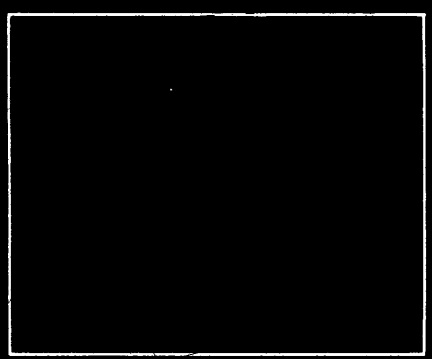
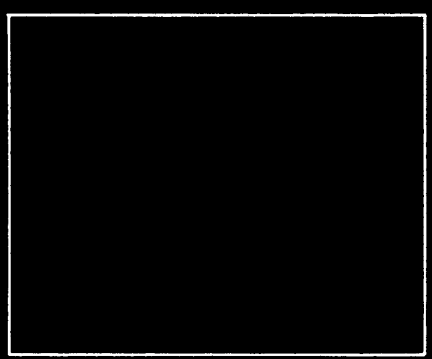
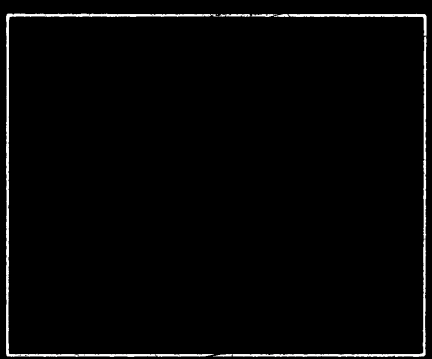
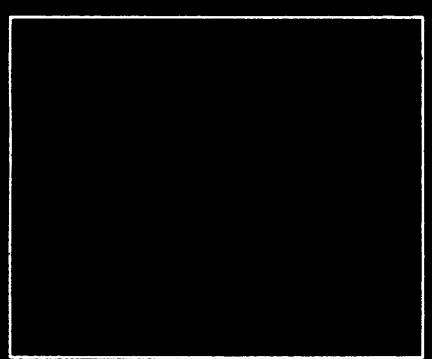
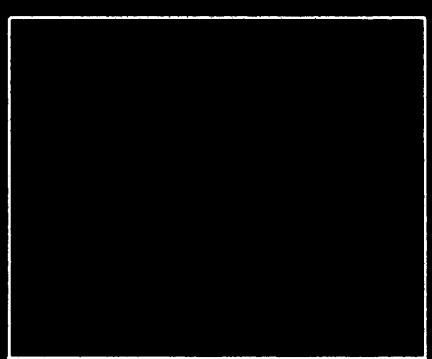
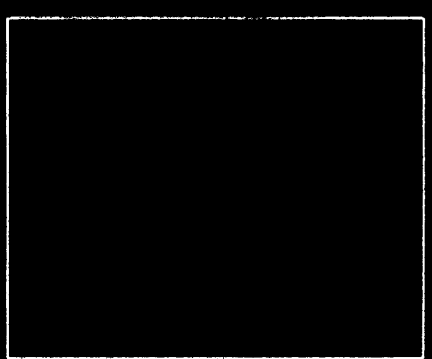
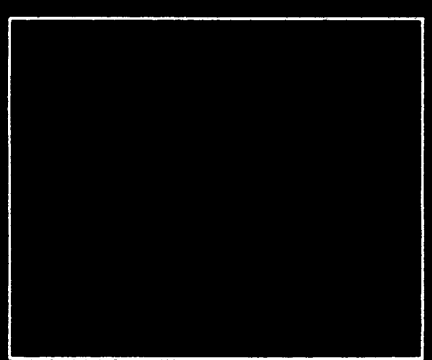
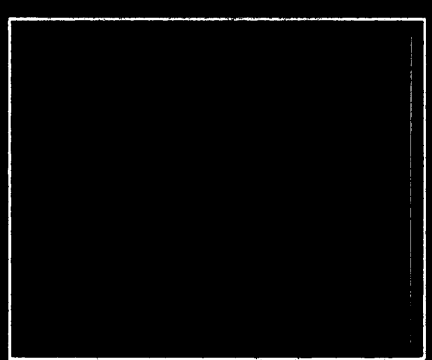
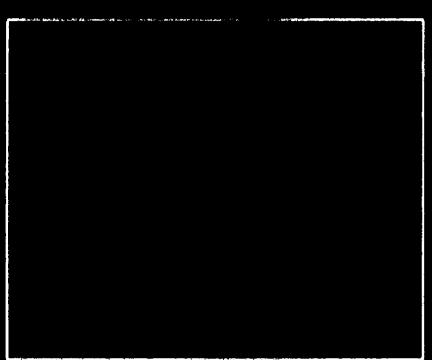
DAI NAMIC

tweemaandelijks tijdschrift mei - juni 1983

een uitgave van dainamic v.z.w.
verantw. uitgever w. hermans, heide 4 - 3171 westmeerbeek

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
Freddy De Raedt
Wilfried Hermans
René Rens
Jos Schepens
Roger Theeuws
Bruno Van Rompaey
Jef Verwimp

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
Heide 4
B 3171 Westmeerbeek
België
tel. : 016/69.86.23

Kredietbank Westmeerbeek
nr. 406-3016141-33
BTW : 420.840.834

Lidgelden

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

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

DAINAMIC

PERSONAL COMPUTER USERS CLUB

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

belangrijke ASCII-waarden in DAIpc

functie/symbool	HEX	DEC
back-space	8	8
TAB	9	9
linefeed	A	10
clear screen	C	12
CURSOR UP	10	16
CURSOR DOWN	11	17
CURSOR LEFT	12	18
CURSOR RIGHT	13	19
space-bar	20	32
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	ENQ	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

NEWSLETTER 16

Westmeerbeek, juni 1983

Beste leden,

Met Newletter 16 bieden we U weer een bonte verzameling programma's, artikels en tips. Aan de namen van de auteurs kan je merken dat er bijdragen zijn uit heel Europa, de vertaaldiensten zorgen er verder wel voor dat dit alles voor de diverse nationaliteiten te begrijpen is. Voor de Nederlandse leden wordt bestellen en betalen een stuk eenvoudiger : Just van Dunne is zo vriendelijk geweest voor DAInamic een POSTGIRO nummer in Nederland te openen. Hier kan je voortaan terecht voor alle verrichtingen i.v.m. software,hardware en abonnementen.

J.F. van Dunne
Hoflaan 70
3062 JJ ROTTERDAM

GIROnummer : 4083817 Telefoon : 010/144802

Wij wensen U nog een prettige vakantie, tot binnen 2 maanden.

Dear members,

With Newsletter 16 we offer you a lot of programs, articles and hints for your DAI computer. Many thanks to the authors all over Europe. Special thanks to Bill Read, Colin Hards, Dave Atherton, Cedric Dufour and all those who take care of the translations. BASICODE II is a new version of the HOBBYSCOOP transmission standard, written for DAI by Th. van Lieshout. Maybe you can capture Hilversum radio on the short waves ? In next issue we hope to bring you an extended test-report on KEN-DOS, the system should be ready by that time. Rumours are that INDATA is also preparing new drives , we will let you know.

We wish you sunny holidays, enjoy newsletter 16.
p.s. don't take WOM too serious ...
Wilfried Hermans

NEWSLETTER 16

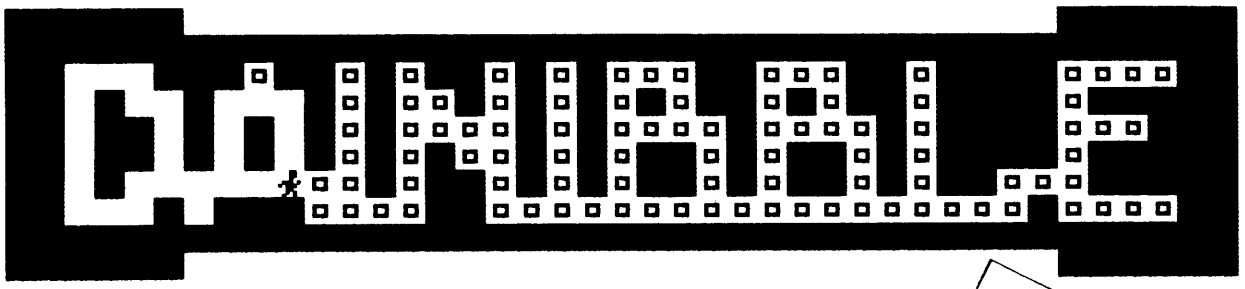
INHOUD — CONTENTS

DAInamic subscription rates :

Benelux	: 900 Bfr
Europe	: 1000 Bfr
Outside Europe (Air Mail)	: 1400 Bfr

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



NEW SOFTWARE

**************** D O Y O U L I K E A C T I O N ?

Five little men are waiting till they can jump into the maze of DAINIBBLE, where they will be running around under your control till they are nibbled by a monster. After his death a man gets a decent funeral and the place is marked with a cross. A man cannot walk over the place where another man has been buried.

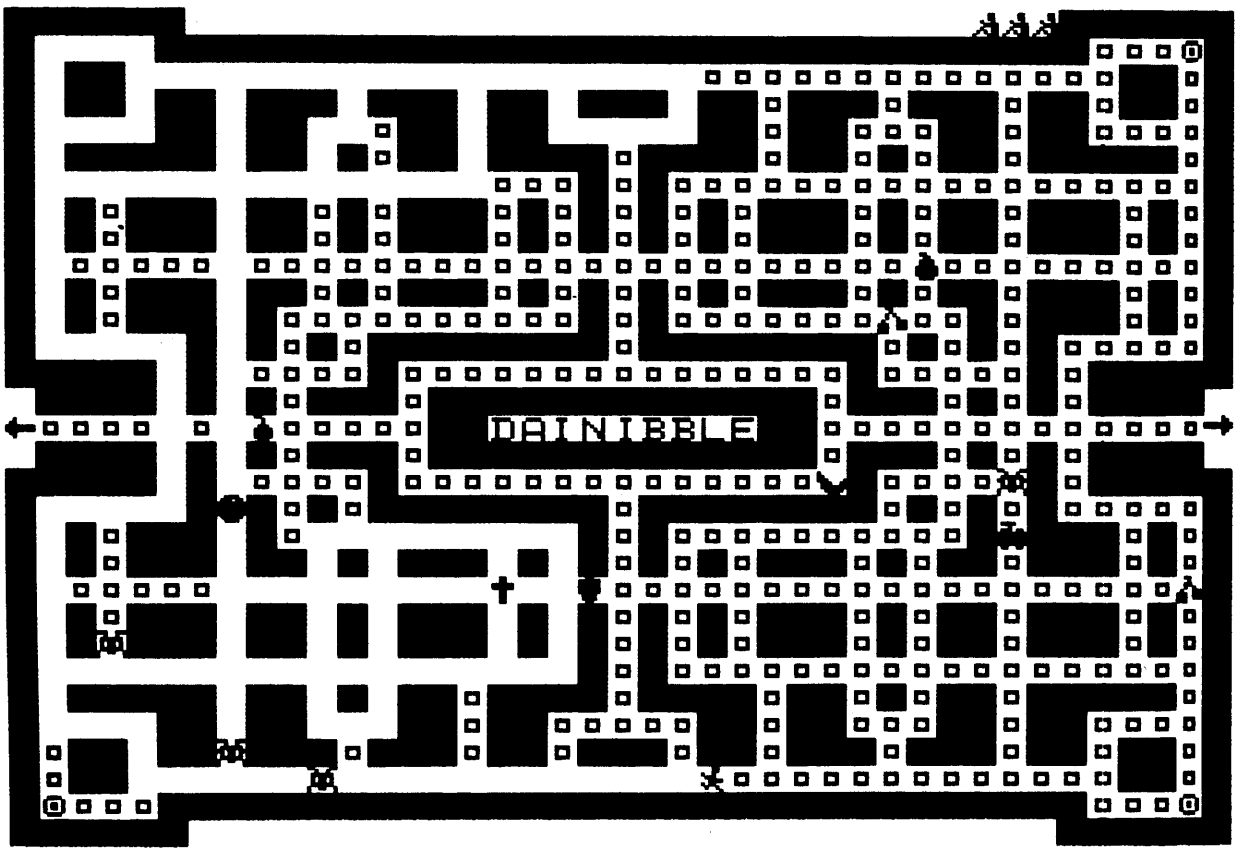
You can nibble the fruit that is moving around the maze, but you will have to nibble also dots to let the fruit appear. You get points for everything you nibble
Four monsters are trying to nibble you, but you can nibble them if a vitamine is nibbled by you and you are still extra strong because of that.

So, if you like it to nibble around, you better play DAINIBBLE.

This game is completely in machinelanguage, so it's as fast as you want it to be.

price : audio : 800 Bfr
DCR : 950 Bfr

****
* Author Hendrik-Jan van Randen *
****



SFGT

First there was FGT, with many facilities and medium speed, due to the restrictions of the DRAW-routine in DAI-BASIC. Then there was FFGT, high speed, but with restrictions of bit-access.
Now there is SFGT, a high-speed text-in-graphics program, offering the best of both previous programs

NEW SOFTWARE

SPECIFICATIONS of SFGT :

* compatible for all graphic modes
* speed : up to 300 characters/second
* the same call-procedure as FGT : CALLM 800,A$
* up to 10 tables of characters in memory at the same time, choice of table with 1 POKE !
* parameters:
  - horizontal position
  - vertical position
  - horizontal step/character
  - vertical step/character
  - vertical step on OFF/SCREEN (no error)
  - horizontal step on OFF/SCREEN (no error)
  - left margin of screen WINDOW
  - right margin of screen WINDOW
  - top of screen WINDOW
  - bottom of screen WINDOW
  - matrix start
  - matrix lenght
  - string-start (implied MID$)
  - string-length (implied MID$)
  - up/down/left/right contruction of strings
  - retain horizontal position
  - retain vertical position
  - high speed flag
  - INKEY-function
  - define background & foreground of characters
  - delay/character
  - combination of characters
* most of the parameters are optional, 3 or 4 POKES are sufficient..
* build-in animation mode

The SFGT package includes :

- SFGT
- manual
- software-manual : a program demonstrating the use of the SFGT-parameters
- a sofisticated TABLE-CREATOR (with build-in byte-shift manipulations)
- a CONVERSION-program to convert FGT-tables to SFGT-tables.

SFGT was written by Henri & Andre Lambrecht.

price of the package : audio : 1000 Bfr
DCR : 1150 Bfr

*****

DAInamic DEBUGGING TOOL

NEW SOFTWARE

DDT USERS-GUIDEpage 01

Dainamic Debugging Tool (DDT) by W. COREMANS

1.DESCRIPTION :

DDT is a program ment for testing machine language programs
The functions this program can handle are :

1. Putting up to 255 (default = 100) breakpoints in a mlp (Machine Language Program). This is done in an edit-like way with the possibility to change, insert or delete breakpoints
2. Testing the mlp in 3 different modes :
   -A trace mode to test the mlp step by step
   -A breakpoint mode which runs your mlp in real time until a breakpoint is reached
   -A breakpoint mode which gives you also breaks at rom-entrypoints (ex. bankswitching)
3. Ability to change all the 8080 processor's registers and flags during testing
4. Full control of the stack. You can display the stack-contents and change it in a similar way as you do with breakpoints (i.e. change, insert or delete 16-bit data, returnaddresses or anything else which is on the stack)
5. Disassemble your mlp or the firmware in any rom-bank, with an option to display bytes after RST1, RST4 and RST5 as DATA
6. A number of options concerning the display of mnemonics and or registers, tracing RST instructions or not, disable/enable interrupts a.o.

DDT is partly written in BASIC ,partly in machine language. The machine language part is placed in 2 stringarrays which are loaded and relocated by the BASIC program. This means it has no fixed place in memory, so the mlp to be tested can be located anywhere

Running DDT is very easy : first load your mlp, eventualy adapt the heap pointer and at last LOAD and RUN DDT. When your mlp is located in higher ram addresses (ex. near to the screen-ram) you don't have to change the heap pointer, so your mlp will be located in the free ram space behind the BASIC program

Breakpoints can be set at will into the mlp you want to test, but be sure the breakpoint points to the begin of an 8080 instruction. If this is confusing then just keep in mind that you may specifie all addresses displayed in a disassembling run, exept when the address is followed by DATA

Special care has been taken concerning interrupt handling. Your mlp can change the interrupt mask at will or even reset the interrupt system, DDT will always find its way back.

For more specific information see the complete DDT USERS-GUIDE delivered with TOOLKIT 5

SUPER DCR-LIST

TOOLKIT 5

```
        NEWS 13-14-15

        *** DCR 123 ***                  FF
        <<<<<<<<<<< A >>>>>>>>>>>
        01-2Programmes of Newsletters <<<<<<<<<<< B >>>>>>>>>>>
            13-14-15 (c) DAInamic     20-2Newsletters 13-14-15 (c)
 123    02-0DEMO C.POELS                  DAInamic 1983
        03-0MULTI PUZZLE C.POELS      21-0ASTERIX (cover of N14)
        04-0 EPROM-PROGRAMMER V3 G.K  22-0 CHARACTER INVASION N15
            NOOPS                         p101
        05-1B000 B195 mlp EPROM-PROG  23-0 3-D : eenbladige omwent
            RAMMER KNOOPS                 elingshyperboloide N15 p
        06-1SOURCE EPROM-PROGRAMMER       129
            G. KNOOPS                 24-0 BLOK IN REVERSE N15 p 1
        07-0 ROTATION ECRAN C.POELS       37
        08-0HISTOIRES                 25-0 SYMMETRISCHE WAAIERS N1
        09-0 SCHNELLER ANALOG DIGITA      5 p 137
            L WANDLER KOPP            26-0 ZANDLOPER N15 p 137
        10-0 LOGIK ANALYSATOR KOPP    27-0 CUBE ROTATION AND TRANS
        11-0 GROETEN UIT HAWAI - ALO      LATING IN SPACE N15 p 134
            HA OE.                    28-2.
        12-0 LINEMODEBYTES & COLOURT
            YPEBYTES N.LOOIJE
        13-0FGT-DEMO (FGT not includ
            ed)
        14-#MODE 7 INITIALISTIE V0.0
 123    15-#MODE 8 INITIALISTIE V0.0
        16-0MODE 7 RELOCATABLE
        17-0MODE 8 RELOCATABLE
        18-03-D GENERATOR J.ROELANTS
        19-0BASIC PAINT DEMO F.DRUIJ
```

NEWS 13-14-15

DCR 123 A

NEW SOFTWARE

01-2-ARR-Programms of Newsletters 13-14-15 (c) DA
Inamic
02-0-BAS-DEMO C.POELS
03-0-BAS-MULTI PUZZLE C.POELS
04-0-BAS- EPROM-PROGRAMMER V3 G.KNOOPS
05-1-OBJ-B000 B195 mlp EPROM-PROGRAMMER KNOOPS
06-1-OBJ-SOURCE EPROM-PROGRAMMER G. KNOOPS
07-0-BAS- ROTATION ECRAN C.POELS
08-0-BAS-HISTOIRES
09-0-BAS- SCHNELLER ANALOG DIGITAL WANDLER KOPP
10-0-BAS- LOGIK ANALYSATOR KOPP
11-0-BAS- GROETEN UIT HAWAI - ALOHA OE.
12-0-BAS- LINEMODEBYTES & COLOURTYPEBYTES N.LOOIJ
E
13-0-BAS-FGT-DEMO (FGT not included)
14-#-DNA-MODE 7 INITIALISTIE V0.0
15-#-DNA-MODE 8 INITIALISTIE V0.0
16-0-BAS-MODE 7 RELOCATABLE
17-0-BAS-MODE 8 RELOCATABLE
18-0-BAS-3-D GENERATOR J.ROELANTS
19-0-BAS-BASIC PAINT DEMO F.DRUIJFF

update 83-05-10

NEWS 13-14-15

DCR 123 B

01-2-ARR-Newsletters 13-14-15 (c) DAInamic 1983
02-0-BAS-ASTERIX (cover of N14)
03-0-BAS- CHARACTER INVASION N15 p101
04-0-BAS- 3-D : eenbladige omwentelingshyperboloi
de N15 p 129
05-0-BAS- BLOK IN REVERSE N15 p 137
06-0-BAS- SYMMETRISCHE WAAIERS N15 p 137
07-0-BAS- ZANDLOPER N15 p 137
08-0-BAS- CUBE ROTATION AND TRANSLATING IN SPACE
N15 p 134
09-2-ARR-.

update 83-05-10

TIMING PROBLEMS

PROBLEMEN BIJ (KORTE) TIJDSMETING OP DE DAIpc

Bij het ontwikkelen van de BASICODE is het mij opgevallen dat de machine-instructietijden van de 8080 microprocessor niet kloppen. Bij uitzoeken bleek dit (volgens mij) de volgende oorzaak te hebben :
De computer is hardwarematig zo opgebouwd dat de microprocessor en de videoprocessor om de beurt gebruik maken van het RAM-geheugen. Echter op het moment van de rasterterugslag (dit is het moment dat de electronenstraal van rechtsonder naar linksboven in het beeld gaat), maakt de videoprocessor geen gebruik van de RAM en draait de microprocessor op volle toeren. Mogelijk treedt dit effect ook op bij de linesync (lijnterugslag). Effectief werkt de microprocessor op gemiddeld 1.17 Mhz. Normaal zult U hiervan weinig merken. Echter bij korte tijdmeting of frequentiemeting, zoals de ontvangst van BASICODE, telex of morse dan wel satellietontvangst, dan zal er niet die frequentie uitkomen die U berekend had. Bovendien zal er een interferentie ontstaan met de 50 Hz van de rasterterugslag (dit is geen brom o.i.d. van het lichtnet). De interferentie is wellicht te verminderen door grbruik te maken van subroutines in het ROMgeheugen (vb delays).
Het genoemde effect treedt nl niet op in de ROM, daar de videoprocessor hier geen gebruik van maakt. Bij tijdsmetingen van meer dan 50 à 100 mS zult U weinig last hebben van het genoemde effect omdat dan de rasterterugslag weggemiddeld wordt. Ik hoop hiermee enkele mensen slapeloze nachten bespaard te hebben.

met vriendelijke groeten,
Th. van Lieshout
Postgalei 5
1687 VP WOGNUM
NEDERLAND
tel 02297-2648

DAINIBBLE

After DAIPANIC,CENTIPEDE,ACROBATES, another 100% machine language game : high resolution (MODE 5) , very fast and very nibble ....

audio : 800 Bfr
DCR : 950 Bfr

SFGT

The very best of text-in-graphics , superspeed and all the options you want ...

audio : 1000 Bfr
DCR : 1150 Bfr

NEW SOFTWARE

TOOLKIT 5

* BASICODE II
* DAInamic Debugging Tool (see description in this issue)
* DAICOM communication program (see Newsletter 15)
* SUPER DCR-LIST (see sample printout in this issue)
* PAINT-routine in machine language (a very fast one)
* SOFTWARE PRINTER SPOOLER (see this issue)
* Gothic characters with FGT

audio : 1000 Bfr

DCR : 1150 Bfr

> > > > > > PROGRAMMEERTECHNIEKEN < < < < < <

Ik wilde deze keer weer eens de werksnelheid van een programma onder de loep nemen. Ik weet heus wel dat ook overzichtelijkheid en veranderbaarheid van een programma belangrijk zijn, vaak zelfs van doorslaggevend belang, maar over de overzichtelijkheid van een programma valt weinig te vertellen. Aanwijzingen kunnen in het algemeen wel gegeven worden maar detailzaken hangen te veel van het programma in kwestie af en hangen ook ten nauwste samen met de smaak van de betrokken programmeur. In snelheidszaken ligt dat anders: als ik een bepaalde methode voorstel kan iedereen zelf nagaan dat die methode sneller werkt dan een andere. Werkt hij niet sneller dan de door U gebruikte methode; ook dan is er geen probleem, ik kan vaststellen dat die methode beter (sneller) is en zal er dan in een volgend artikel op terugkomen. U kunt er natuurlijk ook zelf een artikel over schrijven.

Het eerste probleem is natuurlijk: Hoe test ik de snelheid van een bepaalde instructie of bepaald programmadeel ? Nu, hierbij kan de DAI ons zelf erg goed van dienst zijn. Het best is het om een apart timer-programma te maken dat indien gewenst een onderdeel kan zijn van het grote programma zolang dat nog in het ontwerpstadium is. Hieronder het geraamte van dit programma:

```
5     WAIT TIME 2:POKE #1BF,#FF:POKE #1BE,#FF
95    T1BE=PEEK(#1BE):T1BF=PEEK(#1BF):PRINT(#FFFF-T1BE-T1BF*256)/50.0
```

Iemand die programmeert met de goede gewoonte om de regels te nummeren met 10-vouden kan de regels 5 en 95 eenvoudig om het te testen deel zetten.
Toch behoeft dit programma enige toelichting. De variabelen T1BE en T1BF zijn natuurlijk integer. We maken gebruik van de 20msec klok die de waarde op #1BE en #1BF elke 20 milliseconde met 1 vermindert tot dat die nul is.
Met de POKE's uit regel 5 vullen we deze twee bytes met #FFFF ,eerst de high en dan de low byte; de WAIT TIME ervoor zorgt voor een schone start.
Aangezien ook de WAIT TIME instructie gebruik maakt van deze adressen mag in het te testen programma (deel) geen WAITTIME voorkomen. Andere WAIT's wel maar lijkt mij niet handig omdat de tijd dan door die WAIT's beinvloed wordt.
Het uitlezen van de adressen in regel 95 is het best in deze volgorde; bij andere volgorde kan de timing verkeerd gaan. Voorbeeld: #1BE staat op #35 en #1BF op #F6; als er geen aftelling is tussen de twee PEEK's zal de volgorde er niet toe doen, als er echter tussen de eerste en tweede PEEK in wel een vermindering plaatsvindt zal de uitkomst 20 msec kleiner zijn dus 50.10 ipv 50.12 seconden Geen erg grote fout misschien, maar onjuist; we moeten niet de snelheid beinvloeden met het timerprogramma zelf. Toch is er wel een risico aan verbonden : eens op de tweehonderdzesenvijftig keer staat #1BE op nul en wordt dan met een verminderd dan zal de klok niet 20 ms maar 256 * 20 ms is ruim vijf seconden verkeerd aangeven. Dit verschil is echter zo groot dat dat moet opvallen.
De deling M O E T door 50.0 en niet door 50 omdat we een fpt-uitkomst willen. Als we dit niet doen kappen we alles naar beneden af op hele seconden.

We gaan wat experimenteren met deze timer en komen dan tot vermoedelijk voor velen opzienbarendende resultaten. Ik vermeld met opzet de resultaten niet. Geinterresseerden kunnen gemakkelijk zelf de tests uitvoeren en dan verrast door de resultaten zelf verder gaan proberen. Voor programmadelen die erg belangrijk zijn voor de snelheid zal het bijna altijd lonen een aantal mogelijkheden op snelheid te testen. Veel mensen zien echter niet dat er verschillende methoden zijn. Vandaar enige voorbeelden, waarbij de uitkomsten soms veel verschillen maar soms ook nauwelijks of niet.
Aangezien een enkele instructie zo snel gedaan wordt dat dit met deze methode niet te timen is, zullen we er een FOR - NEXT loop omheen zetten. Het timen van deze FOR - NEXT loop zelf.

```
      10    FOR I=1 TO 5000:NEXT
```

en doe die zowel na IMPFPT als na IMPINT.
daarna proberen we:

```
      10    FOR I=1 TO 10000 STEP 2:NEXT
```

en dit is toch ook een loze loop die 5000 maal doorlopen wordt ?

We gaan er eens iets tussen zetten b.v. A=3 of A=B. De tijden lopen echter zo op dat we de loop tot 1000 gaan beperken. En misschien wel tot 500 of 100 zeker als functies zoals SQR,SIN,COS,TAN en dergelijke gebruikt gaan worden.
Het is wel nuttig om een keer een loop zowel 1000 als 10000 maal te laten uitvoeren om eens te zien dat het inderdaad tien maal zolang duurt. Om redenen van praktische aard is een tijd tussen 10 en 30 seconden het meest aanbevelenswaardig. Korter maakt resultaat onbetrouwbaarder en langer kost te veel tijd. Daarbij deze tijdmeting kan maximaal een periode van #FFFF * 20 ms (dus ruim 20 minuten) meten.

We gaan verder experimenteren met een eenvoudige toewijzing: binnen de loop zetten we : A=8 of A=8.0 of A=B of A=B+4 of A=4+B
dit alles na IMPINT of na IMPFPT. Dit is op zich niet zo belangrijk maar vergelijkt U niet oude FPT-resultaten met nieuwe INT-resultaten. Pas op niet altijd is integer sneller! Gelooft U mij niet ? Probeer A%=SQR(P%) en A!=SQR(P!) maar.

We willen in A de waarde van twee B plus 4 krijgen. Vergelijkt U de volgende mogelijkheden eens.

```
A=2*B+4      erg voor de hand liggend niet waar?
A=B*2+4      anders ?
A=B+B+4      dit is sneller !!!
A=B+4+B of A=4+B+B
A=(B+B)+4    maakt dit iets uit ?
A=B+(B+4)    moet vergissing zijn.
A=(((((B)))))+((((B))))+(((4))) valt mee.
```

We gaan verder: A=B SHL 1 dit is sneller dan A=B*B maar pas op A=B SHL 5 is trager dan A=B*32. Voor de BASIC V1.1 bezitters A=-B of A=0-B
Probeert U ook eens het verschil tussen de gehele FOR - NEXT loop op een regel of op twee of drie regels, dat kan soms ook heel wat uitmaken.In principe altijd proberen de FOR en de NEXT op dezelfde regel te krijgen.

In het tweede deel wil ik wat dieper ingaan op de instructies die het programma op een andere plaats laten vervolgen. De instructies zijn GOTO , GOSUB , IF xx GOTO , IF xx THEN , ON W GOTO en ON W GOSUB.
Ik zal de instructie IF xx THEN gevolgd door een regelnummer nooit gebruiken. Ik vind een verplaatsopdracht zo belangrijk dat je dat duidelijk moet kunnen zien door een GOTO. De logica is voor mij ook: bij GOTO geef je duidelijk aan ergens anders te vervolgen; bij THEN zou men kunnen denken dat alleen die instructies die op de opgegeven regel staan dienen te worden uitgevoerd om daarna weer op het oude punt door te gaan. Dit is wel niet zo maar ik kies toch om de bovengenoemde logische redenen voor IF xx GOTO.

De GOTO is de meest vervloekte instructie in BASIC, vooral de tegenstanders van de italiaanse keuken zijn vaak erg fel. Met GOTO kun je namelijk heerlijke spagetti maken. De voorstanders vinden dit juist een voordeel evenals het totaal onleesbaar (begrijpelijk) worden van programma's. Dit alles onder het motto: 'Schaf nu space-invaders aan en U krijgt er een labyrint bij cadeau. Of zoals ik het zelf graag uitdruk : jump as jump can.

Ik geef even een voorbeeld dat ik reeds in vele inzendingen ben tegengekomen:

```
120     IF A=3 GOTO 140
130     GOTO 160
140     P=2
150     GOTO 160
160     END:REM of hopeless program
```

Natuurlijk werkt het. Deed het 't maar niet dan zou de maker misschien wat gaan nadenken. Als A gelijk aan drie is wordt P twee gemaakt en anders blijft P onveranderd. Maar dit kan toch wel een stuk beter:

```
120    IF A<>3 GOTO 160
140    P=2
160    END:REM of better program
```

Of als we het kort willen houden:

```
120    IF A<>3 GOTO 160:P=2
160    END:REM of shorter version
```

Of als we de GOTO willen vermijden

```
120    IF A=3 THEN P=2
```

Zelfs de IF kunnen we vermijden (ik beweer niet dat dit handiger is)

```
120    P=P+P*(SGN(ABS(A-3))-1)-2*(SGN(ABS(A-3))-1)
```

Gaat U deze regel maar eens na en probeert u ook eens de IF te vermijden, het is een zeer goede oefening.

In een vorig artikel is al eens uitgelegd waarom we sprongen naar hogere regelnummers (dwz ver naar achteren in de listing) moeten trachten te voorkomen in snelheid bepalende delen. Maar wist U dat de ON W GOTO xx sneller werkt dan de IF xx GOTO ? Voorbeeld IF PEEK(I)=1 GOTO 200 is trager dan ON PEEK(I) GOTO 200.

Niet altijd zal echter de te testen variabele geschikt zijn om met de ON W GOTO te gebruiken. Een extra berekening kan de winst dan weer geheel teniet doen. Controleer zeker aan het eind of er geen GOTO's te veel in een programma staan. Het staat zo dom om in regel 50 aan te treffen GOTO 170 en dan op 170 alleen GOTO 220 en dan op 220 alleen GOTO 30. Dit is echt geen fantasie; ik heb programma's (inderdaad meervoud) waar zoits in staat. Maar ik heb er ook begrip voor. Normaal zal niemand zoiets programmeren, maar op regel 170 stond bij een eerdere versie nog een andere instructie voor de GOTO. Later bleek die overbodig maar de GOTO niet. Om een logische programmavoortgang te krijgen keerde men alleen vanaf regel 220 terug naar het begin van het programma. Dus vanaf regel 170 naar regel 220 en niet naar regel 30. Nu beter te begrijpen maar nog steeds niet de snelste of overzichtelijkste versie.

Nu wat de GOSUB betreft. Ook hier tegenstrijdige belangen. De overzichtelijkheid van het programma is gebaat met een uitgang (RETURN) en een ingang per subroutine maar de snelheid kan soms gebaat zijn met meerdere in en uitgangen van een subroutine.
Voorbeeld:

```
70     GOSUB 2000
::
130    GOSUB 2010
::
2000   IF A=5 THEN RETURN
2010   P=2
2020   RETURN
```

Ik kies zelf in zulke gevallen bijna altijd voor 2000 IF A=5 GOTO 2020, dus laat ik overzichtelijkheid preveleren.
De instructie ON xx GOSUB kom ik zo zelden tegen dat ik me afvraag of de mensen hem wel kennen. Probeert U het gerust als de gelegenheid zich voordoet.

Elders in dit blad zult U een artikeltje van Just van Dunne' aantreffen over de GOTO. Wat hij daar uitlegt kan soms zeer handig zijn maar kost wel meer tijd dan de normale GOTO. Het voordeel is dan ook gelegen in gebruik in bijzondere gevallen. Ook zult U nog een klein (2 regels) programmaatje van mij kunnen vinden waar ik de timer om heen heb gezet. In dit programma offer ik echt alles op aan de snelheid en pas verschillende 'gemene' trucs toe om de snelheid te verhogen. Ondermeer laat ik een NEXT door twee verschillende FOR's gebruiken en of dit nog niet 'gemeen' genoeg is ook nog eens een FOR twee NEXTen gebruiken.

Verder zijn er reacties binnengekomen op het 'cirkel'programma van nummer 14. Meerderen hebben vastgesteld dat er snel ronde ellipsen getekend werden en geen echte cirkels. Dank voor uw bijdrage, die wij niet alle kunnen publiceren maar maar wel een ervan of een combinatie. Als uw bijdrage niet expliciet gebruikt is laat dit U er dan niet van weerhouden nogmaals in te zenden. Van alle inzendingen wordt nota genomen.

Succes met de timer en vindt U iets zeer opzienbarends laat het dan even weten.

Frank H. Druijff

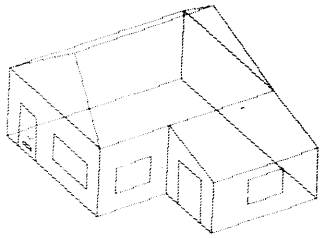
PROBLEM PROGRAM

Problem - program - problem - program - problem - program - problem - program

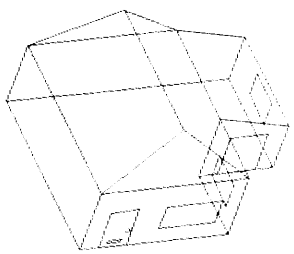
Hereafter follows a little 'basic'program of only two lines (10 and 20).
The lines 5 and 95 are used for timing purpose. Can you do it faster ?
But what does it do? I challenge you, try to understand what this program does before you type it in. But be careful it runs only in a clean memory. So if you have typed it in, put it on cassette, turn your machine off and switch it on again, load the program and run it. Isn't it fast ?

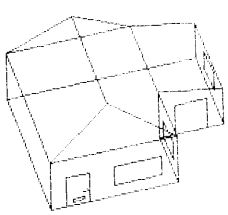
```
5     WAIT TIME 1:POKE #1BE,#FF:POKE #1BF,#FF:POKE #1BE,#FF
10    CURSOR 1,22:POKE #BF61,#32:POKE #131,1:FOR G=0 TO 14:ON PEEK(G+#32F0)
 GOTO 20:FOR I=(G+G+6)*G+#32F3 TO #34E1 STEP G+G+3:POKE I,1:NEXT
20    NEXT:FOR I=#32F0 TO #34E1:ON PEEK(I) GOTO 20:PRINT I+I-#65DD:POKE #7B
,0:NEXT
95    B=PEEK(#1BE):A=PEEK(#1BF):PRINT :PRINT (#FFFF-B-A*256)/50.0
100   REM ****** IMPINT ******
```

```
2     REM ********************************************
3     REM ***    HARDCOPY SOURCE PASCAL          ******
4     REM ***    (c) ALAIN MARIATTE 1983         ******
5     REM *** pour imprimante serie,vitesse ajustable *
6     REM ********************************************
7     REM
8     REM
10    REM LA COMMANDE P  DE TINY PASCAL NE PERMET PAS D'
20    REM UTILISER UNE IMPRIMANTE DONT LA VITESSE EST <> DE
30    REM 9600 Bauds.IL SUFFIT DE VIDER LE BUFFER D'EDITION
40    REM PAR CE COURT PROGRAMME BASIC,IMPLANTE A PARTIR DE
50    REM #8000, COEXISTANT AVEC PASCAL,POUR AVOIR L'EFFET
60    REM DESIRE.
70    REM
80    REM
90    REM
100   POKE #FFF5,#A0:REM VITESSE RS 232.ICI,4800 Bauds
110   POKE #131,0:REM OUVERTURE DU PORT RS 232
120   I=#2000-1
130   I=I+1:REM DEBUT DE L'EDIT BUFFER
140   A=PEEK(I):IF A=0 THEN POKE #131,1:END
150   A$=CHR$(A)
160   PRINT A$;
170   GOTO 130
```



ROTATING BUNGALOW

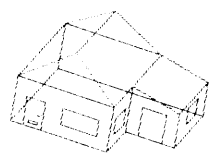


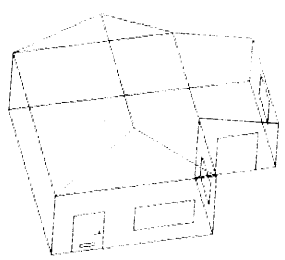


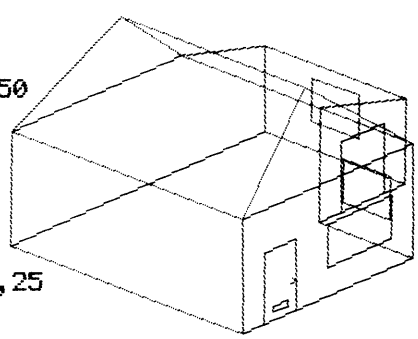
```
1     REM ***********************************
2     REM ***** 3DPLOT VERSIE 19.06.82 *****
3     REM *****  A.VINGERLING, R'DAM   *****
4     REM ***********************************
100   CLEAR 2750:PRINT CHR$(12);:GOSUB 7000:GOTO 103
101   COLORT 5 15 0 0:MODE 0:CURSOR 13,12
102   PRINT "INITIALISATIE, EVEN GEDULD SVP"
103   SF=1.0:GOSUB 2000:REM DATA INLEZEN
104   GOSUB 6000:REM TOON MOGELIJKHEDEN
120   COLORG 15 0 15 0:MODE 6:MODE 6
130   GOSUB 600:REM NIEUWE PLOT TEKENEN
140   GOSUB 810:REM KLEUR WIJZIGEN
150   GOSUB 700:REM OUDE PLOT WISSEN
160   Q%=1-Q%:KS=ABS(KS)
170   IF FL%=1.0 THEN 210
180   A%=GETC:GOSUB 880:IF A%<16 THEN 180
190   IF A%=ASC("H") THEN GOSUB 4000:GOTO 130
200   GOSUB 5000:ON P210% GOTO 210
205   IF A%=ASC("/") AND AA%=0 THEN 180
206   GOTO 230
210   P210%=0:IF AA%>0.0 THEN A%=AAA%(AA%):AA%=AA%-1:GOTO 230
220   AA%=0:FL%=0:GOTO 180
230   IF A%=ASC(".") THEN VF=1.0/0.8:GOTO 250:REM >
240   VF=0.8:REM <
250   IF A%=ASC("1") THEN A%=1:GOTO 280
260   IF A%=ASC("2") THEN A%=2:GOTO 280
270   IF A%<=19 THEN A%=A%-13
280   REM FOR P=1 TO NP:XX!(P)=X!(P):YY!(P)=Y!(P):NEXT
290   IF A%=ASC(",") OR A%=ASC(".") THEN GOSUB 900:GOTO 130
300   IF A%>=0 THEN ON A% GOTO 320,330,400,410,500,510
305   IF A%=ASC("C") THEN GOSUB 800:CF%=1-CF%:GOTO 130
310   GOTO 180
320   KS=-KS
330   FOR P%=1 TO NP%:X=X(P%):Y=Y(P%):X(P%)=X*KC+Y*KS:Y(P%)=Y*KC-X*KS:NEXT
340   GOTO 130
400   KS=-KS
410   FOR P%=1 TO NP%:Y=Y(P%):Z=Z(P%):Y(P%)=Y*KC+Z*KS:Z(P%)=Z*KC-Y*KS:NEXT
420   GOTO 130
500   KS=-KS
510   FOR P%=1 TO NP%:Z=Z(P%):X=X(P%):Z(P%)=Z*KC+X*KS:X(P%)=X*KC-Z*KS:NEXT
520   GOTO 130
600   REM NIEUWE PLOT TEKENEN
610   FOR L%=1 TO NL%:PA%=LA%(L%):PB%=LB%(L%)
620   DRAW X(PA%)+XC%,Y(PA%)+YC% X(PB%)+XC%,Y(PB%)+YC% 17+Q%*2
630   NEXT:RETURN
700   REM OUDE PLOT WISSEN
710   FILL 0,0 XMAX,YMAX 18-2*Q%:RETURN
800   IF CF%=0 THEN FILL 306,0 300,6 15:GOTO 802
801   FILL 306,0 300,6 0
802   RETURN
810   IF CF%=0 THEN COLORG 0 15*(1-Q%) 15*Q% 15:GOTO 812
811   COLORG 15 15*Q% 15-Q%*15 0
812   RETURN
880   QT%=QT%+1:IF QT%<15 THEN DRAW 305,1 300,6 15:DRAW 305,6 300,1 15:RETU
RN
881   DRAW 305,1 300,6 0:DRAW 305,6 300,1 0:IF QT%<30 THEN RETURN
882   QT%=0:RETURN
900   FOR QQ%=1 TO NP%:X(QQ%)=VF*X(QQ%):Y(QQ%)=VF*Y(QQ%):Z(QQ%)=VF*Z(QQ%):N
EXT:RETURN
2000  REM -- DATA INLEZEN --
2005  POKE #75,32:CURSOR 0,5:PRINT "data inlezen:";
2006  CURSOR 15,5:DPI=12.0:GOSUB 5200
2010  XC%=165:YC%=127:Q%=1
2020  READ NP%,NL%:PRINT 1.0
2030  DIM X(NP%),Y(NP%),Z(NP%)
```

```
2040  DIM XX(NP%),YY(NP%)
2050  DIM LA%(NL%),LB%(NL%),AAA%(100.0)
2060  FOR P%=1 TO NP%:READ X(P%),Y(P%),Z(P%)
2070  X(P%)=X(P%)*SF
2080  Y(P%)=(Y(P%)-20.0)*SF
2090  Z(P%)=Z(P%)*SF
2100  CURSOR 20,5:PRINT NP%-P%;:NEXT
2101  CURSOR 15,5:PRINT 0.0
2110  FOR L%=1 TO NL%:READ LA%(L%),LB%(L%)
2111  CURSOR 20,5:PRINT NL%-L%;:NEXT
2120  POKE #75,95:RETURN
4000  GOSUB 6000:MODE 6:FF=1.0:GOSUB 900:RETURN
5000  IF A%=47.0 AND AA%>0.0 THEN FL%=1:GOSUB 800:P210%=1:RETURN
5010  IF A%=47 AND AA%=0 THEN RETURN
5015  AA%=AA%+1
5020  IF A%=49 THEN AAA%(AA%)=50:GOSUB 800:RETURN
5030  IF A%=50 THEN AAA%(AA%)=49:GOSUB 800:RETURN
5040  IF A%=17 THEN AAA%(AA%)=16:GOSUB 800:RETURN
5050  IF A%=16 THEN AAA%(AA%)=17:GOSUB 800:RETURN
5060  IF A%=19 THEN AAA%(AA%)=18:GOSUB 800:RETURN
5065  IF A%=18 THEN AAA%(AA%)=19:GOSUB 800:RETURN
5070  IF A%=46 THEN AAA%(AA%)=44:GOSUB 800:RETURN
5080  IF A%=44 THEN AAA%(AA%)=46:GOSUB 800:RETURN
5081  IF A%=ASC("H") THEN MODE 0:GOSUB 6000:MODE 6:GOSUB 800:RETURN
5090  IF A%<>ASC("S") THEN RETURN
5091  DP=180.0/DPI:PRINT "Hoekverdraaing is nu";:CURSOR 20,3:PRINT 180.0/DP
I;"   (uitgangshoek =";180.0/DPI;" graden)"
5092  PRINT "grotere hoek : bovenste cursortoets + repeattoets":PRINT "klei
nere hoek: onderste cursortoets + repeattoets"
5093  A%=GETC:IF A%=0 THEN 5093
5094  IF A%=94 THEN MODE 6:A%=0:RETURN
5095  IF A%=16 THEN DP=DP+1.0:CURSOR 20,3:PRINT "       ";:CURSOR 20,3:PRIN
T DP;:GOTO 5093
5096  IF A%=17.0 THEN DP=DP-1.0:CURSOR 20,3:PRINT "       ";:CURSOR 20,3:PR
INT DP;:GOTO 5093
5097  IF A%<>13.0 OR DP=0.0 THEN 5093:DPI=180.0/DP
5098  MODE 6:GOSUB 5200:GOSUB 800:A%=0:RETURN
5200  PHI=PI/DPI:KS=SIN(PHI):KC=COS(PHI):RETURN
6000  COLORT 5 5 5 5:MODE 0:PRINT CHR$(12);:IF A%=ASC("H") THEN CURSOR 0,20
6010  PRINT "============================================================"
6011  PRINT "Dit programma plot een ruimtelijke figuur en maakt het"
6020  PRINT "mogelijk deze te wentelen, te vergroten, of te verkleinen"
6030  PRINT "m.b.v de cursortoetsen en met 1,2,<,> en /":PRINT
6040  PRINT CHR$(136);" : roteren naar links                +XZ"
6050  PRINT CHR$(137);" : roteren naar rechts               -XZ"
6060  PRINT CHR$(94);" : kantelen achterover               +YZ"
6070  PRINT CHR$(140);" : kantelen voorover                 -YZ"
6080  PRINT "1 : wentelen met de klok mee          +XY"
6090  PRINT "2 : wentelen tegen de klok in         -XY"
6100  PRINT "> : vergroten (geen shift gebruiken)   +Y"
6110  PRINT "< : verkleinen(geen shift gebruiken)   -Y"
6120  PRINT "/ : alles in omgekeerde volgorde tot de oorspr. figuur"
6125  PRINT "H : HELProutine, toont de verschillende commando's"
6126  PRINT "S : Stapgrootte hoekverdraaing instellen m.b.v cursortoetsen"
6127  PRINT "C : Change; inverteert de kleuren"
6128  IF A%=ASC("H") THEN 6170
6129  PRINT "============================================================"
6130  PRINT "IN DE DATASET : NP= aantal knooppunten"
6140  PRINT SPC(16);"NL= aantal lijnen"
6150  PRINT SPC(16);"x1,y1,z1,x2,y2,z2,= knooppuntscoordinaten"
6160  PRINT SPC(16);"b1,e1,b2,e2,= begin/eind lijn 1,2, enz."
6170  PRINT "============================================================"
6180  PRINT "Het programma begint na aantippen van een toets";
6185  COLORT 5 15 0 0
6190  IF GETC=0.0 THEN 6190
6200  MODE 6:RETURN
```

```
7000  REM MELDING
7001  COLORT 5 5 5 5:MODE 0:PRINT CHR$(12);:CURSOR 0,15
7002  PRINT "00000   000000        000000    0        00000    0000000"
7003  PRINT "     0   0    0        0    0   0        0     0      0   "
7004  PRINT "     0   0    0        0    0   0        0     0      0   "
7005  PRINT " 0000    0    0  000   00000    0        0     0      0   "
7006  PRINT "     0   0    0        0        0        0     0      0   "
7007  PRINT "     0   0    0        0        0        0     0      0   "
7008  PRINT "00000   000000        000       0000000  00000      000  "
7009  COLORT 5 15 0 0:RETURN
7010  DATA 43,49
7020  REM x1,y1,z1,x2,y2,z2..knooppuntscoordinaten
7030  DATA -60,0,-50,20,0,-50,-60,0,50,-60,40,-50
7040  DATA 20,0,-10,20,40,-50,60,0,50,-60,40,50
7050  DATA -20,70,-40,60,0,-10,20,40,-10,20,40,50
7060  DATA 60,30,50,-20,70,40,60,30,-10
7070  DATA -50,0,-50,-50,25,-50,-35,25,-50,-35,0,-50
7080  DATA -20,10,-50,-20,25,-50,10,25,-50,10,10,-50
7090  DATA 20,10,-40,20,25,-40,20,25,-20,20,10,-20
7100  DATA 30,0,-10,30,25,-10,50,25,-10,50,0,-10
7110  DATA 60,10,10,60,25,10,60,25,30,60,10,30,0,0,0
7111  DATA -46,3,-50,     -46,5,-50, -39,5,-50, -39,3,-50
7112  DATA -40,12,-52, -37,12,-52, -37,12,-50
7120  REM b1,e1,b2,..begin- en eindpunt v.d.staven
7130  DATA 1,2,1,3,1,4,2,5,2,6,3,7,3,8,4,6
7140  DATA 4,8,4,9,5,10,5,11,6,9,6,12,7,10,7,13
7150  DATA 8,12,8,14,9,14,10,15,11,15,12,13,12,14,13,15
7160  DATA 16,17,17,18,18,19,20,21,21,22,22,23,23,20,24,25
7170  DATA 25,26,26,27,27,24,28,29,29,30,30,31,32,33
7180  DATA 33,34,34,35,35,32,36,36,37,38
7190  DATA 38,39,39,40,40,37,41,42,42,43
8000  REM NP,NL   aantal knooppunten/lijnen
8010  DATA 43,49
8020  REM x1,y1,z1,x2,y2,z2..knooppuntscoordinaten
8030  DATA -60,0,-50,20,0,-50,-60,0,50,-60,40,-50
8040  DATA 20,0,-10,20,40,-50,60,0,50,-60,40,50
8050  DATA -20,70,-40,60,0,-10,20,40,-10,20,40,50
8060  DATA 60,30,50,-20,70,40,60,30,-10
8070  DATA -50,0,-50,-50,25,-50,-35,25,-50,-35,0,-50
8080  DATA -20,10,-50,-20,25,-50,10,25,-50,10,10,-50
8090  DATA 20,10,-40,20,25,-40,20,25,-20,20,10,-20
8100  DATA 30,0,-10,30,25,-10,50,25,-10,50,0,-10
8110  DATA 60,10,10,60,25,10,60,25,30,60,10,30,0,0,0
8111  DATA -46,3,-50,     -46,5,-50, -39,5,-50, -39,3,-50
8112  DATA -40,12,-52, -37,12,-52, -37,12,-50
8120  REM b1,e1,b2,..begin- en eindpunt v.d.staven
8130  DATA 1,2,1,3,1,4,2,5,2,6,3,7,3,8,4,6
8140  DATA 4,8,4,9,5,10,5,11,6,9,6,12,7,10,7,13
8150  DATA 8,12,8,14,9,14,10,15,11,15,12,13,12,14,13,15
8160  DATA 16,17,17,18,18,19,20,21,21,22,22,23,23,20,24,25
8170  DATA 25,26,26,27,27,24,28,29,29,30,30,31,32,33
8180  DATA 33,34,34,35,35,32,36,36,37,38
8190  DATA 38,39,39,40,40,37,41,42,42,43
9999  END
10000 REM MELDING
10001 COLORT 5 5 5 5:MODE 0:PRINT CHR$(12);:CURSOR 0,15
10002 PRINT "00000   000000        000000    0        00000    0000000"
10003 PRINT "     0   0    0        0    0   0        0     0      0   "
10004 PRINT "     0   0    0        0    0   0        0     0      0   "
10005 PRINT " 0000    0    0  000   00000    0        0     0      0   "
10006 PRINT "     0   0    0        0        0        0     0      0   "
10007 PRINT "     0   0    0        0        0        0     0      0   "
10008 PRINT "00000   000000        000       0000000  00000      000  "
10009 COLORT 5 15 0 0:RETURN
```







FGT-DEMO

```
10    GOTO 1500
80    REM ..................................................
100   REM .     Here is a small demonstration program     .
110   REM .     which should hopefully solve your FGT     .
120   REM .     problems. Read through the listing        .
130   REM .     carefully after observing what happens    .
140   REM .     on the screen. You should then be able    .
150   REM .     to duplicate the effects in your own      .
160   REM .     programs. The subroutine from lines       .
170   REM .     1030-1050 should be used in all your      .
180   REM .     BASIC programs that use FGT. You may      .
190   REM .     of course compress the lines by using     .
200   REM .     colons.                                   .
210   REM .                              Dave Atherton    .
220   REM ..................................................
1001  REM FGT BASIC CONTROL SUBROUTINE
1002  REM CC=TEXT COLOUR (0-15)
1003  REM X=X START POSN (0-335):Y=Y START POSN (0-256)
1004  REM SP=SPACING BETWEEN CHARACTERS(0-255)
1005  REM DF=DIMENSION OF CHARACTERS(0-15)
1006  REM VF=TEXT PRINTED VERTICALLY(1) OR NOT(0)
1007  REM ZF=LEFT TO RIGHT(0) OR RIGHT TO LEFT(1)
1008  REM PF=NEW X,Y USED(0),EXISTING POSN USED NEW XY IGNORED(1)
1009  REM T$=TEXT PRINTED (UP TO 255 CHARACTERS)
1010  REM FF=BACKGROUND FILLED IN (1) OR NOT (0)
1011  REM FC=FILLCOLOUR FOR BACKGROUND (0-15)
1012  REM ID=VERTICAL SCALE(0-255)
1030  POKE #2F0,FC%*#40+CC%
1035  POKE #2F1,FF%*#80+PF%*#40+ZF%*#20+VF%*#10+DF%
1040  POKE #2F2,X% MOD 256
1041  POKE #2F3,X%/256
1042  POKE #2F4,Y%
1045  POKE #2F5,SP%
1047  POKE #2F6,ID%
1050  CALLM #300,T$:RETURN
1500  MODE 0:PRINT CHR$(12):CLEAR 1000
1510  DIM A$(3):FOR AA%=0 TO 2:READ A$(AA%):NEXT
1512  DATA "THIS SHOWS THE","FGT PROGRAM","WITH THE"
1530  GOSUB 2000:SOUND 0 0 15 0 FREQ(600):WAIT TIME 10:SOUND OFF
1550  GET%=GETC:IF GET%=0 THEN 1550:END
2000  REM FGT PARAMETERS
2010  X%=20:Y%=130:SP%=9:FC%=1:CC%=0:DF%=0:ID%=15:PF%=0:ZF%=0:FF%=0:ST%=15:
VF%=0
2013  COLORG 8 15 5 2
2014  A$(3)="STANDARD CONDITIONS":GOSUB 3000
2015  VF%=1:A$(3)="VERTICAL FLAG SET":GOSUB 3000:VF%=0
2016  DF%=1:A$(3)="DIMENSIONS ALTERED":GOSUB 3000:DF%=0
2017  ZF%=1:X%=200:Y%=10:A$(3)="DIRECTION FLAG SET":GOSUB 3000:ZF%=0
2018  SP%=18:A$(3)="SPACING ALTERED":GOSUB 3000:SP%=9
2019  PF%=1:A$(3)="PRINT FLAG SET":GOSUB 3000:PF%=0
2020  CC%=RND(16):A$(3)="COLOUR ALTERED":GOSUB 3000:CC%=0
2021  FC%=7:A$(3)="FILLCOLOUR UNSET":GOSUB 3000:FC%=1
2022  ST%=10:A$(3)="LINE SPACING ALTERED":GOSUB 3000:ST%=15
2023  FF%=1:A$(3)="FF FLAG SET":GOSUB 3000:FF%=0
2024  ID%=180:A$(3)="ID ALTERED":GOSUB 3000:ID%=15
2070  REM WITH THIS YOU SHOULD BE ABLE TO DO YOUR OWN FGT
2080  RETURN
3000  MODE 5
3005  FOR F%=X% TO X%+3*ST% STEP ST%
3010  T$=A$(S%):S%=S%+1:Y%=Y%-ST%*(-ZF%*2+1)
3030  GOSUB 1030:NEXT:WAIT TIME 50:S%=0
3060  X%=20:Y%=130:RETURN
```

SORTING DEMO

```
1     CLEAR 1000:DIM S$(10.0),A$(10.0)
10    REM SORTEERDEMO
15    COLORT 8 0 8 8:RF$="N"
19    POKE #75,32
20    GOSUB 1500
30    PRINT "            # VISISORT #"
31    PRINT TAB(20);"bert maertens ":FOR SX!=1.0 TO 37.0:A$=A$+CHR$(64):NEXT:PRI
NT A$
32    A$=""
40    PRINT :PRINT "        1) BUBBLE SORT":PRINT "        2) VERTRAAGD SORTEREN"
50    PRINT "       3) SHELL-METZNER SORT":PRINT "       4) EINDE PROGRAMMA"
80    PRINT :PRINT "Uw keuze :"
81    CH!=GETC:IF CH!=0.0 THEN 82
82    CH!=CH!-48.0
90    IF CH!<1.0 OR CH!>4.0 OR CH!<>INT(CH!) THEN 81
100   IF CH!=4.0 THEN PRINT CHR$(12):END
110   GOSUB 1500:GOSUB 160
120   Y!=22.0:FOR K=1 TO NS:A$(K)=S$(K):CURSOR 2,Y!:PRINT A$(K):Y!=Y!-1.0:NEXT
130   NE=0:NC=0:CURSOR 20,22:PRINT "# ITEMS =";NS:CURSOR 20,21:PRINT "# VERGELIJ
KEN =";NC:CURSOR 20,20:PRINT "# UITWISSELEN =";NE
140   ON CH! GOSUB 310,400,540
150   CURSOR 0,12:PRINT "LIJST OPNIEUW SORTEREN <J/N>"
151   RF!=GETC:IF RF!=0.0 THEN 151
152   RF$=CHR$(RF!):GOTO 20
160   IF LEFT$(RF$,1)="J" THEN 250
170   NS$="10":PRINT "Hoeveel items sorteren (MAX 9)";
171   NS!=GETC:IF NS!=0.0 THEN 171
172   NS=NS!-48.0:IF NS<2.0 OR NS>10.0 OR NS<>INT(NS) THEN 171
173   PRINT NS
180   F1$="C":PRINT "Door (G)ebruiker of (C)omputer"
181   R1!=GETC:IF R1!=0.0 THEN 181
182   R1$=CHR$(R1!)
190   IF LEFT$(R1$,1)="G" THEN 240
200   R$="N":PRINT "(N)ummers of (L)etters"
201   R!=GETC:IF R!=0.0 THEN 201
202   R$=CHR$(R!)
203   IF R$="L" OR R$="N" THEN 210
204   GOTO 201
210   FOR K=1 TO NS
220   IF LEFT$(R$,1)="L" THEN S$(K)=CHR$(RND(26.0)+65.0)
222   IF LEFT$(R$,1)="N" THEN S$(K)=CHR$(RND(10.0)+48.0)
230   NEXT:GOTO 250
240   GOSUB 1500:PRINT "maximaal 1 teken per item":FOR K=1 TO NS:PRINT "ITEM #";
K;" = ";
241   S!=GETC:IF S!=0.0 THEN 241
242   S$(K)=CHR$(S!):PRINT S$(K):NEXT
250   RF$="N":GOSUB 1500
260   RETURN
270   CURSOR 0,12:PRINT "duw een toets om verder te gaan"
271   IF GETC=0.0 THEN 271
290   CURSOR 0,12:PRINT SPC(31)
300   RETURN
310   CURSOR 9,23:PRINT "-BUBBLE SORT-":GOSUB 270
320   FOR I=1 TO NS-1
330   FOR J=I+1 TO NS
340   X=I:Y=J:GOSUB 710:IF A$(I)<=A$(J) THEN 360
350   GOSUB 750
360   NEXT J
370   NEXT I
380   CURSOR 0,13:PRINT " SORTEREN BEEINDIGD....":GOSUB 270
390   RETURN
400   CURSOR 3,23:PRINT "  -VERTRAAGDE SORTERING-":GOSUB 270
410   J=0:R=0:I=0
420   I=I+1
```

```
430   IF I=NS THEN 520
440   J=I:R=J+1
450   X=J:Y=R:GOSUB 710:IF A$(R)>=A$(J) THEN 470
460   J=R
470   R=R+1
480   IF R<=NS THEN 450
490   IF I=J THEN 420
500   GOSUB 750
510   GOTO 420
520   CURSOR 0,13:PRINT "SORTEREN BEEINDIGD...":GOSUB 270
530   RETURN
540   CURSOR 7,23:PRINT "-SHELL-METZNER SORT-":GOSUB 270
550   M=NS
560   M=INT(M/2.0)
570   IF M=0.0 THEN 690
580   P=NS-M
590   H=1
600   I=H
610   J=I+M
620   X=I:Y=J:GOSUB 710:IF A$(I)<=A$(J) THEN 660
630   GOSUB 750
640   I=I-M
650   IF I>=1 THEN 610
660   H=H+1
670   IF H>P THEN 560
680   GOTO 600
690   CURSOR 0,13:PRINT "SORTEREN BEEINDIGD....":GOSUB 270
700   RETURN
710   NC=NC+1:CURSOR 35,21:PRINT NC
720   CURSOR 5,23-X:PRINT CHR$(136):CURSOR 5,23-Y:PRINT CHR$(136):CURSOR 0,0:WAI
T TIME 50
730   CURSOR 5,23-X:PRINT " ":CURSOR 5,23-Y:PRINT " "
740   RETURN
750   FOR K=2 TO 8 STEP 2
760   CURSOR K,23-I:PRINT " ":CURSOR K+2,23-I:PRINT A$(I)
770   CURSOR K,23-J:PRINT " ":CURSOR K+2,23-J:PRINT A$(J)
780   WAIT TIME 5:NEXT K
790   CURSOR 10,23-I:PRINT " ":CURSOR 12,23-I:PRINT A$(I):DF=J-I
800   FOR K=1 TO DF
810   CURSOR 12,23-(I+K-1):PRINT " ":CURSOR 12,23-(I+K):PRINT A$(I)
820   CURSOR 0,0:WAIT TIME 20
830   CURSOR 10,23-(J-K+1):PRINT " ":CURSOR 10,23-(J-K):PRINT A$(J)
840   NEXT K
850   FOR K=12 TO 4 STEP -2
860   CURSOR K,23-J:PRINT " ":CURSOR (K-2),23-J:PRINT A$(I)
870   CURSOR K,23-I:PRINT " ":CURSOR (K-2),23-I:PRINT A$(J)
880   WAIT TIME 5:NEXT K
890   NE=NE+1:CURSOR 35,20:PRINT NE
900   TEMP$=A$(I):A$(I)=A$(J):A$(J)=TEMP$
910   RETURN
1499  GOTO 1499
1500  MODE 0:PRINT CHR$(12):FOR X!=#BFEF TO #BA2D STEP -#86:POKE X!,#6A:NEXT:RET
URN
```

DAI VIDEO HARDWARE

DAI - VIDEO hardware: messcherpe plaatjes in zwart-wit!

Aangezien het werken met de DAI in combinatie met een gewone TV een zeer vermoeiende bezigheid bleek, heb ik me nogal wat moeite getroost hierin verbetering te brengen.

De problemen waar het hier om gaat treden vooral op bij het verwerken van tekst (ontwikkelen van programma's). Als hoofdschuldige moet de karaktergrootte worden aangemerkt: 64 tekens per regel is gewoon teveel voor een normale TV. Verder bleken die karakters niet zuiver stil te staan. Bovendien hadden de letters last van allerlei "rafelige" uitwaaiers vooral bij het gebruik van verkeerde kleuren. Wie een beeld voltikt met dezelfde letters kan 'genieten' van een diagonaal en lopend strepenpatroon. Het totaal geeft ongeveer de indruk van een plaatje waar een watergordijn voor langs stroomt.

Overigens: bij een spelletje 'Spaceinvader' heb ik geen enkel probleem met de beeldkwaliteit. Vooral bij het ontwikkelen van programma's is echter uiterste concentratie vereist. De extra belasting van de hierboven beschreven matige beeldkwaliteit zal m.i. elke serieuze programmeur spoedig de hoop doen opgeven: hij/zij ziet letterlijk door de bomen het bos niet meer !!!

Om een lang verhaal kort te maken: het is uiteindelijk gelukt een zeer bevredigende èn betaalbare beeldkwaliteit te krijgen m.b.v. een zwart-wit monitor en enige eenvoudige wijzigingen aan de DAI-print. Voordat ik die ga beschrijven wil ik nog graag in het kort mijn niet-succesvolle pogingen verwoorden:

-K.T.V. gebruiken als monitor. Ik had tot mijn beschikking een Blaupunkt met FM 121 chassis. Op het schema vond ik het IC TDA 3300. Deze heeft extra RGB-aansluitingen (pen 24, 25, 26), met pen 23 als vermoedelijke schakelaar.
Tot op heden heb ik echter niemand kunnen vinden die me op dit punt verder kan helpen (ik ben geen electronicus); ik ben er echter bijna zeker van dat via dit IC een eenvoudige RGB-aansluiting te realiseren moet zijn.
In dit verband denk ik dat er vele computerhobbyisten zijn die hun K.T.V. van een monitor aansluiting willen voorzien: een gat in de markt dus. Wie zich erin wil werpen kan ik in ieder geval helpen aan het FM 121 schema...

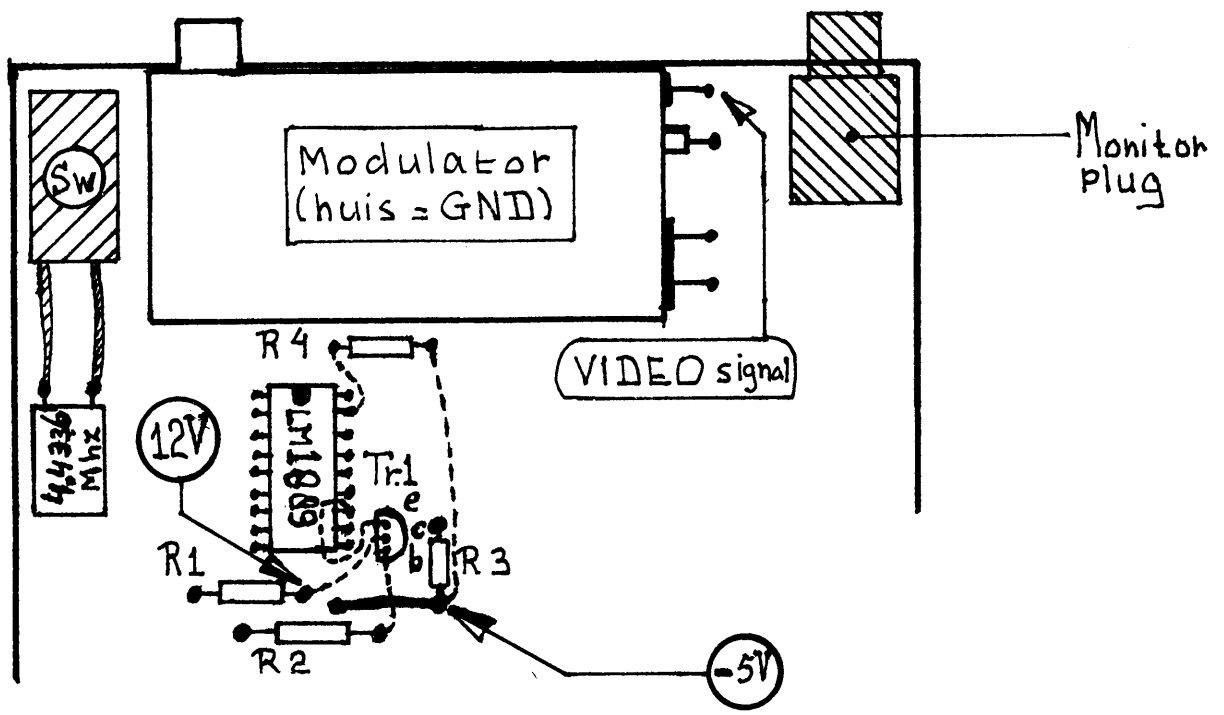
-DAI-video kaart verbeteren.
Als voorbeeld hiervoor gebruikte ik het verhaal van Anton Doornenbal (DAInamic maart/juni 1982). Als eerste werd de ZNA 134 aansluiting gewijzigd (interliniering). Dat gaf een kleine verbetering.
Vervolgens de zaken die verband houden met bandbreedte aangepakt (kleur en helderheid). Het bleek dat kleur- en helderheidsbandbreedte niet afzonderlijk konden worden gewijzigd. Hoewel ik me bij Anton persoonlijk heb kunnen overtuigen van de prima beeldkwaliteit van zijn installatie, ben ik van verbetering in mijn eigen geval niet zo zeker. De twijfel is ook weer niet zo groot dat ik de zaak weer in de oorspronkelijke toestand terugbreng...

-Gebruik zwart-wit monitor. Een demonstratie bij een handelaar, waarbij het signaal afgetakt werd van de modulator leverde niet het monitor beeld zoals ik dat kende van andere computers.

Over het beeld liep een diagonaal patroon van lijnen, welke door contrastverhoging enigszins te onderdrukken waren. Hoewel men probeerde de indruk te wekken dat dit ècht normaal en ik een beetje abnormaal was ging de koop dus niet door... Toch kocht ik later tweedehands zo'n kubus uit Hong-Kong; voor honderd gulden: dat kùn je niet weigeren!
Teneinde het geheel goed te laten samenwerken ging ik te rade bij Anton Doornenbal. De hierna weergegeven zwart-wit monitoraansluitingen komen dan ook van zijn hand. Beide werken ze perfect. Hoewel tevens de eerder genoemde wijzigingen interliniering/helderheid/kleur zijn doorgevoerd zijn die toch van ondergeschikt belang.
Het lopende diagonaallijnenpatroon bleek te worden veroorzaakt door een kleurpuls welke de kleurenontvanger doet omschakelen op kleurenweergave. Dit signaal is bij een zwart-wit monitor uiteraard overbodig, ja blijkt zelfs uitermate storend te werken.

ZWART-WIT MONITOR AANSLUITINGEN

Onderstaande figuur toont de PAL-videokaart met enkele van de belangrijkste componenten. Ik hoop dat ook de minder ervaren knutselaars hiermee uit de voeten kunnen.

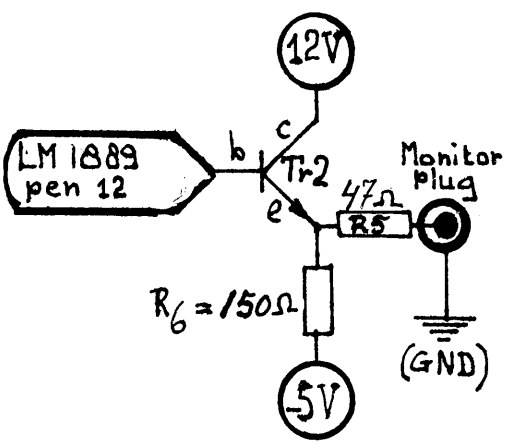


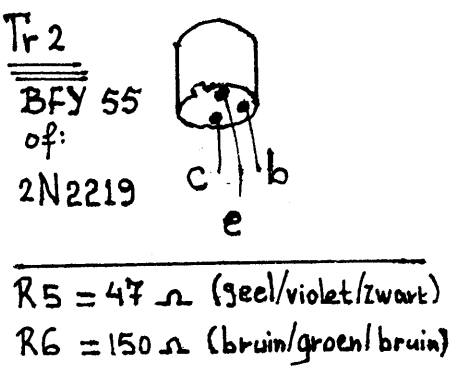
R1 = 820 Ω (grijs/rood/bruin)
R2 = 1k5 (bruin/groen/rood)
R3 = 1 kΩ (bruin/zwart/rood)
R4 = 2k7 (rood/violet/rood)
Tr1 = 2N 3704

1-SUPERSIMPEL-aansluiting

Installeer de monitorplug op de (gearceerd) aangegeven plaats. Verbind het modulatorhuis (=GND) met de buiten kant van de plug. De binnenkant van de plug wordt verbonden met het punt "VIDEOsignal". Dit is de 'normale' monitoraansluiting.
De gewraakte kleurpuls is gesitueerd rond het kristal 4,4336 Mhz. De puls wordt effectief gesmoord door dit kristal eenvoudigweg kort te sluiten. Maak de bedrading absoluut niet langer dan nodig is, daar het kristal anders zou kunnen worden beschadigd!
In de figuur is de kortsluiting uitschakelbaar gemaakt d.m.v. een lampdrukknopschakelaartje (Sw). Via een gaatje in het deksel van de DAI kan deze m.b.v. een potlood o.i.d. bediend worden. K.T.V. en zw-monitor mogen ook samen gebruikt worden. Echter met de schakelaar geopend is de kleurpuls aanwezig, en dus de monitor weergave niet optimaal; met gesloten schakelaar verdwijnt de kleurpuls en levert de K.T.V. slechts zwart-wit beelden. Dit is dus een nadeel.
De volgende oplossing laat volledig onafhankelijk gebruik van K.T.V. en zw-monitor toe.

2-SUPERDELUX-aansluiting





Hier blijft het kristal ongemoeid. De monitor plug wordt nu m.b.v. een stukje afgeschermde kabel verbonden met een transistortrapje. Deze transistor met de twee weerstanden werden aan de soldeerzijde ongeveer onder Tr. 1 geplaatst. De nieuwe transistor is een BFY 55; een 2N 2219 voldoet ook. De mantel van de signaal kabel naar de monitor plug wordt zo dicht mogelijk bij de transistor op een GND-punt gesoldeerd. We hebben nu een nette aftakking van het nog 'onbedorven' videosignaal gemaakt en dit op het voor de monitor vereiste niveau van 1 Volt gebracht.
Een aardige tip voor wie meer wil weten over de werking van een dergelijke videokaart: kijk eens in Elektuur nr 231 (jan 1983). Hierin wordt een VideoAudioModulator beschreven, waarin de LM1889 is gebruikt.

Geert Hospers (met dank aan A. Doornenbal)
Hoffmannlaan 555
5011 WL Tilburg (NL) tel.013-561844

CONTINUATION LINES

SCREEN CONTINUATION LINES

The DAI monitorprogram enables the use of extended lines on the screen. Up to 3 extended lines are possible; each line is indented 6 spaces and a continuation 'C' is placed at its beginning.

Mr. Markus Sigg, Worblingen, Germany, designed the following ML routine to manipulate with these extended lines.
It enables the use of more than 3 extended lines, it cancels the continuation 'C' and it enables the use of more or less spaces on the extended lines.

```
INIT        PUSH H
            LXI H,START         addr. alternative routine
            SHLD :006C          load RST5 vectoraddr.
            POP H
            RET

START       PUSH H              On entry, A contains last char
            PUSH PSW
            CPI :0D             Car.ret?
            JZ READY            Then abort
            LDA :0072           Get lsbyte cursorpos.
            LHLD :007A          Get lsbyte last pos on line
            CMP L               End of line reached?
            JNZ READY           Abort when not
            CALL :DD5E          Else: print car.ret

SPACE       MVI A,:20           Load Ascii 'space'
            MVI H,:06           Nr. of indented spaces
OUTC        RST5                Output to screen
            DATA :03
            DCR H
            JNZ OUTC            Next space when not ready

READY       POP PSW
            POP H
            JMP :C6FD           Perform original RST5
```

The INIT routine must be performed once at the beginning of the program. It replaces the RST5 vector address on #006C/D with with the address of the alternative routine. [This can also be done under Utility with V5=START].

The rest of the routine will be runned by the DAI monitor program every time a RST5 (screen handling) routine is performed.
If no indent is required, the part 'SPACE' can be skipped.
The number of spaces on the indented lines can be changed by changing the contents of the H-register in the 'SPACE'-routine.
After this routine is performed, the original RST5 routine on address #C6FD is performed.

This alternative routine has no influence on the RS232 output.

Jan Boerrigter

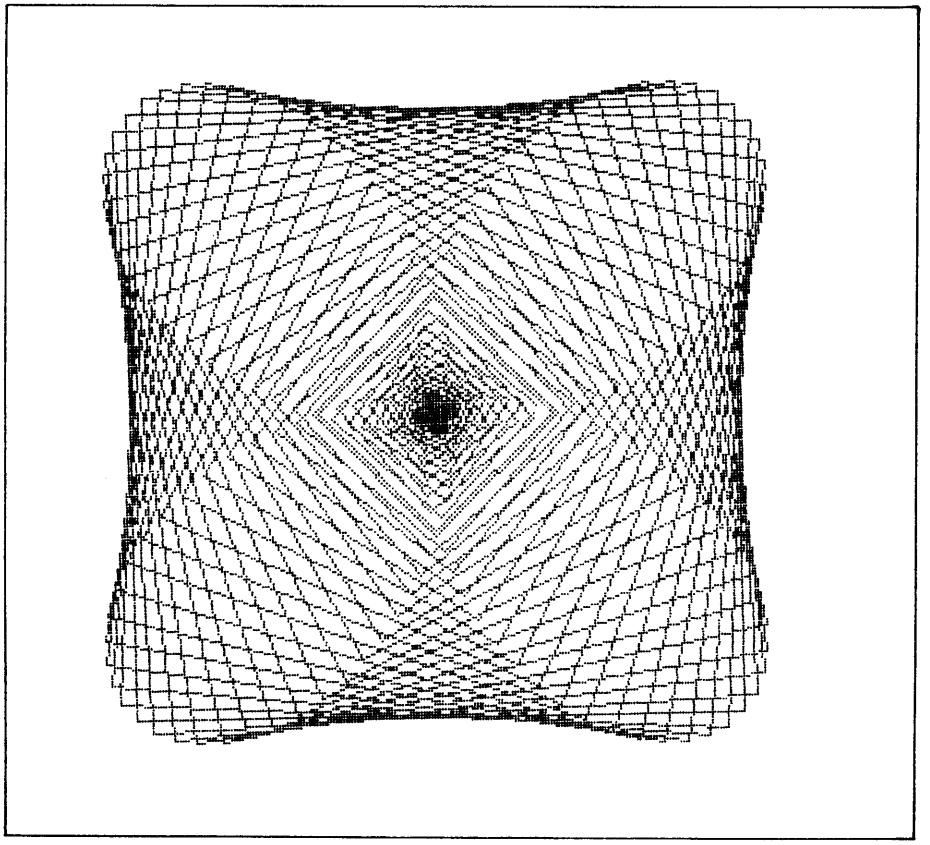
program identification

title	:	SCREEN TO BUFFER / BUFFER TO SCREEN
author	:	W.Hermans
purpose	:	To hold an entire picture in buffer,and restore it
comment	:	actual program is for MODE 3/4, could be adapted for other MODES(lines 7/8/9 & 28/29/30)

```
001                             ORG   :0D00
002                     *MOVE SCREEN TO BUFFER
003 0D00 E5                     PUSH  H
004 0D01 C5                     PUSH  B
005 0D02 D5                     PUSH  D
006 0D03 F5                     PUSH  PSW
007 0D04 112E17                 LXI   D,5934     : TIMES TO MOVE
008 0D07 21EFBF                 LXI   H,:BFEF    : SOURCE
009 0D0A 010090                 LXI   B,:9000    : DESTINATION
010 0D0D 7E             CONT    MOV   A,M
011 0D0E 02                     STAX  B
012 0D0F 2B                     DCX   H
013 0D10 0B                     DCX   B
014 0D11 1B                     DCX   D
015 0D12 7A                     MOV   A,D
016 0D13 B3                     ORA   E
017 0D14 C20D0D                 JNZ   CONT
018 0D17 F1                     POP   PSW
019 0D18 D1                     POP   D
020 0D19 C1                     POP   B
021 0D1A E1                     POP   H
022 0D1B C9                     RET
023                     *MOVE  BUFFER TO SCREEN
024 0D1C E5                     PUSH  H
025 0D1D C5                     PUSH  B
026 0D1E D5                     PUSH  D
027 0D1F F5                     PUSH  PSW
028 0D20 112E17                 LXI   D,5934     : TIMES TO MOVE
029 0D23 210090                 LXI   H,:9000    :SOURCE
030 0D26 01EFBF                 LXI   B,:BFEF    :DESTINATION
031 0D29 7E             CONT2   MOV   A,M
032 0D2A 02                     STAX  B
033 0D2B 2B                     DCX   H
034 0D2C 0B                     DCX   B
035 0D2D 1B                     DCX   D
036 0D2E 7A                     MOV   A,D
037 0D2F B3                     ORA   E
038 0D30 C2290D                 JNZ   CONT2
039 0D33 F1                     POP   PSW
040 0D34 D1                     POP   D
041 0D35 C1                     POP   B
042 0D36 E1                     POP   H
043 0D37 C9                     RET
044 0D38                        END
```

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

CONT   0D0D   CONT2  0D29
```



2-D TRANSFORMATIONS

```
10    REM 2D TRANSFORMATIONS.
20    CLEAR 256:MODE 0
50    DIM CO!(5.0,2.0):REM CO(i,1)=Xi, CO(i,2)=Yi
51    FOR I=1 TO 5:FOR J=1 TO 2:READ CO!(I,J):NEXT:NEXT
52    DATA 1,1,-1,1,-1,-1,1,-1,0,0
60    S!=1.318:REM .....................SCALING
62    R!=2.82:RAD!=180.0/PI
63    CS!=COS(R!/RAD!):SN!=SIN(R!/RAD!):REM ..ROTATION
64    XT!=168.0:YT!=128.0:REM ............TRANSLATION
100   COLORG 8 0 5 15:MODE 6:SF!=(-1E-2)
110   FOR N=1 TO 64
120   REM PLOT
122   CO!(5.0,1.0)=CO!(1.0,1.0):CO!(5.0,2.0)=CO!(1.0,2.0)
124   FOR I=1 TO 4:J=I+1
126   DRAW CO!(I,1.0)+XT!,CO!(I,2.0)+YT! CO!(J,1.0)+XT!,CO!(J,2.0)+YT! 0:NE
XT
130   REM ROTATION
131   FOR I=1 TO 4
132   X1!=CO!(I,1.0)*CS!+CO!(I,2.0)*SN!
133   Y1!=CO!(I,2.0)*CS!-CO!(I,1.0)*SN!
134   CO!(I,1.0)=X1!:CO!(I,2.0)=Y1!:NEXT
140   REM SCALING
142   FOR I=1 TO 4:FOR J=1 TO 2:CO!(I,J)=CO!(I,J)*S!:NEXT:NEXT
150   S!=S!+SF!:IF S!<0.0 THEN SF!=1E-2
199   NEXT
999   END
```

***** N I E U W E T E C H N I S C H E O N T W I K K E L I N G E N *****

In het Info-bulletin van de Nascom gebruikers club vonden wij een toch ook voor DAI-gebruikers interessant artikel over nieuwe hardware ontwikkelingen. Met medeweten en toestemming van deze club en auteur nemen wij het hier over.

De WOM

Dit is een nieuw geheugen IC waarmee een revolutie is ontketend in de wereld van de solid state memory's. Al direct na de aankondiging van dit nieuwe IC door de japanse fabrikant Jijusti plaatste IBM een bestelling van 10 miljoen dollar. Het gevolg is, dat dit IC waarschijnlijk pas eind 1983 op de Europese markt beschikbaar zal komen. Vermoedelijk zal het voor de hobbymarkt nog wel wat langer gaan duren. Wat is er nu zo bijzonder aan deze memorychip, dat zelfs IBM zo gretig heeft gereageerd ?

Welnu, ten eerste de prijs. De 16K x 8 versie zal, voor zover het zich nu laat aanzien, zeker onder de een dollar gaan komen. De momenteel nog in ontwikkeling zijnde 64K x 8 versie zal de prijs van 3 dollar niet gaan overschrijden.

Ten tweede de specifcatie's. De 16K x 8 versie is pin-compatible met de 2764 eprom. De data-sheets geven aan dat het hier gaat om een single 5 volt memory-chip met in de huidig beschikbare versie een accesstime van 50 nano-seconde zodat deze IC's uitermate geschikt zijn voor toepassingen in combinatie met snelle of zeer snelle proseccors. Naar het zich laat aanzien, zal deze nieuwe ontwikkeling een grote invloed gaan hebben op de toepassing van Winchester disk systemen. Solid state blijft namelijk altijd betrouwbaarder dan een systeem met mechanische (dwz bewegende) componenten. Geheugenbanken van vele Mega bytes op een printplaat zijn nu eenvoudig te verwezelijken en zullen grote concurrenten gaan worden voor de hard-disk systemen.

Ten derde de power dissipatie. Indien geselecteerd neemt de 16K x 8 versie slechts 85 milliwatt op en in de mode -niet geselecteerd- daalt het verbruik zelfs tot 15 milliwatt. Ook deze gegevens wijzen op de mogelijkheid van grote geheugenbanken met een minimum aan warmteontwikkeling.

Hoe is dit nu allemaal mogelijk ? Welnu,voor zover het te lezen was in de specificaties zijn een aantal nieuwe vindingen gecombineerd op zodanige wijze dat bovenstaande resultaten een feit werden. Deze vindingen betreffen ondermeer:

- een op de chip uitgevoerde AC/AC converter
- een schuifregister volgens het FINO-principe
- twee schuifregisters volgens het FISO-principe
- een parallelle input die intern bi-directioneel wordt gecomplementeerd
- een refresh systeem gebaseerd op het 4711 principe
- een non-directionele interne bus structuur
- een a-synchrone zichzelf continu corrigerende baudrate

De toepassing van de WOM wordt vooral gezien in systemen waarbij data-opslag voor langere tijd en in grote kwantiteit noodzakelijk is.

Gebruikte afkortingen: FINO - First In ,Never Out
FISO - First In ,Sometimes Out
4711 - bekend Keuls patent
WOM - Write Only Memory

Tot zover dit nieuwtje over de WOM. Ik zal trachten iedereen op de hoogte te houden over nieuwe ontwikkelingen op dit gebied en nodig alle leden uit om bijdragen hierover aan de redactie op te sturen.

FIAT

```
100   REM
110   REM
120   REM
130   REM SOMEBODY NEEDS A FIAT MARK ?
140   REM
150   REM
160   REM ... Gianni Uliana
170   REM
180   REM
190   REM
200   REM
10010 COLORG 14 0 0 0
10015 MODE 5
10020 FILL 55,90 290,150 0
10040 DRAW 96,90 111,150 14
10050 DRAW 97,90 112,150 14
10060 DRAW 98,90 113,150 14
10070 DRAW 155,90 172,150 14
10080 DRAW 156,90 173,150 14
10090 DRAW 157,90 174,150 14
10100 DRAW 215,90 231,150 14
10110 DRAW 216,90 232,150 14
10120 DRAW 217,90 233,150 14
10130 FOR I=40.0 TO 55.0
10131 DRAW I,90 I+15,150 0:NEXT
10140 FOR I=275.0 TO 290.0
10141 DRAW I,90 I+15,150 14:NEXT
10149 REM F
10150 FOR I=55.0 TO 60.0
10151 DRAW I,105 I+10,140 14:NEXT
10152 DRAW 66,140 92,140 14:DRAW 66,139 92,139 14
10153 DRAW 66,138 91,138 14:DRAW 66,137 91,137 14
10154 DRAW 65,125 86,125 14:DRAW 65,124 86,124 14
10155 DRAW 65,123 85,123 14:DRAW 65,122 85,122 14
10159 REM I
10160 FOR I=125.0 TO 130.0
10161 DRAW I,103 I+10,140 14:NEXT
10169 REM A
10170 FOR I=195.0 TO 200.0
10180 DRAW I-25,103 I,140 14:DRAW I+10,103 I,140 14:NEXT
10181 FOR I=115.0 TO 119.0
10182 DRAW 182,I 203,I 14:NEXT
10189 REM T
10190 FOR I=245.0 TO 250.0
10191 DRAW I,103 I+10,140 14:NEXT
10195 DRAW 240,136 271,136 14
10196 DRAW 240,137 271,137 14
10197 DRAW 241,138 272,138 14
10198 DRAW 241,139 272,139 14
10199 DRAW 241,140 272,140 14
10200 WAIT TIME 100:GOTO 10020
```

FIAT

ON ERROR GOTO

```
SPL V1.1  PAGE 1

0000                    TITL      'ON ERROR GOTO V2.0'
0000          ;To activate:
0000          ;*UT <RETURN>
0000          ;>V5 C6FD-300 <CURSOR LEFT>
0000          ;>B
0000          ;*POKE #6,LINENUMBER IAND #FF:POKE #7,LINENUMBER SHR 8
0000          ;To deactivate:
0000          ;*POKE #6,0:POKE #7,0
0000          ;-------------------------------------------------------------------
0000          ; STACK DURING ERRORHANDLING
0000          ;
0000          ;When an error occurs a jump is performed to the errorprint-
0000          ;routine which goes via ODA50H. This routine is as follows:
0000          ;
0000          ;           ADDRES          STACKP    TOS         ROUTINE
0000          ;ODA3DH     CALL ODA50H     SP+2H     ODA40H      Runtime error
0000          ;    Stackbase for all errors;
0000          ;ODA50H     CALL ODD55H     SP-00H    ODA53H      Print CR before
0000          ;ODD55H     PUSH D          SP-02H    DE          errormsg is
0000          ;ODD56H     PUSH H          SP-04H    HL          printed.
0000          ;ODD57H     RST5
0000          ;ODD58H     DB   0CH        SP-06H    ODAE1H      get CURX,CURY
0000          ;0028H      NOP
0000          ;0029H      PUSH H          SP-08H    HL          Change to ROMbank
0000          ;002AH      LHLD 6CH                              2 6C=300H
0000          ;002DH      PCHL
0000          ;0300H      PUSH PSW        SP-0AH    PSW         On error routine
0000          ;
0000          ;-------------------------------------------------------------------
0000  @=0006 ERRLIN  EQU       6H              ;Free bytes after RST 0
0000          ;
0000                  ORG       300H
0300  F5              PUSH PSW                  ;Save PSW
0301          ;                                 ;HL may be corrupted because
0301          ;                                 ;contents is already on stack.
0301  2A0600          LHLD      ERRLIN          ;Check if line to goto
0304  7C              MOV A,H                   ;on error
0305  B5              ORA L
0306  CA2103          JZ        OUT             ;quit if 0
0309          ;
0309  210A00          LXI H     0AH             ;Offset to stackbase
030C  39              DAD SP                    ;Add stackpointer to find
030D    '     ;                                 ;original caller
030D  7E              MOV A,M                   ;check if caller is ODA53H
030E  FE53            CPI       53H
0310  C22103          JNZ       OUT             ;if not continu RST 5
0313  23              INX H
0314  7E              MOV A,M
0315  FEDA            CPI       0DAH
0317  C22103          JNZ       OUT             ;if caller is NOT ODA50H
031A  23              INX H                     ;Check if runtime error
031B  7E              MOV A,M
031C  FE40            CPI       40H
031E  CA2503          JZ        ONERR           ;If no running program
0321  F1      OUT     POP PSW                   ;restore stack continu
0322  C3FDC6          JMP       0C6FDH          ;with normal RST 5
0325          ;
0325  2A0600  ONERR   LHLD      ERRLIN          ;Get linenumber to goto
```

ON ERROR GOTO

```
SPL V1.1  PAGE 2      ON ERROR GOTO V2.0

0328            ;                                   ;in HL
0328  CDF6CA              CALL        0CAF6H        ;find this line in BASIC
032B  D24D03              JNC         UNDEFL        ;text buffer, if undefined
032E            ;                                   ;line run error
032E  44                  MOV B,H                   ;BC start line in BASIC
032F  4D                  MOV C,L                   ;text buffer
0330  210001              LXI H       100H          ;start of basic pointers
0333  3E15                MVI A       15H           ;in HL, clear evt.GOSUB and
0335  3600    LOOP       MVI M       0H            ;FOR NEXT levels
0337  23                  INX H
0338  3D                  DCR A
0339  C23503              JNZ         LOOP          ;loop till ready
033C  F3                  DI
033D  323101              STA         131H          ;Output to screen
0340  324000              STA         40H           ;Force ROMbank 0
0343  3206FD              STA         0FD06H
0346  FB                  EI
0347  3100F9              LXI SP      0F900H        ;reset stack pointer continu
034A  C392C8              JMP         0C892H        ;with BASIC execution
034D            ;
034D  210000    UNDEFL    LXI H       0H            ;If the lineno in ERRLIN does
0350  220600              SHLD        ERRLIN        ;not exist errors are enabled.
0353  3E04                MVI A       4H            ;code for UNDEFINED LINENUMBER
0355  C3F5D9              JMP         0D9F5H        ;print this message. Back to
0358            ;                                   ;BASIC monitor.
0358                      END
```

```
0300 F5 2A 06 00 7C B5 CA 21 03 21 0A 00 39 7E FE 53
0310 C2 21 03 23 7E FE DA C2 21 03 23 7E FE 40 CA 25
0320 03 F1 C3 FD C6 2A 06 00 CD F6 CA D2 4D 03 44 4D
0330 21 00 01 3E 15 36 00 23 3D C2 35 03 F3 32 31 01
0340 32 40 00 32 06 FD FB 31 00 F9 C3 92 C8 21 00 00
0350 22 06 00 3E 04 C3 F5 D9 00 00 00 00 00 00 00 00
```

```
1     REM ON ERROR GOTO DEMO/N.P. LOOIJE
2     REM all levels of FOR NEXT/GOSUB will be cleared
10    MODE 0:PRINT CHR$(12)
20    POKE 6,10:POKE 7,0:REM ON ERROR GOTO 10
30    A=A+1:PRINT A:IF A>10 THEN 50:REM 10 times ON ERROR
40    DOT A,B C:REM not possible in MODE 0
50    POKE 6,0:POKE 7,0:REM ON ERROR off
60    GOTO 40:REM now an errormessage will be printed
```

UPPER TO LOWER CASE

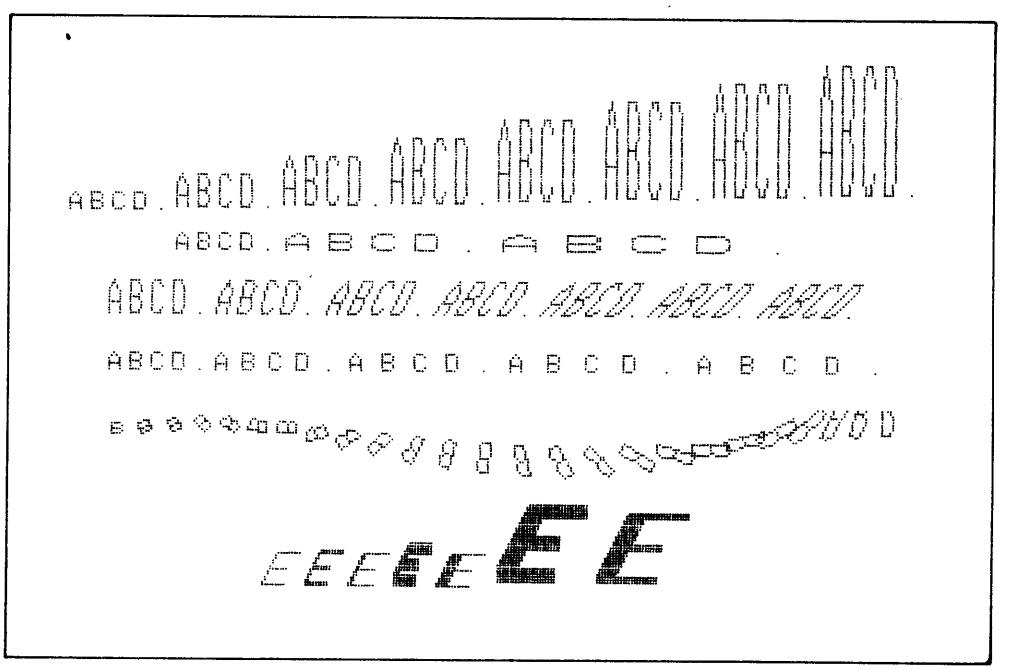
```
002                     *
003                     *
004                     * CONVERTS UPPER CASE CHARACTERS TO LOWER
005                     * CASE WHEN PLACED IN DATA STATEMENTS.
006                     *
007                     * HL: ADDRESS IN TEXTBUFFER
008                     * DE: END TEXTBUFFER
009                     * BC: OFFSET OF START NEXT TEXTLINE
010                     *  C: NR OF DATABYTES TO BE CHECKED
011                     *
012                             ORG    :400
013                     *
014 0400 F5             INIT    PUSH   PSW
015 0401 C5                     PUSH   B
016 0402 D5                     PUSH   D
017 0403 E5                     PUSH   H
018 0404 2AA102                 LHLD   :02A1        GET STBBGN
019 0407 EB                     XCHG
020 0408 2A9F02                 LHLD   :029F        GET TXTBGN
021 040B 0600                   MVI    B,:00
022                     *
023 040D CD14DE         LNCNT   CALL   :DE14        COMPARE HL-DE
024 0410 D23B04                 JNC    READY        ) ABORT WHEN
025 0413 CA3B04                 JZ     READY        ) FINISHED
026                     *
027 0416 4E             LKDATA  MOV    C,M          LENGTH TEXTLINE IN C
028 0417 23                     INX    H
029 0418 23                     INX    H
030 0419 23                     INX    H            POINTS TO TOKEN
031 041A 7E                     MOV    A,M          GET TOKEN
032 041B FEA2                   CPI    :A2          DATA STATEMENT?
033 041D CA2604                 JZ     DATA
034 0420 2B                     DCX    H
035 0421 2B                     DCX    H
036 0422 09             NEXTLN  DAD    B            GET ADDR NEXT TEXTLINE
037 0423 C30D04                 JMP    LNCNT
038                     *
039 0426 23             DATA    INX    H            POINTS TO LENGTH DATA
040 0427 4E                     MOV    C,M
041 0428 0C                     INR    C
042 0429 23             NXTBYT  INX    H            ) UPDATE ADDR +
043 042A 0D                     DCR    C            ) LENGTH POINTERS
044 042B CA0D04                 JZ     LNCNT        NEXT LINE WHEN DONE
045 042E 7E                     MOV    A,M          GET BYTE
046 042F CD02DE                 CALL   :DE02        CHECK IF UPPER CASE CHAR
047 0432 D22904                 JNC    NXTBYT
048 0435 C620           CHANGE  ADI    :20          CHANGE TO LOWER CASE
049 0437 77                     MOV    M,A
050 0438 C32904                 JMP    NXTBYT
051                     *
052 043B E1             READY   POP    H
053 043C D1                     POP    D
054 043D C1                     POP    B
055 043E F1                     POP    PSW
056 043F C9                     RET
057                     *
058 0440                        END
```

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

CHANGE 0435   DATA   0426   INIT   0400   LKDATA 0416
LNCNT  040D   NEXTLN 0422   NXTBYT 0429   READY  043B
```

```
1     REM GRAFTEXT, MAAKT GEBRUIK VAN EEN FGT-TABEL.
2     COLORG 8 0 0 0:MODE 5A:COLORG 8 0 0 0
5     GTCS$="GRAFTEXT":C=0:X0=60:Y0=0:H!=6.0:W!=3.0:G!=0.0:R!=45.0:S!=45.0:
TH=1:TW=6:GOSUB 40000
6     WAIT TIME 100:MODE 5A       ***** FIRST LOAD FGT *****
9     GTCS$="ABCD."
10    X0=5:Y0=150:W!=1.0:G!=0.0:R!=0.0:S!=0.0:TH=0:TW=0
11    FOR H!=1.0 TO 8.0:GOSUB 40000:NEXT
12    X0=45:Y0=135:H!=1.0:FOR W!=1.0 TO 3.0:GOSUB 40000:NEXT
13    X0=20:Y0=110:H!=2.0:W!=1.0:FOR S!=0.0 TO 90.0 STEP 15.0:GOSUB 40000:N
EXT
14    X0=20:Y0=90:H!=1.0:W!=1.0:S!=0.0:FOR G!=0.0 TO 8.0 STEP 2.0:GOSUB 400
00:NEXT
15    X=25:X0=X:Y0=60:H!=0.6:W!=1.0:S!=0.0:G!=0.0:GTCS$="B"
16    FOR R!=0.0 TO 360.0 STEP 15.0:GOSUB 40000:X=X+12:X0=X:Y0=60:H!=H!+0.1
:NEXT
17    X0=80:Y0=2:H!=2.0:W!=2.0:S!=30.0:R!=0.0:G!=0.0:GTCS$="E":GOSUB 40000
18    TH=1:TW=0:GOSUB 40000:TH=0:TW=1:GOSUB 40000
19    TH=4:TW=0:GOSUB 40000:TH=0:TW=4:GOSUB 40000
20    H!=4.0:W!=4.0:TH=5:TW=5:GOSUB 40000:TH=2:TW=6:GOSUB 40000
99    END
999   END
40000 REM GRAF TEXT
40002 RAD!=57.2958:HS!=H!:SW!=W!:SG!=G!/SW!
40004 HX!=COS(R!/RAD!):HY!=SIN(R!/RAD!)
40008 VX!=-HY!+HX!*SIN(S!/RAD!):VY!=HX!+HY!*SIN(S!/RAD!)
40010 DHX!=HX!*SW!:DHY!=HY!*SW!:DVX!=VX!*HS!:DVY!=VY!*HS!
40014 DSX!=DHX!*(8.0+SG!):DSY!=DHY!*(8.0+SG!)
40100 FOR M=0 TO LEN(GTCS$)-1:GTC=ASC(MID$(GTCS$,M,1)):GTI=#580
40120 GTT1=PEEK(GTI):GTT2=PEEK(GTI+1):IF GTT1=#5A AND GTT2=#FF GOTO 40140
40122 IF GTT1<>GTC THEN GTI=GTI+2*GTT2+2:GOTO 40120
40130 GTI=GTI+2
40131 FOR N=GTI TO GTI+2*(GTT2-1) STEP 2
40132 GTX=(PEEK(N) IAND #F0) SHR 4:GTY=PEEK(N) IAND #F
40133 GTX1=X0+GTY*DVX!+GTX*DHX!:GTY1=Y0+GTY*DVY!+GTX*DHY!
40135 GTX=(PEEK(N+1) IAND #F0) SHR 4:GTY=PEEK(N+1) IAND #F
40136 GTX2=X0+GTY*DVX!+GTX*DHX!:GTY2=Y0+GTY*DVY!+GTX*DHY!
40137 FOR T1=0 TO TW:FOR T2=0 TO TH
40138 DRAW GTX1+T1,GTY1+T2 GTX2+T1,GTY2+T2 C:NEXT:NEXT
40139 NEXT:X0=X0+DSX!+TW:Y0=Y0+DSY!
40140 NEXT
40150 RETURN
```

GRAFTEXT



RANDOM-ACCESS

Tessenderlo 27 april, 1983

Dear Dai-Diskunit-owners,

As owner of a dual Disk-drive of the firm Dai I noticed that to complete the Disk Operating System there was need for a Random Acces posibility.

Thanks to the artikel in Newsletter 14 I developed a method that is close to the original Random acces feature.
When we read a sector from disk, this sector will occupy 128 bytes in memory adressed by the HL register.When the content of the HL register is a VARPTR from a string with a length of 128 the information from the sector is stored in this string.We only have to set the length of the string,define which track and sector we need and call a routine.Upon return the string contains the sectors information.The same applys for writing a sector to disk.The only important thing is the length of the string.It always must be 128 bytes.When not you can destroy parts of other variables.
By now you will ask: How do I now which track and sector I have to poke ? To get this information we look in the directory. Whe therefore use the ASCII-FILE method used in the Auto-Menu on ower system disk.Here are the results:

```
Example:   FILENAMEBAS-----N####
           11111111222333345677
```

1: 8 Bytes for the filename.
2: 3 Bytes for the extension.
3: 5 Bytes not used.
4: 1 byte for the attribute
5: 8 bit number for first track of file.
6: 8 bit number for first sector of file.
7: 16 bit number for length of file

With the statement ASC() we retrieve the desired information. Multiply the track with #12 and add the sectors to it.Then assign the value to the variable OFFSET.With the variable P you can choose which sector from the file you want to acces. Caution: P=0 will return the first sector. RUN 900 reads a sector into R$, RUN 950 writes R$ to disk.R$ must always be If you want to use drive :0 change line 7 and 22 into:
MVI A,:30.

Good luck !

Couwberghs Frans
Boekdonkstraat 13
3980 Tesenderlo
013/666340

RANDOM-ACCESS

```
1010  PRINT "   PAGE 01    RANDOM GET PUT"
1070  PRINT
1080  PRINT "   002                          ORG   :19CD"
1090  PRINT "   003 19CD F5                  PUSH  PSW"
1100  PRINT "   004 19CE C5                  PUSH  B"
1110  PRINT "   005 19CF D5                  PUSH  D"
1120  PRINT "   006 19D0 E5                  PUSH  H"
1130  PRINT "   007 19D1 3E31                MVI   A,:31"
1140  PRINT "   008 19D3 0600                MVI   B,0"
1150  PRINT "   009 19D5 0E00                MVI   C,0"
1160  PRINT "   010 19D7 210000              LXI   H,0"
1170  PRINT "   011 19DA 23                  INX   H"
1180  PRINT "   012 19DB CD1C0A              CALL  :A1C"
1190  PRINT "   013 19DE E1                  POP   H"
1200  PRINT "   014 19DF D1                  POP   D"
1210  PRINT "   015 19E0 C1                  POP   B"
1220  PRINT "   016 19E1 F1                  POP   PSW"
1230  PRINT "   017 19E2 C9                  RET"
1240  PRINT "   018 19E3 F5                  PUSH  PSW"
1250  PRINT "   019 19E4 C5                  PUSH  B"
1260  PRINT "   020 19E5 D5                  PUSH  D"
1270  PRINT "   021 19E6 E5                  PUSH  H"
1280  PRINT "   022 19E7 3E31                MVI   A,:31"
1290  PRINT "   023 19E9 0600                MVI   B,0"
1300  PRINT "   024 19EB 0E00                MVI   C,0"
1310  PRINT "   025 19ED 210000              LXI   H,0"
1320  PRINT "   026 19F0 23                  INX   H"
1330  PRINT "   027 19F1 CD320A              CALL  :A32"
1340  PRINT "   028 19F4 E1                  POP   H"
1350  PRINT "   029 19F5 D1                  POP   D"
1360  PRINT "   030 19F6 C1                  POP   B"
1370  PRINT "   031 19F7 F1                  POP   PSW"
1380  PRINT "   032 19F8 C9                  RET"
1390  PRINT "   033 19F9                     END"
1400  PRINT
1410  PRINT
1420  PRINT "900   REM GET A SECTOR FROM DISK"
1430  PRINT "910   TRACK=(P+OFFSET)/#12:SECTOR=(P+OFFSET) MOD #12"
1440  PRINT "920   IF SECTOR=0 THEN SECTOR=18:TRACK=TRACK-1"
1450  PRINT "930   POKE #19D4,TRACK:POKE #19D6,SECTOR:POKE #19D8,PEEK(VARPT
R(R$))"
1460  PRINT "950   REM PUT A SECTOR ON DISK"
1470  PRINT "960   TRACK=(P+OFFSET)/#12:SECTOR=(P+OFFSET) MOD #12"
1480  PRINT "970   POKE #19EA,TRACK:POKE #19EC,SECTOR:POKE #19EE,PEEK(VARPT
R(R$))"
1490  PRINT "980   POKE #19EF,PEEK(VARPTR(R$)+1):CALLM #19E3:RETURN"
```

NEGATIEVE CURSOR

```
100   REM *** DE NEGATIEVE CURSOR ***************
110   REM *** GESCHREVEN DOOR : DE BONT CORNEEL *
120   REM ********************* 14 - 5 - 1983 ***
130   REM ****************************************
140   GOTO 400
200   REM *** BUITEN-MARGE SUBROUTINE
210   IC=#BFEE-((23-YP)*#86):POKE IC,208+ZP
215   IF YP>0 THEN POKE IC-#86,208
220   FOR IX=0 TO LEN(G$)-1:T$=MID$(G$,IX,1):GX=ASC(T$)
230   IP=#BFE7-((23-YP)*#86)-(XP*2):POKE IP,GX
240   XP=XP+1:NEXT
250   RETURN
300   REM *** GETC-ROUTINE
310   CURSOR 0,0:PRINT SPC(58);:POKE #75,95
320   CURSOR 10,0:PRINT "DRUK OP EEN TOETS OM TE VERVOLGEN";
330   CC=INT(15*RND(1)):IF CC=8 THEN 330
340   POKE #B3E4,CC+208:SOUND 1 0 15 0 FREQ((CC*50)+50)
350   WAIT TIME 3:SOUND OFF :G=GETC:G=GETC:G=GETC
360   G=GETC:IF G=0 THEN WAIT TIME 3:I=I+1:IF I<30 THEN 360
370   I=0:IF G=0 THEN 330
380   POKE #75,32:RETURN
400   REM *** TITEL
410   MODE 0:COLORT 8 0 8 8:PRINT CHR$(12);:POKE #75,32
420   CURSOR 55,7:ZP=-1:FOR I=0 TO 23:XP=-3+I:YP=23-I
430   G$="DEMONSTRATIE VAN NEGATIEVE CURSOR"
440   ZP=ZP+1:IF ZP=16 THEN ZP=0
450   IF ZP=8 THEN ZP=ZP+1
460   GOSUB 200:NEXT:PRINT
470   GOSUB 300:RESTORE
500   PRINT CHR$(12);:COLORT 8 0 8 8:CURSOR 55,7
510   FOR YP=23 TO 1 STEP -1:READ G$:ZP=0
520   XP=INT(3*RND(1)):IF RND(10)>5 THEN XP=XP*(-1)
530   GOSUB 200:NEXT
540   GOSUB 300:GOTO 400
600   REM *** INFO
610   DATA "DEMONSTRATIE VAN EEN NEGATIEVE CURSOR"
620   DATA "**************************************"
630   DATA "DIT PROGRAMMA DEMONSTREERT U EEN SUB-"
640   DATA "ROUTINE,WAARIN EEN NEGATIEVE CURSOR  "
650   DATA "WORDT GEBRUIKT.EEN OPGEGEVEN STRING  "
660   DATA "WORDT OP ELKE GEWENSTE PLAATS OP HET "
670   DATA "SCHERM WEGGESCHREVEN,NET ALS BIJ HET "
680   DATA "GEBRUIK VAN EEN CURSOR-STATEMENT !!  "
690   DATA "HET GROTE VERSCHIL IS DAT MEN HIER   "
700   DATA "CURSOR X-WAARDEN TOT -3 KAN OPGEVEN. "
710   DATA "OOK BESTAAT DE MOGELIJK DE REGELS TE "
720   DATA "KLEUREN.DIT KAN MEN KLAARSPELEN DOOR "
730   DATA "BIJ HET AANROEPEN DER SUBROUTINE DE  "
740   DATA "VARIABELE 'ZG' EEN WAARDE TUSSEN 0   "
750   DATA "EN 15 MEE TE GEVEN. (0=ZWART)        "
760   DATA "DEZE SUBROUTINE KAN ALS VOLGT AANGE- "
770   DATA "SPROKEN WORDEN.MEN BEPAALT EERST:    "
780   DATA "XG : (X-AS) EEN GETAL TUSSEN -3 EN 58"
790   DATA "YG : (Y-AS) EEN GETAL TUSSEN 0 EN 23 "
800   DATA "ZG : (KLEUR) EEN GETAL TUSSEN 0 EN 15"
810   DATA "G$ : DE STRING (TOT MAX 62 LETTERS)  "
820   DATA "WAARNA MEN EEN 'GOSUB 200' UITVOERT  "
830   DATA "(XG=-3:YG=9:ZG=7:G$='DEMO':GOSUB 200)"
```