N A S C O M N Y T

1979/1980

NASCOM BRUGERGRUPPE, SIDEVOLDEN 23, 2730 HERLEV;(02) 91 71 82 gironr.8-13 50-29

HEJ!!!

Dette er første nr. af meddelelser fra Nascom brugergruppe. I bedes venligst besvare spørgeskemaet bagerst i bladet og fremsende det til ovenstående adresse.

På samme adresse modtages også programmer (meget gerne i en kopibar form) fra alle og i alle sværhedsgrader.

Vi vil efter modtagelsen af skemaer fastlægge gruppens arbejde, som vi håber vil blive til glæde og fornøjelse for alle parter.

Da dette nr. ikke er den rene filantropoli, bedes I indsende 8,50 kr. på vedlagte girokort for dette første nr. (til dækning af kopiudgift og porto). Men derefter må vi blive enige om en eller anden form for betaling for medlemsskab (jævnfør spørgeskema).

Indhold:

Vi håber på mange reaktioner på dannelsen af denne danske N A S C O M B R U G E R G R U P P E.

sir Asbjørn

FLASHING SIGNAL:

Det er nødvendigt ved særlige programmer at have et 'flashing' signal. Denne routine tænder og slukker et bogstav med variabel tidsinterval for tænd og sluk.

```
0c50    21 E0 09    LD HL,9E0H      sæt pointer til midt skærm
 C53    0E 3F       LD C,?          sæt ? i C
 C55    71          LD(HL),C        sæt ? på skærmen
 C56    CD 80 0C    CALL C80H       Kald on routine
 C59    0E 20       LD C,space      sæt blank i C
 C5B    71          LD(HL),C        sæt karakter på skærm
 C5C    CD 80 0D    CALL D80H       kald off routine
 C5F    C3 53 0C    JUMP C53        hop til start af on/off pgr.
-----
 C80    06 40       LD B,40H        sæt tidspause
 C82    CD 35 00    Call KDEL       delayroutine i monitor
 C85    10 FB       DJNZ C82H       gentag indtil B er 0 (7ms x 64)
 C87    C9          RET             tilbage til hovedprogram
-----
 D80    06 40
 D82    CD 35 00
 D85    10 FB
 D87    C9
```

Tændtiden kan reguleres i C81H
Sluktiden kan reguleres i D81H

→

Opgaven:

Tænkes løst v.h.a. basic.
Kender du udtrykket primtalsørken?
Det er en række tal,hvor der ikke findes et primtal. Prøv om du kan finde en sådan ørken, hvor der mindst er 50 sammensatte tal mellem 2 efterfølgende primtal.
Svar bringes i næste nr. og eventuelle løsninger kan indsendes til redaktionen

Nyttige routiner i Z80 kode:

multiplikation 8bitx8bit.

faktorer i E og C,produkt i AE

```
AF          mult8   XOR A        CLEAR ACC
0608                LD  B,8      SET COUNTER
CB1B                RR E         1. FAKTOR BIT I CY
3001                JR NC,$+3    DET VAR NUL
81          M8      ADD A,C      ADD MULTIPLIKAND
1F                  RRA          ROTET ACC IND I CY
CB1B                RR E
10FB                DJNZ M8      COUNTER EJ NUL
C9                  RET
```

multiplikation 16bitx16bit

faktorer i DE,AC, produkt i HLAC

```
210000      mult16  LD HL,0       CLEAR ACC
0610                LD B,16      SET COUNTER
1F                  RRA          1.FAKTOR I CY
CB19                RR C
3001        M16A    JR NC,M16B   DET VAR NUL
19                  ADD HL,DE    ADD MULTIPLIKAND
CB1C        M16B    RR H         ROTER ACC IND I CY
CB1D                RR L
1F                  RR A
CB19                RR C
10F4                DJNZ M16A    COUNTER EJ NUL
C9                  RET
```

Division 8bit med 8bit

E er dividend, C er divisor, A er kvotient og B er resten.

```
AF          dv8       XOR A       CLEAR ACC
0608                  LD B,8      LOOP COUNTER
CB13        d8a       RL E        ROTER CY IND I ACC DIVIDEND
17                    RLA         CY VIL VÆRE UDE
91                    SUB C       FORSØG AT TRÆKKE DIVISOR FRA
3001                  JR NC,$+3   SUBT? GODT NOK
81                    ADD A,C     GENSKAB ACC  SÆT CY
10F7                  DJNZ D8a
47                    LD B,A      PUT REST I B
7B                    LD A,E      HENT KVOTIENT
17                    RLA         SKIFT SIDSTE RESULTAT BIT
2F                    CPL         KOMPLEMENTER BITS
C9                    RET
```

Division 16bit med 8bit

AE dividend, C divisor, E kvotient, A resten og CY ON for overflow
(hvis A>=C som input)

```
B9          dv168     CP C        CHECK FOR OVERFLOW
3F                    CCF
D8                    RET C       A>= C
0608                  LD B,8      LOOP COUNTER
1805oo                JR dv168c
91          dv168a    SUB C       SUBT DIVISOR
37          dv168b    SCF         TVING RESULTAT TIL1
05                    DEC B
280c                  JR Z,DV168d COUNTER ER 0
CB13        dv168c    RL E        TOTER CY IND I ACC DIVIDEND
17                    RLA
38f6                  JR C,dv168a BEHØVES 9 BIT ACC
91                    SUB C       PRØV AT FRATRÆKKE DIVISOR
30r4                  JR NC,dv168b SUBT OK , UD A 1
81                    ADD A,C     GENDAN ACC
A7                    AND A       GØR CY TIL 0
10F4                  DJNZ dv168c  COUNTER IKKE 0
CB13        dv168d    RL E        HENT SIDSTE RESULTATBIT,0 CY
C9                    RET
```

KØB - SALG

TINY basic +super tiny basic sælges til EPROMs pris 330 kr. Kræver udvidet N1. 02-91 71 82

Fra London importeres NAS-SYS til Nascom 1 i 2 EPROM sælges med klub-rabat til p.t. 301,50 kr. (klubrabat 15% på 25 pund).

Der er forhandlinger om ydelse af 10% rabat på udstyr til Nascom indkøbt gennem klubben. Nærmere oplysninger i næste nr.

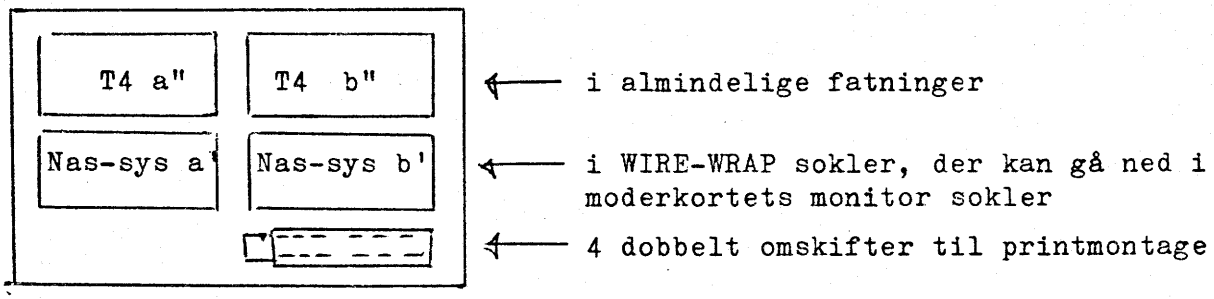
PRINTER ØNSKE$!!

Redaktionen

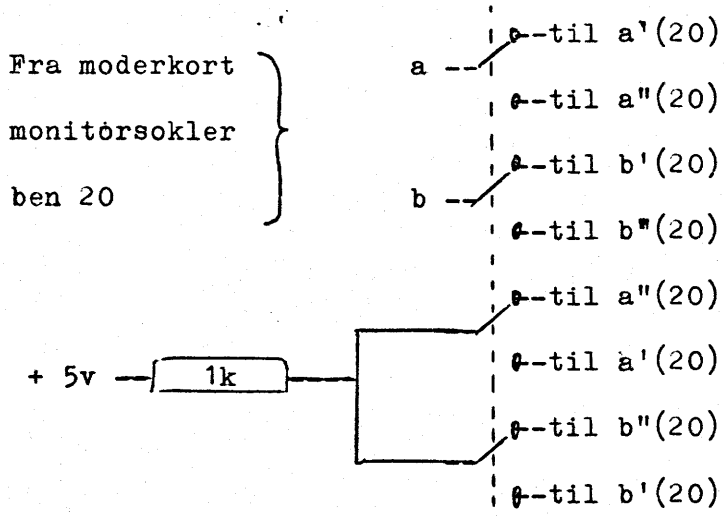
```
  0 rem asbjørn lind
 10 cls
 20 print "programmet finder ugedage i det 20. årh."
 30 read a$
 40 dim u$(7)
 50 print
 60 for k=1 to 7
 70 read u$(k)
 80 next
 90 input"dag,måned,år:";d,m,a
100 if d<0 or d>31 then 140
110 if m<0 or m>12 then 140
120 if a<1900 or a>1999 then 140
130 goto 150
140 print"forkert datoangivelse":goto 90
150 a=a-1900
160 if m<3 then a=a-1:m=m+12
170 m=m-3
180 x=int(365.25xa)
190 y=int((153xm+2)/5)
200 n=x+y+d-306
210 r=n-int(n/7)x7
220 print"det er en";u$(r+1):print:print
230 goto 90
240 data, mandag,tirsdag,onsdag,torsdag
250 data fredag,lørdag,søndag
```

NAS-SYS på NASCOM 1

(samtidig med T4 eller T2)



Printpladen parallelforbinder alle ben på a" og a' og på b" og b' undtagen ben 20, der er ført til den 4-dobbelte omskifter således:



Fordel: Du kan bruge alle dine gamle programmer, samtidig med at få NAS-SYS fordele - bl.a. en god skærmeditering - basic bliver en drøm at arbejde med - memory bliver overhovedet ikke berørt af et monitorskift. Du kan bruge programmer beregnet til N2.

Ulempe: Du bliver nød til at lære nye monitorroutiner at kende.

SIDSTE NYHEDER,

der er udviklet et styrekort til miniflop (4 på et kort) med dobbelt side og dobbelt density indeholdende et CMP styresystem.

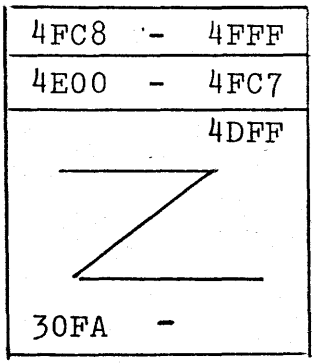
Den varmeste nyhed:

der er under udarbejdelse et interfacekort (dansk produceret) til 2 stk. Philips minicassetteenheder (6000 baud), hvor man forsøger at oversætte COMAL til Nascom.

Basic renumber(til N1 og N2 16k udgave).

Denne renumber er skrevet i maskinkode med start i $4E00_H$.
Den skifter nr. på alle sætninger og på alle henvisninger efter GOTO, GOSUB, THEN, RESTORE, ON .. GOTO og ON .. GOSUB.

Da Nascoms 8k basic er ret uigennemsigtig og da programmet skal fylde mindst muligt er der følgende skønhedsfejl!
Hvis der ikke er plads til det nye linienr. efter henvisningerne, sættes en stjerne og der udskrives:" EJ UDFØRT : nnnn" med det nr. der skulle havet stået der.. Hvis der er flere af disse udskrives de i nr. rækkefølge med laveste nr. først!Man kan afhjælpe dette problem ved at gøre god plads efter henvisningerne.
Hvis det nye nr. fylder mindre sættes stjerne bag tallet, men det betyder intet for kørslen.
Hvis der er henvisning til ikke eksisterende linienr., sættes stjerner, der udskrives ingen fejlmelding!!!!



Stack til renumber
renumber routinen
stack 2 til renumber + stak til basic
basicprogram

Der er ikke taget højde for følgende: At renumber stack 2 går ned i basicprogrammet. Det vil dog kun ske med meget lange basicprogrammer. Hver basiclinie bruger 5 bytes - en ON linie bruger 3x(antallet af henvisninger)+2 i stack 2.

Renumber i brug.

Når basic programmet spørger memory størrelse tastes:19960_{nl}
Når du ønsker at renumber tryk på reset - tast E $4E00_{nl}$.
Hvis der ingen fejl er skrives "slut" og en tilfældig tast nedtrykkes og du er tilbage i basic igen.
Ved fejl skrives fejlmelding, er der flere end 11 stoppes programmet, der kan fortsættes efter afskrift ved at trykke tilfældig tast (skærm slettes), derefter som ovenfor.
Programmet går ud fra at du starter med linienr. 10 med en linie-

afstand på 10.
Hvis dette skal ændres gøres følgende:

DOKE 19976,liniestart: DOKE 20293,linieafstand nl

Renumber med nascom 2

Der er i programmet søgt undgået monitorroutiner så vidt muligt.
Ved brug i N2 skal der ændres følgende:

4E18	30	rettes til	4E18 10
4EC0	30	- -	4EC0 10
4F57	CD 3E	- -	4F57 CF 00
4F7B	1E	- -	4f7B 0C
4FA9	CD 3E	- -	4FA9 CF 00
4FAD	0C 10	- -	4FAD fd ff

RENUMBER ROUTINE

4E00 - 4FC8

4E00	FD	21	FF	4D	31	FF	4F	21	58
4E08	0A	00	22	C7	4F	21	0A	08	CB
4E10	22	CF	4F	CD	74	4F	21	FA	49
4E18	30	CD	7E	4F	CA	BA	4E	D5	D7
4E20	EB	ED	52	E5	C1	C5	3E	88	C9
4E28	D5	E1	C1	C5	E5	ED	B1	28	5D
4E30	1F	FE	88	28	15	FE	8B	28	11
4E38	13	FE	8C	28	0B	FE	A9	28	25
4E40	05	E1	E1	E1	18	D3	C6	3C	23
4E48	C6	1A	C6	02	C6	01	18	D9	F6
4E50	F5	C5	0E	00	E5	FD	75	00	BD
4E58	FD	74	FF	DD	21	CA	4F	FE	2B
4E60	FF	20	04	2B	36	2C	23	7E	FF
4E68	FE	20	28	4B	FE	2C	20	02	93
4E70	36	FF	FE	30	FA	7C	4E	FE	E3
4E78	3A	FA	AC	4E	78	B7	28	27	72
4E80	81	FD	77	FE	21	00	00	DD	BF
4E88	21	CA	4F	29	E5	29	29	D1	41
4E90	19	DD	7E	00	D6	30	5F	16	CD
4E98	00	19	DD	23	10	ED	FD	75	6E
4EA0	FD	FD	74	FC	CD	89	4F	E1	DE
4EA8	C1	F1	18	81	36	2A	DD	77	F5
4EB0	00	04	DD	23	23	18	B0	0C	F9
4EB8	18	FA	FD	E5	DD	E1	21	FA	D3
4EC0	30	CD	7E	4F	FD	21	FF	4D	42
4EC8	CA	8F	4F	D5	E5	FD	E5	E1	3B
4ED0	DD	E5	D1	B7	ED	52	ED	5B	EF

```
4ED8  C7 4F 28 63 C5 FD 6E FD F4
4EE0  FD 66 FC ED 42 20 52 19 47
4EE8  DD E5 DD 21 BD 4F FD E5 E4
4EF0  FD 21 CA 4F 3E 2F DD 4E 0D
4EF8  00 DD 46 01 C6 01 ED 42 60
4F00  F2 FC 4E 09 FD 77 00 FD 05
4F08  23 DD 23 DD 23 0D 20 E4 8B
4F10  01 05 00 21 CA 4F 3E 30 0D
4F18  BE FA 20 4F 0B 23 18 F8 CC
4F20  FD E1 FD 7E FE B9 FA 4F C8
4F28  4F 28 04 03 2B 36 20 FD 78
4F30  5E 00 FD 56 FF ED B0 DD A9
4F38  E1 C1 CD 89 4F 18 8E E1 55
4F40  2B 72 2B 73 21 0A 00 19 0E
4F48  22 C7 4F E1 C3 C1 4E E5 67
4F50  C5 ED 5B CF 4F 21 40 00 2B
4F58  19 3E 0B BC 20 06 CD 3E F6
4F60  00 CD 74 4F E5 0E 0A 22 5E
4F68  CF 4F D1 21 B3 4F ED B0 66
4F70  C1 E1 18 C1 21 0A 08 22 8F
4F78  CF 4F EF 1E 00 C9 5E 23 3C
4F80  56 23 4E 23 46 23 7A B3 4F
4F88  C9 11 FB FF FD 19 C9 31 BB

4F90  FF 4D ED 5B CF 4F 21 18 CA
4F98  00 19 22 CF 4F 01 04 00 45
4FA0  21 AF 4F ED 5B CF 4F ED 61
4FA8  B0 CD 3E 00 C3 0C 10 53 E4
4FB0  4C 55 54 45 4A 20 55 44 3C
4FB8  46 30 52 54 3A 10 27 E8 7C
4FC0  03 64 00 0A 00 01 00 00 81
```

Benchmarks

(MÆRKER I EN SNEDKERS HØVLEBENK!)

BM1-BM8 BRUGES TIL AT BEDØMME EN COMPUTERS INTERNE HASTIGHED.

DER FINDES OGSÅ TEST FOR DISKRUTINER, UDSKRIFT OG INDLÆSNING.

BM1

```
300 PRINT 'S'
400 FOR K=1 TO 1000
500 NEXT K
700 PRINT 'E'
800 END
```

BM2

```
300 PRINT 'S'
400 K=0
500 K=K+1
600 IF K<1000 THEN 500
700 PRINT 'E'
800 END
```

BM3

```
300 PRINT 'S'
400 K=0
500 K=K+1
510 A=K/K*K+K-K
600 IF K<1000 THEN 500
700 PRINT 'E'
800 END
```

BM4

```
300 PRINT 'S'
400 K=0
500 K=K+1
510 A=K/2*3+4-5
600 IF K<1000 THEN 500
700 PRINT 'E'
800 END
```

BM5

```
300 PRINT 'S'
400 K=0
500 K=K+1
510 A=K/2*3+4-5
520 GOSUB 820
600 IF K<1000 THEN 500
700 PRINT 'E'
800 END
820 RETURN
```

BM6

```
300 PRINT 'S'
400 K=0
430 DIM M(5)
500 K=K+1
510 A=K/2*3+4-5
520 GOSUB 820
530 FOR L=1 TO 5
540 NEXT L
600 IF K<1000 THEN 500
700 PRINT 'E'
800 END
820 RETURN
```

BM7

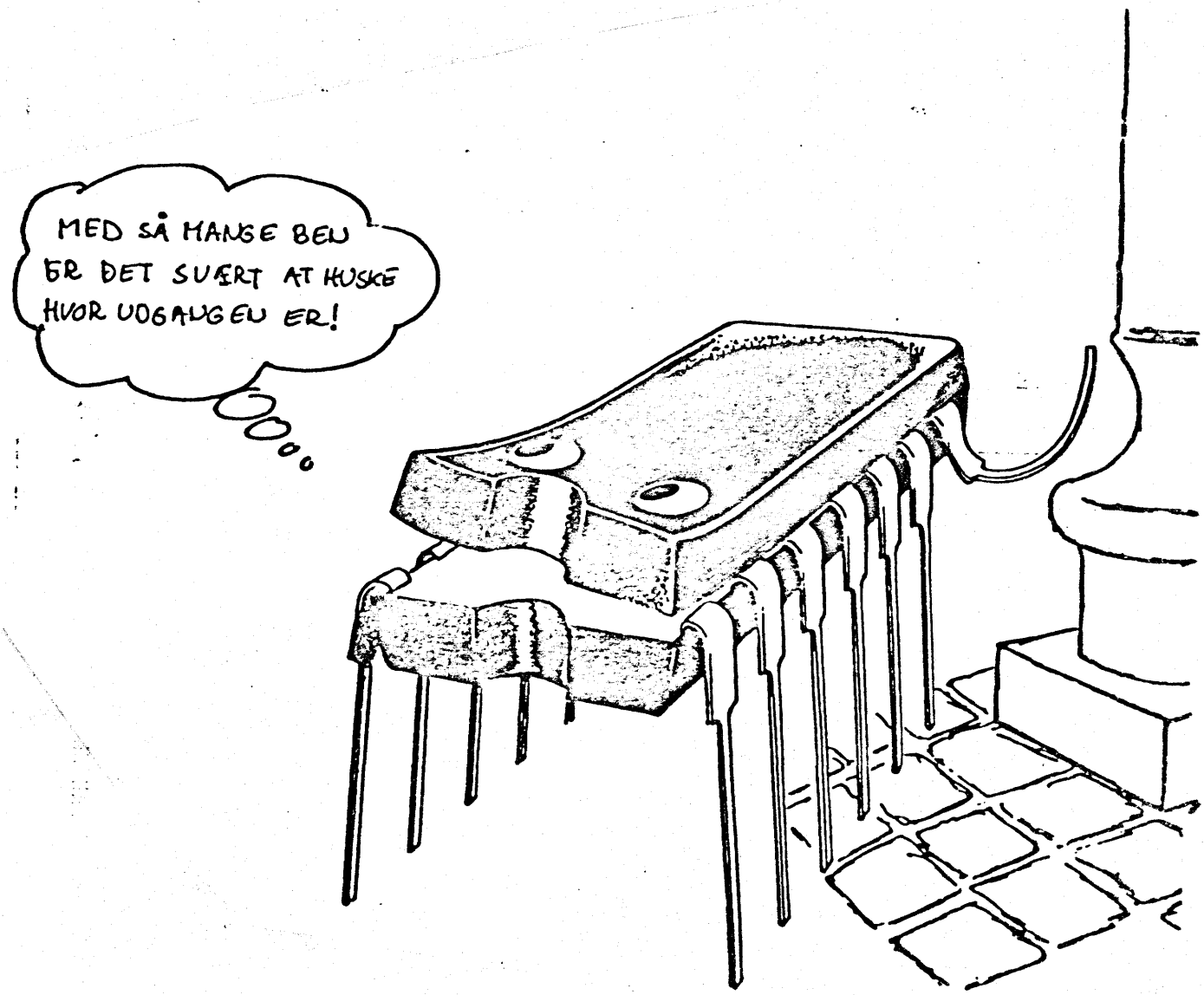
```
300 PRINT 'S'
400 K=0
430 DIM M(5)
500 K=K+1
510 A=K/2*3+4-5
520 GOSUB 820
530 FOR L=1 TO 5
535 M(L)=A
540 NEXT L
600 IF K<1000 THEN 500
700 PRINT 'E'
800 END
820 RETURN
```

BM8

```
300 PRINT 'S'
400 K=0
500 K=K+1
530 A=K↑2
540 B=LOG(K)
550 C=SIN(K)
600 IF K<100 THEN 500
700 PRINT 'E'
800 END
```

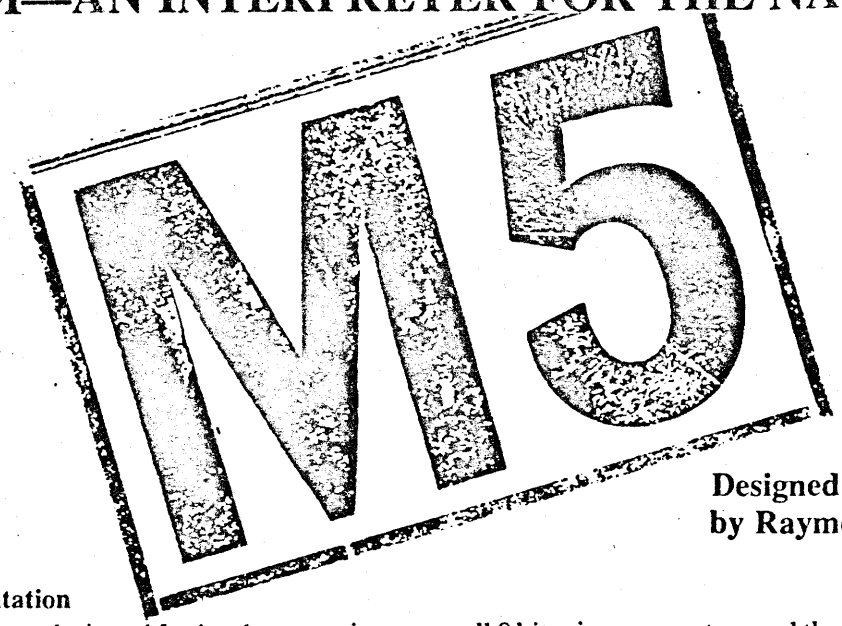
THE Z80 MNEMONICS

MSB / LSB	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	NOP	DJNZ d	JRNZ, d	JR NC, d	LD B,B	LD D,B	LD H,B	LD (HL), B	ADD B	SUB B	AND B	OR B	RET NZ	RET NC	RET PO	RET P	0
1	LD BC, nn	LD DE, nn	LD HL, nn	LD SP, nn	LD B,C	LD D,C	LD H,C	LD (HL), C	ADD C	SUB C	AND C	OR C	POP BC	POP DE	POP HL	POP AF	1
2	LD (BC), A	LD (DE), A	LD (nn), HL	LD (nn), A	LD B,D	LD D,D	LD H,D	LD (HL), D	ADD D	SUB D	AND D	OR D	JP NZ, nn	JP NC, nn	JP PO, nn	JP P, nn	2
3	INC BC	INC DE	INC HL	INC SP	LD B,E	LD D,E	LD H,E	LD (HL), E	ADD E	SUB E	AND E	OR E	JP nn	OUT A,port	EX (SP), HL	DI	3
4	INC B	INC D	INC H	INC (HL)	LD B,H	LD D,H	LD H,H	LD (HL), H	ADD H	SUB H	AND H	OR H	CALL NZ, nn	CALL NC, nn	CALL PO, nn	CALL P, nn	4
5	DEC B	DEC D	DEC H	DEC (HL)	LD B,L	LD D,C	LD H,L	LD (HL), L	ADD L	SUB L	AND L	OR L	PUSH BC	PUSH DE	PUSH HL	PUSH AF	5
6	LD B,n	LD D,n	LD H,n	LD (HL), n	LD B, (HL)	LD D, (HL)	LD H, (HL)	HALT	ADD (HL)	SUB (HL)	AND (HL)	OR (HL)	ADD n	SUB n	AND n	OR n	6
7	RLCA	RLA	DAA	SCF	LD B,A	LD D,A	LD H,A	LD (HL), A	ADD A	SUB A	AND A	OR A	RST 0	RST 10H	RST 20H	RST 30H	7
8	EX AF, AF'	JR d	JR Z,d	JR C,d	LD C,B	LD E,B	LD L,B	LD A,B	ADC B	SBC B	XOR B	CP B	RET Z	RET C	RET PE	RET M	8
9	ADD HL, BC	ADD HL, DE	ADD HL, HL	ADD HL, SP	LD C,C	LD E,C	LD L,C	LD A,C	ADC C	SBC C	XOR C	CP C	RET	EXX	JP (HL)	LD SP, HL	9
A	LD A, (BC)	LD A, (DE)	LD HL, (nn)	LD A, (nn)	LD C,D	LD E,D	LD L,D	LD A,D	ADC D	SBC D	XOR D	CP D	JP Z,nn	JP C,nn	JP PE, nn	JP M,nn	A
B	DEC BC	DEC DE	DEC HL	DEC SP	LD C,E	LD E,E	LD L,E	LD A,E	ADC E	SBC E	XOR E	CP E	See Manual	IN A,port	EX DE, HL	EI	B
C	INC C	INC E	INC L	INC A	LD C,H	LD E,H	LD L,H	LD A,H	ADC H	SBC H	XOR H	CP H	CALL Z,nn	CALL C,nn	CALL PE, nn	CALL M,nn	C
D	DEC C	DEC E	DEC L	DEC A	LD C,L	LD E,L	LD L,L	LD A,L	ADC L	SBC L	XOR L	CP L	CALL nn	See Manual	See Manual	See Manual	D
E	LD C,n	LD E,n	LD L,n	LD A,n	LD C, (HL)	LD E, (HL)	LD L, (HL)	LD A, (HL)	ADC (HL)	SBC (HL)	XOR (HL)	CP (HL)	ADC n	SBC n	XOR n	CP n	E
F	RRCA	RRA	CPL	CCF	LD C,A	LD E,A	LD L,A	LD A,A	ADC A	SBC A	XOR A	CP A	RST 8	RST 18H	RST 28H	RST 38H	F
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	



5: A high level language in 3/4 K!

M5 SYSTEM—AN INTERPRETER FOR THE NASCOM ONE



Designed and Implemented by Raymond Anderson

0.0 The M5 Language

0.1 Nascom Implementation

The M5 interpreter was designed for implementation on small 8 bit microcomputers and the Nascom one standard system was an ideal choice because of its popularity and use of a fairly powerful processor (the Z80).

0.2 Introduction

The M5 system is entered by typing EC60 when M5 has been entered into user RAM. The prompt 'M5:' should then appear at the bottom of the screen, indicating that the system is in the command mode. Commands which may be entered now are:

I	Input	a new program and destroy the previous one. System responds with a newline and waits for the user to enter a program. Input is terminated by a semi-colon, which returns the user to the command made.
L	List	the program currently in store and return to command mode.
R	Run	the current program starting at the first symbol, after printing a newline.
E	Edit	the current program, inserting the character pointer at the place the last instruction was executed—or where an error was found. (See section on editor.)
RS	RESET	the Nascom. This will cause a return to Nasbug. However, the current program and value of X will be maintained ready for typing EC60 to resume programming. RESET must also be used to star a looping program.

0.3 Initialisation

When entering M5 for the first time after loading it, it is best to initialise the user work area by entering and running a null program. This is done as follows: (Underlined characters are typed by the system.)

M5:Input
; (I.E. Terminate input after entering nothing!!!)

M5:R (Null program simply results in a carriage return.)

M5: (System is now initialised.)

0.4 Other commands

M5 will respond with a new prompt to any unknown command letter.

0.5 Errors on input

A backspace will delete the last character only when in input mode. It may seem misleading if used to backspace up a line. (Try it and see!)

Backspaces can be inserted into a string in the program by using the INSERT command in EDIT mode.

Semicolons are illegal characters inside an M5 program.

Shift-Backspace is a legal character in strings.

1.0 BASIC M5 LANGUAGE PRINCIPLES

1.1.0 M5 Arithmetic

The basic elements handled in standard M5 are 16 bit unsigned integers, which are adequate for most games and simple simulation or number manipulation. Numbers are in the range 0 - 65535 (decimal) and are modulo 65536 so 65536 seems the same as zero to the language.

Operators permitted in M5 are:

* (multiply) / (divide) + (add) – (subtract) (-1) & (+1)

the last two are included for faster execution if required, and for compact programming of loop control. (See later).

1.1.1 The Stack

An important aspect of M5 which is quite powerful once it is understood, is its stack based (Reverse polish) expression analysis. This system requires no parentheses and it can be used to evaluate arbitrary expressions quickly. The M5 algebraic system is similar to that found on some calculators and the analogy with a calculator is used in these notes.

1.1.2 The Current Value

On a pocket calculator, the idea of a current value is easy to understand as it appears on the display and is often called "x". In M5 there is also a current value (called "X"), and it is altered only in the following circumstances:

1) If a number appears in the program (not in a string) x takes its value.
2) On encountering an identifier A-7 x takes the value stored there.
3) On encountering a ? (not after =) x takes its value from the keyboard.
4) After a diadic operator (/ - + *) x becomes the result.
5) If x is incremented or decremented (using & or #).

1.1.3 Variables

As in most other languages, M5 has variables A-7 and a special one @.
One of these variables becomes current by simply quoting it in the program.
(point 2 above).
X may be stored in asvariable by simply using =k where k is a variable name.
If = ? is used, the current value (x) is displayed as a decimal number on the screen. (This is how numbers are output in M5).

EXAMPLES (These are all legal M5 programs—Try if unsure!)

(i)	A	What is in location A is now also in x (the current value).
(ii)	ABC	x takes on the values in A then B then C and keeps the value C.
(iii)	23	x becomes 23.
(iv)	23A	x becomes 23 , then x becomes A (i e. the number in A).
(v)	23 456	x becomes 23 and then x becomes 456.
(vi)	A=B	x becomes A , then this value is stored in B.
(vii)	A=B=C=D	x becomes A , then this value is put into B , C and D.
(viii)	A=? D=?	x becomes A and this is displayed, then x becomes B and is displayed.
(ix)	=?=A	x what is in x (left from last program) is displayed and put in A).

N.B. If you want to check what is going on, put the characters: =? in your program and x at these points will be printed.
For neatness and readability use: =? " " which separates No's by a space.
E.G. 23=?" " 1 1 1 1 1 =?" will produce: 00023 11111 as output if run.

1.1.4 Calculating

When a comma is encountered in an M5 program, the value of x is put on the top of the stack—pushing down all other members.
We can represent the stack diagramatically to show what happens.

Imagine the M5 program A,33,,BA where initially A=1 and B=2
step: abcdefgh (could have run 1=A2=3 before)

and follow it step by step:

STEP	SYMBOL	MEANS	x	top-STACK-bottom->	y (top element of stack)
a	A	load A	1	- - - -	unknown
b	,	push x	1	1 - - -	1
c-d	33	load 33	33	1 - - -	1
e	,	push x	33	33 1 - -	33
f	,	push x	33	33 33 1 -	33
g	B	load B	2	33 33 1 -	33
h	A	load A	1	33 33 1 -	33

Note that the top member of the stack is called y .

So far, we have no means of removing items from the top of the stack. We do this by using operators such as + / * - .

The operators work on x and y and put the result in x, removing y from the stack.
Operators therefore do the following:

Operator	Function	Remarks
#	x := x-1	This is the pound sign on the Nascom
$	x := x+1	Much faster than .1+ which is equivalent
+	x := x+y	y is lost. Overflow not detected (M5 2.1)
-	x := y-x	y is lost. Underflow not detected.
*	x := x*y	y is lost. Overflowing bits put in @
/	x := y/x	y is lost. Remainder is put in @

EXAMPLES

Program A,B+=? Initially A=1 B=2
step: abcdef

STEP	SYMBOL	MEANS	x	y	Rest of stack	
a	A	load A	1	?	- - - -	
b	,	push x	1	1	- - - -	
c	B	load B	2	1	- - - -	
d	+	x:=x+y	3	-	- - - -	
e-f	=?	display	3	-	- - - -	(3 is displayed on screen 00003).

The program displays the result of A+B

Program to evaluate (2*3) + (7-2) and display it.

Program 2,3*, 7,2- + = ? i.e. add result of 2,3* to 7,2- and display.

step: abcd e fghi j kl

STEP	SYMBOL	MEANS	x	y	Rest of stack	
a	2	load 2	2	?	-	-
b	,	push x	2	2	-	-
c	3	load 3	3	2	-	-
d	*	x:=x*y	6	-	-	-
e	,	push x	6	6	-	-
f	7	load 7	7	6	-	-
g	,	push x	7	7	6	-
h	2	load 2	2	7	6	-
i	-	x:=y-x	5	6	-	-
j	+	x:=x+y	11	-	-	-
kl	=?	display	11	-	-	-

00011 appears on screen - the answer

NOTE The operators # and & only affect x and are equivalent to ,1- and ,1+ (although faster and shorter).

Imagine we want to store the result of multiplying N by M in A.

In Basic this is A=M*N

But in M5 this is M,N*=A

Here are some further examples of expressions:

BASIC	M5
Z=N*M*A	N,M*,A*=Z OR N,M,A,**=Z
Z=(N+M)*A	N,M+,A*=Z
Z=(N+M)*(A-M)	N,M+,A,'-*=Z
Z=N*N	N,*=Z
Z=N*N*N*N	N,*,*=Z OR N,,,***=Z (N.B. M5 ONLY NEEDS TO GET N ONCE)

1.2 Getting Data In

Data in M5 is Input from the keyboard. The program requests a number from the keyboard when it encounters a LOAD ? i.e. a ? in the program, not following =.

A number is terminated by any non numeric character. Usually the user will type a space after the number and the program will continue on the same line, otherwise he will use a newline after typing the number.

EXAMPLE ? , ? * = ? will prompt for a number, then another and print the product.

1.3 String print

Any string of characters surrounded by quotes ' " ' is printed to the display exactly as written– including newlines etc.

N.B. A jump will find labels in a string so beware of using (in a string.

A nicer version of the program above is:

"NUMBER" ?, "TIMES BY"?*" IS "=?

A newline is produced by a newline between quotes.

1.4 Loops and jumps

A way of repeating operations is almost essential in a programming language. In M5 this is done by using using jumps and labels.

A label is represented in M5 by in where n is any symbol which can be entered at the keyboard.

Examples are: (A (! (1 (.

A jump is represented by)kn where n is a symbol which matches a label, and k is a condition code indicating what condition involving x or x and y must be true for the jump to occur.

Valid condition codes are as follows:

CONDITION CODE CHARACTERS:

Character	Jump occurs if:	Comments:
U	–unconditional–	U stands for unconditional
Z	value of x is 0	Z stands for zero
N	value of x is not 0	N stands for non zero
E	x=y (top 2 on stk)	E stands for equal
X	x=y	X looks like a notequal sign
L	x = y	L stands for less than or equal
G	x = y	G stands for greater than
M	–unconditional–	M is monitor . jump to editor.

EXAMPLES of valid jump symbols are:

)UA)N! (X$)G((Z. matching labels above.

when a jump symbol is reached, the condition indicated by K is tested and if it is found to be true, a jump is made to the first occurence of a label with matching identifier symbol.

EXAMPLES:

```
(i)    2000 (A  "HELLO" #  )NA        prints out "HELLO" 2000 times.

(ii)      0 (A =?  " "  &  )NA        prints out numbers from 0 to 65535
                                      separated by spaces. (Thinks 65536=0 ).

(iii)    (A  )UA                      loops until RESET is pressed.

(iv)   0=N (A N=? &=N . 5555 )GA      prints out numbers from 0 to 5555.
```

2.0 WRITING PROGRAMS

M5 is a powerful language when all its features are properly understood, but it can be a little confusing for the beginner. There is fortunately an easy way of generating programs which can be used until familiarity with M5 is achieved. The method is to write the program in a more standard language and then translate into M5. While this method does not exploit the valuable 'current variable' feature of M5, it will yield workable programs which are easier to follow in many ways. The program can then be optimised when it has started to work.

EXAMPLE: A Program to print a table of squares from 1 to 30.

BASIC	M5
10 PRINT "TABLE OF SQUARES"	"TABLE OF SQUARES "
20 N=0	0=N
30 N=N+1	(B N,1+ = N
40 PRINT N, N*N	N=? " " N,N*=? " "
50 IF N = 20 GOTO 30	N , 20)XB
60 END	)M

NOTE: Newlines in output must be included between quotes in M5 programs. The numbers in M5 are not spaced on output, hence the space in the line equivalent to line 40.

The M5 produced will be completely sound and will run at about the same speed as the tiny Basic program.

If the M5 is optimised, keeping N in "x" as much as possible and using the free layout and the & operator, the speed will be considerably faster, perhaps 4-5 times faster than a fast tiny basic.

Optimised:

```
"TABLE OF SQUARES
" 0=N (B N&=N=? " " ,*=? "
"N,20 )XB )M
```

3.0 THE EDITOR

3.0 Introduction

The M5 Editor is entered by typing E when in the command mode.

The edit prompt of E: will appear when the editor is ready to accept input.

The editor will show the point where the last instruction was executed when it is entered by positioning a cursor at this location. The cursor is a shaded in square which is denoted here by a — (underline).

The cursor indicates the current position of the character pointer, and the character pointed at by the cursor appears at the top right of the screen. All manipulation of text is done relative to this cursor because there are no line numbers in M5.

The character indicating end of file in M5 is a null character which appears as a box when it is pointed at.

A hazard in the M5 interpreter is that the pointer can be moved into the actual M5 Interpreter. A Rule must therefore be: DO NOT use any Delete or insert commands unless you can see where the pointer is positioned.

3.1 Commands

To manipulate the text of a program, the user must be able to position the cursor in the required area and then operate on the text. Commands to move the pointer are as follows:

>	Move cursor forward one place.
<	Move cursor backward one place.
R	Rewind–i.e. move cursor to the start of the file.
N	Move the cursor to the start of the next time (stop at end of prog.)

These commands may be repeated and if followed by a newline, will result in a printout of the text with the cursor in its new position.

EXAMPLE: You have typed in a program as follows:

```
(A "HELLO THERE " N=? " IS N
WHAT NUMBER DO YOU WANT" ; . . . . . etc
```

And you want to move the cursor to the spelling error.

Use: RN i.e. move to start, move down a line, move in 5 characters.

Using a space instead of a newline will not print out the text but will carry out the actions and return the edit prompt.

Once we have moved the prompt to where we want to make adjustments we have commands to delete and insert characters.

D	Remove (delete) the character pointed at by the cursor. The cursor now points to the next character along.
Innnn;	Insert the string nnnn before the character pointer. The terminator is a ;* Cursor points to same character.

EXAMPLE: Edit ABCDERTYIJKLMNOP to replace RTY by FGH

ABCDEFRTYJKLMNOP

E:R Move pointer to start the along 7 characters (to Y)

ABCDEF–TYIJKLMNOP Character R appears at top R.H. side of screen.

E:D Delete current character.
ABCDEF–YIJKLMNOP T appears at top right.

E:DD Delete two more.

ABCDEF–JKLMNOP I appears at top right.

E:IGHI; Insert correct characters.

ABCDEFGHI–KLMNOP string now correct– O still current character.

When editing is complete, the command W is used to return to command mode.

4.0 ERROR MESSAGES

When a large program is written concisely in M5, errors may be difficult to detect so good errr diagnostics at runtime were included.

If a syntax error occurs, one of the following messages will appear:

SYM	FRR	x	The symbol x is not allowed in M5 (except in a string).
10	ERR	x	The symbol x is not a valid identifier, and an attempt was made to copy a value into it. (e.g. =x occurred.)
JID	ERR	x	The label x was not found when a jump occurred to it.
JC	ERR	x	The symbol x occurred in a jump condition position and is not a valid code (one of U A N Z X G E M).
	ERR	x	The symbol x caused an error to occur. (Not one of above.)

In addition to giving the error type, the editing cursor is set up to point at the faulty symbol, so when the editor is entered from the monitor to correct the error, the cursor is in the correct position for amendments.

(N.B. in M6, JID errors are detected before the program starts to execute.)

5.0 SAMPLE PROGRAMS IN M5

```
Number summing program    (A"INPUT A NUMBER"?," THANKS
                            NOW INPUT 2 MORE NUMBERS"?,"AND"?"GOOD!
                            THEIR SUM IS "++=? "
                            ")NA "THEIR SUM WAS ZERO - TYPE 0 FOR MORE FUN OR
                            1 TO END "?)ZA "GOODBYE!" )M

Factorial of a number:    1=N ? )78 (A  =M , N# =N M# )NA (8 N=?

MS 24 hour clock:
(N.b. remove all          )US (D  N#=N )ND
 spaces for good          H=?" HRS "M=?" MINS "S=?" SECS
 timekeeping )            " L=N S&=S , T )XD
                            0=S M&=M , T )XD
                            0=M H&=H ,24 )XD
                            0=H )UD
(Start put at end)        (S 1750=L  60=T
                          "SET HRS"?=H"SET MINS"?=M"SECS"?=S"
                          " )UD
```

Note that the main timing loop is at the beginning for higher speed.
1750 is the timekeeping constant. make smaller to speed up clock.

```
Square root of a number:  256=M  ?=N  (1  N,M/ , M )LS +,2/=M )U1
                          (S " "M=?" "
Method used is very fast but a little hard to follow.

Prime numbers:              1=T
                          (N T&&=T
                            1=G
                          (A G&&=G
                            T,G/,G )GP
                            @ )NA )UN
                          (P T=? "
                            " )UN

This can be compacted to only one line of course, ( a bit baffling though ):

1=T(NT&&=T1=G(AG&&=GT,G/,G)GP@)NA)UN(PT=?" ")UN
```

Nye distributører

Fra den 1/1 1980 har distributionen af den engelske microcomputer NASCOM, ændret struktur.

Den tidligere form, Polydatas eneforhandling, er ændret til en forhandlerkæde bestående af fem forretninger/firmaer.

Grunden

Grunden til denne ændring er bl.a., et ønske om at give computerfolk uden for København, en chance til at se NASCOM produkter færdigsamlet – og i funktion.

De fem forhandlere er:

Polydata, Strandboulevarden 17, 2300 København S, Pops Elektronik, Rolighedsvej 17, 1958 København V, O. B. Carlsen, Ørstedsgade 19, 6400 Sønderborg, WK Electronic, Skoletorvet, 8600 Silkeborg, Odense Microcenter, Kratholmsvej 31B, 5260 Odense S.

Vejledning

Disse forhandlere vil altid kunne demonstrere og vejlede i brugen af NASCOM produkter. De kan også reparere eventuelle defekter på disse – hvis det skulle blive aktuelt.

STOP PRESS!

Piazodan
DATARAMA
BERNHARD BANNES ALLE 17A, 2000 F

HANDLER OGSÅ MED NASCOM.

LISTNING AF N5:

KONTROLSUM INDTASTES IKKE!

Addr	Bytes								Bytes								
0C50	D6	3F	CD	01	0E	5E	23	56	24	18	3B	E1	ED	52	EB	18	35	0F
0C60	C3	3E	0E	EF	3F	00	21	00	CA	00	CD	25	0E	CD	14	0E	38	9B
0C70	F8	EB	18	21	62	6B	FD	21	83	0A	0E	AF	FD	46	01	FD	4E	DA
0C80	00	ED	42	38	03	3C	18	F9	Y3	09	C6	30	CD	3B	01	FD	23	BC
0C90	FD	23	0D	20	E5	DD	23	DD	AB	7E	00	FE	20	28	F7	FE	1F	7C
0CA0	23	F3	FE	3F	28	BD	30	A8	C1	FE	2C	28	30	FE	3D	28	33	CC
0CB0	FE	29	CA	74	0D	FE	23	28	77	46	FE	26	28	3F	FE	28	28	E6
0CC0	36	FE	2D	28	95	FE	2A	28	3A	39	FE	2F	28	56	FE	28	28	06
0CD0	0E	FE	22	28	6C	B7	CA	3E	5D	0E	C3	54	0D	D5	18	86	DD	96
0CE0	23	18	B2	DD	23	DD	7E	00	3V	D6	3F	28	88	DA	C7	0D	CD	3V~
0CF0	01	0E	73	23	72	18	9E	E1	AA	19	EB	18	99	13	18	96	1B	95
CD00	18	93	C1	3E	10	21	C0	00	E8	CB	7A	28	04	09	30	01	13	D3.
0D10	3D	28	09	EB	29	EB	29	30	E3	EF	13	19	EC	EB	22	C0	0B	03
0D20	C3	95	0C	42	48	21	C0	00	3F	D1	3E	10	29	EB	29	EB	30	AC
0D30	02	23	B7	ED	42	13	F2	3C	89	0D	09	CB	83	3D	20	EC	18	0A
0D40	DC	DD	23	DD	7E	00	FE	22	A4	CA	95	0C	B7	CA	3E	0E	CD	5A
CD50	38	01	18	ED	D6	30	FE	0A	AC	30	13	21	00	00	DD	7E	00	2V
0D60	DD	23	CD	14	0E	38	F6	E8	75	DD	2B	C3	97	0C	EF	53	59	7E
0D70	4D	00	18	57	DD	7E	01	FE	93	4E	28	31	FE	55	28	5B	FE	00
CD80	5A	28	23	08	E1	E5	B7	ED	AV	52	08	FE	45	28	24	FE	58	DV
CD90	28	23	FE	4C	28	22	FE	47	C1	28	23	FE	4D	CA	3E	0E	EF	40
CDA0	4A	00	DD	23	18	25	7A	B3	61	28	30	18	14	7A	B3	20	2A	B6
0DB0	18	0E	08	18	F3	03	18	F6	0C	08	30	1F	18	03	08	38	1A	91
CDC0	DD	23	DD	23	C3	95	0C	EF	20	49	44	00	EF	20	45	52	52	5A
0DD0	20	00	DD	7E	00	CD	3B	01	61	18	64	D0	4E	02	31	FA	0F	C8
0DE0	21	FE	0E	06	28	7E	23	B8	A1	28	0D	B7	C2	E5	0D	DD	23	95
0DF0	DD	23	EF	4A	00	18	D0	7E	9C	B9	20	EA	E5	DD	E1	C3	95	C3
0E00	0C	07	4F	06	00	21	BE	08	60	09	C9	10	27	E8	03	64	00	6E
0E10	0A	00	01	00	06	30	FE	0A	37	D0	29	54	5D	29	29	19	5F	9A
0E20	16	00	19	37	C9	CD	3E	00	68	C3	3B	01	EF	1F	00	21	FD	61
0E30	0E	23	7E	B7	C8	CD	3B	01	75	18	F7	AF	77	23	77	EF	1F	23
0E40	4D	35	3A	00	CD	25	0E	FE	08	4C	CC	2B	0E	FE	49	CA	D3	&B
0E50	CE	FE	52	20	09	EF	1F	00	F3	DD	21	FD	0E	18	A0	FE	45	6A
0E60	20	DC	DD	E5	E1	4E	36	7F	10	E5	79	32	F6	0B	CD	2B	0E	0D
0E70	E1	71	EF	1F	45	3A	00	CD	2A	25	0E	FE	44	23	3A	FE	1F	7A
0E80	28	E3	FE	3E	20	01	23	FE	17	3C	20	01	2B	FE	52	28	22	B8
CE90	FE	4E	28	34	FE	57	28	A6	69	FE	49	20	DB	CD	25	0E	FE	E6
CEA0	3B	28	D4	E5	4E	77	23	79	2B	B7	20	F9	77	23	77	E1	23	9B
0EB0	18	EA	21	FF	0E	28	18	BF	F0	E5	DD	E1	DD	7E	01	DD	77	19
0EC0	00	B7	28	B3	DD	23	18	F3	6B	7E	B7	28	AB	23	FE	1F	20	3E
0ED0	F7	18	A4	EF	6E	70	75	74	47	1F	00	21	FD	0E	23	CD	25	46
0EE0	0E	FE	3B	CA	3A	0E	77	FE	BC	1D	20	F2	2B	18	F0	D4	00	2C

Execute from 0C60. Program starts at 0EFF.