E GEOGRAFIE
IMATOLOGIE

ISCH DIAGRAM

ASSIFIKATIE VAN BAGNOULS

N POLYGONEN

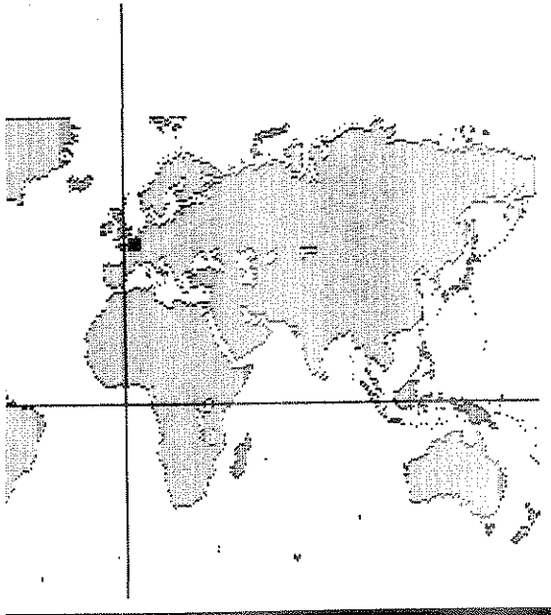
KAARTSCHAALBEREKENINGEN

BEREKENEN VAN DE HELLINGSGRAAD

FIEL EN VERTIKALE OVERDRIJVING

KALENDER

MA+E M DJ-84



DAI MAMIC

25

tweemaandelijks tijdschrift november - december 1984

COLOFON

DAInamic verschijnt tweemaandelijks.
Abonnementsprijs is inbegrepen in de jaarlijkse contributie .
Bij toetreding worden de verschenen nummers van de jaargang toegezonden.

DAInamic redactie :

Dirk Bonné	wdw
Freddy De Raedt	Herman Bellekens
Wilfried Hermans	Frans Couwberghs
René Rens	Guido Govaerts
Bruno Van Rompaey	Daniël Govaerts
Jef Verwimp	Frank Druijff
Cedric Dufour	Willy Coremans

Vormgeving : Ludo Van Mechelen.

U wordt lid door storting van de contributie op het rekeningnr. **230-0045353-74** van de **Generale Bankmaatschappij, Leuven,** via bankinstelling of postgiro
Het abonnement loopt van januari tot december.

DAInamic verschijnt de pare maanden.
Bijdragen zijn steeds welkom.

CORRESPONDENTIE ADRESSEN.
Redactie en software bibliotheek

Wilfried Hermans
Mottaart 20
3170 Herselt
Tel. 014/54 59 74

Kredietbank Herselt
nr. 401-1009701-46
BTW : 420.840.834

Lidgelden / Subscriptions

Bruno Van Rompaey
Bovenbosstraat 4
B 3044 Haasrode
België
tel. : 016/46.10.85

Generale
Bankmaatschappij
Leuven
nr. 230-0045353-74

Voor Nederland :
GIRO : 4083817
t.n.v. J.F. van Dunne
Hoflaan 70
3062 JJ ROTTERDAM
Tel. : (010) 144802

Pour la France :
DAInamic FRANCE
C. Dufour
Rue Lavoisier 9
59149 DUNKERQUE
Tel. 02866 3339

Inzendingen : Games & Strategy

Frank Druijff
's Gravendijkwal 5A
NL 3021 EA Rotterdam
Nederland
tel. : 010/25.42.75

DAINAMIC
PERSONAL COMPUTER USERS CLUB

4		3		2		1	
HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC
1	4096	1	256	1	16	1	1
2	8192	2	512	2	32	2	2
3	12288	3	768	3	48	3	3
4	16384	4	1024	4	64	4	4
5	20480	5	1280	5	80	5	5
6	24576	6	1536	6	96	6	6
7	28672	7	1792	7	112	7	7
8	32768	8	2048	8	128	8	8
9	36864	9	2304	9	144	9	9
A	40960	A	2560	A	160	A	10
B	45056	B	2816	B	176	B	11
C	49152	C	3072	C	192	C	12
D	53248	D	3328	D	208	D	13
E	54344	E	3584	E	224	E	14
F	61440	F	3840	F	240	F	15

belangrijke ASCII-waarden in DAIpc

functie/symbool	HEX	DEC
back-space	8	8
TAB	9	9
linefeed	A	10
clear screen	C	12
CURSOR UP	10	16
CURSOR DOWN	11	17
CURSOR LEFT	12	18
CURSOR RIGHT	13	19
space-bar	20	32
Ø	30	48
A	41	65
a	61	97
pijltje rechts	89	137
pijltje links	88	136
pijltje boven	5E	94
pijltje onder	8C	140
volle blok	FF	255
verticale lijn	A	10
horizontale lijn	B	11
6 hor. lijnen	1D	29

ASCII - HEX - ASCII CONVERSION TABLE

LSD	MSD 0 000	1 001	2 010	3 011	4 100	5 101	6 110	7 111
0 0000	NUL	DLE	SP	0	@	P	`	p
1 0001	SOH	DC1	!	1	A	Q	a	q
2 0010	STX	DC2	"	2	B	R	b	r
3 0011	ETX	DC3	#	3	C	S	c	s
4 0100	EOT	DC4	$	4	D	T	d	t
5 0101	ENG	NAK	%	5	E	U	e	u
6 0110	ACK	SYN	&	6	F	V	f	v
7 0111	BEL	ETB	'	7	G	W	g	w
8 1000	BS	CAN	(	8	H	X	h	x
9 1001	HT	EM	)	9	I	Y	i	y
A 1010	LF	SUB	*	:	J	Z	j	z
B 1011	VT	ESC	+	;	K	[	k	{
C 1100	FF	FS	,	<	L	\	l	|
D 1101	CR	GS	-	=	M	]	m	}
E 1110	SO	RS	.	>	N	↑	n	~
F 1111	SI	VS	/	?	O	←	o	DEL

DAINAMIC

NEWSLETTER 25

Herselt, december '84

Beste Leden,

Met een extra dik nummer (88 pagina's + mini-poster) hebben we deze 25-ste uitgave iets feestelijker willen maken. We zijn ervan overtuigd dat er genoeg materiaal in zit om heel actief de feestdagen door te brengen.

Een terugblik op de voorbije manifestaties :

20 october Nijvel : een 250-tal bezoekers bevestigden dat het een goed idee was om deze dag te organiseren. We mochten een dertigtal nieuwe leden noteren en verschillende DAI-gebruikers maakten voor de eerste keer kennis met onze uitgebreide software-bibliotheek. Onze gelukwensen aan de initiatiefnemers C. Poels en F. Duluins.

17 november Utrecht : groot, druk, onoverzichtelijk... we hebben 's morgens een half uurtje moeten zoeken om onze eigen kramen te vinden ! We hopen dat U ons toch heeft kunnen vinden. Zoals in Nijvel was er veel belangstelling voor onze experimenten met de video camera interface, ook vanwege niet-DAI gebruikers.

24 november Diepenbeek : Het onderwijssymposium georganiseerd door SCHOOL en COMPUTER kende een groot succes. Onze DAI computer was ruim vertegenwoordigd : in de software catalogus vonden wij demonstraties door : Marc Antrop, Bruno Van Rompaey, INDATA, TRON, J. Mergaerts, M. De Jaeger, F. Cerulus en ondergetekende.

Ondanks de aangekondigde posttarief verhogingen kunnen wij de contributie ongewijzigd houden
Subscription prices will be the same in 1985 :

Benelux : 1000 Bfr — Europe : 1100 Bfr — Air Mail : 1500 Bfr

Collective subscriptions (for schools only) :

2 and more subscriptions on the same address and paid together : - 30 % from the second subscription on.

voor België en andere landen : DAInamic subscriptions
B. Van Rompaey
Bovenbosstraat 4
B-3044 HAASRODE
Tel. (016) 46.10.85
Bancaccount : 230-0045353-74 of Gen. Bank Leuven

voor Nederland : J.F. Van Dunne
Hoflaan 70
NL-3062 JJ ROTTERDAM
Tel. (010) 144802
giro : 4083817

pour la France : DAInamic FRANCE
C. Dufour
Rue Lavoisier 9
59149 DUNKERQUE
Tel. 02866 3339

Noot : Maakt U zich geen zorgen om wisselkoersen, U kan gerust 1000 Belgische Franken overschrijven via de Nederlandse giro, F. Druijff heeft het voor ons uitgeprobeerd.

NEW SOFTWARE :

BOEKHOUDPROGRAMMA :	audio2000 Bfr DCR2150 Bfr
DBASIC V2.2 + ext. update :	250 Bfr
FYSISCHE GEOGRAFIE :	audio1000 Bfr DCR1150 Bfr

SOON AVAILABLE :

COMPILATION 83-84 : We don't know yet how many tapes it will take (DCR), but as this collection will offer programs from Newsletter 16 to Newsletter 25, it will be a lot of software !

SUPER-BASE : At last a 100 % machine-language database program, using the total amount of RAM to hold up to 1000 records, depending on the field-size. Ultrafast sorting, build-in mail-merge, formatting for all kind of labels and so on...

PATROUILLER : The latest creation of P. Janin, if you ever saw PHOENIX, you know what we are talking about... guide your CAR on the surface of the moon and encounter a lot of missions to complete your race...

The first issue of 1985 will deal specially with DCE-bus and other I/O projects. If you have some projects ready, please send description to be published in Newsletter 26.

We invite you to dive into this jubilee-issue, till next year,

W. Hermans

REMARK

PROGRAMMEERTECHNIEKEN

Structuren in basicprogramma's zijn het onderwerp van deze keer. Langzaamaan begint ook basic steeds meer instructies te kennen die niet echt noodzakelijk zijn, maar die programmeren direct of later prettiger maken. Met instructies die direct het programmeren prettiger maken bedoel ik bv FOR-NEXT, WHILE-WEND, REPEAT-UNTIL. Instructies die vooral later hun vruchten afwerpen als U het programma nogeens doorneemt om er een paar kleine veranderingen in aan te brengen zijn bv PROCEDURE en FUNCTION en labels. Deze mogelijkheden zitten jammer genoeg niet alle in de standaard DAI-basic. Maar gelukkig brengt D-BASIC daar nu verandering in. Hiermee verandert de basic in een soort PASCAL. Sommige fabrikanten hebben een soortgelijke BASIC al verheven tot nieuwe taal nl COMAL.

Maar laten we bij het begin beginnen en eerst de loops of lussen bekijken. De naamgeving is duidelijk afkomstig uit de tijd dat er alleen nog maar in machinetaal of assembler werd geprogrammeerd. Het is als volgt te zien : als je het programma stap voor stap volgt met een vingertje langs de listing, je vaststelt dat er programmadelen zijn die meermalen worden uitgevoerd. Heb je dan een pen in de hand zie je vanzelf een lus (in het engels loop) ontstaan. Vaak wordt het dan een volledige warboel. Zeker als er meerdere loops in elkaar en helemaal door elkaar zitten. De structuur moet dan geheel van de programmeur komen. Maar om in minimum basic dus zelfs zonder FOR-NEXT een loop op te zetten zijn er vele methodes. Eens kwam ik een artikel tegen met de kop '24 ways to program a loop'. Wat zijn deze methodes dan en waarin verschillen ze van de andere ? De body bestaat uit een of meer instructies die een zeker aantal malen (N) uitgevoerd dienen te worden. Er zijn een paar principiele verschillen aan te wijzen : de teller die het aantal bijhoudt telt op tot N of telt af vanaf N. De teller telt tot N of tot en met N; bij dalen tot 0 of tot en met 0 De controle, of de grens reeds bereikt is, wordt vooraf gedaan of achteraf.

Een aantal voorbeelden :

```
   -A-                  -B-                  -C-                  -D-
10 T=0                  10 T=1               10 T=0               10 T=0
20 REM begin            20 REM begin         20 T=T+1             20 IF T=N GOTO 90
-- body                 -- body              30 IF T>N GOTO 90    30 REM begin
50 REM eind             50 REM eind          40 REM begin         -- body
60 T=T+1                60 T=T+1             -- body              60 REM eind
70 IF T<N GOTO 20       70 IF T<=N GOTO 20   70 REM eind          70 T=T+1
                                             80 GOTO 20           80 GOTO 20
                                             90 REM vervolg       90 REM vervolg
```

Op het eerste gezicht weinig verschil totdat we ons bedenken hoevaak de body nu wordt uitgevoerd. Mijn leerlingen krijgen ALTIJD deze structuur te leren en dan later pas bv FOR-NEXT. Ik vraag ook altijd hoeveel zij eronder durven te verwedden dat de body inderdaad N maal is uitgevoerd en niet N-1 of N+1 maal. Een programma, waarvan U niet zeker weet dat het goed werkt, is nog slechter dan een programma, dat niet werkt. Dat laatste geeft geen antwoorden maar het eerste misschien foute antwoorden. Controleer de werking vooraf met kleine waarden voor N, zodat U dat zelf eenvoudig kunt controleren . Vervang de body daartoe door bv 'PRINT "TEST "$T' en neem voor N een getal onder de tien. Nu de verschillen : Bij de versies A en B zal de loop minimaal eenmaal doorlopen worden, ook als N kleiner dan een is en bij de versies C en D zal dit niet gebeuren. Daar wordt de loop (bij niet-negatieve N tenminste) precies N maal doorlopen. In dit opzicht hebben C en D dus de voorkeur boven A en B. Maar er zijn meer regels gebruikt en ook meer GOTO's en dat maakt het programma niet overzichtelijker. Wel is nu het begin en het einde van de loop gemarkeerd met

een GOTO. Bij D kunnen we trouwens door de test op gelijkheid met N grandioos de mist ingaan als N en/of T een breukdeel bevatten. Een ander verschil tussen C en D zit in de plaats van de verhoging van T. Bij C wordt er altijd verhoogt en bij D alleen als de loop daadwerkelijk doorlopen wordt. Merk op dat de verhoging van T ook voor de body, maar na de test, geplaatst kan worden. Bij A en B is het voornaamste verschil in de waarde van T tijdens het uitvoeren van de body. Wordt de waarde van T in de body gebruikt is dat dus een criterium om te kiezen tussen A en B. Andere varianten zijn nog te bedenken: N van 11 t/m 40 of van -10 t/m 20. (Klopt dit wel ?) In plaats van T van 1 t/m 30 te laten gaan hem van 10 t/m 300 te laten gaan in stappen van tien in plaats van een. Merk op dat onze taal bij 10 t/m 300 wel duidelijk is in het er bij horen van 300 maar niet met het er bij horen van 10.

De verschillen komen neer op de volgende zaken :

1) vanaf startwaarde of van en met startwaarde
2) tot eindwaarde of tot en met eindwaarde
3) eerst uitvoeren dan controle of eerst controle en dan eventueel uitvoeren
4) verhoging voor controle of verhoging na controle
5) verhoging voor uitvoering body of er na
6) controle op gelijkheid (gevaarlijk !) of ongelijkheid
7) controle op < of <= respectievelijk > of >=
8) T is zuivere loopteller (dwz telt de doorgangen) of is reeds voorbewerkt voor gebruik in de body.

Alle combinaties hiervan leveren dus meer dan de al genoemde 24 mogelijkheden. We komen nu toe aan de FOR-NEXT instructie . Dit programmeert vanzelfsprekend veel simpeler en soms ook eleganter. We hebben geen GOTO meer nodig dus wordt het geheel overzichtelijker. MAAR als U geen antwoord weet te geven op de volgende vragen mag u hem niet gebruiken !!!! Hoe is de volgorde bij de FOR-NEXT eerst controle of eerst uitvoeren van de body ? Eerst de body en dan de verhoging of net omgekeerd ? Eerst de controle of eerst de verhoging ? Als het U lastig valt op zulke theoretische vragen te antwoorden kijk dan naar de practische vragen aan de hand van de volgende voorbeelden.

```
10 FOR I=0 TO 20:...........:NEXT    wat is de waarde van I na regel 10 ?
20 FOR I=5 TO 12:...........:NEXT    hoe vaak wordt de body uitgevoerd ?
30 FOR I=6 TO 6:............:NEXT    hoe vaak wordt de body uitgevoerd ?
40 FOR I=8 TO 5:............:NEXT    hoe vaak wordt de body uitgevoerd ?
```

Moeilijker :

```
50 FOR I=1 TO 20:.......:I=I+1:NEXT  hoe vaak wordt de body uitgevoerd ?
60 FOR I=1 TO 10:.......:I=I-1:NEXT  hoe vaak wordt de body uitgevoerd ?
```

Nog lastiger :

```
Wat is het verschil tussen FOR I%=0 TO 6 en FOR I!=0 TO 6 ?
Wat is het verschil tussen FOR I%=0 TO 6 EN FOR I%=0.0 TO 6.0 ?
Wat is het verschil tussen eindigen met NEXT en NEXT I ?
```

En tot slot de vragen waar bijna niemand correct antwoord op blijkt te kunnen geven: Hoeveel niveaus van FOR-NEXT kent de DAI ? Mag men straffeloos uit en FOR-NEXT loop springen ?

We zullen een aantal van de hierboven beschreven gevallen bespreken. Niet alle want de eerste vier kunt U vanzelfsprekend zelf intikken en dan controleren. Foei als het nodig is en driewerf foei als U het niet zeker weet en dan toch niet controleert. De problemen met regelnummers 50 en 60 zijn lastiger. De DAI bepaalt aan het begin van de loop hoeveel maal deze doorlopen moet worden. Hij

doet dit door het verschil eindwaarde min beginwaarde te delen door de grootte van de stap. Is de stapgrootte 0 krijgen we de foutmelding 'DIVISION BY ZERO'. Een interne (integer) teller houdt het aantal doorgangen van de loop bij. Vervolgens wordt de body eenmaal uitgevoerd en dan zowel de interne teller als de door de programmeur opgegeven variabele verhoogd (of verlaagd) . Veranderen we in de body dus de eigen teller (loopcounter) zal deze wel veranderen, maar het aantal malen dat de loop doorlopen wordt, wordt met de interne teller bijgehouden en zal onveranderd blijven . Het resultaat is dat bij regel 50 de body 20 maal wordt uitgevoerd en I via de oneven getallen uitkomt op 41. Bij regel 60 zal de body 10 maal uitgevoerd worden en I op 1 blijven. (steeds een er af en een er af aan het eind) Het verschil tussen gebruik van een integer teller I% of een floating point I! is alleen de snelheid van werken, vooral te merken met kleine body's. Opgelet trouwens bij het werken met integertellers in dit verband : FOR I%=20 TO 30 STEP .5:.....:NEXT geeft de foutmelding 'DIVISION BY ZERO', evenals de berekening van de stapgroottevariabele, die een breukdeel oplevert. Bij integervariabelen worden die dan afgerond; wordt het 0 krjgen we een foutmelding, maar in de andere gevallen een onjuist werkend programma . Ik heb geen verschil kunnen vinden in het gebruik van NEXT en NEXT I.

Het aantal niveaus dat de DAI aankan is wisselend en dit veroorzaakt dan ook een aantal vreemde zaken. Om te onthouden waar de FOR-NEXT body begint zet de DAI een aantal gegevens op een speciale plaats in het geheugen (de stack) Deze stack wordt echter ook gebruikt door de ROM-routines en bv de interrupt routines. Tik ter testing het volgende programma in :

```
10  FOR A=0 TO 2
20  FOR B=0 TO 2                Niet erg dat er geen NEXT'n staan
::  ::: ::: :: :
::  ::: ::: :: :
140 FOR N=0 TO 2                het programma moet ontsporen voor de test
150 FOR O=0 TO 2
160 FOR P=0 TO 2
```

U zult zien dat U na RUN bijna altijd de melding 'STACK OVERFLOW IN LINE 160'. Voegt U aan het programma toe 155 PRINT "ufghtf" komt U al bij 155 op stack overflow. Vaak intikken van RUN levert ook soms een stack overflow op regels met lagere nummers. Bij mij 140 als minimum. Dit verschijnsel is te verklaren door het gebruik van de stack door ROM-routines en de interrupt handling. Bv zal de 20ms interrupt (en de cursor-'interrupt' als onderdeel daarvan) niet altijd op hezelfde moment in het programma komen. Het maximum is dus bij een normale stack 15 niveaus, maar het blijft gevaarlijk zo diep te gaan , daar op andere plaatsen dit problemen met stackruimte kan geven. Heeft het programma zelf al subroutines (GOSUB) levert het helemaal vrijwel zeker problemen op. Een aardige tip voor luie (= slimme ?) programmeurs : de teller mag best een arrayvariabele zijn. dus kan de gewenste informatie als volgt gevonden worden.

```
10 DIM T(20)
20 FOR T(I)=0 TO 2:I=I+1:GOTO 20
```

RUN geeft zoals verwacht 'STACK OVERFLOW IN LINE 20' en PRINT I levert 14 op.

En dan nu het grote misverstand. Velen denken dat ze bij de DAI ongestraft uit een FOR-NEXT loop mogen springen. D I T I S N I E T W A A R ! ! ! ! ! De goede programmeur wist reeds dat het principieel nooit kan . Een zeer grote stack kan het probleem enige tijd uitstellen maar het komt gegarandeerd. Het laatste voorbeeld liet het eigenlijk al zien. We springen daar uit de FOR-NEXT en we krijgen inderdaad een stack overflow. Maar de ontwerpers van de DAI hebben iets heel slims bedacht. Als er een FOR-NEXT loop opgestart wordt en er is al eerder een FOR-NEXT loop gestart met dezelfde teller dan wordt die teller

opnieuw gebruikt. De stack wordt daarmee dus niet extra belast en zo is het in veel gevallen dus toch mogelijk geworden om uit een FOR-NEXT te springen . Ter controle kunt U even intikken '10 FOR I=0 TO 2:GOTO 10'. U krijgt bij RUN geen stack overflow. De consequenties hiervan zijn, zoals hiervoor geschetst, dat U slechts zelden stack overflow zult krijgen door springen uit een FOR-NEXT loop Alleen die gevallen waar steeds uit FOR-NEXT loops wordt gesprongen en al die loops een andere loopcounter hebben, zal het bij een groter aantal fout gaan . Conclusie : Het is aan te bevelen steeds dezelfde variabelenamen te gebruiken als loopcounter. Voor de duidelijkheid van het programma is het niet aan te raden, maar het kan zonder problemen :

```
100 FOR H=1 TO 1000:H=GETC:IF H=0 THEN NEXT        Binnen 1000 keer reageren
110 PRINT "KLAAR"                                   anders gaan we verder.
```

U begrijpt dat met maar een soort loop nl de FOR-NEXT, terwijl er zoveel mogelijk zijn er naar nieuwe constructies werd gezocht. Ik behandel de WHILE-WEND en de REPEAT-UNTIL. Een belangrijk verschil met de FOR-NEXT is het feit dat nu bij aanvang nog niet bekend is hoe vaak de loop doorlopen moet worden. Dit impliceert dus tevens, dat we, bij een foutieve programmering, in een eindeloze loop kunnen komen te zitten.

Om te beginnen de REPEAT-UNTIL. De syntax voor deze instructie is voor D-BASIC als volgt : REPEAT [statement(s)] : UNTIL <logical expression> De dubbele punt voor UNTIL is noodzakelijk, maar zoals aangegeven met de rechte '[ ]' haken de statements niet. Net zoals bij de FOR-NEXT, worden de statements minstens eenmaal uitgevoerd. Enkele voorbeelden voor het gebruik :

```
10 REPEAT H=GETC:UNTIL H<>0                          wachten op toetsaanslag
20 REPEAT T=T+1:UNTIL GETC<>0       controleer de reactietijd tot toetsaanslag
30 REPEAT DOT X,Y 23:Y=Y+1:UNTIL Y>W     maak langzaam een kolom van hoogte W
40 REPEAT GOSUB 2000:UNTIL T=0      doe de subroutine op 2000 tot dat T nul is
```

Over WHILE-WEND is iets meer te vertellen. De syntax ervoor is in wezen niet correct en ook nog eens verschillend bij D-BASIC en TURTLE-BASIC . Let dus op het verschil :

*** D-BASIC : WHILE <logical expression> DO [statement(s)] : WEND

De 'DO' is hier in wezen overbodig. Ook zonder dat woord is het te begrijpen wat er gedaan moet worden. De body (= [statement(s)]) wordt uitgevoerd zolang aan de voorwaarde (= <logical expression>) wordt voldaan. Is de voorwaarde al bij de start onjuist wordt de loop NIET doorlopen. Voorbeelden voor gebruik :

```
10 WHILE GETC=0 DO WEND        Wacht op een toetsaanslag
*** Volgens de syntax van D-BASIC zou er tussen DO en WEND een : moeten staan.
20 WHILE T>0 DO T=T-1:WEND     Tel af tot nul
```

TURTLE-BASIC WHILE <logical expression> DONOT <linenumber>:[statement(s)]:WEND

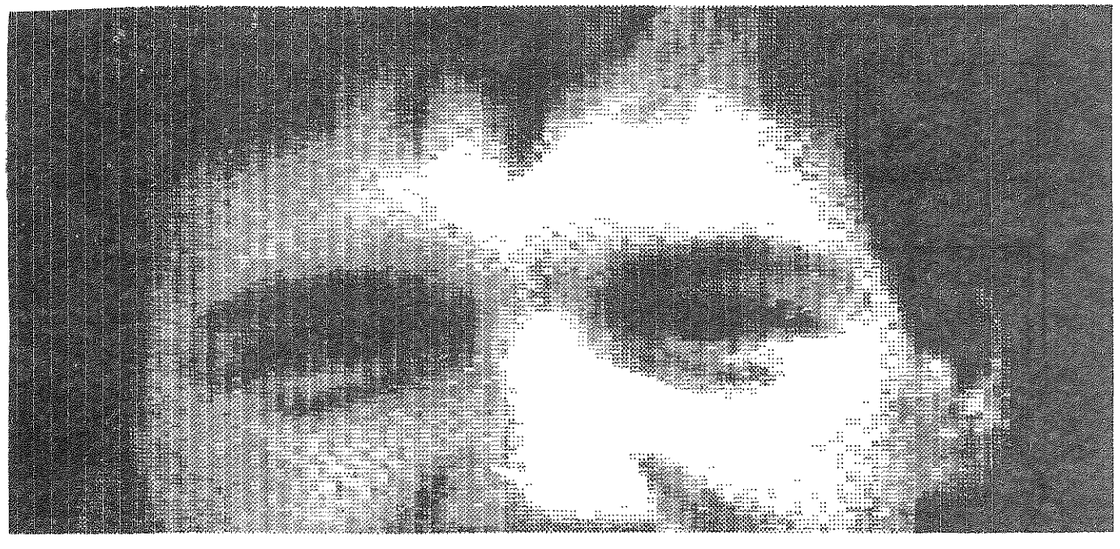
De WEND is hier in wezen overbodig, daar bij logisch programmeren na de WEND juist het regelnummer volgt dat achter de 'DONOT' staat. Ik vind het niet een bepaald mooie instructie in TURTLE-BASIC. Zolang aan de voorwaarde wordt voldaan zal niet naar de regel achter de 'DONOT' gesprongen worden, maar de statements tot de WEND uitgevoerd worden. Een nadeel van de WHILE-WEND in TURTLE-BASIC is het feit dat ze niet genest kunnen worden.

Het volgend nummer kom ik hier nog op terug. Evenals de opmerking, dat er geen verschil is tussen NEXT en NEXT I.

Frank H. Druijff

BANKOVERSCHRIJVINGEN

```
10   REM *** BANKOVERSCHRIJVINGEN ***
20   POKE #131,1:POKE #75,#FF:PRINT CHR$(12):COLORT 14 0 14 14
30   A%=1.0:T%=0.0
40   N%=0.0:T%=T%+B%
45   REM ------------------------------ SCHERMOPBOUW
50   PRINT :PRINT :PRINT
100  PRINT TAB(15);"*** INVOER VAN DE GEGEVENS ***"
110  PRINT :PRINT
120  PRINT " 1. DATUM (DDMMJJ)            :"
130  PRINT " 2. REK.NR van de BEGUNSTIGDE :"
140  PRINT " 3. BEDRAG                    :"
150  PRINT " 4. NAAM van de BEGUNSTIGDE   :"
160  PRINT " 5. STRAAT + HUISNUMMER       :"
170  PRINT " 6. POSTCODE + GEMEENTE       :"
180  PRINT " 7. MEDEDELING                :"
190  REM ------------------------------ GEGEVENSINVOER
200  CURSOR 31,16:INPUT DA$:DA$=LEFT$(DA$,2.0)+"."+MID$(DA$,2,2)+"."+RIGHT$(DA$,2):CURSOR 31,16:PRINT DA$;"   "
205  IF X%=1.0 GOTO 410
210  CURSOR 31,15:INPUT NB$
215  REM * PRINTEN van het REK.NR in STANDAARDVORM en CONTROLE op de CHECK-DIGIT
220  CURSOR 31,15:PRINT LEFT$(NB$,3);"-";MID$(NB$,3,7);"-";RIGHT$(NB$,2);" "
230  IF VAL(RIGHT$(NB$,2))=((VAL(LEFT$(NB$,6)) MOD 97)*10000+VAL(MID$(NB$,6,4))) MOD 97 GOTO 280
240  CURSOR 46,15:PRINT "REK.NR ONJUIST"
250  FOR Z%=#BBBF-50.0*2.0-3.0 TO Z%-29.0 STEP -2.0:POKE Z%,#FF:NEXT
260  COLORT 14 0 15 0
270  GOTO 210
280  COLORT 14 0 14 14
285  IF X%=1 GOTO 410
290  CURSOR 31,14:INPUT B$
300  B%=VAL(B$):L%=LEN(B$)
305  REM * PRINTEN van het BEDRAG en PLAATSEN van het PUNTTEKEM
310  IF L%>3.0 AND L%<7.0 GOTO 330
320  IF L%>6.0 GOTO 340
325  GOTO 360
330  B$=LEFT$(B$,L%-3)+"."+RIGHT$(B$,3):GOTO 360
340  LL%=LEN(LEFT$(B$,L%-6))
350  B$=LEFT$(B$,L%-6)+"."+MID$(B$,LL%,3)+"."+RIGHT$(B$,3)
360  CURSOR 31,14:PRINT B$;"   "
365  IF X%=1.0 GOTO 410
370  CURSOR 31,13:INPUT NA$:CURSOR 31,13:PRINT NA$;"   "
375  IF X%=1.0 GOTO 410
380  CURSOR 31,12:INPUT AD$:CURSOR 31,12:PRINT AD$;"   "
385  IF X%=1.0 GOTO 410
390  CURSOR 31,11:INPUT WO$:CURSOR 31,11:PRINT WO$;"   "
395  IF X%=1.0 GOTO 410
400  CURSOR 31,10:INPUT MED$:CURSOR 31,10:PRINT MED$;"   "
405  IF X%=1.0 GOTO 410
410  X%=0.0
420  PRINT
495  REM ------------------------------ CORRECTIEROUTINE
500  CURSOR 0,8:INPUT "CORRECTIES NODIG (J/N) ";COR$:PRINT
510  IF COR$="J" GOTO 530
520  IF COR$<>"N" AND COR$<>"J" GOTO 500
525  GOTO 600
530  INPUT "TIK GEWENST NUMMER IN ";N%:PRINT
540  IF N%<1 OR N%>7 GOTO 530
550  X%=1.0
560  ON N% GOTO 200,210,290,370,380,390,400
600  CURSOR 0,2:PRINT "ZET DE PRINTER AAN"
610  PRINT "DRUK OP DE SPATIEBALK OM VERDER TE GAAN"
620  IF GET$<>32.0 GOTO 620
625  REM ------------------------------ PRINTROUTINE
630  PRINT CHR$(12):PRINT :PRINT :PRINT
640  PRINT "============================================================":PRINT
650  POKE #131,0
660  PRINT TAB(2);DA$:PRINT :PRINT
670  PRINT TAB(24);LEFT$(NB$,3);"-";MID$(NB$,3,7);"-";RIGHT$(NB$,2);TAB(42);B$:PRINT :PRINT
680  PRINT TAB(28);NA$:PRINT
690  PRINT TAB(28);AD$:PRINT
700  PRINT TAB(28);WO$:PRINT
710  PRINT MED$
720  POKE #131,1
725  PRINT "============================================================":POKE #131,0
730  FOR P%=0 TO 10:PRINT :NEXT
740  POKE #131,1
760  CURSOR 0,0:INPUT "NOG EEN OVERSCHRIJVING (J/N) ";NOG$
770  IF NOG$="N" GOTO 800
780  IF NOG$<>"N" AND NOG$<>"J" GOTO 760
790  PRINT CHR$(12):A%=A%+1:GOTO 40
800  IF A%=1 GOTO 890
810  PRINT :PRINT :PRINT
820  PRINT "============================================================":PRINT
830  POKE #131,0
840  PRINT TAB(2);DA$:PRINT :PRINT
850  PRINT TAB(24);"***-********-***";TAB(42);T%+B%:PRINT :PRINT
860  PRINT TAB(28);"GEZAMENLIJKE OVERSCHRIJVING":PRINT :PRINT :PRINT :PRINT :PRINT
870  PRINT A%;" BIJLAGEN"
875  POKE #131,1:PRINT "============================================================":POKE #131,0
880  PRINT CHR$(12)
890  POKE #131,1:END
```



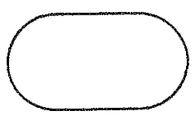
NOP	00000000	- - - - -	1	4

No operation
Deze opdracht heeft geen enkel resultaat tenzij het laten voorbijgaan van 4 klokperiodes. Deze opdracht kan bijvoorbeeld gebruikt worden om een overbodige instructie te vervangen zonder dat daarbij het hele programma dient herschreven te worden. De NOP instructie laat eveneens toe in een programma plaatsen te voorzien waar achteraf een instructie kan toegevoegd worden.

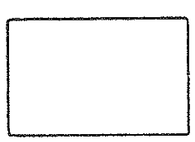
4.3. Opbouw van een programma

Als we een programma wensen op te bouwen, gaan hieraan enkele belangrijke stadia vooraf.
Het probleem of de toepassing waarvoor een programma dient geschreven te worden, dient vooraf in al zijn facetten grondig bestudeerd en geanalyseerd te worden.
Als resultaat van deze analyse wordt een organigram opgebouwd, waarin aan de hand van tekensymbolen de verschillende fasen gesymboliseerd worden.
We merken op dat een programma zoveel mogelijk in kleine modules dient geschreven te worden. Onder een module verstaat men een programmagedeelte dat een logisch geheel vormt en zelfstandig bepaalde akties uitvoert.
Het voordeel van deze modules is dat ze afzonderlijk kunnen getest worden op foutloze werking. Ze kunnen tevens door andere programma onderdelen opgeroepn worden als subroutines.
Men dient een programma steeds te voorzien van de nodige commentaar (documenteren), niet alleen om achteraf het programma vlot te kunnen lezen maar tevens om aan derden toe te laten uw programma te begrijpen. Men noemt dit een programma documenteren.

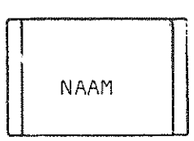
4.4. Symbolen gebruikt in organigrammen



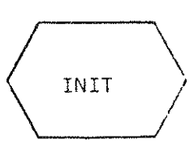
Dit symbool duidt het begin van een programma (START) of het einde van een programma (END) aan, alnaargelang de ingeschreven tekst.



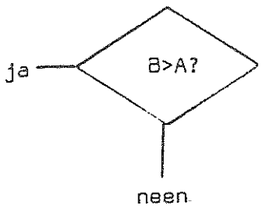
In dit symbool wordt een programma onderdeel aangegeven dat kan afgehandeld worden door één of meerdere instructies.



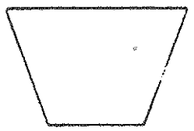
Dit symbool geeft het oproepen aan van de subroutine NAAM.



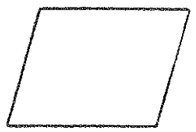
Subroutines die opgeroepen worden om bouwstenen te initialiseren worden ook voorgesteld door dit symbool.



De in dit symbool geschreven vergelijking of tekst wordt getest. Het programma wordt verdergezet in de richting van de 'ja-uitgang' of de 'neen-uitgang' alnaargelang het resultaat van de test.



Symbool voor manuele invoer (bijvoorbeeld van een klavier).



Symbool voor machinale in- en uitvoer (bijvoorbeeld naar een regeldrukker).

Als het organigram meerdere pagina's lang is, geeft dit symbool aan dat de lijnen met hetzelfde nummer dienen doorverbonden te worden.

4.5. Programmavoorbeelden voor de 8080/8085

Voorbeeld 1 : optelling van twee 8 bits getallen

Veronderstel dat in RAM op adres A12D het hexadecimaal getal B6 staat en op het adres ØEØ1 het getal 1A.
We wensen deze getallen op te tellen en het resultaat terug te schrijven op het adres ØEØ1.
Het programma bevindt zich bijvoorbeeld in ROM vanaf het adres ØØ21 en gebruik makend van de mnemonics, luidt het :

```
LDA    A12Ø    H
LXI    H,EØ1   H
ADD    M
MOV    M,A
```

LDA ØA12D H	laadt de accumulator met de inhoud van de geheugencel A12D. In hexadecimale notatie moeten getallen steeds beginnen met een cijfer (vandaar de Ø voor A12D en eindigen met H. Zonder H wordt het een decimaal getal, met een B wordt het een binair getal.

LXI H,ØEØ1 H — laadt het registerpaar onmiddellijk met het adres ØEØ1.

ADD M — telt de inhoud van de accumulator op bij de inhoud van de geheugencel die geadresseerd wordt door het HL registerpaar (ØEØ1). Het resultaat komt in de accumulator.

MOV M,A — Brengt de inhoud van de accumulator naar het adres aangegeven door het HL register (ØEØ1).

Om duidelijk te zien wat er gebeurt in functie van de tijd, hebben we onderstaande tabel voor elke instructie de inhoud gegeven van de registers en geheugencellen die bij dit programma betrokken zijn.

ROM			RAM			
adr	Mnemonic	Hexacode	HL	ACCU	A12D	ØEØ1
ØØ21	LDA ØA12D H	3A	?(1)	?	B6	1A
ØØ22		2D	?	?	B6	1A
ØØ23		A1	?	B6	B6	1A
ØØ24	LXI H,ØEØ1 H	21	?	B6	B6	1A
ØØ25		Ø1	?	B6	B6	1A
ØØ26		ØE	ØEØ1	B6	B6	1A
ØØ27	ADD M	86	ØEØ1	DØ(2)	B6	1A
ØØ28	MOV M,A	77	ØEØ1	DØ	B6	DØ

(1) ? wil zeggen dat de inhoud niet nader bekend is en een gevolg is van een vorig programma of van een startprocedure van de microprocessor.

(2) DØ is de hexadecimale som van B6 en 1A.

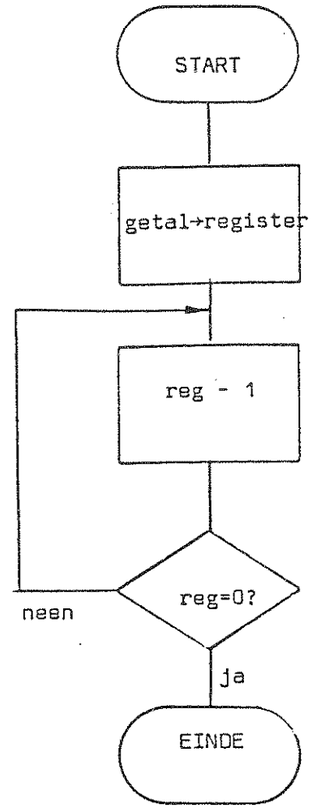
Het programma kan niet onder de vorm van mnemonics in het programmatiegeheugen gebracht worden, maar wel onder de vorm van de hexadecimale code van kolom 3.

Stel zelf het programma op voor het optellen van twee 16 bits getallen
a) met 8 bits optelinstructies (ADA, ADC)
b) met de 16 bits optelinstructie DAD.

Voorbeeld 2 : programmatie van een korte vertragingslus

In een register wordt een getal ingeschreven. De registerinhoud wordt met één verminderd en er wordt getest of de inhoud nul is. Indien de inhoud niet nul is, wordt de registerinhoud opnieuw met één verminderd en terug getest.

ORGANIGRAM :



PROGRAMMA :

```
        MVI     B,FF     H
LUS:    DCR B
        JNZ LUS
```

LUS staat in dit programma als symbolisch adres LABEL genoemd. (LUS is het adres waar de instructie DCR B is ondergebracht.
Als we over een assembler vertaalprogramma beschikken, kunnen we het programma rechtstreeks invoeren in mnemonics, waarbij gebruik gemaakt kan worden van labels, dit zijn symbolische namen (adressen) waaraan door de assembler een bepaalde waarde wordt toegekend. Deze mag tot zes alfanumerische karakters bevatten met als eerste een letter, een vraagteken of een @. De label BEGIN wordt gevolgd door een dubbelpunt. Labels mogen geen mnemonics zijn die gebruikt worden als opcode. Andere assembler directieven zijn ORG adres (PC=adres), END (afsluiting) en EQU uitdrukking of adres waarbij de waarde toegekend wordt aan een label.
Voorgaand programma wordt dus :

```
BEGIN:  ORG     1000     H
TEL     EQU     FF       H
        MVI     B,TEL
LUS:    DCR     B
        JNZ     LUS
        END
```

Als we echter niet over een assembler vertaalprogramma beschikken, moeten we het programma onder hexadecimale vorm manueel invoeren.
Veronderstel dat we dit programma inschrijven vanaf het adres 1ØØØ H in RAM, dan wordt het symbolisch adres overeenkomstig met de label LUS gelijk aan 1ØØ2 H.

```
        MVI     B,FF     H
1002    DCR     B
        JNZ     1002
```

Omzetting van de mnemonics in hexadecimale code geeft tenslotte :

```
1000   04 FF
1002   05
1003   02 02 10
```

Let er op dat bij het invoeren van de adressen de minst beduidende byte het eerste komt en slechts daarna de meest beduidende byte.
Wensen we de vertragingstijd van deze lus te kennen, dan dienen we eerst het aantal klokperiodes per instructie te kennen. Voor de processor 8080 wordt dit :

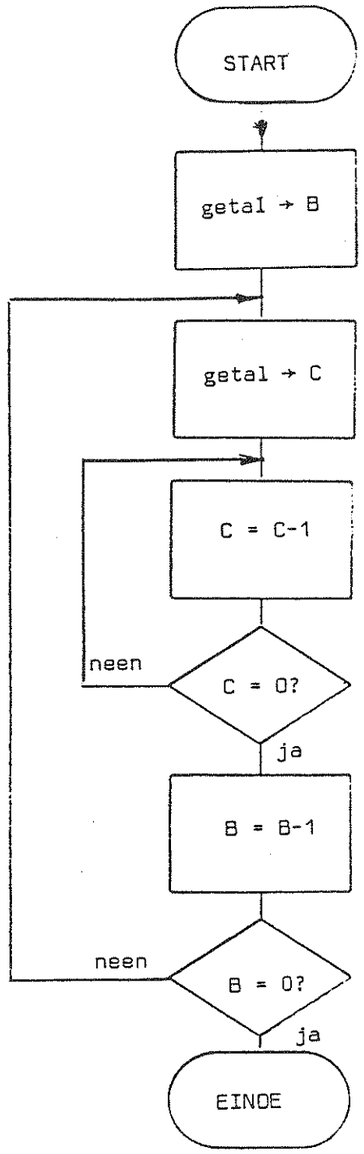
```
        MVI     B,FF     H        7 klokperiodes
LUS:    DCR     B                 5 klokperiodes      wordt 255 (FF)
        JNZ     LUS              10 klokperiodes      maal uitgevoerd
```

De totale vertragingstijd is : 7 + (5+10) 255 = 3822 klokperiodes.
Indien we werken met een klok van 2MHz (periode 500ns) dan is de vertragingstijd 3832 x 0,5 = 1916µs.
Indien kleine tijdsaanpassingen gewenst zijn, kunnen we enkele instructies inlassen die geen invloed hebben op de tellerwerking (bv. NOP, MOV A,A ; ORA A,A ; XCHG).
Bij het inlassen van 1 NOP na DCRB wordt de vertragingstijd 2,4ms.

Voorbeeld 3 : programmatie van een dubbele vertragingslus

Het gebruik van een dubbele vertragingslus in genestelde lussen, kan aanzienlijk langere vertragingstijden geven.

ORGANIGRAM :



PROGRAMMA :

P102

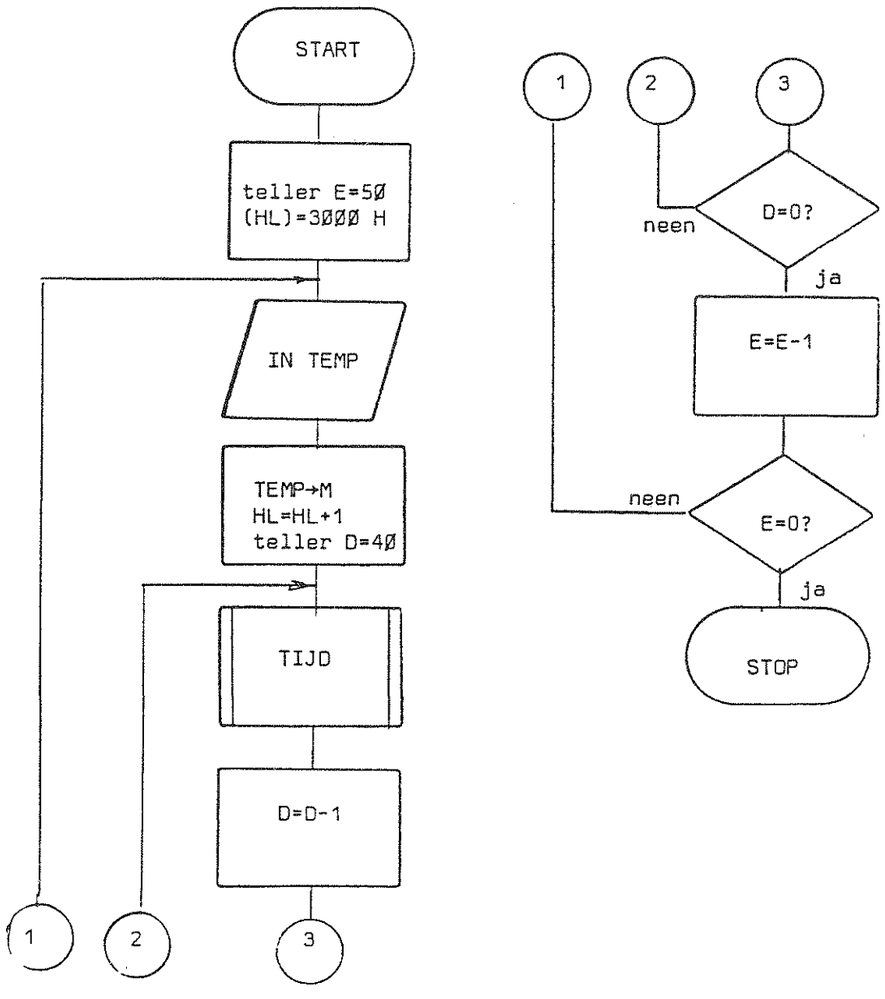
			Klokperiodes	Aantal malen doorlopen
	MVI	B,FF	7	1x
LUS2	MVI	C,FF	7	255x
LUS1	DCR	C	5	255x255
	JNZ	LUS1	10	
	DCR	B	5	255x
	JNZ	LUS2	10	

Totale vertragingstijd :
$7+255(10+5+7)+255^2(5+10)$ = 980.992 klokperiodes of 490496µs = 0,49sec.
Indien deze vertragingslussen als subroutine opgeroepen worden, vergroot de tijd met de duur van een CALL instructie (17T) en RET (10T).
Bij het gebruik van de processor 8085 moeten we rekening houden met enkele tijdsaanpassingen. Zo duurt een DCR 4T (i.p.v. 5T) JNZ 10T bij een sprong en 7T bij een voortzetting van het programma.
Het wordt aan de lezer overgelaten hiermee de nieuwe tijdsvertragingen te berekenen.

Voorbeeld 4 : Het inschrijven van meetresultaten

Aan kanaal 2 van een microprocessorsysteem is een thermometer aangesloten die de gemeten temperatuur geeft onder de vorm van een 8-bits code.
We wensen om de 20 seconden een meting te verrichten en in te schrijven in opeenvolgende geheugenplaatsen en dit vanaf geheugenplaats 3ØØØ.
Na 50 metingen dient het programma te worden stopgezet. Een vertragingsroutine van 0,5 seconden kan opgeroepen worden onder de naam TIJD.

ORGANIGRAM :



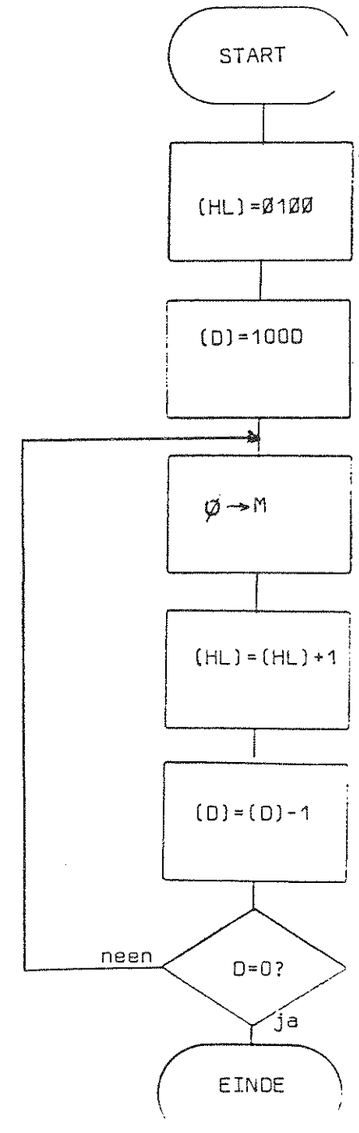
PROGRAMMA :

```
        MVI     E,50            ; teller laden met 50
        LXI     H,3000          ; adres register laden met 3000
LUS3    IN      2               ; thermometer lezen
        MOV     M,A             ; lezing inschrijven in geheugen
        INX     H               ; adresregisters incrementeren
        MVI     D,40
LUS4    CALL    TIJD            ; vertragingslus 20sec
        DCR     D
        JNZ     LUS4
        DCR     E               ; teller decrementeren
        JNZ     LUS3            ; testen of teller op nul staat
        HLT
```

Voorbeeld 5 : op 0 stellen van geheugen

We wensen 100 opeenvolgende geheugenplaatsen vanaf adres Ø1ØØ H op nul te zetten.

ORGANIGRAM :



PROGRAMMA :

```
        LXI     H,100           ; adres register laden met Ø1ØØ
        MVI     D,100           ; teller laden met 1ØØ
WISSEN  MVI     M,ØØ            ; geheugencel op nul zetten
        INX     H               ; adres register incrementeren
        DCR     D               ; teller decrementeren
        JNZ     WISSEN          ; teller testen op nul
```

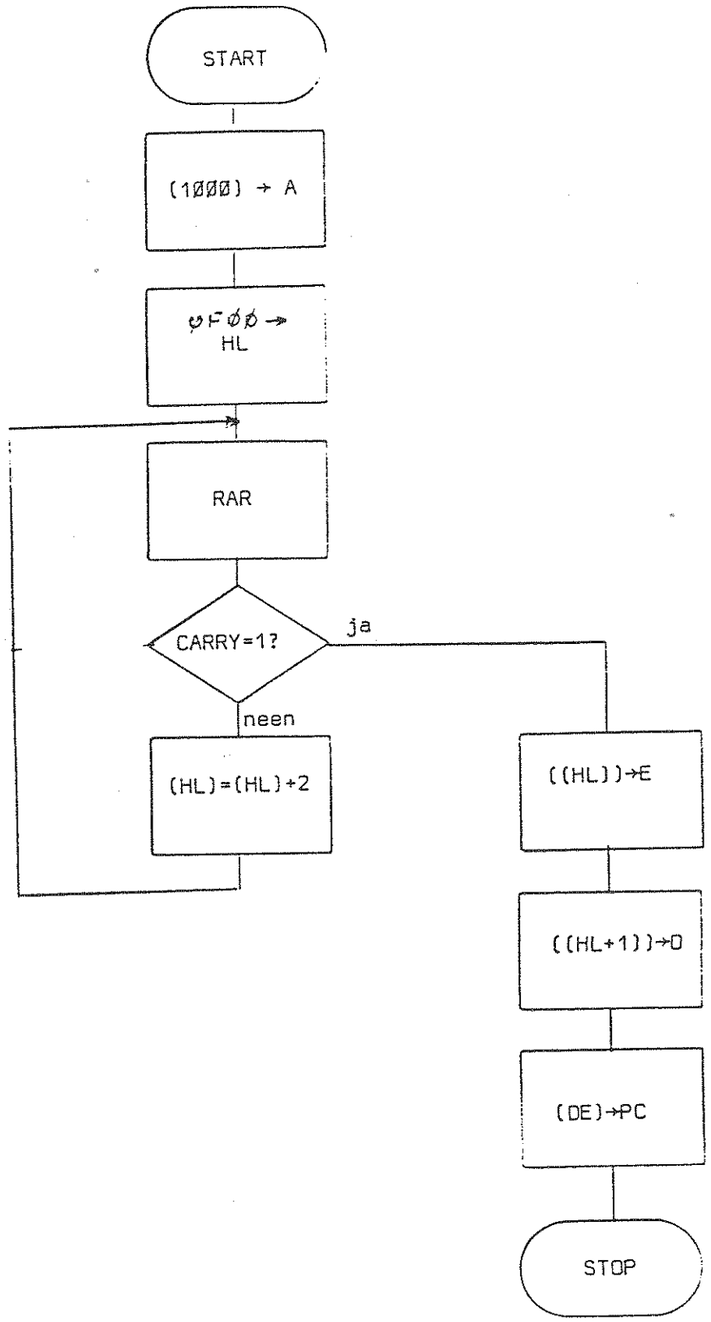
Voorbeeld 6 : programmavertakking

In de geheugenplaats 1ØØØ bevindt zich een byte waarvan slechts één bit gelijk is aan 1.
Naargelang de plaats van deze bit in de byte dient het programma verder gezet te worden op een adres dat kan gelezen worden in de geheugentabel vanaf adres ØFØØ (voor bit 0) tot ØFØE (voor bit 7).

ORGANIGRAM : (zie pag. 106)

PROGRAMMA :

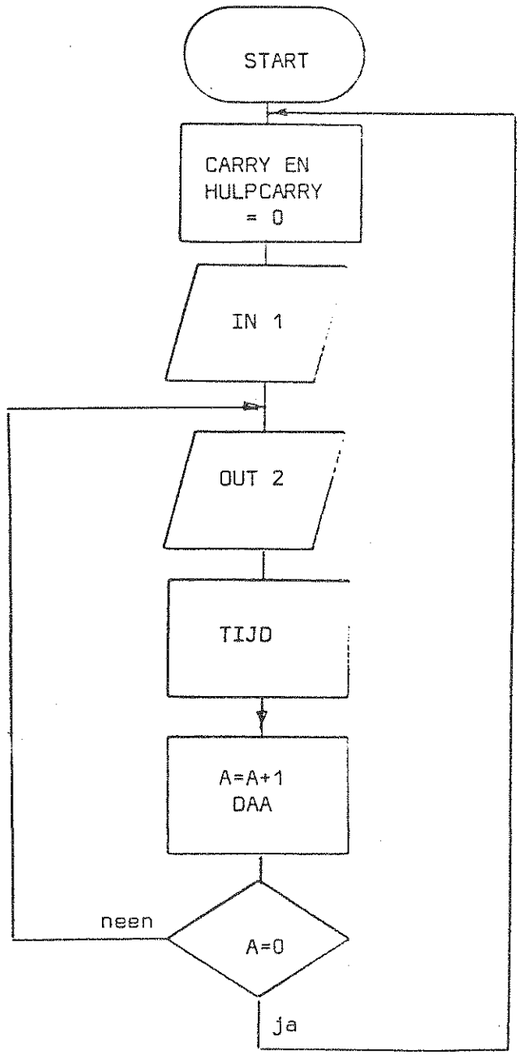
```
0001            LDA     1000        H ; byte in accumulator
0002            LXI     H,0F00      H ; adres register laden met ØFØØ
0003 LUS:       RAR                   ; verschuiven naar rechts
0004            JC      DOEL          ; carry testen
0005            INX     H             ; adres register 2x ophogen
0006            INX     H
0007            JMP     LUS           ; volgende bit testen
0008 DOEL:      MOV     E,M           ; LSB van adres in E
0009            INX     H
0010            MOV     D,M           ; MSB van adres in D
0011            XCHG                  ; inhoud van DE naar HL
0012            PCHL                  ; programmateller laden met
                                        adres waar het programma
                                        verder gezet dient te worden.
```



Voorbeeld 7 : teller

Aan kanaal 1 van een microprocessorsysteem wordt een getal (N) van twee decimalen in BCD aangeboden.
We wensen aan kanaal twee een teller te verkrijgen, die decimaal telt vanaf het getal N tot 99. Elk getal dient gedurende 0,5 sec. aanwezig te blijven. We beschikken daartoe over een vertragingsroutine van 0,5 sec. die kan opgeroepen worden onder de naam TIJD.

ORGANIGRAM :



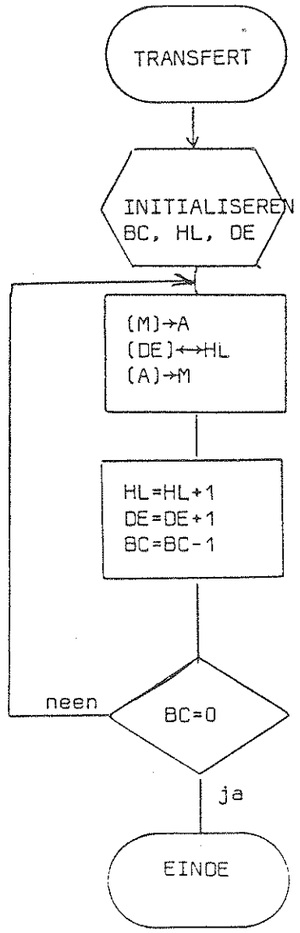
PROGRAMMA :

```
        XRA     A           ; carry en hulpcarry op nul
BEGIN   IN      1           ; getal lezen
UIT     OUT     2           ; getal afdrukken
        CALL    TIJD        ; 0,5 sec. wachten
        INR     A           ; accumulator ophogen
        DAA                 ; accu. omzetten in BCD
        CPI     0           ; testen overgang 99/0
        JNZ     UIT         ; indien ≠ 0 verder tellen
        JMP     BEGIN       ; indien = 0 herbeginnen
```

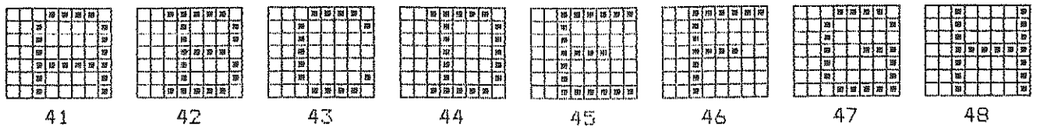
Voorbeeld 8 : verplaatsing van een blok informatie in geheugen

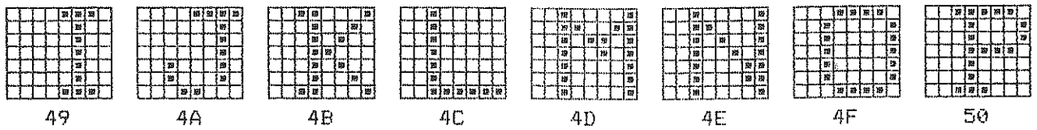
Verondersteld dat we een blok informatie, waarvan de lengte zich in het registerpaar BC bevindt, wensen te verplaatsen van het beginadres dat zich in HL bevindt naar het beginadres dat zich in DE bevindt.

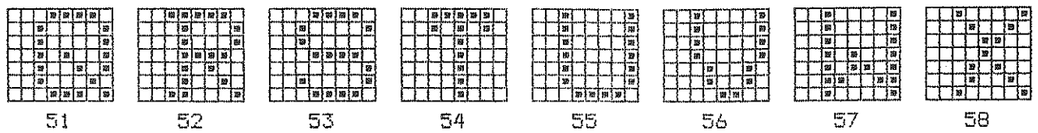
ORGANIGRAM :

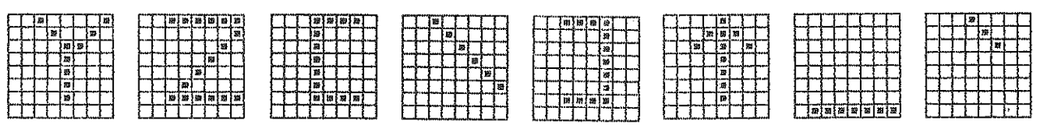


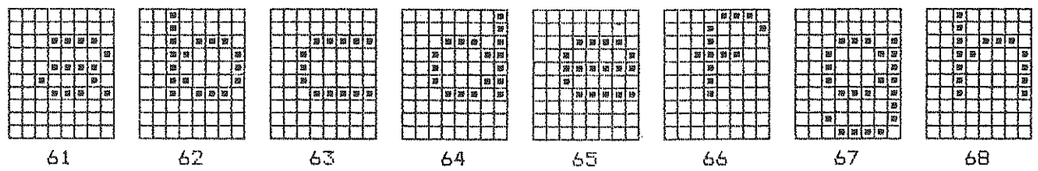
DAI CHARACTER SET

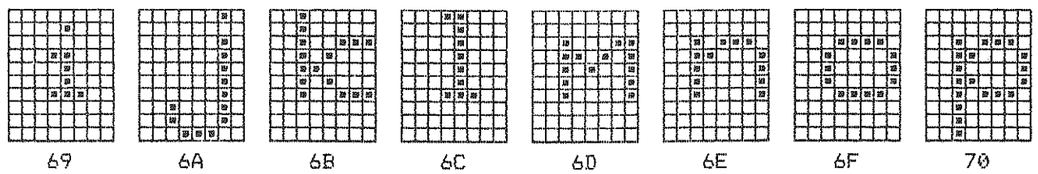


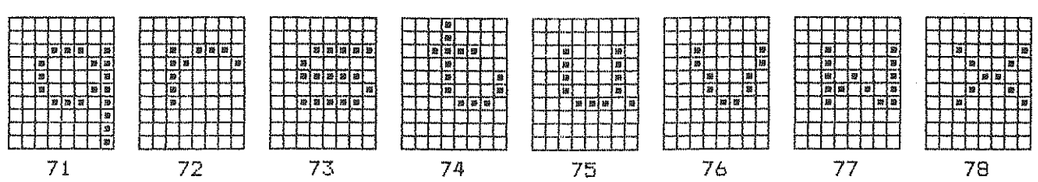


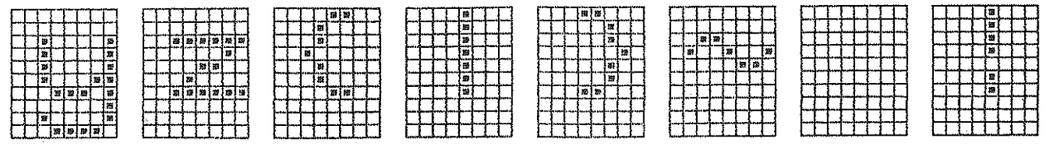


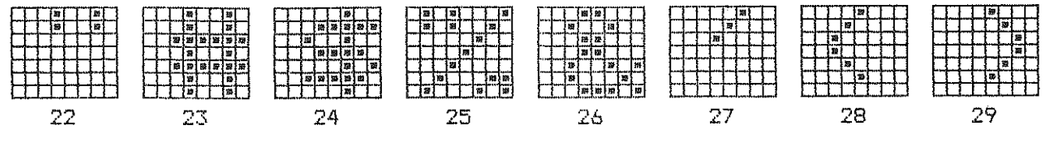


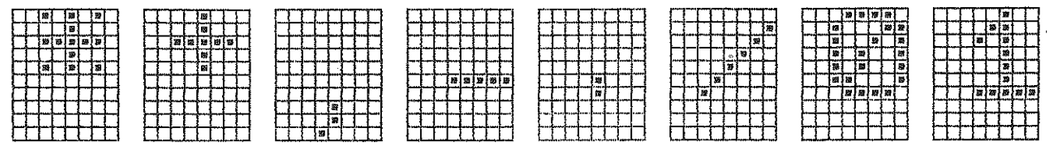


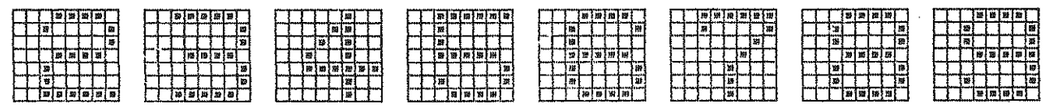


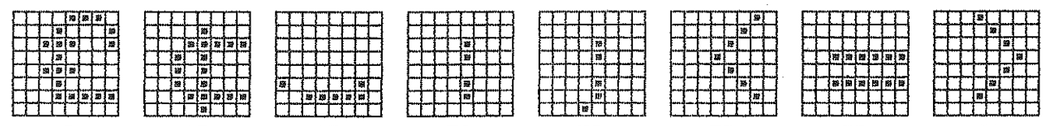


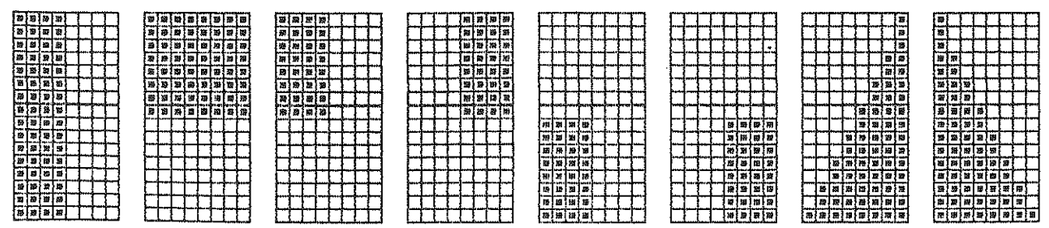


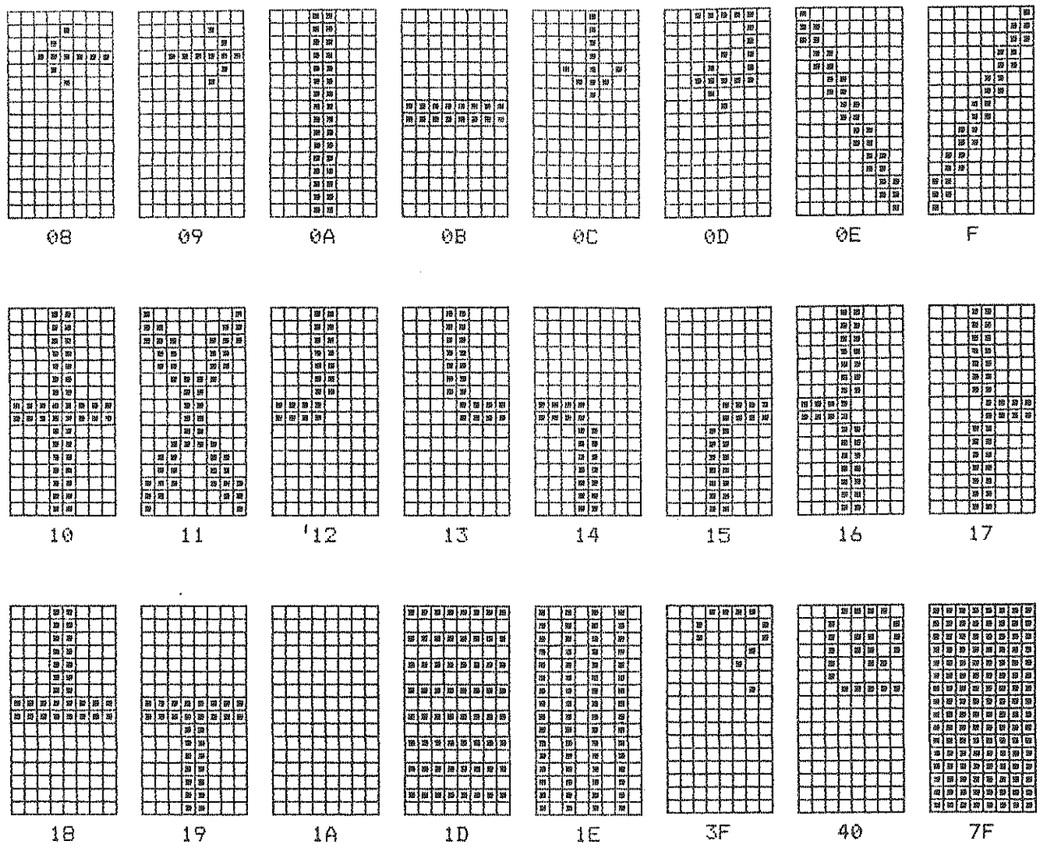












- JEU DE CARACTERERES DU DAI -

Les differentes matrices ci-dessus représente le jeu complet des caractères du DAI, soit 128 au total. La taille des matrices à été adaptée à celle des caractères.Ainsi les majuscules sont dans une matrice de 8 points verticalement, alors que les caractères semi-graphiques sont inscrit sur 16 points . Dans tous les cas la première ligne de la matrice est la meme que celle du caractère correspondant . La lecture des dessins des caractères ayant été réalisée manuellement, si vous decouvrez une erreur faites le moi savoir !

Cédric Dufour

DAIminiCALC

AVANT DE CHARGER LE PROGRAMME TAPER 'IMPINT'

Ce programme permet le calcul de formules utilisant :

1. Le symbole # pour introduire une formule
   des lettres A-Z ou les symboles] [< >
   qui désignent une ligne du tableau
   des opérateurs +,-,* ou /
   des constantes entières ou décimales
   des parenthèses ()
   La touche return pour terminer la formule
   Exemple : #(A*100+B)/(C*1.89+D)
   Le résultat est attribué à la ligne ou la formule est écrite
   La formule est valable pour toute une ligne(1-14).
   Après analyse chaque formule est memorisée et transformée en ligne de programme(dank-u Mr Looije)
   C'est la forme algébrique usuelle (priorité de * et /)

2. Les données sont introduites dans le tableau en déplaçant la fenêtre colorée avec les flèches
   et taper espace + la valeur de la donnée + return.
   Les données numeriques ont priorité sur les alphanumériques.
   Une donnée numérique peut être entrée comme alphanumérique en lui incorporant un symbole par ex /: 12/ 3/82
   Une donnée <>0 peut être recopiée dans la fenêtre voisine:
   il faut employer curseur + shift
   La rapidité du calcul dépend du nombre de formules
   La rapidité de l'affichage dépend des dimensions des tableaux

3. En plus du mode introduction des données et formules il existe des commandes (tapez /) :
   /L pour Load un écran from DCR
   le titre apparait
   N=skip au suivant
   space=load
   /S pour save un écran sur DCR avec titre obligatoire
   /C pour effectuer les calculs
   /T pour recopier une colonne dans une autre
   /D pour recopier une ligne dans une autre
   /P pour imprimer le tableau avec le titre du save
   /R ou F pour revenir en mode entrée des données
   /I pour insérer une ligne avec décalage du tableau vers le bas
   les formules sont aussi décalées
   /Z pour annuler une rangée
   /! position supérieure gauche de l'écran
   /= position supérieure droite
   /z position inférieure gauche
   /? position inférieure droite par rapport au tableau 30x14

4. Pour changer un symbole dans une formule (chiffre ou lettre) on se place sur la ligne ou se trouve la formule à introduire
   Exemple: en ligne M se trouve la formule #(A*G)/1.383
   on veut que la formule de la ligne T devienne: #(A*K)/1.282
   on place la fenêtre sur la ligne T et on tape #T:GK:32

5. En cas d'arrêt du programme avec erreur taper RUN 100 et ensuite /= pour récupérer le tableau et poursuivre le programme.

6. Il existe une version disque (KEN-DOS) plus rapide pour les chargements de tableaux mais qui utilise plus de mémoire de masse.

```
1       FOR DD=0 TO 8:READ C:POKE #F800+DD,C:NEXT
2       DATA #C5,#CD,#79,#D8,#CD,#18,#C9,#C1,#C9
4       GOSUB 8900
5       II=1
6       GOSUB 52800:II=II+1:IF II<=30 GOTO 6

10      REM DAIminiCALC  /Ph. WANET 3/83 ----   IMP INT !!!
20      CLEAR 16000:DIM R$(30.0),A(30.0,15.0),A$(30.0,15.0),PIL$(30.0),
        PR$(10.0),PR(10.0),P(10.0),T$(0.0),E$(21.0)
23      TTA$="  TEXTES  ":FOR IK=1 TO 14:AA$=STR$(IK):AA$=LEFT$(AA$,LEN(AA$)-2)
24      TTA$=TTA$+SPC(8-LEN(AA$))+AA$:NEXT:TTA$=TTA$+"          "
25      POKE #131,1:PRINT :PRINT :INPUT "DATE : ";DATE$:FOR JJ=1 TO 60:S$=S$+"
        ":NEXT
27      FOR IK=0 TO 21:E$(IK)=S$:NEXT:FOR IK=0 TO 30:R$(IK)=S$+S$+S$:NEXT
30      COLORT 8 0 14 3
40      POKE #75,32:CC=#FF:PC=31:YY=1:YZ=1:GOSUB 5000:INIT=1:GOSUB 3700
98      CURSOR 0,1:PRINT "'Space'=entr. donnee.'Shift+cur.'=recopier donnee":
        PRINT "'#'=formule.'/'=commandes";:CURSOR 0,0:WAIT TIME 20:GOSUB 1500
99      CX=CURX:CY=CURY:GOSUB 1000:CURSOR CX,CY
100     QK=GETC:IF QK=0 THEN 100
110     IF QK>15.0 AND QK<20.0 THEN CC=0:GOSUB 1000
112     IF QK>19 AND QK<24 THEN 2000
115     IF QK=35.0 THEN CURSOR 0,0:PRINT SPC(59);:CURSOR 0,0:PRINT "#";:A$(YZ,
        15.0)="":GOSUB 3000:GOTO 99
120     IF QK=17.0 THEN GOSUB 4000
130     IF QK=16.0 THEN GOSUB 4100
140     IF QK=18.0 THEN GOSUB 4200
150     IF QK=19.0 THEN GOSUB 4300
160     IF QK>15.0 AND QK<20.0 THEN CC=#FF:GOTO 99
170     IF QK=32 THEN 500
195     IF QK=47 THEN CURSOR 0,0:PRINT SPC(59);:CURSOR 0,0:PRINT "/";:GOSUB
        6000:GOTO 98
299     GOTO 99
500     CURSOR 0,0:INPUT S$
510     LI=8:IF XZ=0 THEN LI=12
520     IF LEN(S$)>LI THEN S$=RIGHT$(S$,LI)
530     LK=LEN(S$):SS=0:FN=0
535     IF LK=0.0 THEN 590
537     SI=1:IF LEFT$(S$,1)="-" THEN SI=-1:LK=LK-1:S$=RIGHT$(S$,LK)
540     FOR IK=0 TO LK-1
550   1   A$=MID$(S$,IK,1):IF A$>="0" AND A$<="9" THEN SS=SS*10+ASC(A$)-48:GOTO
      1   570
560   1   FN=1:SS=0:GOTO 580
570     NEXT IK
580     IF FN=0 THEN A(YZ,XZ)=SI*SS:IF SI=(-1) THEN S$="-"+S$
590     A$(YZ,XZ)=S$

600     REM IF SI%=(-1.0) THEN S$="-"+S$
603     LK=LEN(S$):S$=SPC(LI-LK)+S$:R$(YZ)=LEFT$(R$(YZ),12+(XZ-1)*LI)+S$+
        RIGHT$(R$(YZ),(14-XZ)*8)
605     E$(YY)=MID$(R$(YZ),(XZ-XX)*8,60)
610     CURSOR 0,23-YY:PRINT E$(YY);:CURSOR 0,0:PRINT SPC(58);:CURSOR 0,0
620     GOTO 99
700     CURSOR 0,0:PRINT S$;:RETURN
800     LK=LEN(S$):SS=0
810     IF LK=0.0 THEN 870
820     SI=1:IF LEFT$(S$,1)="-" THEN SI=-1:LK=LK-1:S$=RIGHT$(S$,LK)
830     FOR II=0 TO LK-1
840   1   A$=MID$(S$,II,1):IF A$>="0" AND A$<="9" THEN SS=SS*10+ASC(A$)-48:GOTO
      1   860
```



```
845   1   IF A$=" " THEN 860
850   1   SS=0:GOTO 870
860       NEXT II
865       IF SI=(-1.0) THEN S$="-"+S$
870       SS=SI*SS:RETURN
1000      POKE #75,32:POKE #74,1:X1=12+XX*8:DD=0:IF XX=0 THEN DD=6
1010      AA=#BFEF-YY*#86-X1*2+5+DD
1030      FOR IK=AA TO AA-14-DD STEP -2
1040  1   POKE IK,CC
1050      NEXT IK
1060      CURSOR 0,1:PRINT SPC(60);:CURSOR 0,0:PRINT SPC(59);:CURSOR 0,1:PRINT
          A$(YZ,15.0)
1070      POKE #75,#FF:POKE #74,0:RETURN
1500      SOUND 0 0 15 0 FREQ(500.0):WAIT TIME 2:SOUND OFF :RETURN
2000      SS=A(YZ,XZ):S$=A$(YZ,XZ)
2005      IF QK=21.0 THEN GOSUB 4000
2010      IF QK=20.0 THEN GOSUB 4100
2020      IF QK=22.0 THEN GOSUB 4200
2030      IF QK=23.0 THEN GOSUB 4300
2032      LI=8:IF XZ=0 THEN LI=12
2034      IF LEN(S$)>LI THEN S$=RIGHT$(S$,LI)
2035      A$(YZ,XZ)=S$
2037      IF S$<>"" AND SS=0 THEN 2050
2040      A(YZ,XZ)=SS:GOSUB 20000
2050      GOTO 603

3000      REM ENTREE DES FORMULES & NPI
3005      INPUT EX$:IF EX$="" THEN 3020
3010      IF LEFT$(EX$,1)="~" THEN 3800
3020      EX$="#"+EX$+",":PP=1:TR$="":PL=0:C2$="":PAR=0
3030      GOSUB 3440:REM LIRE
3040      PIL$(PP)=C2$:PP=PP+1
3050      GOSUB 3440:REM LIRE
3055      IF C2$="<" THEN C2$=CHR$(92)
3056      IF C2$=">" THEN C2$=CHR$(94)
3060      IF C2$=" " THEN 3050
3070      IF C2$="(" THEN 3140
3080      IF C2$>="A" AND C2$<=CHR$(94) THEN 3170
3090      IF C2$="+" OR C2$="-" OR C2$="*" OR C2$="/" THEN 3200
3100      IF C2$=")" THEN 3260
3110      IF C2$>="0" AND C2$<="9" THEN 3340
3120      IF C2$="," THEN 3310
3130      ER=1:GOTO 3430

3140      REM (
3150      PAR=PAR+1:IF C1$>="A" AND C1$<=CHR$(94) THEN ER=2:GOTO 3430
3160      PIL$(PP)=C2$:PP=PP+1:GOTO 3050

3170      REM OPERANDE
3180      IF (C1$>="A" AND C1$<=CHR$(94)) OR C1$=")" THEN ER=3:GOTO 3430
3190      TR$=TR$+C2$:GOTO 3050

3200      REM OPERATEUR
3210      IF C1$="+" OR C1$="-" OR C1$="*" OR C1$="/" OR C1$="(" OR C1$="#" THEN
          ER=4:GOTO 3430
3220      C$=C2$:GOSUB 3570:SPRI=PRI
3230      C$=PIL$(PP-1.0):GOSUB 3570
3240      IF SPRI>PRI THEN PIL$(PP)=C2$:PP=PP+1:GOTO 3050
3250      TR$=TR$+PIL$(PP-1.0):PP=PP-1:GOTO 3230
```

```
3260    REM )
3270    PAR=PAR-1:IF C1$="+" OR C1$="-" OR C1$="*" OR C1$="/" THEN ER=5:GOTO
        3430
3280    IF PIL$(PP-1.0)="(" THEN PP=PP-1:GOTO 3050
3290    IF PIL$(PP-1.0)="#" THEN ER=6:GOTO 3430
3300    TR$=TR$+PIL$(PP-1.0):PP=PP-1:GOTO 3280
3310    IF PAR<>0 THEN ER=9:GOTO 3430
3320    IF PIL$(PP-1.0)="#" THEN A$(YZ,15.0)=EX$:II=YZ:GOSUB 52800:CURSOR 0,0:
        PRINT SPC(58);:CURSOR 0,0:RETURN
3330    TR$=TR$+PIL$(PP-1.0):PP=PP-1:GOTO 3320

3340    REM CONSTANTE
3350    FLP=0
3360    GOSUB 3440:IF C2$>="0" AND C2$<="9" THEN 3360
3390    IF C2$="." THEN FLP=FLP+1:GOTO 3360
3400    IF FLP<=1 THEN 3060
3410    ER=10
3430    PRINT " Erreur de syntaxe ";ER;:WAIT TIME 300:CURSOR 0,0:PRINT SPC(58);:
        CURSOR 0,0:RETURN

3440    REM LIRE  C2
3450    C1$=C2$
3460    IF C1$="," THEN RETURN
3470    C2$=MID$(EX$,PL,1):PL=PL+1:RETURN

3570    REM PRIOR(C$)
3580    PRI=0:FOR IK=1 TO 8
3590  1  IF PR$(IK)=C$ THEN PRI=PR(IK):RETURN
3600    NEXT IK:ER=8:RETURN

3700    REM TABLE DE PRIORITE
3710    PR$(1.0)="#":PR(1.0)=0
3720    PR$(2.0)="(":PR(2.0)=1
3730    PR$(3.0)="+":PR(3.0)=2
3740    PR$(4.0)="-":PR(4.0)=2
3750    PR$(5.0)="*":PR(5.0)=3
3760    PR$(6.0)="/":PR(6.0)=3
3770    PR$(7.0)=")":PR(7.0)=4
3780    PR$(8.0)=",":PR(8.0)=5
3790    RETURN
3800    L$=MID$(EX$,1,1):LK=LEN(EX$)-3:EX$=RIGHT$(EX$,LK)
3805    IF RIGHT$(EX$,1)<>":" THEN EX$=EX$+":"
3810    LS=ASC(L$)-64:IF LS=(-4) OR LS=(-2) THEN LS=LS+32
3820    S$=A$(LS,15.0):IF S$="" THEN 3840
3825    GOSUB 3900:A$(YZ,15.0)=S$
3830    II=YZ:EX$=S$:GOSUB 52800
3840    CURSOR 0,0:PRINT SPC(58);:CURSOR 0,0:RETURN
3900    LL=LEN(S$)-1:FOR IK=0 TO LL
3910  1  FOR JJ=0 TO LK STEP 3:C1$=MID$(EX$,JJ,1):C2$=MID$(EX$,JJ+1,1)
3915  2   IF MID$(EX$,JJ+2,1)<>":" THEN PRINT " erreur de syntaxe ";:S$="":GOTO
      2   3950
3920  2   IF MID$(S$,IK,1)=C1$ THEN S$=LEFT$(S$,IK)+C2$+RIGHT$(S$,LL-IK)
3930  1  NEXT JJ
3940    NEXT IK
3950    RETURN
4000    IF YY=21.0 THEN 4040
4010    IF YY<21.0 THEN YY=YY+1
4020    IF YZ<30 THEN YZ=YZ+1
4030    RETURN
4040    IF YZ<30 THEN YZ=YZ+1:GOSUB 4500
```



```
4050    RETURN
4100    IF YY=1 THEN 4140
4110    IF YY>1 THEN YY=YY-1
4120    IF YZ>1 THEN YZ=YZ-1
4130    RETURN
4140    IF YZ>1 THEN YZ=YZ-1:GOSUB 4500
4150    RETURN
4200    IF XX=0 THEN 4240
4210    IF XX>0 THEN XX=XX-1
4220    IF XZ>0 THEN XZ=XZ-1
4230    RETURN
4240    IF XZ>0 THEN XZ=XZ-1:GOSUB 4500
4250    RETURN
4300    IF XX=6 THEN 4340
4310    IF XX<6 THEN XX=XX+1
4320    IF XZ<14 THEN XZ=XZ+1
4330    RETURN
4340    IF XZ<14 THEN XZ=XZ+1:GOSUB 4500
4350    RETURN
4500    FOR KK=1 TO 21
4510  1  GOSUB 4600
4520    NEXT KK
4530    GOSUB 5000:RETURN
4600    E$(KK)=MID$(R$(KK+YZ-YY),(XZ-XX)*8,60):RETURN
5000    POKE #75,32:POKE #74,1:PRINT CHR$(12);
5001    COLORT 9 9 9 9:FOR HK=#BFE8 TO #BF6D STEP -2:POKE HK,#FF:NEXT
5002    PRINT MID$(TTA$,(XZ-XX)*8,60);
5003    FOR IK=1 TO 22:AA=#BFEF-#86*(IK-1)-4:JJ=63+IK+YZ-YY:IF JJ=92 OR JJ=94
        THEN JJ=JJ-32
5004    POKE AA,JJ:POKE AA-3,#FF:NEXT IK:PRINT
5005    IF INIT=0 THEN 5075
5008    FOR JJ=1 TO 21:PRINT E$(JJ):NEXT JJ:GOTO 5075
5010    PRINT :FOR JJ=1 TO 30
5020  1  LK=LEN(A$(JJ,0.0)):A$=SPC(12-LK)+A$(JJ,0.0)
5025  1  FOR IK=1 TO 14:IF A(JJ,IK)>0.0 THEN AA=A(JJ,IK):SA$="":GOSUB 5500
5030  2   IF A(JJ,IK)<0.0 THEN AA=-A(JJ,IK):SA$="-":GOSUB 5500
5040  2   IF A(JJ,IK)=0.0 THEN 5060
5050  2   A$=A$+SPC(8-LK)+AA$+ESP$:GOTO 5065
5060  2   LS=LEN(A$(JJ,IK)):IF LS>8.0 THEN LS=8:A$(JJ,IK)=RIGHT$(A$(JJ,IK),8):
      2   GOTO 5065
5061  2   IF LS>0.0 THEN A$=A$+SPC(8-LS)+A$(JJ,IK)+ESP$:GOTO 5065
5062  2   A$=A$+SPC(8)+ESP$
5065  1  NEXT:PRINT A$;:IF LEN(A$)<80.0 THEN PRINT
5066  1  IF FLFOR=1.0 THEN PRINT A$(JJ,15.0)
5068  1  IF FLFOR=0.0 THEN PRINT
5070    NEXT:CURSOR 0,0:PRINT SPC(50);
5075    COLORT 9 15 14 3
5080    POKE #74,0:POKE #75,#FF:RETURN
5500    AA$=""
5510    QK=AA/10:RK=AA-10*QK:AA$=CHR$(48+RK)+AA$:AA=QK:IF QK>0 THEN 5510
5520    AA$=SA$+AA$:LK=LEN(AA$):IF LK>8 THEN AA$="overflow":LK=8
5530    RETURN
6000    CURSOR 0,1:PRINT "Load.Save.Calcul.Decal.Transla.Ins.Print.Fin.Zero ! =
        z ? <>";:CURSOR 0,0
6005    PK=GETC:IF PK=0 THEN 6005
6010    PRINT CHR$(PK);
6015    IF PK=80 THEN 6300
6020    IF PK=83 THEN GOSUB 7000:GOTO 6200
6030    IF PK=76 THEN GOSUB 8000:GOTO 6200
6040    IF PK=67 THEN GOSUB 9000:GOTO 6200
```

```
6050    IF PK=84 THEN GOSUB 12000:GOTO 6200
6060    IF PK=68 THEN GOSUB 13000:GOTO 6200
6070    IF PK=33 THEN XZ=XX:YZ=YY:GOSUB 4500:GOTO 6200
6080    IF PK=61 THEN XZ=XX+8:YZ=YY:GOSUB 4500:GOTO 6200
6090    IF PK=63 THEN XZ=XX+8:YZ=YY+9:GOSUB 4500:GOTO 6200
6100    IF PK=122 THEN XZ=XX:YZ=YY+9:GOSUB 4500:GOTO 6200
6110    IF PK=73.0 THEN GOSUB 10000:GOTO 6200
6120    IF PK=90 THEN GOSUB 11500:GOTO 6200
6130    IF PK=62 THEN GOSUB 6500:GOTO 6200
6140    IF PK=60 THEN GOSUB 6600:GOTO 6200
6200    CURSOR 0,1:PRINT SPC(59);:CURSOR 0,0:RETURN
6300    PRINT :INPUT " AVEC FORMULES O/N ";T$:FLFOR=0:IF T$="O" OR T$="O " THEN
        FLFOR=1
6301    POKE #131,0:PRINT :PRINT CHR$(27);"-";CHR$(1);CHR$(14);T$(0.0);:PRINT
        CHR$(27);"-";CHR$(0);
6310    PRINT :ESP$=" ":PRINT CHR$(15);:GOSUB 5010:POKE #131,1:ESP$="":GOSUB
        5000:GOTO 6200
6500    INPUT "Recopier (a droite) jusqu'a ";T$:IF T$="" THEN T$="14"
6510    TK=VAL(T$):KK=YZ:SA=A(KK,XZ):FOR JJ=XZ+1 TO TK:A(KK,JJ)=SA:SS=SA:GOSUB
        20000:A$(KK,JJ)=S$:GOSUB 15000:NEXT JJ
6520    KK=YY:GOSUB 4600:CURSOR 0,23-KK:PRINT E$(KK);:RETURN
6600    INPUT "Recopier (a gauche) jusqu'a ";T$:IF T$="" THEN T$="1"
6610    TK=VAL(T$):KK=YZ:SA=A(KK,XZ):FOR JJ=XZ-1 TO TK STEP -1:A(KK,JJ)=SA:SS=
        SA:GOSUB 20000:A$(KK,JJ)=S$:GOSUB 15000:NEXT JJ
6620    KK=YY:GOSUB 4600:CURSOR 0,23-KK:PRINT E$(KK);:RETURN
7000    CURSOR 0,1:PRINT SPC(59);:CURSOR 0,1:PRINT "titre  I=IDEM(que
        precedent)  S=SKIP  R=REW 1  N=annul.save";
7001    CURSOR 0,0:PRINT SPC(59);:CURSOR 0,0:INPUT T$:CURSOR 0,2:IF T$="I" AND
        T$(0)<>"" THEN 7006
7002    IF T$="R" THEN CALLM #F000:REM REW2:LOOK
7003    IF T$="S" THEN CALLM #F000:REM SKIP
7004    IF T$="I" OR T$="S" OR T$="R" THEN 7000
7005    T$(0)=T$:IF T$="" OR T$="N" THEN RETURN
7006    SAVEA T$ T$(0.0)+" "+DATE$
7010    SAVEA A$
7020    RETURN

8000    REM CURSOR 0,1:PRINT SPC(59);
8005    LOADA T$ :CURSOR 0,1:PRINT SPC(58);:CURSOR 0,1:PRINT T$(0.0);
8006    CURSOR 0,0:PRINT SPC(59);:CURSOR 0,0:PRINT " <S>=SKIP 1   <R>=REWIND 1
        <C>=GARDER FORM  <space>=LOAD ";
8007    P!=GETC:IF P!=0 THEN 8007
8008    IF P!=67 THEN GOSUB 8600:GOTO 8014
8009    FLAGFO=0
8012    IF P!=83.0 THEN 8050
8013    IF P!=82 THEN 8070
8014    LOADA A$ :GOSUB 8900
8016    FOR IK=1 TO 30:LK=LEN(A$(IK,0.0)):R$(IK)=SPC(12-LK)+A$(IK,0.0):FOR JJ=1
        TO 14:S$=A$(IK,JJ):LK=LEN(S$):GOSUB 800:A(IK,JJ)=SS
8017  2   IF LK=0 THEN S$=SPC(8)
8018  2   GOSUB 20030:R$(IK)=R$(IK)+S$
8019    NEXT JJ:NEXT IK
8020    GOSUB 4500
8021    IF FLAGFO=1 THEN GOSUB 8700:GOTO 8030
8022    GOSUB 8900:II=1
8023    EX$=A$(II,15.0):GOSUB 52800
8025    II=II+1:IF II<31 THEN 8023
8030    RETURN
8050    CALLM #F000:REM SKIP1
8060    GOTO 8000
```



```
8070    CALLM #F000:REM REW3
8075    GOTO 8000
8600    FOR II=0 TO 30:PIL$(II)=A$(II,15):NEXT II:FLAGFO=1
8610    RETURN
8700    FOR II=0 TO 30:A$(II,15)=PIL$(II):NEXT II
8710    RETURN
8900    CURSOR 0,0:PRINT SPC(59);:CURSOR 0,0:PRINT "Busy    Even geduld
        Attendre svp!";:RETURN

9000    REM CALCULE
9005    CURSOR 0,0:PRINT " Je suis occupe ";
9030    FOR JJ=1 TO 14:PP=0
9100  1  GOSUB 55000
9300    NEXT JJ
9450    CURSOR 0,0:PRINT SPC(58);:CURSOR 0,0
9460    GOSUB 4500
9500    RETURN
10000   INPUT "NSERER A PARTIR DE LA LIGNE No ";I$:IF I$<="9" THEN IK=VAL(I$):
        I$=CHR$(IK+64):GOTO 10010
10005   IK=ASC(I$)-64:IF IK=(-2) OR IK=(-4) THEN IK=IK+32
10010   FOR JJ=30 TO IK+1 STEP -1
10020 1  FOR KK=0 TO 14:A$(JJ,KK)=A$(JJ-1.0,KK):A(JJ,KK)=A(JJ-1.0,KK):NEXT KK:
      1  R$(JJ)=R$(JJ-1.0)
10025 1  IMIN=IK
10030   T$=A$(JJ-1.0,15.0):GOSUB 10500:A$(JJ,15.0)=T$:NEXT JJ:GOSUB 4500
10040   II=IMIN
10050   EX$=A$(II,15):GOSUB 52800
10060   II=II+1:IF II<31 THEN 10050
10100   RETURN
10500   LK=LEN(T$):IF LK=0 THEN RETURN
10505   FOR KK=0 TO LK-1:C$=MID$(T$,KK,1)
10510 1  IF C$>=I$ AND C$<=CHR$(94) THEN T$=LEFT$(T$,KK)+CHR$(ASC(C$)+1)+
      1  RIGHT$(T$,LK-KK-1)
10520   NEXT KK:RETURN
11000   FOR KK=0 TO 14:A$(JJ,KK)=A$(IK,KK):A(JJ,KK)=A(IK,KK):NEXT:RETURN
11500   CURSOR 1,0:PRINT SPC(56);:CURSOR 0,0:INPUT "ZERO LIGNE OU COLONNE
        (L/C/ R=fin )";T$
11505   IF T$<>"L" AND T$<>"C" THEN CURSOR 0,0:PRINT SPC(59);:RETURN
11510   IF T$="L" THEN GOSUB 11600:GOTO 11500
11520   IF T$="C" THEN GOSUB 11700:GOTO 11500
11600   INPUT L$:IF (L$<"A" AND L$<>"<" AND L$<>">") OR L$>CHR$(94) THEN RETURN
11610   JJ=ASC(L$)-64:IF JJ=(-2) OR JJ=(-4) THEN JJ=JJ+32
11620   FOR KK=0 TO 14:A$(JJ,KK)="":A(JJ,KK)=0:NEXT:R$(JJ)=SPC(124):GOSUB 4500:
        RETURN
11700   INPUT JJ:FOR KK=1 TO 30:A$(KK,JJ)="":A(KK,JJ)=0:S$=SPC(8):GOSUB 15000:
        NEXT:GOSUB 4500:RETURN
12000   INPUT IK:INPUT " vers ";JJ:CURSOR 0,0:IF IK>14 OR JJ>14 OR IK<0 OR JJ<0
        THEN 12000
12010   FOR KK=1 TO 30:SS=A(KK,IK):A(KK,JJ)=SS:S$=A$(KK,IK):A$(KK,JJ)=S$:IF S$=
        "" THEN GOSUB 20000
12012 1  IF LEN(S$)<>8.0 THEN GOSUB 21000
12015   GOSUB 15000:NEXT KK
12020   GOSUB 4500:RETURN
13000   INPUT DA$:INPUT " vers ";AA$:IF (DA$<"A" AND DA$<>"<" AND DA$<>">") OR
        DA$>CHR$(94) OR (AA$<"A" AND AA$<>"<" AND AA$<>">") OR AA$>CHR$(94)
        THEN RETURN
13010   IK=ASC(DA$)-64:IF IK=(-2) OR IK=(-4) THEN IK=IK+32
13020   JJ=ASC(AA$)-64:IF JJ=(-2) OR JJ=(-4) THEN JJ=JJ+32
13030   GOSUB 11000
13035   R$(JJ)=R$(IK)
```

```
13040   GOSUB 4500:RETURN

14000   REM LISTE DES TTITRES
14005   DIM T$(0,0):IK=0
14010   IK=IK+1:POKE #131,0:LOADA T$ :PRINT IK;" - ";T$(0,0)
14020   CALLM #F000:REM SKIP1
14030   POKE #131,1
14040   GOTO 14010
15000   IF LEN(R$(KK))<124.0 THEN R$(KK)=R$(KK)+SPC(124-LEN(R$(KK)))
15010   R$(KK)=LEFT$(R$(KK),JJ*8+4)+S$+RIGHT$(R$(KK),112-8*JJ):RETURN
20000   S$="":IF SS=0.0 THEN 20030
20005   SI=1:IF SS<0 THEN SI=-1:SS=-SS
20010   QQ=SS/10:RR=SS MOD 10:S$=CHR$(RR+48)+S$:IF QQ<>0 THEN SS=QQ:GOTO 20010
20015   IF SI<0.0 THEN S$="-"+S$
20020   IF LEN(S$)>8 THEN S$="overflow":LK=8:RETURN
20030   S$=SPC(8-LEN(S$))+S$:LK=8:RETURN
21000   IF LEN(S$)>8.0 THEN S$=RIGHT$(S$,8):GOTO 21020
21010   S$=SPC(8-LEN(S$))+S$
21020   RETURN
25000   S$="":IF SS=0 THEN RETURN
25010   SI=1:IF SS<0 THEN SI=-1:SS=-SS
25020   QQ=SS/10:RR=SS MOD 10:S$=CHR$(RR+48)+S$:IF QQ<>0 THEN SS=QQ:GOTO 25020
25030   IF SI<0 THEN S$="-"+S$
25040   IF LEN(S$)>8 THEN S$="overflow"
25050   RETURN
52800   LEI=LEN(EX$)-1:IF LEI<2 THEN LINE$=LEFT$(STR$(55005.0+II*10),6):GOSUB
        53000:LINE$=LEFT$(STR$(55006.0+II*10),6):GOSUB 53000:RETURN
52805   IK=0
52810   C$=MID$(EX$,IK,1):IF C$="#" THEN EX$=RIGHT$(EX$,LEI):LEI=LEI-1:GOTO
        52810
52815   IF C$="," THEN EX$=LEFT$(EX$,LEI):LEI=LEI-1
52816   IF C$="[" THEN EX$=LEFT$(EX$,IK)+"A0%"+RIGHT$(EX$,LEI-IK):LEI=LEI+2
52820   IF C$="<" THEN EX$=LEFT$(EX$,IK)+"A1%"+RIGHT$(EX$,LEI-IK):LEI=LEI+2
52830   IF C$="]" THEN EX$=LEFT$(EX$,IK)+"A2%"+RIGHT$(EX$,LEI-IK):LEI=LEI+2
52840   IF C$=">" THEN EX$=LEFT$(EX$,IK)+"A3%"+RIGHT$(EX$,LEI-IK):LEI=LEI+2
52850   IK=IK+1:IF IK<=LEI+1 THEN 52810
52860   LINE$=LEFT$(STR$(55005.0+II*10.0),6)+"SS%="+EX$:GOSUB 53000
52865   LINE$=LEFT$(STR$(55006+II*10),6)+"A%("+STR$(II)+",JJ%)=SS%:GOSUB 25000:
        A$("+STR$(II)+",JJ%)=S$"
52867   LINE$=LINE$+":GOSUB 21000:KK%="+STR$(II)+":GOSUB 15000"
52870   GOSUB 53000:RETURN
52900   IJ=1
52910   LINE$=LEFT$(STR$(55500+IJ*10),6):GOSUB 53000:LINE$=LEFT$(STR$(55000+
        IJ*10),6)
52920   IF IJ>26 THEN LINE$=LINE$+"A"+CHR$(21+IJ):GOTO 52940
52930   LINE$=LINE$+CHR$(64+IJ)
52940   LINE$=LINE$+"=A%("+STR$(IJ)+",JJ%)"
52945   GOSUB 53000
52950   IJ=IJ+1:IF IJ<31 THEN 52910
52960   STOP
53000   LINE$=LINE$+CHR$(13)+CHR$(0):A=VARPTR(LINE$):A=PEEK(A)+PEEK(A+1)*256+1:
        B=A+LEN(LINE$)
53010   POKE #A2,A MOD 256:POKE #A3,A SHR 8:POKE #A4,B MOD 256:POKE #A5,B SHR 8:
        POKE #135,2:CALLM #F800:POKE #135,0:RETURN

55000   REM *******
55010   A=A(1.0,JJ)
55020   B=A(2.0,JJ)
55030   C=A(3.0,JJ)
55040   D=A(4.0,JJ)
```



```
55050   E=A(5.0,JJ)
55060   F=A(6.0,JJ)
55070   G=A(7.0,JJ)
55080   H=A(8.0,JJ)
55090   I=A(9.0,JJ)
55100   J=A(10.0,JJ)
55110   K=A(11.0,JJ)
55120   L=A(12.0,JJ)
55130   M=A(13.0,JJ)
55140   N=A(14.0,JJ)
55145   SS=J+K
55146   A(14.0,JJ)=SS:GOSUB 25000:A$(14.0,JJ)=S$:GOSUB 21000:KK=14.0:GOSUB 15000
55150   O=A(15.0,JJ)
55155   SS=((B*D/100+E)*I*F/1000/J+G)*103.42/C+H+0.9
55156   A(15.0,JJ)=SS:GOSUB 25000:A$(15.0,JJ)=S$:GOSUB 21000:KK=15.0:GOSUB 15000
55160   P=A(16.0,JJ)
55165   SS=((B*D/100+E)*I*F/1000/K+G)*103.42/C+H+0.9
55166   A(16.0,JJ)=SS:GOSUB 25000:A$(16.0,JJ)=S$:GOSUB 21000:KK=16.0:GOSUB 15000
55170   Q=A(17.0,JJ)
55175   SS=((B*D/100+E)*I*F/1000/L+G)*103.42/C+H+0.9
55176   A(17.0,JJ)=SS:GOSUB 25000:A$(17.0,JJ)=S$:GOSUB 21000:KK=17.0:GOSUB 15000
55180   R=A(18.0,JJ)
55185   SS=(B*D/100+E)*F/J/C*0.10342+0.9
55186   A(18.0,JJ)=SS:GOSUB 25000:A$(18.0,JJ)=S$:GOSUB 21000:KK=18.0:GOSUB 15000
55190   S=A(19.0,JJ)
55195   SS=(B*D/100+E)*F/K/C*0.10342+0.9
55196   A(19.0,JJ)=SS:GOSUB 25000:A$(19.0,JJ)=S$:GOSUB 21000:KK=19.0:GOSUB 15000
55200   T=A(20.0,JJ)
55205   SS=(B*D/100+E)*F/L/C*0.10342+0.9
55206   A(20.0,JJ)=SS:GOSUB 25000:A$(20.0,JJ)=S$:GOSUB 21000:KK=20.0:GOSUB 15000
55210   U=A(21.0,JJ)
55215   SS=(B*D/100+E)*F*I/J/C*0.10342+0.9
55216   A(21.0,JJ)=SS:GOSUB 25000:A$(21.0,JJ)=S$:GOSUB 21000:KK=21.0:GOSUB 15000
55220   V=A(22.0,JJ)
55225   SS=(B*D/100+E)*F*I/K/C*0.10342+0.9
55226   A(22.0,JJ)=SS:GOSUB 25000:A$(22.0,JJ)=S$:GOSUB 21000:KK=22.0:GOSUB 15000
55230   W=A(23.0,JJ)
55235   SS=(B*D/100+E)*F*I/L/C*0.10342+0.9
55236   A(23.0,JJ)=SS:GOSUB 25000:A$(23.0,JJ)=S$:GOSUB 21000:KK=23.0:GOSUB 15000
55240   X=A(24.0,JJ)
55250   Y=A(25.0,JJ)
55260   Z=A(26.0,JJ)
55270   A0=A(27.0,JJ)
55280   A1=A(28.0,JJ)
55290   A2=A(29.0,JJ)
55300   A3=A(30.0,JJ)
56000   RETURN
```

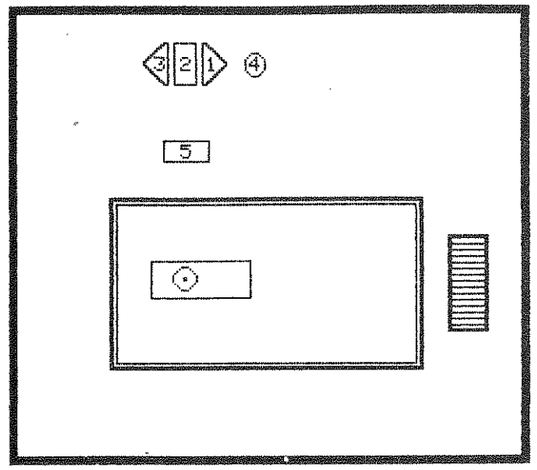
DEMO

TARASCON ete 83

```
 DEBUT        1/ 4    5/ 5    8/ 6    1/ 9    1/10   15/10    1/ 4   25/ 5    1/ 9
   FIN        4/ 5    7/ 6   30/ 8   30/ 9   14/10   31/10   24/ 5   30/ 8   30/ 9
     AVION   12500   12500   13230   12500   12500   12500   12500   13230   12500
  TRANSFERT   2000    2000    2000    2000    2000    2000     800     800     800
 COEFF.BEN.     82      82      82      82      82      82      82      82      82
    CHANGE      56      56      56      56      56      56      56      56      56
     NUITS       4       4       4       4       4       4       4       4       4
 HOTELS   :  SEKIS   SEKIS   SEKIS   SEKIS   SEKIS   SEKIS  DEURTH  DEURTH  DEURTH
    DOUBLE    1600    2300    2900    2900    2300    1600    1600    1650    1650
#,
    SINGLE    2000    2800    3400    3400    2800    2000    2100    2200    2200
#,
  SUP REPAS    600     600     600     600     600     600     500     500     500
#,

#,
  TOTAL DBL  22098   24075   26691   25770   24075   22098   21250   22312   21391
#(C+(D+I*G)*F/100)*103.42/E+400,
    SINGLE   23228   25488   28103   27183   25488   23228   22663   23866   22945
#(C+(D+J*G)*F/100)*103.42/E+400,
   DBL/jour   1130    1624    2048    2048    1624    1130    1130    1165    1165
#I*F*1.0342/E,
   SGL/jour   1412    1977    2401    2401    1977    1412    1483    1553    1553
#J*F*1.0342/E,
  SUP REPAS    423     423     423     423     423     423     353     353     353
#K*F*1.0342/E,
  DEBUT      1/ 4    5/ 5    8/ 6    1/ 9   15/10    1/ 4    8/ 6    1/ 9   1/10
#,
  FIN        4/ 5    7/ 6   30/ 8   14/10   31/10    7/ 6   30/ 8   30/ 9  31/10
#,
   HOTELS     BECH    BECH    BECH    BECH    BECH   SEKIM   SEKIM   SEKIM   SEKIM
#,
  TRANSFERT   2000    2000    2000    2000    2000    4000    4000    4000    4000
#,
    DOUBLE    1400    2050    2750    2750    1400     550     600     600     550
#,
    SINGLE    1600    2700    3400    3400    1600     800     900     900     800
#,
  SUP REPAS    750     750     750     750     750
#,
    AVION     12500   12500   13230   12500   12500   12500   13230   12500   12500
#,
  TOTAL DBL  21532   23369   26267   25346   21532   20544   21606   20685   20544
#(Y+(U+V*G)*F/100)*103.42/E+400,
    SINGLE   22098   25205   28103   27183   22098   21250   22453   21532   21250
#(Y+(U+W*G)*F/100)*103.42/E+400,
   DBL/jour    988    1447    1942    1942     988     388     423     423     388
#V*F*1.0342/E,
   SGL/jour   1130    1906    2401    2401    1130     565     635     635     565
#W*F*1.0342/E,
  SUP REPAS    529     529     529     529     529
#X*F*1.0342/E,
```

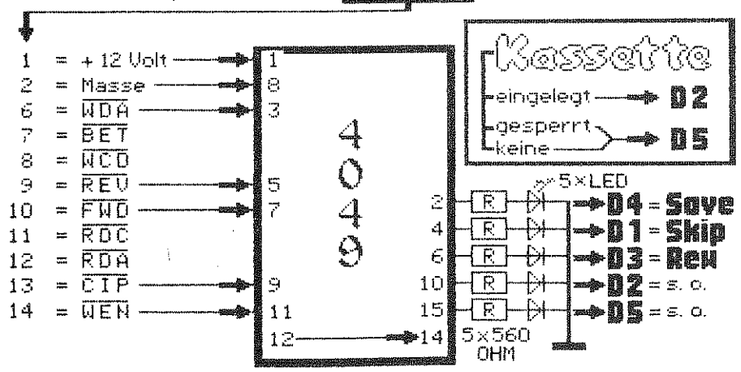

Hardy → DCR-Funktionsanzeige ← Strobel

Mit 5 Leuchtdioden alles in Blick ! ! !

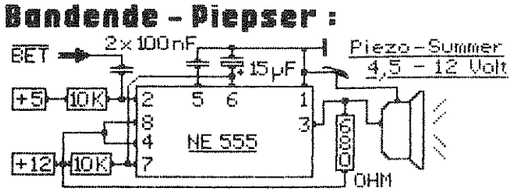


DCR-FRONTANSICHT

In der DCR geht ein Flachbandkabel von der Zusatzplatine zur Laufwerksplatine, dort ist es angelötet Die Anschlußpunkte sind numeriert.

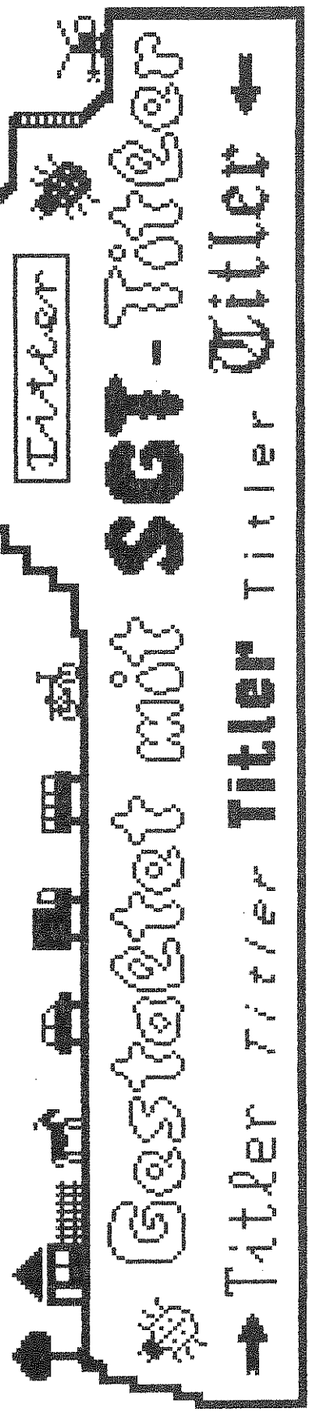


Bandende - Piepser :



Layout : Hardy Strobel, Neuselsbrunn 51, 8500 Nürnberg 50

Ausführung : Jean Marchand



De nieuwe DAI's

Na lange jaren te hebben doorgebracht met de vervaardiging van een enkel mikro model, beslist DAI het geweer van schouder te veranderen door diversifikatie van het aanbod.

De nieuwe machines zijn van het type « trois pièces » zoals men in de modewereld zegt : een systeembehuizing, een monitor en een toetsenbord.

De monitor is een Sanyo, die 225 × 528 punten in 16 kleuren afficheert; het toetsenbord en de centrale eenheid zijn gesierd met vlakke skai, die het in een Lodewijk XV salon fantastisch zouden moeten doen. Vier modellen worden voorgesteld : de DAI-T, die eeneenvoudige terminal is, en de DAI-T 1, 2 of 3, die monoposten zijn met 1, 2 of 3 diskette-drives van 5,25'', die door de DOS V3.0 elk met 640 K worden toebedeeld.

De CPU is steeds die goeie ouwe 8080. Naar waarheid krijgt hij wel wat ouderdom, maar bij Intel verzekert men, dat deze processor meestal slechts op 5 of 10 % van zijn mogelijkheden wordt aangewend; indien DAI dit percentage opvoert, dan is het mogelijk, dat hij tot prestaties komt, die vergelijkbaar zijn met de moderne kringen.

Twee programma's werden ons getoond op de op Bureau 84 voorgestelde prototypes : een spreadsheet en een tekstverwerking. Hun oorspronkelijkheid ligt in het systematische gebruik van kleur om de aktiviteiten te beschrijven.

Maar het grote probleem blijft steeds hetzelfde : de programma's. Omdat deze machines over geen enkele andere bron beschikken, vermits ze slechts kompatibel zijn met elkaar, moet men op een specifieke produktie wachten om de toepassingen te ontwikkelen. Voor het ogenblik moeten zowat 5 tot 6 programma's beschikbaar zijn; men dient toe te geven, dat de keuze niet zeer ruim is.

Maar in ieder geval, good luck ! □

```
2     REM :    ************************************
3     REM :    *                                  *
4     REM :    *        R.MARCEL  FEVRIER 1982    *
5     REM :    *                                  *
6     REM :    *           FEU D'ARTIFICE         *
7     REM :    ************************************
100   POKE #75,32:MODE 6:MODE 0:PRINT CHR$(12):COLORT 9 14 9 9
110   POKE #BDD7,#4F:CURSOR 1,19:PRINT "FEU":POKE #BC45,#5F:CURSOR 3,16:PRINT "D'ARTIFICE"
115   POKE #BAB3,#4F:CURSOR 0,13:PRINT "-----":WAIT TIME 200
120   CF=9.0:C1=3.0:C2=14.0:C3=15.0
130   MODE 6:COLORG CF C1 C2 C3
140   FOR J=32.0 TO 8.0 STEP -12.0
150   A=C1:IF J=20.0 THEN A=C2
160   IF J=8.0 THEN A=C3
170   GOSUB 500:XA=XN:YA=YN
180   FOR I=0.0 TO 2.0*PI STEP PI/128.0:GOSUB 500:DRAW XMAX/2-XA,YMAX/2-YA XMAX/2-XN,YMAX/2-YN A
:XA=XN:YA=YN:NEXT I
190   NEXT J
220   FOR K=1.0 TO 100.0:Q=1.0-Q
230   COLORG 0 Q*RND(16.0) RND(16.0) RND(16.0)
240   WAIT TIME 20+RND(100.0):NEXT
250   GOTO 130
500   R=J*(1.0-COS(12.0*I)*3.0)
510   XN=R*COS(I):YN=R*SIN(I):RETURN
```


EPROM PACK

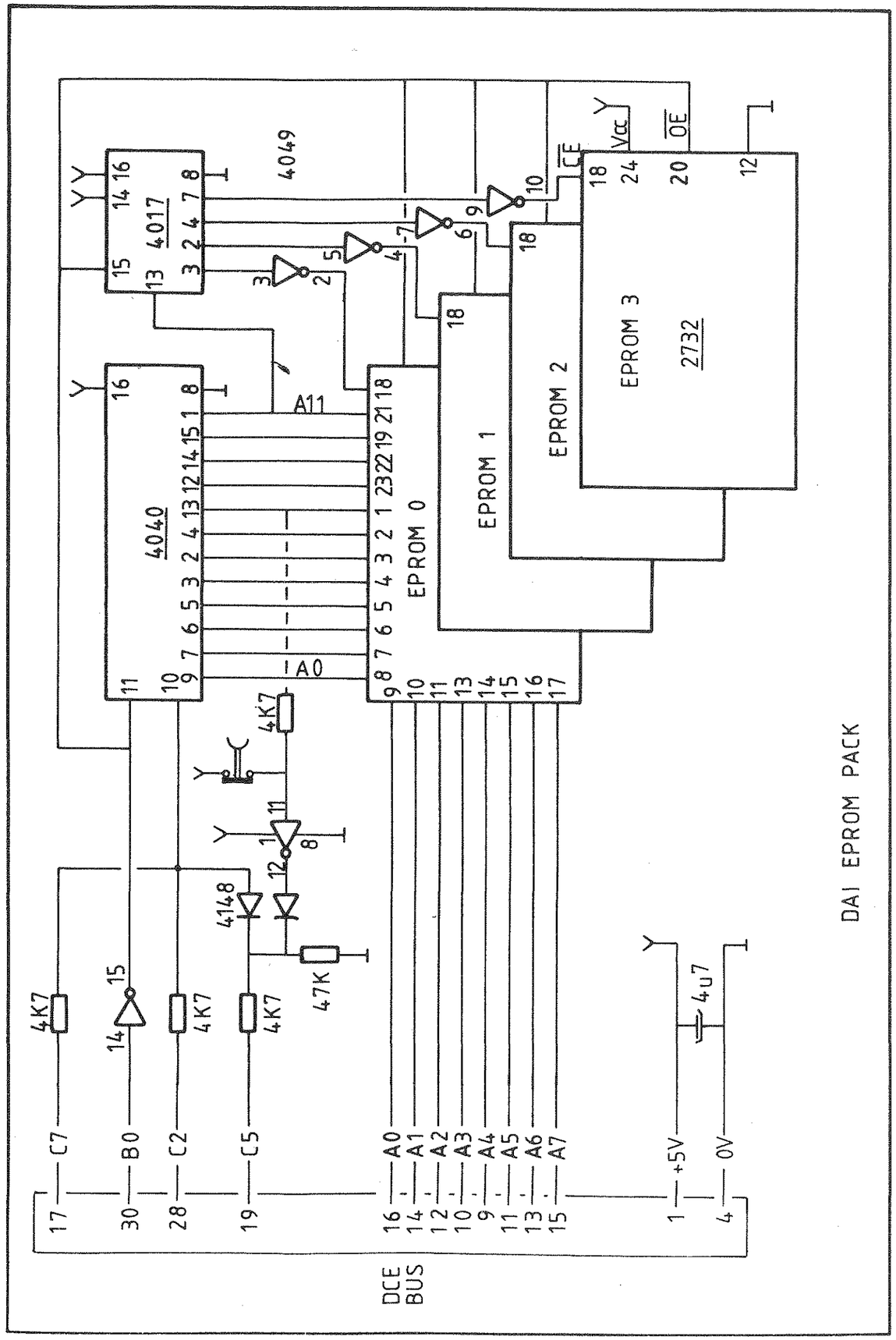
```
1     ;
2     ;
3     ;
4     ;WAARSCHIJNLIJK HEEFT IEDERE DAI-GEBRUIKER ZICH WEL
5     ;EENS GEERGERD AAN HET STEEDS WEER MOETEN LADEN VAN DE
6     ;VEEL GEBRUIKTE ROUTINES OF PROGRAMMA'S. MET NAME
7     ;DIEGENEN DIE VEEL MET ASSEMBLER WERKEN EN EEN AUDIO-
8     ;CASSETTERECORDER ALS OPSLAGMEDIUM GEBRUIKEN.
9     ;DE DCE-BUS WORDT TIJDENS DE POWER-UP ROUTINE OF NA
10    ;EEN RESET DOOR MIDDEL VAN EEN IN DE FIRMWARE OPGENO-
11    ;MEN ROUTINE GECONTROLEERD OP EEN EVENTUEEL AANGESLO-
12    ;TEN APPARAAT DAT WIL COMMUNICEREN.
13    ;INDIEN DIT ZO IS, ZAL DE OP DE BUS AANGEBODEN DATA OP
14    ;DE STACK GEPLAATST WORDEN. DEZE DATA MOET EEN PRO-
15    ;GRAMMA ZIJN, DAT IN DE RUIMTE VAN DE STACK PAST (F800
16    ;-F900). DIT PROGRAMMA WORDT DOOR DE DCE-BOOTSTRAP GE-
17    ;START. MET BEHULP VAN DIT PROGRAMMA, DAT OP ZIJN
18    ;BEURT OOK WEER EEN BOOTSTRAP KAN ZIJN, WORDT DE DATA
19    ;OPGEHAALD VAN DE GEWENSTE ROUTINE OF HET GEWENSTE
20    ;PROGRAMMA.
21    ;NATUURLIJK MOET DEZE BOOTSTRAP DE DATA OP DE JUISTE
22    ;PLAATS GAAN SCHRIJVEN. DUS ZAL HET BEGIN-ADRES EN
23    ;HET EIND-ADRES VAN DE ROUTINE OF VAN HET PROGRAMMA
24    ;MEEGEGEVEN MOETEN WORDEN.
25    ;VERDER IS ER EEN STUK HARDWARE NODIG OM DE IN DE
26    ;EPROM OPGESLAGEN DATA OP DE DCE-BUS TE DOEN ZETTEN.
27    ;DE BOVENGENOEMDE FILOSOFIE HEEFT GELEID TOT DE ONT-
28    ;WIKKELING VAN HET DAI-EPROM-PACK.
29    ;DEZE ONTWIKKELING WORDT DOOR ENKELE DAI-GEBRUIKERS,
30    ;DIE ZICH MET ASSEMBLER (DNA EN SPL) BEZIG HOUDEN,
31    ;AL ENIGE TIJD TOEGEPAST. NA HET AANZETTEN VAN DE DAI-
32    ;MACHINE IS BIJVOORBEELD SPL DIRECT BESCHIKBAAR.
33    ;DE LAADTIJD WORDT IN DE PRAKTIJK NIET BEMERKT, OMDAT
34    ;DEZE SLECHTS 1-2 SEC. BEDRAAGT
35    ;ER ZIJN DIVERSE BOOTSTRAP PROGRAMMA'S IN GEBRUIK.
36    ;EEN TWEE STUKS ZIJN HIER AFGEDRUKT.
37    ;DE EERSTE VERSIE IS MINDER DAN 64 BYTES GROOT EN
38    ;LAADT EEN AANEENGESLOTEN BLOK VAN EEN OF MEERDERE
39    ;PROGRAMMA'S. DE WEERSTAND VAN 4K7 BIJ HET VERBREEK-
40    ;CONTACT IS DAN OP PIN 4 VAN DE CD4040 AANGESLOTEN.
41    ;DE TWEEDE VERSIE IS GROTER DAN 64 BYTES EN IS BEDOELD
42    ;OM GESCHEIDEN PROGRAMMA'S OF ROUTINES TE LADEN OP
43    ;TEVOREN BEPAALDE ADRESSEN. DE WEERSTAND VAN 4K7 IS
44    ;DAN AANGESLOTEN ZOALS HET SCHEMA AANGEEFT OP PIN 13
45    ;VAN DE CD4040.
46    ;ER WORDT GEWERKT AAN EEN BOOTSTRAP WAARBIJ ER DIRECT
47    ;EEN MENU VERSCHIJNT, WAARDOOR MEN KAN KIEZEN UIT
48    ;BIJVOORBEELD: A-BASIC  B-SPL  C-TEKSTVERWERKEN.
49    ;EEN ADVIES MET BETREKKING TOT DE HARDWARE:
50    ;DE C-MOS IC'S EN DE EPROMS ZIJN SOMS MOEILIJK
51    ;VERKRIJGBAAR, WAARDOOR DE PRIJZEN HOOG KUNNEN LIGGEN.
52    ;ONDERZOEK DUS EERST DE VERKRIJGBAARHEID VAN DEZE COM-
53    ;PONENTEN. HET IS NIET NOODZAKELIJK DIRECT ALLE EPROMS
54    ;TE INSTALLEREN, MET EEN WERKT HET OOK.
55    ;
```

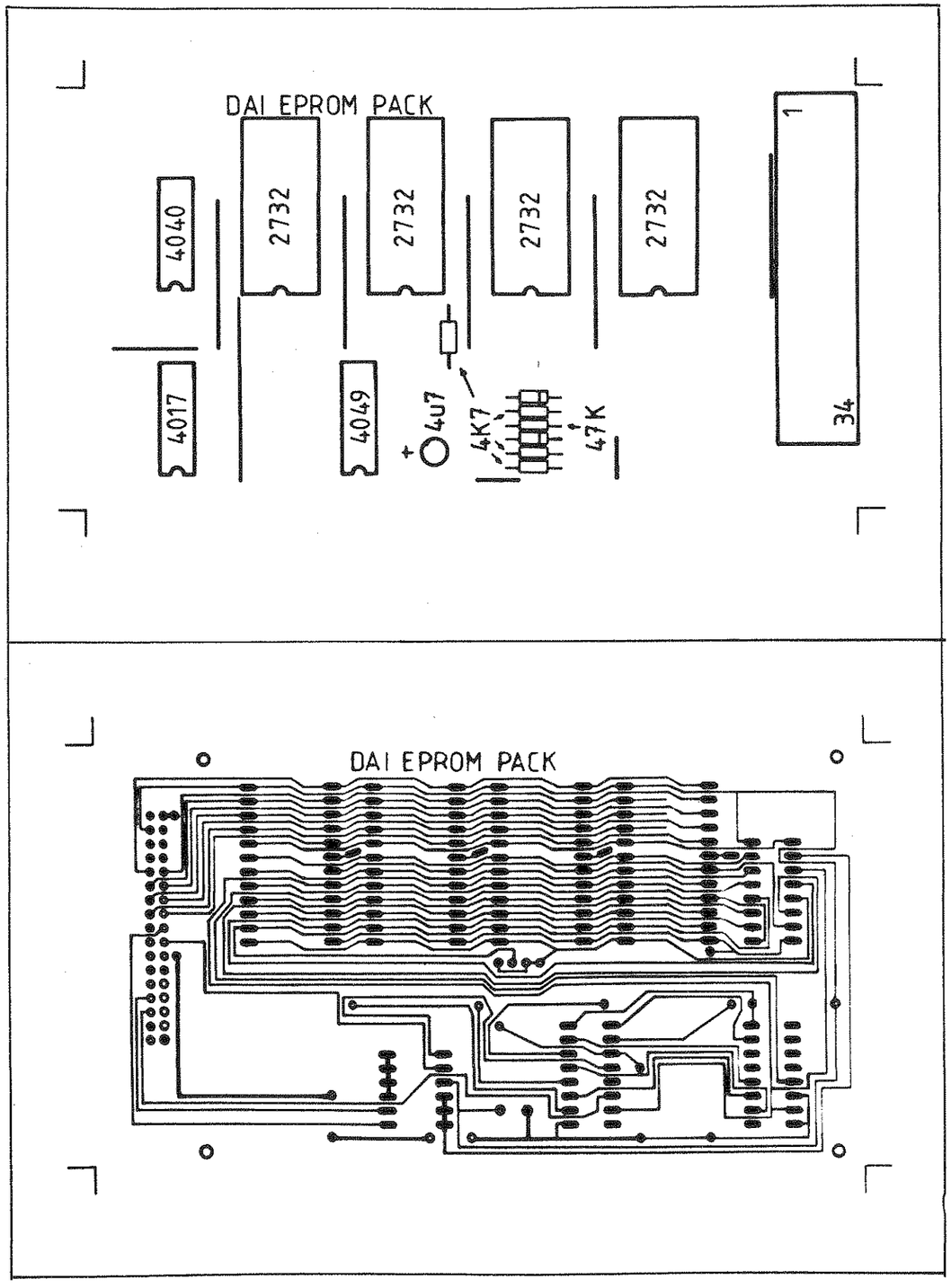
```
56      ;DE HARDWARE.
57      ;DE MAXIMALE GEHEUGENCAPACITEIT IS 16K. (4X(4KX8))
58      ;HIER GAAT DE BOOTSTRAP VAN AF.


SPL

59      ;HET SCHEMA GEEFT DE PENAANSLUITINGEN VAN DE DCE-BUS
60      ;WEER MET DE FUNCTIE ER VAN.
61      ;POORT A(0-7)     DATABUS
62      ;       C(2,5,7)  CONTROLEERT DE COMMUNICATIE
63      ;       B(0)      RESET
64      ;
65      ;C7 (IN)          IS ER DATA GELDIG? INIEN C7="1" DAN
66      ;                 WORDT DE DATA OP POORT A INGELEZEN.
67      ;B0 (OUT)         RESET (SOFTWARE)
68      ;C2 (OUT)         DATA-REQUEST
69      ;C5 (IN)          DATA-PRESENT.
70      ;                 INDIEN C5="1" WORDT ER NAAR DE
71      ;                 DCE-INPUTROUTINE GESPRONGEN.
72      ;A0-A7 (IN)       DATA IN
73      ;
74      ;DRUKTOETS        INDIEN HET CONTACT IS GEOPEND TIJDENS
75      ;                 DE POWER-UP ROUTINE OF EEN RESET WORDT
76      ;                 DE EPROM INHOUD GELADEN.
77      ;
78      ;4017B            DECADE COUNTER
79      ;4040B            BINARY COUNTER (12-STAGE  2+12=4096)
80      ;4049B            BUFFER (6X)
81      ;2732             EPROM (4KX8=4096)
82      ;
83      ;BRONNEN
84      ;DAINAMIC         '80 NR.1  BLZ 3-4   DE DCE-BUS
85      ;                 N.B.: IN DE TEKENING DE 1 EN DE 2 VAN
86      ;                       P1 EN P2 VERWISSELEN.
87      ;
88      ;                 '82 NR.9  BLZ 9     DCE-CONNECTIONS
89      ;                                     JOS SCHEPENS.
90      ;
91      ;                 '83 NR.14 BLZ 28-29 INITIALISATION
92      ;                                     DCE-PERIPHERALS
93      ;                 N.B.: ADRES #EFA7-#EFB2 PORT A
94      ;                       MOET ZIJN PORT B
95      ;
96      ;INTEL            2732 EPROM
97      ;DATA-SHEETS      8255 P.P.I.
98      ;
99      ;FIRM-WARE        DCE-INPUTROUTINE
100     ;MANUAL           B.J.  BOERRIGTER
101     ;
102     ;
```



```
103             TITL    'BOOTSTRAP A.J.'
104     ;
105     HEAP    EQU     2ECH
106     PPI     EQU     0FE00H
107     STACK   EQU     0F800H
108     ;
109             ORG     0F800H
110     ;
111             LHLD    ENDA            ;EXECUTED AFTER
112             XCHG                    ;RESET
113             LHLD    STARTA
114     ;
115     START   LDA     PPI             ;LOAD DATA
116             MOV M,A                 ;STORE DATA
SPL V1.1  PAGE 3        BOOTSTRAP A.J.
117             MOV A,H                 ;)CHECK
118             SUB D                   ;)
119             JZ      NOT             ;)FOR
120             MOV A,L                 ;)
121             SUB E                   ;)END-
122             JZ      ENDR            ;)ADDRESS
123     ;
124     NOT     MVI A   4H              ;RESET PC2
125             STA     PPI+3H
126     NOG     LDA     PPI+2H          ;)WAIT
127             ANI     80H             ;)C7
128             JNZ     NOG             ;)LOW
129             MVI A   5H              ;+PC2
130             STA     PPI+3H          ;+HIGH
131             INX H
132             JMP     START
133     ENDR    LXI H   HEAP
134             RET
135             ORG     0F82FH
136     STARTA  DW      8400H
137     ENDA    DW      0B2FFH
138             END
139     ;
140     ;
141             TITL    'BOOTSTRAP J.A.P. 840109'
142     ;
143     ;*1* IF HEAPPOINTERS MUST NOT BE CHANGED,
144     ;*1* PUT IN THIS 2 INSTRUCTIONS.
145     ;
146             ORG     0F800H
147     COUNT1  EQU     297H
148     ;
149             NOP                     ;(E5) PUSH H  *1*
150             LXI H   TABLE
151             PUSH H
152     ;
153     BLOCK   MOV C,M                 ;LOW-
154             INX H                   ;--START ADDRESS
155             MOV B,M                 ;HIGH-
156             INX H
157             MOV E,M                 ;LOW-
158             INX H                   ;--END ADRESS
159             MOV D,M                 ;HIGH-
160             INX H
161             SHLD    COUNT1
162             PUSH B                  ;(
163             POP H                   ;( BE TO HL
164             CALL    COMPAR
```

```
                                        ;HL=DE
165             JNZ     START1          ;NOT THEN START1
166             POP H                   ;YES THEN POP END ADDRESS
167             LDA     0FE01H          ;!
168             ANI     0FEH            ;!
169             STA     0FE01H          ;!RESET EPROMPACK
170             NOP                     ;(E1) POP H  *1*
171             RET                     ;END OF ROUTINE
172   ;
173   START1    POP B                   ;KILL SAVED END ADDRESS
174             LXI B   0FE03H
SPL V1.1  PAGE 4        BOOTSTRAP J.A.P. 840109

175   START2    LDA     0FE00H          ;GET DATA
176             MOV M,A                 ;STORE DATA
177             MVI A   4H              ;) RESET PC2
178             STAX B                  ;)        =INCR. COUNTER
179             INR A                   ;) A=05
180             STAX B                  ;) SET PC2
181             INX H                   ;) NEXT ADDRESS
182             CALL    COMPAR          ;END ADDRESS REACHED?
183             JNZ     START2          ;NOT THEN START2
184             PUSH H                  ;YES THEN PUSH END ADDRESS
185             LHLD    COUNT1
186             JMP     BLOCK
187   ;
188   ;COMPARE HL-DE.
189   COMPAR    MOV A,H
190             SUB D
191             RNZ
192             MOV A,L
193             SUB E
194             RET
195   ;
196   ;START- AND END-ADDRESS TABLE.
197   ;
198   TABLE     DW      800H            ;STARTADDRESS (FIRST PART)
199             DW      810H            ;EN ADDRESS+1
200             DW      83EH            ;STARTADDRESS (SECOND PART)
201             DW      843H            ;END ADDRESS+1
202             DW      900H            ;STARTADDRESS (THIRD PART)
203             DW      999H            ;END ADDRES+1
204   ;         ,,      ,,              ;,,     ,,      ,,
205   ;         ,,      ,,              ;,,     ,,      ,,
206             DW      0H              ;) AT THE END ALWAYS
207             DW      0H              ;) 8X ZERO
208   ;
209             END
```



ERRATA DATAFLEX

ERRATTA DATAFLEX ERRATTA DATAFLEX ERRATTA DATAFLEX ERRATTA DATAF

Mr Siegfried Brys meld ons een bug in het Dataflex programma op de Dataflex diskette. Er zijn problemen wanneer een bestand vol is en men toch probeert het aan te vullen.(Ø - ENTER) Volgende wijzigingen worden voorgesteld :

```
1000 FOR P%=USED%+1 TO LENGTH%:R$=""
```

wordt :

```
1000 IF USED%=LENGTH% GOTO 1075
1005 FOR P%=USED%+1 TO LENGTH%:R$=""
```

en :

```
1070 GOSUB  950:NEXT P%:GOSUB 300:PRINT "FILE IS FULL,  IT NEEDS
EXTENDING !";
```

wordt :

```
1070 GOSUB 950:NEXT P%
1075 GOSUB 300:PRINT "FILE IS FULL, IT NEEDS EXTENDING !";
```

!!! Type : *UT, >Z3, >B voor U editeert. !!!

Mocht U problemen hebben hiermee stuur Uw copy terug, wij zenden U de gewijzigde versie terug.

Gedurende de boodschap 'FILE IS FULL ....' moet U de spatiebalk indrukken om terug in het menu te komen.

Mocht er nog iemand op een ei zitten, aarzel niet in Uw pen te kruipen.

Couwberghs Frans
Boekdonkstraat 13
3980 Tessenderlo
013/666340

FEESTWENSEN

1985

```
2     MODE 0:PRINT CHR$(12)
3     COLORT 8 5 6 14
4     FOR I=#BFEE TO #B3E4 STEP -#86:POKE I,#F0+RND(15):NEXT
5     A$="           +-                                                ":GOSUB 100
10    A$="          +$*-                                               ":GOSUB 100
20    A$="         +*****-                    +-                      ":GOSUB 100
30    A$="        +******$*-                 +**-                     ":GOSUB 100
40    A$="       +$*******$***-    +-        +***$*-                  ":GOSUB 100
50    A$="          !!          +$*-        +*******-                ":GOSUB 100
60    A$="                     +*****-      +$*******$**-            ":GOSUB 100
70    A$="                    +*******-    +***$******$**-           ":GOSUB 100
80    A$="    +-             +******$**-          !!                 ":GOSUB 100
81    A$="   +**-           +*****$******-                          ":GOSUB 100
83    A$="  +*$**-        +*********$*****-        PRETTIGE         ":GOSUB 100
85    A$=" +*******-    +*$**********$****-         KERSTDAGEN       ":GOSUB 100
86    A$="+*$*****$*- +*****************$**-       & EEN VOORSPOEDIG  ":GOSUB 100
87    A$="*****$***********$*******$*******-            1985         ":GOSUB 100
90    A$="    !!    +*********************-                          ":GOSUB 100
91    A$="                    !!                                     ":GOSUB 100
92    A$="                                                           ":GOSUB 100
94    A$="                                                           ":GOSUB 100
95    A$="============================================================":GOSUB 100
96    A$="door u toegewenst door:                  Jeroen Overvoorde ":GOSUB 100
98    A$="============================================================":GOSUB 100
99    GOTO 99
100   FOR I=0 TO 59:W=ASC(MID$(A$,I,1)):IF W=32 THEN PRINT " ";:NEXT
110   IF W=42 THEN PRINT CHR$(127);:NEXT
120   IF W=43 THEN PRINT CHR$(6);:NEXT
130   IF W=45 THEN PRINT CHR$(7);:NEXT
140   IF W=33 THEN PRINT " ";:XX=CURX*2:YY=CURY*#86:POKE #B3E5-XX+YY-9,#FF:NEXT
145   IF W=61 THEN PRINT CHR$(1);:NEXT
150   IF W=36 THEN PRINT CHR$(127);:XX=CURX*2:YY=CURY*#86:POKE #B3E5-XX+YY-9,56:NEXT
155   IF I>59 THEN PRINT :RETURN
160   POKE #B3E4+CURY*#86,#CD:POKE #B3E4+(CURY-1)*#86,#C8
170   PRINT CHR$(W);:NEXT
```