiplexorul C7 realizeaza multiplexarea culorii de "paper" si "ink" iar C8 multiplexarea culorii de margine si centru ecran.

Informatia de culoare margine ecran este inscrisa in registrul C9 (tip 74LS174) pe biti de date D0, D1, D2, prin intermediul portului de iesire 254 (FE).

Semnalele de sincronizare linii, cadru se obtin prin decodificarea fazelor numaratoarelor din sincrogenerator (vezi pag. 6). Formele de unda pentru semnalele de sincronizare precum si pentru semnalul de margine ecran BORD si pentru semnalul de stingere BL sint prezentate mai jos.



4.7 TASTATURA

Tastatura este formata dintr-o matrice de fire 8x5 (vezi pagina 8/10).

Fiecare tasta este legata la intersectia unei linii cu o coloana, astfel incit la apasarea unei taste un fir orizontal se conecteaza cu unul vertical. Firele orizontale (rinduri) sint conectate la partea superioara a magistralei de adresa a8-a15 prin intermediul circuitului separator A9. Cele cinci fire verticale (coloane) sint conectate la magistrala de date prin intermediul circuitului A8.

La fiecare intrerupere primita, CPU scaneaza tastatura, executind instructiuni IN (input) de la portul FE (hexa) si pe adresa superioara o singura linie de adresa este la zero. Daca o tasta a fost apasata, octetul venit de la portul 254 (FE) contine un zero pe bitul corespunzator tastei apasate.

Linie de adresa pusa la zero	Adresa I/O	Taste selectate
A15	32766	SPACE, CS, M, N, B
A14	49150	CR, L, K, J, H
A13	57324	P, O, I, U, Y
A12	61438	0, 9, 8, 7, 6
A11	63486	1, 2, 3, 4, 5
A10	64510	Q, W, E, R, T
A9	65022	A, S, D, F, G
A8	65278	CS, Z, X, C, V

4.8 INTERFATA AUDIO

Permite conectarea la o sursa de inregistrare/redare audio (casetofon, magnetofon).

Conectorul audio are urmatoarea asignare:

1,4 - Iesire - nivel 500 mV
- Impedanta de iesire 500 ohmi

3,5 - Intrare 1-4 V
- Impedanta sursa semnal max. 10 Kohmi

Linia de iesire audio (vezi pagina 7) este controlata prin intermediul bistabilului corespunzator din registrul C9.

Starea acestui bistabil este modificata prin scriere la portul 254 (FE), prin intermediul bitului de date D3.

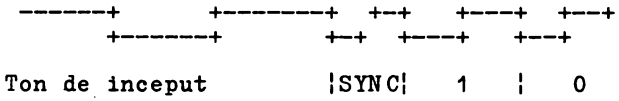
Scriind alternativ 0 sau 1 se poate transmite o forma de unda rectangulara pe iesirea de linie audio, aceasta putind fi inregistrata pe banda audio. Volumul si frecventa tonului obtinut depind de durata pe care starea bistabilului din registrul C9 a ramas neschimbata.

La citire semnalul de pe banda magnetica este filtrat si amplificat (vezi pagina 8), semnalul obtinut controlind starea bitului 6 de pe magistrala de date in timpul operatiilor de citire de la portul 254.

Toate fisierele pe banda magnetica sint inregistrate ca doua blocuri de informatie : header si bloc de date.

Header-ul contine informatie despre datele care urmează, anume: numele fisierului si numarul de octeti ai blocului.

Fiecare tip de bloc incepe cu un ton de aproximativ 5 s pentru header si 2 s pentru blocul de date. Tonul de inceput este o unda rectangulara de 619,4 microsecunde intre fiecare schimbare de stare corespunzind unei frecvente de 807 Hz. Sfirsitul tonului de inceput este semnalat printr-un impuls de durata scurta SYNC, care sta pe zero 190,6 microsecunde si 210 pe unu. Urmeaza apoi fara nici o pauza pulsuri de date. Daca este un zero pulsul va avea o durata de 244,3 microsecunde pe zero si tot atit pe unu. Daca este un unu, pulsul dureaza dublu.

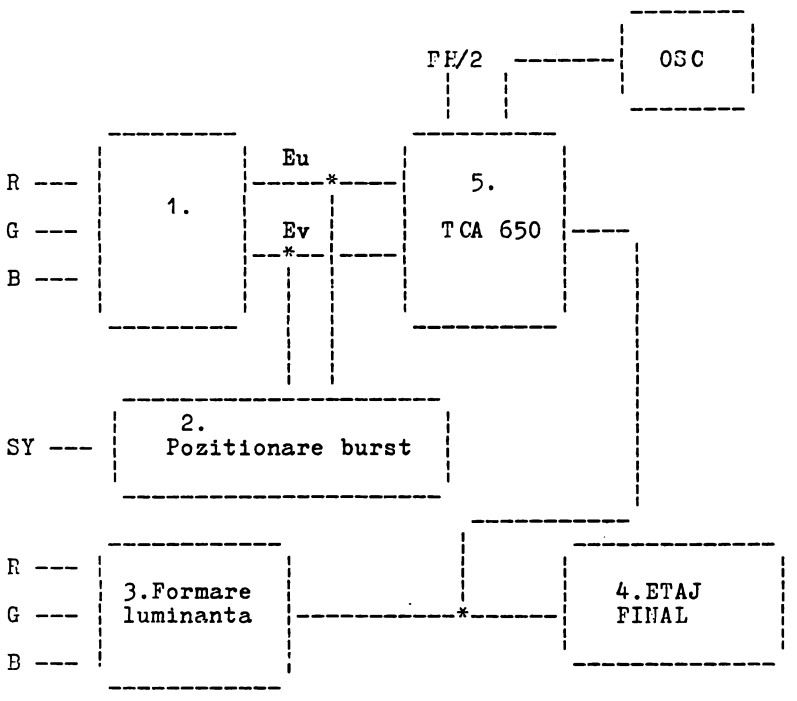


SUNET

Sunetul este generat prin intermediul starii bistabilului din registrul C9 (vezi pagina 7), care este controlata de bitul D4 de pe magistrala de date in timpul instructiunilor de scriere la portul 254 (FE).

4.9 CODORUL PAL

Functinarea blocului de codificare a semnalelor R,G,B intr-un semnal PAL se poate urmari pe schema bloc.



Blocul 1 reprezinta o schema de matriciere a semnalelor R, G, B, astfel incit la iesire se obtin semnalele diferenta de culoare Eu si Ev (vezi pagina 9).

Matricierea se realizeaza prin intermediul decodificatorului A1 si o retea de diode si rezistente. Corespunzator celor trei intrari R, G, B (rosu, verde, albastru) se activeaza o anumita iesire a decodificatorului A1 si se obtin nivelele cerute de standard pentru Eu si Ev.

Potentiometrii R6 si R7 servesc la stabilirea corecta a componentei de curent continuu la intrarea in TCA 650.

Blocul 2 primeste la intrare semnalul de sincronizare SY si scoate la iesire doua impulsuri in opozitie de faza, situate in timp in locul unde se introduce semnalul de burst (sincronizare de culoare).

Blocul 3 primeste la intrare semnalele R, G, B si scoate la iesire semnalul de luminanta prin insumare si ponderare corespunzatoare.

Blocul 4 reprezinta etajul de iesire pentru semnalul video-compus codificat PAL. Acest semnal poate fi utilizat pentru vizualizare pe monitor color PAL sau, dupa modulare, pe TV color.

Blocul 5 primeste la intrare semnalele Eu si Ev si genereaza semnalul de crominanta. Blocul folseste circuitul TCA 650.

Frecventa de subpurtatoare PAL se obtine de la un oscilator cu cuart notat OSC, care furnizeaza o frecventa de 4,433618 MHz.

Circuitul TCA 650 realizeaza modulatia in cuadratura a celor doua semnale Eu si Ev. Semnalul de crominanta fiind diferit de la o linie la alta, circuitul TCA are nevoie si de o frecventa egala cu jumatate din frecventa liniilor notata FH/2.

linia de rang n: Uc = Ev*COS(OMEGAp*T)+Eu*SIN(OMEGAp*T)
linia de rang n+1: Uc= -Ev*COS(OMEGAp*T)+Eu*SIN(OMEGAp*T)

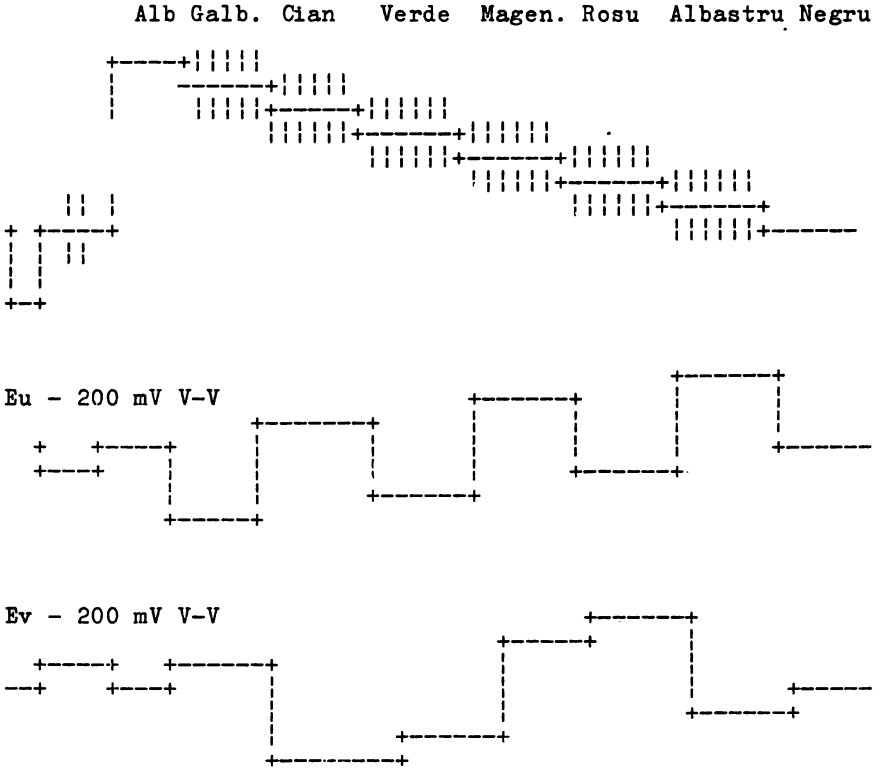
Semnalul de crominanta generat in blocul 4 compus cu cel de luminanta generat in blocul 3 este aplicat etajului final.

DECODIFICARE

Culoare	R	G	B	Rang A1	Pin
Alb	1	1	1	7	*
Galben	1	1	0	6	9
Cian	0	1	1	5	10
Verde	0	1	0	4	11
Magenta	1	0	1	3	12
Rosu	1	0	0	2	13
Albastru	0	0	1	1	14
Negru	0	0	0	0	*

Deoarece culorile negru si alb nu produc semnal de crominanta, pinii corespunzatori din decodificator sint nefolositi.

Formele de unda corespunzatoare barelor color sint



4.10 CONECTORII

Conectorul de extensie.

B		A	
A11 -	I 28	I	
A9 -	I 27	I	- A10
BUSACK -	I 26	I	- A8
ROMCS -	I 25	I	- RFSH
A4 -	I 24	I	- M1
A5 -	I 23	I	-
A6 -	I 22	I	- +12 V
A7 -	I 21	I	- WAIT
RESET -	I 20	I	- -5 V
BUSRQ -	I 19	I	- WR
	I 18	I	- RD
	I 17	I	- IORQ
	I 16	I	- MREQ
	I 15	I	- HALT
GND -	I 14	I	- NMI
IORGE -	I 13	I	- INT
A3 -	I 12	I	- D4
A2 -	I 11	I	- D3
A1 -	I 10	I	- D5
A0 -	I 9	I	- D6
CLK -	I 8	I	- D2
GND -	I 7	I	- D1
GND -	I 6	I	- D0
SLOT -	I 5	I	- SLOT
	I 4	I	-
+5V -	I 3	I	- D7
A12 -	I 2	I	- A13
A14 -	I 1	I	- A15

13A - INT - intreruperi

- conectat la linia de intreruperi Z80A
- conectat printr-o rezistenta de 680 ohmi la sursa de intreruperi de pe placa logica (bistabilul A3).
- poate fi utilizat de un dispozitiv extern pentru a genera intreruperi pentru Z80A sau, legat la +5V, dezactiveaza intreruperile de pe placa.

14A-NMI

- in mod normal la nivel logic 1
- poate fi utilizat de un dispozitiv extern pentru a forta un salt la adresa 102 (66 hexa) si sa execute cod masina de acolo.

17A - IORQ - cerere I/O

- indica ca partea mai putin semnificativa a magistralei de adresa contine o adresa valida a unui port I/O. Partea superioara are continutul acumulatorului daca instructiunea este utilizata in cod masina. Din BASIC se poate specifica o adresa pe 16 biti. Aceasta apare pe A0-A15 cind IORQ este activ.

21A - WAIT

- poate fi utilizat de dispozitive externe mai lente pentru a se sincroniza cu CPU. Nu trebuie activata linia WAIT mai mult de 1 ms pentru a nu strica reimprospatarea memoriei suplimentare.

8B - CLOCK - 3,5 MHZ

- poate fi utilizat de dispozitive externe pentru a se sincroniza cu CPU.
- poate fi oprit cind procesorul acceseaza memoria video.

13B - IORGE

- pus la +5V, logica de pe placa nu mai primeste IORQ.

19B - BUSRQ - conectat la Z80A

- poate fi utilizat pe un dispozitiv extern pentru a prelua magistralele CPU. Controlul este predat dupa incheierea ciclului masina curent.

25B - ROMCS

- poate fi conectat la +5V pentru a dezactiva memoria ROM de pe placa.

CONECTOR VIDEO

```
  SYNC - I 8    I
         I   15 I - HSINC
     G - I 7    I
         I   14 I - VIDEO PAL
     R - I 6    I
         I   13 I - GND
     B - I 5    I
         I   12 I - GND
 VSYNC - I 4    I
         I   11 I - GND
  FH/2 - I 3    I
         I   10 I -
       - I 2    I
         I    9 I - +5V
   +5V - I 1    I
```

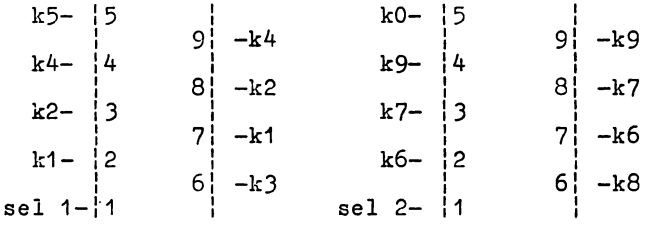
Semnalele SYNC, R, G, B permit conectarea unui monitor color cu intrare RGB (de exemplu monitor color Electronica 001), sau televizor cu priza PERITEL.

Semnalul VIDEO PAL permite conectarea unui monitor color PAL (de exemplu monitor color Electronica 002), sau monitor alb negru cu intrare video compus.

Semnalul FH/2 are o frecventa egala cu jumatate din frecventa de linie.

El poate fi folosit impreuna cu semnalele SYNC, R, G, B pentru a obtine semnal video codificat pentru alt sistem TV folosind o interfata corespunzatoare.

Conectorii joy-stick dubleaza practic functiile primului rind de taste.



JOYSTICK 1 JOYSTICK 2

Cele doua manete tip joystick sint de fapt niste simple contacte normal deschise care se inchid atunci cind sint manevrate intr-o directie (sus, jos, stinga, dreapta).

La fiecare pin de conector s-a notat tasta care este dublata in momentul in care se inchide contactul dintre pinul respectiv si pinul de selectie notat SEL 1, respectiv SEL 2.

FUNCTII	JOYSTICK 1	JOYSTICK 2
STINGA	Tasta 1	Tasta 6
DREAPTA	Tasta 2	Tasta 7
JOS	Tasta 3	Tasta 8
SUS	Tasta 4	Tasta 9
FOC	Tasta 5	Tasta 0

CONECTOR AUDIO

1,4 - Iesire 500 mV
3,5 - Intrare 1-5 V
2 - Masa

CONECTOR TV

Serveste la conectarea televizorului prin intermediul mufei de antena.

4.11 SURSA DE ALIMENTARE

Alimentatorul extern furnizeaza o tensiune redresata nestabilizata de minim 9V in sarcina.

Tensiunea de +5V se obtine cu ajutorul unui stabilizator integrat tip 7805.

Pentru tensiunile de +12V si -5V se foloseste un convertor format din tranzistorii T12, T13 (vezi pagina 10/10) si componentele pasive aferente.

T12 formeaza o reactie de curent pentru oscilatorul format din T13, L4, C82, R104.

Functionarea circuitului se bazeaza pe tensiunea inversa care apare prin L4 la fiecare ciclu de oscilatie. Tensiunea inversa ridica colectorul lui T13 la 13V, D19 conduce si incarca condensatorul C87. Cind D17 nu conduce, C87 se descarca asigurind 12V pentru memorie. Daca tensiunea de 12V scade atunci T13 conduce mai mult, creste frecventa de oscilatie si tensiunea se ridica la valoarea ei nominala.

Sursa de -5V consta din componentele R100, D16, D17, R105, C88; cind tensiunea se ridica la 13V in colectorului T13, C83 se incarca prin D18 la aproximativ 12V. Cind colectorul T13 cade la 0V, polul negativ al condensatorului C83 devine -12V. C83 se descarca prin D16 si R100. Tensiunea de pe C88 este mentinuta constant la -5V de dioda Zener D17.

4.12 PROGRAME DE TEST

Programele de test pentru calculatorul HC-85 se afla intr-un PROM de 2 Kocteti care se insereaza pe extensie. Se pot executa urmatoarele teste hard:

1 - verifica procesorul Z80A;
2 - verifica memoria de ecran in alb-negru;
3 - verifica memoria video intre adresele 5B00-8000(H).
4 - permite operatorului sa verifice fiecare tasta prin vizualizarea tastaturii pe ecranul TV, fiecare tasta apasata aparind colorata mai intens;
5 - verifica culorile prin afisarea mirei cu bare colorate;

Pentru verificarea memoriei suplimentare operatorul poate alege unul din urmatoarele teste:

1 - ADDRESS - descopera greseli in logica adresarii;
2 - BARBER - descopera circuite defecte;
3 - GALPAT - test de memorie (11 ore);
4 - PINGPONG - test de memorie (4 min.);
5 - SART - verifica saturarea amplificatoarelor de scriere/citire;
6 - WALC - verifica ciclii FETCH din aceasta memorie.

Toate aceste teste sint selectate prin apasarea tastei 1 (corespunzator rindului 1) sau 2 (corespunzator rindului 2 de memorie suplimentara) si prima litera din numele testului.

4.13 DEPANARE

Echipamente necesare pentru depanare:

- Osciloscop 10 MHz
- Alimentator
- TV color
- Monitor color PAL
- aparat de masura
- programe de test

Sinopticul dat mai jos pentru localizarea defectelor nu cuprinde decit defecte care apar mai des. Microcalculatorul fiind un dispozitiv complex, nu se pot lua in consideratie toate cazurile care pot sa apara. Pentru depanare este bine sa se foloseasca la nevoie o placa martor, pentru compararea formelor de unda vizualizate pe osciloscop.

DEFECT	ACTIUNI PENTRU REMEDIERE
Nu avem imagine TV	Se verifica sincrogeneratorul: -se verifica oscilatorul 14 MHz -se verifica lantul de divizare F3,D5,D4,D3,E3
Imagine stabila cu dungi verticale	Se verifica logica de formare a semnalelor ACC, REQ, introducind pe extensie un PROM de test
Imagine neagra sau cu patratele colorate aleator	Se verifica tensiunile de alimentare la memoria RAM Se verifica RAM de afisare si program
Nu se initializeaza	Se verifica functionarea intreruperilor Se verifica memoria PROM
Nu ia anumite taste	Dupa verificarea separata a tastaturii si a cablului de tastatura se verifica circuitul A8
Apar dungi colorate in interiorul caracterului	Se verifica functionarea circuitului D7
Memorie suplimentara	Se pune in evidenta defectul tastind: PRINT PEEK 23732 + PEEK 23733 * 256 Daca rezultatul nu este 65535,se introduc pe extensie testele de memorie si se depisteaza memoria defecta

ANEXA A

Lista componente HC 85

NR. CRT.	DENUMIRE / CARACTERISTICI (COD VEST - COD EST)			POZITIE	CANT. (BUC)
1.	Z80A CPU	-		B11	1
2.	SN74LS00N	-	K555LA3	B6,B8,B10,D10,E2,F4	6
3.	SN74LS04N	-	K555LN1	A2,B7,E1,F2	4
4.	SN74LS08N	-	K555LI1	B3,B9	2
5.	SN74LS32N	-	K555LL1	C3,C10,D2	3
6.	SN74LS74N	-	K555TM2	A3,B4,C4,C6,D1	5
7.	SN74LS138N	-	74LS138PC	A1,A16	2
8.	SN74LS86N	-	K555LP5	B5	1
9.	SN74LS157N	-	K555KP16	C7,C8,D11,E4,E10	5
10.	SN74LS161N	-	K555IE10	C5,D3,D4,D5,E3,F3	6
11.	SN74165N	-	74165PC	D6	1
12.	SN74LS174N	-	K555TM9	C9	1
13.	SN74LS257N	-	K555KP11	E5,F5	2
14.	SN74LS373N	-	K555IR22	A8,A9	2
15.	SN74LS374N	-	K555IR23	D7	1
16.	I2716	-	K573RF5	B12-B15 D12-D15	8
17.	MK4116	-	K565RU3	E6-E9,E11-E18 F6-F9,F11-F18	24
18.	I8216	-	K589AP16	D8,D9	2
19.	UA7805	-	MA7805		1
20.	TCA 650			D0	1

TRANZISTOARE

1.	TRANZISTOR BC 107 B
2.-4.	TRANZISTOR BC177
5.	TRANZISTOR BC 172
6.	TRANZISTOR 2N2369
7.-10.	TRANZISTOR BC 107
11.	TRANZISTOR BD 135
12.	TRANZISTOR BC 177
13.	TRANZISTOR BD135

DIODE

1.-13.	DIODA 1N4148
14.	DIODA PL 10V
15.	
16.	DIODA 1N4148
17.	DIODA PL 5V6
18.	DIODA 1N4148
19.	DIODA BA 157
20.	DIODA GERMANIU

QUARTZ

1.	QUARTZ 14MHZ
2.	QUARTZ 4,4336 MHZ PAL

CONDENSATORI

1.-2.	CONDENSATOR 68 PF
3.	CONDENSATOR 12 PF
4.	CONDENSATOR 100 NF
5.	CONDENSATOR 100 PF
6.-7.	CONDENSATOR 100 NF
8.	CONDENSATOR 4, MF
9.	CONDENSATOR 27 PF
10.	CONDENSATOR 4.7 NF
11.-12.	CONDENSATOR 100 NF
13.	CONDENSATOR 47 PF
14.	CONDENSATOR 27 PF
15.	CONDENSATOR 10 MF
16.	CONDENSATOR 100 NF
17.	CONDENSATOR 12 PF
18.	CONDENSATOR 47 PF
19.	CONDENSATOR 100 NF
20.	CONDENSATOR 470 PF
21.	CONDENSTOR 33 MF
22.	CONDENSATOR 1 NF
23.	CONDENSATOR 10 MF
24.	CONDENSATOR 100 NF
25.	CONDENSATOR 10 MF
26.	CONDENSATOR 2.6 NF
27.	CONDENSATOR 470 PF
28.-29.	CONDENSATOR 270 PF
30.	CONDENSATOR 100NF
31.	CONDENSATOR 10 MF
32.	CONDENSATOR 100 NF
33.	CONDENSATOR 10 MF
34.-70.	CONDENSATOR 100 NF
71.	CONDENSATOR 10 MF
72.	CONDENSATOR 100 NF
73.	CONDENSATOR 10 MF
74.	CONDENSATOR 100 NF
75.	CONDENSATOR 150 MF
76.	CONDENSATOR 10 MF
77.	Neutilizat
78.	CONDENSATOR 10 MF
79.-82.	CONDENSATOR 100 NF
83.	CONDENSATOR 1.5 MF
84.	CONDENSATOR 100 NF
85.	CONDENSATOR 10 MF
86.	CONDENSATOR 100 NF
87.-88.	CONDENSATOR 150 MF
89.-90.	CONDENSATOR VARIABIL 3-10 PF
91.	CONDENSATOR 10 MF
92.	CONDENSATOR 47 PF
93.	CONDENSATOR 100 PF
94.	CONDENSATOR 100 NF
95.-96.	CONDENSATOR 270 PF

REZISTENTE

1.	5.1 KOHMI
2.	100 KOHMI
3.	5.1 KOHMI
4.	100 KOHMI
5.	10 KOHMI
6.-7.	POTENTIOMETRU 5 KOHMI
8.	10 KOHMI
9.	820 OHMI
10.	4.7 KOHMI
11.	39 KOHMI
12.	5.1 KOHMI
13.	10 KOHMI
14.	180 KOHMI
15.	1.8 KOHMI
16.	2.7 KOHMI
17.	100 OHMI
18.	1.8 KOHMI
19.	15 KOHMI
20.	1.8 KOHMI
21.	1 KOHMI
22.	75 OHMI
23.	150 OHMI
24.	9.1 KOHMI
25.	5.1 KOHMI
26.	3.9 KOHMI
27.	15 KOHMI
28.	1.8 KOHMI
29.	820 OHMI
30.	1.8 KOHMI
31.	1 KOHMI
32.	2 KOHMI
33.	3.5 KOHMI
34.	1.5 KOHMI
35.	1.5 KOHMI
36.	3.5 KOHMI
37.	4.7 KOHMI
38.	620 OHMI
39.-40.	390 OHMI
41.	620 OHMI
42.	4.7 KOHMI
43.-44.	1 KOHMI
45.	330 OHMI
46.-48.	1 KOHMI
49.	50 OHMI
50.-53.	1 KOHMI
54.	330 OHMI
55.	75 OHMI
56.	1 KOHMI
57.	75 OHMI
58.	330 OHMI
59.-60.	1 KOHMI
61.	330 OHMI
62.	75 OHMI
63.	1 KOHMI
64.	75 OHMI
65.	330 OHMI
66.	10 KOMI
67.	2 KOHMI
68.	1 KOHMI

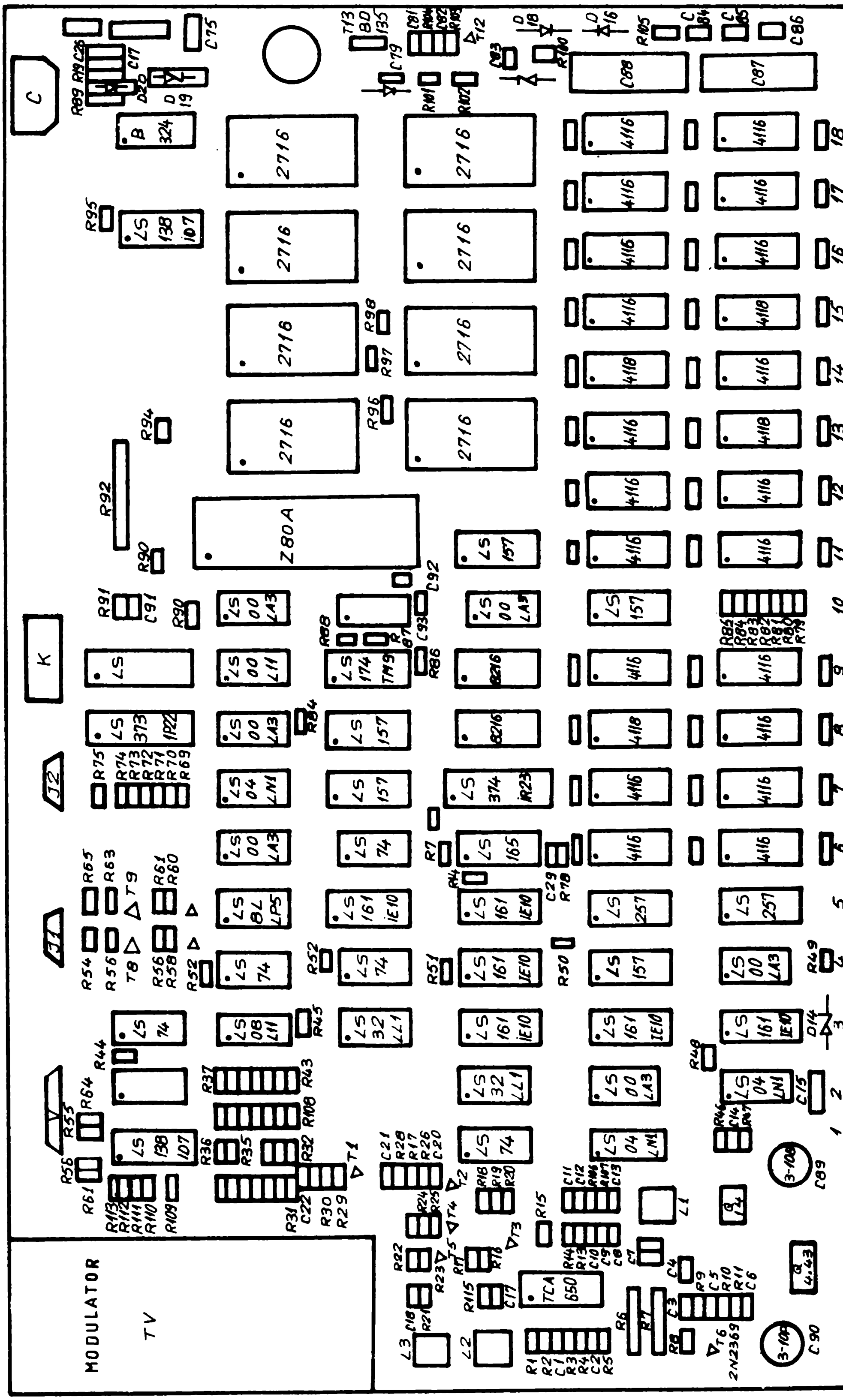
69.-73.	10 KOHMI
74.	1 KOHMI
75.	10 KOHMI
76.-77.	1 KOHMI
78.	680 OHMI
79.-87.	330 OHMI
88.	680 OHMI
89.	100 KOHMI
90.	1 KOHMI
91.	22 KOHMI
92.	RETEA 10 KOHMI
93.	10 KOHMI
94.-95.	680 OHMI
96.	10 KOHMI
97.	1 KOHMI
98.	1.5 KOHMI
99.	27 KOHMI
100.	56 OHMI
101.	1.8 KOHMI
102.	1 KOHMI
103.	100 OHMI
104.	15 OHMI
105.	100 KOHMI
106.-107.	2.7 KOHMI
108.	2 KOHMI
109.	1 KOHMI
110.	1.8 KOHMI
111.	1.2 KOHMI
112.	2 KOHMI
113.	5.6 KOHMI
114.	1 KOHMI
115.	33 KOHMI
116.	1 KOHMI

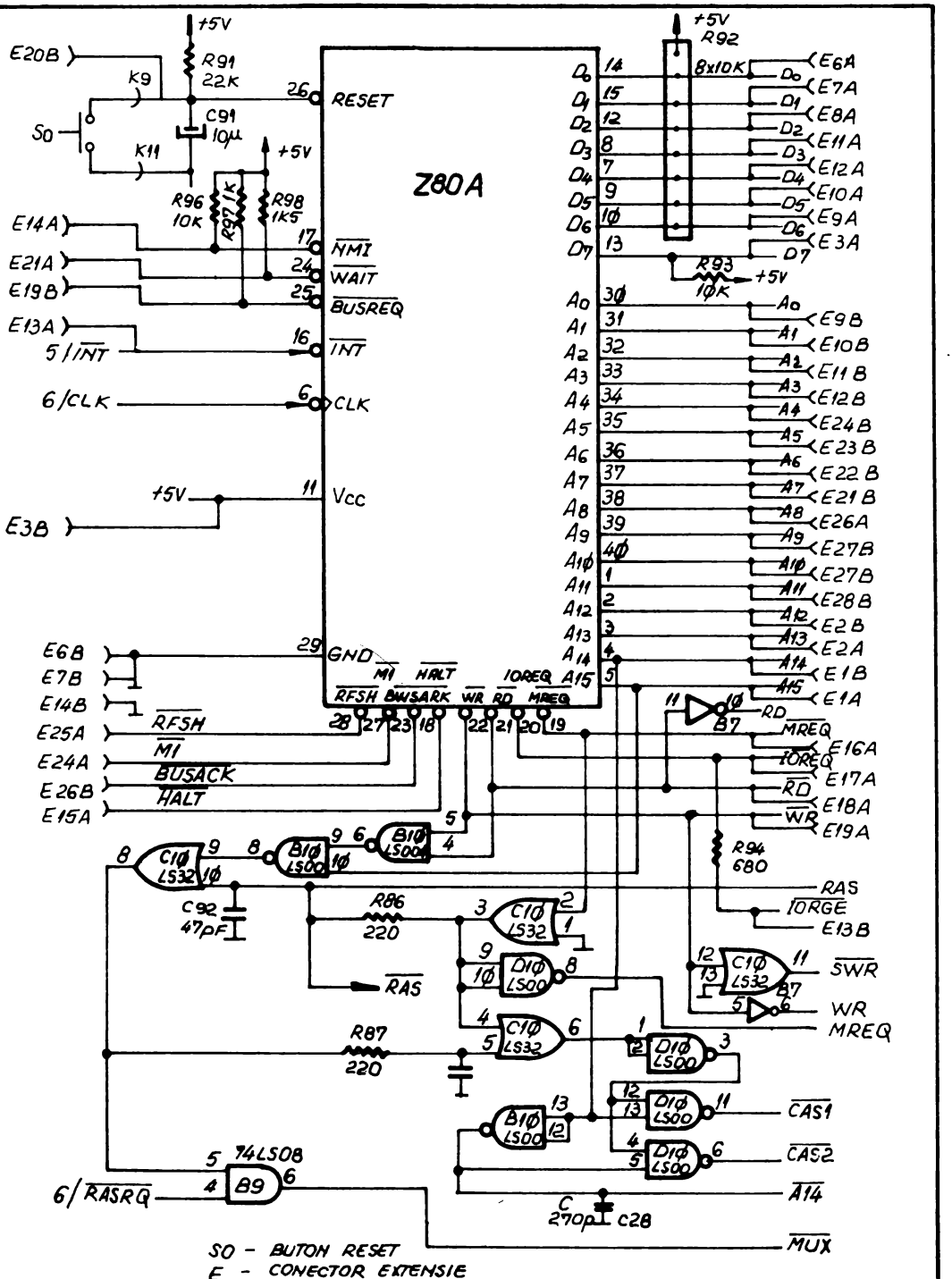
CONECTORI

1.	CONECTOR ALIMENTARE 303154		1
2.	" AUDIO 5 contacte 303608A		1
3.	" " mama 200860		1
4.	DIP HEADER CABLU PLAT 16 contacte		2
5.	" MOLEX 3 contacte mama		1
	TASTA 120023	670344007	40
	DIFUZOR		1
	BUTON RESET 220028		1
	MODULATOR TV		1
	Soclu lipit 16 pini		2
	Alimentator HC		1
	Cablu plat 16 fire		30 cm

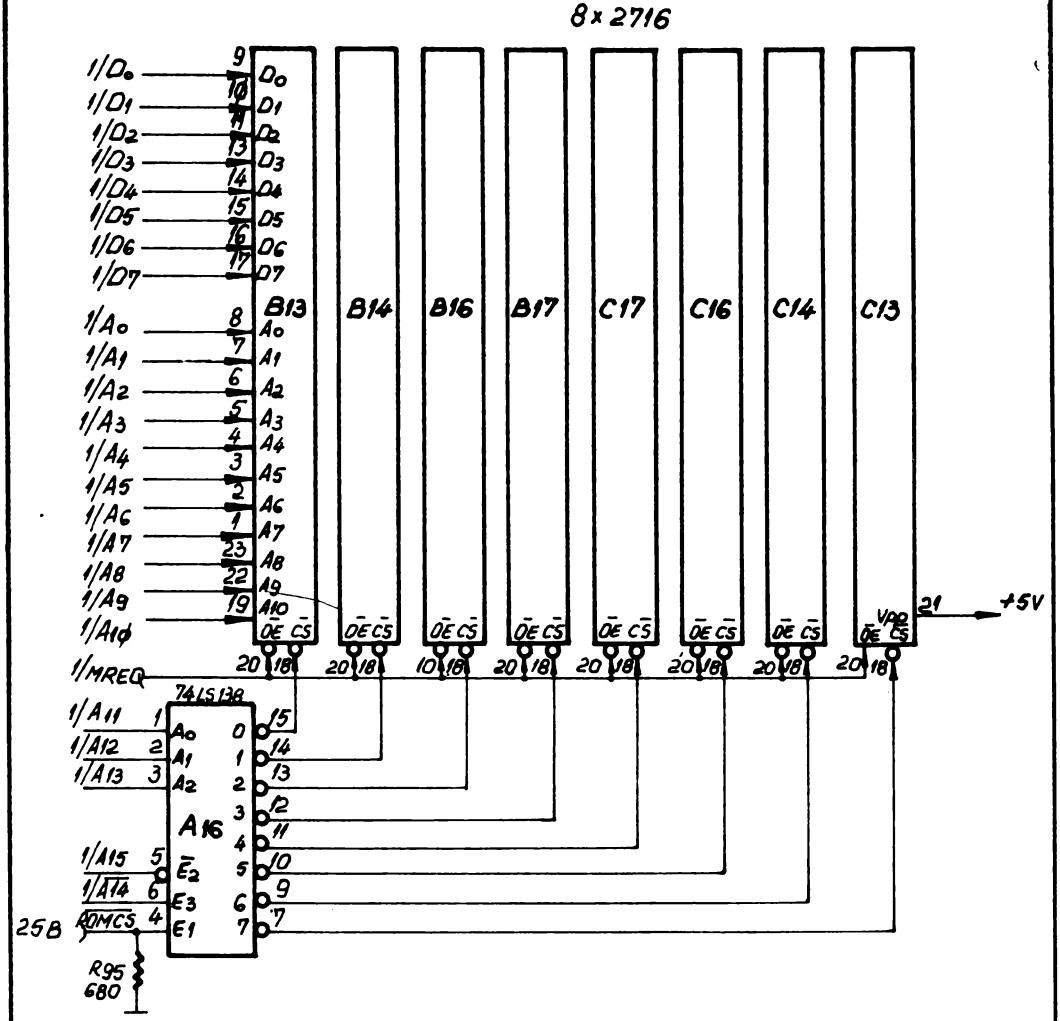
A N E X A B

S C H E M E L O G I C E

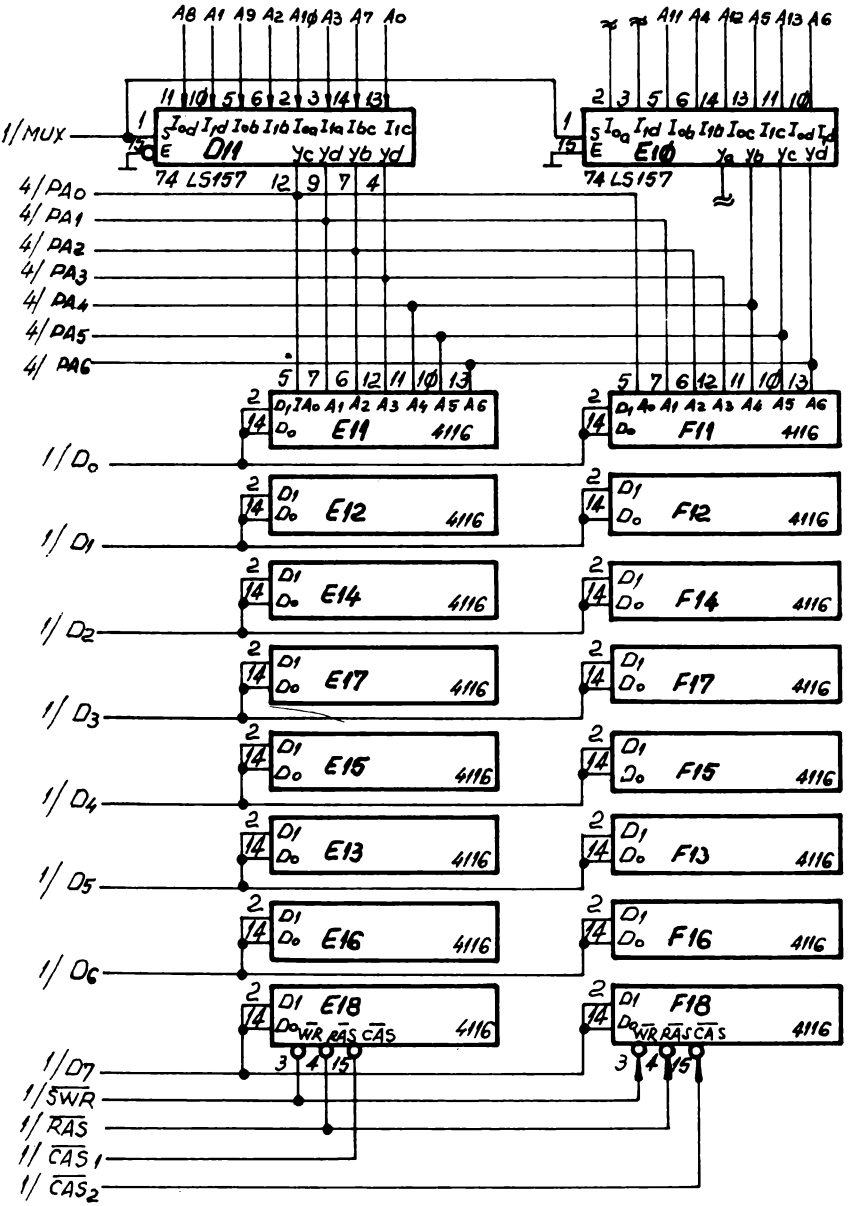

MODULATOR
TV
Z80A
2716
4116
4118
LS 138
LS 157
LS 00
LS 161
LS 74
LS 374
LS 373
LS 165
LS 257
LS 174
LS 32
LS 04
LS 08
8216
TCA 650
2N2369
J1
J2
K
C
B 324
L1
L2
L3
C87
C88
C89
C90



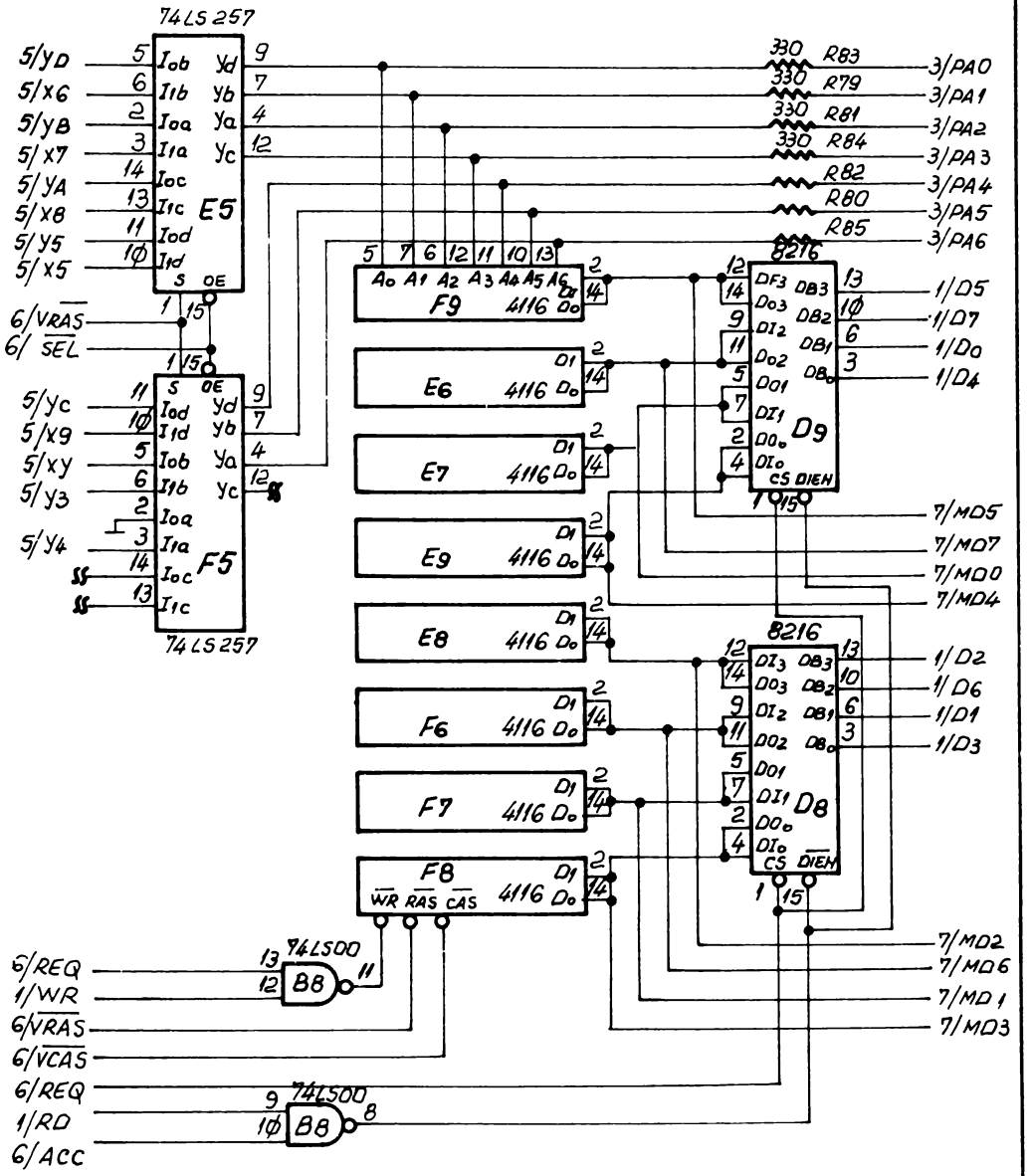
HC85 LOGIC DIAGRAMS			URC
ICE.	Rev.	869.400.001	
		CPU	Fila 1 din 10



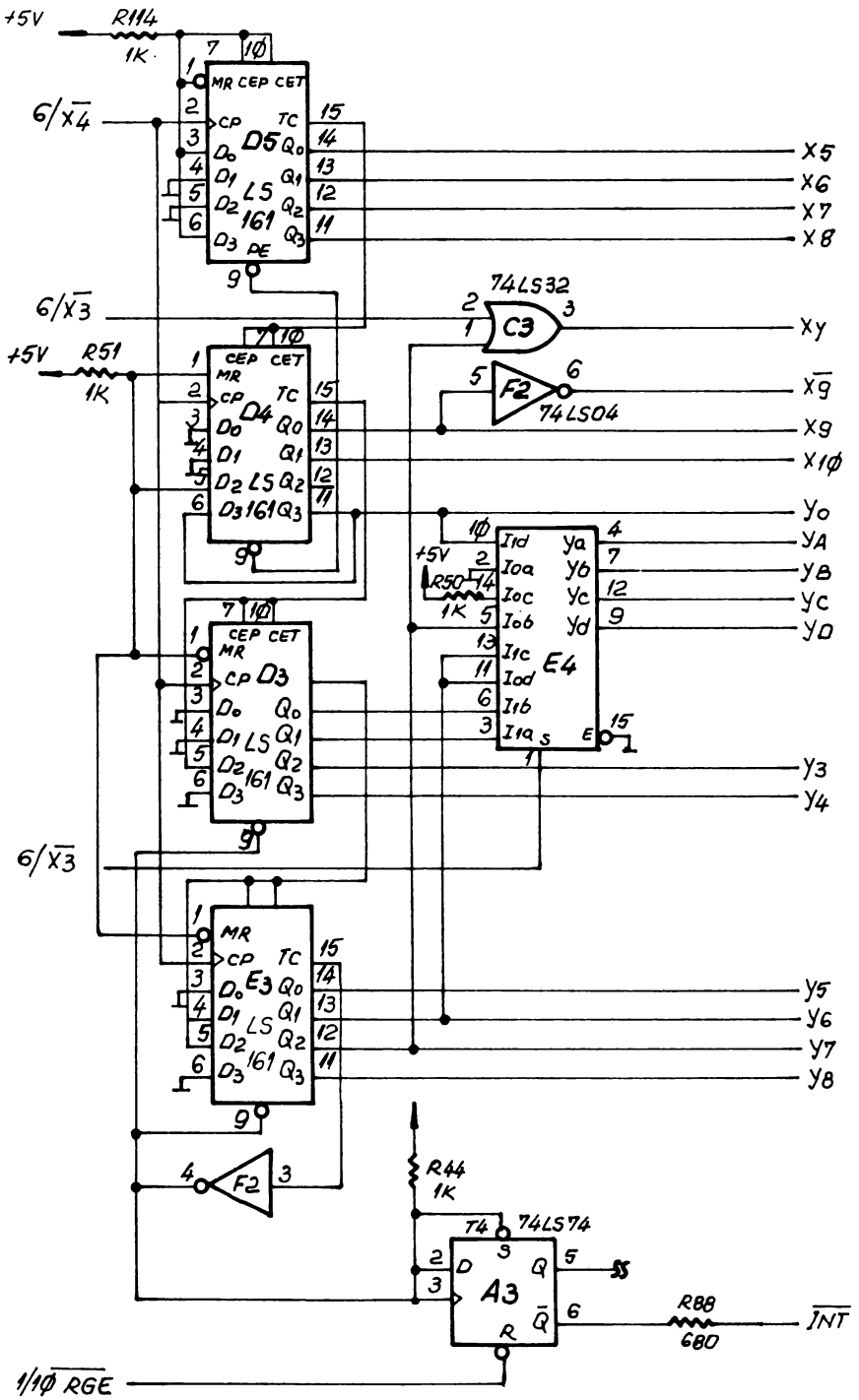
HC 85 LOGIC DIAGRAMS			URC
I.C.E.	Rev.	869. 400.001	
		ROM	Fila 2 din 10



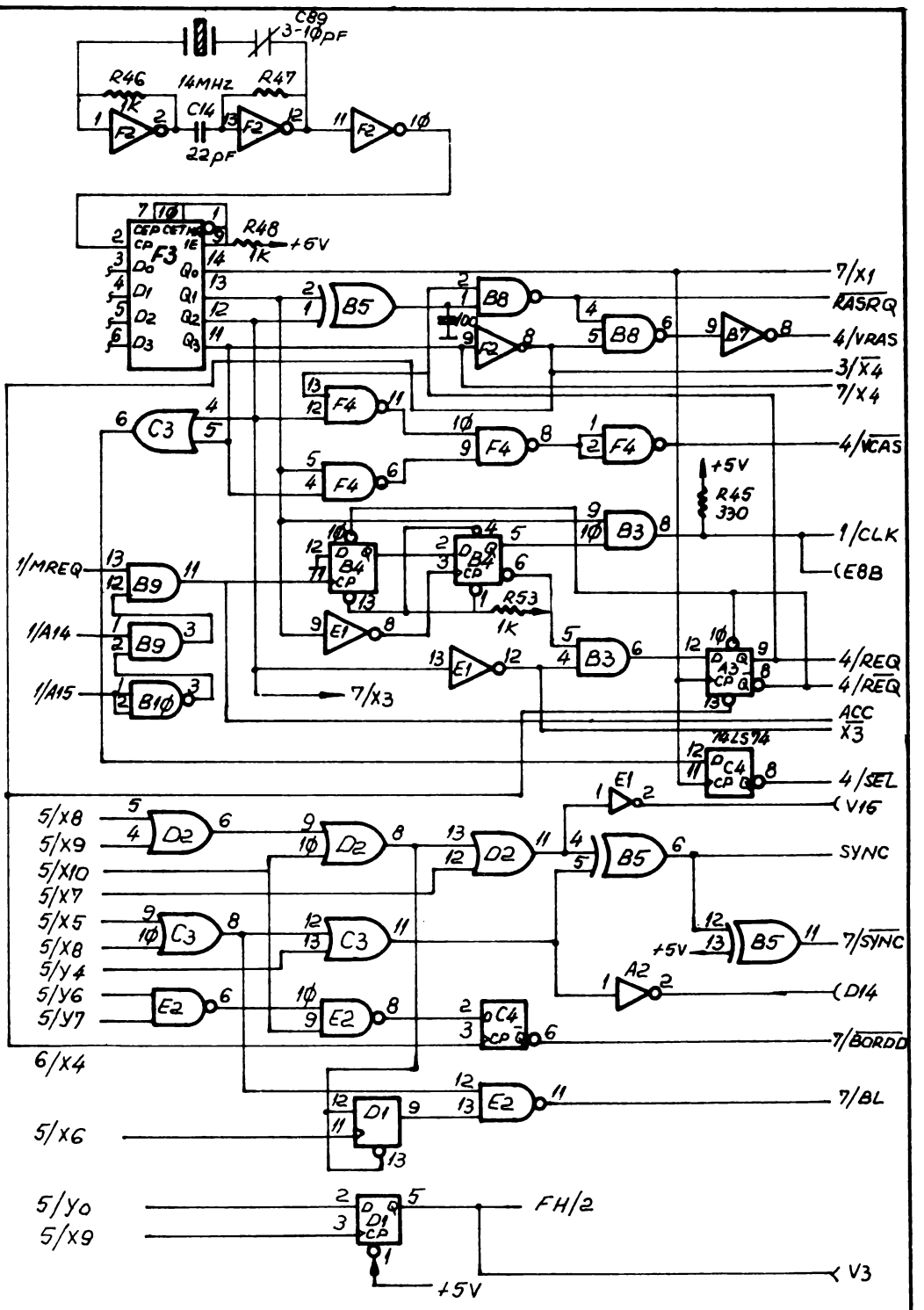
HC 85 LOGIC DIAGRAMS			URC
I.C.E.	Rev.	869 400 001	
		32K RAM	Fila 3 din 10



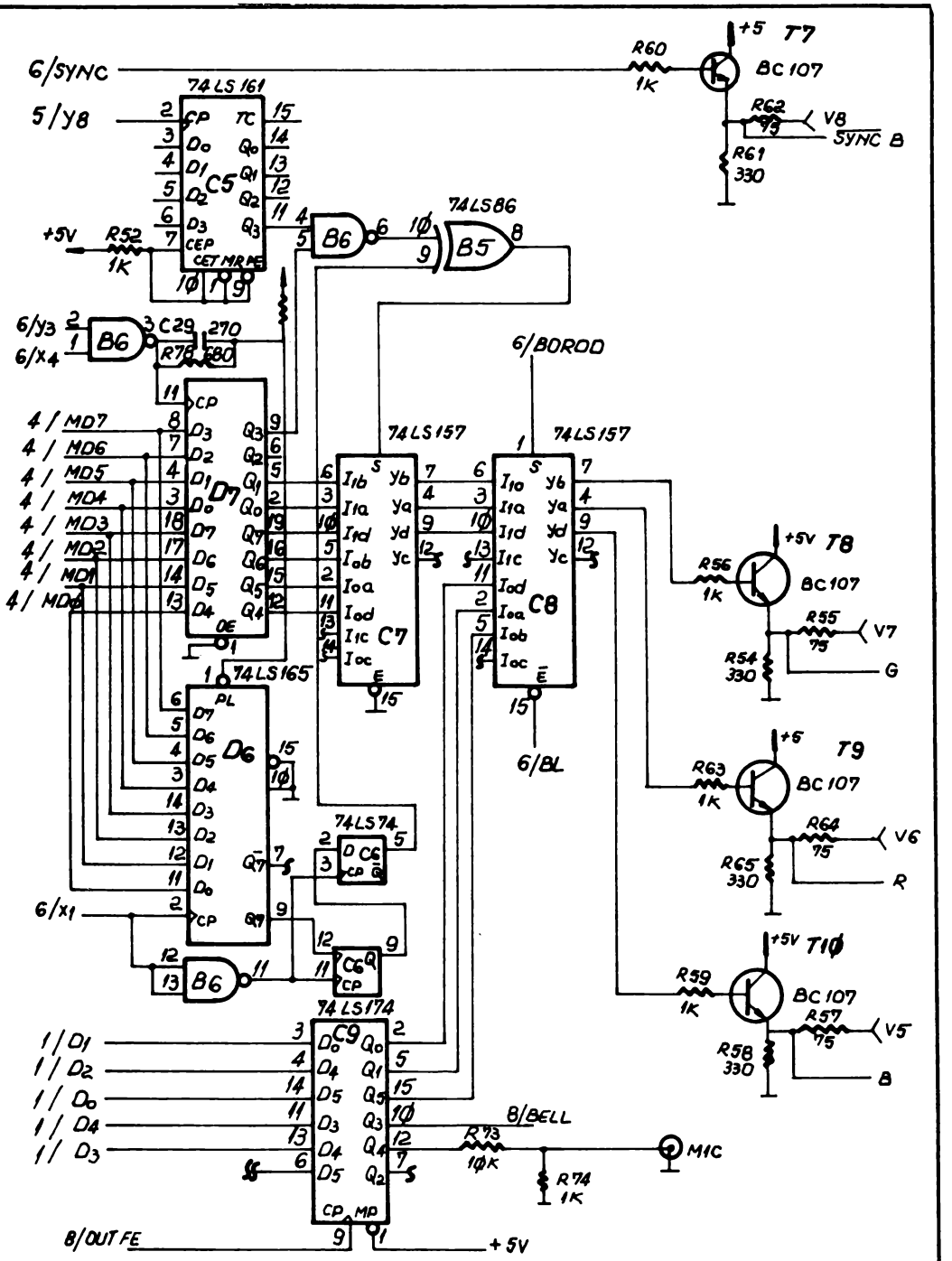
HC 85 LOGIC DIAGRAMS			URC
I.C.E.	Rev.	869.400.001	
		VIDEO RAM	Fila 4 din 10



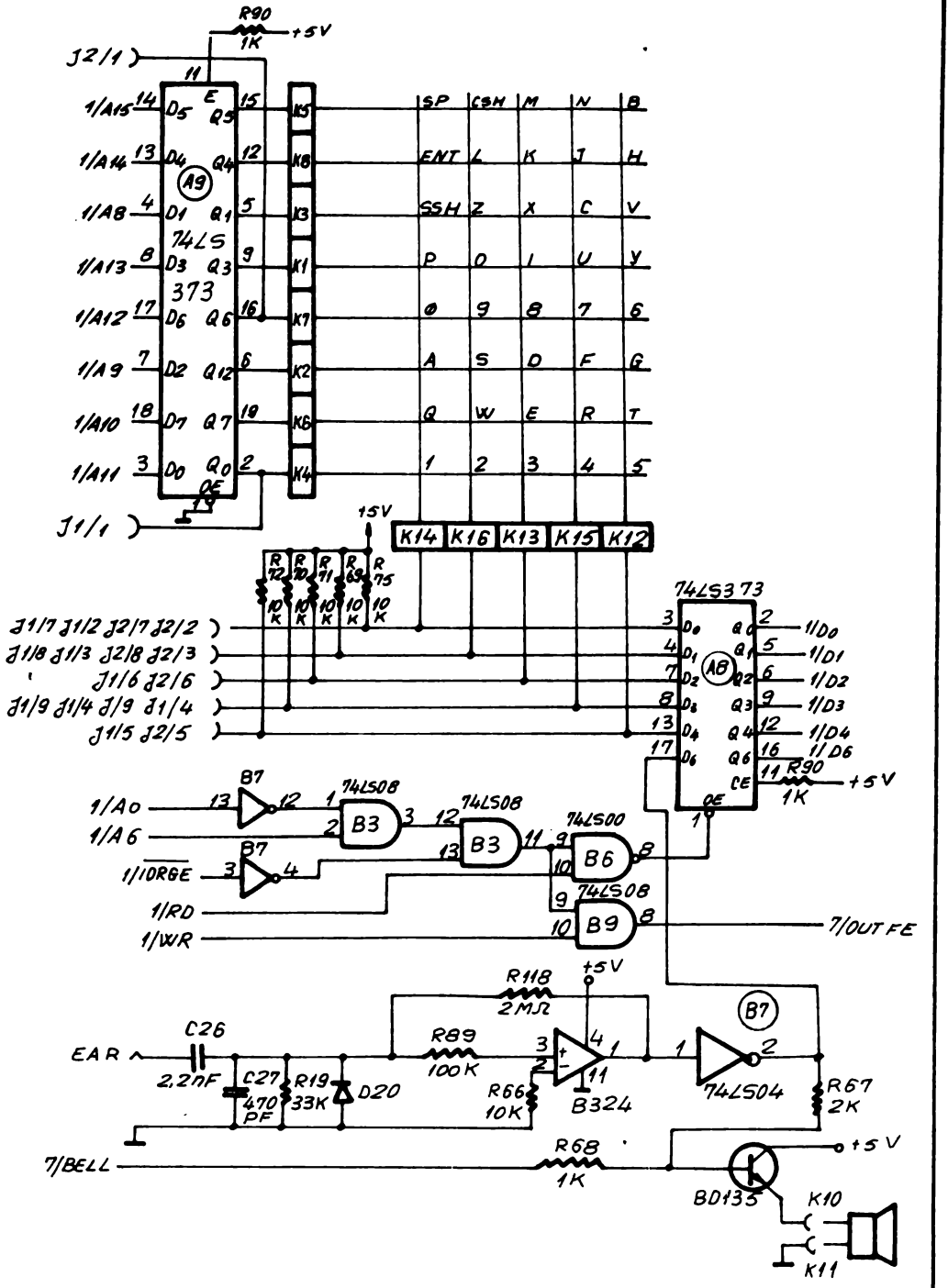
HC 85 LOGIC DIAGRAMS			URC
I.C.E.	Rev.	869.400.001	
		SINCRO GENERATOR	Fila 5 din 10



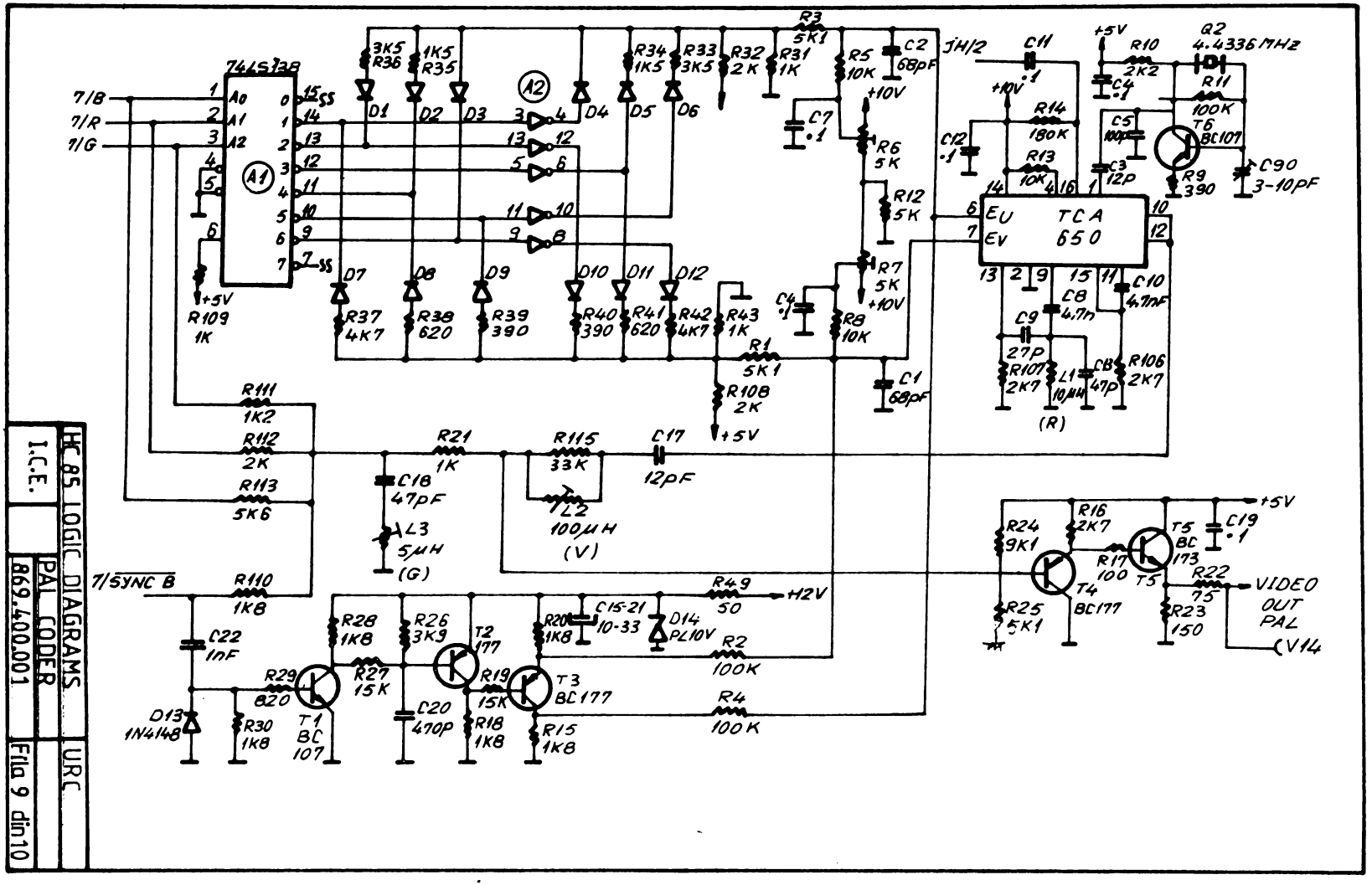
HC 85 LOGIC DIAGRAMS			URC
I.C.E.	Rev.	869.400.001	
		TIMING	Fila 6 din 10



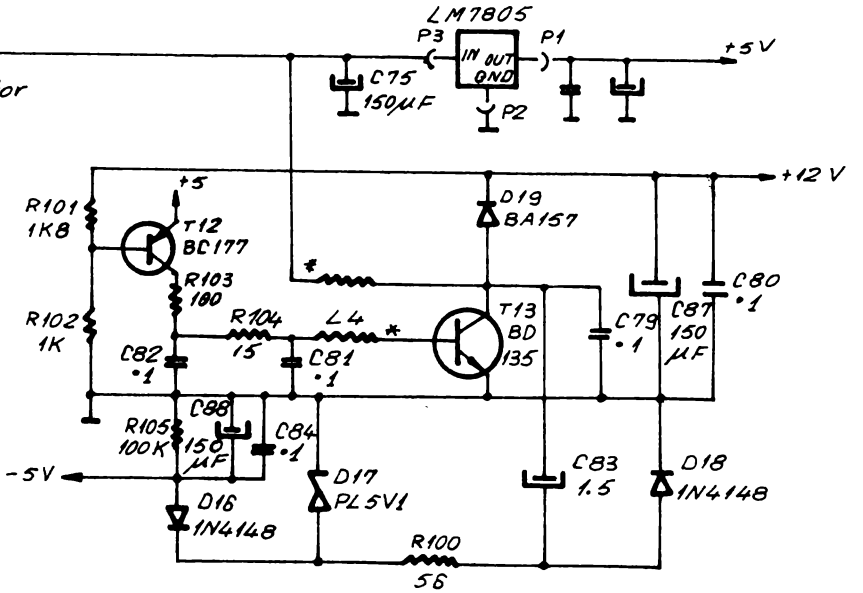
HC85 LOGIC DIAGRAMS			URC
I.C.E.	Rev.	869 400 001	
		ATTRIBUTES	Fila 7 din 10



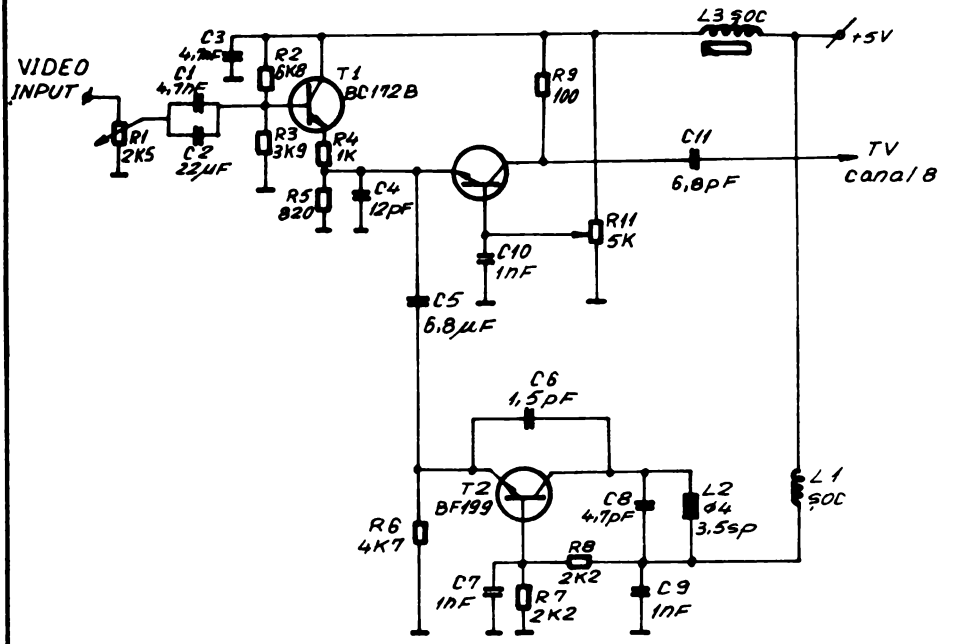
HC 85 LOGIC DIAGRAMS			URC
I.C.E.		KEY BOARD & CASS	
		869.400.001	Fila 8 din 10



I.C.E.	HC 85 LOGIC DIAGRAMS	URC
	PAL CODER	
869.400.001		
Fila 9 din10		

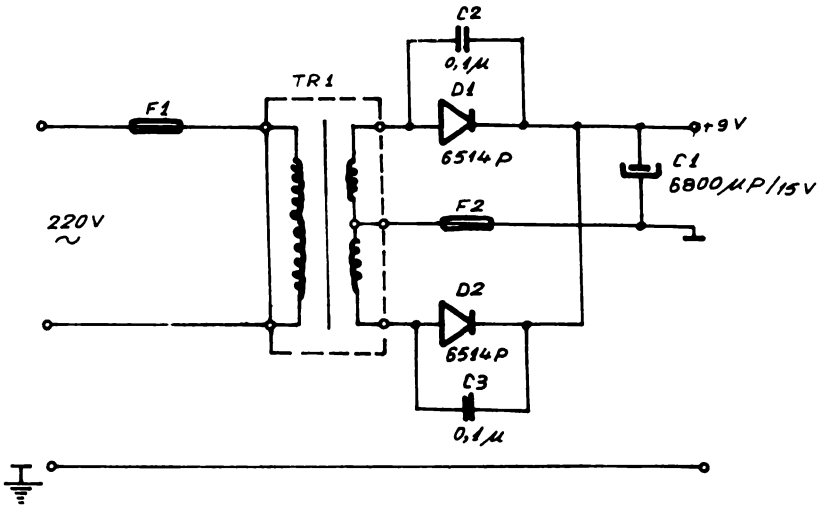


HC 85 LOGIC DIAGRAMS			URC
I.C.E.		SUPPLY	
		869.400.001	Fila 10 din 10

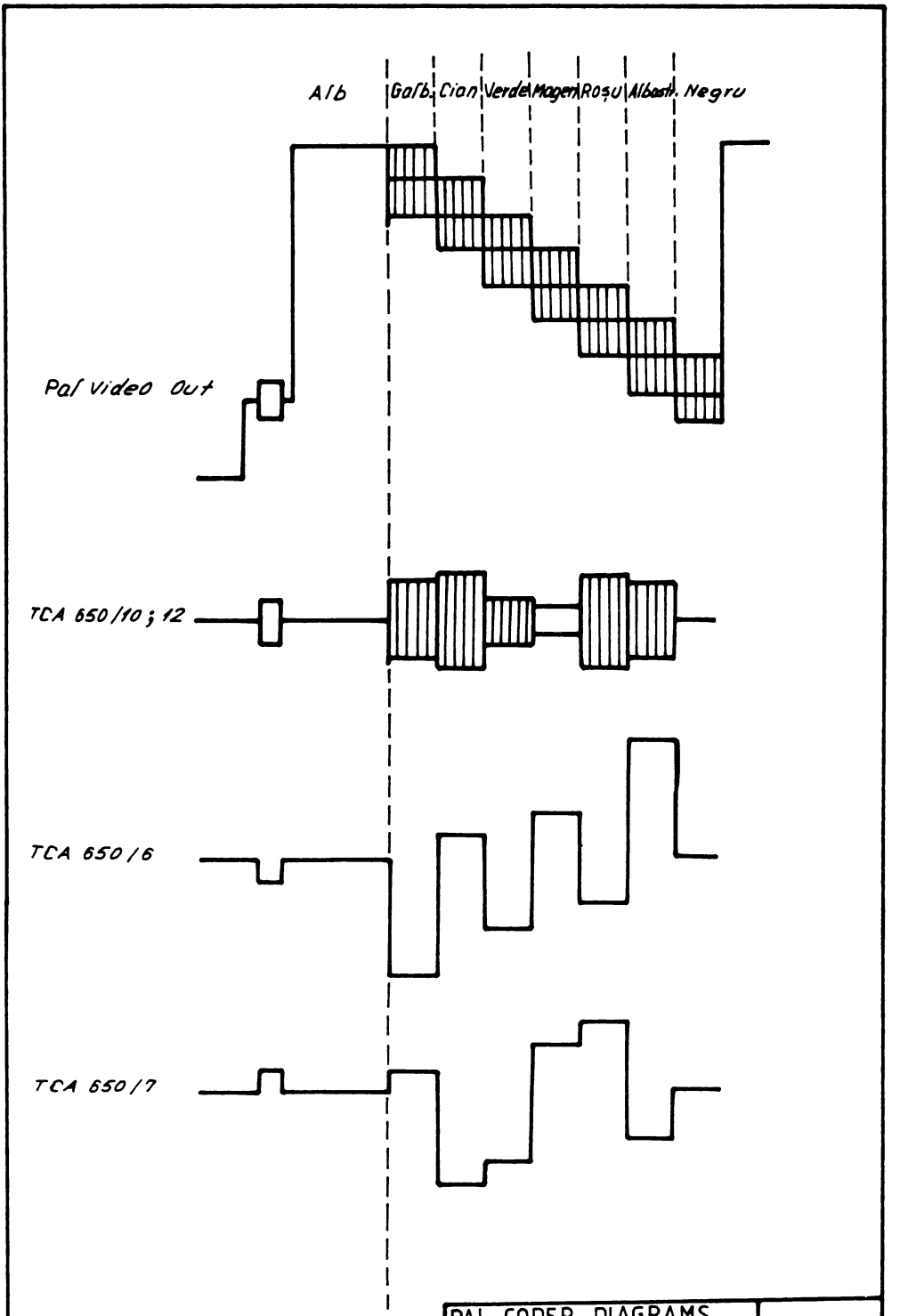


NOTA: Modulatorul se fabrică la Electronica.

HC 85 LOGIC DIAGRAMS			URC
I.C.E.		TV MODULATOR	
		869.400.001	Fila 1 din 1

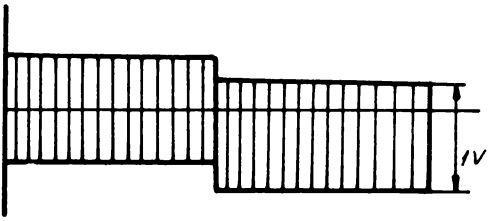


HC 85 LOGIC DIAGRAMS			URC
I.C.E.		HC POWER SUPPLY	
		869.400.001	Fila 1 din 1

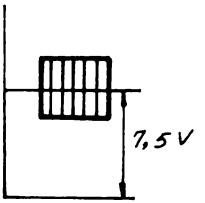


PAL CODER DIAGRAMS			
I.C.E.		869.400.001	Fila 1 din 2

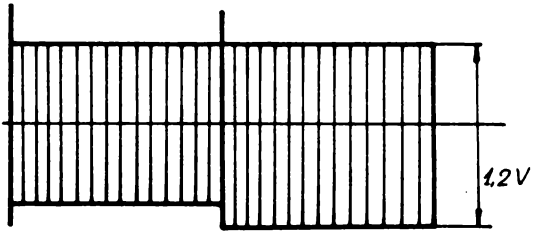
AC
TCA 650/13



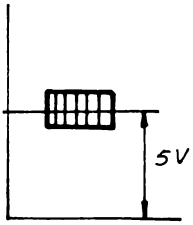
DC



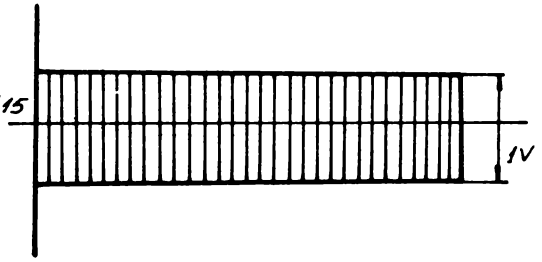
AC
TCA 650/9



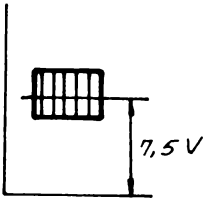
DC



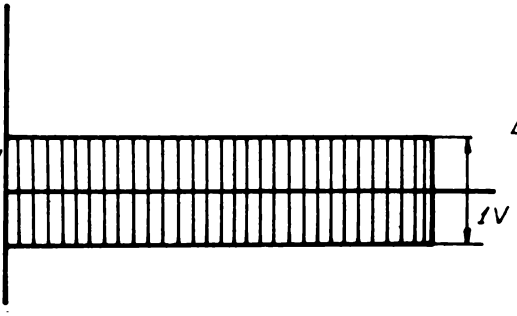
AC
TCA 650/15



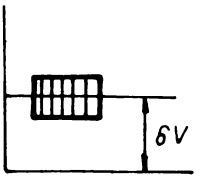
DC



AC
TCA 650/11



DC



PAL CODER DIAGRAMS			URC
I.C.E.		869.400.001	Fila 2 din 2

TĂIAȚI ÎN LUNGUL LINIEI

INDOIȚI INDOIȚI

De la :

Numele : .....

Funcția : .....

Adresa la .....

serviciu : .....

← SE PREȘEAZĂ AICI →

IMPRIMAT PENTRU CORESPONDENȚĂ TEHNICĂ ȘI COMERCIALĂ

Către :

INTREPRINDEREA DE CALCULATOARE ELECTRONICE

Compartimentul Pregătire a Documentației pentru Beneficiari

Str. Ing. GEORGE CONSTANTINESCU Nr.2 - 78.009.- BUCUREȘTI

INDOIȚI INDOIȚI

← Se aplică adeziv aici →

IMPRIMAT IN I.C.E.

NOTA EDITORULUI

ACEST FORMULAR ESTE DESTINAT SPORIRII RELAȚIILOR DE COLABORARE ÎNTRE BENEFICIARI ȘI INTREPRINDEREA NOASTRĂ

GREȘELILE SEMNALATE, SUGESTIILE PENTRU COMPLETAREA SAU REDUCEREA MATERIALULUI TEMATIC PRECUM ȘI COMENTARIILE GENERALE VOR FI FOLOSITE PENTRU RIDICAREA NIVELULUI CALITATIV AL PUBLICAȚIILOR NOASTRE

VĂ RUGĂM SĂ DEFINIȚI SUCCINT LOCAȚIA COMENTATĂ FOLOSIND CODUL PUBLICAȚIEI, NUMĂRUL DE PAGINĂ RÎND ȘI CARACTER, DUPĂ CAZ.

FIȘA PENTRU COMENTARII :

Titlul publicației _____

_____

codul _____ locația _____

_____

ice felix®
78009 2 G.Constantinescu St.,BUCAREST 2,ROMANIA,Tx.11626 felix r, ☎ 886030
Data Processing Systems

ELECTRONUM FOREIGN TRADE COMPANY
70201 33 Al.Sahia St. BUCHAREST 2 ROMANIA
P.O.Box 1380 Tel. 90/13 88 37 Telex 11547 : 11584