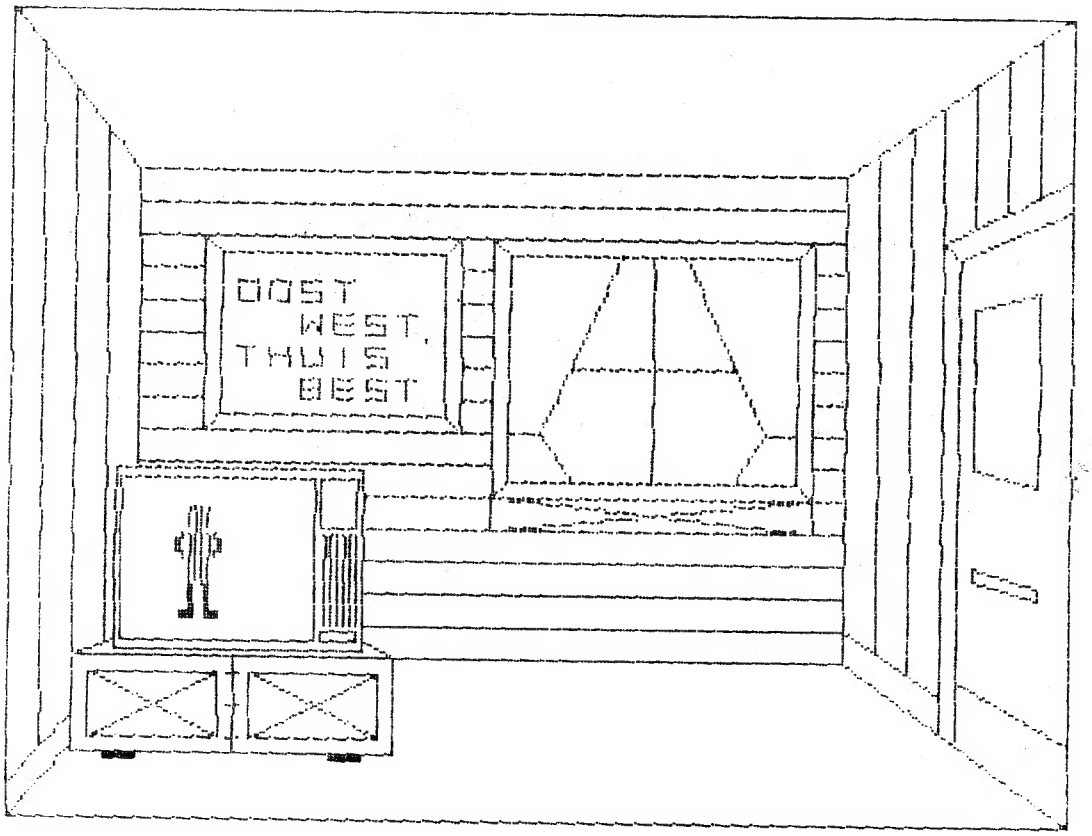
DAI NAMIC

NUMMER 4 *** APRIL 81



GEDRUKTE PERIODIEK verschijnt tweemaandelijks

Verantw. Uitgever : W.HERMANS HEIDE 98 3171 WESTMEERBEEK

COLOFON

DAInamic verschijnt tweemaandelijks.
abonnementsprijs is inbegrepen in de jaarlijkse contributie:
750 Bfr 50 Gld 50 Dm
Bij toetreding worden de verschenen nummers van de jaargang toegezonden.
DAInamic redactie:

Dirk Bonné
Freddy De Raedt
Wilfried Hermans
Jules Meulenbergs
Jos Schepens
Roger Theeuws
Bruno Van Rompaey
Jef Verwimp

vormgeving :Ludo van Mechelen

U wordt lid door storting van de contributie op nr406-3016141-33 van KREDIETBANK WESTMEERBEEK,via bank-instelling of POSTGIRO.
Abonnement loopt van januari tot december.

U kan telefonisch contact nemen op nr 016/698623.

---

correspondentieadres:

DAInamic
Heide 98
3171 WESTMEERBEEK BELGIE

DAInamic verschijnt de eerste week van de pare maanden.
Bijdragen zijn steeds welkom.

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

belangrijke ASCII-waarden in DAIpc

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

ASCII - HEX - ASCII CONVERSION TABLE

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

DAInamic
DAI USERS CLUB

Westmeerbeek april 81

Beste DAInamic-leden,

Zoals beloofd vindt U in dit nummer een samenvattend artikel over de priemgetallen.Onze wiskundige had hiervoor 14 pagina's nodig.Ook het artikel over FAST GRAF TEXT is nogal lijvig uitgevallen,zodat nummer 4 duidelijk een themanummer is.
Een aantal zeer interessante artikels en programma's zijn daardoor naar het volgende nummer verschoven:
-een uitvoerige studie van de prestaties van de MATHCHIP(9511)
-Waar raken we een EPROM kwijt in onze DAIpc?
-een uitvoerige studie van de PADDLE-functie
-schema van RS232 interface.(rond 5501)
-een paar cassette-clippers:om lastige signalen toch in te lezen.
-interface met een SPEECH SYNTHESISER BOARD(mogelijk te beluisteren op 11 april)
-beschrijving van een goedkope EPROM-programmer
-evaluatie van EPSON MX-80
-DOT DRAW & FILL ENTRY POINTS
Duidelijk nog geen gebrek aan copij.Toch willen we graag onze nieuwe wedstrijd voorstellen: DAI-application.
We verwachten een beschrijving van een project waarvoor U de DAIpc heeft ingeschakeld.We denken hier vooral aan de hardwaretoepassingen of uitbreidingen. Leuke programma's zijn natuurlijk NIET uitgesloten! Zorg er voor dat ook minder technisch geschoolden wat kunnen meenemen van uw proza! Technische schema's graag in goed contrast zwart/wit,formaat :maximaal A4.
De winnaar wordt deze keer bepaald door de leden,de prijs:
een MATHCHIP 9511 !!!
Kon U de verplaatsing op 11 april niet maken?In nummer 5 vertellen we U wel wat er te horen en te zien was...

We wensen U veel leute en lering

de redactie

BLADWIJZER

LINT VOOR TX-80 *** RIBBON FOR TX-80
Bij de opmaak van de NEWSLETTERS moeten wij altijd zorgen voor een goede zwarting van de copien op de printer.Vele linten geven na een paar beurten al erg grijze afdrukken.Een lint dat erg goed voldoet is het volgende:PELIKAN GR 1 13mm 1/2" zwart/black.
Het spoel past echter niet en het lint moet met de hand opgespoeld worden.Daartegenover staat echter een uitstekende kwaliteit op redelijk lange termijn.

Een KLEURRIJKE TIP van de heer T.G.VERBEKT:
Het kan voorkomen dat na aansluiten van een willekeurige TV er toch geen kleur wordt gegenereerd.Dit probleem deed zich ook bij mij voor.Na intensief speurwerk bleek dat als je een parasitaire capaciteit op het kristal (UHF-kaart) hield, de kleur wel werd gegenereerd.Een condensator van 10 pF blijkt meestal een afdoende oplossing te geven.De condensator wordt parallel over het kristal gesoldeerd.

Wij plannen publicatie van de ledenlijst in het JULI/AUGUSTUS nummer. Mogen wij de leden die bezwaar hebben tegen publicatie van hun naam en adres verzoeken dit voor 1 juli te berichten aan de redactie?

Interesse voor COMPUTERS & ONDERWIJS ? Dan loont het beslist de moeite contact te nemen met DIDACOM,zie adres op achterflap.
Dat DAIpc door de grafische en soundmogelijkheden het onderwijs-toestel bij uitstek is had U toch al vermoed?

You have an interesting article or program in french,english or german language? Please send it to DAInamic for publication.

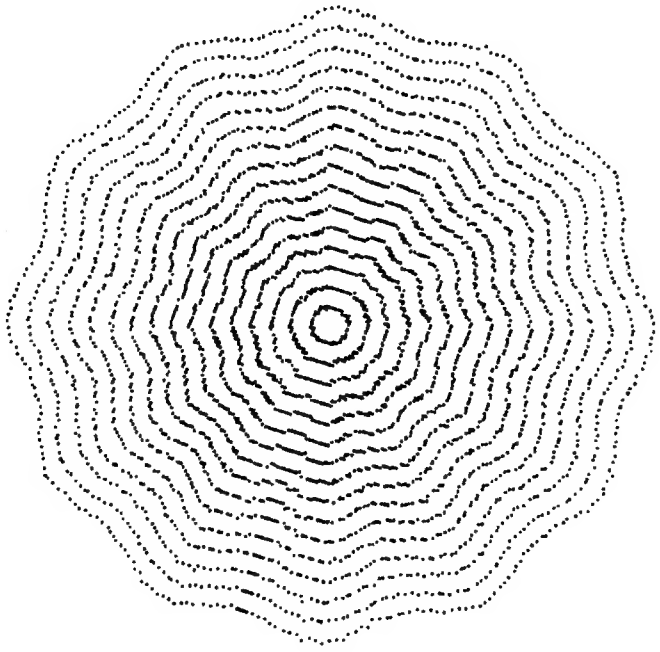
---

11 april *** DAInamic bijeenkomst *** 11 april ***DAInamic bijeenkomst

LIST

*** KANTEN KLEEDJE J.C.DELVOYE PAGE 01

```
1          REM kanten kleedje j.c.delvoye
5          COLORG 8 1 2 3
10         MODE 6
20         FOR A=YMAX/8.0 TO YMAX/1400.0 STEP -2.0
30     1     FOR I=0.0 TO 2.0*PI STEP PI/150.0
40     2       R=A*(3.0-COS(12.0*I)/10.0)
50     2       X=R*COS(I)
60     2       Y=R*SIN(I)
65     2       DOT XMAX/2-X,YMAX/2-Y 1
80     2       NEXT I
90     1     NEXT A
100        FOR B=1.0 TO 10.0
110    1     FOR A=1.0 TO 15.0
120    2       COLORG 0 A 12 3
130    2       WAIT TIME 150
140    2       NEXT A:NEXT B
160        GOTO 160
```



```
1      REM PROGRAMMA OM ALLE TEKENS TE LATEN ZIEN
5      PRINT CHR$(12)
10     FOR I=J*128+0 TO 127+J*128
20     X=I MOD 16
30     Y=(I-I MOD 16)/16
35     IF J=1 THEN Y=Y-8
40     CURSOR 7+3*X,22-2*Y
50     IF I=12 THEN GOTO 90
60     PRINT CHR$(I)
90     NEXT
100    J=1-J
110    IF GETC<>0 GOTO 5:GOTO 110
```

ROM ROUTINES & ENTRY PONTS AN INTRODUCTION

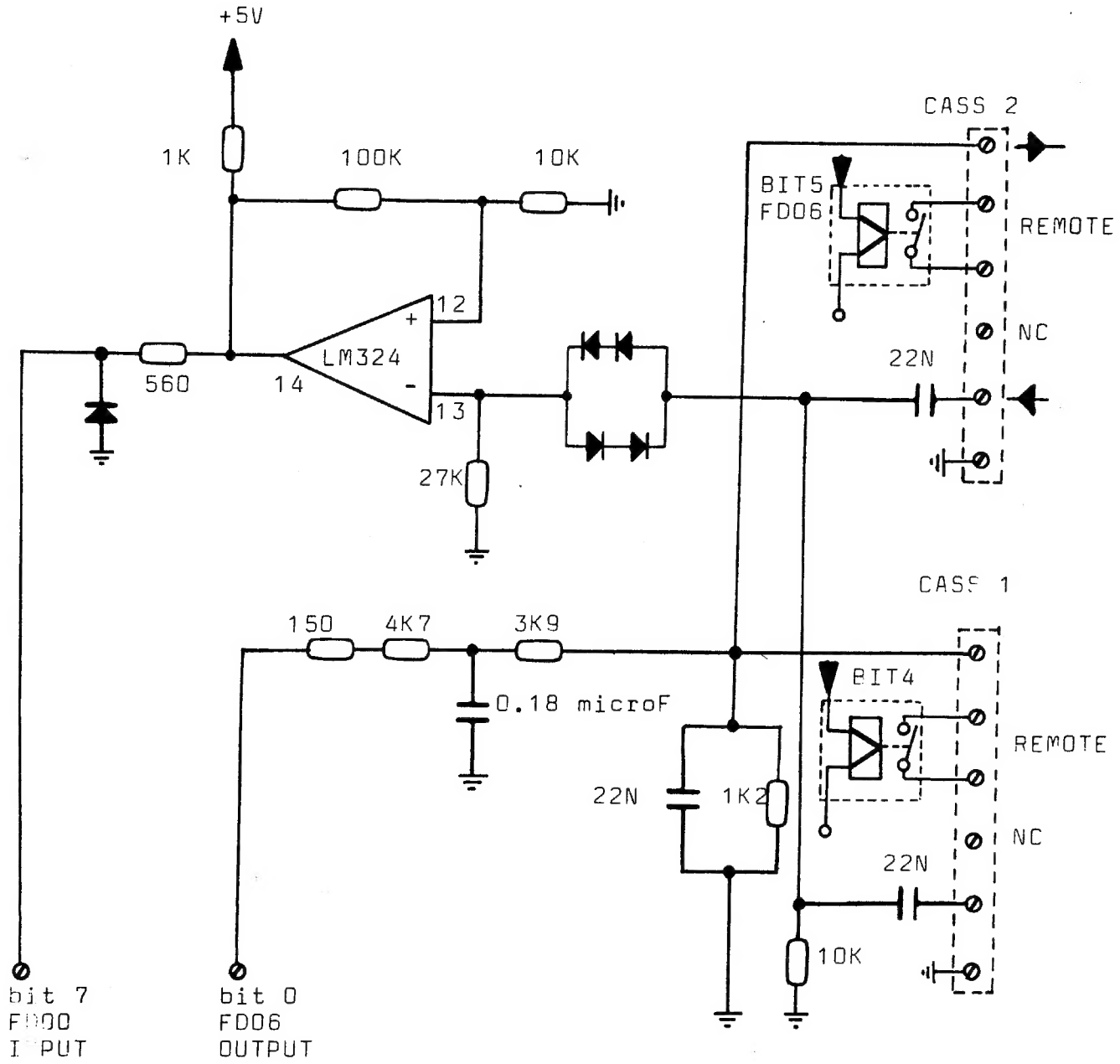
De meeste BASICROM-ROUTINES zijn NIET zo maar direct bruikbaar vanuit BASIC-programma's met CALLM.
De ROUTINE-CALL moet meestal gebeuren volgens een bepaald protocol:
- 8080 registers initialiseren
- correcte bank switching
- reference adressen initialiseren

Vanuit BASIC kan je niet de 8080 registers initialiseren zodat meestal een paar machinetaal instructies de eigenlijke CALL moeten vooraf gaan. Dit "miniprogramma" is dan wel op te roepen met CALLM.
De commando en functie-set is bij DAI-BASIC erg uitgebreid en meestal is er dan ook wel een eenvoudige BASIC oplossing,dit in tegenstelling tot de kortere BASIC-versies,waar je talloze keren moet beroep doen op CALLM, SYS of USR.
Kennis van de ENTRYPOINTS zal dus vooral nuttig en onontbeerlijk zijn bij de ontwikkeling van machinetaal programma's.
De ROM-BANK-SWITCHING valt ook nogal eenvoudig uit :dit wordt bij DAIpc opgelost door de "vreemde combinatie" RST X gevolgd door "DATA X".Diegenen die al met de DISASSEMBLER gestoeid hebben zullen waarschijnlijk al wel vraagtekens geplaatst hebben bij deze combinatie!
Zo zullen we vooral ontmoeten:

RST 1 + DATA voor UTILITY PACKAGE + ENCODE BASIC
RST 4 + DATA voor MATH PACKAGE
RST 5 + DATA voor SCREEN DRIVER PACKAGE

In tegenstelling tot de meeste ENTRYPOINTS zijn er wel een hele boel adressen en pointers die wel erg belangrijk kunnen zijn voor BASIC programma's,in zoverre bekend komen deze dan ook aan bod.
We starten met de routine "SSETM"=CHANGE MODE,wie nog niet erg vertrouwd is met 8080 machinetaal kan de informatie toch bestuderen door de eenvoudige routine op p.9 aan te passen.(lijn 304...)
OPMERKING: Bij CALLM worden de registers niet automatisch op stack bewaard ,een PUSH van de gebruikte registers zal dus meestal nodig zijn om een veilige terugkeer naar BASIC te garanderen.Bij verdere beschrijvingen duiden we dit mogelijk aan door PUSHALL,POPALL wat dus staat voor PUSH H,PUSH B, PUSH D,PUSH PSW resp. POP PSW,POP D,POP B,POP H.

CASSETTE INTERFACE DAIpc



overgetekend van de print,dus onder voorbehoud.

```
PAGE 01    FLASH : CHANGE COLOR 2 (4 COLOR MODE)

002                    *TO START : CHANGE VECTOR 7 TO HEX 305
003                    *TO CHANGE FLASHPERIOD SET PERIO (MIN 2 - MAX FF)
004                    *PERIO = 0 : NO FLASHING, ALWAYS COL0
005                    *PERIO = 1 : NO FLASHING, ALWAYS COL1
006                    COLCNB  EQU   :BFF6
007                    COLCH   EQU   :A
008                    VEC7    EQU   :D9A9
009                            ORG   :300
010 0300 19            PERIO   DATA  :19        X 20 MS = 1/2 FLASHPERIOD
011 0301 05            COL0    DATA  :5
012 0302 0F            COL1    DATA  :F
013 0303 00            TIM     DATA  :0         COUNTER INCR EACH 20 MS
014 0304 00            TFL     DATA  :0         FLAG FOR COLOR 0 OR 1
015 0305 F5                    PUSH  PSW
016 0306 E5                    PUSH  H
017 0307 210303                LXI   H,TIM      ADRES TIM IN H,L
018 030A 3A0003                LDA   PERIO
019 030D FE00                  CPI   :0
020 030F CA2203                JEQ   SET0       ALWAYS COL0
021 0312 FE01                  CPI   :01
022 0314 CA2E03                JEQ   SET1       ALWAYS COL1
023 0317 BE                    CMP   M
024 0318 D23F03                JGE   INCR       NO COL CHANGE IF TIM<=PERIO
025 031B 3A0403                LDA   TFL
026 031E B7                    ORA   A
027 031F CA2E03                JZ    SET1       TEST FLAG AND JUMP TO CHANGE
028 0322 AF            SET0    XRA   A
029 0323 320403                STA   TFL
030 0326 3A0103                LDA   COL0
031 0329 E60F                  ANI   :0F
032 032B C33803                JMP   SETC
033 032E 3E01          SET1    MVI   A,:1
034 0330 320403                STA   TFL
035 0333 3A0203                LDA   COL1
036 0336 E60F                  ANI   :0F
037 0338 C6A0          SETC    ADI   :A0        CONVERT COL TO COLCNTR BYTE
038 033A 32F6BF                STA   COLCNB
039 033D 3601                  MVI   M,:01      PRESET TIM
040 033F 34            INCR    INR   M
041 0340 E1                    POP   H
042 0341 F1                    POP   PSW
043 0342 C3A9D9                JMP   VEC7       START ROM PROGRAM RESTART 7
044 0345                       END

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

COL0   0301   COL1   0302   COLCH  000A   COLCNB BFF6
INCR   033F   PERIO  0300   SET0   0322   SET1   032E
SETC   0338   TFL    0304   TIM    0303   VEC7   D9A9
```

```
PAGE 01    FLASH : CHANGE COLOR 2 (4 COLOR MODE)

002                    *TO START : CHANGE VECTOR 7 TO HEX 305
003                    *TO CHANGE FLASHPERIOD SET PERIO (MIN 2 - MAX FF)
004                    *PERIO = 0 : NO FLASHING, ALWAYS COL0
005                    *PERIO = 1 : NO FLASHING, ALWAYS COL1
006                    COLCNB  EQU   :BFF6
007                    COLCH   EQU   :A
008                    VEC7    EQU   :D9A9
009                            ORG   :300
010 0300 19            PERIO   DATA  :19        X 20 MS = 1/2 FLASHPERIOD
011 0301 05            COL0    DATA  :5
012 0302 0F            COL1    DATA  :F
013 0303 00            TIM     DATA  :0         COUNTER INCR EACH 20 MS
014 0304 00            TFL     DATA  :0         FLAG FOR COLOR 0 OR 1
015 0305 F5                    PUSH  PSW
016 0306 E5                    PUSH  H
017 0307 210303                LXI   H,TIM      ADRES TIM IN H,L
018 030A 3A0003                LDA   PERIO
019 030D FE00                  CPI   :0
020 030F CA2203                JEQ   SET0       ALWAYS COL0
021 0312 FE01                  CPI   :01
022 0314 CA2E03                JEQ   SET1       ALWAYS COL1
023 0317 BE                    CMP   M
024 0318 D23F03                JGE   INCR       NO COL CHANGE IF TIM<=PERIO
025 031B 3A0403                LDA   TFL
026 031E B7                    ORA   A
027 031F CA2E03                JZ    SET1       TEST FLAG AND JUMP TO CHANGE
028 0322 AF            SET0    XRA   A
029 0323 320403                STA   TFL
030 0326 3A0103                LDA   COL0
031 0329 E60F                  ANI   :0F
032 032B C33803                JMP   SETC
033 032E 3E01          SET1    MVI   A,:1
034 0330 320403                STA   TFL
035 0333 3A0203                LDA   COL1
036 0336 E60F                  ANI   :0F
037 0338 C6A0          SETC    ADI   :A0        CONVERT COL TO COLCNTR BYTE
038 033A 32F6BF                STA   COLCNB
039 033D 3601                  MVI   M,:01      PRESET TIM
040 033F 34            INCR    INR   M
041 0340 E1                    POP   H
042 0341 F1                    POP   PSW
043 0342 C3A9D9                JMP   VEC7       START ROM PROGRAM RESTART 7
044 0345                       END

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

COL0   0301   COL1   0302   COLCH  000A   COLCNB BFF6
INCR   033F   PERIO  0300   SET0   0322   SET1   032E
SETC   0338   TFL    0304   TIM    0303   VEC7   D9A9
```

P roblemen
R aadsels
I nzichten
E n
M ethodes

GETALLEN

1. Definities

- Elk natuurlijk getal dat juist twee verschillende delers heeft is een priemgetal.
- Een natuurlijk getal dat geen priemgetal is, wordt een samengesteld getal genoemd.

Gevolgen: a. 1 is geen priemgetal.
b. 2 is het enige even priemgetal.
c. de twee enige delers van een priemgetal p zijn 1 en p.

2. Belang van de priemgetallen

Elk samengesteld getal is precies op één manier te schrijven als een product van priemgetallen. Men noemt dit ontbinden in priemfactoren. De priemgetallen kunnen bijgevolg als bouwstenen van de natuurlijke getallen worden beschouwd. Dit zou als gevolg kunnen hebben dat vele betrekkingen tussen natuurlijke getallen uitsluitend met priemgetallen moeten kunnen worden uitgedrukt. Dit is echter verre van waar. Tot op heden zitten er in de theorie van de priemgetallen nog vele onopgeloste raadsels, zodat het belang van de priemgetallen in de structuur van de natuurlijke getallen misschien toch moet gerelativeerd worden!?
Aan het einde van dit artikel vermelden we enkele belangrijke onopgeloste "geheimen" van de priemgetallen.

3. Algoritmen om priemgetallen te bepalen

Twee belangrijke algoritmen kunnen gebruikt worden om priemgetallen op te sporen:

a. het algoritme gebaseerd op de zeef van Eratosthenes
b. het algoritme gesteund op het systematisch testen op mogelijke priemdelers

We zullen beide algoritmen algemeen toelichten en laten volgen door een aantal BASIC-programma's, gebaseerd op deze algoritmen. Alle afgedrukte programma's werden ingestuurd door leden van DAInamic. De redactie heeft sommige een weinig aangepast en van een aantal gelijkaardige een equivalent gemaakt.
We danken in dit verband volgende leden (alfabetische orde): H Assink, M.J. Berkx, J Beumers, J Davids, Desausois, T de Vries, K Esveld, F Geuther, G Goethals, L Moentack, J Schepens, H van Cooten en R Verboom. Onze bijzondere felicitaties gaan naar Luc Moentack die ons de snelst lopende uitwerking van één van beide algoritmen toestuurde (11 seconden).

3.1 De Zeef van Eratosthenes voor priemgetallen tot 1000

Dit algoritme is gebaseerd op de successieve eliminatie van alle veelvouden van de priemgetallen tot de vierkantswortel uit 1000.Dat dit laatste voldoende is wordt gestaafd door het feit dat elk samengesteld getal niet groter dan 1000 noodzakelijk een priemdeler moet hebben kleiner dan of gelijk aan de vierkantswortel uit 1000.
Deze eliminatie kan gebeuren door in 1000 (500) adressen een bepaalde code (bv 0) te poken en daarna voor de niet-priemgetallen de adresinhoud te wijzigen. Met de peek-instructie kunnen nadien de gezochte priemgetallen gegenereerd worden.
Alle gepubliceerde programma's werden uitgeprint na een IMP INT, zodat alle variabelen integers zijn, tenzij ze expliciet met de floating-point-notatie (!) staan aangegeven.

Programma 1:

```
5     REM 11 SECONDEN
10    CLEAR 1100:DIM P(250.0):PRINT CHR$(12)
30    INPUT "FROM 1 TILL  ";LS:PRINT
60    GOSUB 500:K=1
90    FOR M=0 TO L-1:PRINT TAB(K);P(M);
100   K=K+5:IF K=56 THEN PRINT :K=1
110   NEXT M:END
500   P(0.0)=2:I=3:L=0:GOSUB 600:RS!=SQR(LS)
510   I=I+2:IF I>RS! THEN GOSUB 700:RETURN
520   R!=SQR(I):FOR J=1 TO 12
530   IF P(J)>R! THEN GOSUB 600:GOTO 510
540   IF (I/P(J))*P(J)=I GOTO 510
550   NEXT J
600   L=L+1:P(L)=I:A=3*I+20000:B=20000+LS:C=2*I
610   FOR D=A TO B STEP C:POKE D,1:NEXT D:RETURN
700   L=1:FOR N=20003 TO B STEP 2
720   IF PEEK(N)<>1 THEN P(L)=N-20000:L=L+1
730   NEXT N:RETURN
```

Commentaar: Dit programma laat toe de priemgetallen te berekenen gelegen tussen 1 en LS , waarbij LS waarden kan aannemen gelegen tussen 3 en 1600. We onderstellen verder in deze commentaar dat LS gelijk is aan 1000. Het programma reserveert 1000 geheugenplaatsen om (eventueel) een niet-priemgetalcode in te stockeren. In de lijnnummers 500 tot 550 worden de priemgetallen tot $\sqrt{1000}$ bepaald en in de subroutine in 600 en 610 worden telkens de onpare veelvouden van deze priemgetallen als kandidaat-priemgetal weggepookt. De routine in 700-730 slaat alle gevonden priemgetallen op in een array P, die tenslotte in de statements (60) 90-110 wordt uitgeprint. Belangrijk voordeel: alle berekende priemgetallen blijven in het geheugen beschikbaar.

Programma 2:

***** 14 SECONDEN *****

```
      REM 14 seconden ***
5        PRINT CHR$(12):PRINT " 2",
10       CLEAR 10000:M=1000:DIM A(1):P=VARPTR(A(1)):N=SQR(M)
20       M=M/2:FOR I=1 TO M:POKE P+I,0:NEXT
30       FOR I=1 TO N:IF PEEK(P+I)=#AB THEN 100:K=2*I+1:PRINT K,:J=I
40         J=J+K:IF J>M THEN 100:POKE P+J,#AB:GOTO 40
100        NEXT:FOR I=N+1 TO M:IF PEEK(P+I)<>#AB THEN PRINT 2*I+1,
110        NEXT:END
```

Commentaar: In 500 opeenvolgende adressen wordt code 0 gepookt; deze 500 adressen corresponderen met de 500 oneven getallen die moeten worden onderzocht; in lijn 40 wordt in de geheugenplaatsen die horen bij de veelvouden van een gevonden priemgetal éAB gezet; lijn 100 test op deze niet-priemgetalcode om de priemgetallen vanaf $\sqrt{1000}$ tot 1000 zelf te genereren. Met deze uitwerking van het algoritme wordt 2 niet als priemgetal bekomen, zodat dit vooraf moet gegeven worden.

Een tweede manier om de eliminatie van de priemveelvouden te bekomen, is gebruik te maken van een array. Elk element van die array kan men laten corresponderen met een oneven getal tussen 0 en 1000. Bij de start van het programma staat elk element van die array op 0. Door systematisch de inhoud van die array-elementen die horen bij de veelvouden van gevonden priemgetallen te wijzigen, elimineert men de niet-priemgetallen.

Programma 3:

*** 15 SECONDEN

```
      REM 15 seconden
5        CLEAR 2500
10       PRINT CHR$(12)
20       DIM P(250,1)
30       PRINT "START!!!":PRINT 2,:FOR K=0 TO 1:FOR I=1 TO 249
40           IF P(I,K)=0 THEN II=I*2+1+K*500:PRINT II,:GOSUB 100
50           NEXT I:NEXT K:PRINT "KLAAR!!!":END
100      IF K=0 THEN FOR J=I TO 249 STEP II:P(J,0)=1:NEXT J
110      JJ=J-250:IF JJ<250 THEN FOR J=JJ TO 249 STEP II:P(J,1)=1:NEXT
C        J
120      RETURN
```

Commentaar: Dit programma illustreert de Zeef van Eratosthenes door middel van een array P. Daar de maximale dimensie van een array-index 255 is,moet een 2-dimensionale matrix gebruikt worden om de 500 oneven getallen voor te stellen. Deze gedwongen constructie eist zeker tijd op bij het elimineren van de veelvouden. Lijn 40 test op de priemgetalcode en in de subroutine (100-120) wordt de inhoud van de array-elementen die horen bij de veelvouden van een gevonden priemgetal op 1 gezet (niet-priemgetalcode). Ook in dit programma wordt 2 niet als priemgetal gevonden, zodat de instructie PRINT 2, vereist is.

Programma 4:

Commentaar: De uitwerking van het algoritme volgens de Zeef is in dit programma analoog met deze in programma 3. Alleen correspondeert hier elk natuurlijk getal met een array-element, zodat nu ook het even priemgetal 2 ontstaat vanuit het algoritme. Dit eist echter tijdens de verwerking heel wat supplementaire testen en bewerkingen, zodat een langere duurtijd voor de verwerking noodzakelijk wordt.

*** 51 SECONDEN ***

```
      REM 51 seconden ***
2         REM dit programma berekent de priemgetallen van 2 tot 1000
3         REM in 51 seconden; er wordt gebruik gemaakt van de
4         REM "zeef van eratosthenes" door uit een rij getallen van
5         REM 1 tot 1000 alle veelvouden van elk gevonden priemgetal
6         REM te verwijderen; de rij getallen vormen de indices van
7         REM de 'array' r; omdat op de dai pc de dimensie van een 'array
C         '
8         REM niet de waarde 1000 mag hebben, is de 'array' 2-dimensionaa
C         l
9         REM gemaakt; hierdoor wordt het programma helaas langzamer.
10        CLEAR 5000:DIM R(3,250)
20        A=250:B=1000
30        S=INT(SQR(1000))
40        FOR T=2 TO B:I=INT((T-1)/A):J=T-A*I:IF R(I,J)=1 THEN 70
50          PRINT T,:IF T>S THEN 70
60          FOR K=T TO B STEP T:I=INT((K-1)/A):J=K-A*I:R(I,J)=1:NEXT K
70          NEXT T:END
```

Algemene opmerkingen : bij het Zeefalgoritme van Eratosthenes

- snel algoritme
- het "moeilijke" in dit algoritme is, indien het zeven gebeurd door middel van een array, het verband creëren tussen de indexwaarden van de 2-dimensionale array en het corresponderende priemgetal.

3.2 Algoritme gesteund op het systematisch testen op mogelijke priemdelers

Dit algoritme steunt op de eigenschap dat een getal N een priemgetal is als het niet deelbaar is door alle priemgetallen kleiner dan of gelijk aan $\sqrt{N}$.
De meest efficiënte vorm van dit algoritme bestaat erin elk oneven getal tussen 3 en 1000 te testen op mogelijke priemdelers, kleiner dan of gelijk aan zijn vierkantswortel. Wordt zulke priemdeler gevonden dan is het onderzochte getal niet priem.
Een moeilijkheid bij de opbouw van dit algoritme is het uittesten op priemde-

lers kleiner dan of gelijk aan de vierkantswortel uit het onderzochte getal. Hoe groter dit getal wordt, hoe meer priemdelers aan bod komen. De meest efficiënte manier hier is tijdens de verwerking een array op te bouwen met alle priemgetallen tot $\sqrt{1000}$. Op deze wijze worden geen overbodige testen en bewerkingen uitgevoerd en wordt de snelste tijd bekomen.
Sommige leden trachten dit probleem "benaderend" op te lossen door alle mogelijke priemdelers vooraf in een array te stoppen en deze dan bij elk priemonderzoek te gebruiken.

Programma 5a:

*** 28 SECONDEN ***

```
      REM 28 seconden ***
5        PRINT CHR$(12)
10       CLEAR 3000:DIM A(15.0)
20       PRINT " 2"," 3",
22       A(2.0)=3:E=3:S=2
25       B!=SQR(1000.0)
30       FOR I=3 TO 999 STEP 2
40         FOR J=2 TO S
50           IF I MOD A(J)=0.0 THEN 100
60           NEXT J
70         IF I<B! THEN A(E)=I:E=E+1:S=S+1
80         PRINT I,
100        NEXT I
110      END
```

Commentaar: In dit programma wordt de array met de priemdelers tot $\sqrt{1000}$ systematisch opgebouwd; dit programma verloopt duidelijk trager indien hier niet beperkt wordt in de array tot de priemgetallen kleiner dan of gelijk aan $\sqrt{1000}$. (programma 5b)

Programma 5b:

*** 110 SECONDEN ***

```
      REM 110 seconden ***
5        PRINT CHR$(12)
10       CLEAR 3000:DIM A(200.0)
20       A(1.0)=2:E=2:S=1:PRINT " 2",
30       FOR I=3 TO 999 STEP 2
40         FOR J=1 TO S
50           IF I MOD A(J)=0.0 THEN 100
60           NEXT J
70         A(E)=I:E=E+1:S=S+1
80         PRINT I,
100        NEXT I
110      END
```

Programma 6:

```
***  47 SECONDEN ***
```

```
      REM 47 seconden ***
2        DIM P!(10.0):P!(0.0)=2.0:P!(1.0)=3.0:P!(2.0)=5.0:P!(3.0)=7.0:P
C        !(4.0)=11.0:P!(5.0)=13.0:P!(6.0)=17.0
3        P!(7.0)=19.0:P!(8.0)=23.0:P!(9.0)=29.0:P!(10.0)=31.0:PRINT "ST
C        ART"
4        PRINT "2";:FOR A!=3.0 TO 1000.0 STEP 2.0:FOR B!=0.0 TO 10.0:IF
C             FRAC(A!/P!(B!))=0.0 THEN 6:NEXT B!:IF CURX>55.0 THEN PRINT
C
5          A=A!:PRINT A;:NEXT A!
6        IF A!=P!(B!) THEN 4:NEXT A!:PRINT " END":END
```

Commentaar: In dit programma worden alle priemgetallen tot $\sqrt{1000}$ in een array P gezet. Alle oneven getallen worden daarna op al deze delers getest. Dit geeft noodzakelijk een langere verwerkingstijd. Een tweede element dat een rol speelt in de langere tijdsduur is het gebruik van de FRAC-operator die floating-point variabelen eist. Deze laatste vragen meer verwerkingstijd dan integers.

Programma 7:

```
***  52 SECONDEN ***
```

```
      REM 52 seconden ***
5        POKE #131,1:PRINT CHR$(12):POKE #131,0:PRINT
10       PRINT "    ***** PRIME NUMBERS FROM 1 TILL 1000 *****"
20       PRINT :PRINT :PRINT "                              EXECUTION TIME = 5
C        2 SEC."
30       PRINT :PRINT "                         (without MATH-chip)":PRINT :
C        PRINT
40       CLEAR 1000
50       DIM P!(167.0)
60       P!(1.0)=3.0:I!=3.0:L!=1.0:PRINT " 2"," 3",
70       I!=I!+2.0:IF L!=167.0 GOTO 120
80       R!=SQR(I!):FOR J!=1.0 TO 167.0
90         IF P!(J!)>R! THEN L!=L!+1.0:P!(L!)=I!:P=I!:PRINT P,:GOTO 70
100        M!=I!/P!(J!):IF INT(M!)=M! GOTO 70
110        NEXT
120      PRINT :PRINT :PRINT "                ***** TERMINATED *****"
130      PRINT :PRINT "       *** ALL PRIME NUMBERS REMain in memory ***
C        "
```

Programma 10:

```
***  57 SECONDEN ***
```

```
      REM 57 seconden ***
10       PRINT CHR$(12):PRINT "STARTING":PRINT
20       FOR N=2 TO 3:K!=2.0:M!=N/K!:L!=INT(M!):IF L!<>M! OR L!=1.0 THE
C          N PRINT N,:NEXT N
30       FOR N=5 TO 1000 STEP 2:FOR K!=2.0+T! TO SQR(N) STEP 2.0:M!=N/K
C            !:L!=INT(M!):IF M!<>L! THEN NEXT K!:PRINT N,:T!=1.0
40         NEXT N
50       PRINT "FINISHED":END
```

Commentaar:

Programma's 8,9 en 10 wijken af van het algemeen voorgestelde algoritme doordat alle oneven getallen X tot aan $\sqrt{X}$ getest worden op alle mogelijke oneven delers, en niet alleen op de mogelijke priemdelers tot $\sqrt{X}$. Het tijdsverschil tussen deze programma's wordt vooral verklaard door het gebruik van de snelle MOD-operator in programma 8 en de floating-point variabelen in programma 10.

Programma 11:

```
***  NOG MEER PRIEMGETALLEN....***
```

```
      REM nog meer priemgetallen....***
2        DIM S(53.0):FOR K=1 TO 53:READ S(K)
3          G=G+S(K):FOR N=9 TO SQR(G) STEP 2:IF G MOD (N)=0 THEN 3:NEXT
C           N:PRINT G,
4          NEXT K:FOR K=6 TO 53:GOTO 3
5          DATA 2,1,2,2,4,2,4,2,4,6,2,6,4,2,4,6,6,2,6,4,2,6,4,6,8,4,2,4
C          ,2,4,8,6,4,6,2,4,6,2,6,6,4,2,4,6,2,6,4,2,4,2,10,2,10
```

Commentaar: Merkwaardig aan dit programma is de DATA-lijn. Door middel van dit DATA-statement wordt een getallentabel gecreëerd, waaruit alle 2-,3-,5- en 7-vouden zijn verdwenen. Elk getal G dat in de tabel voorkomt moet bijgevolg nog getest worden op priemdelers van 9 tot $\sqrt{G}$. Het programma test in feite alle mogelijke oneven delers van 9 tot $\sqrt{G}$ (lijn 3).Door alle data-elementen van deze tabel in een array in te lezen, kunnen steeds grotere getallen van deze tabel op hun priemeigenschap worden onderzocht. Het programma eindigt dan ook slechts met een foutmelding indien het berekende priemgetal te groot wordt om met een integervariabele te worden voorgesteld. De DATA-gegevens die de eigenlijke tabel zonder 2-,3-,5- en 7-vouden opstellen zijn de laatste 48. Telkens deze doorlopen zijn,

Programma 10:

```
***  57 SECONDEN ***
```

```
      REM 57 seconden ***
10       PRINT CHR$(12):PRINT "STARTING":PRINT
20       FOR N=2 TO 3:K!=2.0:M!=N/K!:L!=INT(M!):IF L!<>M! OR L!=1.0 THE
C          N PRINT N,:NEXT N
30       FOR N=5 TO 1000 STEP 2:FOR K!=2.0+T! TO SQR(N) STEP 2.0:M!=N/K
C            !:L!=INT(M!):IF M!<>L! THEN NEXT K!:PRINT N,:T!=1.0
40         NEXT N
50       PRINT "FINISHED":END
```

Commentaar:

Programma's 8,9 en 10 wijken af van het algemeen voorgestelde algoritme doordat alle oneven getallen X tot aan $\sqrt{X}$ getest worden op alle mogelijke oneven delers, en niet alleen op de mogelijke priemdelers tot $\sqrt{X}$. Het tijdsverschil tussen deze programma's wordt vooral verklaard door het gebruik van de snelle MOD-operator in programma 8 en de floating-point variabelen in programma 10.

Programma 11:

```
***  NOG MEER PRIEMGETALLEN....***
```

```
      REM nog meer priemgetallen....***
2        DIM S(53.0):FOR K=1 TO 53:READ S(K)
3          G=G+S(K):FOR N=9 TO SQR(G) STEP 2:IF G MOD (N)=0 THEN 3:NEXT
C           N:PRINT G,
4          NEXT K:FOR K=6 TO 53:GOTO 3
5          DATA 2,1,2,2,4,2,4,2,4,6,2,6,4,2,4,6,6,2,6,4,2,6,4,6,8,4,2,4
C          ,2,4,8,6,4,6,2,4,6,2,6,6,4,2,4,6,2,6,4,2,4,2,10,2,10
```

Commentaar: Merkwaardig aan dit programma is de DATA-lijn. Door middel van dit DATA-statement wordt een getallentabel gecreëerd, waaruit alle 2-,3-,5- en 7-vouden zijn verdwenen. Elk getal G dat in de tabel voorkomt moet bijgevolg nog getest worden op priemdelers van 9 tot $\sqrt{G}$. Het programma test in feite alle mogelijke oneven delers van 9 tot $\sqrt{G}$ (lijn 3).Door alle data-elementen van deze tabel in een array in te lezen, kunnen steeds grotere getallen van deze tabel op hun priemeigenschap worden onderzocht. Het programma eindigt dan ook slechts met een foutmelding indien het berekende priemgetal te groot wordt om met een integervariabele te worden voorgesteld. De DATA-gegevens die de eigenlijke tabel zonder 2-,3-,5- en 7-vouden opstellen zijn de laatste 48. Telkens deze doorlopen zijn,

werden deze veelvouden uit 210 (= 2.3.5.7) natuurlijke getallen geëlimineerd.

Een aantal programma's zijn nog verder verwijderd van het basisalgoritme omdat alle (ook de even) getallen worden getest en/of omdat oneven delers worden opgespoord tot N/2, daar waar $\sqrt{N}$ voldoende is.
We publiceren ook hiervan enkele programma's om de lezer te tonen dat in dit geval de verwerkingstijd snel toeneemt. Het al dan niet gebruik maken van integer-variabelen heeft ook in dit geval duidelijk invloed op de duurtijd.

Programma 12:

*** 447 SECONDEN ***

```
      REM 447 seconden ***

5         REM ***** inleiding priemgetallen *****
10        MODE 0:PRINT CHR$(12):COLORT 8 8 0 0
20        CURSOR 15,12:PRINT "PRIEMGETALLEN TOT 1000"
30        FOR Z!=1.0 TO 10.0
40          COLORT 8 0 0 0:WAIT TIME 30
50          COLORT 8 8 0 0:WAIT TIME 20
60          NEXT Z!:COLORT 8 0 0 0:PRINT CHR$(12)

100       REM ***** het programma *****
105       PRINT " 2.0",
110       FOR A!=2.0 TO 1000.0
120         FOR B!=2.0 TO INT(A!/2.0)
130           IF FRAC(A!/B!)=0.0 THEN NEXT A!:END
140         NEXT B!:PRINT A!,:NEXT A!
```

Programma 13:

*** 277 SECONDEN ***

```
      REM 277 seconden ***
2         B!=2.0:PRINT "S":PRINT 2.0,:FOR N!=3.0 TO 1001.0 STEP B!:FOR K
C             !=B!+A! TO N!/B! STEP B!:M!=N!/K!:IF M!<>INT(M!) THEN NEXT:
C           PRINT N!,:A!=1.0:NEXT
3         NEXT N!:PRINT "F"
```

4. Aantal priemgetallen en hun verdeling in de natuurlijke getallen

De verzameling van de priemgetallen is oneindig. Deze stelling werd reeds door Euclides bewezen. De spreiding van de priemgetallen in de verzameling van de natuurlijke getallen wordt nochtans groter naarmate de getallen groter worden. Het eerste Priemgetallen-Theorema beschrijft de asymptotische dichtheid van de priemgetallen als volgt:

$$\lim_{x \to \infty} \frac{\pi(x)}{x/\ln x} = 1$$

Hierin stelt $\pi(x)$ het aantal priemgetallen voor niet groter dan x. Dit theorema werd in 1895 bewezen door J Hadamard en C de la Vallée-Poussin. Het stelt dus dat het aantal priemgetallen kleiner dan of gelijk aan een gegeven getal x, voor zeer grote x-waarden gegeven wordt door $x/\ln x$.

Volgens het tweede Priemgetallen-Theorema kan de asymptotische dichtheid van de priemgetallen beschreven worden door:

$$\lim_{x \to \infty} \frac{\pi(x)}{\int_2^x \frac{dt}{\ln t}} = 1$$

Dit betekent dat voor grote waarden van x het aantal priemgetallen kleiner dan of gelijk aan x gegeven wordt door de bepaalde integraal $\int_2^x dt/\ln t$. Voor de onbepaalde intergraal vindt men volgende ontwikkeling:

$$\int \frac{dt}{\ln t} = \ln(\ln x) + \ln x + \frac{(\ln x)^2}{2.2!} + \frac{(\ln x)^3}{3.3!} + \frac{(\ln x)^4}{4.4!} + \ldots$$

Met een eenvoudig programma kan met de DAI pc deze formule geëvalueerd worden tot exponent 19, zodat we in staat zijn om voor verschillende waarden van x de berekeningen uit te voeren.

In onderstaande tabel hebben we voor een aantal x-waarden $\pi(x)$ opgegeven, en hun respectievelijke benaderde waarden voor $x/\ln x$ en $\int_2^x dt/\ln t$ berekend.

x	$\pi(x)$	$x/\ln x$	$\int_2^x dt/\ln t$
2	1	2,885	0
10	4	4,343	5,120
100	25	21,715	29,081
1000	168	144,765	176,563
1500	239	205,108	246,762

x	$\pi(x)$	x/lnx	$\int_2^x dt/\ln t$
2000	303	263,127	313,751
2500	367	319,528	378,541
3000	430	374,702	441,677
3500	489	428,894	503,501
4000	550	482,273	564,242
4500	610	534,961	624,068
5000	669	587,048	683,102
5500	725	638,606	741,441
6000	783	689,694	799,164
6500	842	740,356	856,328
7000	900	790,633	912,991
7500	950	840,557	969,192
8000	1007	890,155	1024,97
8500	1059	939,453	1080,35
9000	1117	988,470	1135,38
9500	1177	1037,230	1190,06
10000	1229	1085,740	1244,43
15000	1754	1559,932	1773,96
20000	2262	2019,491	2280,98 *
25000	2762	2468,74	2776,37
30000	3245	2910,09	3261,25
35000	3732	3345,09	3737,58
40000	4203	3774,78	4206,73
45000	4675	4199,95	4669,72
50000	5133	4621,17	5127,29
55000	5590	5038,9	5580,05
60000	6057	5453,5	6028,5
70000	6935	6274,51	6913,5
80000	7837	7086,05	7785,92
90000	8713	7889,5	8553,56 **
100000	10095	8685,889	9387,17

* vanaf hier moet exponent 18 in de ontwikkeling worden genomen

** vanaf hier geldt exponent 17

5. Enkele onopgeloste problemen in de theorie der priemgetallen

a. Als een getal p priem is en p+2 eveneens, noemt men dat getallenpaar een

priemgetallentweeling. De gemiddelde afstand tussen twee priemgetallen neemt steeds toe. Priemgetallentweelingen komen bijgevolg ook steeds zeldzamer voor. Houdt dit verschijnsel ooit op of is de verzameling van de priemgetallentweelingen oneindeig ? Is(1000000009649 , 1000000009651)de grootste priemgetallentweeling ?

b. Een tweede belangrijk raadsel is het Vermoeden van Goldbach. Dit kan als volgt geformuleerd worden: elk even getal groter dan 2 is de som van twee priemgetallen. Dit vermoeden is nog niet algemeen bewezen. Toch heeft men nog geen even getal gevonden waarvoor het niet waar is.

6. Een toemaatje: ooit al gehoord van de "gelukkige getallen" ?

Net zoals de priemgetallen kunnen de gelukkige getallen eveneens met een zeefproces worden bepaald. Dit algoritme kan als volgt worden omschreven:

1. schrijf alle natuurlijke getallen op tot 100 (we onderstellen hier dat alle gelukkige getallen tot 100 worden bepaald)
2. elimineer elk tweede getal: op deze wijze verdwijnen alle even getallen. De tabel die overblijft is :

1 3 5 7 9 11 13 15 17 19 21 23 25 27 ....

2. In plaats van nu elk 3-voud te elimineren zoals bij de zeef van Eratosthenes, schrappen we nu elk derde getal. De tabel die ontstaat is dan:

1 3 7 9 13 15 19 21 25 27 ....

3. In deze tabel staat na de 3 het cijfer 7, zodat nu elk zevende getal wordt geschrapt. We bekomen:

1 3 7 9 13 15 21 25 27 31 33 37 43 45...

4. Vervolgens wordt elk negende getal geëlimineerd:

1 3 7 9 13 15 21 25 31 33 37 43 45 49 51 55 63..

5. Na eliminatie van respectievelijk elk 13de,15de en tenslotte elk 21ste getal worden de gelukkige getallen tot 100 bekomen:

1 3 7 9 13 15 21 25 31 33 37 43 49 51

63 67 69 73 75 79 87 93 99

Deze tabel bestaat uit 23 gelukkige getallen, waarvan er 9 priem zijn en 14 samengesteld.

In het door de redactie onderzochte gebied (gelukkige getallen tot 100) is het Vermoeden van Goldbach overdraagbaar naar de gelukkige getallen: d.w.z. elk even getal is de som van twee gelukkige getallen. Is dit steeds zo ?

7. Enkele programma-ideetjes

1. Een programma dat tot 1000 alle priemgetallentweelingen afdrukt.

2. Een programma dat voor alle even getallen groter dan 2 en kleiner dan 1000 de twee priemgetallen afdrukt waarvan het de som is. (Vermoeden van Goldbach)

3. Een programma dat tot 1000 alle gelukkige getallen afdrukt.

4. Een programma dat tot 1000 alle getallen afdrukt die priem en gelukkig zijn.

Alles opsturen naar het gekende redactie-adres!!

8. Slot

Voor die enkelen die na het uittesten van alle programma's uit dit artikel de priemgetallen tot 1000 nog niet uit het hoofd kennen, drukken we ze als besluit af: ***PRIEMGETALLEN TOT 1000***

2	3	5	7	11
13	17	19	23	29
31	37	41	43	47
53	59	61	67	71
73	79	83	89	97
101	103	107	109	113
127	131	137	139	149
151	157	163	167	173
179	181	191	193	197
199	211	223	227	229
233	239	241	251	257
263	269	271	277	281
283	293	307	311	313
317	331	337	347	349
353	359	367	373	379
383	389	397	401	409
419	421	431	433	439
443	449	457	461	463
467	479	487	491	499
503	509	521	523	541
547	557	563	569	571
577	587	593	599	601
607	613	617	619	631
641	643	647	653	659
661	673	677	683	691
701	709	719	727	733
739	743	751	757	761
769	773	787	797	809
811	821	823	827	829
839	853	857	859	863
877	881	883	887	907
911	919	929	937	941
947	953	967	971	977
983	991	997		

COLOUR GRAPHICS

Inexpensive aids to presentation

by Robin Bradbeer

EARLY personal computers, say around 1976, were very limited in their graphics capabilities. A video screen capable of displaying 16 lines of 24 characters was considered normal. There was, therefore, little possibility of using a video screen to present other than basic alphanumeric information.

The introduction of memory-mapping techniques, where the individual character blocks on the screen corresponded to single memory locations, allowed more flexibility and machines like the Pet introduced reasonably-sophisticated graphics at a low price.

Since those days, two developments have occurred which could revolutionise the approach to the presentation of information on a screen, especially in the educational field. The first development was high-resolution graphics. Instead of the more usual 25 × 40 picture element (pixel) resolution, high-resolution graphics have up to 256 × 392 pixels. That allows much higher definition of diagrams on the video screen. For educational use, the advantages of high-resolution graphics are obvious.

The second development is inexpensive colour graphics. That development is so new that there are still many arguments about the uses and advantages of colour in graphics.

Most graphics systems work in similar ways. Whatever form of output is used, nearly every video display employs similar methods of generating the characters. A screen which can display 25 lines of 40 characters requires 1,000 memory locations to store the information. In the so-called memory-mapped systems, each character position on the screen corresponds to a particular memory location in the video memory.

All the computer does is to transfer the contents of the memory on to the screen. As each memory location can store one byte of eight bits, that means that 2^8 or 256 characters are available.

The most common method of encoding characters is the ASCII system. As it has only seven bits of information, only 128 characters are available. Most personal computer systems add another 128 to make up the number and they do so by creating graphics characters.

There is no standard for them so that Pet graphic symbols, for example, have different codes from Sorceror graphics. The codes are interpreted into letters and graphics symbols by a ROM called a character generator. It is clear that all this memory detracts from the amount of memory available for use in the system by the user.

Another method of generating video information is to split the screen into a series of discrete points, say 312 × 210. That requires a good deal more memory. Each point on the screen then corresponds to one bit in the memory, hence eight points require one byte of memory; 312 × 210 points therefore need 8K bytes.

The high-density graphics capability is available on a few computers, like the Apple II. It is a useful facility for educational users but unless complicated graphical analyses are needed, it is not ideal for business use.

Atari 800.

Limitations

To use graphics to the optimum effect involves adapting to the machine language of the computer. It is possible to do so from Basic with systems which have a character generator ROM but that limits the user to the characters available on the computer.

On the Pet, for example, it limits a user to 128 characters within the 25 × 40 format. Other systems have similar restraints.

Those with high-resolution graphics, however, like the Apple II or 380-Z, offer certain facilities which allow users not only to create their own characters but also to vary the number of pixels available for use. The more memory needed for graphics, the less is required for program memory. Consequently high-resolution graphics requires a system with a good deal of RAM.

High-resolution black-and-white graphics are available on a number of computer systems. The favourite method is to have a board which plugs in to the main computer system. That is how Research Machines, Acorn and most bus-structured systems work.

(continued on next page)

(continued from previous page)

Other systems have a built-in high-resolution capability. They include the Texas Instruments TI 99/4, DAI, PC1, Atari 400 and 800, Apple II and ITT 2020. Coincidentally, they are also machines which use colour.

Historically, colour graphics was available only as an expensive extra to mainframe or minicomputers. They needed high-quality colour monitors to give the definition required. The latest colour computers provide the video output as a normal TV signal and use normal colour television sets. Although this is cheaper, it restricts the resolution available.

With colour graphics, in its simplest form, at least two bytes are needed to store the information in each pixel. The first byte will define the character, the second the colour. In practice that is achieved by using less memory, with four bits — a nibble — being used to store the colour information. So most systems have the capacity to work with 16 colours.

Six systems

Of the six systems generally available on the U.K. market, three have been around for some time. The Apple II — and its later ITT equivalent — was the first low-cost colour system. The version available in the U.K. has the ability to split the screen into graphics or text or to partition the screen so that the lower four lines are text and the upper 21 are graphics.

Its maximum resolution is 280 × 192 points. Each point is addressable and the graphics generation requires each point to be programmed, although some commands from Basic permit that to be achieved easily. Lines can be drawn and individual dots specified with single commands.

The Apple has many programs available and is a favourite with many people. It is beginning to show its age, however, and is now more expensive than some of its newer competitors offering better graphics facilities.

The 380-Z has a plug-in graphics board which gives a resolution of up to 319 × 191 points, with a choice of four grey levels or colours for each point. Sixteen grey levels or colours are possible with a resolution of 159 × 95 points. It is possible to mix graphics and text and the plug-in board requires 16K bytes of RAM to function properly.

In the 319 × 191 mode, the user can program four options per point, as there are two bits per point. Normally one would use black and three grey levels or background colour and three colours. Thus it is possible to plot three graphs of different shades or different colours.

The relationship between the choice programmed at a point — 0, 1, 2 or 3 — and the output is defined by a high-speed RAM which acts as a look-up table, producing eight outputs. In black-and-white systems those outputs drive an eight-bit digital-to-analogue converter. In colour systems the eight bits drive separate two-bit D-A channels for the different primary colours.

The contents of the look-up table RAM are under software control, so the user can program a particular set of grey scales or colours, and can change that set in a short time between successive television scans.

	Graphics mode	Columns	Rows: Full	Rows: Split	No. of colours	RAM required
Text	0	40	24	—	2*	993
	1	20	24	20	5	513
	2	20	12	10	5	261
mixed text/ graphics	3	40	24	20	4	273
	4	80	48	40	2	537
	5	80	48	40	4	1017
	6	160	96	80	2	2025
	7	160	96	80	4	3945
	8	320	192	160	1*	7900

Two luminances may be defined for one colour.
Four text lines are provided at the foot of each split screen.

Table 1 — Graphics options for Atari 800.

Identical

The board can be programmed to provide a 159(X) × 95(Y) array of pixels. In that mode the area of the screen covered by graphics is identical to that in the 319 × 191 mode, so that four lines of scrolling text can also be obtained if the graphics and 380-Z VDU boards are coupled.

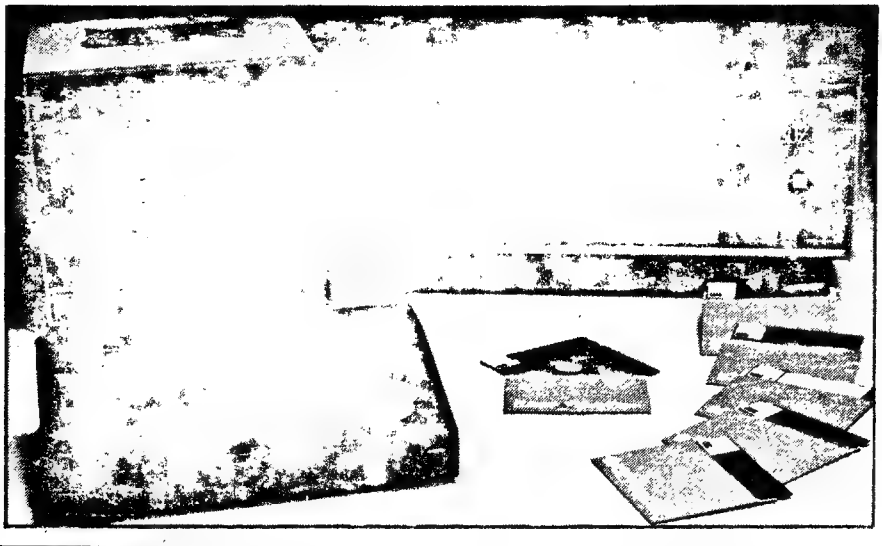
In the 159 × 95 mode, four bits are used to define each point, resulting in 16 choices of output, usually grey levels or colours. In addition, the graphics board stores two separate 159 × 95 mode pictures, so that one can be displayed while another is being generated.

The use of a look-up table, as well as allowing the user to select four or 16 grey scales or colours from a very wide range of choices, also results in features which may at times be useful. For example, in the 319 × 191 mode one can arrange for the two bits per pixel to correspond to two separate pictures. Either of the two pictures can be displayed, or both superimposed, or one masked by the other or one masked by the inverse of the other.

In the 159 × 95 mode, each of the two stored pictures can be arranged to be one picture with four bits per pixel, two pictures with two bits per pixel, or four pictures with one bit per pixel, with

(continued on next page)

380-Z.



(continued from previous page)

superimpositions or maskings as described for the 319 × 191 mode.

The look-up table features may be of use to scientists and mathematicians in allowing them two separate superimposable 319 × 191 pictures, or to producers of computer-aided learning material who want selective masking, or changing foregrounds with constant backgrounds.

The Texas Instruments TI 99/4 was conceived originally as a home computer but with its integral sound generator and speech synthesiser. Americans are using it in schools extensively. Its colour graphics capability is impressive — up to 192 × 256 points in 16 colours. You can also program your own characters or graphics shape, called sprites, and even give them priorities so that they know which sprite is on "top" when two overlap.

Unfortunately, it does not work on the European colour standard and requires an expensive American-standard TV set. TI will launch a PAL version this summer but is keeping a low profile at the moment.

Recent arrival

The Acorn Atom has reached the market only recently and its colour graphics plug-in module has not really been used widely enough to generate much comment. It has a resolution of 256 × 192 in black and white and generates three-colour graphics up to 128 × 192 pixels.

The Atari 800 has been on the market in the States for about a year and has been the subject of very favourable comment. At the moment it suffers from the same problem as the TI 99/4 but PAL versions are beginning to make their way to Europe and should be available generally shortly.

The Atari 800 and its cheaper brother, the 400, typify the way that colour graphics is progressing. Many American and Japanese companies are now supplying colour as standard, and at a reasonable price.

Apple.

The Atari 800 costs around £800 but sells for $800 in the States. It should be available here for around £500 if enough are sold. The Atari 800 has a resolution of 320 × 192 pixels in its highest graphics mode, with up to four colours available.

Like the Atari, the DAI and DC1 are two of the latest machines on the U.K. market, so it is worth looking at them in greater detail. Like the TI 99/4 and Exidy Sorcerer, the Atari 800 has a plug-in ROM-pac which allows the user to configure the machine easily. An 8IL Microsoft Basic pack is standard, although other language options are becoming available.

RAM is in 16K modules which plug into slots in the body of the machine. The operating system of the computer is also in a plug-in ROM, a 10K module. The potential for future upgrade therefore is clear. The video output plugs into a normal colour TV and it has a socket for a video recorder. Other peripherals available are disc drives, printers, an RS232C interface, games paddles and a light pen.

There is an on-screen editing facility and cursor control is possible directly from the keyboard; clear screen and tab controls are also provided. All keys repeat if held down for more than one second.

The screen format can be varied from 12 × 20 to 320 × 192. Table one indicates the options available. In normal text mode, 24 lines of 38 characters is the 'dafault' situation. It is also possible to have line deletion and insertion, as well as the more usual character deletion and insertion.

Enormous potential

The cassette recorder uses two channels for recording. One is digital, the other audio. Consequently a special tape recorder is required. Digital data transfer is at 600 baud. The audio channel allows speech to be interlinked with data, the sound emitting from the TV loudspeaker. The potential for educational use of the facility is enormous.

The Atari 800 has four integral sound generators for musical tones or game sounds. They cover four octaves and have variable volume and tone. There is a small internal loudspeaker.

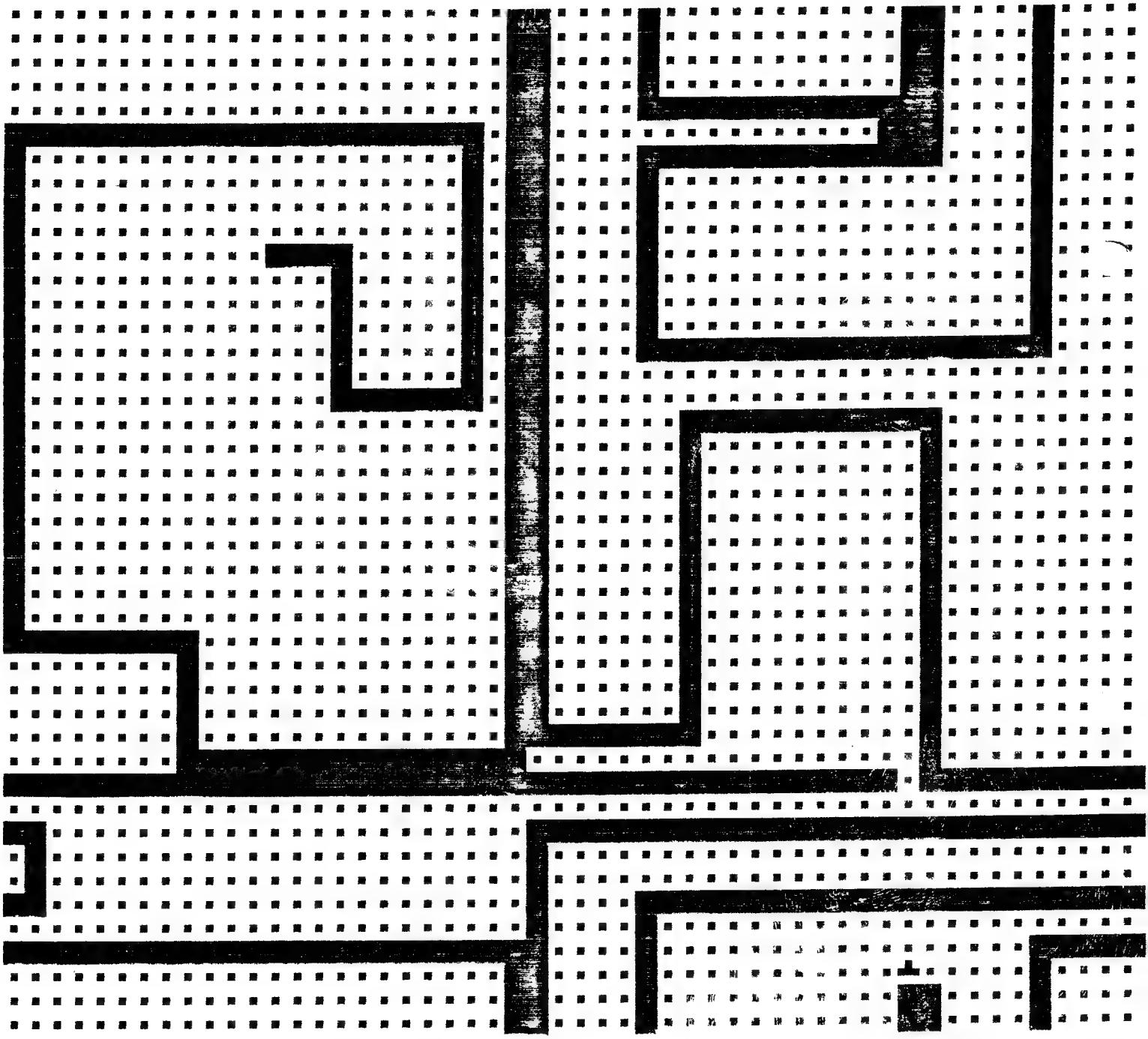
The software available within the Basic for graphics programming is fairly limited. Although it is possible to select from 128 colour/luminance combinations, only five colours can be displayed at one time. The more successful graphics commands include DRAWTO, which draws a line from the previously-plotted location to the

(continued on page 23)

Table 2 — Mode definition table, DAI PC1.

Number	Graphics size	Text size	Type of graphics
0	—	24 × 60	—
1	72,65	—	16 colour
1A	72,65	4 × 60	16 colour
2	72,65	—	4 colour
2A	72,65	4 × 60	4 colour
3	160,130	—	16 colour
3A	160,130	4 × 60	16 colour
4	160,130	—	4 colour
4A	160,130	4 × 60	4 colour
5	336,256	—	16 colour
5A	336,256	4 × 60	16 colour
6	336,256	—	4 colour
6A	336,256	4 × 60	4 colour

LOOK



screen copy met MX-80 printer (EPSON).
Uit het programma "inpakken"(surround) van de heer DRUIJFF.
Het programma loopt in MODE 4.

(continued from page 21)

defined co-ordinates and SETCOLOR, which is used to define one of the five colours available.

Although the DAI personal computer is only now beginning to appear on the U.K. market, it has a long and interesting history. Developed originally as a contender in the TI personal computer stakes, only to be jettisoned in favour of the 99 4, the DAI machine has a real multi-national flavour — it was designed by two British graduates and built in Belgium.

Data Applications is an industrial control company with many years' experience of designing and building systems and that is evident in the personal computer. It has obviously been designed by engineers with a definite application as a home computer.

Easiest to program

The basic computer has a 24K ROM containing the Basic interpreter, machine language utility and general housekeeping routines, and 48K of RAM. There is an optional 9511 math chip which provides hard-wired arithmetical operation.

The DAI personal computer obviously has been designed to provide not only the best colour graphics capability for less than £1,000 but also the easiest programmability of those functions.

Output ports at the rear consist of two analogue paddle inputs; RS232C 25-pin connector; two cassette interfaces; a 34-pin DCE-BUS, an industrial control bus; and, internally, a 50-pin 8080 bus which bears a resemblance to the S-100 bus and a UHF video connector.

The Basic resides in 24K ROM and is one of the most powerful available at this level. With the utility program and colour control options the 24K ROM is well-used. The system is supplied with two manuals and connecting leads for video and tape recorder. The first manual is a 72-page introductory text aimed at the complete novice.

Some could complain that the manual is somewhat trite — it starts by telling you how to plug-in and switch-on — but it is better to under-estimate the potential users' knowledge than vice-versa.

From that simple beginning the user is introduced to the major attribute of the DAI, the colour graphics, which are ridiculously easy to operate. There are 16 colours which, with the colour control on a TV turned down, are a perfect grey scale between black and white. The colours are labelled 0 to 15.

The screen can be defined in 13 ways — from 24 × 60 characters, through a mixture of graphics and text to 336 × 256 dots. Table two shows the configurations available. In each case the background and foreground can be defined precisely, using the COLORT and COLORG commands. Each configuration needs different RAM requirements.

A detailed look at the graphics generator is useful, although most other machines use a similar system. At the start of each line on the TV screen, two bytes are taken from memory which define the mode for that line and may update the colour RAM by two bytes. They are the Control and Colour bytes. The rest of the line is colour or character information and the number of bytes used for it is a characteristic of the particular mode.

Definition

The screen can operate at a number of different definitions horizontally — e.g., dots scan. In the highest-definition graphics mode there are 352 visible dots across the screen. The two lower definitions have respectively one-half and one-quarter of that number. There are about 520 scans visible on a 625-line television set and the screen hardware can draw only, at minimum, two scans per line, due to the inter-lacing. That gives a maximum definition of 260 by 352, which is close to the 3:4 ratio of the screen sides. Thus circles are round.

Characters are fitted on to the grid by using eight columns of dots per character, the dot positions being defined for each character by a ROM. That allows 44 characters per line maximum — 22 or 11 in lower definition modes. A fourth horizontal definition provides for a high-density character mode with 66 characters per line.

Sixteen colours, including white and black, can be displayed by the system. Whenever a four-bit code is used to describe a colour, it selects from that range of possibilities. In some modes — characters and or four-colour graphics — a set of four of the colours, not necessarily distinct, is loaded into a set of colour registers.

Any two-bit code describing a colour selects an entry from the registers. Vertical definition is set by a four-bit field in the control byte. In graphics mode that permits repetition of the information to fill any even number at scans from 2 to 32.

In character mode it defines the number of scans occupied by each line of characters; thus the vertical spacing on the screen can be changed to allow anything between an 8 × 7 — the sensible minimum — and 8 × 16 character matrix, giving between 35 and 15 lines of characters on the screen.

Background colour

Background or letter colour may be changed from the Basic. When operating from within Basic, three drawing facilities can be used. The screen is defined in X and Y co-ordinates, XMAX and YMAX being defined by the mode chosen. A dot may be placed at co-ordinates X, Y by programming DOT X, Y C, where C is the defined colour of the dot.

A line is drawn by defining the beginning and end co-ordinates, e.g., DRAW X1, Y1, X2, Y2 C will draw a line of colour C between X1, Y1 and X2, Y2. FILL X1, Y1 X2, Y2 draw a rectangle with opposite corners X1, Y1 X2, Y2.

Eight other colours are available as well as the 16 mentioned. The others act on the machine code directly and allow the graphics to be changed very simply under program control. The user effectively is implementing complicated machine code instructions from simple Basic statements.

Allied to the graphics capability are the three programmable sound synthesisers. They allow the volume and frequency of each sound channel to be defined from Basic and even provide the option of tremelo and glissando. Volume is also set from the Basic commands. The SOUND command specifies a channel to which it applies, an envelope to be used, the required volume and frequency. A simple sound command would be:

SOUND 0 1 15 0 FREQ (1000)

(continued on next page)

Table 3.

Graphical resolution	Display colour modes	Required picture space	Available program and work space
65 × 88	4 or 16	1.5K	46.0K
130 × 176	4 or 16	5.8K	42.0K
260 × 352	4 or 16	22.8K	25.0K
240 × 528	4 or 16	32K	16.0K non-square

The above are examples of the RAM requirement for possible all-graphics screen configurations. Usage will be affected by the screen driver package used.

(continued from previous page)

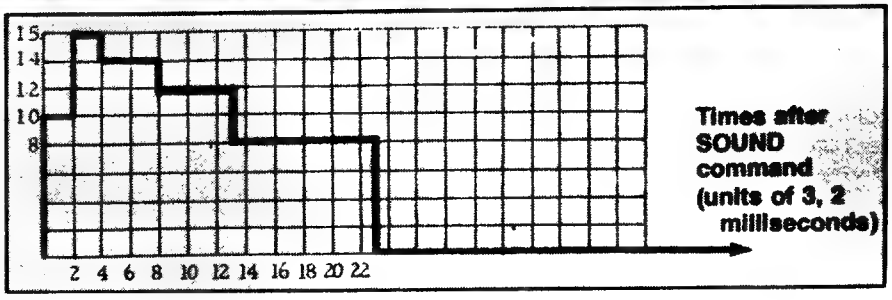
That would set channel 0, using envelope number 1, at a volume of 15 and frequency 1,000Hz. The ENVELOPE statement allows the volume of a note to be changed rapidly, in the same way as a musical instrument. Thus the rise and fall in volume for a note can be specified. The command specifies a set of pairs of volume and time. The volume constants are in the range 0 to 15 and the time is in units of 3.2 milliseconds.

Edit facility

For example, the command ENVELOPE 0 10, 2; 15, 2; 14, 4; 12, 5; 8, 10; 0 sets a volume envelope as shown in the diagram. A white noise generator is also provided.

Within Basic is an interesting edit facility, called EDIT; a very powerful feature, it lists the program for editing. Screen editing, using cursor control, is possible only in EDIT mode. Individual lines or 24-line blocks can be edited. All control characters, e.g., CR, are shown.

Lines are presented in page mode, the width being determined by the length of the line. A window of 24 lines by 60 columns is presented, the window being moved by cursor control keys. The EDIT mode can be used to merge Basic language programs or routines as well.



The system also has a comprehensive Error message capability with clear error types being displayed in plain English — there is no Error No. N problem.

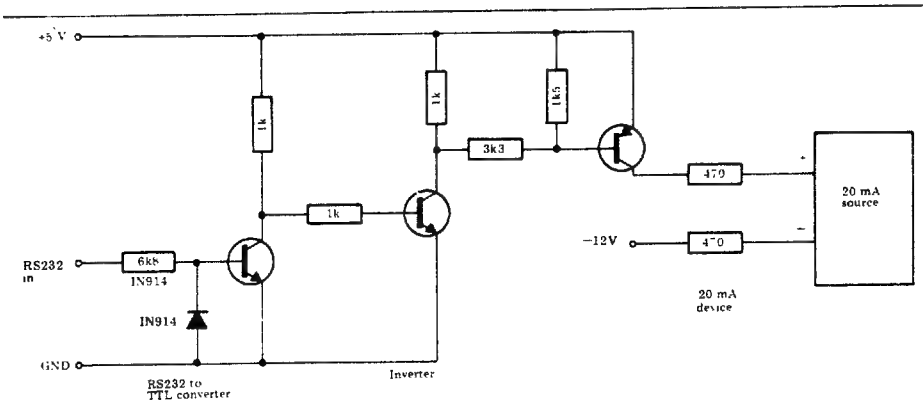
The second instruction manual is probably one of the most comprehensive available but could be better laid-out and in a more logical order. It contains all the information needed — it is somewhat frustrating flipping backwards and forwards all the time. Numerous program examples are given and the demonstration cassette supplied whets the appetite; it is a pity some loading errors were encountered.

It is also possible to plug-in a 9511 maths chip which speeds the execution of trigonometric functions by around a factor of 10.

Think of future

Future products provided by the Americans and Japanese promise to make up our minds for us. Tandy recently has introduced a colour system selling for less than $400. Commodore has a Pet-compatible system on the stocks for even less. Sharp and other Japanese companies are close behind.

With those systems selling in the U.K. for less than £200, colour will become the usual method for displaying information. That should be taken into account when thinking about future purchases of computer equipment.



Low-cost RS232 to 20 mA converter — see 'Printer Interface'.

```
5       REM ROTERENDE ELLIPSEN  J.BEUMERS
6       REM Dit programma roteert een ellips over 15 gr,30 gr....
10      MODE 6:COLORG 0 5 10 15
20      CLEAR 3000
30      DIM A(250.0),B(250.0)
40      FOR X=0.0 TO 2.0*PI STEP PI/125.0
50      A(N)=120.0*COS(X):B(N)=30.0*SIN(X)
60      N=N+1.0:NEXT
70      FOR FI=0.0 TO PI*11.0/12.0 STEP PI/12.0
80      P=SIN(FI):Q=COS(FI)
90      FOR N=0.0 TO 249.0
100     DOT Q*A(N)-P*B(N)+167.0,P*A(N)+Q*B(N)+127.0 10:NEXT
110     NEXT:DOT 167,127 15
120     GOTO 120
```

F A S T G R A F T E X T (The story of FGT)

We vinden het programma FGT uit onze bibliotheek zo belangrijk dat we er graag een speciaal verhaal aan wijden.
Deels om U een idee te geven van de mogelijkheden van dit programma,deels om diegenen die het programma al gebruiken nog wat extra informatie en mogelijkheden aan te bieden.

WAT IS FGT ?

FGT is een programma in machinetaal dat karakters of tekeningen kan maken op het scherm.De routine is bruikbaar voor alle grafische modes.(ook de 16 kleuren modes).
Het programma bevindt zich op de adressen vanaf é300 tot +/-é900.
(Over die +/- straks nog meer).

WELKE VOORDELEN BIEDT FGT ?

Velen hebben al de routine uit het handboek "GRAFTEXT" gebruikt in hun programma's.
FGT is gebouwd op hetzelfde stramien,de mogelijkheden zijn echter nog sterk uitgebreid.
Het fundamentele voordeel is natuurlijk SNELHEID.Deze moet toch wel een paar honderd keer hoger liggen dan die van het BASIC programma.
Voor bepaalde toepassingen is deze hoge snelheid onontbeerlijk.
GEHEUGENGEBRUIK:Met het BASIC programma zit de karakterinformatie dubbel in het geheugen:1 keer in DATA statements,1 keer in de HEAP in ARRAY-vorm.(vandaar ook de vertraging bij de start).
FGT kan in het geheugen blijven zitten,U kan een nieuw programma laden en opnieuw FGT aanroepen.Programma's met FGT zijn dan ook altijd heel wat korter.(lijnnummers 40040-40200 verdwijnen).
FLEXIBILITEIT:Met het bijgeleverde programma "TABLE CREATOR" kan U op een tamelijk eenvoudige manier zelf karakter-en symbool-tabellen creeren.
EXTRA MOGELIJKHEDEN:FILLFLAG,FILLCOLOR,POINTERFLAG,SPACE,ID.....
AANPASSING:programma's die ontwikkeld zijn met de BASICroutine zijn heel eenvoudig om te bouwen naar FGT.De lijnnummers 1000-1040 vervangen de hele BASIC-routine.(zie FGT WORD GAME)

Het gebruik van FGT

U bepaalt een string(A&="KJHGDFTIUKJHDHDFHDHFSJDHG")
Indien nodig pookt U bepaalde parameters weg op de voorziene locaties.(meestal zijn X,Y en CC(current color) voldoende).
U roept de routine aan met CALLMé300,A& en daar gaat FGT....

WELKE PARAMETERS ZIJN VOORZIEN?

X beginpunt op X-AS
y beginpunt op Y-AS
CC current color:kleur waarin getekend moet worden(0...23)
DF dimensionflag:vergrotingsfactor 0...15
als DF groter dan 0 worden de karakters getekend met dubbele lijnen.Voor bepaalde toepassingen kan dit door een pook vermeden worden.

VF vertikaal flag: 0= horizontaal
1= vertikaal

ZF zinflag 0=van links naar rechts
1=van rechts naar links

PF positionflag 0=de nieuwe X en Y worden gebruikt
1=de string wordt getekend na de vorige string (zoals PRINT A&;....)

SP spatie tussen de karakters of symbolen
ID index vertikaal
A& bevat het/de karakter(s).
belangrijk:A& mag ook een geindexeerde variabele zijn uit een ARRAY ! bv B&(3,4)

FF fillflag indien=1 wordt de achtergrond van het karakter vooraf gewist met de kleur, bepaald in FC.
De afmetingen van de FILL worden bepaald door SP en ID.

FC de fillcolor.

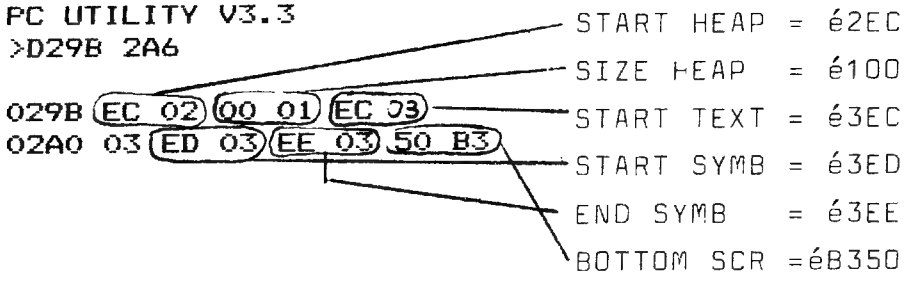
De routine bevat een AUTOINDEX:als bij het tekenen van een string de rand bereikt wordt, dan wordt automatisch een volgende lijn genomen.

BASIC & MACHINETAAL IN DAI BASIC

Bij gebruik van machinetaal en basic moeten we steeds opletten dat de routines mekaar niet gaan storen. De boosdoener bij deze mogelijke conflicten is meestal BASIC, die zonder maatregelen geheugenplaatsen van machinetaal zou kunnen afsnoepen.
Daarom maken we duidelijke afspraken ivm geheugengebruik.
In DAI-BASIC zijn daarvoor een aantal POINTERS terbeschikking (geheugenwijzers zou een goede vertaling kunnen zijn).
Deze pointers zijn:

START OF HEAP	29B 29C
SIZE OF HEAP	29D 29E
START OF TEXT	29F 2A0
START OF SYMBOL	2A1 2A2
END OF SYMBOL	2A3 2A4
BOTTOM SCREEN	2A5 2A6

Bekijken we deze POINTERS bij een "blanco" 48K machine:



Voor andere geheugenconfiguraties zal U dezelfde informatie krijgen behalve BOTTOM SCREEN.
Bij POWER ONN en bij HARD RESET worden deze pointers telkens opnieuw geinitieerd met deze waarden,de informatie komt uit de BASIC ROMS.
Na HARD RESET is uw BASIC programma blijkbaar niet meer aanwezig in het geheugen.Het is er echter nog wel,maar de geheugenwijzers vinden uw programma niet meer terug.
Het hoofdstukje in het handboek "WHAT TO DO IF AN ACCIDENTAL..." p.124 is dus heel eenvoudig te omschrijven:
Bekijk in UT de inhoud van de POINTERS ,noteer dit en zet de informatie daar terug na RESET.

Bekijken we de situatie na invoer van een kort BASIC programma:

```
10    A=5
20    B=A+A
30    PRINT B
40    END
```

```
PC UTILITY V3.3
>D29B 2A6

029B EC 02 00 01 EC 03
02A0 03 10 04 1F 04 50 B3
```

START HEAP =é2EC(ongewijzigd)
SIZE HEAP =é100(ongewijzigd,
er is geen CLEAR geweest)
START TEXT =é3EC(ongewijzigd)
START SYMB =é410
END SYMB =é41F
BOTTOM SCRN=éB350(ongewijzigd)

Onze 4 programmalijnen bevinden zich dus van é3EC tot é410
In de SYMBOL TABLE zijn 2 variabelen terecht gekomen A en B,
deze vinden we terug op de adressen é410-é41F.
Na HARD RESET bevatten de POINTERS terug de informatie van p.3,
indien U bovenstaande informatie invoert is uw BASIC programma
teruggevonden.

We zoeken het BASIC programma en de SYMBOL TABLE op in het geheugen:

D3EC 40F (de TEXT BUFFER)

```
03EC 0A 00 0A A5          lijn 10 |  Lijn 20
03F0 40 02 14 00 00 00 05 | 0A 00 14 A5 40 09 A0 40 02
0400 40 02 | 08 00 1E AD 01 10 40 09 FF | 03 00 28 84 00
>          lijn 30                        lijn 40
```

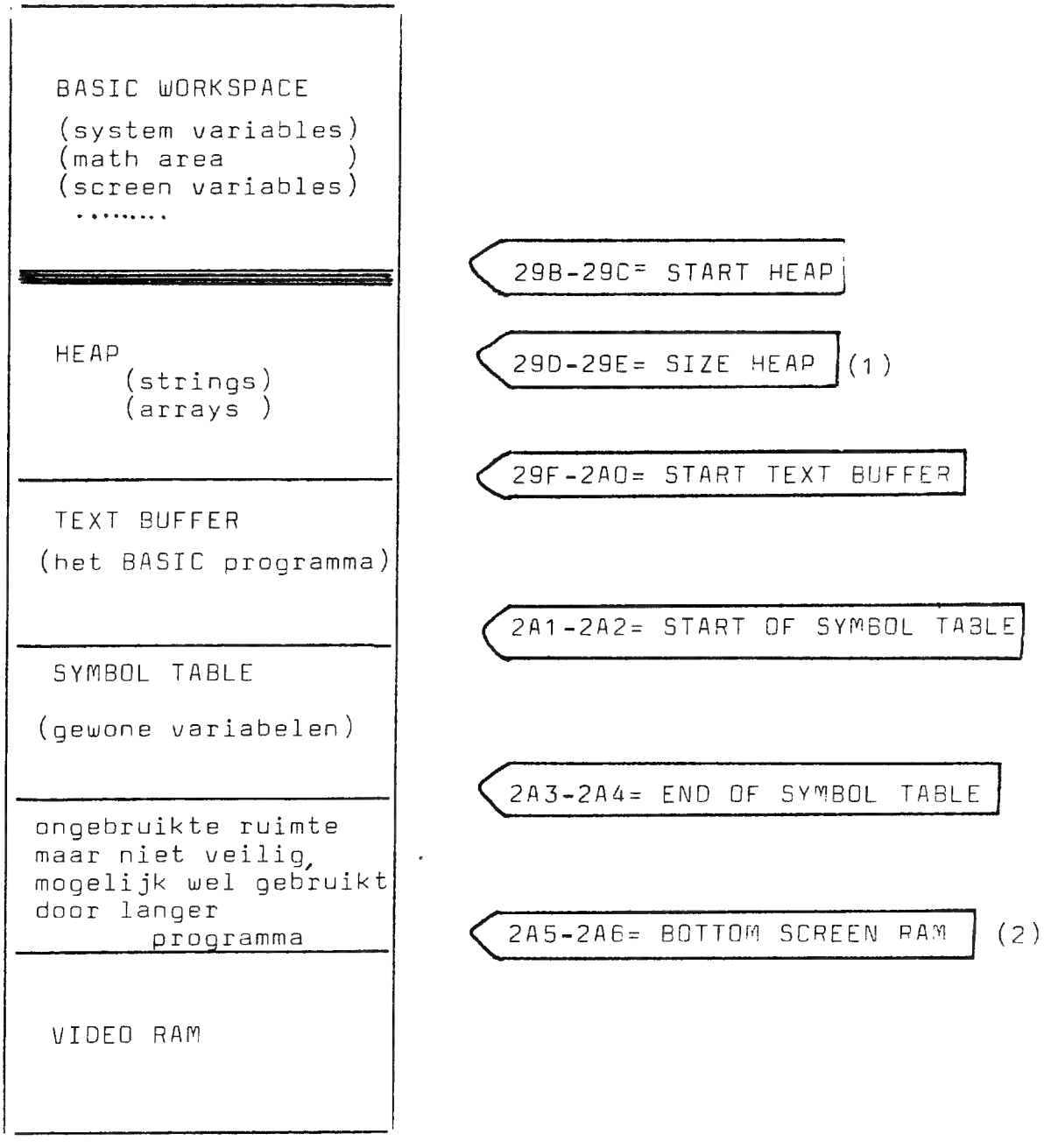
D410 41F (de SYMBOL TABLE)

```
0410 11 41 14 00 00 00 00 11 42 14 00 00 00 00 00 00
       (A)                    (B)
```

na RUN: 00 00 00 05 00 00 00 0A

Nu brengen we de POINTERS in verband met een vereenvoudigde memorymap en dan wordt het wel duidelijk waar we FGT en andere machinetaal programma's veilig kunnen laten opereren.

R A M M A P



(1) Als deze ruimte te klein is kan de boodschap "OUT OF STRING SPACE" verschijnen,een grotere"CLEAR" zal meer ruimte creeren voor strings en arrays,de volgende POINTERS worden namelijk opgeschoven.

(2) Als het BASICprogramma tegen deze POINTER aanloopt verschijnt de boodschap "OUT OF MEMORY"

Waarschijnlijk is het al duidelijk dat alle geheugenlocaties vanaf START HEAP tot einde RAM door BASIC kunnen geclaimed worden voor strings,arrays,BASIC programma,SYMBOL TABLE of VIDEO RAM.(meer dan 22K in MODES 5&6!).
De aangewezen methode om veilige ruimte te reserveren voor ons machinetaal programma is dan ook:

-de pointer "START HEAP" veranderen,zodat deze wijst naar een hogere RAM-locatie.
-de overige pointers aanpassen.

Op p.3 van dit verhaal kunnen we vaststellen dat "START HEAP" normaal wijst naar adres é2EC. é2 is de high byte (inhoud van é29C, éEC is de low byte (inhoud van é29B).
De "START HEAP" pointer kunnen we wijzigen door POKE of door SUBSTITUE in UTILITY.
vb.: POKE é29B,é00:POKE é29C,é09
De "START HEAP" pointer wijst nu naar é900,zodat de ruimte é300-é900 gereserveerd is voor machinetaal.
De overige POINTERS moeten echter eerst aangepast worden door CLEAR XXXX of NEW !!
Nu kunnen we ongestoord een machinetaal programma plaatsen op de adressen tussen é300 en é900.(inlezen in UT met R of intikken met SUBSTITUTE of laten wegzetten door BASIC met POKE,de instructies kunnen vb ingelezen worden van DATA lijnen).
Op de tekening van p.4 is deze ruimte voorgesteld door de gearceerde ruimte.
We hadden al verteld dat FGT zich bevindt op de adressen é300-é900.We kunnen ons de moeite van het aanpassen van de POINTERS besparen door deze POINTERS mee te SAVEN, samen met het machinetaalprogramma.Bij het inlezen is de organisatie van de RAM al klaar voor gebruik.
We SAVEN vb in UT : W29B 900 FGT+POINTERS
Het is zelfs mogelijk om een BASIC programma mee te SAVEN in UT.We moeten SAVEN vanaf é29B tot de inhoud van de pointer op é2A3-é2A4.(end symbol table) .FGT is op tape gezet volgens deze methode:de pointers+FGT+BASIC programma.(lijnnummers 1000-1040 van FGT WORD GAME).Na READ in UT is alles klaar om een nieuw BASIC programma met FGT te creeren.

Het is ook mogelijk om FGT te laden en te saven vanuit BASIC. Dit kan erg prettig zijn voor programma's die definitief klaar zijn vb spelletjes,educatieve programma's.... .
Het inlezen kan dan gebeuren met LOAD, het BASIC programma zorgt dan verder er wel voor dat FGT op zijn juiste plaats terecht komt.Zoals U op onderstaande listing kan merken gebeurt dit door FGT in een paar arrays te stoppen en te SAVEN met SAVEA.
Bij het inlezen gebeurt dan tweemaal de volgende truuc:
De array wordt ingelezen en in de HEAP geplaatst,dan wordt de HEAP POINTER gewijzigd zodanig dat het ingelezen stuk machine-taal onder de HEAP zit.Op lijnnummers 50007-50010 kijken we door middel van een CHECKSUM na of FGT al aanwezig is,dit om te vermijden dat we bij iedere RUN opnieuw de arrays gaan laden.

```
50000
50005 REM TEST PROGRAM ALREADY IN MEMORY
50007 FOR I%=#300 TO #526:CHS%=CHS%+PEEK(I%):NEXT
50010 IF CHS%=62180 OR CHS%=61612 THEN PRINT :PRINT "FGT PROGRAM ALREADY IN MEMORY, END INIT":GOTO 50080
50020 POKE #29B,#EC:POKE #29C,2:CLEAR 2000:REM RESET HEAPPOINTERS
50030 PRINT :PRINT "SET READY ARRAYS WITH FGT PROGRAM, TYPE SPACE"
50040 IF GETC<>#20 GOTO 50040
50045 PRINT :PRINT "LOADING ARRAY WITH PROGRAM"
50050 DIM A%(163):LOADA A% "P/PFGT V2.0":REM FIST BYTE ON #2F0
50060 POKE #29B,#7C:POKE #29C,5:CLEAR 2000:REM SET HEAP AFTER FGT PROGRAM
50065 PRINT :PRINT "LOADING ARRAY WITH TABLE"
50070 DIM A%((#900-#580)/4+1):LOADA A% "P/TFGT V1.1"
50075 PRINT :PRINT "INIT READY"
50080 POKE #29B,#900 MOD #FF:POKE #29C,#900/#FF:CLEAR 2000
50085 REM FOR ADR=#376 TO #378:POKE ADR,0:NEXT:REM CANCEL ERROR REPORT
50090 PRINT :PRINT :GOTO 4
50200 REM ARRAY COPY
50205 PRINT :PRINT :PRINT "ARRAY COPY"
50210 REM 1) FGT PROGRAM (#2F0  #526)
50220 DM%=(#526-#2F0)/4+1:ADR%=#2F0:A$="P/PFGT V2.0"
50230 GOSUB 50300
50240 REM 2) FGT TABLE (#580  #8FF)
50250 DM%=(#900-#580)/4+1:ADR%=#580:A$="P/TFGT V1.1"
50260 PRINT :PRINT :GOTO 50300
50300 DIM A%(DM%)
50310 FOR I%=0 TO DM%
50330 A%(I%)=(PEEK(ADR%) SHL 24) IOR (PEEK(ADR%+1) SHL 16) IOR (PEEK(ADR%+2) SHL 8) IOR PEEK(ADR%+3)
50340 ADR%=ADR%+4:NEXT
50350 PRINT :PRINT "SAVE ARRAY ";A$
50360 PRINT :PRINT "SET RECORD, START TAPE, TYPE SPACE"
50370 IF GETC<>#20 GOTO 50370
50380 SAVEA A% A$
50390 RETURN
50500 REM POKE VARIABELS AND CALL MLP
50510 C%=FC%*#40+CC%:F%=FF%*#80+PF%*#40+ZF%*#20+VF%*#10+DF%:POKE #2F0,C%:POKE #2F1,F%
50520 POKE #2F2,X% MOD 256:POKE #2F3,X%/256:POKE #2F4,Y%
50530 POKE #2F5,SP%:POKE #2F6,ID%
50540 CALLM #300,A$:RETURN
```

Omdat deze techniek van machinetaal laden met arrays ook bruikbaar is voor andere routines gaan we even door met de bespreking van dit programma:

50020 RESET HEAP POINTER OP é2EC(standard)

50030-50050 LOAD ARRAY MET EERSTE DEEL MACHINETAALPROG.

50060 ZET HEAP POINTER OP é57C(boven het vorige deel ml)

50065-50075 LOAD TWEEDE DEEL MET ARRAY

50080 ZET HEAP POINTER ANDERMAAL HOGER
TELKENS CLEAR XXXX OM ANDERE POINTERS AAN TE PASSEN!

50085 POKES OM EEN DEEL VAN FGT TE WISSEN(ERROR REPORT)

50090 TERUG NAAR PROGRAMMA MET "GOTO ..."
"GOSUB" is niet bruikbaar omdat er na verplaatsing van de HEAP POINTERS (en dus na verschuiving van het programma) een verkeerd terugkeeradres op STACK zou zitten.

50200-50390 Tweemaal het ml programma in ARRAY onderbrengen en SAVEN als ARRAY.

50500-50540 Nogmaals een paar lijnen met initiatie en aanroep van FGT.C en F zijn samengestelde vlaggen waarin over de verschillende bits FC,CC FF,PF,ZF,VF,DF ondergebracht worden. In het FGTprogramma worden deze bits afzonderlijk geëvalueerd.

Het SAVEN en LADEN van FGT in ARRAYS is erg handig voor definitieve programma's.Wanneer de tabel van FGT echter uitgebreid wordt moet er opnieuw gerekend worden want dan moet de dimensie van de ARRAYS aangepast worden.Daarom verkiezen we toch de methode van WRITE en READ is UT,de HEAP POINTERS inclusief.

Het FGT-pakket bestaat uit 4 programma's:

1. FGT machinetaal+pointers+BASIC subroutine

(Na R in UT is alles klaar om een nieuw programma te creeren)

2. NEW DEMO FGT
3. FGT TABLE CREATOR (zie volgende bladzijden)
4. FGT WORD GAME (zie listing)

FGT TABLE CREATOR

Table Creator is een programma dat de mogelijkheid geeft om de tabel van FGT eenvoudig uit te breiden met nieuwe characters of tekeningen.
Dit gebeurt in een matrix van 16 X 16,U hoeft evenwel niet de volledige matrix te gebruiken. Indien U vb een matrix van 8 X 8 prefereert gebruikt U bij het invoeren alleencoordinaten tot en met 7. 0,0 is een element van de matrix!

Hoe ziet de tabel van FGT er uit?

De tabel bestaat uit aaneengesloten drawinformaties voor characters.
U vindt telkens terug: 1) een byte met de ASCII waarde
2) een byte met het aantal DRAWopdrachten

voorbeeld:codering van het character "A"(eerste in de tabel)

0580 41 05 11 15 51 55 13 53 15 37 37 55 42 0A 11 17

op hex 580 zit de ASCII waarde van "A":hex 41
hex 581 Ø5:er volgen 5 dubbelbyte draw informaties
hex 582-58B : de DRAWinformaties
vb: hex 582 hex 583
11 15 draw in de matrix van 1,1 naar 1,5

op hex 58B volgt de ASCII waarde van "B":hex 42

de tabel wordt afgesloten met hex 5A,hex FF

Om de tabel uit te breiden:
laad eerst het FGT demoprogramma,laad daarna FGT TABLE CREATOR.
Na RUN vraagt het programma:
"DECIMAAL ADRES EINDE HUIDIGE TABEL (2118=STANDAARD)"?
Op 2118 zit het einde van de tabel zoals ze in het FGTdemo programma geladen wordt,Indien U ondertussen de tabel al heeft uitgebreid moet U natuurlijk het huidige einde gebruiken.
Is er nog geen uitbreiding gebeurd dan voeren we in 2118.
De werkmatrix wordt nu gecontrueerd (MODE 4A)(6A)
Er wordt nogmaals gevraagd:STARTADRES ,hier wordt nu wel telkens het actuele tabeleinde toegelicht,indien alles goed gaat voert U telkens het gegeven adres in.

Na invoer van het startadres wordt het character gevraagd.
De standaardtabel bevat de Hoofdletters+leestekens,zodat het voor de hand ligt om verder te gaan met de kleine letters.
U kan vb invoeren "a",en dan de kleine letter "a" construeren,
U kan "a" evenwel ook gebruiken als tekening en een willekeurige tekening construeren.Die tekening kan U dan later gebruiken met CALLMhex300,A& (vooraf:A&="a").

U krijgt nu een menu met:
E=ENTER K=KILL S=SHOW D=DUMP T=TOTAL

T geeft U in grafische mode een overzicht van de huidige inhoud van de tabel.Als U een nieuw character geconstrueerd heeft vraagt u eerst DUMP!
D DUMP zet de informatie van de nieuwe tekening in de tabel en sluit de tabel af.T toont U daarna of de nieuwe tekening in orde is.
K KILL zet de adrespointer terug op het vorige einde en laat U opnieuw beginnen aan de constructie.
S SHOW TABLE:geeft een overzicht van de ingevoerde drawopdrachten.
E ENTER:om telkens nieuwe drawopdrachten in te voeren.
X1,Y2,X2,Y2 ? voer de gegevens in,gescheiden door comma's.
15 is de maximum coordinaat.
Om een punt in te voeren neemt U X1=X2 en Y1=Y2.
Schuine lijnen die niet onder een hoek van 45° lopen worden niet opgenomen in de tabel,omdat dit geen goede resultaten geeft metgrotere schaalfactoren. U kan ze natuurlijk wel invoeren punt per punt.
Nadat U de coordinaten heeft ingevoerd wordt de matrix gevuld met de overeenkomstige punten en verschijnt de vraag:
DEZE WAARDEN INVOEREN ?J/N
J:de drawopdracht wordt opgenomen in een werktabel.
N:de lijn wordt gewist,de opdracht wordt niet opgenomen.
Mogelijk worden hierbij punten gewist van vorige drawopdrachten, de gegevens zitten toch juist in de tabel.
Als de constructie klaar is:D (DUMP) en T(TOTAL).
Als de printer aanstaat krijgt U ook een grafische printout van het nieuwe character(met EPSONgrafieken) +het adres+ASCIIwaarde, U hoeft dan niets te noteren.

FGT TABLE CREATOR 3

FGT opent knappe perspectieven:

-invoeren van de tekeningen van het schaakspel,het grafisch gedeelte wordt dan een eenvoudige klus,en vraagt slecht +/- 1.5K geheugenruimte,door de snelhied krijgt het programma dan ook professionele allures.(Als de schaakstrategie dan op peil staat)

-compatibiliteit met andere micro's:Toestellen zoals PET,OSI CHALLENGER,SHARP MZ-80,TRITON,TRS-80,VIDEO GENIE gebruiken grafische characters om grafieken te simuleren.
Construeer met FGT TABLE CREATOR de betreffende CHARACTERSET (de meeste zijn in een matrix van 8X8),bestudeer de memorymap van de videoram en U kan het programma zo omzetten.

-U hoeft zich niet te beperken tot de 16X16 matrix:
gebruik bvb 4 tekeningen die hor.en vert. aansluiten en samen een tekening vormen:U kan dan een matrix van 32X32 gebruiken!!!
Dit is een ideaal middel om in DAIgrafieken constructies te gebruiken die anders toch nogal wat reken-en tekenwerk vragen, zoals cirkels,ruiten,ventjes,auto's enz....

-U hoeft zich ook niet te beperken tot eenkleurige symbolen:
maak complementaire matrix in andere kleuren,zet de FILLFLAG OFF, en construeer de volgende matrix op dezelfde plaats!
Houdt in 16kleurenmodes wel rekening met de mogelijke kleuren-conflicten!
-Neem eens contact met de redactie voor je aan een nieuwe tabel begint,we hoeven niet allemaal hetzelfde werk over te doen.
Wie een nieuwe tabel instuurt krijgt de bestaande tabellen in ruil...

succes...

```
      REM fgt word game ***

10        REM word game met fgt
20        REM het woord wordt ingelezen uit data
30        REM de letters worden door elkaar gegooid
40        REM dit nieuwe woord wordt gedisplayd
50        REM de speler tracht het originele woord
60        REM in te typen

100       REM opgave
108       DIM A$(255)
109       READ A$:IF A$<>"STOP" THEN A$(QQ)=A$:QQ=QQ+1:GOTO 109
110       R=RND(QQ):A$=A$(R)
112       CNT!=0      (parameters die maar 1 keer bepaald moeten worden)
113       CC=12:DF=0:VF=0:ZF=0:PF=1:SP=8:ID=10
114       COLORG 8 0 3 12:MODE 4:FILL 0,YMAX/2 XMAX,YMAX 0
115       X=20:Y=YMAX-50:GOSUB 1000
116       L=LEN(A$):DIM B$(25)
117       FOR T=0 TO L-1
118   1     R=RND(25)     de letters willekeurig verspreiden over B&(X)
120   1     IF B$(R)<>"" THEN 118:B$(R)=MID$(A$,T,1)
124   1     NEXT:C$="":FOR X=0 TO 25
125   1     IF B$(X)<>"" THEN C$=C$+B$(X)
127   1     NEXT (nieuw woord samenstellen)
128       CALLM #300,C$ : aanroep van FGT
   plaats hier GOSUB 40040 indien U onze routine uit het handboek
200       REM input  (A&=C&)                                  gebruikt.

202       REM initiate input
203       CC=0:X=20:Y=40:GOSUB 1000
210       FOR X=0 TO L-1
215   1     D$=MID$(A$,X,1)
220   1     IF EF=1 THEN EF=0:SD$="OUT OF TIME..":GOTO 340
225   1     G=GETC:IF G=0 THEN CNT!=CNT!+0.1:GOSUB 2000:GOTO 220
230   1     IF G=ASC(D$) THEN 235
232   1     SOUND 1 0 15 0 FREQ(100):WAIT TIME 10:SOUND OFF
234   1     GOTO 220
```

```
235   1     SOUND 1 0 15 0 FREQ(1000):WAIT TIME 5:SOUND OFF
240   1     CALLM #300,D$ (de letter displayen)
250   1     NEXT
299         WAIT TIME 50

300         REM score
310         SC=L*100-CNT!*10
320         SC$=STR$(SC):SC$=LEFT$(SC$,LEN(SC$)-2)
330         SD$="SCORE ="+SC$
340         CC=12:X=20:Y=YMAX-20:GOSUB 1000
350         CALLM #300,SD$ (score displayen)
360         IF GETC=0 THEN 360
370         GOTO 110
1000    *   C=FC**40+CC:F=FF**80+PF**40+VF**20+DF
1010    *   POKE #2F0,C:POKE #2F1,F
1020    *   POKE #2F2,X MOD 256:POKE #2F3,X/256:POKE #2F4,Y
1030    *   POKE #2F5,SP
1035    *   (FOR X=#2F2 TO #2F4:POKE X+5,PEEK(X):POKE X+8,PEEK(X):NEXT)
1040        RETURN
        *dit vervangt lijnnummers 40040-40200 uit het BASIC-programma

2000        REM time score
2005        CNT=CNT!
2007        IF CNT+5>=XMAX THEN EF=1:RETURN
2010        FILL CNT,10 CNT+5,30 3
2020        RETURN
5000        DATA 1234567890 (een test-woord)
5004        DATA VIOLISTE,FIETSPAD,HANDBOEI,TELEGRAM,RIOOL
5005        DATA SCHAAR,FIETSBEL,MISTLAMP,DIREKTEUR,OVEN
5010        DATA OORLEL,RADIO,VILTSTIFT,HORLOGE,VULPEN,POTLOOD
5015        DATA BLOEMKOOL,ACCU,LUIDSPREKER,CASSETTE,TANG
5020        DATA BATTERIJ,GLOEILAMP,TANDPASTA,SCHOENLEPEL,COWBOY
5025        DATA BRIEF,CARBON,SCHRIJFMACHINE,CACTUS,GEWEI
5030        DATA BUREAU,TELEVISIE,KABELNET,OOGARTS,POEDER
5035        DATA SPORTTAS,VLOERBEDEKKING,FOTOTOESTEL,PROJECTOR
5040        DATA PILOOT,AGENDA,HANDREM,asbak,vegetarier
5045        DATA TOILET,BEDDENSPREI,OLIEBOL,DENNENBOOM
5050        DATA JOJO,ZADELLEER,UIEN,EIEREN,TROMPET
5055        DATA VUURWERK,VOETZOEKER,ROEIBOOT,STRIJKBOUT,POTLOOD
5060        DATA WORTEL,CASSETTE,PRULLENBAK,LAMP,PLATENSPELER
5070        DATA STICKER,SOLDEERBOUT,GITAAR,HAMER,DRUMSTEL
5090        DATA BENZINE,GEHAKTBAL,TENNIS,RACKET,COMPUTER
5095        DATA SCHILDERIJ,CALCULATOR,POSTER,VENTILATOR
5097        DATA ETUI,GORDIJNEN,MAPJE,BUREAU,KLADBLOK,BONGO
5100        DATA SCHEIDSRECHTER,SCHOENSMEER,VETER,ONTBIJTKOEK
5200        DATA STOP (gaat U zelf maar door)
            de woordenschat is ontleend uit het programma"HANGMAN"
```

IN OTHER WORDS

39 What to look for in coming editions,new contest:an article about an application of DAI.(SOFT or HARD).First price:9511 !!
41 Ribbon for TX-80:PELIKAN GR1,Troubles with colour generation? A capacitor across the crystal might help.
42 A crochetwork+screen copy++a programm to print the characterset.
43 CALLM in DAI-BASIC, an introduction
44 Entry points of routine SET MODE
45 Schematics cassette interface
46 FLASH:a machine language routine that flashes COLORG in background operation.(Interrupt 7)
47 PRIME NUMBERS
59 A new challenge:the happy numbers
60 Here they are : the prime numbers
61 From "Educational Computing":COLOUR GRAPHICS
64 Screen copy on MX-80 from the game "SURROUND".We had to learn about TRS-80 Graphics to create this one!
66 A low cost RS232 to 20 mA converter from PCW.
67 FAST GRAF TEXT
68 Parameters of FGT
69 BASIC & machine language in DAI BASIC
70 TEXT BUFFER & SYMBOL TABLE
71 A RAM MAP: A safe place for ml procedures: change the contents of é29B and é29C (The START HEAP POINTER), execute CLEAR,and you have a safe area for ml programms.
73 How to LOAD and SAVE machine language programs through ARRAYS. Don't use GOSUB to CALL such routine:The return address on STACK doesn't macht to RETURN.(Use GOTO)
75 FGT TABLE CREATOR
78 FGT WORD GAME.(Could be used with our routine in the handbook GRAFTEXTSUBDEMO,but might be too slow for the game)

We will supply more translations(as we did this time with articles from NEWSLETTER 2),but we can't avoid the delay.

Some programms of the library that are available in FGT version:

555 DESIGN PROGRAM
MASTERMIND
ALGORITHM OF HORNER
MUSIC ARTIST
CLOCK READING TRAINING
WORD GAME
NEW DEMO

A nice "FILM TITLE SIMULATION"

```
2000 W$="                From The Paramount Picture ''HATARI''  BABY ELEPHANT WALK          "
2002  POKE #75,32
2005  FOR WW=0.0 TO 1.0
2010  FOR W=0.0 TO LEN(W$)-19.0
2020  B$=MID$(W$,W,19)
2030  CURSOR 20,11:PRINT B$
2040  WAIT TIME 10:NEXT:NEXT
2050  B$=MID$(W$,58,19)
2060  CURSOR 20,11:PRINT B$
2080  RETURN
```

```
 0 zwart
 1 blauw
 2 d.rood
 3 rood
 4 paars
 5 groen
 6 d.bruin
 7 l.bruin
 8 grijs
 9 blauw
10 oranje
11 rose
12 l.blauw
13 l.groen
14 geel
15 wit
```

alle adressen in HEXvorm!

```
29B-29C   start heap
29D-29E   size  heap
29F-2A0   start text buffer
2A1-2A2   start symbol table
2A3-2A4   end of symbol table
2A5-2A6   bottom screen ram

75        cursor symbol
74        cursor mode
72-73     cursor position

40,28 cass motor 1 ON
40,18 cass motor 2 ON
40,30 1 and 2 OFF
```

```
131,0   output scrn+
        RS232
131,1   screen only
131,2   edit buffer
135,2   read from
        edit buffer
```

MODE	XMAX	YMAX
1/2	71	64
3/4	159	129
5/6	335	255

```
COLORG R1 R2 R3 R4
       20 21 22 23
16 :R2*R1 R4*R3
17 :R1*R2 R3*R4        32K 7XXX
18 :R3*R1 R4*R2        12K 2XXX
19 :R1*R3 R2*R4         8K 1XXX
```

```
MERGE
°CLEAR XXX
°LOAD"A"
°EDIT BREAK/BREAK
°LOAD"B"
°POKE 135,2
```

```
IMP INT *** IMP FPT
°IMP FPT
°CLEAR XXXX
°EDIT BREAK/BREAK
°IMP INT
°POKE 135,2
```

LIJN	CTRL	COLOR	LIJN	CTRL	COLOR
23	BFEF	BFEE	11	B9A7	B9A6
22	BF69	BF68	10	B921	B920
21	BEE3	BEE2	9	B89B	B89A
20	BE5D	BE5C	8	B815	B814
19	BDD7	BDD6	7	B78F	B78E
18	BD51	BD50	6	B709	B708
17	BCCB	BCCA	5	B683	B682
16	BC45	BC44	4	B5FD	B5FC
15	BBBF	BBBE	3	B577	B576
14	BB39	BB38	2	B4F1	B4F0
13	BAB3	BAB2	1	B46B	B46A
12	BA2D	BA2C	0	B3E5	B3E4

CTRL&COLOR BYTES IN A-MODE

MODE	CTRL	COLOR	LIJN
1A/2A	BAE7	BAE6	3
	BA61	BA60	2
	B9DB	B9DA	1
	B955	B954	0
3A/4A	ACD3	ACD2	3
	AC4D	AC4C	2
	ABC7	ABC6	1
	AB41	AB40	0
5A/6A	7557	7556	3
	74D1	74D0	2
	744B	744A	1
	73C5	73C4	0

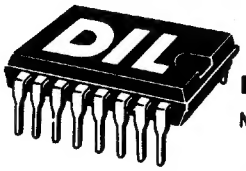
```
FD00 b2 page signal
     b3 serial out rdy
     b4 right paddle
     b5 left  paddle
     b6 random data
     b7 cass. input
FD01    Trigger paddle
FD04 0-3 volume ch.1(0)
     4-7 volume ch.2(1)
FD05 0-3 volume ch.3(2)
     4-7 volume noise
FD06 b0 cass.out
     b1/2 paddle select
     b3 paddle enable
     b4 cass motor 1
     b5 cass motor 2
     b6/7 ROM BANK SWITCH
```

```
FF00 ser.inp.buff
FF01 b0-6 keyb.inp.
        b7 in7 DCE
FF02 Interr.reg.
FF03 b1 frame error
        b2 overrun error
        b3 rec.buf.loaded
        b4 trans.buf.empty
FF04 COMMAND REGISTER
FF05 BAUD RATE REGISTER
FF06 ser.out buf.
FF07 keyb.output
FF08 interr.mask reg.
```

```
TEST EVENT
PEEK(éFD00) IAND 32
PEEK(éFD00) IAND 16
PEEK(éFD00) IAND 48
```

```
FF09 TIMER 0
FF0A TIMER 1
FF0B TIMER 2
FF0C TIMER 3
FF0D TIMER 4
8253
CH 0 FC00/FC01
CH 1 FC02/FC03
CH 2 FC04/FC05
STATUS FC06/FC07
```

DAI-DEALERS *** COMPUTERVERENIGINGEN



D.I.L.-ELEKTRONIKA
Mijnsherenlaan 108, 3081 CH Rotterdam

ALLE DOE-HET-ZELF ELEKTRONIKA - TECHN. TIJDSCHRIFTEN EN -BOEKEN

LEGOTRONICS
Middenstraat 8
8800 ROESELARE BELGIE
tel. 051/207878

ORDIMAX
Rue de la Bonnefemme 11
4030 GRIVEGNEE BELGIE

MULTISOFT
Rue Bargue 25
75015 PARIS FRANCE
7838837

TELEC
Steenstraat 40
9711 GP GRONINGEN NEDERLAND

MSB R.NEDELA
MARTKSTRASSE 3
POSTFACH 1420
D7778 MARKDORF GERMANY

COMPAC
Plaats 25
2513 AD DEN HAAG NEDERLAND

HCC NEDERLAND hobby computer club
Prinsenhof 11
2641 RN PIJNACKER
NEDERLAND

DAI BRUSSEL
Raketstraat 60
1130 BRUSSEL BELGIE
02/2166010

HCC BELGIE
Borkelstraat 51
2120 SCHOTEN BELGIE
031/589674

DAI NEDERLAND
Van Vollenhovenstraat 15A
3016 BE ROTTERDAM NEDERLAND
010/361288

Stichting BASICned
Tolakkerweg 81
3739 JJ HOLL.RADING NEDERLAND

DIDACOM computers&onderwijs
p/a I.BROEKMAN AVENBEECK 98
2182 RZ HILLEGOM
2520/18032 NEDERLAND