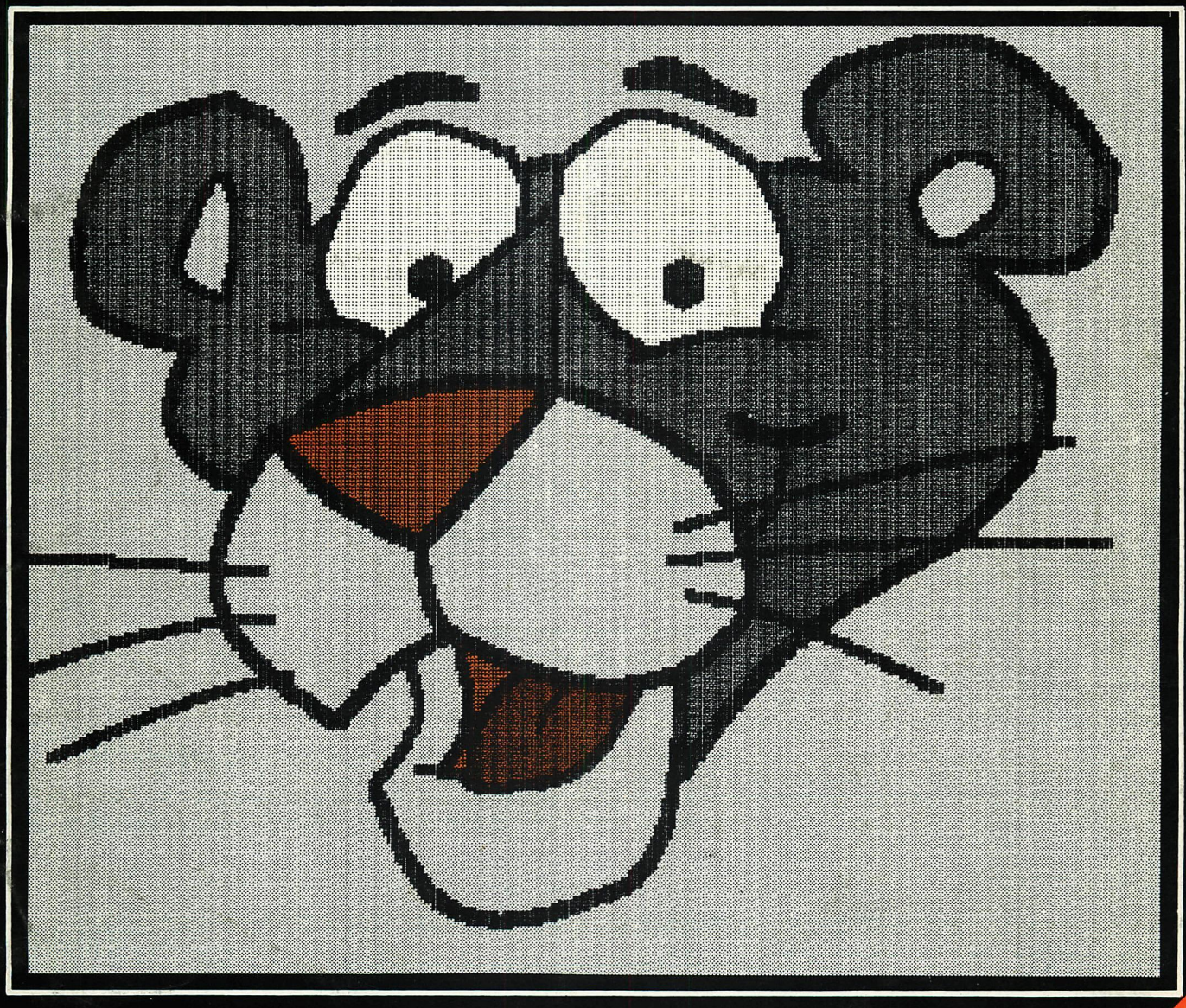
DAI NAMIC

21

tweemaandelijks tijdschrift maart - april 1984



personal computer users club

een uitgave van dainamic v.z.w.
verantw. uitgever w. hermans, mottaart 20 - 3170 herselt

International

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
	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

Voor Nederland :

GIRO : 4083817
t.n.v. J.F. van Dunne'
Hoflaan 70
3062 JJ ROTTERDAM
Tel. : (010) 144802

Generale Bankmaatschappij Leuven
nr. 230-0045353-74

Inzendingen : Games & Strategy

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

DAI NAMIC

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
0	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

Herselt, april 1984

Beste leden,

De jaarlijkse bijeenkomst in Tongelsbos is weer achter de rug, de nazorg hiervan is nog bezig, verlofdagen in de drukkerij ... U begrijpt het al, ons blad is weer een paar weken achter op het schema.
Geen nieuwe software in dit nummer, er liggen wel een aantal pakketten op de testbank, die komen volgende keer beslist aan bod.
Met tevredenheid (en opluchting) kunnen we U een nieuwe mankracht voorstellen: Frans Couwberghs. Reeds een hele tijd actief in de DAInamic-equipe is Frans nu fulltime werkzaam voor de vereniging. In het kader van activiteiten met vrijstelling van stempelcontrole neemt Frans ons heel wat werk uit handen , we wensen dat hij nog een tijdje doorgaat, maar hopen toch dat hij spoedig zijn plaats op de arbeidsmarkt terug mag innemen.
In dit nummer heel wat informatie over INDATA en KEN-DOS floppy-sytemen, wij hopen dat deze artikels kunnen bijdragen tot een verantwoorde keuze..
Er gaan geruchten dat in Nederland nog een nieuw floppy-systeem in ontwikkeling zou zijn, we berichten wel indien we nieuws ontvangen.
Samen met DIALOGUE-informatique onderzoeken we de mogelijkheid om hun programma's aan een democratische prijs aan te bieden, dit zou echter van de vereniging een belangrijke investering vragen, wij ontvangen graag uw suggesties hieromtrent..
De PACMAC-wedstrijd is gestreden, ziehier de resultaten genoteerd door H.Bellekens en K. Van de Perre :

Results of PAC-MAN contest on our 4th meeting

1.	Goyvaerts Guido	Westmeerbeek	594.760	(TV-monitor , INDATA)
2.	Goyvaerts Daniel	Westmeerbeek	579.680	(printer MIKROSHOP HAGELAND)
3.	Van Rompaey Rita	Ramsel	397.820	(software packet)
4.	Smits P.	Deurne	346.080	(")
5.	Cheng Kenneth	Arnhem	310.610	(")
6.	Duluins Fabrice	Nivelles	302.350	(reduction card)
7.	Catry Henk	Buggenhout	299.270	(")
8.	Wijbouw Didier	Oostende	225.300	(")
9.	Assink	Eindhoven	198.270	(")
10.	De Boer A.	Zwaanhoek	182.390	(")

Dear members,

Above you can find the results of our PACMAC-contest on our 4th meeting. There were only 15 players, but I think that the enormous highscore of the Goyvaerts-brothers has frightened a lot of members to try their chance.
In this issue , no new software is announced, we are testing a lot for the moment, so look out for next issue !
Yannick Dupagne has tested the INDATA-floppies, Frans Couwberghs talks about KEN-DOS : these articles should assist you in your choice of the best suited system for your applications.
When visiting INDATA, we saw a lot of new faces : new sales-managers, a new marketing-director (Mr Art), and a new director (Mr Leonard). There were talking in mysterious terms about something that should happen in september (of this year!) ... we can only wait and see.

we start with edition of newsletter 22 right now, till then ...

Wilfried Hermans

NEWSLETTER 21

INHOUD — CONTENTS

DAInamic subscription rates :

Benelux : 1000 Bfr (renewal before 1 feb : 900 Bfr)
Europe : 1100 Bfr (renewal before 1 feb :1000 Bfr)
Outside Europe 1500 Bfr (" " " " 1400 Bfr)
(Air Mail)

pay to : Dainamic SUBSCRIPTIONS
B.Van rompaey
Bovenbosstraat 4
3044 HAASRODE-BELGIUM

* by check or
* on Bancaccount nr 230-0045353-74
of Generale Bank Leuven c/o DAInamic

Mijn eerste machinetaalprogramma

Ik bezit nu ongeveer anderhalf jaar een DAI en heb mijn programma's tot een paar maanden geleden allemaal in BASIC geschreven.
Vooral bij het "scrollen" van het scherm schoot BASIC echter te kort en ben ik me daarom in machinetaal gaan interesseren. Toen ik mijn eerste programma's, waar ik machinetaal in gebruikte, opstuurde werd mij gevraagd om, als iemand die net met machinetaal was begonnen, eens een artikel te schrijven over het beginnen met programmeren in machinetaal en welke problemen ik daarbij tegenkwam.
Dit deed ik graag en hier is dan het resultaat van mijn werk. Ik ga ervan uit dat men wel wat afweet van BYTES, BITS, CPU, opbouw van geheugen etc.. Dit artikel is dus bestemd voor die mensen, die al een aardige ervaring met BASIC hebben en als volgende stap een assembler hebben gekocht (of er een wilen gaan kopen), maar nog niet goed weten wat ze er mee aan moeten; die mensen zou ik graag willen helpen met hun eerste stap in machinetaal.

Ik wilde beginnen met de voor- en nadelen van machinetaal t.o.v. BASIC.
Machinetaal heeft maar een voordeel boven BASIC en dat is zijn snelheid, maar dit maakt dan ook wel een dusdanig verschil dat het b.v. voor spelletjes zeker de moeite waard is.
Verder heeft machinetaal alleen maar nadelen t.o.v. BASIC, omdat machinetaal dichter bij de computer en daardoor verder van de mens staat. Enkele van de belangrijkste nadelen zijn :

- Machinetaal kent maar 8 variabelen, registers genaamd (dit zijn de registers A,B,C,D,E,H,L en 'M').
- Deze registers kunnen slechts getallen van 0 tot 255 bevatten of als men 2 registers (B,C of D,E of H,L) samenneemt van 0 tot 65535. Inclusief de grenzen.
- Alle bewerkingen zoals optellen,aftrekken etc. kunnen vrijwel alleen in het in het A register plaatsvinden en bovendien kent machinetaal alleen maar optellen, aftrekken en enkele logische commando's.

Ik hoop dat U nog niet ontmoedigd bent en anders kan ik U vertellen dat het na even wennen en wat oefening best meevalt.
Ik zal U nu zoveel mogelijk vertellen wat ik weet,hierbij gebruik ik de 8080 instructionset (o.a. te vinden in DAInamic nr 18) met de assemblercodes.
Dit zijn afkortingen (zgn. mnemonics), die gemakkelijker zijn te onthouden dan de getallen welke de computer eigenlijk leest, hier is een assembler voor nodig (DNA of SPL).

Het blad bestaat uit 3 delen :
1) De naam van de instructie en wat voor een soort parameters hij er achter verwacht dit kan zijn :

reg = een register
rp = een registerpaar (B,C of D,E of H,L of PC of SP (of TOS))
data = een getal van 0-255
addr = een getal van 0-65535 (het adres van een BYTE in het geheugen)

2) Dit beschrijft de instruktie volgens de volgende regels :
   - Als om een register(paar) haakjes staan betekent dat, dat het gaat om de inhoud van dat register(paar).
   - als om een adres of om een registerpaar,waar al haakjes om staan, haakjes staan betekent het dat het gaat om de inhoud van dat adres.
   - <-- is het zelfde als in BASIC "=" ((A)<--(B) betekent dus A=B).

Opmerkingen :
- PC = Programcounter ; deze houdt bij bij welke regel (of beter gezegd : op welk adres) het machinetaalprogramma bezig is.
- SP = Stackpointer ; Deze houdt bij op welke adres in de stack het laatst iets is gezet; Op de stack zet men of de PC als men met de CALL groep een subroutine aanroept,of een ander register dat men met PUSH bewaart. (Op te roepen met respectievelijk RET en POP).
- Z,CY,P en S zijn FLAGS (= een 0(=niet waar) of een 1(=waar)) die "geset" (1 gemaakt) of "gereset" (0 gemaakt) worden bij bepaalde instrukties afhankelijk van de uitkomst.
  Bij welke instructies staat in DAInamic 15 van blz. 96 tot blz. 100. Hier staat ook wanneer welke FLAG "geset" wordt. Bovendien staat in dat artikel van iedere instruktie een BASIC equivalent en ik beveel daarom dat artikel zeker aan.
- Een streepje boven A of C (bij CMA en CMC) betekent dat enen nullen worden en omgekeerd.

3) Hierin staat het (hexadecimale) getal,dat als de CPU het leest de bijbehorende instructie uitvoert.
Opmerkingen :
- PSW is een registerpaar bestaande uit A en de register die alle FLAGS bevat.
- Sommige instructies vragen om een getal (b.v. MVI A,data = maak de inhoud van A gelijk aan data), dan staat er "dd" achter dit betekent dat er een willekeurige BYTE volgt.
- Sommige instructies vragen om een adres (b.v. JMP addr = GOTO addr) ,dan staat er "al ah" achter,dit betekent dat er twee willekeurige BYTES volgen,de eerste met het LAAGSTE gedeelte van het adres ,de tweede met het HOOGSTE; !!! dus JMP :02EF wordt vertaald als "C3 EEF 02". (de : betekent een hexadecimaal getal en C# is de code van JMP)

Men kent nu weliswaar alle instructies, maar alleen daarmee kan men nog moeilijk een programma maken, daarom geef ik nu enkele tips en aanwijzingen
- M is geen gewoon gewoon register maar de inhoud van de BYTE die door H,L wordt aangegeven : MVI M,A betekent dus POKE (H,L),A .
- Een LOOP is gemakkelijk te maken door iets als :

```
    MVI  B,100      * B=100
    LXI  H,:BFEE    * (H,L)=#BFEE:REM  ) LOOP  MOV M,B       * POKE (H,L),B:REM
) --> DIT KAN VAN ALLES ZIJN
    DCR  B          * B=B-1
    JNZ  LOOP       * IF B<>0 GOTO LOOP
```

  Let hierbij wel op dat b.v. "DCX rp" geen FLAGS verandert.
- Als men een bepaalde waarde even niet nodig heeft moet men die op een dumpadres wegpoken,om de registers vrij te houden.
  Om een soort variabelen te hebben gebruik ik de volgende manier :
  (Dit is in DNA geschreven.) VAR EQU :addr * VAR bevat het dump adres

```
    STA   VAR     * zet A op adrse VAR en
    LDA   VAR     * haal hem weer terug
```

  Zo is het ook mogelijk om een ARRAY te maken : VAR EQU :addr * idem

```
    LXI   H,VAR   * (H,L) bevat adres van VAR
    LXI   D,el    * (D,E) bevat welk element
    DAD   D       * (H,L) bevat adres van element
    MOV   A,M     * A bevat element
```

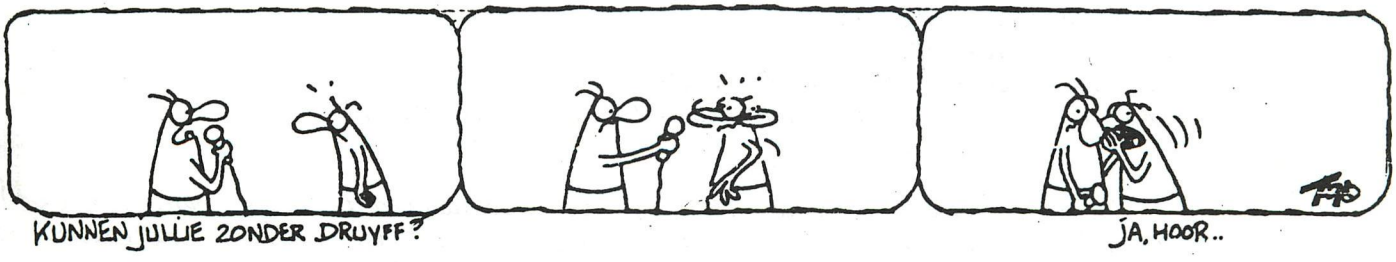
Dit is zo'n beetje wat ik in dit artikel kwijt wou en ik hoop dat het een beetje leesbaar en te volgen was,daar ik totaal geen ervaring heb in het

schrijven van dergelijke stukjes. Ik hoop dat U nu iets meer van machinetaal weet en dat U er mooie programma's mee gaat maken. Mocht U nog vragen hebben dan zal ik ze graag beantwoorden.

Mark Hooykaas
Bosrand 5
9451 AE Rolde
Drenthe Holland
05924-1948

```
10    REM NATO STARS / F.H. Druijff - 4/84
20    MODE 6:COLORG 9 13 0 0
30    XM=XMAX/2:YM=YMAX/2:R=100:M=20:GOSUB 100
40    XM=XM/2:YM=YM/2:R=R/2:M=M/2:GOSUB 100:YM=YMAX-YM:GOSUB 100
50    XM=XMAX-XM:GOSUB 100:YM=YMAX-YM:GOSUB 100:GOTO 40
99    GOTO 99
100   X=XM+R:P=XM-R:Y=YM+R:Q=YM-R
110   FOR I=0 TO M:DRAW X,YM XM+I,YM+I 21:DRAW P,YM XM-I,YM-I 21
120   DRAW XM,Q XM+I,YM-I 21:DRAW XM,Y XM-I,YM+I 21:NEXT
130   I=M:DRAW X,YM XM+I,YM-I 21:DRAW P,YM XM-I,YM+I 21
140   DRAW XM,Q XM-I,YM-I 21:DRAW XM,Y XM+I,YM+I 21:RETURN
```

```
10    REM CYCLOIDS / F.H. DRUIJFF 8/83
20    CLEAR 1000:DIM S!(61),C!(61):COLORG 0 5 10 15:MODE 6
30    P!=80.0:Q!=100.0:V!=-PI:W!=5.0*PI
40    FOR A!=V! TO W! STEP PI/10.0:I=I+1:S!(I)=SIN(A!):C!(I)=COS(A!):NEXT
50    FOR A!=0.1 TO 3.4 STEP 0.3:B!=A!*26.0:XO=13.0*V!+P!:YO=B!+Q!
60    N=0:FOR T!=V! TO W! STEP PI/10.0:N=N+1
70    XN=13.0*(T!-A!*S!(N))+P!:YN=Q!-B!*C!(N)
80    DRAW XO,YO XN,YN 21:DOT XO,YO 23:YO=YN:XO=XN:NEXT:NEXT
```



INDATA-FLOPPIES

Namur, le 23 mars 83

C'est depuis la fin de l'année précédente que je manipule les doubles drives 320 K et 640 K de la firme Indata. Et c'est avec grand plaisir qu'à l'école où j'enseigne, nous avons oublié le Memocom, pourtant excellent, ou le 2 x 80 K qui me fut prêté une semaine durant. L'acquisition de systèmes rapides et fiables de stockage de l'information était devenue urgente. En effet, l'utilisation fréquente du DAI en ordinateur pédagogique ou en système de traitement de texte, son utilisation pour la gestion comptable de l'école, celle prochaine pour la gestion de fichiers élèves et professeurs,... nécessitaient rapidité et souplesse.

Nous avons donc fait l'acquisition de trois unités : deux de 320 K et une de 640 K. C'est muni du double drive 320 K que je compose ce texte, au moyen de "La Plume du Dai", traitement de texte implanté personnellement. Sur l'unité plus puissante, nous avons implanté un programme de gestion comptable de l'institut. C'est dans ces conditions que je me permets de présenter une vision critique des floppies Indata.

La firme Indata commercialise trois unités différentes de doubles drives : le 160 K (deux disques de 80 K), le 320 K (2 x 160 K) et le 640 K (2 x 320 K). La première unité, munie de son dos V1.0 a déjà fait l'objet de sévères critiques dans cette revue. J'ai eu l'occasion d'en manipuler un exemplaire durant quelques jours, ma déception fut grande : lenteur et faiblesse du DOS. Je vous parlerai des nouvelles unités, celles qui, j'en suis sûr, valent vraiment leur prix. Mais c'est à vous d'en juger!

Voici les premières caractéristiques des trois unités:

160 K	2 x 80 K	SF SD 40 pistes de 16 secteurs de 128 octets	54479 F TVAc
320 K	2 x 160 K	SF SD 40 pistes de 16 secteurs de 256 octets	59087 F TVAc
640 K	2 x 320 K	SF DD 80 pistes de 16 secteurs de 256 octets	73637 F TVAc

NOTE : Elles sont toutes en simple face SF.
SD pour simple densité
DD pour double densité
le 160 K tourne sous le DOS V1.0 ou sous CP/M
les 320 K et 640 K tournent sous les DOS V2.0 et 2.1

Je ne parlerai plus des 160 K, les connaissant très peu. Il semble toutefois qu'ils aient été révisés à ce jour, et qu'ils soient nettement plus fiables que les précédents. Sont-ils toujours aussi lents?

Les nouveaux systèmes sont particulièrement rapides. En mesurant, grâce à l'horloge interne du DAI - plus fiable que la trotteuse de ma montre et qui n'est pas arrêtée par les accès au disque - le temps de chargement d'une image en MODE 6, depuis la mise en marche des moteurs jusqu'à la fin du chargement, je suis arrivé à 15,78 secondes. Ce qui est très bien. Le programme utilisé était le suivant:

```
10 MODE 6
20 POKE #1BF,#FF:POKE #1BE,#FF
30 POKE #131,3:PRINT "DLOAD image":POKE #131,1
40 A=PEEK(#1BE):B=PEEK(#1BF):PRINT (#FFFF-256*A-B)/50;"sec"
```

Par la même méthode, j'ai mesuré le temps de lecture d'un secteur sur disque. Il m'a fallu 1,88 secondes pour le premier - le temps de mettre le moteur en marche et de lire ce secteur -, et 0,08 seconde pour chacun des suivants. Cette rapidité est due, en partie, à la présence d'un buffer de 2 K - place pour 8 secteurs consécutifs - à l'intérieur même des unités. Le temps - à ma montre - d'un BACKUP est de l'ordre de 60 secondes pour un 320 K (118 pour un 640 K), celui d'un IDISK est de 20 secondes (37). Ces temps sont bons, et j'en ai pleine satisfaction.

De plus, les disques se montrent très fiables. Je n'ai pas encore su les mettre en défaut pour des opérations de lecture ou d'écriture. La seule condition est le branchement sur une bonne prise,...

Le DOS V2.0 est identique pour les systèmes 320 K et 640 K. Son jeu d'instruction est le même que celui du DOS V1.0 des 160 K. Voilà qui ne dépaysera personne. Mais les routines sont modifiées, et les accès au DOS sous moniteur sont donc différents. Vous aurez remarqué que les secteurs comptent 256 octets, ce qui est le double de ceux des 160 K. Bien que le DOS soit le même pour les deux "grosses" versions, le formattage propre à chacun des lecteurs interdit au 640 K la lecture des disques 320 K. La seule lecture possible est celle du directory : le lecteur 320 K peut lire le directory - table des matières - du 640 K et inversément. C'est tout! Voilà qui est bien malheureux, et qui nous oblige au transfert des programmes et des tableaux par la cassette. C'est lent mais efficace...

Le DOS - ensemble des commandes vous permettant la manipulation des disquettes - ne se trouve pas en ROM, ou seule la manipulation des cassettes est permise. Il faut donc le charger en RAM, au moyen d'une disquette qui le contient. C'est-à-dire une disquette qui contient le fichier $DKBOOTS long d'un secteur qui permet le chargement du DOS, le fichier $MSTRDOS long de 25 secteurs. Cette disquette d'initialisation est à insérer dans le lecteur gauche : le drive 0. A l'allumage simultané de tout le système, certains heureux jouissent d'un auto-chargement du DOS. Personnellement, j'ai toujours dû actionner le bouton RESET, normalement une seule fois. Si, sur la disquette, se trouve un programme machine baptisé $USER.BIN, il sera chargé et exécuté - par exemple la conversion du clavier en AZERTY -, et s'il se trouve un programme Basic nommé $USER.BAS, il sera ensuite chargé et exécuté, toujours en second lieu (AUTOSTART d'un programme).

A l'initialisation des disques, la piste 0 est réservée au directory. Celui-ci accepte 96 noms. Il vous restera donc aussi 39 pistes (79 pour le 640 K) ou encore 159744 (323584) octets libres sur le disque. Bien sûr, si vous désirez bénéficier de l'AUTOSTART d'un programme, il vous sera nécessaire de déduire les 26 secteurs nécessaires au DOS. Il vous restera alors 156928 (320768) octets sur disque. Ce qui est confortable.

L'enregistrement de chaque tableau, programme ou fichier est accompagné de son nom - au maximum 8 caractères -, son type - au maximum 3 caractères : BAS pour un programme Basic, BIN pour un code machine, STR, INT et FPT pour les tableaux de chaînes, d'entiers et de réels respectivement, ce que vous désirez pour les fichiers, par exemple TXT pour les textes rédigés par ce programme de traitement de texte, etc... - et son attribut - W si l'enregistrement est protégé contre l'écriture et N sinon -. La place occupée en secteurs complets, et divers renseignements nécessaires au système d'exploitation : fichier effacé ou non, position du premier secteur du fichier,...

Le DOS V2.0 permet l'utilisation simultanée des deux drives - baptisés 0 et 1 gauche et droit - et de deux cassettes ordinaires. Le passage disque - cassette est aisé et sans problème. Le DOS V2.1 permet la commande d'un maximum de 3 DCR en chaînant à la table d'instructions du DOS celles du Memocom, présentes dans la ROM auxiliaire.

La place mémoire occupée par le DOS n'est pas démesurée : la version 2.0 s'étend de 300 à 1B00 (hexadécimal), tandis que la version 2.1 s'étend de 300 à 1C00 si vous n'utilisez pas de DCR et de 300 à 1D00 sinon. Vous consommez donc environ 5 K RAM avec le DOS complet. Cependant, la commande TINYDOS vous permet de réduire cette place à 1,5 K RAM. Bien sûr, dans ce cas, vous n'aurez plus aucun accès aux commandes du DOS.

La différence entre les deux DOS n'est pas terrible. La nouvelle version, disponible depuis peu, permet la manipulation simultanée des disques et de trois DCR (numérotés 1, 2 et 3), alors que l'ancienne ne le permet nullement. De plus, la version 2.1 ajoute au DOS les commandes Basic RREC et WREC de lecture et écriture directe de secteurs sur disque. La version 2.0 ne le permet que sous moniteur, ou sous langage machine.

Les commandes Basic incluses au DOS sont nombreuses. Elles s'ajoutent aux commandes habituelles LOAD, SAVE, LOADA et SAVEA. De plus vous pouvez les augmenter et constituer ainsi sur disque une véritable extension de votre DAI. Les voici:

RESETD réinitialise la carte contrôleur des drives

DIR affiche le contenu de la disquette : son directory (nom extension attribut nombre de secteurs utilisés pour chaque enregistrement) Le nombre de secteurs libres est fourni.

TINYDOS réduit le DOS à un simple remplacement des cassettes par les disquettes; il ne demande plus que 1,5 K RAM.

COPY copie d'un enregistrement d'un disque à l'autre ou sur un même disque

BACKUP copie l'intégralité d'un disque sur l'autre

COMPACT copie tous les enregistrements non effacés d'un disque sur l'autre

LOCAL permet la manipulation manuelle de la cassette en déshabilitant la commande remote de celui-ci

REMOTE à l'inverse du précédent ordre, rend à l'ordinateur le contrôle moteur de la cassette

AUTO permet la numérotation automatique des lignes de programmes Basic par pas de 10

CREATE crée un fichier dont on donne le nom et éventuellement la longueur

DELETE efface un fichier (n'en supprime que le nom dans le directory)

VERIFY vérifie la qualité d'un enregistrement, d'un disque ou des deux

IDISK initialise et formatte une nouvelle disquette

PROTECT protège un enregistrement contre l'écriture

RECOVER déprotège un enregistrement protégé contre l'écriture ou restaure celui dont le nom a été effacé (par DELETE)

RENAME change le nom d'un enregistrement

DSAVE sauvegarde une portion de mémoire comme W le fait sous moniteur (avec en plus la possibilité d'introduction d'une adresse de lancement de la routine autre que la première adresse. Celle-là ne pouvant servir que pour l'AUTOSTART).

DLOAD chargement de la portion mémoire (comme R sous moniteur, mais sans possibilité de décalage)

OPENI ouvre un fichier pour la lecture

ASSIGN DISK pour travailler avec les disques en lecture et écriture

ASSIGN CASSETTE pour travailler avec les cassettes en lecture et écriture

ASSIGN FROM DISK (ou CASSETTE) pour utiliser les disques (ou cassettes) en lecture

ASSIGN TO DISK (ou CASSETTE) pour utiliser les disques (ou cassettes) en écriture

ASSIGN OUTPUT TO SCREEN, PRINTER ou DISK pour assigner les ordres PRINT à l'écran, l'écran et l'imprimante, ou les disques

ASSIGN INPUT FROM KEYBOARD, RS232, ou DISK pour que les ordres INPUT ou GETC lisent le clavier, l'entrée RS232 ou le disque.

Le DOS 2.1 ajoute :

RREC lit un secteur sur disque et le place en mémoire à partir de l'adresse spécifiée (accès direct)

WREC écrit un secteur (accès direct)

ASSIGN DCR (je suppose) pour manipuler les DCR.

Ce jeu de commandes est extensible. J'ai ajouté par exemple la commande CLS - qui efface l'écran, mais non l'imprimante... -. De plus, vous pouvez, avec un peu d'habileté vous constituer une librairie de nouvelles commandes disponibles sur disque, chargées par un DLOAD ... et exécutées par les nouveaux ordres que vous avez introduits: SCOPY ou JUMBO pour un screen-copy (copie d'écran) en petit ou grand format,...

Toutes les commandes du DOS peuvent être introduites dans un programme Basic. Pour ce faire, vous devez, au moyen d'un POKE #131,3, les envoyer par un PRINT au DOS. Ceci est moins commode qu'une commande directement accessible en Basic, mais ne pose aucun problème de mise au point. Par exemple, pour que votre programme lise le directory de la disquette 1, vous commanderez :

```
10 POKE #131,3:PRINT "DIR1":POKE #131,1
```

Un des gros avantages d'une telle organisation des commandes est de permettre, pour chaque ordre, l'ouverture, la lecture,... l'accès à un fichier dont le nom se trouve dans une chaîne; ou à un secteur dont le numéro se trouve dans une variable,...

Le plus dur est d'ouvrir, écrire, et fermer un fichier séquentiel. Pour ce faire, vous devrez jongler avec des CHR$(1), CHR$(2) et CHR$(3). Avec un peu d'habitude, on y arrive... sans trop de peine!

De plus, le manuel d'utilisation vous montre comment manipuler les ordres du DOS depuis le moniteur directement, ou par programme machine. Toutes les adresses importantes vous sont fournies : ouverture de fichier, fermeture,...

Les sytèmes d'exploitation ne sont pas compatibles avec CP/M. Toutefois, le CP/M - ou plutôt un CP/M au formattage propre à Indata semble-t-il - est disponible pour le 160 K (il doit coûter environ 10000 francs TVAc). Un véritable CP/M, commençant à l'adresse 0 et acceptant le véritable formattage du CP/M - ou le DAI lira et écrira tous les disques écrits sous CP/M - est en préparation et sortira bientôt (?) pour le même prix. Il laissera à l'utilisateur environ 40 K RAM libres. Voilà qui est prometteur, et qui me rend impatient...

Ce que vous recevrez avec vos disques n'est pas lourd mais très intéressant. Il s'agit d'un manuel d'utilisation en anglais de plus de 50 pages. Je l'ai trouvé suffisamment explicite bien qu'entièrement en anglais. Il s'agit aussi de trois disquettes : l'une contenant une série de programmes de démonstration, une autre contenant une série de programmes d'initiation à la manipulation du DAI, et la dernière contenant le DOS, un programme OLDFMT qui vous permet de lire des disquettes écrites sous le DOS V1.0 des anciennes disquettes, un programme basic de menu général et une routine de transformation du clavier en AZERTY.

Le DOS V2.0 - ou version 2.1 - fonctionne très bien. Mais on peut regretter certaines options et la pauvreté de la gestion des fichiers.

En premier lieu, l'allocation n'est pas dynamique. Vos fichiers, tableaux ou programmes sont stockés de manière continue - séquentielle - sur disque. Chargez un programme par LOAD, ajoutez lui quelques lignes et essayez de le resauver, vous obtiendrez un malheureux "END OF FILE"... il faudra détruire l'ancien programme avant de tenter une nouvelle sauvegarde. On a vite rempli ses disques avec des programmes effacés... Il vous faudra donc régulièrement compacter vos disques pour récupérer toute cette place perdue!

En ce qui concerne les possibilités de gestion des fichiers séquentiels, on peut regretter la pauvreté du DOS, surtout lorsqu'on a déjà tâté un bon DOS comme le DOS 3.3 sur APPLE II. En effet, dès qu'un fichier est fermé pour l'écriture, il n'est plus possible de l'ouvrir à nouveau pour y ajouter d'autres noms. Il faudra le recopier dans un nouveau fichier que vous étendrez. Ce qui est lent pour de longs fichiers, et peu économique en place disque. Un fichier ne peut être ouvert que pour une écriture ou une lecture exclusivement. L'ouverture vous place systématiquement en début de fichier. Il n'est pas possible, sauf par programme - lent et ennuyeux - d'atteindre le n° enregistrement directement. Pas de POSIT ni d'APPEND.

Les fichiers à accès direct se résument à peu de chose. Vous pouvez lire ou écrire un secteur entier. C'est tout. Ce sont des PEEK et POKE qui vous placeront les données dans les variables adéquates. Vous devrez aussi tenir une table reprenant la liste des secteurs déjà occupés, une autre décrivant la forme des enregistrements : 6 caractères pour la date, 30 pour le nom, etc... Vous ne pouvez travailler que sur des blocs d'un secteur (256 octets), à moins d'une gymnastique telle celle développée pour le programme de comptabilité de l'école,

afin de lire et d'écrire des quarts de secteurs. Par bonheur, et c'est là un véritable avantage, toutes les pistes contiennent le même nombre de secteurs! S'il fallait encore calculer ce nombre de secteurs en fonction du numéro de piste!

Mais il y aura bientôt le DOS 3! Compatible CP/M, il travaillera en allocation dynamique. Les enregistrements ne seront donc plus séquentiels mais placés en morceaux là où il y a de la place! De plus, ce DOS occupant un peu moins de place en RAM que l'actuel sera relogeable partout en mémoire grâce à la commande MOVDOS - qui rappelle le MOVCPM -. Il reprendra toutes les facilités du DOS 2.1 en lui ajoutant les possibilités de positionnement à l'intérieur d'un fichier - sorte de POSIT -, la création de formats d'enregistrements - ou FIELD -, l'allongement d'anciens fichiers séquentiels - sorte d'APPEND -, la lecture et l'écriture dans le même fichier, l'ouverture simultanée de plusieurs fichiers,... au format CP/M. Ce DOS sera deux fois plus rapide que l'actuel, la vitesse de transfert étant de 17 KB/s. Il sera possible de connecter jusqu'à quatre unités de disques, et un utilitaire permettra de relire les disques écrits sous DOS 2.1. Etc... Je l'attends impatiemment, et ne suis pas le seul! Mais jusqu'à quand donc?

Après autant de notes, et avant de conclure cet article, j'adresse quelques conseils à ceux que l'utilisation du DOS actuel a rendus ou pourrait rendre perplexes... Aux autres, j'adresse mon salut!

DUPAGNE Yannick
Club DAI-Namur

NOTES :

L'ordre ASSIGN INPUT FROM RS232 pourrait permettre l'utilisation du clavier d'une machine à écrire Brother CE-50 en remplacement d'un vieux clavier rebondissant,... si l'entrée par l'RS232 acceptait les ordres du DOS. Hélas, il n'est rien.

La routine AZERTY prévue sur le disque contenant le DOS n'est pas utilisable telle quelle. En effet, son initialisation omet de modifier en mémoire les pointeurs en 2A7 et 2A8 vers la table des codes ASCII. Il faudra le faire vous même.

Des problèmes de lecture et d'écriture sur disque peuvent se poser si le système est branché sur une mauvaise prise... bien sûr! Le 640 K branché dans le bureau de l'économe de l'école, et là uniquement, fonctionne mal... une perte due à un autre appareil placé sur le même circuit en est responsable. Le DAI fut déplacé! Depuis lors les problèmes sont terminés.

J'ai aussi, par mégarde, placé la main depuis l'écran d'un moniteur Barco jusqu'au chassis d'un double drive. Cela m'a coûté une fameuse étincelle ... et envoyé un lecteur en réparation. Veillez à la qualité de votre terre, l'électricité statique ne pardonne pas.

L'AUTOSTART des programmes Basic fonctionne parfaitement. Mais le DOS oublie d'initialiser en 118 le "RUN FLAG". Attention donc si, dans un programme chargé et exécuté par AUTOSTART, vous commettez, lors de l'introduction d'une donnée, une erreur - exemple : vous répondez AA à un INPUT B ou vous frappez un O au lieu d'un zéro - l'ordinateur refusera la réponse en vous renvoyant un SYNTAX ERROR et en arrêtant le programme définitivement, sans vous donner de numéro de ligne. Commencez donc tout programme Basic chargé par AUTOSTART par POKE #118,#FF. Plus aucun problème ne se posera alors.

Lors de la commande DIR, d'affichage du directory d'une disquette, le défilement des noms ne peut pas être interrompu comme dans un listing Basic. Avec plus de vingt noms, il vaut mieux commencer par un POKE #FF05,#10 pour ralentir l'affichage!

Il est malvenu de tenter d'interrompre l'exécution d'une routine du DOS par un BREAK. Vous aurez du mal à récupérer le contrôle de votre système... Placez plutôt un disque - même vide - dans le lecteur démuni dont vous avez demandé le directory.

Enfin, si vous avez reçu votre système 320 K avec le mien, vous aurez remarqué que le DOS V2.0 fourni sur la disquette contenant le programme OLDFMT n'est pas "régulier". En ce qui concerne le TINYDOS en tout cas. Des problèmes de lecture de longs programmes se posent. Faites comme moi: demandez une copie du DOS 2.1 chez Indata, ou reprenez le DOS au départ du disque "démo". Dans ce cas, vous n'aurez aucun problème, si ce n'est que le nombre de secteurs libres renseigné par un DIR est trop grand de 640!

```
1     REM
2     REM Dezimal-Bruchumwandlung
3     REM
4     REM von Willi Herrmann / D-4320 HATTINGEN
5     REM
10    CLEAR 256:MODE 0:PRINT CHR$(12)
20    X%=0:Y%=0:X=1.0:A$="  "
30    INPUT "Dezimalzahl ";A:A=ABS(A)
40    Z%=INT(A):A=A-Z%:IF A=0.0 GOTO 110
50    B=1.0/A:D=B
60    IF FRAC(D)=0.0 GOTO 100
70    D=1.0/FRAC(D):X=X*D:IF B*X+0.5<101.0 GOTO 60
80    IF X<1000.0*X/D THEN A$="~ "
90    X=X/D
100   X%=INT(X+0.5):Y%=INT(B*X+0.5)
110   IF X%=1.0 AND Y%=1.0 THEN Z%=Z%+1:X%=0:Y%=0
115   PRINT TAB(30);" ";A$;Z%;"  ";X%;" /";Y%
120   PRINT :GOTO 20
```

```
Dezimalzahl ?0.75          0   3 / 4

Dezimalzahl ?1.0625        1   1 / 16

Dezimalzahl ?0.111111      0   1 / 9

Dezimalzahl ?5             5   0 / 0
```

KEN-DOS

Tessenderlo 02/04/84

Test report Ken-dos.

Here is - at last - the test report concerning the new Floppy-disc system designed by Kenneth Gooswit for the DAI computer.

That, designing the system, exchange of programs and data between existing peripherals and speed of the system has been the major factor is clear when connecting the system. The Hold-line of the CPU has to be connected whit the DCE-bus by means of a soldered wire. An EPROM-card has to be placed on the X-bus. These two actions are very clear described in the manual. The system has to be connected by means of the provided flat-cable whith the DCE-bus. The system is constructed in a way that, peripherals who where connected whith your computer, now simply can be connected whith the floppy-system. The DOS commando-set allows exchange between DCR, cassette and disk whith no restrictions. This applies to every Basic and machine language program whithout relocating or adapting heap-pointers. Besides the DCE-bus connection the system needs a mains-connection. It is recommended to place a mains-filter between this connection.

The used drives are of the Shugart type, density and num-ber of tracks (40 or 80) have no influence on the hardware when track to track steptimes are smaller or equal to 6 msec. The capacity of the system is completely to be defined by the user and can increase to 3,2 Mbyte when using four double-sided drives. A minor hardware modification is necessary to handle 8" drives, but only single density is possible.

The used disk's are 5"1/2 soft sectored. They are format-ted on 40 or 80 tracks with five sectors per track, in this way creating 200 or 400 kbyte. This is for one side, Ken-dos looks at a double drive as two single drives. The first three tracks are used for directory, leaving 185 or 385 kbyte for the user.

The disk operating system is completely stored in EPROM allowing the exchange of programs using the lower part of RAM starting at #2EC. Next to the operating system are sockets for placing five EPROMS, the capacity (2 kbyte to 16 kbyte)is chosen cutting or connecting jumpers. These EPROMS can hold (often used) programs who, by means of a simple command can be placed in their working area. The unit tested by me incorporated FWP. The time necessary to move the program was so short that it did not invite me in mesuring it. In these EPROM's can - later -the CP/M operating be placed.

The DOS command table is complete, and usable in comman mode as well in programs in the same way as it is for DCR. The table contains commands for loading and saving of programs in basic or UT. When displaying the directory you can, by pressing one key, LOAD or LOAD:RUN a program. The commands LOCK, UNLOOCK, PRT and CLR in conjunction whith the command code, enables you to protect disks or files against reading, writing and formatting. The command DCR enables the whole DCR command set to allow reading and writing to DCR. The command DISK performs a return to the disk command set. In the same way the command CAS enables reading and writing to cassette. In this manner every existing Basic or macine language program

can be copied on disk.

A testing report would not be complete when it doesn't give times about formatting and copying disk's. Formatting a disk whit a capacity of 200 kbyte needs, between pressing the return-key and the reappearing of the prompt 35 seconds. Copying a disk depends on the used space on it, but needs for a fully used disk the same time as for formatting, 35 seconds. Loading and saving times are only noticed when the programs are of a considerable length. An example : loading a picture in MODE 6 needs four (4) seconds. This opens the door for animation by means of floppy-disk. These times are all mesured without heaving the drive motors ready.

The system has the possibility to create and work with random files. The size of a record (1024 bytes) is a bit unhandy but it happens fast. On the other hand, it is very easy to divide the sector over a string array by means of a small ML routine so it is ready for direct use. Developing a database is therefore one of my next missions.

On the end I can only give you my impressions on the system and say something about prices. Three cheers for Ken, who realised the designing and development of the system. Hip hip hurrah, at last a solid floppy disk system for the DAI computer. The price is depending on the drives used and is variable between 44500 and 74900 Bfr. The term of delivery is yet still a problem. The waiting-list for purchasing a system is already enormous. The tested system generated no reading or writing errors during a period of six hours in witch it was swithed on. For more information and purchase contact :

For Belgium : MIKROSHOP HAGELAND
HERSELTSESTEENWEG 103
3220 AARSCHOT
BELGIUM
TEL : 016/56 87 70

All other countries : MIPI v.o.f.
P.O. BOX 40
1616 ZG HOOGKARSPEL
THE NETHERLANDS

Ther is no doubt about it, the DAI computer will take another high flight whith this system.

Couwberghs Frans.

KEN-DOS

Tessenderlo, 02/04/84

Testverslag Ken-dos.

Hier volgt dan eindelijk het testverslag over het nieuwe Floppy-disk systeem dat ontworpen is door Kenneth Gooswit voor de DAI-computer.

Dat bij het ontwerp van dit systeem de uitwisseling van programma's tussen de reeds bestaande opslagmedia en de snelheid van dit nieuwe medium centraal hebben gestaan werd me reeds bij het aansluiten van het systeem duidelijk. Het naar de DCE-bus brengen van de HOLD-line van de CPU door middel van een soldeerverbinding is de enige hardware verandering die aan Uw computer moet uitgevoerd worden naast het aanbrengen van het EPROM-board op de X-bus. Deze handelingen zijn in het manual uitvoerig beschreven. Verder wordt het systeem gewoon met de flat-cable waarmee het is uitgerust met de DCE-bus verbonden. De constructie is zodanig dat U apparaten die met de DCE-bus van Uw computer verbonden waren nu met het Floppy-systeem verbindt. De commando-set is zodanig dat uitwisseling tussen DCR, Floppy en cassette zonder meer mogelijk zijn. Dit geldt zonder enige uitzondering voor elk programma, geschreven in BASIC, machinetaal of Tiny-Pascal, zonder reloceren of de heap-pointers aan te passen. Naast de DCE-bus aansluiting benodigd het apparaat een net-aansluiting welke gebeurt met een stevige kabel welke rand en pen aarding toelaat. Het is aan te bevelen deze aansluiting te voorzien van een netfilter.

De te gebruiken drives zijn van het Shugart type, de densiteit en het aantal tracks (40 of 80) heeft geen invloed op de hardware zolang de track-to-track steptime kleiner of gelijk is aan 6 milliseconden. De capaciteit van het systeeem is dus door de gebruiker volledig zelf te bepalen en kan zelfs oplopen tot 3,2 Mbyte wanneer vier double-side drives gebruikt worden. Door een kleine hardware verandering kunnen ook 8 inch drives gebruikt worden, maar dan enkel in single density.

De gebruikte schijven zijn 5"1/2 inch soft sectored. Ze worden geformatteerd op 40 of 80 tracks met vijf sectors per track. Zodoende krijgt men 200 of 400 kbyte per schijf. Het moet wel gezegd worden dat dit per zijde is, Ken-dos ziet een dubbele drive als twee enkele. De eerste drie tracks worden gebruikt voor directory zodat per schijf 185 of 385 kbyte ter beschikking staat van de gebruiker.

Het disk operating system zit in Eprom om de uitwisseling van programmas die het Ram gedeelte vanaf #2EC gebruiken mogelijk te maken. Naast dit operating system is nog plaats vrij voor vijf EPROMs waarvan het type door jumpers te leggen of te verbreken kan gevarieerd worden van 2 kbyte to 16 kbyte. Hierin kunnen -door de gebruiker veel aangewende - programmas geplaats worden die door een simpel commando naar het werkgeheugen verplaatst worden. Bij het door mij getestte toestel was FWP aanwezig. De tijd benodigd om het te verplaatsen is practisch onbestaande en nodigt je niet uit om hem te meten. Hierin kan ook - later - het CP/M operating system gepaatst worden.

De commando tabel is compleet, en zowel bruikbaar in direct mode als in programmas op dezelfde manier als dit is voor de DCR commandos. De tabel bevat commandos voor het laden en wegschrijven van basic en machinetaal

programmas. Vanuit de directory kan door middel van een enkele toets te drukken een programma geladen of geladen en gerund worden. De commandos LOCK, UNLOCK, PRT en CLR welke samenwerken met het het commando CODE laten het beveiligen van schijven of afzonderlijke programmas toe tegen lezen, schrijven en formatteren. Het commando DCR schakelt de volledige DCR commando-set - die uitgebreid is met het commando DISK - in zodat lezen en schrijven van en naar DCR mogelijk is. Het commando DISK laat U toe terug te keren naar de disk commandoset. Evenzo laat het commamdo CAS communicatie met de audio-cassette mogelijk. Op deze manier kan elk reeds bestaand programma van DCR of cassette overgezet worden op schijf.

Een testverslag ware niet volledig als het geen tijden bevatte over het formatteren en copieren van een schijf. Het formatteren van een schijf met een capaciteit van 200 kbyte heeft tussen het drukken op de returntoets en het terug verschijnen van de cursor 35 seconden nodig. Een BACKUP is afhankelijk van de inhoud van de te copieren schijf maar heeft voor een volle schijf dezelfde tijd nodig als het formatteren, 35 seconden. Het laden van programmas is alleen merkbaar wanneer de lengte aanzienlijk is maar zal nooit meer tijd vergen dan vijf a zes seconden. Een voorbeeld : het laden van een tekening in MODE 6 vergt vier seconden. Dit opent de deur voor animatie door middel van floppy. Deze tijden zijn allemaal gemeten zonder de drive motors op voorhand op snelheid te laten komen.

Het systeem heeft de mogelijkheid om met random files te werken. De grootte van de ingelezen sector (1024 bytes) is wel een beetje onhandig maar gebeurt zeer snel. Verder is het niet moeilijk om deze sector te verdelen over een string array door middel van een kleine machinetaal routine zodat hij direct bruikbaar is. Het ontwikkelen van een Data-base is dan ook een van m'n volgende opdrachten.

Verder blijft er me niets over dan m'n algemene indrukken aan U over te maken en iets te zeggen over de kostprijs. Hoed af voor Ken, die het ontwerp en op punt stellen van dit systeem verwezenlijkt heeft. Hip hip hoera, eindelijk een degelijk floppy-systeem voor de DAI-computer. De prijs is afhankelijk van de gebruikte drives en schommelt tussen 44500 Bf en 74900 Bfr. De leveringstijden vormen echter een probleem en de wachtlijst voor de aanschaf van een systeem is nu reeds enorm. Het door mij geteste systeem vertoonde tijdens gebruik geen lees of schrijffouten na een periode van ongeveer zes uren ingeschakeld te zijn geweest. Voor nog meer informatie en aanschaf kan U terecht bij :

Voor Belgie : MIKROSHOP HAGELAND
HERSELTSESTEENWEG 103
3220 AARSCHOT
BELGIE
TEL : 016/56 87 70

Alle andere landen : MIPI v.o.f.
P.O. BOX 40
1616 ZG HOOGKARSPEL
THE NETHERLANDS

Het lijd geen twijfel dat de DAI-computer door middel van dit systeem een nieuwe hoge vlucht zal nemen.

Basic-variables in machine language program by C.D.Esveld

Sometimes it can be desirable to replace a part of a Basic program by machine language to do things faster.
In order to get access to the already defined Basic variables we have to find out where those variables are stored. This can be done by examining the symbol table, the start of which is pointed to by memory address 2A1/2A2. In the symbol table every variable used by the program is present with its name (in ASCII), its value and some other information (for array-variables the symbol table gives a pointer to the HEAP).
For example, if you want to use the variable VAR in machine code, look for the sequence 56 41 52 in the symbol table and mark the byte which is preceding the name. This byte, the so-called type/length byte, is 04 for floating point variables, 14 for integer variables and 2x for string variables with a length of x bytes. Subtract the address of the start of the symbol table (as indicated by 2A1/2A2) from the address of the type/length and you have the so-called offset for the variable VAR. Adding this offset value to #4000 gives you the reference number which is to be used by the ROM routine starting at #E95A. Load the reference number into the registers B and C with LXI B, then CALL #E95A and as a result the memory address of VAR is present in the registers H and L. The registers B and C now contain the offset of the next variable.

Of course there is the possibility to use the variable address directly but one cann't be sure that this address will always be the same as the operating system determines the location of the symbol table.

After you have found out your reference numbers you don't have in general the possibility to add new variable names to your Basic program unless you do the whole job again.

Next program is an illustration of the concept mentioned above and it also shows some mathematical features using the ROM routines which are available with RST 4. The name MACC stands for the mathematics accumulator, which is 4 bytes long.
Note that this program is not suitable to demonstrate gain in speed.
In the case you want to use array variables, things are a little more complicated but this article may be a good start to solve that problem.

BASIC-VARIABLES IN MACHINE LANGUAGE

```
001                     *THIS MACHINE LANGUAGE PROGRAM REPLACES
002                     *LINES 50, 60 AND 70 OF THE NEXT BASIC PROGRAM
003                     *     IMP INT
004                     *     10 A=6
005                     *     20 B=11
006                     *     30 C=2
007                     *     40 D=0
008                     *     50 FOR D=A TO A+B STEP C
009                     *     60 PRINT D
010                     *     70 NEXT D
011                     *     80 END
012                     *REPLACE THE MENTIONED LINES BY:
013                     *     50 CALLM #1000
014                     *
015                     *THIS PROGRAM IS AN APPLICATION OF ROM ROUTINES
016                     *WITHIN AN EXCISTING BASIC PROGRAM
017                     *
018                     *PROGRAMMER: C.D.ESVELD
019                     *
020                     *THE FOLLOWING ENTRYPOINTS ARE DERIVED FROM THE
021                     *DAI FIRMWARE MANUAL BY B.J.BOERRIGTER
022                     XICOMP  EQU   :C015     INTEGER COMPARE
023                     PINT    EQU   :DB53     PRINT INTEGER NUMBER
024                     CRLF    EQU   :DD5E     LINEFEED
025                     RARRN   EQU   :E95A     SEARCH IN SYMBOL TABLE
026                             ORG   :1000
027 1000 C5             ENTRY   PUSH  B         SAVE ALL REGISTERS
028 1001 D5                     PUSH  D
029 1002 E5                     PUSH  H
030 1003 F5                     PUSH  PSW
031 1004 014D10                 LXI   B,REFA    OFFSET VARIABLE A
032 1007 CD5AE9                 CALL  RARRN     LOOK FOR A IN SYMBOL TABLE
033 100A E5                     PUSH  H         SAVE ADDRESS OF A
034 100B E7                     RST   4
035 100C 0C                     DATA  :0C       COPY A IN MACC
036 100D CD5AE9                 CALL  RARRN     LOOK FOR B IN SYMBOL TABLE
037 1010 E7                     RST   4
038 1011 4E                     DATA  :4E       ADD B TO A IN MACC
039 1012 215510                 LXI   H,APLUSB
040 1015 E7                     RST   4
041 1016 0F                     DATA  :0F       COPY MACC IN APLUSB
042 1017 E1                     POP   H         ADDRESS OF A
043 1018 E7                     RST   4
044 1019 0C                     DATA  :0C       COPY A IN MACC
045 101A 015310                 LXI   B,REFD    OFFSET VAR. D
046 101D CD5AE9                 CALL  RARRN
047 1020 E5                     PUSH  H         SAVE ADDRESS OF D
048 1021 E7                     RST   4
049 1022 0F                     DATA  :0F       MACC --> D
050 1023 CD53DB         PRINTD  CALL  PINT      PRINT D
051 1026 CD5EDD                 CALL  CRLF      LINEFEED
052 1029 015110                 LXI   B,REFC    OFFSET VAR. C
053 102C CD5AE9                 CALL  RARRN
```

```
PAGE 02

054 102F E7                 RST   4
055 1030 4E                 DATA  :4E       ADD C TO CONTENT OF MACC
056 1031 E1                 POP   H         GET ADDRESS OF D
057 1032 E5                 PUSH  H         SAVE IT AGAIN
058 1033 E7                 RST   4
059 1034 0F                 DATA  :0F       MACC --> D
060 1035 215510             LXI   H,APLUSB
061 1038 E7                 RST   4
062 1039 0C                 DATA  :0C       A+B --> MACC
063 103A E1                 POP   H
064 103B E5                 PUSH  H
065 103C CD15C0             CALL  XICOMP    COMPARE A+B AND D
066 103F FA4710             JM    EXIT      RETURN IF D>A+B
067 1042 E7                 RST   4
068 1043 0C                 DATA  :0C       D --> MACC
069 1044 C32310             JMP   PRINTD
070 1047 E1          EXIT   POP   H
071 1048 F1                 POP   PSW
072 1049 E1                 POP   H
073 104A D1                 POP   D
074 104B C1                 POP   B
075 104C C9                 RET
076 104D 4002        REFA   DBL   :0240
077 104F 4009        REFB   DBL   :0940
078 1051 4010        REFC   DBL   :1040
079 1053 4017        REFD   DBL   :1740
080 1055             APLUSB RES   4
081 1059                    END

*****************************
* S Y M B O L   T A B L E *
*****************************

APLUSB 1055   CRLF   DD5E   ENTRY  1000   EXIT   1047
PINT   DB53   PRINTD 1023   RARRN  E95A   REFA   104D
REFB   104F   REFC   1051   REFD   1053   XICOMP C015


]#P.
1000 C5 D5 E5 F5 01 4D 10 CD 5A E9 E5 E7 0C CD 5A E9
1010 E7 4E 21 55 10 E7 0F E1 E7 0C 01 53 10 CD 5A E9
1020 E5 E7 0F CD 53 DB CD 5E DD 01 51 10 CD 5A E9 E7
1030 4E E1 E5 E7 0F 21 55 10 E7 0C E1 E5 CD 15 C0 FA
1040 47 10 E7 0C C3 23 10 E1 F1 E1 D1 C1 C9 40 02 40
1050 09 40 10 40 17
```

```
*****************************
* S Y M B O L   T A B L E *
*****************************          cont.. ERASE TO END OF SCREEN

BLANK  B311   EXIT   B32A   OUTC   DD60


]#P.
B300 F5 C5 D5 E5 21 75 00 36 20 EF 0C E5 1C 7B 95 47
B310 4C 3E 20 CD 60 DD 05 C2 11 B3 4F FE 01 CA 2A B3
B320 0D 3E 0D CD 60 DD 43 C3 11 B3 E1 EF 09 21 75 00
B330 36 5F E1 D1 C1 F1 C9
```

Erase to end of screen

Some time ago I was adapting a TRS-80 program which contained the instruction PRINT CHR$(31). The aim of this instruction is to clear the screen starting from the current position of the cursor up to the end of the screen. As my program already contained a machine language part I decided to write a subroutine in machine language to do the same. The routine was stored at the end of the RAM but now there was a problem in using the Dainamic Bootstrap Loader V2 (see Dainamic 12, 283). The DBL expects the Basic program after the machine language part, so I made the following changes in DBL:
Substitute the content of address 3CD to 50 and the next address to B3. Put zeros in the addresses 3DC, 3DD and 3DE and everything is right.

C.D.Esveld

```
001                     OUTC    EQU   :DD60
002                             ORG   :B300
003 B300 F5                     PUSH  PSW
004 B301 C5                     PUSH  B
005 B302 D5                     PUSH  D
006 B303 E5                     PUSH  H
007 B304 217500                 LXI   H,:75
008 B307 3620                   MVI   M,:20     BLANK CURSOR
009 B309 EF                     RST   5
010 B30A 0C                     DATA  :0C       ASK CURSOR POSITION
011 B30B E5                     PUSH  H         SAVE X,Y COORD.
012 B30C 1C                     INR   E
013 B30D 7B                     MOV   A,E       XMAX (#3C) IN REG. A
014 B30E 95                     SUB   L         SUBTRACT VALUE OF X
015 B30F 47                     MOV   B,A       NUMBER OF BLANKS
016 B310 4C                     MOV   C,H       NUMBER OF LINES
017 B311 3E20           BLANK   MVI   A,:20
018 B313 CD60DD                 CALL  OUTC
019 B316 05                     DCR   B
020 B317 C211B3                 JNZ   BLANK
021 B31A 4F                     MOV   C,A
022 B31B FE01                   CPI   :01
023 B31D CA2AB3                 JZ    EXIT
024 B320 0D                     DCR   C
025 B321 3E0D                   MVI   A,:0D     LINEFEED
026 B323 CD60DD                 CALL  OUTC
027 B326 43                     MOV   B,E
028 B327 C311B3                 JMP   BLANK
029 B32A E1             EXIT    POP   H
030 B32B EF                     RST   5
031 B32C 09                     DATA  :09       SET CURSOR POSITION
032 B32D 217500                 LXI   H,:75
033 B330 365F                   MVI   M,:5F     NORMAL CURSOR
034 B332 E1                     POP   H
035 B333 D1                     POP   D
036 B334 C1                     POP   B
037 B335 F1                     POP   PSW
038 B336 C9                     RET
039 B337                        END
```

TEXT IN DATA GENERATOR

'TEXT IN DATA' GENERATOR

Use:

This program has the possibility to create datastatements without typing linenumbers and the word 'DATA'.
You may start at any linenumber you want. Already present lines with the same linenumber will be erased !
By changing the statement "DATA" in line 710 by the statement 'REM', the program can be used as a REM-statement generator.
The 3 last typed in textlines remain visible in a coloured window. Textlines longer than 60 characters are truncated to 60 characters.
The input is stopped with 'BREAK'. Now all datalines can be seperated from the program by means of the editor. Then they can be merged to any other program.

Description:

After the startmenu (line 200), the 3 last typed in textlines are printed on the screen. These lines are stored in T1$,T2$ en T3$. After the input of a new textline, the contents of these 3 strings is 'moved'.
From line 500, te new textline is assembled from single character-inputs. The linenumber is converted to a string and the keyboard is set for lower-case characters.
To prevent problems with string handling in other programs, an empty textstring is changed into TEKST$=" ".
From line 700, conversion of the textstring to a data-statement occurs. Linenumber en 'DATA' are added to the textstring. By means of VARPTR, the start of the textstring is found and stored in #00F4/F5. The length of the textstring is stored in #00F6 (free RAM locations).
By means of a machine language routine, the conversion to a BASIC DATA-statement is done. Now the BASIC line will be added to the program, complete with linenumber.

The m.l.routine, which is moved into the stack-RAM, is as follows:

```
      PUSH ALL
      MVI A,:01
      STA   :0135     Inputsource is a string
      XRA A
      STA   :0118     Clear run-flag to prevent break on
                      an error-message
      MOV C,A         Clear line count
      LHLD  :00F4     Get start of text line
      SHLD  :0132     into pointer for encoding a textstring
      LDA   :00F6     Get stringlength
      STA   :0134     Load encoding counter
      CALL  :C918     Encode string to a statement
      XRA A
      STA   :0135     Inputsource is keyboard
      DCR A
      STA   :0118     Set runflag
      POP ALL
      RET
```

Jan Boerrigter - Jan.1984

```
1     REM ##################################################
2     REM ####                                          ####
3     REM ####        "TEXT IN DATA" GENERATOR          ####
4     REM ####                                          ####
5     REM #### auto-line number + auto-data statement ####
6     REM ####                                          ####
7     REM #### V1.0 - (C) - B.J.Boerrigter - Dec.1983 ####
8     REM ####                                          ####
9     REM ##################################################
50    PRINT CHR$(12):MODE 0:COLORT 8 0 0 0:CLEAR 5000
60    GOSUB 1000
70    LINE=30000:FIRST=1:T1$=" ":T2$=" ":T3$=" "
80    FOR I=1 TO 60:CLR$=CLR$+" ":NEXT
100   PRINT CHR$(12):POKE #BDD6,#C5:CURSOR 13,18
101   PRINT "###  T E X T   I N   D A T A  ###"
102   POKE #BC44,#C8:CURSOR 15,15
110   PRINT "Do not use ";CHR$(#22);"double quotes";CHR$(#22);" !!"
114   PRINT :IF LEN(TEKST$)>=60 GOTO 200
120   CURSOR 12,13:PRINT "Comma's seperate data-statements !!"
130   IF FIRST=0 GOTO 180
140   CURSOR 15,8:PRINT "First line number = ";LINE
150   CURSOR 15,6:INPUT "Change       < Y/N > ";VER$
160   IF VER$="Y" OR VER$="y" THEN CURSOR 15,4:INPUT "First line number = ";LINE
170   IF FIRST=1 THEN FIRST=0:GOTO 100
180   CURSOR 12,5:PRINT "Type textline (max. 60 characters):"
190   POKE #B920,#C5:POKE #B78E,#C8
200   CURSOR 0,10:PRINT T1$+SPC(60-LEN(T1$))
202   PRINT T2$+SPC(60-LEN(T2$))
204   PRINT T3$+SPC(60-LEN(T3$))
206   T1$=T2$:T2$=T3$
208   CURSOR 0,3:PRINT CLR$:PRINT CLR$:PRINT CLR$:CURSOR 0,3
210   LINE$=LEFT$(STR$(LINE),LEN(STR$(LINE))-2)
216   TEKST$="":POKE #2C3,#FF
218   CH=GETC:IF CH=0 GOTO 218
219   IF CH=13 GOTO 223
220   CH$=CHR$(CH):TEKST$=TEKST$+CH$:PRINT CH$;:GOTO 218
223   IF LEN(TEKST$)>=60 GOTO 208
224   IF TEKST$="" THEN TEKST$=CHR$(34)+" "+CHR$(34)
225   T3$=TEKST$
230   TEKST$=LINE$+"DATA"+TEKST$+CHR$(13)
240   PTR=VARPTR(TEKST$)
250   PTR=PEEK(PTR)+PEEK(PTR+1)*256:LGTH=PEEK(PTR)
255   PTR=PTR+1
260   POKE #F4,PTR IAND 255:POKE #F5,PTR SHR 8
270   POKE #F6,LGTH
280   CALLM #F800
300   LINE=LINE+1
310   GOTO 200

1000  RESTORE:FOR I=0 TO 41:READ J:POKE #F800+I,J:NEXT:RETURN
1010  DATA #F5,#E5,#D5,#C5, #3E,#01,#32,#35, #01,#AF,#32,#18
1020  DATA #01,#4F,#2A,#F4, #00,#22,#32,#01, #3A,#F6,#00,#32
1030  DATA #34,#01,#CD,#18, #C9,#AF,#32,#35, #01,#3D,#32,#18
1040  DATA #01,#C1,#D1,#E1, #F1,#C9

2000  REM This data-statement generator has the possibility to create
2001  REM DATA-statements without typing line numbers and the command
2002  REM 'DATA'.
2003  REM You may start with any line number. Evt. already
2004  REM existing identical line numbers are deleted.
2005  REM By changing in line 230 'DATA' into 'REM', this program
2006  REM can be used to generate REM-statements.
2007  REM The input can be stopped with 'BREAK'. All produced lines
2008  REM can now be seperated via the editor in order to merge them
2009  REM with other programs.
2010  REM The 3 last input lines are made visible in the
2011  REM coloured window.
```

DAInamic 84-21 93

P R O G R A M M E E R T E C H N I E K E N

Deze keer wilde ik wat vertellen over het gebruik van programma's op de DAI en het programmeren op de DAI om bepaalde persoonlijke problemen met behulp van de computer te kunnen oplossen. Eerst het gebruik van programma's; hiervoor wilde ik niet een klein programma nemen maar een echt groot programma, waar veel over te vertellen valt. In de bibliotheek van DAInamic is hiervoor een ruime keus. Programma's in deze catagorie zijn bijvoorbeeld de assemblers DNA en SPL, compilers zoals PASCAL en FORTH (is er wel maar wordt niet geleverd) de tekst in grafische modes met FGT, FFGT en SFGT en CHARACTER GENERATOR , grafische hulpprogramma's zoals GRAFIC TABLET, C.L.I.O. en binnenkort SIMPLE DRAWING en vanzelfsprekend de wordprocessors WP , DAINATEXT EN FWP.

Over deze laatste nu wilde ik hier schrijven. Als U de aankondiging van FWP hebt gelezen, bent U mogelijkerwijs nog niet overtuigd van de noodzaak om FWP aan te schaffen. Zeker als U al in het bezit bent van DAINATEXT die U destijds verkreeg door een update van WP. U krijgt misschien de gedachte: Ik laat DAINATEXT nu vervangen door FWP en dan heb ik er een paar kleine voordeeltjes bij en over een paar maanden komt SFWP uit die dan weer een Super versie is van FWP en zo blijven we aan de gang. Nu is dit iets waarvan ik van harte hoop dat dat inderdaad het geval zal zijn. Steeds door gaan en steeds betere programma's lijkt mij geen nadeel. Daarbij is het argument, dat er straks nog een beter programma is, misschien wel waar maar dat programma is er nog niet. Auto's van 1995 zijn waarschijnlijk een stuk zuiniger en stiller dan de huidige, maar dat is toch geen argument om nu maar te gaan fietsen ?

Genoeg hierover, ik wil het werken met FWP bespreken. Dat FWP prettiger in het gebruik is, is ook voor degenen die het niet aanschaffen duidelijk merkbaar.Ja zelfs die mensen profiteren van het werk van Ger Gruiters . Ze kunnen dat zien door bv mijn artikelen tot en met DAInamic nummer 18 te bekijken, die alle mbv DAINATEXT tot stand kwamen en dan in DAInamic 19 & 20 kijken, waar uittestversies van FWP gebruikt zijn en tenslotte naar dit artikel dat met de definitieve versie is gemaakt. Het zal U dan opvallen dat de regels aan de rechterkant gelijk zijn. Dit was theoretisch in DAINATEXT ook mogelijk maar steeds als ik het trachtte te gebruiken, bleek er bij het afdrukken weer iets fout te zijn gegaan. Zie artikelen waarbij soms een letter op de volgende regel kwam. De oorzaak is te zoeken in de regellengte die ook voor de printer werd vastgelegd is mij eens verteld. Een linkerkantlijn de zogenaamde marge was een verschrikking in DAINATEXT. Zei U een marge van 8 te willen hebben bij een regellengte van 78 dan kon U kiezen tussen of de laatste 8 letters op de volgende regel of geen waaarschuwing voor regeleinde en dat zelf bijhouden. De regels allemaal evenlang maken ging dan ook niet meer. Misschien trap ik met deze kritiek sommige mensen op het hart, daar de fout bij mij lag en ik DAINATEXT blijkbaar niet goed kon gebruiken, maar als ik met de veel snellere FWP niet zulke fouten maak, is voor mij het voordeel evident. Een tweede voordeel van FWP boven DAINATEXT is de veel grotere gebruikersvriendelijkheid. Ik zat mij altijd knap te ergeren aan de tientallen vragen, die je moet beantwoorden bij DAINATEXT. 'wilt U echt afdrukken' ; 'staat de printer aan' ; 'wilt U een marge' ; 'hoe groot moet de marge zijn' ; 'wilt U een nieuwe pagina' ; 'hoeveel regels per pagina' ; 'wilt U bladnummering' ; 'welk nummer om te beginnen' enz. enz. enz. Bij FWP is dit gelukkig op een veel betere manier opgelost . In een zogenaamde defaultmenu worden alle gewenste parameters bijgehouden. Verandert men een van deze waarden zal de werking van FWP vanaf dat moment eventueel (hoeft niet u kunt iets wijzigen wat U toch niet gebruikt) gewijzigd zijn. Het reeds aanwezige bestand blijft ongewijzigd !!!! En zo hoort het ook. U zet de parameters op de manier zoals U dat wenst en als U dan bv wilt printen behoeft alleen nog maar P ingedrukt te worden plus een space om vergisingen te voorkomen. Derde voordeel: slaat de machine op reset, dan zal in bijna alle gevallen het gehele bestand nog aanwezig zijn. (Een netstoring van enige seconden of langer ver-

nietigd wel alles.) Vierde voordeel: het programma is bijzonder geheugenefficient; het standaard bestand (buffer 1) is ruim 24K groot (ongeveer 5 vellen A4 zeer dicht beschreven, in de praktijk door blanco regels, marge, alinea's ed wel tot 10 vellen). Daarnaast is er nog een tweede buffer van ook nog eens 4K. Dus veel meer dan DAINATEXT ooit aankan. Ook de snelheid om een en ander weg te schrijven resp. in te lezen is veel groter. Al deze voordelen zijn voor sommigen misschien al redenen genoeg om FWP aan te schaffen, maar er is nog veel meer. Omdat FWP volledig in machinetaal is geschreven is de snelheid ten opzichte van de vorige tekstverwerkers enorm. Tevens is het nu ook straffeloos mogelijk vanuit FWP naar BASIC te gaan en daar iets uit te rekenen, indien gewenst in een programma, om daarna FWP weer te vervolgen. De ruimte in BASIC bedraagt 8K dus is MODE 4 net mogelijk. De groottes van de buffers en BASIC zijn theoretisch aan te passen (contact met Ger Gruiters opnemen), maar naar mijn ervaring en die van andere testers uitstekend gekozen. De variabelen van het BASIC-programma kunnen indien gewenst direct gebruikt worden in de tekst. Ger Gruiters heeft voor een voorbeeldprogramma gezorgd. Intikken van ^A^ in de tekst zal tot gevolg hebben dat bij uitprinten ^A^ automatisch wordt vervangen door de waarde van de variabele A uit het BASICprogramma. Daar ik deze faciliteit hier niet gebruik blijft er gewoon ^A^ staan. Anders was er toch wel een tweede manier geweest om ^A^ letterlijk op te nemen in de tekst. Er is namelijk een mogelijkheid om een ASCII-code vlak voor afdrukken te vervangen door een andere, hiermee kunnen we bv ^ toch op de printer krijgen door bv @ in te tikken als ASCII 40 en die te laten vervangen door ^. Zo kunnen we ook de controlebytes die FWP zelf gebruikt toch laten afdrukken.

Een vreselijk groot voordeel voor mij als schrijver van deze artikelen en als docent computerkunde is de mogelijkheid om een programma in BASIC in FWP te plaatsen en dan voor afdrukken te verfraaien. Een voorbeeld staat in dit artikel. Om dit te doen moeten we eerst naar BASIC. Hier geven we EDIT na en CLEAR 2500, vervolgens [break],[break] en dan UT. In utility kijken we met DA2 A5 (of SA2 en meerdere spaties) naar de adressen van de EDITbuffer. Er zal meestal iets staan als 02 A0 38 A1 wat betekent dat de EDITbuffer van A002 tot A138 loopt. We verplaatsen de EDITbuffer van BASIC dan naar een buffer van FWP met MA002 A138 3000 als we naar buffer 1 willen en MA002 A138 9000 als we naar de andere buffer willen. De eventuele inhoud van die buffer wordt wel vernield, maar daarom is het handig om het juist in buffer 2 te plaatsen zodat U het later gemakkelijk in buffer 1 op de gewenste plaats kunt invoegen. Slimme programmeurs kunnen het ook direct op de juiste plaats zetten, maar kunnen dat zelf wel uitvinden. Grote programma's zullen waarschijnlijk in aparte delen naar FWP getransporteerd moeten worden. De 'listing' van een programma kunnen we dan op allerlei manieren aanpassen aan onze wensen. Deze wensen kunnen bv zijn: inspringen bij subroutines, regels overslaan ter verduidelijking van de structuur van het programma , opmerkingen bij het programma dus niet in het programma, lange regels niet aan begin van regel dus tussen de regelnummers laten beginnen en niet afbreken midden in statement, marges en paginanummering en vele andere mogelijkheden. Voor mij wordt alles netter en voor lezers van DAInamic prettiger omdat de intikfouten die ik wel eens maakte bij het overnemen van een BASIC-programma nu niet meer kunnen voorkomen. Omgekeerd gaat het ook: we tikken een programma in FWP in, gaan naar BASIC en tikken POKE #135,2. Staan er lege regels of teksten in FWP is dit niet erg, U krijgt alleen een hele rij 'SYNTAX ERROR'-s op het scherm. Maar nog steeds zijn de mogelijkheden die FWP biedt niet uitgeput. We kunnen namelijk FWP ook gebruiken voor een data-base. We tikken de gegevens, die we willen bewaren,in in buffer 1 en gaan die gegevens aanpassen aan de eisen, die wij er aan stellen. Zo kunnen we bv gaan sorteren. Om alle misverstanden te voorkomen; dit wordt niet door FWP gedaan maar kunt U zelf vrij eenvoudig doen met een BASIC-programma dat buffer 1 gebruikt als veld dat bewerkt moet worden. Een voorbeeld dat ik nu niet verder zal uitwerken, maar dat een en ander wel zal verduidelijken. Ik wil een bestand hebben van al mijn platen. Ik tik daartoe al mijn platen mbv FWP in en plaats de gegevens aldus in ASCII-vorm tussen #3000 en #9000. Als ik met een

nieuwe plaat begin zet ik een marker, waarvoor ik een niet door FWP gebruikte ASCII-code onder de 20 neem of een teken dat ik toch nergens gebruik zoals # of]. Eveneens met behulp van dit soort tekens kan ik de gegevens zoals plaat-titel, artiest, genre, datum, e.d. scheiden. Ben ik klaar met invoeren van de gegevens dan kan ik een basicprogramma nemen (schijven) dat de gegevens sor-teert. We lezen de gegevens dan met PEEK en schrijven met POKE of mbv kleine machinetaalroutines voor verplaatsen van records. (in beide betekenissen !) Deze machinetaalroutines kunnen handig geplaatst worden in buffer 2. Maar nu kom ik toch aan het tweede deel van dit artikel waarin ik wilde laten zien hoe ik een praktisch probleem, dat vrijwel iedereen weleens heeft meegemaakt, op een voor mij veel minder vermoeiende manier heb aangepakt. Dat de DAI hierbij behulpzaam was zult U al vermoeden.

Ik kwam in de gelukkige omstandigheid dat ik een eigen, hobbykamer in ons huis kon gaan inrichten. In deze kamer zouden echter verschillende meubelen ge-plaatst dienen te worden. Deze meubelen hebben vanzelfsprekend bepaalde afme-tingen en het zou passen en meten worden om alles wat ik in de kamer hebben wilde, ook daadwerkelijk op een bruikbare manier te plaatsen. Ik had er echter geen zin in om steeds iets anders te gaan proberen door met zware meubelen heen en weer te lopen zeulen. Ik zocht en vond een oplossing en wel een die lichamelijk een stuk minder vermoeiend is.

Ik maakte een programmaatje waarmee ik mijn meubelen kon verplaatsen in de ka-mer. De werking van het programma zal ik aan de hand van de listing uitleggen. Tevens heb ik de listing gegeven in een vorm die erg handig is bij gebruik in het onderwijs. Ik geef er veel commentaar bij ,dat in de leerfase erg nuttig kan zijn, maar voor geoefende programmeurs veel te uitgebreid is. Dus zal ik deze opmerkingen ook niet in REM's geplaatst willen zien. We kunnen het be-schouwen als het een samenvatting van de mondelinge toelichting, die ik er bij zou hebben gegeven.

kamerindeling

```
10      REM KAMERINDELING / F.H. DRUIJFF - 19840402
            Identificatie van het programma
20      MODE 6:COLORG 8 14 9 14:CLEAR 3000:DIM X(23),Y(23):GOTO 900
            Deel van initialisatie van het programma.
            De CLEAR moet hier staan, want die mag nooit in een subroutine
            geplaatst worden. Als dat toch gebeurt 'verliest' de DAI zijn
            correcte terugkeeradres. Meestal zal programma dan 'crashen'.
30      XL=X+L:YB=Y+B:DRAW X,Y XL,Y K:DRAW XL,Y XL,YB K
40      DRAW XL,YB X,YB K:DRAW X,YB X,Y K:RETURN
            Het tekenen van de rechthoek.
            Voor de snelheid zetten we deze routine voor in ht programma.
            Merk op dat voor snelheidswinst eerst XL en YB worden berekend.
60      L=L*255/294:B=B*255/294:IF F=1 THEN R=L:L=B:B=R:F=0
            In de juiste schaal zetten van de gegevens en als de vlag gezet
            is (F=1) Lengte(=L) en Breedte(=B) verwisselen.
70      GOSUB 30
80      K=17:GOSUB 30:H=GETC:IF H=0 GOTO 80:K=16:GOSUB 30
            Tekenen van object en wissen indien er actie volgt.
90      IF H<24 THEN X=X+X(H):Y=Y+Y(H):GOTO 80
            Veranderen van de coordinaten van ons basispunt.
100     K=17:H$=CHR$(H):IF H$>"Z" THEN H=H-32:F=1:H$=CHR$(H)
            Toetsaanslag omzetten in tekst. Sneller en overzichtelijker.
            Vlag zetten als SHIFT werd gebruikt.
            Elke toetsaanslag automatisch in hoofdletters zetten.
```

voorwerpen

```
200     IF H$="B" THEN L=170:B=85:GOTO 60
210     IF H$="L" THEN L=71:B=46:GOTO 60
220     IF H$="S" THEN L=195:B=49:GOTO 60
230     IF H$="H" THEN L=97:B=43:GOTO 60
240     IF H$="C" THEN L=65:B=58:GOTO 60
250     IF H$="K" THEN L=51:B=58:GOTO 60
260     IF H$="V" THEN L=375:B=294:GOTO 60
270     IF H$=" " THEN K=19:GOSUB 30:X=1:Y=1:L=0:B=0:GOTO 80
280     IF H$="W" THEN K=23:GOSUB 30:K=18:GOSUB 30
290     GOTO 80
```

initialisatie

```
900     X(18)=-1:X(19)=1:X(22)=-10:X(23)=10
910     Y(16)=1:Y(17)=-1:Y(20)=10:Y(21)=-10:GOTO 80
```

Dit is het vullen van de array's die voor de verplaatsing zorgen.

Zoals U ziet is het programma met FWP al van verschillende aanwijzingen voorzien. Regels zonder regelnummer vanzelfsprekend niet intikken. En natuurlijk voor intikken eerst IMPINT geven. De werking van het programma is misschien al duidelijk, maar juist omdat iedereen, die dit programma zal willen gebruiken er een eigen versie van zal maken, is het beter nog enig commentaar te geven. Na de initialisatie (regels 20, 900 en 910) staat in de listing op 30 en 40 het plaatsen van het gekozen voorwerp. Dit staat voor in het programma om zoveel mogelijk snelheid te krijgen. Om dezelfde reden worden ook XL en YB uitgerekend. Het tekenen van het gekozen voorwerp gebeurt met behulp van de set kleuren 16 t/m 19. Hierdoor is het mogelijk voorwerpen door andere reeds geplaatste heen te bewegen zonder iets te beschadigen. Leest mijn artikel daarover nogeens door als het aan de hand van het gebruik hier niet geheel duidelijk is. Het programma is een typisch voorbeeld van een 'custom'made programma. Ik bedoel hiermee dat er nog geen controles/beschermingen/uitleg inzitten. Als U buiten het beeld gaat krijgt U vanzelf een foutmelding. Maar als we dan met MODE 6:RUN vervolgen kunt U probleemloos doorgaan. Om een voorwerp te wissen eerst het gekozen voorwerp op de te wissen positie zetten, daarna W intikken en U kunt het naar willekeur verplaatsen of weer vervangen door een ander voorwerp. Veranderen van voorwerp is net zo simpel als het kiezen van ervan : Tik gewoon de letter in die bij dat voorwerp hoort. Ik koos voor mijn situatie B-bureau, L-ladenkast, S-schuifdeurenkast, H-hoge kast, C-computer, K-kluisje, V-vertrek. De laatste is nodig omdat ik ook hier de subroutine wilde gebruiken Cursor start in (0,0), we moeten dan direct het vertrek plaatsen (V-[ ]) dan zal de cursor vervolgens steeds op (1,1) terugkomen. Het plaatsen van een voorwerp doen we met de spatiebalk. Willen we een ander terugkeerpunt voor de cursor of een grotere cursor kunnen we dat in regel 270 (spatie) aanpassen. Bv met X=100:Y=80:L=2:B=2. De lengte en breedte kunnen gewisseld worden door de passende letter met SHIFT in te drukken. Ik hoop dat ik hiermee voldoende inzicht heb gegeven in de werking van het programma en het daarmee voor U tot een spierpijnvoorkomend hulpprogramma heb kunnen maken.

Frank H. Druijff

P.S. De cursorbewegingen van FWP kunnen op geschikte machines versneld worden door op #0EC2, #1257 en #137F de #BE te vervangen door #BB. Probeer zelf of uw machine geschikt is.

```
2     REM
10    REM ***********************************************
20    REM ***      PRINT   USING   SUR  DAI  48K      *****
30    REM ***      format defini par l'utilisateur    *****
40    REM ***      (c) ALAIN MARIATTE    &   L'O.I.   *****
50    REM ***********************************************
52    REM
54    REM PRINT USING
56    REM
60    REM Le format est defini par l'utilisateur dans la chaine USING$.
62    REM Le nombre de chiffres a gauche du point est celui de la partie
63    REM entiere non-signee (ne pas compter la place d'un signe eventuel).
65    REM La valeur a ecrire doit etre  dans la variable VALEUR (virg.flott
.)
70    REM GOSUB 1000 formate & affiche le nombre a l'endroit du curseur cou
rant
72    REM en cadrant a droite et SANS effectuer de saut a la ligne.
74    REM On peut ainsi faire plusieurs USING's sur une meme ligne.
76    REM Le format de la partie entiere est verifie.S'il est trop grand pa
r
78    REM rapport au format defini,la mention ** BAD USING ** apparait.
80    REM Les zeros non-significatifs (cadrage part. entiere) sont notes "o
".
82    REM La partie decimale est tronquee au format-utilisateur defini.
84    REM Un format-utilisateur SANS point decimal ecrit la valeur entiere
85    REM suivie d'un point.
86    REM Avec le programme DOUBLE-PRECISION,modifier la ligne 2047:
88    REM 2047  IF LG%<=12 THEN ZERO$="0"
90    REM
100   REM ***************** DEMO ********************
110   REM
120   PRINT CHR$(12):PRINT
130   INPUT "DEFINISSEZ LE FORMAT USING (ex:####.##):";USING$:PRINT
140   INPUT "VALEUR A ECRIRE:";VALEUR:PRINT
150   FOR I%=1 TO 30:PRINT CHR$(137);:NEXT
160   GOSUB 1000:REM appel routine print using
170   PRINT :REM force le saut de ligne apres le print using
180   GOTO 140
190   REM
900   REM ************** S/P PRINT USING *************
910   REM
1000  L0%=LEN(USING$)
1010  P%=-1:REM P%=position point decimal dans le format
1020  PP%=-1:REM PP%=position point decimal dans valeur
1030  PE%=-1:REM PE%=position d'un eventuel Exposant
1032  REM on cherche la position du point dans le format
1035  FOR I%=0 TO L0%-1
1040  IF MID$(USING$,I%,1)="." THEN P%=I%
1050  NEXT
1060  IF P%=(-1) THEN P%=L0%+1:GOTO 1080:REM integer
1070  ND%=L0%-P%-1:REM nombre de decimales dans le format
1075  REM analyse de la valeur a cadrer
1080  USE$=STR$(VALEUR):L%=LEN(USE$)
1090  FOR I%=0 TO L%-1
1095  IF MID$(USE$,I%,1)="E" THEN PE%=I%:REM reperage de l'exposant
1100  IF MID$(USE$,I%,1)="." THEN PP%=I%:REM pos.point dec.
1110  NEXT
1115  IF PE%<>(-1.0) THEN 2000:REM traitement nbre a exposant
1120  IF PP%=(-1.0) THEN PP%=L%:REM integer
1125  IF PP%>P%+1.0 THEN PRINT "** Bad Using **":GOTO 1160
1130  PRINT SPC(P%-PP%+1);:REM cadrage a droite
1140  PRINT LEFT$(USE$,PP%);:REM ecriture partie entiere
1145  IF PP%=L% OR FRAC(VALEUR)=0.0 THEN 1160:REM inutile d'ecrire .0
```

```
1150  PRINT ".";
1152  DEC$=RIGHT$(USE$,L%-PP%-1):REM decimales a ecrire
1154  IF LEN(DEC$)<=ND% THEN PRINT DEC$;:GOTO 1160:REM nbre dec.< au format
1156  PRINT MID$(USE$,PP%+1,ND%);:REM ecriture des decimales
1160  RETURN
1900  REM ---------- Traitement Exposants positifs -----
2000  IF MID$(USE$,PE%+1,1)="-" THEN 3000:REM cas Ex.negatif
2010  GAUCHE$=LEFT$(USE$,PE%):LG%=LEN(GAUCHE$):REM partie a gauche de l'Exp
.
2020  IF MID$(USE$,2,1)="." THEN G$=LEFT$(GAUCHE$,2)+RIGHT$(GAUCHE$,LG%-PP%
-1):GAUCHE$=G$
2022  REM on a supprime le point decimal de la chaine GAUCHE$
2030  LG%=LEN(GAUCHE$):EX$=RIGHT$(USE$,L%-PE%-1):EX%=VAL(EX$):REM valeur de
 l'Exposant
2040  ZERO%=EX%+2-LG%:REM nbre de zero a ajouter
2042  IF ZERO%=0 THEN 2060:REM pas de zero a ajouter
2045  ZERO$="o":REM cas zeros non significatifs
2047  IF LG%<=6 THEN ZERO$="0"
2050  FOR I%=1 TO ZERO%:GAUCHE$=GAUCHE$+ZERO$:NEXT
2060  USE$=GAUCHE$:L%=LEN(USE$)
2070  PE%=(-1):PP%=(-1):GOTO 1090
2900  REM -----------Traitement Exposants negatifs ------
3000  GAUCHE$=LEFT$(USE$,PE%):LG%=LEN(GAUCHE$):REM part.a gauche de l'exp.
3005  IF LG%=2.0 THEN 3020:REM cas pas point decimal
3010  IF MID$(GAUCHE$,2,1)="." THEN G$=LEFT$(GAUCHE$,2)+RIGHT$(GAUCHE$,LG%-
PP%-1):GAUCHE$=G$:REM suppr.point dec.
3020  LG%=LEN(GAUCHE$):GAUCHE$=RIGHT$(GAUCHE$,LG%-1):REM sppression blanc e
n tete
3030  EX$=RIGHT$(USE$,L%-PE%-2):EX%=VAL(EX$):REM valeur Exposant
3040  ZERO%=EX%-1:REM nbre de zeros a ajouter
3050  USE$="0.":ZERO$="0"
3060  FOR I%=1 TO ZERO%:USE$=USE$+ZERO$:NEXT
3070  USE$=USE$+GAUCHE$:REM creation de la chaine non signee
3080  IF SGN(VALEUR)=(-1.0) THEN USE$="-"+USE$:GOTO 3100
3090  USE$=" "+USE$:REM si positif,un blanc en tete
3100  L%=LEN(USE$):PE%=(-1):PP%=(-1):GOTO 1090
```

Aan alle leden die interesse hebben in het besturen van processen met onze DAI, deel ik hetvolgende mee;

- Ik ben bezitter van twee stappenmotoren en wens zelf een kleine plotter te maken voor het tekenen van printen.
- Het formaat van de tekening is juist groot genoeg voor het tekenen van twee eurokaarten. (I60 x ?I0 mm.)
- De plotter zou bestaan uit :
  - a- twee stappen motoren 48 steps per omwenteling.
  - b- twee drivers met constante stroom sturing.
  - c- een single board processor met 8085 en 8I55.
  - d- de voorziene nauwkeurigheid is I/20 mm.
  - e- een voeding.
  - f- software om alles te sturen.
  - g- de mogelijkheid voor leden om de plotter na te maken met stukken die in de handel voor een een geringe prijs te krijgen zijn.
- --Indien er leden zijn die me kunnen helpen met schema's of software voor het sturen van de motoren zou dit een welkome hulp zijnin mijn programma.
- Zou het mogelijk zijn om hulp te krijgen bij kleine mechanische montages ? Mits vergoeding eventueel.
- Ook heb ik interesse voor een snelle A/D omvormer voor de DAI. (500 kHz indien mogelijk)
  Enkel het type en de prijs zijn doorslaghevende elementen

Mijn nieuw adres is:

termote wouter
stationsstraat 84
8030 BEERNEM

F.W. Biekart
Thomsonplein 6
2565 KS Den Haag
Nederland

DAI-INTER

's-Gravenhage, 26 januari 1984

Geachte heer

Hierbij stuur ik U een klein mlp programma, dat het mogelijk maakt een interrupt service routine in Basic te schrijven.
Het testen van niet al te tijdkritische doch ingewikkelde interrupt routines (alvorens ze in assembler te schrijven) en het ontwikkelen van eenvoudige real-time toepassingen is hierdoor mogelijk.
Hoewel misschien geen stijlbloempje wat programmeren betreft is het naar mijn mening mogelijk m.b.v. het bijgeleverde kommentaar de werking te doorgronden.
Opstarten: 1) DAI aan 2) UT↵ 3) S029C 02-04↵ 4) R↵
5) V7 D9A9-0300↵ 6) B↵ 7) LOAD↵ 8) RUN↵
9) Vul voor interrupt interval-time b.v. 1 (sec) in.

Mocht dit iets voor Dainamic zijn, dan kunt U het plaatsen.

Voorts ben ik via een artikel in een blad op mijn vakgebied geattendeerd op het bestaan van een redelijk uitgebreid dynamisch proces simulatie pakket (blok-geörienteerd) voor de DAI-PC.
Het is ontwikkeld aan de Technische Hogeschool Twente en na kontakt te hebben opgenomen is de eventuele verspreiding aan mij overgelaten.
Hoewel misschien wat vaktechnisch, sluit ik een kopie van het eerder genoemde artikel bij.
Ik kan - indien hiervoor belangstelling bestaat - extra (technische) informatie geven.

In de hoop binnenkort van U bericht te mogen ontvangen, teken ik,

Met vriendelijke groeten,

Fred Biekart (070-603292)

```
0000                            PRT         BFH
0000        :
0000                TITL        'DAIINTER V1.0'
0000        ;
0000        :DATE:              08-JAN-84 /2
0000        ;
0000        :BY:                F.W. Biekart / Den Haag
0000        ;
0000        :PURPOSE:
0000        :                   Makes it possible to execute (part of) a BASIC program
0000        :                   if a (time controlled) interrupt occurs and returns to the
0000        :                   original BASIC-program afterwards.
0000        :                   [ a kind of GOSUB on (time-)interrupt ]
0000        :METHOD:
0000        :                   After a number of calls of the changed clock interrupt
0000        :                   routine RST 7 (according the user supplied time inter-
0000        ;                   val between interrupts) the execution of a (BASIC) 'main'
0000        :                   program is interrupted (via a changed RST 5 routine).
0000        :                   The (BASIC) execution continues at a user supplied linenumb.
0000        :                   This line is the first line of the 'interrupt' program.
0000        :                   When a CALLM #350 is encountered, the (BASIC) execution
0000        :                   will continue at the point where it left the 'main'
0000        :                   program. (see also articles "DAI restart routines" and
0000        :                   "ON ERROR GOTO" of N.Looije in DAINAMIC 15 & 16 /1983)
0000        :REMARKS:
0000        :                   The user supplied time interval between interrupts should
0000        :                   be larger than the total handling time of a 'interrupt'
0000        :                   program.
0000        :                   IMPORTANT:
0000        :                   A new interrupt request before the handling of the
0000        :                   previous interrupt is terminated will reset the interrupt
0000        :                   select flag (0EH is set to 00H : disables the interrupt
0000        :                   possibility until this flag is explicitly set)
0000        :                   **If interrupts are not granted :
0000        :                     1) increase time interval between interrupts: (#1E/1F)
0000        :                     2) clear the interrupt-counter          (#26/27 = 0/0)
0000        :                     3) clear the interrupt exec./request flag    (#0F = 0)
0000        :                     4) set interrupt select flag               (#0E = #FF)
0000        :USE:
0000        :                   *UT <return>
0000        :                   >S029C 02-04 <cursor left>
0000        :                   >R
0000        :                   >V7 D9A9-0300 <return>
0000        :                   >B
0000        :
0000        :                   ++ In the BASIC-program ++
0000        :   ....            POKE #E,#0:REM disable interrupt possibility
0000        :   ....            POKE #F,#0:REM reset interrupt execution/request flag
0000        :   ....            POKE #6,LINENUMBER IAND #FF:REM give first linenumber
0000        :   ....            POKE #7,LINENUMBER SHR 8   :REM of 'interrupt' program
0000        :   ....            POKE #1E,INTERTIME IAND #FF:REM time interval between
0000        :   ....            POKE #1F,INTERTIME SHR 8   :REM interrupts 20ms units
0000        :   ....            POKE #F,#FF:REM enable interrupt possibility
0000        :   ....            ----
0000        :   ....            ----
0000        :   LINENUMBER      ----      :REM start (BASIC) 'interrupt' program
0000        :   ....            ----
0000        :   ....            CALLM #350:REM end interrupt: return to 'main' program
```

```
0000                ;  ....          ----
0000                ;
0000                ;-------------------------------------------------------------
0000  @=0006 INTLIN EQU     6H          ;BASIC-line to goto on interrupt      (*)
0000  @=000E INTSFL EQU     0EH         ;interrupt select flag                (*)
0000  @=000F IBUSFL EQU     0FH         ;interrupt execution/request flag
0000  @=0016 RETLIN EQU     16H         ;pointer return BASIC-line after interrupt
0000  @=001E TIMIBI EQU     1EH         ;time interval between interrupts     (*)
0000  @=002A INTTMR EQU     2AH         ;interrupt timer
0000  @=0115 TRAFL  EQU     115H        ;trace flag
0000  @=01BE TIMER  EQU     1BEH        ;timer (e.g. used by WAIT TIME)
0000  @=01C0 CTIMR  EQU     1C0H        ;cursor timer
0000                ;
0000                ;                                                  (*) = user supplied
0000                ;-------------------------------------------------------------
0000                ;
0000                      ORG     300H
0300                ;
0300  F5    VRST7  PUSH PSW             ;save PSW
0301  2ABE01       LHLD    TIMER       ;check timer
0304  7C           MOV A,H
0305  B5           ORA L
0306  CA0D03       JZ      CURSOR      ;if 0 :check cursor timer
0309  2B           DCX H               ;else :
030A  22BE01       SHLD    TIMER       ;decrement timer
030D  21C001 CURSOR LXI H   CTIMR       ;check cursor timer
0310  35           DCR M               ;decrement cursor timer
0311  C21803       JNZ     INTFLG      ;if<>0:check interrupt select flag
0314  360F         MVI M   0FH         ;else :
0316  EF           RST 5               ;reload cursor timer and flash cursor
0317  12           DB      12H
0318  3A0E00 INTFLG LDA       INTSFL      ;check interrupt select flag
031B  B7           ORA A
031C  CA4B03       JZ      OUT7        ;if 0 :quit
031F  2A2600       LHLD    INTTMR      ;check interrupt timer
0322  7C           MOV A,H
0323  B5           ORA L
0324  C24703       JNZ     DCROUT      ;if<>0:decrement int. tim. and quit
0327  3EFF         MVI A   0FFH        ;else :
0329  210F00       LXI H   IBUSFL      ;-IF   not busy with int. exec. AND the
032C  BE           CMP M               ;      request of prev. int. is granted
032D  C23703       JNZ     REQINT      ;-THEN generate interrupt request
0330  2F           CMA                 ;-ELSE reset interrupt select flag
0331  320E00       STA     INTSFL      ;      (cancels the inter. possibility)
0334  C34B03       JMP     OUT7        ;      and quit
0337  320F00 REQINT STA     IBUSFL      ;set interrupt bussy flag  (FFH = set)
033A  321501       STA     TRAFL       ;set trace flag            (FFH = set)
033D  216303       LXI H   VRST5       ;change RST 5 vector into 363H
0340  226C00       SHLD    6CH
0343  2A1E00       LHLD    TIMIBI      ;reload interrupt timer
0346  23           INX H
0347  2B     DCROUT DCX H               ;decrement interrupt timer
0348  222600       SHLD    INTTMR
034B  F1     OUT7   POP PSW             ;restore stack
034C  E1           POP H
034D  FB           EI                  ;enable interrupts
034E  C9           RET                 ;return  / end of INT7-routine
034F  00           NOP
```

```
0350            ;
0350  2A1600 RETINT  LHLD      RETLIN        ;get pointer return line after interrupt
0353  44              MOV B,H
0354  4D              MOV C,L
0355  210200          LXI H     2H          ;calc. the (stack-)pointer to return addr.
0358  39              DAD SP                ;of the CALLM #350 stat.-routine in HL
0359  36E8            MVI M     0E8H        ;and change
035B  23              INX H                 ;the return address
035C  36C8            MVI M     0C8H        ;into E8C8H
035E  AF              XRA A
035F  320F00          STA       IBUSFL      ;reset interrupt execution/request flag
0362  C9              RET                   ;return / end of CALLM #350
0363            ;
0363  F5      VRST5   PUSH PSW              ;save PSW
0364  210C00          LXI H     0CH         ;calculate the (stack-)pointer to find the
0367  39              DAD SP                ;the original caller
0368  7E              MOV A,M
0369  FEFA            CPI       0FAH        ;check if caller is C8FAH (..FAH part)
036B  C27803          JNZ       ONRST5      ;if not :execute normal RST 5
036E  23              INX H
036F  7E              MOV A,M
0370  FEC8            CPI       0C8H        ;check if caller is C8FAH (C8..H part)
0372  C27803          JNZ       ONRST5      ;if not :execute normal RST 5 routine
0375  C37C03          JMP       EXEINT      ;else   :execute interrupt routine
0378  F1      ONRST5  POP PSW               ;restore stack
0379  C3FDC6          JMP       0C6FDH      ;continue with normal RST 5 routine
037C            ;
037C  23      EXEINT  INX H                 ;calc. and set the stackpointer to the
037D  F9              SPHL                  ;return BASIC-line after interrupt
037E  E1              POP H                 ;load point. to the ret. BASIC-line in HL
037F  221600          SHLD      RETLIN      ;save it in memory locations 16/17H
0382  33              INX SP                ;adjust stackpointer
0383  33              INX SP                ;adjust stackpointer
0384  2A0600          LHLD      INTLIN      ;load BASIC-line to goto on inter. in HL
0387  CDF6CA          CALL      0CAF6H      ;calc. pointer to this BASIC-line
038A  D29E03          JNC       UNDEFL      ;if BASIC-line not defined give err. mess.
038D  44              MOV B,H               ;pointer to BASIC-line in BC
038E  4D              MOV C,L
038F  AF              XRA A
0390  F3              DI
0391  321501          STA       TRAFL       ;reset trace flag
0394  21FDC6          LXI H     0C6FDH      ;restore normal RST 5 routine
0397  226C00          SHLD      6CH         ;vector = C6FDH
039A  FB              EI
039B  C392C8          JMP       0C892H      ;continu with BASIC-line to goto on inter.
039E            ;
039E  210000 UNDEFL   LXI H     0H          ;prepare
03A1  220600          SHLD      6H          ;error message UNDEFINED LINENUMBER
03A4  3E04            MVI A     4H
03A6  C3F5D9          JMP       0D9F5H      ;print error mes. / return to BASIC-monitor
03A9                  END
```

*LIST (basic) EXAMPLE OF USE DHIINTER V1.0 /2

```
30    PRINT CHR$(12)
40    POKE #E,#0:REM ***> disable interrupt possibility
50    POKE #F,#0:REM ***> clear interrupt-timer
60    POKE #6,1000 IAND #FF:REM ***> install linenumber
70    POKE #7,1000 SHR 8:REM ******> start 'interrupt' program
80    PRINT "Interrupt time-interval [sec]":PRINT
90    PRINT "The interrupt time-interval value should be larger"
100   PRINT "than the handling time of the 'interrupt' program."
110   PRINT "If this is not the case, a time based execution of"
120   PRINT "the 'interrupt' program will not be possible and the"
130   PRINT "interrupt possibility will be switched off (#0E := 0).":PRINT
140   PRINT "Lower limit value time interval : see note above"
150   PRINT "Upper limit value time interval : (2^16-1)/50 = 1310 sec"
160   PRINT :INPUT "Give interrupt time-Interval [sec] :";IT
170   ITT%=50*IT
180   POKE #1E,ITT% IAND #FF:REM ***> install desired interrupt
190   POKE #1F,ITT% SHR 8:REM ******> time-interval
200   POKE #E,#FF:REM **************> enable interrupt possibility
210   CURSOR 0,9:PRINT "The 'main' program":PRINT "executed when not interrupted"
220   CURSOR 34,9:PRINT "The 'interrupt' program":CURSOR 34,8:PRINT "executed every";IT;" sec"
230   FOR I%=14 TO 127:MC%=MC%+1:CURSOR 5,6:PRINT CHR$(I%);"  ";MC%
240   IF PEEK(#E)=0 THEN PRINT :PRINT "*** INTERRUPT TIME-INTERVAL TOO SHORT ***"
250   NEXT:GOTO 230
260   REM
1000  REM ***> start 'interrupt' program / executed when interrupt occurs <***
1010  IC%=IC%+1:CURSOR 39,6:PRINT IC%
1020  CALLM #350:REM ***> end 'interrupt' program /return to 'main' program
```

DAI-INTER

>0300 3A8 #300 – #3A8 mLp DHIINTER V1.0 /2

```
0300 F3 2A BE 01 7C B5 CA 0D 03 2B 22 BE 01 21 C0 01
0310 35 C2 18 03 36 0F EF 12 3A 0E 00 B7 CA 4B 03 2A
0320 26 00 7C B5 C2 47 03 3E FF 21 0F 00 BE C2 37 03
0330 2F 32 0E 00 C3 4B 03 32 0F 00 32 15 01 21 63 03
0340 22 6C 00 2A 1E 00 23 2B 22 26 00 F1 E1 FB C9 00
0350 2A 16 00 44 4D 21 02 00 39 36 E8 23 36 C8 AF 32
0360 0F 00 C9 F5 21 0C 00 39 7E FE FA C2 78 03 23 7E
0370 FE C8 C2 78 03 C3 7C 03 F1 C3 FD C6 23 F9 E1 22
0380 16 00 33 33 2A 06 00 CD F6 CA D2 9E 03 44 4D AF
0390 F3 32 15 01 21 FD C6 22 6C 00 FB C3 92 C8 21 00
03A0 00 22 06 00 3E 04 C3 F5 D9
```

Simulation program (BASIM) for personal computers

R. P. Offereins and J. W. Meerman *

SUMMARY

The paper describes a block oriented interactive simulation program BASIM written in BASIC for personal computers. It details the interaction possibilities between user and computer by describing the available commands and their effects. It describes the organization of the data storage, the task of the various routines and the method of determining the computation sequence. Finally the required memory space is given and the computation speed is compared with similar programs.

1. INTRODUCTION

Block oriented simulation of dynamic systems originates from the time that it was the only method for obtaining simulation results using analogue computers [9]. Several authors describe block oriented simulation programs [1-8]. There are a number of advantages in using such a block oriented interactive simulation language instead of some general purpose language.

- The user keeps a close relation with the actual physical system via its block diagram.
- The language is very simple and can be learned in a very short time.
- The conversational interactive capability facilitates changes of model and parameters at any moment.
- The sequence of calculations and of plotting variables is automatically arranged in the right way.

The simulation language BASIM as described in this paper is written in BASIC. It is developed for personal computers. It is similar to and based on the program THTSIM [1, 3], which was developed by the Control and Automation Group of the Twente University of Technology and which is now successfully used by some hundreds of industries and institutes. THTSIM can also accept models formulated with bond graphs [2, 3]. THTSIM originally is written in assembler for PDP-11 minicomputers or LSI-11 microcomputers. Meerman announced THTSIM for personal computers [3].
An assembler version on Apple II and FORTRAN versions on CP/M and other computers are available now [8]. The disadvantage of BASIC is its slowness. The advantage is that it is an interactive language which makes it possible to mix BASIC and BASIM. An early version of BASIM is described in [7]. Experience with the Apple II assembler version shows a difference in computing time of roughly a factor 20. The calculation part of BASIM is also in machine language giving an increase in speed of a factor 10. The greater part containing input, interaction with the user, error checking, preparing the sequence of calculations, etc. can remain in BASIC.

2. COMPUTER CONFIGURATION

The personal computer configuration used for developing and using the program is a DAI personal computer with 48 k RAM, a B/W portable TV and an audio cassette. The personal computer is based on the 8080 A microprocessor. The version used contains the arithmetic chip AM9511. Integers and floating point numbers both use 4 bytes. The DAI has good graphic possibilities with a resolution of 256 X 352 pixels of 4 colours or grey levels which can be chosen at will. The cost of the configuration as described was $ 1600,- in 1980.

3. DESCRIPTION OF SIMULATION LANGUAGE

The central element of the language is the block. It is essentially an element with one output variable, that depends on a number of input variables and a number of constants (parameters). By interconnecting a number of blocks in such a way that input variables of a block are connected to output variables of other blocks, a structure is obtained which represents the model of a system and which is called a block diagram. As a simple example we consider in this paper the block diagram of the second order system of fig. 1.
This second order system is the model of a mass spring system governed by the differential equation.

$$M\ddot{x} + F\dot{x} + C x = K$$

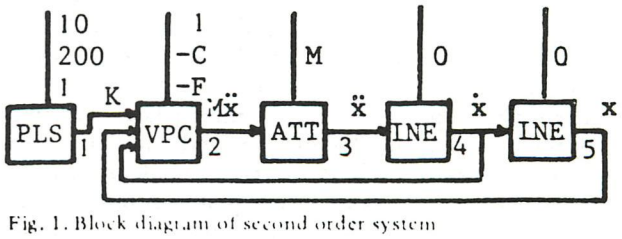


Fig. 1. Block diagram of second order system

* Department of Electrical Engineering, Twente University of Technology, P. O. Box 217, 7500 AE Enschede, The Netherlands.

The block type is denoted with a three character mnemonics. In this example PLS = pulse function, ATT = attenuation, INE = integration, VPC = product of variables and constants; the output variable is y; input variables are $i_1, i_2 .. i_n$ and parameters are $p_1, p_2 \cdots p_m$.

The formulas for these blocks are

PLS : $y = p_3$ for $p_1 < t_1 < p_2$

$y = o$ for $t \leqslant p_1$ and $t \geqslant p_2$

VPC : $y = i_1 p_1 + i_2 p_2 + \ldots i_n p_n$

ATT : $y = i/p$

INE : $y_{n+1} = y_n + T/2 (3i_n - i_{n-1})$

Each block in the diagram is uniquely identified by a number which corresponds with its output variable. For simulating the dynamic system of fig. 1 the block numbers, the interconnections and the parameters are inserted into the computer as shown in the following listing, which is explained below.

```
TYPE COMMAND ? NS
SOURCE  K OR T: ? K
STRUCTURE
?1 PLS
?2 VPC 1 5 4
?3 ATT 2
?4 INE 3
?5 INE 4
?
PARAMETERS (P1, P2 ... Pn)
1 ? 10 ? 200 ? 1
2 ? 1 ? -0.1 ? -0.5
3 ? 10
4 ? 0
5 ? 0
TIMING
DELTA ?1 FINTIM ?100
PLOT BLOCKS (NR SPACE NR. ETC.) ? 0 1 4 5
RANGES
RANGE NR 0 MIN ? 0 MAX ? 100
RANGE NR 1 MIN ? -2 MAX ? 10
RANGE NR 4 MIN ? -20 MAX ? 20
RANGE NR 5 MIN ? -2 MAX ? 2
TYPE COMMAND?
```

After loading the program and typing RUN the computer asks by printing ? the information from the user. The first question asks for a command which is in this case NS (New Structure). For model input two sources are possible : K (Keyboard) or T (Tape). In the example K is given. The user then defines for each block of the structure its number, the type and the input-numbers. The structure input is finished by a carriage return. Then the computer asks for the parameters of each block. If it knows the number of parameters it asks for each parameter separately. If it does not know this number it first asks for this number and then for the parameters separately. They are inserted by the user. Then the computer asks for the timing data consisting of the time step DELTA and the total simulation time FINTIM. After this the numbers of the block outputs to be plotted are asked for. The first one (0 in this case) is the X-value and the remaining ones are Y-values. In the example the outputs 1, 4 and 5 are plotted against time as 0 is used for the time variable. Finally the computer asks the ranges of the variables to be plotted consisting of a minimum and a maximum value such that MAX≥MIN. The model input is finished by a TYPE COMMAND question. During the input appropriate error messages are used when giving improper inputs.

4. COMMAND LIST

The commands available are detailed below. From this list the conversational capabilities of the program become clear. The computer asks for a command by printing TYPE COMMAND and this question has to be answered by one of the commands of the list. TYPE COMMAND? is printed after loading the program and initiating it by the BASIC-command RUN, after having executed a previous command and after interrupting a simulation run by pressing the space bar.

The commands are :

- NS (New Simulation). The computer prepares itself for the input of a model and asks for the necessary input data as shown in the previous section.
- NG (New Graph). The computer clears the screen and draws 10 horizontal and 10 vertical grid lines for plotting graphs. The lower part of the screen remains available for 4 character lines where further conversation between user and computer is printed.
- S (Simulate). Starts Simulation from the initial conditions. Outputs of integrators are set to the initial conditions and plot points for time = 0 are plotted. The simulation is continued until the time equals the value FINTIM, which is indicated by printing END RUN or until it is stopped by pressing the space bar.
- P (Proceed Simulation). The program continues simulation from the existing conditions. The command can be used after END RUN. Then automatically the time range of the graphs is adapted to display the nex part of the graph. It also can be used after interrupting a simulation run. Changes of structure or parameters can be introduced etc. and then the run can be continued without further changes of variables.
- TA (Type All). Prints all data of the model. The screen is cleared and the block data including output and parameters are displayed with one block per line.
- TT (Type Timing). Prints DELTA and FINTIM.
- TPL (Type Plots). Prints plotnumbers and plotranges.
- TX (Type number X). Asks for a block number. After inserting it the data of this block are printed.
- CS (Change Structure). New blocks can be inserted in a format as described in the previous section. Carriage return finishes the structure change.
- CT (Change Timing). Values of DELTA and FINTIM can be changed.
- CPL (Change Plotblocks). The blocks to be plotted can be changed followed by a range input.
- CR (Change Range). The range of a plotblock can be changed.
- CP (Change Parameter). Asks for a block number. After inserting it the program asks for the parameters of this block.

- DX (Delete number X). Asks for a block number to be deleted.
- VX (Value number X). Asks for a block number of which the output value has to be printed.
- STOP. Interrupts the program for saving or loading models structures to and from tape and for using direct BASIC instructions.

5. DATA STORAGE

A two dimensional 205 X 6 array PAR (205,5) is defined for storing model structure and model parameters. Each row of this array has 6 X 4 bytes containing the data of one block.
The maximum number of blocks is 200; row 0 is used for timing data, rows 201 to 205 are used for plot data. Each block has a maximum of 4 parameters and 6 inputs. The first 8 bytes are used for numbers between 0 and 200 respectively representing the block type, the number of input signals and the actual numbers of the input signals. The last 4 X 4 bytes are used for maximal 4 parameters. Saving on tape and loading from tape of model structure and model parameters is done by saving and loading the array PAR.
A one-dimensional array SIG (205) is used for storing the output values of the blocks.
SIG (0) contains the variable TIME. An array SEQ (52) of 4 X 52 bytes is used for storing the numbers of the blocks of the model in order to increase the number value. An array SOR (52) of 4 X 52 bytes is used for storing the numbers of the blocks of the model in the computation sequence.
A character string TYP$ is defined containing a series of sets of 5 characters. Each set contains three characters representing the mnemonics of a block type (e.q. ATT, CST, etc.), one character denoting the number of parameters and one character denoting the number of inputs.
The setnumber in the series is the code which the computer uses for the block type.

6. PROGRAM ROUTINES

The program consists of the routines mentioned below.
- INITIALIZE. This is the program for defining the arrays and the character string mentioned in the previous section. It is entered after loading the program by the BASIC instruction RUN. It also contains the data for the machine language routines. They are loaded in the proper RAM space by this routine.
- MONITOR. This is the central program to which the computer returns after INITIALIZE and after having executed a BASIM command. It also controls two loops : an outer loop for the time steps and an inner loop for the calculation of the various blocks in one time step. The subroutines for each block type are called from the inner loop including the blocks for graph plotting. A maching language program is available for the inner loop. Slight alterations in this monitor program (removing and adding REM statements for some BASIC commands) enable the use of calculation routines in BASIC or machine language or both.
- STRUCTURE INPUT. This subroutine handles the input of a new structure from keyboard. It is entered after the command NS.
- BLOCK INPUT. This subroutine handles the input of the data (type and input numbers) of one block. It is entered from the subroutine STRUCTURE INPUT and after commands for changing the structure.
- PLOT BLOCK INPUT. This subroutine handles the input of the block numbers to be plotted. It is used after the command NS and the command CPL (Change Plotblocks).
- RANGES INPUT. This subroutine handles the input of the ranges after the command NS, CPL or CR (Change Range).
- PARAMETERS INPUT. This subroutine handles the input of all parameters of a structure from Keyboard after the command NS (New Structure).
- PARAMETERS ONE BLOCK. This subroutine handles the input of the parameters of one block. It is used by the subroutine PARAMETERS INPUT and after the command CP (Change Parameters).
- TIMING INPUT. This subroutine handles the input of timing data after the command NS (New Structure) and CT (Change Timing).
- SEQ. This subroutine puts the numbers of the blocks of the model in the array SEQ (52) in the order of increasing number value. It is used by the routines NS, CS (Change Structure) and DX (Delete block X).
- SOR. This subroutine rearranges the numbers of the array SEQ (52) in array SOR (52) in computation sequence before actual simulation.
- COMPUTE BLOCK. These are subroutines, one for each block type, to update the output values of the blocks during each time step of the simulation. The machine language program contains also a routine for each block.
- BASIM COMMANDS. These are subroutines for handling the user commands mentioned before. They are entered from the MONITOR.

7. COMPUTATION SEQUENCE

An essential feature of a digital simulation method for dynamic systems is the sequence of calculations. At time nT the output variables of all blocks are known, starting with the initial condition for n = 0. Then for each block a new output variable valid for time (n+1)T has to be calculated. A general block diagram for a dynamic system has a structure as shown in fig. 2.
The equations associated with this system are

$$\frac{d\underline{x}}{dt} = \underline{f}(\underline{x}, \underline{u})$$

$$\underline{y} = \underline{g}(\underline{x}, \underline{u})$$

The example of fig. 1 also has this structure.
The loop structure of fig. 2 generally implies that for integrating vector $\underline{f}$ to $\underline{x}$ we have implicit equations. They could be solved using interpolating digital integration methods by iterative procedures which, if conver-

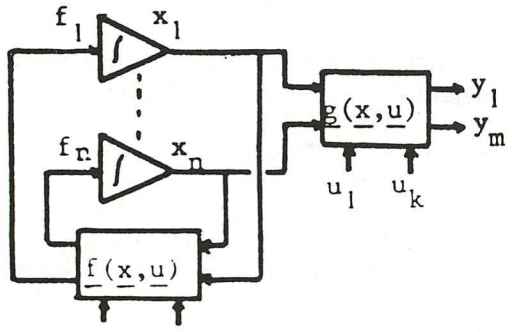


Fig. 2. General block diagram

gent, would result in a solutation at the cost of a large computation time as the equations $\underline{f}(\underline{x},\underline{u})$ have to be calculated a number of times for each time step. To avoid excessive computation times extrapolating digital integration methods can be used with only one computation per time step. Also predicting/interpolating digital integration methods could be used, which first predict a value of x for a future time by an extrapolating integration method and then determine the value of x by an interpolating integration method using the predicted value of f. These methods require some calculations of $\underline{f}(\underline{x},\underline{u})$ per time step and they must have an increase in accuracy as compared with an extrapolating method to compensate for the decrease in accuracy arising from the larger value of time step T that is necessary. The program BASIM uses two integration methods :

an extrapolating integration method (INE) with integration formula

$$x_{n+1} = x_n + T/2\,(3f_n - f_{n-1})$$

and an interpolating integration method (INI) with integration formula

$$x_{n+1} = x_n + T/2\,(f_{n+1} + f_n)$$

The table below shows the accuracy of both integration methods for sinusoidal signals as a function of the frequency w expressed as $E = (A_d - A)/A$. Here A_d is the amplitude of the digitally integrated signal and A is the amplitude of the accurately integrated signal.

Error E for sinusoidal signals :

wT	1,6	1	0,4	0,2
INE	0,97	0,35	0,065	$0.004(wT)^2$
INI	0,22	0,073	0,013	$0,0008(wT)^2$

The computations sequence is such that first the new output values of the extrapolating blocks are calculated, then the blocks with no input, and finally the remaining blocks in such an order that an output is calculated if all input values valid for time (n+1)T are known. For the example of fig. 1 this procedure results in the computing sequence 5, 4, 1, 2, 3 if the two integrating blocks are extrapolating (INE) blocks. It is possible to use an interpolating block INI instead of number 5. In that case the computation sequence becomes 4, 5, 1, 2, 3.
If the routine SOR which determines the sequence of calculation cannot find such a sequence, this is caused by the fact that a loop without an extrapolating block is present or that an input number is encountered with no corresponding output number.
In that case an error message is printed and the user has either to insert an extrapolating block in a loop or to correct the input number error. A loop without an extrapolating block can be corrected by changing an integration INE into an integration INI or by inserting a block DLE, which is an extrapolating block.

8. SORTING METHOD

The routine SOR puts the blocks in the array SOR (52) in the computation sequence. The blocks have been put already in the order of increasing number value in the array SEQ (52) by the routine SEQ. This subroutine SEQ has also determined the total number of blocks NB, the number of extrapolating blocks NM and it has set all output values of the array SIG (205) to one except those belonging to not-extrapolating blocks, which are set to zero. Now the routine SOR scans the blocks of the array SEQ. As soon as a block with output value one is detected the sum of the inputs, which are in the array SIG (205), is determined. If this sum is zero the block number is placed in the array SOR and its output in the array SIG (205) is set to zero. This process is repeated until the NM extrapolating blocks are put in the first NM places of the array SOR in reversed order so that a block selected earlier comes after a block selected later. If after NM scans through the array SEQ the number of blocks put in the array SOR is smaller than NM an error message is printed and the routine is finished. The user has to make his correction of the model structure. After placing the NM extrapolating blocks the remaining blocks are placed. First the output values of the not-extrapolating blocks are set to one. Then the blocks of the array SEQ are scanned and when a block with output value one is encountered it is considered for placing in the array SOR. If there is no input it is put in this array and its output value is set to zero. If there are inputs the sum of these inputs is determined. If this sum is zero the block is placed in the array SOR and its output value is set to zero. The blocks are placed in SOR in the order of selection. This process is repeated until NB blocks are put in the array SOR. If after NB-NM scans the number of blocks in SOR is smaller than NB, an error message results. The blocks for drawing plots are finally added to the list.

9. PROGRAM FLEXIBILITY

As the program is written in BASIC it is simple to alter or to extend if the user knows BASIC. Adding a new block type is a matter of programming a few BASIC lines and extending the TYP$ in the routine INITIATE with 5 characters containing the mnemonics for the new block, the number of inputs and the number of parameters. The MONITOR instruction : ON TYP(X) GOSUB has to be extended with the line number where the new block subroutine starts.
The existing BASIM version already contains a number

of block types BT0, BT1 to BT5 which have to be user-programmed on prereserved program lines. The existing routines only contain the BASIC line RETURN. They must be programmed before introducing a model from keyboard or tape.
As the computer does not know the number of parameters it first asks for this number and after that it asks for each parameter separately.
It is possible to use simultaneously blocks preprogrammed in BASIC, user programmed blocks and blocks preprogrammed in machine language which use internal DAI routines.

10. MEMORY SPACE AND COMPUTATION SPEED

The RAM memory space required for the program, data storage and graphics is

- 6,5 K for model structure, parameters and output. Each block has a maximum of 6 input numbers and 4 parameters.
- 16 K for the BASIC program.
- 20 K for the high resolution 256 X 352, 4 colour graphics.
- 4 K for machine language routines. These data are stored twice. They are loaded from DATA statements in the BASIC program into the program area of the memory by the routine INITIALIZE.

This means that 0,5 K is available for programming user defined blocks. For each time step the computation time for the ML version can roughly be determined by the formula

$$14 + 7\,NP + 3\,NB \text{ millisec}$$

where NP is the number of graphs (maximum 3) and NB is the number of blocks (maximum 200).
The computation times required for calculating one time step of the model of fig. 1 is given in the table below together with the times of other THTSIM realisations as described in [8]. Plotting times are not included.

Computer	Language	Time millisec./time step
PDP 11	MACRO 11	2.9-15
LSI 11	MACRO 11	8,3
LSI 11/23	MACRO 11	4,8
LSI 11	FORTRAN	21
OSBORNE	CP/M FORTRAN	30
BICBOARD	CP/M FORTRAN	68
APPLE	ASSEMBLER	18
APPLE (with AM9511)	ASSEMBLER	7,5
DAI (with AM9511)	BASIC	160
DAI (with AM9511)	ML-ROUTINES	15

CONCLUSIONS

BASIM is a very flexible and yet powerful block oriented simulation language which is now available for all personal computers using BASIC. The cost of such a tool for simulating dynamic systems is such that a wide spread use for educational purposes is possible. Replacing part of the program by machine language routines will increase the speed by a factor 10.

REFERENCES

1. KRAAN R. A. : "THTSIM, a conversational simulation program on a small digital computer." Journal A, vol. 15, 4, 186-190 (1974).
2. DIXHOORN J. J. VAN : "Simulation of bond graphs on minicomputers", Journal of Dynamic Systems, Measurement and Control, March, 9-14 (1977).
3. MEERMAN J. W. : "Bond graph modelling techniques THTSIM, software for the simulation of continuous dynamic systems on small and very small computer systems". International Journal of modelling and Simulation. Vol. 1; No. 1, 52-56 (1981).
4. TIERNEGO M. J. L. : "Bond graph modelling and simulation techniques applied to a three axis driven pendulum." International Journal of Modelling and Simulation. Vol. 1; No. 1, 62-66 (1981).
5. BOSCH P. P. J. VAN DEN (1981) : "PSI-Software tool for control system design." Journal A. Vol. 22; No. 2, 55-61 (1981).
6. BOSCH P. P. J. VAN DEN, and SCHOUTEN H. P. R. (1976). "SIM – An interactive simulation program for both continued and discrete systems." Proceedings 8th AICA Congress, Delft.
7. OFFEREINS R. P. and MEERMAN J. W. (1982) : "Simulation Program (BASIM) for personal computers". Paper presented at IFAC/IFIP symposium on software for computer control, Madrid, October 5-8, 1982.
8. MEERMAN J. W. : "Dynamic System Simulation with personal computers and THTSIM". Paper to be presented at SCS multi-conference on Modeling and Simulation on micro-computers, San Diego. January 27-29, 1983.
9. IDZERDA H. H., ENSING L., JANSSEN J. M. L., OFFEREINS R. P. : "Design and Applications of an Electronic Simulator for Control Systems" Transactions of the Society of Instrument Technology, Vol. 7, No. 3 (1955).

R. P. OFFEREINS is professor of control engineering at the Twente University of Technology since 1965. Before that he worked with SHELL in the field of process control and Philips-Signaal in the field of digital fire control.

J. W. MEERMAN was born in Neede, in 1943. He studied Electrical Engineering at Twente University of Technology. Presently he is with the Control and Automation Group of Twente University of Technology. His main research subject is the usage of microcomputersystems in control and simulation.

```
10    PRINT "*****************************************************************"
20    PRINT "*** RECHTECKSCLANGEN BASIC - TINY-PASCAL - MACHINE-LANGUAGE ***"
30    PRINT "*****************************************************************"
40    PRINT
50    PRINT "The programs move a rectangle of random breadth XS and height"
60    PRINT "YS with random x and y steps DX,DY trough the screen."
70    PRINT "If the rectangle hits a border of the screen, it is reflected."
80    PRINT
90    PRINT "The differences of the 3 programs in speed are impressive, if"
95    PRINT "it is considered that each one uses the FILL function."
100   PRINT
110   PRINT "Event 0 is used to produce a new rectangle."
```

BASIC

```
1     REM *** RECHTECKSCHLANGEN BASIC ***
2     REM ***   MARKUS SIGG 18/6/82   ***
10    COLORG 8 1 3 5:MODE 4
20    XS%=RND(#20):XM%=XMAX-XS%:DX%=RND(#10)
30    YS%=RND(#20):YM%=YMAX-YS%:DY%=RND(#10)
40    X%=XMAX/2:Y%=YMAX/2
50    FILL X%,Y% X%+XS%,Y%+YS% 21+(X%+Y%) MOD 3
60    X%=X%+DX%:IF X%>XM% OR X%<0 THEN DX%=-DX%:X%=X%+DX%+DX%
70    Y%=Y%+DY%:IF Y%>YM% OR Y%<0 THEN DY%=-DY%:Y%=Y%+DY%+DY%
80    IF PEEK(#FD00) IAND #20=0 THEN 50:GOTO 10
```

DTP

```
!*** RECHTECKSCHLANGEN TINY-PASCAL ***
          MARKUS SIGG 18/6/82          !

VAR X,Y,XS,YS,XM,YM,DX,DY:INTEGER;

PROC MODE(M);
 BEGIN MEM[#7D0]:=M;
       CALL(#7FD)
 END;

PROC COLORG(C1,C2,C3,C4);
 BEGIN MEM[#7D0]:=C1;MEM[#7D1]:=C2;MEM[#7D2]:=C3;MEM[#7D3]:=C4;
       CALL(#82B)
 END;

FUNC XMAX;
 BEGIN MEM[#7D2]:=0;MEM[#7D5]:=0;MEM[#7D6]:=0;
       CALL(#912);
       XMAX:=256*MEM[#7D4]+MEM[#7D3]
 END;

FUNC YMAX;
 BEGIN MEM[#7D2]:=0;MEM[#7D5]:=0;MEM[#7D6]:=0;
       CALL(#912);
       YMAX:=MEM[#7D1]
 END;
```

```
FUNC RND;
 VAR I,R:INTEGER;
 BEGIN R:=Ø;
       FOR I:=1 TO 4 DO R:=(MEM[#FDØØ] AND #4Ø OR R) SHR 1;
       RND:=R SHR 1
 END;

BEGIN REPEAT COLORG(8,1,3,5);MODE(6);
              XS:=RND;XM:=XMAX-XS;DX:=RND SHR 1;
              YS:=RND;YM:=YMAX-YS;DY:=RND SHR 1;
              X:=XMAX DIV 2;Y:=YMAX DIV 2;
              MEM[#7D4]:=Ø;MEM[#7D6]:=Ø; ! High bytes not used
              REPEAT MEM[#7D3]:=X;MEM[#7D1]:=Y;
                     MEM[#7D5]:=X+XS;MEM[#7D2]:=Y+YS;
                     MEM[#7DØ]:=21+(X+Y) MOD 3;
                     CALL(#8CC);          ! Draw !
                     X:=X+DX;
                     IF (X>XM) OR (X<Ø) THEN BEGIN DX:=-DX;
                                                   X:=X+DX+DX
                                             END;
                     Y:=Y+DY;
                     IF (Y>YM) OR (Y<Ø) THEN BEGIN DY:=-DY;
                                                   Y:=Y+DY+DY
                                             END
              UNTIL MEM[#FDØØ] AND #2Ø=#2Ø
       UNTIL XM=Ø ! never true !
END.
```

PAGE Ø3

```
*****************************
* S Y M B O L   T A B L E *
*****************************
```

COL	Ø09E	DX	Ø4C1	DY	Ø4C2	EVENT	Ø49C
LOOP	Ø4AB	OK	Ø46Ø	RANDOM	Ø4A7	REPEAT	Ø446
START	Ø4ØC	X	Ø4BB	XM	Ø4BF	XMAX	Ø09F
XMAX/2	Ø04F	XS	Ø4BD	Y	Ø4BC	YM	Ø4CØ
YMAX	Ø081	YMAX/2	ØØ4Ø	YMOVE	Ø48Ø	YS	Ø4BE

```
ØØ1                     *** RECHTECKSCHLANGEN MACHINE-LANGUAGE ***
ØØ2                     ***         MARKUS SIGG 16/6/82          ***
ØØ3                     *
ØØ4                     * startadress: #4ØØ
ØØ5                     *
ØØ6                     XMAX     EQU    159
ØØ7                     YMAX     EQU    129
ØØ8                     XMAX/2   EQU    79
ØØ9                     YMAX/2   EQU    64
Ø1Ø                     COL      EQU    :9E        adress of COLORG registers
Ø11                     *
Ø12                              ORG    :4ØØ       -
Ø13 Ø4ØØ 218891                  LXI    H,:9188    -
Ø14 Ø4Ø3 229EØØ                  SHLD   COL        -
Ø15 Ø4Ø6 21A3B5                  LXI    H,:B5A3    -
Ø16 Ø4Ø9 22AØØØ                  SHLD   COL+2      -- COLORG 8 1 3 5
Ø17 Ø4ØC 3EØ6           START    MVI    A,:Ø6      -
Ø18 Ø4ØE EF                      RST    5          -
Ø19 Ø4ØF 18                      DATA   :18        -- MODE 4
Ø2Ø Ø41Ø CDA7Ø4                  CALL   RANDOM     -
Ø21 Ø413 32BDØ4                  STA    XS         -- XS=RND(#2Ø)
Ø22 Ø416 2F3C                    TCA               -
Ø23 Ø418 C69F                    ADI    XMAX       -
Ø24 Ø41A 32BFØ4                  STA    XM         -- XM=XMAX-XS
Ø25 Ø41D CDA7Ø4                  CALL   RANDOM     -
Ø26 Ø42Ø B71F                    LRA               -
Ø27 Ø422 32C1Ø4                  STA    DX         -- DX=RND(#1Ø)
Ø28 Ø425 CDA7Ø4                  CALL   RANDOM     -
Ø29 Ø428 32BEØ4                  STA    YS         -- YS=RND(#2Ø)
Ø3Ø Ø42B 2F3C                    TCA               -
Ø31 Ø42D C681                    ADI    YMAX       -
Ø32 Ø42F 32C2Ø4                  STA    YM         -- YM=YMAX-YS
Ø33 Ø432 CDA7Ø4                  CALL   RANDOM     -
Ø34 Ø435 B71F                    LRA               -
Ø35 Ø437 32C2Ø4                  STA    DY         -- DY=RND(#1Ø)
Ø36 Ø43A 3E4F                    MVI    A,XMAX/2   -
Ø37 Ø43C 32BBØ4                  STA    X          -- X=XMAX/2
Ø38 Ø43F 3E4Ø                    MVI    A,YMAX/2   -
Ø39 Ø441 32BCØ4                  STA    Y          -- Y=YMAX/2
Ø4Ø Ø444 26ØØ                    MVI    H,Ø        high bytes not used, (D is*
Ø41 Ø446 3ABBØ4         REPEAT   LDA    X          *zeroed by RANDOM)
Ø42 Ø449 5F                      MOV    E,A        x1
Ø43 Ø44A 3ABDØ4                  LDA    XS
Ø44 Ø44D 83                      ADD    E
Ø45 Ø44E 6F                      MOV    L,A        x2
Ø46 Ø44F 3ABCØ4                  LDA    Y
Ø47 Ø452 47                      MOV    B,A        y1
Ø48 Ø453 3ABEØ4                  LDA    YS
Ø49 Ø456 8Ø                      ADD    B
Ø5Ø Ø457 4F                      MOV    C,A        y2
Ø51 Ø458 7B                      MOV    A,E
Ø52 Ø459 8Ø                      ADD    B
Ø53 Ø45A E6Ø3                    ANI    3
Ø54 Ø45C C26ØØ4                  JNZ    OK         color must not be Ø (8)
Ø55 Ø45F 3C                      INR    A
Ø56 Ø46Ø C614           OK       ADI    2Ø         color
```

```
Ø57 Ø462 EF                    RST   5          -
Ø58 Ø463 24                    DATA  :24        -- FILL X,Y X+XS,Y+YS COLOR
Ø59 Ø464 3AC1Ø4                LDA   DX         -
Ø6Ø Ø467 83                    ADD   E          -
Ø61 Ø468 32BBØ4                STA   X          -- X=X+DX
Ø62 Ø46B 4F                    MOV   C,A        -
Ø63 Ø46C 3ABFØ4                LDA   XM         -
Ø64 Ø46F B9                    CMP   C          -- IF X>XM OR X<Ø THEN*
Ø65 Ø47Ø D28ØØ4                JNC   YMOVE      -
Ø66 Ø473 3AC1Ø4                LDA   DX         -
Ø67 Ø476 2F3C                  TCA              -
Ø68 Ø478 32C1Ø4                STA   DX         -- *DX=-DX
Ø69 Ø47B 87                    LLA              -
Ø7Ø Ø47C 81                    ADD   C          -
Ø71 Ø47D 32BBØ4                STA   X          -- *X=X+DX+DX
Ø72 Ø48Ø 3AC2Ø4      YMOVE     LDA   DY         -
Ø73 Ø483 8Ø                    ADD   B          -
Ø74 Ø484 32BCØ4                STA   Y          -- Y=Y+DY
Ø75 Ø487 4F                    MOV   C,A        -
Ø76 Ø488 3ACØØ4                LDA   YM         -
Ø77 Ø48B B9                    CMP   C          -- IF Y>YM OR Y<Ø THEN*
Ø78 Ø48C D29CØ4                JNC   EVENT      -
Ø79 Ø48F 3AC2Ø4                LDA   DY         -
Ø8Ø Ø492 2F3C                  TCA              -
Ø81 Ø494 32C2Ø4                STA   DY         -- *DY=-DY
Ø82 Ø497 87                    LLA              -
Ø83 Ø498 81                    ADD   C          -
Ø84 Ø499 32BCØ4                STA   Y          -- *Y=Y+DY+DY
Ø85 Ø49C 3AØØFD      EVENT     LDA   :FDØØ      -
Ø86 Ø49F E62Ø                  ANI   :2Ø        -
Ø87 Ø4A1 CA46Ø4                JZ    REPEAT     -- IF PEEK(#FDØØ) IAND #2=Ø*
Ø88 Ø4A4 C3ØC Ø4               JMP   START      -- GOTO 1Ø          *THEN 5Ø
Ø89 Ø4A7 Ø6ØØ        RANDOM    MVI   B,Ø
Ø9Ø Ø4A9 16Ø4                  MVI   D,4
Ø91 Ø4AB 3AØØFD      LOOP      LDA   :FDØØ
Ø92 Ø4AE Ø7                    RLC
Ø93 Ø4AF Ø7                    RLC
Ø94 Ø4BØ 3EØØ                  MVI   A,Ø
Ø95 Ø4B2 17                    RAL
Ø96 Ø4B3 BØ                    ORA   B
Ø97 Ø4B4 15                    DCR   D
Ø98 Ø4B5 C8                    RZ
Ø99 Ø4B6 17                    RAL
1ØØ Ø4B7 47                    MOV   B,A
1Ø1 Ø4B8 C3ABØ4                JMP   LOOP
1Ø2                  *
1Ø3 Ø4BB ØØ          X         DATA  Ø
1Ø4 Ø4BC ØØ          Y         DATA  Ø
1Ø5 Ø4BD ØØ          XS        DATA  Ø
1Ø6 Ø4BE ØØ          YS        DATA  Ø
1Ø7 Ø4BF ØØ          XM        DATA  Ø
1Ø8 Ø4CØ ØØ          YM        DATA  Ø
1Ø9 Ø4C1 ØØ          DX        DATA  Ø
11Ø Ø4C2 ØØ          DY        DATA  Ø
111                  *
112 Ø4C3                       END
```

BASICODE GERMANY

```
1       PRINT CHR$(12)
2       PRINT "BASICODE BY TH V LIESHOUT"
3       PRINT :PRINT " 1 LOADING BASICODE STANDARD (MID$-CORRECTED)"
4       PRINT :PRINT " 2 SAVING BASICODE STANDARD (MID$-CORRECTED)"
5       PRINT :PRINT " 3 LOADING BASICODE (NOT CORRECTED)"
6       PRINT :PRINT " 4 SAVING BASICODE (NOT CORRECTED)"
7       PRINT :PRINT " 5 CANCEL PROGRAM , RESTORE BASICODE"
8       PRINT :PRINT " 6 RESTORE BASICODE"
9       PRINT :PRINT " 7 RUN PROGRAM":GOTO 11
10      GOTO 1000
11      PRINT :PRINT "TYPE 1-7"
12      A=GETC:IF A<49.0 OR A>56.0 THEN 12
13      PRINT CHR$(12):ON A-48 GOTO 14,15,16,17,18,19,10
14      PRINT "TYPE CALLM #306":PRINT :END
15      CALLM #30C:LIST 1000-:CALLM #30F:GOTO 1
16      PRINT "TYPE CALLM #303":PRINT :END
17      CALLM #309:LIST:CALLM #30F:GOTO 1
18      PRINT "TYPE CALLM #318":PRINT :END
19      PRINT "TYPE CALLM #31B":PRINT :END
20      CLEAR A:GOTO 1010
100     PRINT CHR$(12):RETURN
110     IF HO>=0 AND VE>=0 AND HO<=39 AND VE<=23 THEN CURSOR HO,23-VE
111     IF HO>39 THEN HO=39.0
112     IF VE>23.0 THEN VE=23.0
113     RETURN
120     HO=CURX:VE=23.0-CURY:RETURN
200     O=GETC:IN$=CHR$(O):IF O=0 THEN IN$=""
201     RETURN
210     GOSUB 200:IF O=0 THEN 210
211     RETURN
250     ENVELOPE 0 15:SOUND 1 0 15 0 FREQ(1500.0):WAIT TIME 10:SOUND OFF :RETURN
260     RV=RND(1.0):RETURN
270     FR=FRE:RETURN
300     SR$=RIGHT$(STR$(SR),LEN(STR$(SR))-1.0):IF SR<0.0 THEN SR$="-"+SR$
301     RETURN
310     OS=ABS(SR)+0.5*10.0^(-CN):OH=INT(OS):OF=OS-OH+1.0:SR$="":IF OS>=1E9 THEN 3
18
311     IF CN=0.0 THEN OF$="":GOTO 315
312     IF OF=1.0 THEN OF$=".":GOTO 314
313     OF$=RIGHT$(STR$(OF)+"000000000",LEN(STR$(OF))+7.0):OF$=LEFT$(OF$,CN+1)
314     IF LEN(OF$)<CN+1 THEN OF$=OF$+"0":GOTO 314
315     SR$=RIGHT$(STR$(OH),LEN(STR$(OH))-1.0):SR$=LEFT$(SR$,LEN(SR$)-2)+OF$:IF SR
<0.0 AND VAL(SR$)<>0.0 THEN SR$="-"+SR$
316     IF LEN(SR$)<CT THEN SR$=" "+SR$:GOTO 316
317     IF LEN(SR$)>CT THEN SR$=""
318     IF LEN(SR$)<CT THEN SR$=SR$+"*":GOTO 318
319     RETURN
350     POKE #131,3:PRINT SR$;:POKE #131,1:RETURN
360     POKE #131,3:PRINT SR$:POKE #131,1:RETURN
1000    A=900.0:GOSUB 20:REM TEST BASICODE 2
1010    REM * WDR COMPUTERCLUB *
1020    REM * H.+ M.FILLINGER  *
1030    REM * 30.12.1983       *
1040    REM
1050    GOSUB 100
1060    GOSUB 20000
1070    DIM F(17.0),A(17.0)
1080    FOR I=1.0 TO 17.0
1090    READ F(I):A(I)=0.0
1100    NEXT I
1110    ER=0.0
1120    EF=0.0
1130    SE=0.0:I=0.0
1140    REM
1500    REM * VARIABLENLISTE *
```

```
1510  REM F(I)=ANTWORTKENNZAHL
1520  REM A(I)=ANTWORTAUSWAHL
1530  REM ER  =ANZAHL RICHTIGE ANTWORTEN
1540  REM EF  =ANZAHL FALSCHE ANTWORTEN
1550  REM SE  =AUSWERTUNG I.V.H.
1560  REM
1800  REM * ERLAEUTERUNG *
1810  PRINT "DIESES PROGRAMM SOLL IHNEN"
1820  PRINT "HELFEN MIT DEN UNTERPRO-"
1830  PRINT "GRAMMEN DES BASICODE 2"
1840  PRINT "VERTRAUT ZU WERDEN."
1850  PRINT "ZU JEDER FRAGE STEHEN DREI"
1860  PRINT "ANTWORTKENNZIFFERN ZUR"
1870  PRINT "AUSWAHL. EINE DAVON GEBEN"
1880  PRINT "SIE BITTE ALS IHRE ANTWORT"
1890  PRINT "EIN."
1900  PRINT
1910  PRINT "WENN WEITER, BITTE 'RETURN'"
1920  INPUT "EINGEBEN";X$
1930  GOSUB 100
1940  REM
2000  REM * TEST *
2010  PRINT "WELCHE VARIABLEN SIND IM"
2020  PRINT "BASICODE 2 MIT DOPPELTER"
2030  PRINT "GENAUIGKEIT DEFINIERT ?"
2040  PRINT
2050  PRINT "-1- ALLE"
2060  PRINT "-2- S"
2070  PRINT "-3- O"
2080  INPUT A(1.0)
2090  GOSUB 21000
2100  PRINT "WELCHE WERTE HABEN DIE"
2110  PRINT "VARIABLEN HO UND VE "
2120  PRINT "NACH AUSFUEHRUNG DES BEFEHLS"
2130  PRINT "GOSUB 100"
2140  PRINT
2150  PRINT "-1- :VE=1.0:HO=1.0"
2160  PRINT "-2- :VE=0.0:HO=0.0"
2170  PRINT "-3- DIE VARIABLEN WERDEN"
2180  PRINT "    DURCH DIE ANWEISUNG NICHT"
2190  PRINT "    VERAENDERT"
2200  INPUT A(2.0)
2210  GOSUB 21000
2230  PRINT "SIE WOLLEN DEN CURSOR ZUR"
2240  PRINT "UNTERSTEN LINKEN POSITION"
2250  PRINT "DES BILDSCHIRMES, DIE IM"
2260  PRINT "BASICODE 2 ZUGELASSEN IST,"
2270  PRINT "FUEHREN. WIE LAUTET DIE ANWEISUNG"
2290  PRINT
2300  PRINT "-1- HO=24:VE=39:GOSUB 120"
2310  PRINT "-2- HO=0:VE=23:GOSUB 110"
2320  PRINT "-3- HO=1.0:VE=24.0:GOSUB 110"
2330  INPUT A(3.0)
2340  GOSUB 21000
2350  PRINT "WELCHE VARIABLE DIENT ZUR"
2360  PRINT "RESERVIERUNG DES SPEICHER-"
2370  PRINT "PLATZES FUER ZEICHENKETTEN"
2380  PRINT "UND MIT WELCHEM BEFEHL WIRD"
2390  PRINT "DIES ERREICHT?"
2400  PRINT
2410  PRINT "-1- VARIABLE=A,BEFEHL=GOTO 20"
2420  PRINT "-2- VARIABLE=FR,BEFEHL=GOSUB 270"
2430  PRINT "-3- VARIABLE=A,BEFEHL=GOSUB 20"
2440  INPUT A(4.0)
2450  GOSUB 21000
2460  PRINT "WELCHE AUSSAGE IST FALSCH"
```

```
2470  PRINT
2480  PRINT "-1- DIE ANWEISUNG GOSUB 250"
2490  PRINT "    BEWIRKT DIE AUSGABE EINES"
2500  PRINT "    AKUSTISCHEN SIGNALS,SOWEIT"
2510  PRINT "    IHR RECHNER DIESE MOEGLICH-"
2520  PRINT "    KEIT HAT."
2530  PRINT "-2- GOSUB 260:PRINT RV"
2540  PRINT "    GIBT EINE ZUFALLSZAHL AUS,"
2550  PRINT "    INTERVALL : RV>=0 UND RV<=1"
2570  PRINT "-3- GOSUB 270, LAEDT DIE VARIABLE"
2580  PRINT "    FR MIT DEM AKTUELLEN FREIEN"
2590  PRINT "    ZEICHENKETTEN-SPEICHERPLATZ"
2600  PRINT
2610  INPUT A(5.0)
2620  GOSUB 21000
2630  PRINT "WELCHE AUSSAGE IST RICHTIG ZUR"
2640  PRINT "BEDINGUNGSANWEISUNG :"
2650  PRINT "IF I>2 AND I<0 THEN PRINT ELSE END"
2660  PRINT
2670  PRINT "-1- BOOLSCHE OPERATOREN SIND"
2680  PRINT "    UNZULAESSIG"
2690  PRINT "-2- DIE ANWEISUNG IST ZULAESSIG"
2700  PRINT "-3- ELSE IST UNZULAESSIG"
2720  INPUT A(6.0)
2730  GOSUB 21000
2740  PRINT "WELCHES UNTERPROGRAMM STEUERT IHREN"
2750  PRINT "DRUCKER SO, DASS DIE ANWWEISUNG DEM"
2760  PRINT "BEFEHL <<PRINT TAB(20) A$>>ENTSPRICHT"
2770  PRINT "-1- FOR I=1 TO 20"
2780  PRINT "      A$=' '+A$"
2790  PRINT "    NEXT I:SR$=A$:GOSUB 350"
2800  PRINT "-2- SR$=A$"
2810  PRINT "    FOR I=1 TO 20"
2820  PRINT "      SR$=SR$+' '"
2830  PRINT "    NEXT I:GOSUB 350"
2840  PRINT "-3- FOR I=0 TO 20"
2850  PRINT "      A$=' '+A$"
2860  PRINT "    NEXT I:SR$=A$:GOSUB 350"
2870  INPUT A(7.0)
2880  GOSUB 21000
2890  PRINT "SIE WOLLEN IN DER 4.ZEILE DES"
2900  PRINT "BILDSCHIRMES DAS 9. ZEICHEN "
2910  PRINT "ALS LEERZEICHEN DARSTELLEN."
2920  PRINT "DER CURSOR BEFINDET SICH"
2930  PRINT "IN DER OBEREN LINKEN ECKE"
2940  PRINT "IHRES MONITORS. WELCHE ANWEI-"
2950  PRINT "SUNGEN BENUTZEN SIE?"
2960  PRINT "-1- HO=4:VE=9:GOSUB 110"
2970  PRINT "    PRINT ' '"
2980  PRINT "-2- HO=3:VE=8:GOSUB 120"
2990  PRINT "    PRINT ' '"
3000  PRINT "-3- HO=8:VE=3:GOSUB 110"
3010  PRINT "    PRINT ' '"
3020  PRINT
3030  INPUT A(8.0)
3040  GOSUB 21000
3050  PRINT "MIT DEN ANWEISUNGEN HO=8:VE=3"
3060  PRINT "GOSUB 110:PRINT ' ' HABEN SIE EINEN"
3070  PRINT "BESTIMMTEN PUNKT AUF DEM MONITOR ANGE-"
3080  PRINT "SPROCHEN. MIT DER NAECHSTEN ANWEISUNG"
3090  PRINT "SOLL DIE FOLGENDE POSITION DEN BUCHSTABEN"
3100  PRINT "'D' AUSWEISEN. WELCHE ANWEISUNG IST"
3110  PRINT "FALSCH ?"
3130  PRINT "-1- AENDERUNG DER VORHERIGEN ANWEISUNG"
3140  PRINT "    IN : HO=8:VE=3:GOSUB 110:PRINT ' ';"
3150  PRINT "    PRINT 'D'"
```

```
3160  PRINT "-2- PRINT 'D'"
3170  PRINT "-3- HO=HO+1:VE=3:GOSUB 110:PRINT 'D'"
3180  INPUT A(9.0)
3190  GOSUB 21000
3200  PRINT "WELCHES PROGRAMM ERZEUGT"
3210  PRINT "EINE ZUFALLSZAHL L MIT DEM"
3220  PRINT "INTERVALL L>50 UND L<=100"
3230  PRINT "AUSSERDEM SOLL L EINE INTEGER"
3240  PRINT "ZAHL SEIN !"
3250  PRINT "-1- GOSUB 260:L=INT(RV*50)"
3260  PRINT "-2- GOSUB 260:L=INT(RV*50*2+.1)"
3270  PRINT "-3- GOSUB 260:L=INT(RV*50+51)"
3290  INPUT A(10.0)
3300  GOSUB 21000
3310  PRINT "WELCHE AUSSAGE IST RICHTIG"
3330  PRINT "-1- DIE ANWEISUNG GOSUB 200"
3340  PRINT "     IST IDENTISCH MIT DER"
3350  PRINT "     ANWEISUNG INPUT IN$"
3360  PRINT "-2- DIE ANWEISUNG GOSUB 200"
3370  PRINT "     UNTERSCHEIDET SICH VON DER"
3380  PRINT "     ANWEISUNG GOSUB 210 DARIN,"
3390  PRINT "     DASS BEI GOSUB 210 ABGEWARTET"
3400  PRINT "     WIRD, BIS DER BENUTZER EINE"
3410  PRINT "     TASTE GEDRUECKT HAT."
3420  PRINT "-3- DIE ANWEISUNGEN GOSUB 200"
3430  PRINT "     UND GOSUB 210 UNTERSCHEIDEN"
3440  PRINT "     SICH DADURCH, DASS BEI GOSUB"
3450  PRINT "     200 MEHRERE ZEICHEN EINGEGEBEN"
3460  PRINT "     UND DER VARIABLEN IN$ ZUGEORDNET"
3470  PRINT "     WERDEN.":: INPUT A(11.0)
3480  GOSUB 21000
3490  PRINT "X=23.678 WIE LAUTET DIE RICHTIGE"
3500  PRINT "ANWEISUNG UM X FORMATIERT MIT 2"
3510  PRINT "STELLEN HINTER DEM DEZIMALPUNKT"
3520  PRINT "DARZUSTELLEN ?"
3530  PRINT "-1- CT=6:CN=2:SR=X:GOSUB 310:PRINT SR$"
3540  PRINT "-2- CT=5:CN=2:SR=X:GOSUB 310:PRINT X"
3550  PRINT "-3- CT=4:CN=2:SR=X:GOSUB 310:PRINT SR$"
3560  INPUT A(12.0)
3570  GOSUB 21000
3580  PRINT "WELCHE AUSSAGE IST RICHTIG"
3590  PRINT "-1- DIE LAENGE EINER PROGRAMM-"
3600  PRINT "     ZEILE IST VOM BENUTZER FREI"
3610  PRINT "     BESTIMMBAR."
3620  PRINT "-2- EINE PROGRAMMZEILE DARF"
3630  PRINT "     EINSCHLIESSLICH DER "
3640  PRINT "     ZEILENNUMMER 60 ZEICHEN"
3650  PRINT "     NICHT UEBERSCHREITEN"
3660  PRINT "-3- EINE PROGRAMMZEILE DARF"
3670  PRINT "     EINSCHLIESSLICH DER "
3680  PRINT "     ZEILENNUMMER 40 ZEICHEN"
3690  PRINT "     NICHT UEBERSCHREITEN."
3700  INPUT A(13.0)
3710  GOSUB 21000
3720  PRINT "WELCHE AUSSAGE ZUR NACHFOLGENDEN"
3730  PRINT "ANWEISUNG IST RICHTIG"
3740  PRINT "  FOR S=1 TO 10 STEP 2"
3750  PRINT "     ....."
3760  PRINT "  NEXT S"
3770  PRINT "-1- DIE VARIABLE S IST MIT DOPPELTER"
3780  PRINT "     GENAUIGKEIT DEFFINIERT UND DARF"
3790  PRINT "     DESHALB NICHT ALS LAUFVARIABLE"
3800  PRINT "     BENUTZT WERDEN."
3810  PRINT "-2- DIE ANWEISUNG IST IN DIESER"
3820  PRINT "     FORM ZULAESSIG"
3830  PRINT "-3- STEP 2 IST UNZULAESSIG"
```

```
3840  INPUT A(14.0)
3850  GOSUB 21000
3860  PRINT "WELCHE AUSSAGE IST FALSCH"
3870  PRINT "-1- DIE ZEILENNUMMER 1000 IST DIE"
3880  PRINT "     ERSTE PROGRAMMZEILE JEDES"
3890  PRINT "     PROGRAMMES."
3900  PRINT "-2- DIE ZWEITE ZEILENNUMMER"
3910  PRINT "     IST VOM BENUTZER FREI"
3920  PRINT "     BESTIMMBAR."
3930  PRINT "-3- DIE ZWEITE ZEILENNUMMER MUSS"
3940  PRINT "     1010 SEIN, DA DAS UNTERPPROGRAMM"
3950  PRINT "     IN DAS VON ZEILE 1000 VERZWEIGT"
3960  PRINT "     WIRD, SEINERSEITS IN ZEILE 1010"
3970  PRINT "     VERZWEIGT"
3980  INPUT A(15.0)
3990  GOSUB 21000
4000  PRINT "WELCHE AUSSAGE IST RICHTIG"
4010  PRINT "-1- IN BASICODE 2 KOENNEN 4 DIMEN-"
4020  PRINT "     SIONALE FELDER DIMENSIONIERT"
4030  PRINT "     WERDEN."
4040  PRINT "-2- JEDES BENUTZTE FELD MUSS VORHER"
4050  PRINT "     DIMENSIONIERT WERDEN."
4060  PRINT "-3- DAS ELEMENT 0 IST NICHT"
4070  PRINT "     ERLAUBT."
4080  INPUT A(16.0)
4090  GOSUB 21000
4100  PRINT "WELCHE AUSSAGE IST RICHTIG"
4110  PRINT "-1- VARIABLENNAMEN DUERFEN"
4120  PRINT "     NUR AUS EINEM ZEICHEN"
4130  PRINT "     (GROSSER BUCHSTABEN)"
4140  PRINT "     BESTEHEN."
4150  PRINT "-2- VARIABLENNAMEN DUERFEN"
4160  PRINT "     HOECHSTENS AUS ZWEI ZEICHEN"
4170  PRINT "     BESTEHEN, WOBEI DAS ERSTE"
4180  PRINT "     EIN GROSSBUCHSTABE SEIN"
4190  PRINT "     MUSS. DAS ZWEITE ZEICHEN"
4200  PRINT "     KANN FREI BESTIMMT WERDEN."
4210  PRINT "-3- WIE -2- JEDOCH DAS ZWEITE"
4220  PRINT "     ZEICHEN MUSS EIN GROSSBUCH-"
4230  PRINT "     STABE ODER EINE ZIFFER SEIN."
4240  INPUT A(17.0)
4250  GOSUB 21000
10000 REM * AUSWERTUNG *
10010 PRINT
10020 SE=INT(ER*100.0/I)
10030 PRINT "SIE HABEN";ER;"VON";I;"FRAGEN"
10040 PRINT "RICHTIG BEANTWORTET ="
10050 PRINT SE;" %"
10060 SE=INT(EF*100.0/I)
10070 PRINT "SIE HABEN";EF;"VON";I;"FRAGEN"
10080 PRINT "FALSCH BEANTWORTET ="
10090 PRINT SE;" %"
10100 REM
11000 PRINT "WIR FREUEN UNS AUF DIE ZUSENDUNG"
11010 PRINT "EINES VON IHNEN ERSTELLTEN "
11020 PRINT "PROGRAMMS IM BASICODE 2."
11030 PRINT "ES WAERE SCHOEN, WENN SIE EIN"
11040 PRINT "LISTING BEIFUEGEN KOENNTEN."
11050 END
20000 REM TITEL
20010 DIM X$(3.0)
20020 X$(0.0)="DER WDR COMPUTERCLUB"
20030 X$(1.0)="   PRAESENTIERT :"
20040 X$(2.0)="B A S I C O D E  2"
20050 X$(3.0)=" EIN TESTPROGRAMM"
20060 FOR X=0.0 TO 3.0
```

```
20070 L=LEN(X$(X))
20080 FOR I=1.0 TO L
20090 VE=3.0+X
20100 HO=0.0+I
20110 GOSUB 110
20120 PRINT MID$(X$(X),-1+I,1);
20130 NEXT I
20140 NEXT X
20150 FOR I=1.0 TO 1000.0:NEXT I
20160 FOR X=3.0 TO 0.0 STEP -1.0
20170 L=LEN(X$(X))
20180 FOR I=L TO 1.0 STEP -1.0
20190 VE=3.0+X
20200 HO=0.0+I
20210 GOSUB 110
20220 PRINT " ";
20230 NEXT I
20240 NEXT X
20250 GOSUB 100
20260 RETURN
21000 REM * AUSWERTUNG *
21010 I=I+1.0
21020 IF A(I)=F(I) THEN ER=ER+1.0
21030 IF A(I)<>F(I) THEN EF=EF+1.0
21040 IF A(I)=F(I) THEN GOTO 21200
21050 REM FALSCHE ANTWORT
21060 PRINT "LEIDER HABEN SIE DIE FRAGE"
21070 PRINT "FALSCH BEANTWORTET !"
21080 PRINT "DIE RICHTIGE ANTWORT WAR :"
21090 PRINT F(I)
21100 GOTO 21230
21200 REM RICHTIGE ANTWORT
21210 PRINT "DIE FRAGE WURDE RICHTIG"
21220 PRINT "BEANTWORTET !!"
21230 INPUT "WENN WEITER. BITTE 'RETURN'";X$
21240 GOSUB 100
21250 RETURN
25000 REM * ANTWORTEN *
25010 DATA 2.3,2,1,2,3,1,3,2,3
25020 DATA 2,1,2,1,2,2,3
```

MEMORYMAP MODE 5/6

255	BFEF BFEE	BFED BFE9 BFE5 BFE1 BFDD BFD9 BFD5 BFD1 BFCD BFC9 BFC5 BFC1 BFBD BFB9 BFB5 BFB1 BFAD BFA9 BFA5 BFA1 BF9D BF99
254	BF95 BF94	BF93 BF8F BF8B BF87 BF83 BF7F BF7B BF77 BF73 BF6F BF6B BF67 BF63 BF5F BF5B BF57 BF53 BF4F BF4B BF47 BF43 BF3F
253	BF3B BF3A	BF39 BF35 BF31 BF2D BF29 BF25 BF21 BF1D BF19 BF15 BF11 BF0D BF09 BF05 BF01 BEFD BEF9 BEF5 BEF1 BEED BEE9 BEE5
252	BEE1 BEE0	BEDF BEDB BED7 BED3 BECF BECB BEC7 BEC3 BEBF BEBB BEB7 BEB3 BEAF BEAB BEA7 BEA3 BE9F BE9B BE97 BE93 BE8F BE8B
251	BE87 BE86	BE85 BE81 BE7D BE79 BE75 BE71 BE6D BE69 BE65 BE61 BE5D BE59 BE55 BE51 BE4D BE49 BE45 BE41 BE3D BE39 BE35 BE31
250	BE2D BE2C	BE2B BE27 BE23 BE1F BE1B BE17 BE13 BE0F BE0B BE07 BE03 BDFF BDFB BDF7 BDF3 BDEF BDEB BDE7 BDE3 BDDF BDDB BDD7
249	BDD3 BDD2	BDD1 BDCD BDC9 BDC5 BDC1 BDBD BDB9 BDB5 BDB1 BDAD BDA9 BDA5 BDA1 BD9D BD99 BD95 BD91 BD8D BD89 BD85 BD81 BD7D
248	BD79 BD78	BD77 BD73 BD6F BD6B BD67 BD63 BD5F BD5B BD57 BD53 BD4F BD4B BD47 BD43 BD3F BD3B BD37 BD33 BD2F BD2B BD27 BD23
247	BD1F BD1E	BD1D BD19 BD15 BD11 BD0D BD09 BD05 BD01 BCFD BCF9 BCF5 BCF1 BCED BCE9 BCE5 BCE1 BCDD BCD9 BCD5 BCD1 BCCD BCC9
246	BCC5 BCC4	BCC3 BC8F BC8B BC87 BC83 BC7F BC7B BC77 BC73 BC6F BC6B BC67 BC63 BC5F BC5B BC57 BC53 BC4F BC4B BC47 BC43 BC3F
245	BC3B BC3A	BC39 BC35 BC31 BC2D BC29 BC25 BC21 BC1D BC19 BC15 BC11 BC0D BC09 BC05 BC01 BBFD BBF9 BBF5 BBF1 BBED BBE9 BBE5
244	BBE1 BBE0	BBDF BBDB BBD7 BBD3 BBCF BBCB BBC7 BBC3 BBBF BBBB BBB7 BBB3 BBAF BBAB BBA7 BBA3 BB9F BB9B BB97 BB93 BB8F BB8B
243	BB87 BB86	BB85 BB81 BB7D BB79 BB75 BB71 BB6D BB69 BB65 BB61 BB5D BB59 BB55 BB51 BB4D BB49 BB45 BB41 BB3D BB39 BB35 BB31
242	BB2D BB2C	BB2B BB27 BB23 BB1F BB1B BB17 BB13 BB0F BB0B BB07 BB03 BAFF BAFB BAF7 BAF3 BAEF BAEB BAE7 BAE3 BADF BADB BAD7
241	BAD3 BAD2	BAD1 BACD BAC9 BAC5 BAC1 BABD BAB9 BAB5 BAB1 BAAD BAA9 BAA5 BAA1 BA9D BA99 BA95 BA91 BA8D BA89 BA85 BA81 BA7D
240	BA79 BA78	BA77 BA73 BA6F BA6B BA67 BA63 BA5F BA5B BA57 BA53 BA4F BA4B BA47 BA43 BA3F BA3B BA37 BA33 BA2F BA2B BA27 BA23
239	BA1F BA1E	BA1D BA19 BA15 BA11 BA0D BA09 BA05 BA01 B9FD B9F9 B9F5 B9F1 B9ED B9E9 B9E5 B9E1 B9DD B9D9 B9D5 B9D1 B9CD B9C9
238	B9C5 B9C4	B9C3 B9BF B9BB B9B7 B9B3 B9AF B9AB B9A7 B9A3 B99F B99B B997 B993 B98F B98B B987 B983 B97F B97B B977 B973 B96F
237	B96B B96A	B969 B965 B961 B95D B959 B955 B951 B94D B949 B945 B941 B93D B939 B935 B931 B92D B929 B925 B921 B91D B919 B915
236	B911 B910	B90F B90B B907 B903 B8FF B8FB B8F7 B8F3 B8EF B8EB B8E7 B8E3 B8DF B8DB B8D7 B8D3 B8CF B8CB B8C7 B8C3 B8BF B8BB
235	B8B7 B8B6	B8B5 B8B1 B8AD B8A9 B8A5 B8A1 B89D B899 B895 B891 B88D B889 B885 B881 B87D B879 B875 B871 B86D B869 B865 B861
234	B85D B85C	B85B B857 B853 B84F B84B B847 B843 B83F B83B B837 B833 B82F B82B B827 B823 B81F B81B B817 B813 B80F B80B B807
233	B803 B802	B801 B7FD B7F9 B7F5 B7F1 B7ED B7E9 B7E5 B7E1 B7DD B7D9 B7D5 B7D1 B7CD B7C9 B7C5 B7C1 B7BD B7B9 B7B5 B7B1 B7AD
232	B7A9 B7A8	B7A7 B7A3 B79F B79B B797 B793 B78F B78B B787 B783 B77F B77B B777 B773 B76F B76B B767 B763 B75F B75B B757 B753
231	B74F B74E	B74D B749 B745 B741 B73D B739 B735 B731 B72D B729 B725 B721 B71D B719 B715 B711 B70D B709 B705 B701 B6FD B6F9
230	B6F5 B6F4	B6F3 B6EF B6EB B6E7 B6E3 B6DF B6DB B6D7 B6D3 B6CF B6CB B6C7 B6C3 B6BF B6BB B6B7 B6B3 B6AF B6AB B6A7 B6A3 B69F
229	B69B B69A	B699 B695 B691 B68D B689 B685 B681 B67D B679 B675 B671 B66D B669 B665 B661 B65D B659 B655 B651 B64D B649 B645
228	B641 B640	B63F B63B B637 B633 B62F B62B B627 B623 B61F B61B B617 B613 B60F B60B B607 B603 B5FF B5FB B5F7 B5F3 B5EF B5EB
227	B5E7 B5E6	B5E5 B5E1 B5DD B5D9 B5D5 B5D1 B5CD B5C9 B5C5 B5C1 B5BD B5B9 B5B5 B5B1 B5AD B5A9 B5A5 B5A1 B59D B599 B595 B591
226	B58D B58C	B58B B587 B583 B57F B57B B577 B573 B56F B56B B567 B563 B55F B55B B557 B553 B54F B54B B547 B543 B53F B53B B537
225	B533 B532	B531 B52D B529 B525 B521 B51D B519 B515 B511 B50D B509 B505 B501 B4FD B4F9 B4F5 B4F1 B4ED B4E9 B4E5 B4E1 B4DD
224	B4D9 B4D8	B4D7 B4D3 B4CF B4CB B4C7 B4C3 B4BF B4BB B4B7 B4B3 B4AF B4AB B4A7 B4A3 B49F B49B B497 B493 B48F B48B B487 B483
223	B47F B47E	B47D B479 B475 B471 B46D B469 B465 B461 B45D B459 B455 B451 B44D B449 B445 B441 B43D B439 B435 B431 B42D B429
222	B425 B424	B423 B41F B41B B417 B413 B40F B40B B407 B403 B3FF B3FB B3F7 B3F3 B3EF B3EB B3E7 B3E3 B3DF B3DB B3D7 B3D3 B3CF
221	B3CB B3CA	B3C9 B3C5 B3C1 B3BD B3B9 B3B5 B3B1 B3AD B3A9 B3A5 B3A1 B39D B399 B395 B391 B38D B389 B385 B381 B37D B379 B375
220	B371 B370	B36F B36B B367 B363 B35F B35B B357 B353 B34F B34B B347 B343 B33F B33B B337 B333 B32F B32B B327 B323 B31F B31B
219	B317 B316	B315 B311 B30D B309 B305 B301 B2FD B2F9 B2F5 B2F1 B2ED B2E9 B2E5 B2E1 B2DD B2D9 B2D5 B2D1 B2CD B2C9 B2C5 B2C1
218	B2BD B2BC	B2BB B2B7 B2B3 B2AF B2AB B2A7 B2A3 B29F B29B B297 B293 B28F B28B B287 B283 B27F B27B B277 B273 B26F B26B B267
217	B263 B262	B261 B25D B259 B255 B251 B24D B249 B245 B241 B23D B239 B235 B231 B22D B229 B225 B221 B21D B219 B215 B211 B20D
216	B209 B208	B207 B203 B1FF B1FB B1F7 B1F3 B1EF B1EB B1E7 B1E3 B1DF B1DB B1D7 B1D3 B1CF B1CB B1C7 B1C3 B1BF B1BB B1B7 B1B3
215	B1AF B1AE	B1AD B1A9 B1A5 B1A1 B19D B199 B195 B191 B18D B189 B185 B181 B17D B179 B175 B171 B16D B169 B165 B161 B15D B159
214	B155 B154	B153 B14F B14B B147 B143 B13F B13B B137 B133 B12F B12B B127 B123 B11F B11B B117 B113 B10F B10B B107 B103 B0FF
213	B0FB B0FA	B0F9 B0F5 B0F1 B0ED B0E9 B0E5 B0E1 B0DD B0D9 B0D5 B0D1 B0CD B0C9 B0C5 B0C1 B0BD B0B9 B0B5 B0B1 B0AD B0A9 B0A5
212	B0A1 B0A0	B09F B09B B097 B093 B08F B08B B087 B083 B07F B07B B077 B073 B06F B06B B067 B063 B05F B05B B057 B053 B04F B04B
211	B047 B046	B045 B041 B03D B039 B035 B031 B02D B029 B025 B021 B01D B019 B015 B011 B00D B009 B005 B001 AFFD AFF9 AFF5 AFF1
210	AFED AFEC	AFEB AFE7 AFE3 AFDF AFDB AFD7 AFD3 AFCF AFCB AFC7 AFC3 AFBF AFBB AFB7 AFB3 AFAF AFAB AFA7 AFA3 AF9F AF9B AF97
209	AF93 AF92	AF91 AF8D AF89 AF85 AF81 AF7D AF79 AF75 AF71 AF6D AF69 AF65 AF61 AF5D AF59 AF55 AF51 AF4D AF49 AF45 AF41 AF3D
208	AF39 AF38	AF37 AF33 AF2F AF2B AF27 AF23 AF1F AF1B AF17 AF13 AF0F AF0B AF07 AF03 AEFF AEFB AEF7 AEF3 AEEF AEEB AEE7 AEE3
207	AEDF AEDE	AEDD AED9 AED5 AED1 AECD AEC9 AEC5 AEC1 AEBD AEB9 AEB5 AEB1 AEAD AEA9 AEA5 AEA1 AE9D AE99 AE95 AE91 AE8D AE89
206	AE85 AE84	AE83 AE7F AE7B AE77 AE73 AE6F AE6B AE67 AE63 AE5F AE5B AE57 AE53 AE4F AE4B AE47 AE43 AE3F AE3B AE37 AE33 AE2F
205	AE2B AE2A	AE29 AE25 AE21 AE1D AE19 AE15 AE11 AE0D AE09 AE05 AE01 ADFD ADF9 ADF5 ADF1 ADED ADE9 ADE5 ADE1 ADDD ADD9 ADD5
204	ADD1 ADD0	ADCF ADCB ADC7 ADC3 ADBF ADBB ADB7 ADB3 ADAF ADAB ADA7 ADA3 AD9F AD9B AD97 AD93 AD8F AD8B AD87 AD83 AD7F AD7B
203	AD77 AD76	AD75 AD71 AD6D AD69 AD65 AD61 AD5D AD59 AD55 AD51 AD4D AD49 AD45 AD41 AD3D AD39 AD35 AD31 AD2D AD29 AD25 AD21
202	AD1D AD1C	AD1B AD17 AD13 AD0F AD0B AD07 AD03 ACFF ACFB ACF7 ACF3 ACEF ACEB ACE7 ACE3 ACDF ACDB ACD7 ACD3 ACCF ACCB ACC7
201	ACC3 ACC2	ACC1 ACBD ACB9 ACB5 ACB1 ACAD ACA9 ACA5 ACA1 AC9D AC99 AC95 AC91 AC8D AC89 AC85 AC81 AC7D AC79 AC75 AC71 AC6D
200	AC69 AC68	AC67 AC63 AC5F AC5B AC57 AC53 AC4F AC4B AC47 AC43 AC3F AC3B AC37 AC33 AC2F AC2B AC27 AC23 AC1F AC1B AC17 AC13
199	AC0F AC0E	AC0D AC09 AC05 AC01 ABFD ABF9 ABF5 ABF1 ABED ABE9 ABE5 ABE4 ABE3 ABDF ABDB ABD7 ABD3 ABCF ABCB ABC7 ABC3 ABBF
198	ABBB ABB7	ABB3 ABAF ABAB ABA7 ABA3 AB9F AB9B AB97 AB93 AB8F AB8B AB8A AB89 AB85 AB81 AB7D AB79 AB75 AB71 AB6D AB69 AB65
197	AB61 AB5D	AB59 AB55 AB51 AB4D AB49 AB45 AB41 AB3D AB39 AB35 AB31 AB30 AB2F AB2B AB27 AB23 AB1F AB1B AB17 AB13 AB0F AB0B
196	AB07 AB03	AAFF AAFB AAF7 AAF3 AAEF AAEB AAE7 AAE3 AADF AADB AAD7 AAD6 AAD5 AAD1 AACD AAC9 AAC5 AAC1 AABD AAB9 AAB5 AAB1
195	AAAD AAA9	AAA5 AAA1 A99D A999 A995 A991 A98D A989 A985 A981 A97D A97C A97B A977 A973 A96F A96B A967 A963 A95F A95B A957
194	A953 A94F	A94B A947 A943 A93F A93B A937 A933 A92F A92B A927 A923 A91F A91B A917 A913 A90F A90B A907 A903 A8FF A8FB A8F7
193	A8F3 A8EF	A8EB A8E7 A8E3 A8DF A8DB A8D7 A8D3 A8CF A8CB A8C7 A8C3 A8BF A8BB A8B7 A8B3 A8AF A8AB A8A7 A8A3 A89F A89B A897

120 DAInamic 84-21

192	A9C9 A9C8	A9C7 A9C3 A9BF A9BB A9B7 A9B3 A9AF A9AB A9A7 A9A3 A99F A99B A997 A993 A98F A98B A987 A983 A97F A97B A977 A973
191	A96F A96E	A96D A969 A965 A961 A95D A959 A955 A951 A94D A949 A945 A941 A93D A939 A935 A931 A92D A929 A925 A921 A91D A919
190	A915 A914	A913 A90F A90B A907 A903 A8FF A8FB A8F7 A8F3 A8EF A8EB A8E7 A8E3 A8DF A8DB A8D7 A8D3 A8CF A8CB A8C7 A8C3 A8BF
189	A8BB A8BA	A8B9 A8B5 A8B1 A8AD A8A9 A8A5 A8A1 A89D A899 A895 A891 A88D A889 A885 A881 A87D A879 A875 A871 A86D A869 A865
188	A861 A860	A85F A85B A857 A853 A84F A84B A847 A843 A83F A83B A837 A833 A82F A82B A827 A823 A81F A81B A817 A813 A80F A80B
187	A807 A806	A805 A801 A7FD A7F9 A7F5 A7F1 A7ED A7E9 A7E5 A7E1 A7DD A7D9 A7D5 A7D1 A7CD A7C9 A7C5 A7C1 A7BD A7B9 A7B5 A7B1
186	A7AD A7AC	A7AB A7A7 A7A3 A79F A79B A797 A793 A78F A78B A787 A783 A77F A77B A777 A773 A76F A76B A767 A763 A75F A75B A757
185	A753 A752	A751 A74D A749 A745 A741 A73D A739 A735 A731 A72D A729 A725 A721 A71D A719 A715 A711 A70D A709 A705 A701 A6FD
184	A6F9 A6F8	A6F7 A6F3 A6EF A6EB A6E7 A6E3 A6DF A6DB A6D7 A6D3 A6CF A6CB A6C7 A6C3 A6BF A6BB A6B7 A6B3 A6AF A6AB A6A7 A6A3
183	A69F A69E	A69D A699 A695 A691 A68D A689 A685 A681 A67D A679 A675 A671 A66D A669 A665 A661 A65D A659 A655 A651 A64D A649
182	A645 A644	A643 A63F A63B A637 A633 A62F A62B A627 A623 A61F A61B A617 A613 A60F A60B A607 A603 A5FF A5FB A5F7 A5F3 A5EF
181	A5EB A5EA	A5E9 A5E5 A5E1 A5DD A5D9 A5D5 A5D1 A5CD A5C9 A5C5 A5C1 A5BD A5B9 A5B5 A5B1 A5AD A5A9 A5A5 A5A1 A59D A599 A595
180	A591 A590	A58F A58B A587 A583 A57F A57B A577 A573 A56F A56B A567 A563 A55F A55B A557 A553 A54F A54B A547 A543 A53F A53B
179	A537 A536	A535 A531 A52D A529 A525 A521 A51D A519 A515 A511 A50D A509 A505 A501 A4FD A4F9 A4F5 A4F1 A4ED A4E9 A4E5 A4E1
178	A4DD A4DC	A4DB A4D7 A4D3 A4CF A4CB A4C7 A4C3 A4BF A4BB A4B7 A4B3 A4AF A4AB A4A7 A4A3 A49F A49B A497 A493 A48F A48B A487
177	A483 A482	A481 A47D A479 A475 A471 A46D A469 A465 A461 A45D A459 A455 A451 A44D A449 A445 A441 A43D A439 A435 A431 A42D
176	A429 A428	A427 A423 A41F A41B A417 A413 A40F A40B A407 A403 A3FF A3FB A3F7 A3F3 A3EF A3EB A3E7 A3E3 A3DF A3DB A3D7 A3D3
175	A3CF A3CE	A3CD A3C9 A3C5 A3C1 A3BD A3B9 A3B5 A3B1 A3AD A3A9 A3A5 A3A1 A39D A399 A395 A391 A38D A389 A385 A381 A37D A379
174	A375 A374	A373 A36F A36B A367 A363 A35F A35B A357 A353 A34F A34B A347 A343 A33F A33B A337 A333 A32F A32B A327 A323 A31F
173	A31B A31A	A319 A315 A311 A30D A309 A305 A301 A2FD A2F9 A2F5 A2F1 A2ED A2E9 A2E5 A2E1 A2DD A2D9 A2D5 A2D1 A2CD A2C9 A2C5
172	A2C1 A2C0	A2BF A2BB A2B7 A2B3 A2AF A2AB A2A7 A2A3 A29F A29B A297 A293 A28F A28B A287 A283 A27F A27B A277 A273 A26F A26B
171	A267 A266	A265 A261 A25D A259 A255 A251 A24D A249 A245 A241 A23D A239 A235 A231 A22D A229 A225 A221 A21D A219 A215 A211
170	A20D A20C	A20B A207 A203 A1FF A1FB A1F7 A1F3 A1EF A1EB A1E7 A1E3 A1DF A1DB A1D7 A1D3 A1CF A1CB A1C7 A1C3 A1BF A1BB A1B7
169	A1B3 A1B2	A1B1 A1AD A1A9 A1A5 A1A1 A19D A199 A195 A191 A18D A189 A185 A181 A17D A179 A175 A171 A16D A169 A165 A161 A15D
168	A159 A158	A157 A153 A14F A14B A147 A143 A13F A13B A137 A133 A12F A12B A127 A123 A11F A11B A117 A113 A10F A10B A107 A103
167	A0FF A0FE	A0FD A0F9 A0F5 A0F1 A0ED A0E9 A0E5 A0E1 A0DD A0D9 A0D5 A0D1 A0CD A0C9 A0C5 A0C1 A0BD A0B9 A0B5 A0B1 A0AD A0A9
166	A0A5 A0A4	A0A3 A09F A09B A097 A093 A08F A08B A087 A083 A07F A07B A077 A073 A06F A06B A067 A063 A05F A05B A057 A053 A04F
165	A04B A04A	A049 A045 A041 A03D A039 A035 A031 A02D A029 A025 A021 A01D A019 A015 A011 A00D A009 A005 A001 9FFD 9FF9 9FF5
164	9FF1 9FF0	9FEF 9FEB 9FE7 9FE3 9FDF 9FDB 9FD7 9FD3 9FCF 9FCB 9FC7 9FC3 9FBF 9FBB 9FB7 9FB3 9FAF 9FAB 9FA7 9FA3 9F9F 9F9B
163	9F97 9F96	9F95 9F91 9F8D 9F89 9F85 9F81 9F7D 9F79 9F75 9F71 9F6D 9F69 9F65 9F61 9F5D 9F59 9F55 9F51 9F4D 9F49 9F45 9F41
162	9F3D 9F3C	9F3B 9F37 9F33 9F2F 9F2B 9F27 9F23 9F1F 9F1B 9F17 9F13 9F0F 9F0B 9F07 9F03 9EFF 9EFB 9EF7 9EF3 9EEF 9EEB 9EE7
161	9EE3 9EE2	9EE1 9EDD 9ED9 9ED5 9ED1 9ECD 9EC9 9EC5 9EC1 9EBD 9EB9 9EB5 9EB1 9EAD 9EA9 9EA5 9EA1 9E9D 9E99 9E95 9E91 9E8D
160	9E89 9E88	9E87 9E83 9E7F 9E7B 9E77 9E73 9E6F 9E6B 9E67 9E63 9E5F 9E5B 9E57 9E53 9E4F 9E4B 9E47 9E43 9E3F 9E3B 9E37 9E33
159	9E2F 9E2E	9E2D 9E29 9E25 9E21 9E1D 9E19 9E15 9E11 9E0D 9E09 9E05 9E01 9DFD 9DF9 9DF5 9DF1 9DED 9DE9 9DE5 9DE1 9DDD 9DD9
158	9DD5 9DD4	9DD3 9DCF 9DCB 9DC7 9DC3 9DBF 9DBB 9DB7 9DB3 9DAF 9DAB 9DA7 9DA3 9D9F 9D9B 9D97 9D93 9D8F 9D8B 9D87 9D83 9D7F
157	9D7B 9D7A	9D79 9D75 9D71 9D6D 9D69 9D65 9D61 9D5D 9D59 9D55 9D51 9D4D 9D49 9D45 9D41 9D3D 9D39 9D35 9D31 9D2D 9D29 9D25
156	9D21 9D20	9D1F 9D1B 9D17 9D13 9D0F 9D0B 9D07 9D03 9CFF 9CFB 9CF7 9CF3 9CEF 9CEB 9CE7 9CE3 9CDF 9CDB 9CD7 9CD3 9CCF 9CCB
155	9CC7 9CC6	9CC5 9CC1 9CBD 9CB9 9CB5 9CB1 9CAD 9CA9 9CA5 9CA1 9C9D 9C99 9C95 9C91 9C8D 9C89 9C85 9C81 9C7D 9C79 9C75 9C71
154	9C6D 9C6C	9C6B 9C67 9C63 9C5F 9C5B 9C57 9C53 9C4F 9C4B 9C47 9C43 9C3F 9C3B 9C37 9C33 9C2F 9C2B 9C27 9C23 9C1F 9C1B 9C17
153	9C13 9C12	9C11 9C0D 9C09 9C05 9C01 9BFD 9BF9 9BF5 9BF1 9BED 9BE9 9BE5 9BE1 9BDD 9BD9 9BD5 9BD1 9BCD 9BC9 9BC5 9BC1 9BBD
152	9BB9 9BB8	9BB7 9BB3 9BAF 9BAB 9BA7 9BA3 9B9F 9B9B 9B97 9B93 9B8F 9B8B 9B87 9B83 9B7F 9B7B 9B77 9B73 9B6F 9B6B 9B67 9B63
151	9B5F 9B5E	9B5D 9B59 9B55 9B51 9B4D 9B49 9B45 9B41 9B3D 9B39 9B35 9B31 9B2D 9B29 9B25 9B21 9B1D 9B19 9B15 9B11 9B0D 9B09
150	9B05 9B04	9B03 9AFF 9AFB 9AF7 9AF3 9AEF 9AEB 9AE7 9AE3 9ADF 9ADB 9AD7 9AD3 9ACF 9ACB 9AC7 9AC3 9ABF 9ABB 9AB7 9AB3 9AAF
149	9AAB 9AAA	9AA9 9AA5 9AA1 9A9D 9A99 9A95 9A91 9A8D 9A89 9A85 9A81 9A7D 9A79 9A75 9A71 9A6D 9A69 9A65 9A61 9A5D 9A59 9A55
148	9A51 9A50	9A4F 9A4B 9A47 9A43 9A3F 9A3B 9A37 9A33 9A2F 9A2B 9A27 9A23 9A1F 9A1B 9A17 9A13 9A0F 9A0B 9A07 9A03 99FF 99FB
147	99F7 99F6	99F5 99F1 99ED 99E9 99E5 99E1 99DD 99D9 99D5 99D1 99CD 99C9 99C5 99C1 99BD 99B9 99B5 99B1 99AD 99A9 99A5 99A1
146	999D 999C	999B 9997 9993 998F 998B 9987 9983 997F 997B 9977 9973 996F 996B 9967 9963 995F 995B 9957 9953 994F 994B 9947
145	9943 9942	9941 993D 9939 9935 9931 992D 9929 9925 9921 991D 9919 9915 9911 990D 9909 9905 9901 98FD 98F9 98F5 98F1 98ED
144	98E9 98E8	98E7 98E3 98DF 98DB 98D7 98D3 98CF 98CB 98C7 98C3 98BF 98BB 98B7 98B3 98AF 98AB 98A7 98A3 989F 989B 9897 9893
143	988F 988E	988D 9889 9885 9881 987D 9879 9875 9871 986D 9869 9865 9861 985D 9859 9855 9851 984D 9849 9845 9841 983D 9839
142	9835 9834	9833 982F 982B 9827 9823 981F 981B 9817 9813 980F 980B 9807 9803 97FF 97FB 97F7 97F3 97EF 97EB 97E7 97E3 97DF
141	97DB 97DA	97D9 97D5 97D1 97CD 97C9 97C5 97C1 97BD 97B9 97B5 97B1 97AD 97A9 97A5 97A1 979D 9799 9795 9791 978D 9789 9785
140	9781 9780	977F 977B 9777 9773 976F 976B 9767 9763 975F 975B 9757 9753 974F 974B 9747 9743 973F 973B 9737 9733 972F 972B
139	9727 9726	9725 9721 971D 9719 9715 9711 970D 9709 9705 9701 96FD 96F9 96F5 96F1 96ED 96E9 96E5 96E1 96DD 96D9 96D5 96D1
138	96CD 96CC	96CB 96C7 96C3 96BF 96BB 96B7 96B3 96AF 96AB 96A7 96A3 969F 969B 9697 9693 968F 968B 9687 9683 967F 967B 9677
137	9673 9672	9671 966D 9669 9665 9661 965D 9659 9655 9651 964D 9649 9645 9641 963D 9639 9635 9631 962D 9629 9625 9621 961D
136	9619 9618	9617 9613 960F 960B 9607 9603 95FF 95FB 95F7 95F3 95EF 95EB 95E7 95E3 95DF 95DB 95D7 95D3 95CF 95CB 95C7 95C3
135	95BF 95BE	95BD 95B9 95B5 95B1 95AD 95A9 95A5 95A1 959D 9599 9595 9591 958D 9589 9585 9581 957D 9579 9575 9571 956D 9569
134	9565 9564	9563 955F 955B 9557 9553 954F 954B 9547 9543 953F 953B 9537 9533 952F 952B 9527 9523 951F 951B 9517 9513 950F
133	950B 950A	9509 9505 9501 94FD 94F9 94F5 94F1 94ED 94E9 94E5 94E1 94DD 94D9 94D5 94D1 94CD 94C9 94C5 94C1 94BD 94B9 94B5
132	94B1 94B0	94AF 94AB 94A7 94A3 949F 949B 9497 9493 948F 948B 9487 9483 947F 947B 9477 9473 946F 946B 9467 9463 945F 945B
131	9457 9456	9455 9451 944D 9449 9445 9441 943D 9439 9435 9431 942D 9429 9425 9421 941D 9419 9415 9411 940D 9409 9405 9401
130	93FD 93FC	93FB 93F7 93F3 93EF 93EB 93E7 93E3 93DF 93DB 93D7 93D3 93CF 93CB 93C7 93C3 93BF 93BB 93B7 93B3 93AF 93AB 93A7
129	93A3 93A2	93A1 939D 9399 9395 9391 938D 9389 9385 9381 937D 9379 9375 9371 936D 9369 9365 9361 935D 9359 9355 9351 934D

128	9349 9348	9347 9343 933F 933B 9337 9333 932F 932B 9327 9323 931F 931B 9317 9313 930F 930B 9307 9303 92FF 92FB 92F7 92F3
127	92EF 92EE	92ED 92E9 92E5 92E1 92DD 92D9 92D5 92D1 92CD 92C9 92C5 92C1 92BD 92B9 92B5 92B1 92AD 92A9 92A5 92A1 929D 9299
126	9295 9294	9293 928F 928B 9287 9283 927F 927B 9277 9273 926F 926B 9267 9263 925F 925B 9257 9253 924F 924B 9247 9243 923F
125	923B 923A	9239 9235 9231 922D 9229 9225 9221 921D 9219 9215 9211 920D 9209 9205 9201 91FD 91F9 91F5 91F1 91ED 91E9 91E5
124	91E1 91E0	91DF 91DB 91D7 91D3 91CF 91CB 91C7 91C3 91BF 91BB 91B7 91B3 91AF 91AB 91A7 91A3 919F 919B 9197 9193 918F 918B
123	9187 9186	9185 9181 917D 9179 9175 9171 916D 9169 9165 9161 915D 9159 9155 9151 914D 9149 9145 9141 913D 9139 9135 9131
122	912D 912C	912B 9127 9123 911F 911B 9117 9113 910F 910B 9107 9103 90FF 90FB 90F7 90F3 90EF 90EB 90E7 90E3 90DF 90DB 90D7
121	90D3 90D2	90D1 90CD 90C9 90C5 90C1 90BD 90B9 90B5 90B1 90AD 90A9 90A5 90A1 909D 9099 9095 9091 908D 9089 9085 9081 907D
120	9079 9078	9077 9073 906F 906B 9067 9063 905F 905B 9057 9053 904F 904B 9047 9043 903F 903B 9037 9033 902F 902B 9027 9023
119	901F 901E	901D 9019 9015 9011 900D 9009 9005 9001 8FFD 8FF9 8FF5 8FF1 8FED 8FE9 8FE5 8FE1 8FDD 8FD9 8FD5 8FD1 8FCD 8FC9
118	8FC5 8FC4	8FC3 8FBF 8FBB 8FB7 8FB3 8FAF 8FAB 8FA7 8FA3 8F9F 8F9B 8F97 8F93 8F8F 8F8B 8F87 8F83 8F7F 8F7B 8F77 8F73 8F6F
117	8F6B 8F6A	8F69 8F65 8F61 8F5D 8F59 8F55 8F51 8F4D 8F49 8F45 8F41 8F3D 8F39 8F35 8F31 8F2D 8F29 8F25 8F21 8F1D 8F19 8F15
116	8F11 8F10	8F0F 8F0B 8F07 8F03 8EFF 8EFB 8EF7 8EF3 8EEF 8EEB 8EE7 8EE3 8EDF 8EDB 8ED7 8ED3 8ECF 8ECB 8EC7 8EC3 8EBF 8EBB
115	8EB7 8EB6	8EB5 8EB1 8EAD 8EA9 8EA5 8EA1 8E9D 8E99 8E95 8E91 8E8D 8E89 8E85 8E81 8E7D 8E79 8E75 8E71 8E6D 8E69 8E65 8E61
114	8E5D 8E5C	8E5B 8E57 8E53 8E4F 8E4B 8E47 8E43 8E3F 8E3B 8E37 8E33 8E2F 8E2B 8E27 8E23 8E1F 8E1B 8E17 8E13 8E0F 8E0B 8E07
113	8E03 8E02	8E01 8DFD 8DF9 8DF5 8DF1 8DED 8DE9 8DE5 8DE1 8DDD 8DD9 8DD5 8DD1 8DCD 8DC9 8DC5 8DC1 8DBD 8DB9 8DB5 8DB1 8DAD
112	8DA9 8DA8	8DA7 8DA3 8D9F 8D9B 8D97 8D93 8D8F 8D8B 8D87 8D83 8D7F 8D7B 8D77 8D73 8D6F 8D6B 8D67 8D63 8D5F 8D5B 8D57 8D53
111	8D4F 8D4E	8D4D 8D49 8D45 8D41 8D3D 8D39 8D35 8D31 8D2D 8D29 8D25 8D21 8D1D 8D19 8D15 8D11 8D0D 8D09 8D05 8D01 8CFD 8CF9
110	8CF5 8CF4	8CF3 8CEF 8CEB 8CE7 8CE3 8CDF 8CDB 8CD7 8CD3 8CCF 8CCB 8CC7 8CC3 8CBF 8CBB 8CB7 8CB3 8CAF 8CAB 8CA7 8CA3 8C9F
109	8C9B 8C9A	8C99 8C95 8C91 8C8D 8C89 8C85 8C81 8C7D 8C79 8C75 8C71 8C6D 8C69 8C65 8C61 8C5D 8C59 8C55 8C51 8C4D 8C49 8C45
108	8C41 8C40	8C3F 8C3B 8C37 8C33 8C2F 8C2B 8C27 8C23 8C1F 8C1B 8C17 8C13 8C0F 8C0B 8C07 8C03 8BFF 8BFB 8BF7 8BF3 8BEF 8BEB
107	8BE7 8BE6	8BE5 8BE1 8BDD 8BD9 8BD5 8BD1 8BCD 8BC9 8BC5 8BC1 8BBD 8BB9 8BB5 8BB1 8BAD 8BA9 8BA5 8BA1 8B9D 8B99 8B95 8B91
106	8B8D 8B8C	8B8B 8B87 8B83 8B7F 8B7B 8B77 8B73 8B6F 8B6B 8B67 8B63 8B5F 8B5B 8B57 8B53 8B4F 8B4B 8B47 8B43 8B3F 8B3B 8B37
105	8B33 8B32	8B31 8B2D 8B29 8B25 8B21 8B1D 8B19 8B15 8B11 8B0D 8B09 8B05 8B01 8AFD 8AF9 8AF5 8AF1 8AED 8AE9 8AE5 8AE1 8ADD
104	8AD9 8AD8	8AD7 8AD3 8ACF 8ACB 8AC7 8AC3 8ABF 8ABB 8AB7 8AB3 8AAF 8AAB 8AA7 8AA3 8A9F 8A9B 8A97 8A93 8A8F 8A8B 8A87 8A83
103	8A7F 8A7E	8A7D 8A79 8A75 8A71 8A6D 8A69 8A65 8A61 8A5D 8A59 8A55 8A51 8A4D 8A49 8A45 8A41 8A3D 8A39 8A35 8A31 8A2D 8A29
102	8A25 8A24	8A23 8A1F 8A1B 8A17 8A13 8A0F 8A0B 8A07 8A03 89FF 89FB 89F7 89F3 89EF 89EB 89E7 89E3 89DF 89DB 89D7 89D3 89CF
101	89CB 89CA	89C9 89C5 89C1 89BD 89B9 89B5 89B1 89AD 89A9 89A5 89A1 899D 8999 8995 8991 898D 8989 8985 8981 897D 8979 8975
100	8971 8970	896F 896B 8967 8963 895F 895B 8957 8953 894F 894B 8947 8943 893F 893B 8937 8933 892F 892B 8927 8923 891F 891B
99	8917 8916	8915 8911 890D 8909 8905 8901 88FD 88F9 88F5 88F1 88ED 88E9 88E5 88E1 88DD 88D9 88D5 88D1 88CD 88C9 88C5 88C1
98	88BD 88BC	88BB 88B7 88B3 88AF 88AB 88A7 88A3 889F 889B 8897 8893 888F 888B 8887 8883 887F 887B 8877 8873 886F 886B 8867
97	8863 8862	8861 885D 8859 8855 8851 884D 8849 8845 8841 883D 8839 8835 8831 882D 8829 8825 8821 881D 8819 8815 8811 880D
96	8809 8808	8807 8803 87FF 87FB 87F7 87F3 87EF 87EB 87E7 87E3 87DF 87DB 87D7 87D3 87CF 87CB 87C7 87C3 87BF 87BB 87B7 87B3
95	87AF 87AE	87AD 87A9 87A5 87A1 879D 8799 8795 8791 878D 8789 8785 8781 877D 8779 8775 8771 876D 8769 8765 8761 875D 8759
94	8755 8754	8753 874F 874B 8747 8743 873F 873B 8737 8733 872F 872B 8727 8723 871F 871B 8717 8713 870F 870B 8707 8703 86FF
93	86FB 86FA	86F9 86F5 86F1 86ED 86E9 86E5 86E1 86DD 86D9 86D5 86D1 86CD 86C9 86C5 86C1 86BD 86B9 86B5 86B1 86AD 86A9 86A5
92	86A1 86A0	869F 869B 8697 8693 868F 868B 8687 8683 867F 867B 8677 8673 866F 866B 8667 8663 865F 865B 8657 8653 864F 864B
91	8647 8646	8645 8641 863D 8639 8635 8631 862D 8629 8625 8621 861D 8619 8615 8611 860D 8609 8605 8601 85FD 85F9 85F5 85F1
90	85ED 85EC	85EB 85E7 85E3 85DF 85DB 85D7 85D3 85CF 85CB 85C7 85C3 85BF 85BB 85B7 85B3 85AF 85AB 85A7 85A3 859F 859B 8597
89	8593 8592	8591 858D 8589 8585 8581 857D 8579 8575 8571 856D 8569 8565 8561 855D 8559 8555 8551 854D 8549 8545 8541 853D
88	8539 8538	8537 8533 852F 852B 8527 8523 851F 851B 8517 8513 850F 850B 8507 8503 84FF 84FB 84F7 84F3 84EF 84EB 84E7 84E3
87	84DF 84DE	84DD 84D9 84D5 84D1 84CD 84C9 84C5 84C1 84BD 84B9 84B5 84B1 84AD 84A9 84A5 84A1 849D 8499 8495 8491 848D 8489
86	8485 8484	8483 847F 847B 8477 8473 846F 846B 8467 8463 845F 845B 8457 8453 844F 844B 8447 8443 843F 843B 8437 8433 842F
85	842B 842A	8429 8425 8421 841D 8419 8415 8411 840D 8409 8405 8401 83FD 83F9 83F5 83F1 83ED 83E9 83E5 83E1 83DD 83D9 83D5
84	83D1 83D0	83CF 83CB 83C7 83C3 83BF 83BB 83B7 83B3 83AF 83AB 83A7 83A3 839F 839B 8397 8393 838F 838B 8387 8383 837F 837B
83	8377 8376	8375 8371 836D 8369 8365 8361 835D 8359 8355 8351 834D 8349 8345 8341 833D 8339 8335 8331 832D 8329 8325 8321
82	831D 831C	831B 8317 8313 830F 830B 8307 8303 82FF 82FB 82F7 82F3 82EF 82EB 82E7 82E3 82DF 82DB 82D7 82D3 82CF 82CB 82C7
81	82C3 82C2	82C1 82BD 82B9 82B5 82B1 82AD 82A9 82A5 82A1 829D 8299 8295 8291 828D 8289 8285 8281 827D 8279 8275 8271 826D
80	8269 8268	8267 8263 825F 825B 8257 8253 824F 824B 8247 8243 823F 823B 8237 8233 822F 822B 8227 8223 821F 821B 8217 8213
79	820F 820E	820D 8209 8205 8201 81FD 81F9 81F5 81F1 81ED 81E9 81E5 81E1 81DD 81D9 81D5 81D1 81CD 81C9 81C5 81C1 81BD 81B9
78	81B5 81B4	81B3 81AF 81AB 81A7 81A3 819F 819B 8197 8193 818F 818B 8187 8183 817F 817B 8177 8173 816F 816B 8167 8163 815F
77	815B 815A	8159 8155 8151 814D 8149 8145 8141 813D 8139 8135 8131 812D 8129 8125 8121 811D 8119 8115 8111 810D 8109 8105
76	8101 8100	80FF 80FB 80F7 80F3 80EF 80EB 80E7 80E3 80DF 80DB 80D7 80D3 80CF 80CB 80C7 80C3 80BF 80BB 80B7 80B3 80AF 80AB
75	80A7 80A6	80A5 80A1 809D 8099 8095 8091 808D 8089 8085 8081 807D 8079 8075 8071 806D 8069 8065 8061 805D 8059 8055 8051
74	804D 804C	804B 8047 8043 803F 803B 8037 8033 802F 802B 8027 8023 801F 801B 8017 8013 800F 800B 8007 8003 7FFF 7FFB 7FF7
73	7FF3 7FF2	7FF1 7FED 7FE9 7FE5 7FE1 7FDD 7FD9 7FD5 7FD1 7FCD 7FC9 7FC5 7FC1 7FBD 7FB9 7FB5 7FB1 7FAD 7FA9 7FA5 7FA1 7F9D
72	7F99 7F98	7F97 7F93 7F8F 7F8B 7F87 7F83 7F7F 7F7B 7F77 7F73 7F6F 7F6B 7F67 7F63 7F5F 7F5B 7F57 7F53 7F4F 7F4B 7F47 7F43
71	7F3F 7F3E	7F3D 7F39 7F35 7F31 7F2D 7F29 7F25 7F21 7F1D 7F19 7F15 7F11 7F0D 7F09 7F05 7F01 7EFD 7EF9 7EF5 7EF1 7EED 7EE9
70	7EE5 7EE4	7EE3 7EDF 7EDB 7ED7 7ED3 7ECF 7ECB 7EC7 7EC3 7EBF 7EBB 7EB7 7EB3 7EAF 7EAB 7EA7 7EA3 7E9F 7E9B 7E97 7E93 7E8F
69	7E8B 7E8A	7E89 7E85 7E81 7E7D 7E79 7E75 7E71 7E6D 7E69 7E65 7E61 7E5D 7E59 7E55 7E51 7E4D 7E49 7E45 7E41 7E3D 7E39 7E35
68	7E31 7E30	7E2F 7E2B 7E27 7E23 7E1F 7E1B 7E17 7E13 7E0F 7E0B 7E07 7E03 7DFF 7DFB 7DF7 7DF3 7DEF 7DEB 7DE7 7DE3 7DDF 7DDB
67	7DD7 7DD6	7DD5 7DD1 7DCD 7DC9 7DC5 7DC1 7DBD 7DB9 7DB5 7DB1 7DAD 7DA9 7DA5 7DA1 7D9D 7D99 7D95 7D91 7D8D 7D89 7D85 7D81
66	7D7D 7D7C	7D7B 7D77 7D73 7D6F 7D6B 7D67 7D63 7D5F 7D5B 7D57 7D53 7D4F 7D4B 7D47 7D43 7D3F 7D3B 7D37 7D33 7D2F 7D2B 7D27
65	7D23 7D22	7D21 7D1D 7D19 7D15 7D11 7D0D 7D09 7D05 7D01 7CFD 7CF9 7CF5 7CF1 7CED 7CE9 7CE5 7CE1 7CDD 7CD9 7CD5 7CD1 7CCD

64	7CC9 7CC8	7CC7 7CC3 7CBF 7CBB 7CB7 7CB3 7CAF 7CAB 7CA7 7CA3 7C9F 7C9B 7C97 7C93 7C8F 7C8B 7C87 7C83 7C7F 7C7B 7C77 7C73
63	7C6F 7C6E	7C6D 7C69 7C65 7C61 7C5D 7C59 7C55 7C51 7C4D 7C49 7C45 7C41 7C3D 7C39 7C35 7C31 7C2D 7C29 7C25 7C21 7C1D 7C19
62	7C15 7C14	7C13 7C0F 7C0B 7C07 7C03 7BFF 7BFB 7BF7 7BF3 7BEF 7BEB 7BE7 7BE3 7BDF 7BDB 7BD7 7BD3 7BCF 7BCB 7BC7 7BC3 7BBF
61	7BBB 7BBA	7BB9 7BB5 7BB1 7BAD 7BA9 7BA5 7BA1 7B9D 7B99 7B95 7B91 7B8D 7B89 7B85 7B81 7B7D 7B79 7B75 7B71 7B6D 7B69 7B65
60	7B61 7B60	7B5F 7B5B 7B57 7B53 7B4F 7B4B 7B47 7B43 7B3F 7B3B 7B37 7B33 7B2F 7B2B 7B27 7B23 7B1F 7B1B 7B17 7B13 7B0F 7B0B
59	7B07 7B06	7B05 7B01 7AFD 7AF9 7AF5 7AF1 7AED 7AE9 7AE5 7AE1 7ADD 7AD9 7AD5 7AD1 7ACD 7AC9 7AC5 7AC1 7ABD 7AB9 7AB5 7AB1
58	7AAD 7AAC	7AAB 7AA7 7AA3 7A9F 7A9B 7A97 7A93 7A8F 7A8B 7A87 7A83 7A7F 7A7B 7A77 7A73 7A6F 7A6B 7A67 7A63 7A5F 7A5B 7A57
57	7A53 7A52	7A51 7A4D 7A49 7A45 7A41 7A3D 7A39 7A35 7A31 7A2D 7A29 7A25 7A21 7A1D 7A19 7A15 7A11 7A0D 7A09 7A05 7A01 79FD
56	79F9 79F8	79F7 79F3 79EF 79EB 79E7 79E3 79DF 79DB 79D7 79D3 79CF 79CB 79C7 79C3 79BF 79BB 79B7 79B3 79AF 79AB 79A7 79A3
55	799F 799E	799D 7999 7995 7991 798D 7989 7985 7981 797D 7979 7975 7971 796D 7969 7965 7961 795D 7959 7955 7951 794D 7949
54	7945 7944	7943 793F 793B 7937 7933 792F 792B 7927 7923 791F 791B 7917 7913 790F 790B 7907 7903 78FF 78FB 78F7 78F3 78EF
53	78EB 78EA	78E9 78E5 78E1 78DD 78D9 78D5 78D1 78CD 78C9 78C5 78C1 78BD 78B9 78B5 78B1 78AD 78A9 78A5 78A1 789D 7899 7895
52	7891 7890	788F 788B 7887 7883 787F 787B 7877 7873 786F 786B 7867 7863 785F 785B 7857 7853 784F 784B 7847 7843 783F 783B
51	7837 7836	7835 7831 782D 7829 7825 7821 781D 7819 7815 7811 780D 7809 7805 7801 77FD 77F9 77F5 77F1 77ED 77E9 77E5 77E1
50	77DD 77DC	77DB 77D7 77D3 77CF 77CB 77C7 77C3 77BF 77BB 77B7 77B3 77AF 77AB 77A7 77A3 779F 779B 7797 7793 778F 778B 7787
49	7783 7782	7781 777D 7779 7775 7771 776D 7769 7765 7761 775D 7759 7755 7751 774D 7749 7745 7741 773D 7739 7735 7731 772D
48	7729 7728	7727 7723 771F 771B 7717 7713 770F 770B 7707 7703 76FF 76FB 76F7 76F3 76EF 76EB 76E7 76E3 76DF 76DB 76D7 76D3
47	76CF 76CE	76CD 76C9 76C5 76C1 76BD 76B9 76B5 76B1 76AD 76A9 76A5 76A1 769D 7699 7695 7691 768D 7689 7685 7681 767D 7679
46	7675 7674	7673 766F 766B 7667 7663 765F 765B 7657 7653 764F 764B 7647 7643 763F 763B 7637 7633 762F 762B 7627 7623 761F
45	761B 761A	7619 7615 7611 760D 7609 7605 7601 75FD 75F9 75F5 75F1 75ED 75E9 75E5 75E1 75DD 75D9 75D5 75D1 75CD 75C9 75C5
44	75C1 75C0	75BF 75BB 75B7 75B3 75AF 75AB 75A7 75A3 759F 759B 7597 7593 758F 758B 7587 7583 757F 757B 7577 7573 756F 756B
43	7567 7566	7565 7561 755D 7559 7555 7551 754D 7549 7545 7541 753D 7539 7535 7531 752D 7529 7525 7521 751D 7519 7515 7511
42	750D 750C	750B 7507 7503 74FF 74FB 74F7 74F3 74EF 74EB 74E7 74E3 74DF 74DB 74D7 74D3 74CF 74CB 74C7 74C3 74BF 74BB 74B7
41	74B3 74B2	74B1 74AD 74A9 74A5 74A1 749D 7499 7495 7491 748D 7489 7485 7481 747D 7479 7475 7471 746D 7469 7465 7461 745D
40	7459 7458	7457 7453 744F 744B 7447 7443 743F 743B 7437 7433 742F 742B 7427 7423 741F 741B 7417 7413 740F 740B 7407 7403
39	73FF 73FE	73FD 73F9 73F5 73F1 73ED 73E9 73E5 73E1 73DD 73D9 73D5 73D1 73CD 73C9 73C5 73C1 73BD 73B9 73B5 73B1 73AD 73A9
38	73A5 73A4	73A3 739F 739B 7397 7393 738F 738B 7387 7383 737F 737B 7377 7373 736F 736B 7367 7363 735F 735B 7357 7353 734F
37	734B 734A	7349 7345 7341 733D 7339 7335 7331 732D 7329 7325 7321 731D 7319 7315 7311 730D 7309 7305 7301 72FD 72F9 72F5
36	72F1 72F0	72EF 72EB 72E7 72E3 72DF 72DB 72D7 72D3 72CF 72CB 72C7 72C3 72BF 72BB 72B7 72B3 72AF 72AB 72A7 72A3 729F 729B
35	7297 7296	7295 7291 728D 7289 7285 7281 727D 7279 7275 7271 726D 7269 7265 7261 725D 7259 7255 7251 724D 7249 7245 7241
34	723D 723C	723B 7237 7233 722F 722B 7227 7223 721F 721B 7217 7213 720F 720B 7207 7203 71FF 71FB 71F7 71F3 71EF 71EB 71E7
33	71E3 71E2	71E1 71DD 71D9 71D5 71D1 71CD 71C9 71C5 71C1 71BD 71B9 71B5 71B1 71AD 71A9 71A5 71A1 719D 7199 7195 7191 718D
32	7189 7188	7187 7183 717F 717B 7177 7173 716F 716B 7167 7163 715F 715B 7157 7153 714F 714B 7147 7143 713F 713B 7137 7133
31	712F 712E	712D 7129 7125 7121 711D 7119 7115 7111 710D 7109 7105 7101 70FD 70F9 70F5 70F1 70ED 70E9 70E5 70E1 70DD 70D9
30	70D5 70D4	70D3 70CF 70CB 70C7 70C3 70BF 70BB 70B7 70B3 70AF 70AB 70A7 70A3 709F 709B 7097 7093 708F 708B 7087 7083 707F
29	707B 707A	7079 7075 7071 706D 7069 7065 7061 705D 7059 7055 7051 704D 7049 7045 7041 703D 7039 7035 7031 702D 7029 7025
28	7021 7020	701F 701B 7017 7013 700F 700B 7007 7003 6FFF 6FFB 6FF7 6FF3 6FEF 6FEB 6FE7 6FE3 6FDF 6FDB 6FD7 6FD3 6FCF 6FCB
27	6FC7 6FC6	6FC5 6FC1 6FBD 6FB9 6FB5 6FB1 6FAD 6FA9 6FA5 6FA1 6F9D 6F99 6F95 6F91 6F8D 6F89 6F85 6F81 6F7D 6F79 6F75 6F71
26	6F6D 6F6C	6F6B 6F67 6F63 6F5F 6F5B 6F57 6F53 6F4F 6F4B 6F47 6F43 6F3F 6F3B 6F37 6F33 6F2F 6F2B 6F27 6F23 6F1F 6F1B 6F17
25	6F13 6F12	6F11 6F0D 6F09 6F05 6F01 6EFD 6EF9 6EF5 6EF1 6EED 6EE9 6EE5 6EE1 6EDD 6ED9 6ED5 6ED1 6ECD 6EC9 6EC5 6EC1 6EBD
24	6EB9 6EB8	6EB7 6EB3 6EAF 6EAB 6EA7 6EA3 6E9F 6E9B 6E97 6E93 6E8F 6E8B 6E87 6E83 6E7F 6E7B 6E77 6E73 6E6F 6E6B 6E67 6E63
23	6E5F 6E5E	6E5D 6E59 6E55 6E51 6E4D 6E49 6E45 6E41 6E3D 6E39 6E35 6E31 6E2D 6E29 6E25 6E21 6E1D 6E19 6E15 6E11 6E0D 6E09
22	6E05 6E04	6E03 6DFF 6DFB 6DF7 6DF3 6DEF 6DEB 6DE7 6DE3 6DDF 6DDB 6DD7 6DD3 6DCF 6DCB 6DC7 6DC3 6DBF 6DBB 6DB7 6DB3 6DAF
21	6DAB 6DAA	6DA9 6DA5 6DA1 6D9D 6D99 6D95 6D91 6D8D 6D89 6D85 6D81 6D7D 6D79 6D75 6D71 6D6D 6D69 6D65 6D61 6D5D 6D59 6D55
20	6D51 6D50	6D4F 6D4B 6D47 6D43 6D3F 6D3B 6D37 6D33 6D2F 6D2B 6D27 6D23 6D1F 6D1B 6D17 6D13 6D0F 6D0B 6D07 6D03 6CFF 6CFB
19	6CF7 6CF6	6CF5 6CF1 6CED 6CE9 6CE5 6CE1 6CDD 6CD9 6CD5 6CD1 6CCD 6CC9 6CC5 6CC1 6CBD 6CB9 6CB5 6CB1 6CAD 6CA9 6CA5 6CA1
18	6C9D 6C9C	6C9B 6C97 6C93 6C8F 6C8B 6C87 6C83 6C7F 6C7B 6C77 6C73 6C6F 6C6B 6C67 6C63 6C5F 6C5B 6C57 6C53 6C4F 6C4B 6C47
17	6C43 6C42	6C41 6C3D 6C39 6C35 6C31 6C2D 6C29 6C25 6C21 6C1D 6C19 6C15 6C11 6C0D 6C09 6C05 6C01 6BFD 6BF9 6BF5 6BF1 6BED
16	6BE9 6BE8	6BE7 6BE3 6BDF 6BDB 6BD7 6BD3 6BCF 6BCB 6BC7 6BC3 6BBF 6BBB 6BB7 6BB3 6BAF 6BAB 6BA7 6BA3 6B9F 6B9B 6B97 6B93
15	6B8F 6B8E	6B8D 6B89 6B85 6B81 6B7D 6B79 6B75 6B71 6B6D 6B69 6B65 6B61 6B5D 6B59 6B55 6B51 6B4D 6B49 6B45 6B41 6B3D 6B39
14	6B35 6B34	6B33 6B2F 6B2B 6B27 6B23 6B1F 6B1B 6B17 6B13 6B0F 6B0B 6B07 6B03 6AFF 6AFB 6AF7 6AF3 6AEF 6AEB 6AE7 6AE3 6ADF
13	6ADB 6ADA	6AD9 6AD5 6AD1 6ACD 6AC9 6AC5 6AC1 6ABD 6AB9 6AB5 6AB1 6AAD 6AA9 6AA5 6AA1 6A9D 6A99 6A95 6A91 6A8D 6A89 6A85
12	6A81 6A80	6A7F 6A7B 6A77 6A73 6A6F 6A6B 6A67 6A63 6A5F 6A5B 6A57 6A53 6A4F 6A4B 6A47 6A43 6A3F 6A3B 6A37 6A33 6A2F 6A2B
11	6A27 6A26	6A25 6A21 6A1D 6A19 6A15 6A11 6A0D 6A09 6A05 6A01 69FD 69F9 69F5 69F1 69ED 69E9 69E5 69E1 69DD 69D9 69D5 69D1
10	69CD 69CC	69CB 69C7 69C3 69BF 69BB 69B7 69B3 69AF 69AB 69A7 69A3 699F 699B 6997 6993 698F 698B 6987 6983 697F 697B 6977
9	6973 6972	6971 696D 6969 6965 6961 695D 6959 6955 6951 694D 6949 6945 6941 693D 6939 6935 6931 692D 6929 6925 6921 691D
8	6919 6918	6917 6913 690F 690B 6907 6903 68FF 68FB 68F7 68F3 68EF 68EB 68E7 68E3 68DF 68DB 68D7 68D3 68CF 68CB 68C7 68C3
7	68BF 68BE	68BD 68B9 68B5 68B1 68AD 68A9 68A5 68A1 689D 6899 6895 6891 688D 6889 6885 6881 687D 6879 6875 6871 686D 6869
6	6865 6864	6863 685F 685B 6857 6853 684F 684B 6847 6843 683F 683B 6837 6833 682F 682B 6827 6823 681F 681B 6817 6813 680F
5	680B 680A	6809 6805 6801 67FD 67F9 67F5 67F1 67ED 67E9 67E5 67E1 67DD 67D9 67D5 67D1 67CD 67C9 67C5 67C1 67BD 67B9 67B5
4	67B1 67B0	67AF 67AB 67A7 67A3 679F 679B 6797 6793 678F 678B 6787 6783 677F 677B 6777 6773 676F 676B 6767 6763 675F 675B
3	6757 6756	6755 6751 674D 6749 6745 6741 673D 6739 6735 6731 672D 6729 6725 6721 671D 6719 6715 6711 670D 6709 6705 6701
2	66FD 66FC	66FB 66F7 66F3 66EF 66EB 66E7 66E3 66DF 66DB 66D7 66D3 66CF 66CB 66C7 66C3 66BF 66BB 66B7 66B3 66AF 66AB 66A7
1	66A3 66A2	66A1 669D 6699 6695 6691 668D 6689 6685 6681 667D 6679 6675 6671 666D 6669 6665 6661 665D 6659 6655 6651 664D
0	6649 6648	6647 6643 663F 663B 6637 6633 662F 662B 6627 6623 661F 661B 6617 6613 660F 660B 6607 6603 65FF 65FB 65F7 65F3

VAKWERKPROGRAMMA

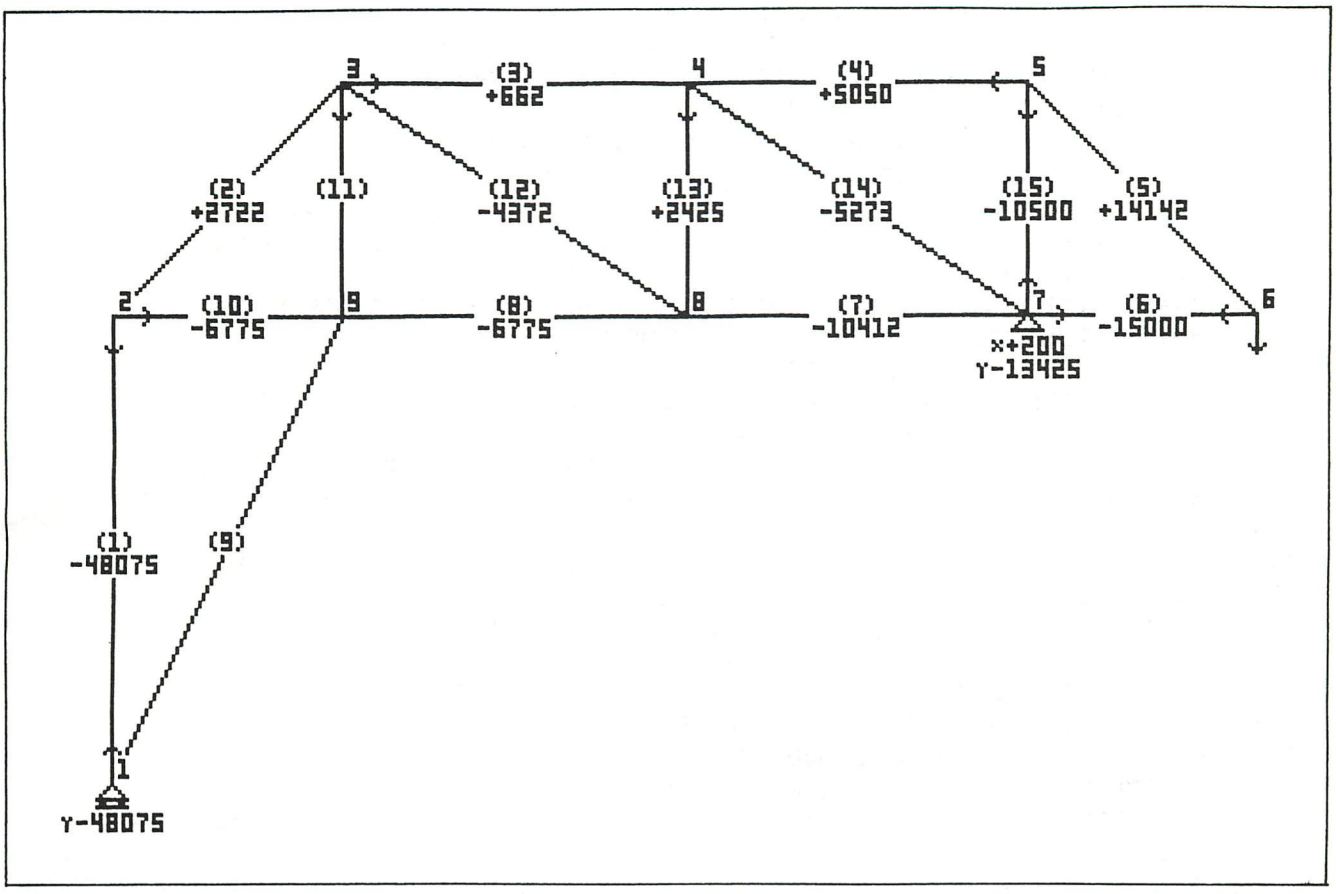
```
10    REM VAKWERKPROGRAMMA -- A.VINGERLING -- R.DAM
11    GOSUB 900
100   REM coefficientenmatrix genereren
105   FOR I=0 TO N:FOR J=0 TO N2-3
110   IF A(0,J)<>I AND A(1,J)<>I GOTO 145
120   IF A(0,J)=I THEN H=A(1,J):GOTO 130
125   H=A(0,J)
130   DX!=KC!(0,H)-KC!(0,I)
131   DY!=KC!(1,H)-KC!(1,I)
135   L!(J)=SQR(DX!*DX!+DY!*DY!)
140   CM!(2*I,J)=DX!/L!(J)
141   CM!(2*I+1,J)=DY!/L!(J)
145   NEXT
150   BV!(2*I)=-B!(0,I)
151   BV!(2*I+1)=-B!(1,I)
155   NEXT
160   CM!(2*VAST-2,N2-2)=1
161   CM!(2*VAST-1,N2-1)=1
162   CM!(2*ROL-1,N2)=1
165   RETURN:REM einde CM-generatie
200   REM LU-decompositie volgens Gauss
210   REM Pivotrij zoeken
211   FOR K=0 TO N2
212   PK=K
213   MAX!=ABS(CM!(K,K))
214   FOR I=K+1 TO N2:IF I>N2 GOTO 216
215   IF ABS(CM!(I,K))>MAX! THEN MAX!=ABS(CM!(I,K)):PK=I
216   NEXT
217   P(K)=PK:REM Pivotrijnummer bewaren
220   REM rij PK verwisselen met rij K
221   FOR J=K TO N2
222   W!=CM!(K,J)
223   CM!(K,J)=CM!(PK,J)
224   CM!(PK,J)=W!
225   NEXT
230   REM bovendriehoeksmatrix genereren vlgs Gauss
231   FOR I=K+1 TO N2:IF I>N2 GOTO 240
232   M!=CM!(I,K)/CM!(K,K)
233   FOR J=K+1 TO N2:IF J>N2 GOTO 235
234   CM!(I,J)=CM!(I,J)-M!*CM!(K,J)
235   NEXT
236   CM!(I,K)=M!
240   NEXT:REM einde bovendriehoeks generatie
250   NEXT:GOTO 255
251   DET!=1.0
252   FOR I=0 TO N2
253   DET!=DET!*CM!(I,I)
254   NEXT
255   RETURN:REM einde LU-decompositie
300   REM LU-terugsubstitutie vlgs Gauss
310   REM Pivotrijwisseling
311   FOR K=0 TO N2
312   W!=BV!(K)
313   BV!(K)=BV!(P(K))
314   BV!(P(K))=W!
320   REM Gaussbewerking
321   FOR I=K+1 TO N2:IF I>N2 GOTO 323
322   BV!(I)=BV!(I)-CM!(I,K)*BV!(K)
323   NEXT
324   NEXT
330   REM terugsubstitutie
331   FOR I=N2 TO 0 STEP -1
332   X!(I)=BV!(I)/CM!(I,I)
333   FOR L=I-1 TO 0 STEP -1:IF I<1 GOTO 335
```

```
334   BV!(L)=BV!(L)-CM!(L,I)*X!(I)
335   NEXT
336   NEXT
337   RETURN:REM einde LU-terugsubstitutie
400   IF XZQ=1 THEN GOTO 406
402   X$="0123456789+-. EVRLHON[]XY":DIM CAR$(LEN(X$)-1)
404   FOR Z=0 TO LEN(X$)-1:READ A$,CAR$(Z):NEXT:XZQ=1:RETURN
406   X1=X:Y1=Y
408   FOR M=0 TO LEN(A$)-1
410   T$=MID$(A$,M,1)
412   IF P1=1 THEN FILL X,Y X+4,Y+6 7
414   FOR Z=0 TO LEN(X$)-1:IF T$=MID$(X$,Z,1) THEN GR$=CAR$(Z):GOTO 418
416   NEXT:Z=LEN(X$)-2
418   IF GR$=" " THEN X=X+4:GOTO 440
420   FOR NN=0 TO LEN(GR$)-1 STEP 4
422   IF VFLAG<>0 GOTO 444
424   IF MID$(GR$,NN,1)="/" THEN X=X+4:GOTO 440
426   ZZ=VAL(MID$(GR$,NN,1)):YY=VAL(MID$(GR$,NN+1,1))
428   JC5=X+ZZ:JC6=Y+VAL(MID$(GR$,NN+1,1))
430   JC7=X+VAL(MID$(GR$,NN+2,1)):JC8=Y+VAL(MID$(GR$,NN+3,1))
432   DRAW JC5,JC6 JC7,JC8 C:IF XIM-1<=0 GOTO 438
434   JC9=X+VAL(MID$(GR$,NN+2,1)):JC10=Y+VAL(MID$(GR$,NN+3,1))
436   DRAW X+ZZ,Y+YY JC9,JC10 C
438   NEXT NN
440   IF X+4>=XMAX THEN X=X1:Y=Y-6
442   NEXT M:RETURN
444   IF MID$(GR$,NN,1)="/" THEN Y=Y-6:GOTO 454
446   JC1=X+VAL(MID$(GR$,NN+1,1)):JC2=Y-VAL(MID$(GR$,NN,1))
448   JC3=X+VAL(MID$(GR$,NN+3,1)):JC4=Y-VAL(MID$(GR$,NN+2,1))
450   DRAW JC1,JC2 JC3,JC4 C
452   NEXT NN
454   IF Y-8<0 THEN Y=Y1:X=X-8
456   NEXT M:RETURN
600   MODE 6:N1=2*N-2:XM!=KC!(0,0):XN!=XM!:YM!=KC!(1,0):YN!=YM!
602   FOR I=0 TO N:KCX!=KC!(0,I):KCY!=KC!(1,I)
604   IF XM!<KCX! THEN XM!=KCX!
606   IF XN!>KCX! THEN XN!=KCX!
608   IF YM!<KCY! THEN YM!=KCY!
610   IF YN!>KCY! THEN YN!=KCY!
612   NEXT:XM!=XM!-XN!:YM!=YM!-YN!
614   FOR I=0 TO N:IF XN!<0 THEN KC!(0,I)=KC!(0,I)-XN!
616   IF YN!<0 THEN KC!(1,I)=KC!(1,I)-YN!
618   NEXT:FX!=(XMAX-34)/XM!:FY!=(YMAX-34)/YM!:XM!=XM!/2:YM!=YM!/2
620   XM2=XMAX/2:YM2=YMAX/2:IF FX!<FY! THEN F!=FX!:GOTO 624
622   F!=FY!
624   FOR I=0 TO N:KC!(0,I)=XM2+(KC!(0,I)-XM!)*F!:KC!(1,I)=YM2+(KC!(1,I)-YM
!)*F!:NEXT
626   FOR I=0 TO N1:A0I=A(0,I):A1I=A(1,I)
628   A=KC!(0,A0I):B=KC!(1,A0I):C=KC!(0,A1I):D=KC!(1,A1I)
630   AA=A:BB=B:CC=C:DD=D:DRAW A,B C,D 0
632   A=0.5*(AA+CC):B=0.5*(BB+DD):FF!=X!(I):C=13+SGN(FF!)*2:IF ABS(FF!)<0.1
 THEN C=0
634   P1=1:A$=STR$(I+1):A$="["+MID$(A$,1,LEN(A$)-3)+"]":GOSUB 682:GOSUB 400
:IF ABS(FF!)<0.1 GOTO 638
636   A$=STR$(INT(FF!+0.5)):A$=LEFT$(A$,LEN(A$)-2):GOSUB 686:GOSUB 682:Y=Y-
6:GOSUB 400
638   NEXT:AV=KC!(0,VAST-1):BV=KC!(1,VAST-1):AR=KC!(0,ROL-1):BR=KC!(1,ROL-1
)
640   P1=0:C=0:X=AV-4:Y=BV-4:A$="V":GOSUB 400:X=AR-4:Y=BR-6:A$="R":GOSUB 40
0
642   FOR I=0 TO N:P1=1:A=KC!(0,I):B=KC!(1,I):R$=STR$(I+1):A$=MID$(R$,1,1):
C=0:X=A+1:Y=B+1:GOSUB 400:P1=0
644   C=11:FF!=B!(0,I):GOSUB 670:FF!=B!(1,I):GOSUB 676:NEXT
646   C=15:A=AV:B=BV:FF!=X!(N2-2):GOSUB 670
648   X=A:Y=B-11:IF B!(1,VAST-1)<0 OR X!(N2-1)<0 THEN Y=Y-6
650   GOSUB 684:A$="X"+A$:GOSUB 400:FF!=X!(N2-1):GOSUB 676:FF!=-FF!:X=A:Y=B-17
```

```
652   IF B!(1,VAST-1)<0.0 OR FF!>0.0 THEN Y=Y-6
654   GOSUB 684:A$="Y"+A$:GOSUB 400
656   C=15:A=AR:B=BR:FF!=X!(N2):GOSUB 676:FF!=-FF!:X=A:Y=B-13
658   IF B!(1,ROL-1)<0.0 OR FF!>0.0 THEN Y=Y-4
660   GOSUB 684:A$="Y"+A$:GOSUB 400
662   IF GETC=0 GOTO 662
664   GOTO 9999
670   IF FF!=0 THEN RETURN
672   Y=B-3:IF FF!>0 THEN X=A+1:A$="H":GOSUB 400:RETURN
674   X=A-10:A$="L":GOSUB 400:RETURN
676   IF FF!=0 THEN RETURN
678   X=A-2:IF FF!<0 THEN Y=B-10:A$="N":GOSUB 400:RETURN
680   Y=B+1:A$="O":GOSUB 400:RETURN
682   X=0.5*(AA+CC)-LEN(A$)*2:Y=0.5*(BB+DD):RETURN
684   FF!=INT(FF!+0.5):A$=STR$(FF!):A$=LEFT$(A$,LEN(A$)-2):X=X-2*(LEN(A$)+1
):GOSUB 686:RETURN
686   IF FF!>0 THEN A$=RIGHT$(A$,LEN(A$)-1):A$="+"+A$
688   RETURN
900   CLEAR 2500:MODE 0:PRINT CHR$(12):PRINT "VAKWERK":PRINT
902   PRINT "A.Vingerling, Sophiakade 5a, 3061 DK R'dam. Tel 010-521507"
904   READ N:N2=2*N-1:N=N-1
906   DIM A(1,N2-3),CM!(N2,N2),B!(1,N),KC!(1,N)
908   DIM L!(N2-3),BV!(N2),P(N2),X!(N2)
912   FOR I=0 TO 1:FOR J=0 TO N2-3:READ A:A(I,J)=A-1:NEXT:NEXT
916   FOR I=0 TO 1:FOR J=0 TO N:READ KC!(I,J):NEXT:NEXT
920   FOR I=0 TO 1:FOR J=0 TO N:READ B!(I,J):NEXT:NEXT
924   READ VAST,ROL
926   FOR J=0 TO N:B!(1,J)=-B!(1,J):NEXT:GOSUB 400
928   PRINT :PRINT "Staafkrachtberekening in een vlak vakwerk"
929   COLORT 5 15 0 0:COLORG 7 15 11 0
930   PRINT "met";N+1;" knooppunten en";2.0*(N+1)-3;" staven"
932   GOSUB 100:GOSUB 200:GOSUB 300
934   PRINT CHR$(12);"Belastingtabel":GOSUB 958
936   FOR I=0 TO N
938   PRINT I+1,B!(0,I),B!(1,I)
940   NEXT:PRINT
942   PRINT CHR$(12);"Staafkrachten":GOSUB 956
944   FOR I=0 TO N2-3
946   PRINT I+1,1+A(0,I);" en ";1+A(1,I),L!(I),X!(I)
948   NEXT:PRINT
950   PRINT CHR$(12);"Steunpuntsreacties":GOSUB 958
952   PRINT VAST,X!(N2-2),X!(N2-1)
953   PRINT CHR$(12);ROL," ",X!(N2):PRINT
954   GOSUB 600:PRINT "Einde vakwerkprogramma ":GOTO 9999
956   PRINT "Staaf","   Tussen","  "," Lengte"," Kracht":PRINT " Nr. "," ",
" ","   [m]","   [N]":RETURN
958   PRINT "Knoopp."," Hor."," Vert.":PRINT " Nr."," [N]","   [N]":RETU
RN
1000  REM aantal knooppunten
1010  DATA 9
1100  REM vormmatrix A%()
1110  DATA 1,2,3,4,5,6,7,8,9,2,3,3,4,4,5
1120  DATA 2,3,4,5,6,7,8,9,1,9,9,8,8,7,7
1200  REM knooppuntscoordinaten KC!()
1210  DATA 0,0,1000,2500,4000,5000,4000,2500,1000
1220  DATA 500,2500,3500,3500,3500,2500,2500,2500,2500
1300  REM belastingen B!()
1310  DATA 0,4850,4900,0,-4950,-5000,0,0,0
1320  DATA 0,50000,500,500,500,10000,0,0,0
1400  REM vast,rol
1410  DATA 7,1
2000  REM minitekst data
2010  DATA 0,35151411213132 34/,1,152524211131/,2,153534332313121121 31/,3,15
3534332313323131 11/
2020  DATA 4,151323333531/,5,151323333231211115 35/,6,351514112131323333 23/,
```

```
7,153534222221/
2040  DATA 8,252515112121313523 23/,9,351514132323343131 11/,+,22241333/,-,13
33/,.,2121/," ",/
2060  DATA E,1115213123332535/,V,008000335380/,R,008002821121617102355582/
2070  DATA L,039303250321/,H,039393759371/,O,202929072947/,N,202920022042/
2090  DATA [,222422312435/,J,222411221524/,X,12343214/,Y,212323142334/

9999  END
```



DEMO-1

```
10    REM ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
20    REM Jeroen Overvoorde Helmbloem 5 3068 AC Rotterdam
30    REM Telefoon 010-210426 Nederland  datum 25-11-1983
40    REM ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
50    REM -------------------------------------------------
60    REM Logo Overvoorde
70    REM -------------------------------------------------
100   MODE 6:COLORG 0 5 13 14:GOSUB 600:E=YMAX-110:L=E+40:C=17:YM=YMAX-20:XM=XMA
X:R=19
105   FILL 26,15+E 30,80+E 21:FILL 1,15+E 5,35+E 21:FILL 1,76+E 26,80+E 21
110   MX=50:MY=15+L:R=15:K=21:GOSUB 5000:R=10:K=20:GOSUB 5000
115   FILL 49,L 65,15+L 20:FILL 38,13+L 64,17+L 21:FILL 49,L 64,5+L 21
120   MX=85:R=15:G=MX+5:K=21:GOSUB 3000:R=10:K=20:GOSUB 3000
125   FILL 71,L 75,15+L 21
130   MX=110:R=15:K=21:GOSUB 5000:R=10:K=20:GOSUB 5000
135   MX=145:R=15:K=21:GOSUB 5000:R=10:K=20:GOSUB 5000
140   FILL 144,L 160,15+L 20:FILL 133,13+L 159,17+L 21:FILL 144,L 159,5+L 21
145   MX=170:G=MX-5:K=21:R=15:GOSUB 1000:G=MX:R=10:K=20:GOSUB 1000
150   FILL 165,L 169,15+L 21:FILL 180,L 184,15+L 21
200   MX=15:MY=15+E:R=18:K=20:G=MX:GOSUB 1000:MY=MY+3:R=15:GOSUB 3000:MY=MY-3
201   R=15:K=22:GOSUB 5000:R=10:K=20:GOSUB 5000
205   FILL 36,15+E 40,30+E 22:FILL 51,16+E 55,30+E 22
210   MX=50:R=15:K=22:G=MX+5:GOSUB 4000:R=10:G=MX:K=20:GOSUB 4000
215   MX=75:R=15:K=22:GOSUB 5000:R=10:K=20:GOSUB 5000
220   FILL 74,E 90,15+E 20:FILL 63,13+E 89,17+E 22:FILL 74,E 89,5+E 22
225   MX=110:R=15:K=22:G=MX+5:GOSUB 3000:R=10:K=20:GOSUB 3000
230   FILL 96,E 100,16+E 22
235   MX=135:R=15:K=22:G=MX+5:GOSUB 4000:R=10:G=MX:K=20:GOSUB 4000
240   FILL 121,15+E 125,30+E 22:FILL 136,16+E 140,30+E 22
245   MX=160:R=15:K=22:GOSUB 5000:R=10:K=20:GOSUB 5000
250   MX=195:R=15:K=22:GOSUB 5000:R=10:K=20:GOSUB 5000
255   MX=230:R=15:K=22:G=MX+5:GOSUB 3000:R=10:K=20:GOSUB 3000
260   FILL 216,E 220,16+E 22:FILL 265,E 269,45+E 22
265   MX=255:R=15:K=22:GOSUB 5000:R=10:K=20:GOSUB 5000
270   MX=290:R=15:K=22:GOSUB 5000:R=10:K=20:GOSUB 5000
275   FILL 289,E 305,15+E 20:FILL 278,13+E 304,17+E 22:FILL 289,E 304,5+E 22
280   FILL 0,E XMAX,E 20
300   MY=E-45:MX=107:GOSUB 1400:MX=125:GOSUB 1500:MX=136:GOSUB 1600:MX=154:GOSUB
 1700
310   MX=172:GOSUB 1600:MX=184:GOSUB 1800:MX=202:GOSUB 1900:MX=214:GOSUB 1600:MX
=232:GOSUB 1600
320   MX=250:GOSUB 1500:MX=262:GOSUB 1900
500   FOR T=#F TO 0 STEP -1:FOR I=#B8E7 TO #BFEF STEP #5A:POKE I,#20+T:WAIT TIME
 3:NEXT I:NEXT T
510   COLORG 0 0 0 0:WAIT TIME 30:COLORG 0 5 0 0:WAIT TIME 30:COLORG 0 0 13 0:WA
IT TIME 30:COLORG 0 0 0 14:WAIT TIME 30
520   COLORG 0 0 0 0:WAIT TIME 30:COLORG 0 0 0 14:WAIT TIME 30:COLORG 0 0 10 14:
WAIT TIME 30:COLORG 0 3 10 14:LOAD "JEROEN DEMO 2"
600   FOR I=#BFEF TO #B8E7 STEP -#5A:POKE I,#2F:NEXT I
610   RETURN
1000  D=R*R:FOR Y=0 TO R:X=SQR(D-Y*Y)
1010  DRAW G,MY+Y MX+X,MY+Y K:NEXT Y
1020  DOT MX+SQR(D),MY 0:RETURN
1400  R=7:K=23:GOSUB 5000:R=6:K=20:GOSUB 5000:DRAW MX-7,MY-12 MX-7,MY+7 23:RETUR
N
1500  R=7:K=23:G=MX+1:GOSUB 3000:R=6:K=20:GOSUB 3000:DRAW MX-6,MY-7 MX-6,MY 23:R
ETURN
1600  R=7:K=23:GOSUB 5000:R=6:K=20:GOSUB 5000:FILL MX,MY-7 MX+7,MY 20
1610  DRAW MX,MY-7 MX+7,MY-7 23:DRAW MX-7,MY MX+7,MY 23:RETURN
1700  R=7:K=23:GOSUB 5000:R=6:K=20:GOSUB 5000:FILL MX-7,MY-7 MX,MY 20:FILL MX,MY
 MX+7,MY+7 20
1710  DRAW MX-7,MY MX+7,MY 23:DRAW MX-7,MY-7 MX,MY-7 23:DRAW MX,MY+7 MX+7,MY+7 2
3:RETURN
1800  R=7:K=23:G=MX-1:GOSUB 1000:R=6:K=20:G=MX:GOSUB 1000:DRAW MX-1,MY-7 MX-1,MY
```

```
+1 23
1810  DRAW MX+6,MY-7 MX+6,MY+1 23:RETURN
1900  R=7:K=23:G=MX+1:GOSUB 4000:R=6:K=20:GOSUB 4000:DRAW MX-6,MY MX-6,MY+12 23
1910  DRAW MX-6,MY+7 MX,MY+7 23:RETURN
2000  D=R*R:FOR Y=0 TO R:X=SQR(D-Y*Y)
2010  DRAW G,MY-Y MX+X,MY-Y K:NEXT Y
2020  DOT MX+SQR(D),MY 0:RETURN
3000  D=R*R:FOR Y=0 TO R:X=SQR(D-Y*Y)
3010  DRAW MX-X,MY+Y G,MY+Y K:NEXT Y
3020  DOT MX-SQR(D),MY 0:RETURN
4000  D=R*R:FOR Y=0 TO R:X=SQR(D-Y*Y)
4010  DRAW MX-X,MY-Y G,MY-Y K:NEXT Y
4020  DOT MX-SQR(D),MY 0:RETURN
5000  D=R*R:FOR Y=0 TO R:X=SQR(D-Y*Y)
5010  DRAW MX-X,MY-Y MX+X,MY-Y K:DRAW MX+X,MY+Y MX-X,MY+Y K:NEXT Y
5020  DOT MX-SQR(D),MY 0:DOT MX+SQR(D),MY 0:RETURN
```

```
10    REM --- SUNDOWN ---
20    REM --- TOMISLAV MIKULIC ---
80    POKE #131,1
90    K=15
100   COLORG 9 3 10 0
110   MODE 2
112   GOSUB 22000
120   AD=#BFEF
130   N=12
140   FOR I=AD TO AD-(N*24) STEP -24
150   POKE I,0
160   NEXT I
170   GOSUB 5000
200   FOR I=AD TO AD-(N*24) STEP -24
210   FOR L=1 TO K
212   WAIT TIME 1
220   POKE I,L
230   NEXT L
240   NEXT I
250   GOSUB 5000
260   GOTO 140
5000  G=GETC:G=GETC:G=GETC
5010  G=GETC:IF G=0 THEN 5010
5020  RETURN
22000 REM CITRO
22010 CX=54:CY=37
22030 R!=6.0
22040 R1=R!+3
22050 R2=R1+5
22100 FOR Y=-R! TO R!
22110 X=SQR(R!*R!-Y*Y)
22120 DRAW X+CX,CY+Y CX-X,CY+Y 22
22130 NEXT Y
22190 N!=12.0
22200 F!=0.0:DF!=2.0*PI/N!
22202 FOR L=0 TO N!-1
22204 F!=L*DF!
22210 X1!=R1*COS(F!)+CX:Y1!=R1*SIN(F!)+CY
22220 X2!=R2*COS(F!)+CX:Y2!=R2*SIN(F!)+CY
22230 DRAW X1!,Y1! X2!,Y2! 22
22240 NEXT L
50000 RETURN
```

```
100   PRINT CHR$(12):COLORT 7 0 7 7
110   CURSOR 0,19:PRINT "SCREEN":POKE #BFEF-4*#86,#4A
120   CURSOR 0,15:PRINT "LAY-OUT":POKE #BFEF-8*#86,#4A
130   CURSOR 10,10:PRINT "4 VERSCHILLENDE DEMO'S":POKE #BFEF-13*#86,#6
A
140   CURSOR 44,9:PRINT "Door Luc Beyens"
150   CURSOR 0,6:PRINT "Druk op de SPATIEBALK om verder te gaan"
160   H=GETC:IF H<>32.0 THEN GOTO 160
200   PRINT CHR$(12):COLORT 13 0 13 13:POKE #74,1:POKE #75,#FF:REM CUR
SOR = BLOKJE
210   CURSOR 0,18:PRINT "KIES UIT DE VOLGENDE MOGELIJKHEDEN:":POKE #BF
EF-5*#86,#6A
220   CURSOR 15,14:PRINT "1 - REKLAMESPOT"
230   CURSOR 15,12:PRINT "2 - ROTERENDE TEKST"
240   CURSOR 15,10:PRINT "3 - VIDEO TEKST"
250   CURSOR 15,8:PRINT "4 - TEKSTMARKERING"
260   CURSOR 40,3:INPUT "UW KEUZE IS: ";A
270   ON A GOTO 1000,2000,3000,4000
1000  PRINT CHR$(12):COLORT 14 0 14 14
1010  CURSOR 25,16:PRINT "REKLAMESPOT"
1020  CURSOR 25,15:PRINT "==========="
1030  CURSOR 11,13:PRINT "... of wat men nog met tekst kan doen..."
1040  CURSOR 35,10:PRINT "Auteur: L.BEYENS"
1050  CURSOR 0,1:PRINT "Druk op de SPATIEBALK om verder te gaan"
1060  H=GETC:IF H<>32.0 THEN GOTO 1060
1070  PRINT CHR$(12):PRINT :PRINT :PRINT
1080  PRINT "Dit programma laat een tekst van max.16 kar. in formaat"
1081  PRINT "#5A letter voor letter op het scherm verschijnen vanaf"
1082  PRINT "links. De tekst wordt automatisch gecentreerd volgens"
1083  PRINT "zijn lengte."
1084  PRINT "Evenals in het programma 'ROTERENDE TEKST' zijn ook hier"
1085  PRINT "alle mogelijkheden inzake kleurkeuze, affichagesnelheid"
1086  PRINT "en aamtal affichages ter beschikking."
1087  PRINT "Na afloop blijft de tekst op het scherm staan tot op de"
1088  PRINT "SPATIEBALK gedrukt wordt."
1089  PRINT
1090  PRINT "Lijn 1400 kan naar believen aangepast worden: GOTO 1100"
1091  PRINT "of GOTO 1200."
1092  PRINT "GOTO 1100 laat toe een nieuwe tekst in te voeren;"
1093  PRINT "GOTO 1200 laat de routine steeds opnieuw herbeginnen tot"
1094  PRINT "op de BREAK-toets gedrukt wordt."
1095  CURSOR 0,1:PRINT "DRUK NU OP DE SPATIEBALK"
1099  H=GETC:IF H<>32.0 THEN GOTO 1099
1100  PRINT CHR$(12):COLORT 8 0 8 8:POKE #75,32
1110  INPUT "Tekst (max.16 kar.)      ";A$:PRINT
1120  INPUT "Achtergrondkleur         ";AK:PRINT
1130  INPUT "Tekstkleur               ";TK:PRINT
1140  INPUT "Aantal affichages        ";AR:PRINT
1150  INPUT "Affichagesnelheid (3-20) ";RS:PRINT
1200  PRINT CHR$(12):COLORT AK TK AK AK
1210  A=0.0
1215  REM CENTREREN
1220  C=LEN(A$):T=0.0
1230  IF C>14.0 THEN P=0.0:GOTO 1310
1240  IF C>12.0 THEN P=1.0:GOTO 1310
1250  IF C>10.0 THEN P=2.0:GOTO 1310
1260  IF C>8.0 THEN P=3.0:GOTO 1310
1270  IF C>6.0 THEN P=4.0:GOTO 1310
1280  IF C>4.0 THEN P=5.0:GOTO 1310
1290  IF C>2.0 THEN P=6.0:GOTO 1310
1300  REM AFFICHAGE
1310  CURSOR P,11
1320  POKE #BFEF-12*#86,#5A
1330  PRINT MID$(A$,T,1)
1340  WAIT TIME RS
```

```
1350  P=P+1.0:T=T+1.0
1360  IF T=C THEN GOTO 1380
1370  GOTO 1310
1380  WAIT TIME 50:A=A+1.0:IF A=AR THEN GOTO 1399
1390  PRINT CHR$(12):GOTO 1220
1399  H=GETC:IF H<>32.0 THEN GOTO 1399
1400  GOTO 200:REM MENU
2000  PRINT CHR$(12):COLORT 14 0 14 14
2010  CURSOR 22,16:PRINT "ROTERENDE TEKST"
2020  CURSOR 22,15:PRINT "================"
2030  CURSOR 0,13:PRINT "Een toepassing op het programma ''FILM TITLE
SIMULATION''"
2040  CURSOR 0,12:PRINT "(The Best of DAInamic 80-81)"
2050  CURSOR 0,1:PRINT "DRUK NU OP DE SPATIEBALK"
2060  H=GETC:IF H<>32.0 THEN GOTO 2060
2070  PRINT CHR$(12):PRINT :PRINT :PRINT
2080  PRINT "Dit programma laat een ingetikte tekst (T$) - max.80 kar.
 -"
2081  PRINT "over het scherm roteren van R naar L, ter hoogte van lijn
 12"
2082  PRINT "(midden van het scherm). De karaktergrootte is #5A."
2083  PRINT "Het aantal rotaties (AR) alsmede de rotatiesnelheid (RS)"
2084  PRINT "kunnen vrij bepaald worden. Dit is evenzo voor de achter-
"
2085  PRINT "grondkleur (AK) en de tekstkleur (TK)."
2086  PRINT "Daarenboven zal, nadat de tekst een aantal keren over het
"
2087  PRINT "scherm gelopen heeft, het laatste gedeelte van de tekst (
C$)"
2088  PRINT "- max.16 kar. - in het midden blijven staan, 10 keer flit
sen"
2089  PRINT "en opnieuw blijven staan tot op de SPATIEBALK gedrukt wor
dt."
2090  PRINT
2091  PRINT "Lijn 2380 kan naar believen aangepast worden: GOTO 2100"
2092  PRINT "of GOTO 2200."
2094  PRINT "GOTO 2100 laat toe een nieuwe tekst in te voeren;"
2095  PRINT "GOTO 2200 laat de rotatie steeds opnieuw herhalen."
2096  PRINT "Met de BREAK-toets kunt U dan uit de routine geraken."
2097  CURSOR 0,1:PRINT "DRUK NU OP DE SPATIEBALK"
2099  H=GETC:IF H<>32.0 THEN GOTO 2099
2100  PRINT CHR$(12):COLORT 8 0 8 8:POKE #75,32:REM CURSOR BLANK
2110  INPUT "Eerste tekstdeel + SPATIE ! ";B$:PRINT
2120  INPUT "Tweede tekstdeel            ";C$:PRINT
2130  INPUT "Achtergrondkleur            ";AK:PRINT
2140  INPUT "Tekstkleur                  ";TK:PRINT
2150  INPUT "Aantal rotaties             ";AR:PRINT
2160  INPUT "Rotatiesnelheid             ";RS:PRINT
2200  PRINT CHR$(12):COLORT AK TK AK AK
2210  C=LEN(C$):B=LEN(B$)+C
2220  A$="                    ":REM 20 SPATIES
2230  A1$=LEFT$(A$,C)
2240  T$=A1$+B$+C$+A$
2250  T=LEN(T$)
2260  FOR A=1.0 TO AR
2270  FOR T1=0.0 TO T-C
2280  T1$=MID$(T$,T1,C)
2285  P=-C+10.0:IF P<0.0 THEN P=1.0:REM CENTREREN
2290  CURSOR P,11:POKE #BFEF-12*#86,#5A
2300  PRINT T1$:WAIT TIME RS
2310  NEXT:NEXT
2320  T2$=MID$(T$,B,C)
2330  CURSOR P,11:PRINT T2$:WAIT TIME 100
2340  FOR F=1.0 TO 10.0
2350  COLORT AK AK AK AK:WAIT TIME 8:COLORT AK TK AK AK:WAIT TIME 8
2360  NEXT
```

```
2370  H=GETC:IF H<>32.0 THEN GOTO 2370
2380  GOTO 200:REM MENU
3000  PRINT CHR$(12):COLORT 14 0 14 14:DIM T$(25.0)
3010  CURSOR 22,16:PRINT "VIDEOTEKST"
3020  CURSOR 22,15:PRINT "=========="
3030  CURSOR 0,13:PRINT "Een toepassing op het programma ''VIDEO TEXT'
'"
3040  CURSOR 0,12:PRINT "(The Best of DAInamic 80-81)"
3050  CURSOR 0,1:PRINT "Druk op de SPATIEBALK om verder te gaan"
3060  H=GETC:IF H<>32.0 THEN GOTO 3060
3070  PRINT CHR$(12):PRINT :PRINT :PRINT
3080  PRINT "Met dit programma kunt U op het scherm meerdere tekstlijn
en"
3081  PRINT "afbeelden. U hebt de keus tussen $ verschillende karakter
s"
3082  PRINT "formaten (#4A -> #7A). Het aantal lijnen hangt af van de"
3083  PRINT "gekozen lettergrootte."
3084  PRINT "Ook hier kunnen achtergrondkleur en tekstkleur vrij"
3085  PRINT "bepaald worden."
3086  PRINT "Door toepassing van bv. de flitsroutine uit het programma
"
3087  PRINT "''ROTERENDE TEKST'', kan de affichage nog aantrekkelijker
"
3088  PRINT "gemaakt worden."
3089  PRINT "De tekst blijft zolang op het scherm staan tot op de SPAT
IE-"
3090  PRINT "BALK gedrukt wordt."
3091  PRINT
3092  PRINT "Wanneer U in lijn 3480 'GOTO 200' wijzigt in 'GOTO 3100',
"
3093  PRINT "kunt U een nieuwe tekst inbrengen, zonder langst het"
3094  PRINT "''MENU'' te passeren."
3095  PRINT "Vanzelfsprekend dient dan de BREAK-toets gebruikt om uit"
3096  PRINT "de routine te geraken."
3098  CURSOR 0,1:PRINT "DRUK NU OP DE SPATIEBALK"
3099  H=GETC:IF H<>32.0 THEN GOTO 3099
3100  PRINT CHR$(12):COLORT 8 0 8 8:POKE #75,32
3110  DIM T$(25.0)
3130  INPUT "ACHTERGRONDKLEUR ";AK:PRINT
3140  INPUT "TEKSTKLEUR       ";TK:PRINT
3150  PRINT "KARAKTERGROOTTE  "
3160  FOR A%=1 TO 4
3170  READ B%
3180  PRINT " ";A%;" = ";B%;
3190  PRINT TAB(11);:READ C%
3200  PRINT "REGELS MET";C%;
3210  PRINT TAB(25);:PRINT "KARAKTERS"
3220  NEXT
3230  RESTORE
3240  INPUT "KEUZE INVOEREN AUB (1-4) ";C:PRINT
3250  FOR D=1.0 TO C
3260  READ B%
3270  READ C%
3280  NEXT
3290  RESTORE
3300  FOR E%=1 TO B%
3310  PRINT "WAT IS DE TEKST VOOR REGEL NR ";E%
3320  PRINT "MAX.";C%;" KARAKTERS!"
3330  INPUT T$(E%):PRINT
3340  IF LEN(T$(E%))>C% THEN GOTO 3320
3350  NEXT
3360  COLORT AK TK AK AK:PRINT CHR$(12);
3370  IF C=2 THEN B%=B%*2
3380  FOR F%=1 TO B%
3390  IF C=1 THEN G=#4A
3400  IF C=2.0 THEN G=#5A
```

```
3410  IF C=3.0 THEN G=#6A
3420  IF C=4.0 THEN G=#7A
3430  PRINT T$(F%)
3440  IF C=2.0 THEN PRINT
3450  POKE #BFEF-((F%-1)*#86),G
3460  NEXT
3470  H=GETC:IF H<>32.0 THEN GOTO 3470
3480  GOTO 200:REM MENU
3500  DATA 8,5
3510  DATA 5,15
3520  DATA 13,37
3530  DATA 23,60
4000  PRINT CHR$(12):COLORT 14 0 14 14
4010  CURSOR 25,16:PRINT "MARKEERSTIFT"
4020  CURSOR 25,15:PRINT "============"
4030  CURSOR 0,13:PRINT "Een toepassing op het programma ''SPOT ON TEX
T''"
4040  CURSOR 0,12:PRINT "(F.H. DRUYFF - DAInamic 10/82)"
4050  CURSOR 0,1:PRINT "DRUK NU OP DE SPATIEBALK"
4060  H=GETC:IF H<>32.0 THEN GOTO 4060
4070  PRINT CHR$(12):PRINT :PRINT :PRINT
4071  PRINT "Met dit programma kunt u het scherm vullen met 1-22 lijne
n"
4072  PRINT "tekst in normale karaktergrootte (#7A)."
4073  PRINT "Zoals bij de vorige programma's kunt U ook hier de achter
-"
4074  PRINT "grondkleur en de tekstkleur vrij kiezen."
4075  PRINT "Daarenboven kunt U op gelijk welke plaats op het scherm"
4076  PRINT "tekstmarkeringen aanbrengen. De markeerkleur evenals de"
4077  PRINT "kleur van de tekst in de gemarkeerde gedeelten is ook"
4078  PRINT "vrij te bepalen."
4079  PRINT "Om nog meer aandacht op de gemarkeerde tekstgedeelten"
4080  PRINT "te vestigen is de mogelijkheid ingebouwd deze te laten"
4081  PRINT "knipperen."
4082  PRINT :PRINT "Door in lijn 4399 ''POKE #FF05,255'' te wijzigen i
n bv."
4083  PRINT "''POKE #FF05,5'' kunt U de affichage laten gebeuren op"
4084  PRINT "leessnelheid."
4085  PRINT :PRINT "Op volgende bladzijde vindt U nog meer nuttige inf
ormatie."
4086  CURSOR 0,1:PRINT "DRUK NU OP DE SPATIEBALK"
4087  H=GETC:IF H<>32.0 THEN GOTO 4087
4088  PRINT CHR$(12):PRINT :PRINT :PRINT "Het inbrengen van een
 bijkomende programmalijn op"
4089  PRINT "lijn nr. 4521, zal tot gevolg hebben dat de markeringen "
4090  PRINT "progressief gevuld worden."
4091  PRINT "Schakel over naar PROGRAMMA-MODE en voer in:"
4092  PRINT "''4521 FOR E = 0 TO 8:POKE D,2^J:NEXT''."
4093  PRINT :PRINT "Een ALTERNATIEVE KNIPPERROUTINE kan erin bestaan"
4094  PRINT "door lijn 4528 als volgt te wijzigen:"
4095  PRINT "''4528 FOR E = 20 TO 1 STEP -1:COLORT AK TK AK MK:"
4096  PRINT "WAIT TIME E:COLORT AK TK MK MT:WAIT TIME E:NEXT''."
4097  PRINT :PRINT "Lijn 4620 laat terugkeren naar het MENU. (Evt. te
wijzigen)."
4098  CURSOR 0,1:PRINT "DRUK NU OP DE SPATIEBALK"
4099  H=GETC:IF H<>32.0 THEN GOTO 4099
4100  PRINT CHR$(12):COLORT 8 0 8 8
4110  CLEAR 3000:DIM T$(25.0),LN(25.0),BP(25.0),AP(25.0)
4120  INPUT "HOEVEEL LIJNEN TEKST (MAX.22)       ";AL:PRINT
4130  INPUT "WELKE ACHTERGRONDKLEUR              ";AK:PRINT
4140  INPUT "WELKE TEKSTKLEUR                    ";TK:PRINT
4150  INPUT "WELKE MARKEERKLEUR                  ";MK:PRINT
4160  INPUT "WELKE TEKSTKLEUR IN DE MARKERINGEN  ";MT:PRINT
4170  INPUT "WILT U DE MARKERINGEN LATEN KNIPPEREN (J/N) ";A$
4200  PRINT :PRINT "TIK NU UW TEKST IN":WAIT TIME 40:PRINT CHR$(12)
4210  FOR A%=1 TO AL
```

```
4220  PRINT A%;:INPUT T$(A%):PRINT
4230  NEXT
4300  PRINT "NOTEER NU WELKE TEKSTGEDEELTEN U WENST TE MARKEREN"
4310  INPUT "HOEVEEL MARKERINGEN ";AM:PRINT
4320  FOR B%=1 TO AM
4330  PRINT "MARKERING NR     ";B%;:PRINT
4340  INPUT "LIJNNUMMER       ";LN(B%):PRINT
4350  INPUT "BEGINPOSITIE     ";BP(B%):BP(B%)=BP(B%)+3.0:PRINT
4360  INPUT "AANTAL POSITIES ";AP(B%):AP(B%)=AP(B%)*2.0-1.0:PRINT
4370  NEXT
4399  POKE #FF05,255
4400  PRINT CHR$(12):POKE #75,32
4410  FOR A%=1 TO AL
4420  PRINT T$(A%)
4430  NEXT
4500  COLORT AK TK MK MT
4505  FOR C%=1 TO AM
4515  FOR D=#BFEF-LN(C%)*#86-BP(C%)*2.0-3.0 TO D-AP(C%) STEP -2.0
4520  POKE D,#FF
4525  NEXT
4526  IF A$="J" THEN GOTO 4528
4527  IF A$="N" THEN GOTO 4535
4528  FOR E=1.0 TO 10.0:COLORT AK TK AK MK:WAIT TIME 8:COLORT AK TK MK
 MT:WAIT TIME 8:NEXT
4535  WAIT TIME 200:NEXT
4599  H=GETC:IF H<>32.0 THEN GOTO 4599
4600  GOTO 200
```

```
10    REM VASERELLI by R.SIP
20    REM enter in utility with -substitute-
30    REM start with : MODE 5 : CALLM #300
```

```
0300 F5 C5 D5 3E 00 32 0C 04 3E 0F 32 0E 04 3E 0F 32
0310 0D 04 3A 0D 04 C6 05 32 0F 04 3A 0E 04 32 10 04
0320 CD CB 03 3A 0D 04 57 3E 24 92 32 0F 04 CD CB 03
0330 3A 0D 04 C6 05 32 0F 04 3A 0E 04 5F 3E 1F 93 32
0340 10 04 CD CB 03 3A 0D 04 57 3E 24 92 32 0F 04 3A
0350 0E 04 5F 3E 1F 93 32 10 04 CD CB 03 3A 0E 04 C6
0360 05 32 0F 04 3A 0D 04 32 10 04 CD CB 03 3A 0E 04
0370 5F 3E 24 93 32 0F 04 3A 0D 04 32 10 04 CD CB 03
0380 3A 0E 04 C6 05 32 0F 04 3A 0D 04 57 3E 1F 92 32
0390 10 04 CD CB 03 3A 0E 04 5F 3E 24 93 32 0F 04 3A
03A0 0D 04 57 3E 1F 92 32 10 04 CD CB 03 3A 0C 04 C6
03B0 01 32 0C 04 3A 0D 04 3D 32 0D 04 C2 12 03 3A 0E
03C0 04 3D 32 0E 04 C2 0D 03 C3 08 03 3A 10 04 11 D0
03D0 02 21 00 00 37 3F 1F D2 DB 03 19 B7 CA E5 03 EB
03E0 29 EB C3 D4 03 11 44 66 19 3A 0F 04 17 47 7D 90
03F0 6F 7C DE 00 67 E5 C1 16 08 3A 0C 04 02 03 3E AA
0400 02 21 59 00 09 E5 C1 15 C2 F9 03 C9 28 01 02 06
0410 02
```

ERRATUM EPROM PROGRAMMER (N 19 p. 386)

PA0 - PA7 : dienen omgewisseld met PB0 - PB7

In het programma :

lijnen 071 & 072 vervangen door :

```
MVI A,AIN
CALL RICOUT
```

tussen lijnen 076 & 077 toevoegen :

```
LDA ROMTYP
CPI 32
JMZ RD16
LXI D,:2260
JMP NXTBLK
```

lijn 77 veranderen in :

```
RD16 LXI D,:2200
```

A VENDRE (cause achat rev.7)

DAI complet UHF/PAL	30000 Bfr
DCR + cable + TOS	13000 Bfr
Televiseur couleur Hitachi	12000 Bfr

	55000 Bfr

L'ensemble d'une seule piece :50000 Bfr

contactez : Mallien JP 183 rue des Nobles 5761 SOYE

Use + , - , * , : , ^ , SQR , ! (faculty) .. to solve these mystery exercices, result should be 6 for all.

example : 2 + 2 + 2 = 6 (very easy indeed this one !)

```
 1  1  1 = 6
 2  2  2 = 6
 3  3  3 = 6
 4  4  4 = 6
 5  5  5 = 6
 6  6  6 = 6
 7  7  7 = 6
 8  8  8 = 6
 9  9  9 = 6
10 10 10 = 6
```

(solutions in next issue)

DAI graphics

SEND YOUR DRAWINGS TO DAINAMIC EDITOR