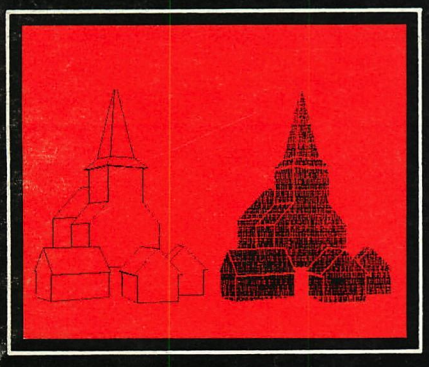
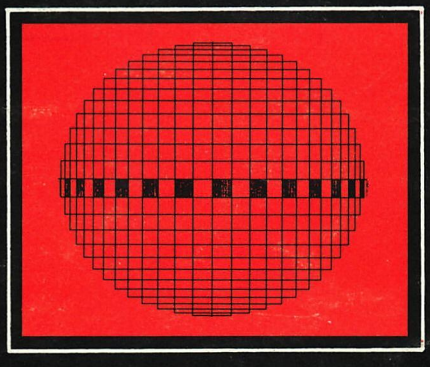
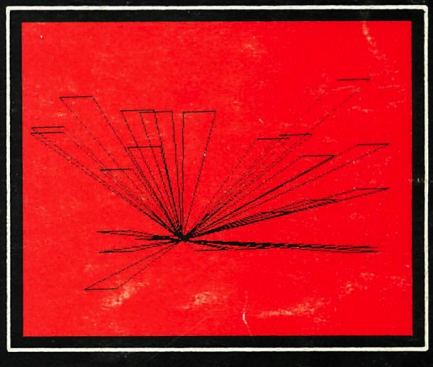
DAI NAMIC

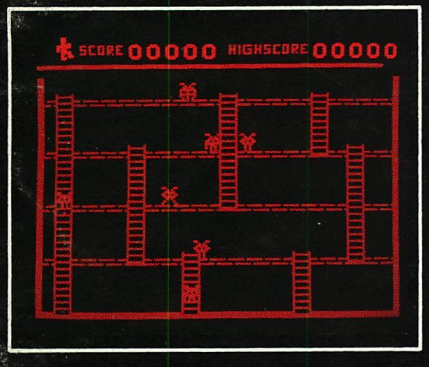
13

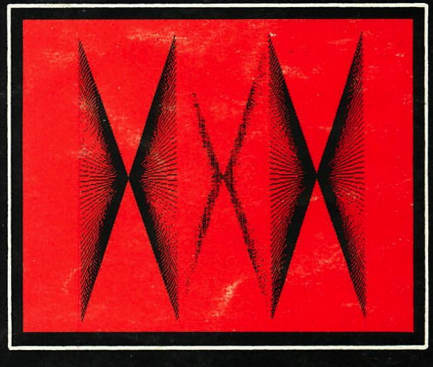
Tweemaandelijks tijdschrift november - december 1982

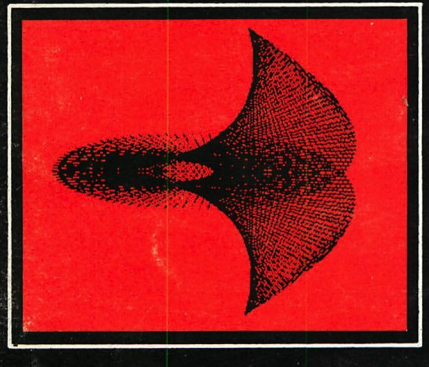


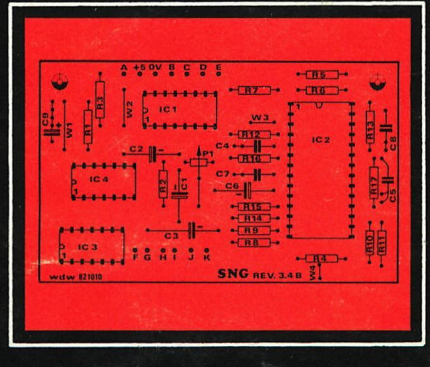


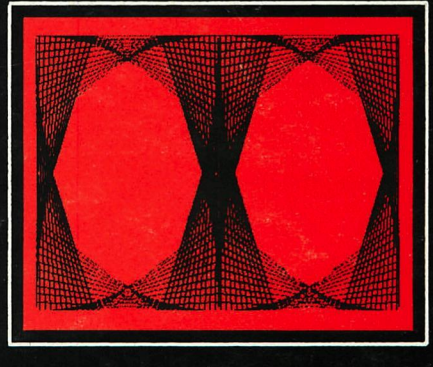












personal computer users club

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

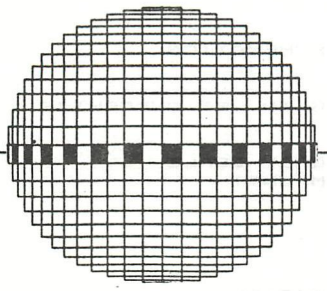
belangrijke ASCII-waarden in DAIpc

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
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 0000	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

DAI NEWS



Westmeerbeek, december 82

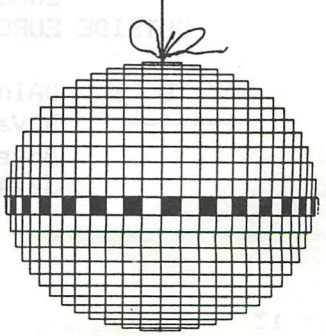
Beste leden,

We besluiten onze redactionele activiteiten van 1982 met dit extra dikke nummer (80 pagina's). Vele inzenders zullen bemerken dat hun bijdrage nog niet aan bod gekomen is. Voor volgend jaar hebben we echter al materiaal klaar voor minstens 3 nummers ... een grote luxe! Mogen we U vragen uw abonnement voor 1983 vóór 1 januari te hernieuwen zodat we nummer 14 tijdig aan alle leden kunnen versturen. MIKROBEL 82 was beslist een succes : de opkomst en belangstelling was zeer groot, diegenen die niet konden aanwezig zijn kunnen op de laatste pagina van dit nummer even de sfeer proeven. We willen hier ook alle medewerkers en standhouders bedanken voor hun aanwezigheid en inzet. We reserveren alvast de feestzaal van Aarschot voor volgend jaar. In April 83 houden we waarschijnlijk onze 'kleine' bijeenkomst op Tongelsbos, daarover meer in volgende edities. De activiteiten rond DAIpc zijn de laatste maanden koortsachtig gestegen : de firma INDATA zet zich in om nieuwe apparaten en uitbreidingen op de markt te brengen. Binnen DIDAISOFT wordt er druk vergaderd en gewerkt : Bruno heeft de idee naar voor gebracht om van het engelse handboek een degelijke nederlandse versie te maken, de taken zijn reeds verdeeld, wij houden U op de hoogte.DIDAISOFT zal ook spoedig de eerste didactische verzameltape uitbrengen. Ook de kwaliteit van de beschikbare software wordt steeds beter : voor de spelletjes-fanaten is DAI-PANIC beslist een enorme verrassing, de mensen die graag zelf programmeren zullen aan TOOLKIT 4 veel plezier beleven. Het ledenaantal stijgt enorm : we zullen spoedig ons 1000-ste lid mogen verwelkomen. In dit nummer vindt U ook een uitgebreid artikel over SUPER NOISE GENERATOR, knutselaars vinden hier alle nodige informatie om deze uitbreiding te maken, maar ook de mensen die niet houden van solderen vinden hierin veel hard- en software informatie.

This is the last issue of 1983, please be so kind to renew your subscription before 1st of january so that we can update the mailing for Newsletter 14. There were a lot of visitors on MIKROBEL 82 : we think we will organise 2 meetings a year : a small one in Tongerlo, and the 'big' one in Aarschot. In this issue you will find some interesting new software : DAI-PANIC and Toolkit 4 are really exciting. We will soon welcome our member number 1000...

we wish you the best for 1983, keep on keying...

W.Hermans



INHOUD — CONTENTS

No part of this book may be reproduced in any form, by print, photoprint, microfilm or any other means without written permission from the publisher.
Niets uit deze uitgave mag worden vervielvoudigd en/of openbaar gemaakt door middel van druk, fotocopie, microfilm of op welke andere wijze ook zonder voorafgaande schriftelijke toestemming van de uitgever.

****
* RENEW YOUR SUBSCRIPTION NOW ! *
****

NEW SUBSCRIPTION RATES :

BENELUX	:	900 Bfr
EUROPE	:	1000 Bfr
OUTSIDE EUROPE (AIRMAIL)	:	1400 Bfr

PAY TO : DAInamic-SUBSCRIPTIONS

B.Van Rompaey
Bovenbosstraat 4
3044 HAASRODE BELGIUM

IN BELGIAN FRANCS.

1. SEND CHEQUE
2. PAY ON BANCACCOUNT NR 230-00455353-74
   GENERALE BANK LEUVEN c/o DAInamic

>>>>>>>>PROGRAMMEER - TECHNIEKEN<<<<<<<<

Ik wil beginnen met mijn excuses te maken voor enige slippertjes in het vorige artikel. Om de redactie werk te besparen had ik niet een tamelijk groot aantal losse blaadjes ingestuurd met respectievelijk tekst(en) en listings maar het hele artikel achter elkaar getikt. Prompt slopen er dan ook enige foutjes in die programma's. Ik hoop echter dat het artikel duidelijk genoeg was omdeze zelf te verbeteren.

Nu het onderwerp voor deze keer : de kleuren 16 t/m 19 als hoofdzaak en de kleuren 20 t/m 23 als bijzaak.
Om maar meteen met het laatste te beginnen, het zou een goede zaak zijn als al onze inzenders zich bij tekenen in de vier-kleuren modes zouden bezighouden met de kleuren 20 t/m 23. Het is toch zo simpel. Aan het begin van het programma geven we een COLORG A B C D met A,B,C en D de door ons gewenste vier kleuren. Geven we nu een teken opdracht in het programma dan gebruiken we kleur 20 als we in kleur A willen tekenen, 21 voor B, 22 voor C en 23 voor kleur D.
Het is trouwens een goede gewoonte eerst de COLORG-instructie te geven en dan pas de MODE-instructie. We voorkomen hiermee dat het beeld eerst fel geel wordt doordat de vorige COLORG-instructie dit als eerste kleur had en dan pas het nu gewenste zwart. Bij de 16-kleuren modes wordt overigens de eerste kleur van de COLORG genomen als achtergrondkleur en de rand krijgt dan ook die kleur. Dit is dan niet met een FILL 0,0 XMAX,YMAX K weg te krijgen maar alleen met een nieuwe COLORG en daarna een nieuwe MODE 1,3 of 5.
Het voordeel van het gebruik van vooral anderen van kleur 20 t/m 23 is gelegen i het feit dat een nieuwe gebruiker de voor hem slecht uitkomende kleurencombinatie gemakkelijk kan wijzigen. Op mijn RGB monitor kan ik uitstekend gebruik maken van COLORG 0 1 2 3 maar ik weet zeker dat de zwart/wit kijkers mij dan zullen vervloeken wat ik omgekeerd doe met 4 5 8 13.

Voordat we nu aan de kleuren 16 t/m 19 toekomen moet ik eerst iets uitleggen over de beeldopbouw intern in het geheugen. Is mijn uitleg niet duidelijk genoeg slaat U dan eens de artikelen van N.J.Looije of J.Boerrigter op in vorige nummers.

Het beeldgeheugen kan gezien worden als twee boven elkaar (of naast elkaar als U dat beter aanspreekt) gelegen geheugens die samen het uiteindelijke beeld bepalen. De geheugenbanken zijn ook hardwarematig gescheiden - de B-bank voor de even genummerde bytes en de C-bank voor de oneven genummerde bytes. Voor het maken van programma's is dit echter voor de programmeur van geen enkel belang.

Voor de duidelijkheid zij nog even herhaald dat wat nu volgt de beeldopbouw voor vierkleuren modes (2,4,6 & 8) is. De zestienkleuren modes hebben een andere structuur, die hier niet aan de orde is.
De kleur die een punt op het beeld zal aannemen wordt als volgt bepaald :
Bij elk punt horen twee bits die bepalen in welke van de vier kleurregisters de kleur voor dat punt moet worden opgezocht.
Eerst het standaardgeval - we geven na reset een COLORG 0 3 9 14 en dan MODE 4. Het gehele beeld zal dan even zwart worden om vervolgens op te schuiven tot 4A. Tikken we nu UT in kunnen we met Display vaststellen dat op de linecontrolbytes na alle bytes op 00 staan. (DB000 B500 bv). Wat gebeurt er nu als we deze bytes een andere waarde geven. We proberen POKE#B000,#AA en zien vier stippen op het beeld verschijnen met een rode kleur. Voor diegenen die nog weinig ervaring hebben met bit-informatie #AA=10101010. Nu vervolgen we met POKE#B002,#66. (#66=01100110) Nu verschijnen er twee kleine rode lijntjes naast de puntjes. POKE#B001,#F0 : Van de vier rode puntjes worden de linker twee geel en er komen blauwe puntjes bij. Met enig nadenken kunnen we uit deze voorbeelden beredeneren hoe de kleur tot stand komt.

bit in even byte	bit in oneven byte	kleur algemeen	kleur in voorbeeld
0	0	20	0 = zwart
1	0	21	3 = rood
0	1	22	9 = blauw
1	1	23	14= geel

Het is echter mogelijk ook in vier kleuren mode toch meer dan vier kleuren te hebben. Het artikel van N.J. Looije over de video screen ram geeft hier veel informatie. Ik wil dit hier niet nogmaals bespreken maar me er toe beperken vast te stellen dat we in elke door ons gewenste regel de colorcontrolbyte zo kunnen kiezen dat de kleurregisters een voor een veranderd kunnen worden.
Per regel kan dan een van de vier kleuren veranderd worden.
Met een beetje handig programmeren kan dit er voor zorgen dat we elk punt precies die kleur (uit 4 kleuren) kunnen geven dat we ons wensen, terwijl we toch ook over alle kleuren over het hele beeld gezien kunnen beschikken.

Maar nu terzake. Ik wilde het hebben over de kleuren 16 t/m 19. In een uitleg die deze naam nauwelijks verdient tracht het handboek ons duidelijk te maken wat de mogelijkheden zijn met deze speciale kleurenset.
In het begin is het nog te volgen, de kleuren boven de 16 zijn geen echte kleuren maar speciale toepassingen. De kleuren 20 t/m 23 refereren aan de kleuren van de COLORG-instructie en de kleuren 16 t/m 19 zetten bytes in video ram aan of uit. Hier komen we terug op de inleiding; de kleur van elk punt in beeld wordt bepaald door twee bits. Deze twee bits zitten in twee verschillende bytes een met een even nummer en een met een oneven. Het oneven nummer steeds een groter dan het even nummer. Door nu een tekenopdracht te geven met kleur 17 zetten we de bits op een die horen bij de even bytes.
Als we na reset en MODE 4 de opdracht FILL 0,0 44,44 K geven maakt het achteraf niets uit of voor K nu 5, 17 of 21 gekozen hebben.
Als we echter niet na reset maar na een ander programma 21 gebruiken kan het zijn dat ons vlak niet groen is maar een andere kleur. Toch is 21 beter dan 5 omdat we na dit andere programma met gebruik van 5 een kans hebben de mededeling COLOUR NOT AVAILABLE te krijgen.
Bij tekenen in kleur 17 is dit risico er niet.
We demonstreren verschil van tekenen in kleur 17 en 5(evt 21 na juiste COLORG)
We tikken MODE 4

```
FILL 22,22 66,66 22
FILL 0,0 44,44 5
```

en zien dat het groene vlak een rechthoek is die een deel van de oranje (22=10 standaard) rechthoek heeft "weggehapt".
Tikken we de opdrachten nogmaals in en gebruiken we dan kleur 17 in plaats van kleur 5 zien we dat het groene vlak geen rechthoek is maar een soort L.

Het deel dat de rechthoeken elkaar overlappen is nu een nieuwe rechthoek met de kleur wit.
We analyseren: door de MODE 4 opdracht zullen we (tenzij we van MODE 4A kwamen) alle bits die de kleur bepalen op nul zetten. Dit geldt ook als we niet kleur 0 als eerste kleur van de COLORG hadden al is het beeld dan wel een andere kleur.
De eerste FILL maakt een oranje rechthoek en zet in het geheugen de bijbehorende bits van de even bytes op nul (!!!!!!!) en de oneven bytes op een.
De tweede FILL zet bij gebruik van kleur 5 de bit van de even bytes op een en de bits van de oneven bytes op nul. Dit betekent dat de bits van de oneven bytes die bij de eerste FILL een waren geworden nu weer nul worden.
De resulterende kleur is dus de kleur die het laatst werd gegeven nl groen(5).
Als we bij de tweede FILL de opdracht geven in kleur 17 worden alleen de bits van de even bytes op een gezet. De bits van de oneven bytes worden NIET veranderd. Dit betekent dat het gebied waar de twee rechthoeken elkaar overlappen bij de eerste FILL enen in de oneven bytes kreeg en bij de tweede FILL met kleur 17 enen in de even bytes dus deze rechthoek kleur 23 (normaal wit) kreeg.

We kunnen dus met 17 alleen de bits van de even bytes op een zetten. Evenzo kunnen we met kleur 16 deze bits weer uitzetten.
Met kleur 19 kunnen we dan de bits van de oneven bytes op een zetten en met kleur 18 kunnen we ze weer op nul zetten.

Wat betekent dit nu voor de programmeur ?
Volgens het handboek zouden we hiermee animatie effecten kunnen verkrijgen maar hiervan moeten we ons toch niet te veel voorstellen.
Het effect is er theoretisch wel maar het tekenen gaat toch te traag om uitzonderingen daargelaten een film effect te verkrijgen.

Maar we zullen eens zien hoe een en ander te verwezenlijken is.
We maken een tekening in kleur 17 met COLORG 0 5 x x en zien deze tekening dan in kleur 5(groen) op het beeld. Als we voor x in de COLORG nul kiezen kunnen we hierna tekenen wat we willen in kleur 19 zonder dat dit zichtbaar wordt. Kleur 19 zet enen in de oneven bytes en dus worden de punten of kleur 22 als de andere bit niet een is of kleur 23 als de andere bit wel een is.
Maar dan wordt dit punt zwart. Ook groene punten worden zwart als er over heen getekend wordt; dit kunnen we voorkomen door ook kleur 23 groen te laten zijn Dus door COLORG 0 5 0 5 te gebruiken.

Als nu de nieuwe tekening met kleur 19 klaar is geven we COLORG 0 0 5 5.

Alle punten met twee bits aan (kleur 23) zijn groen en alle punten met alleen de bit van de oneven byte op een (kleur 22) zijn groen.
De punten waarvan beide bits op nul staan (kleur 20) zijn zwart en alle punten waarvan alleen de bits van de even bytes op een staan (kleur 21) zijn zwart.
We zien dus niets meer van de oude tekening en wel alles van de nieuwe.
Vervolgens gaan we tekening 1 uitwissen. Alle opdrachten van kleur 17 opnieuw maar nu met 16 (bits van even bytes op nul) of als dat sneller is met een FILL 0,0 XMAX,YMAX 16. Dan een nieuwe tekening (3) in kleur 17 maken.
COLORG 0 5 0 5 en tekening 3 is zichtbaar en tekening 2 niet meer.
Tekening 2 uitwissen met kleur 18,tekening 4 in kleur 19 maken en dan weer COLORG 0 0 5 5. tekening 4 is zichtbaar en tekening 3 niet meer.
Tekening 3 uitwissen met kleur 16,tekening 5 in kleur 17 maken en dan weer COLORG 0 5 0 5. enzovoorts,enzovoorts.

We kunnen dus bv een auto tekenen die dan zeer gelijkmatig over het beeld kan bewegen. Als de auto echter uit te veel lijnen bestaat is het tempo toch wel te laag om van animatie te kunnen spreken. Nog net redelijk haalbaar is een tekening met zo'n 5 a 6 lijnen van elk ongeveer twintig punten. Anderen hebben misschien een minder kritische instelling en vinden een groter aantal nog acceptabel. Waar kunnen we dit dan wel voor gebruiken ? Ik denk aan een grotendeels statische tekening (in kleur 23) met slechts een paar bewegende delen, zoals een klok. Bedenk echter wel het nadeel dat U bij deze animaties aan twee kleuren (voor- en achtergrond) gebonden bent.

Toch zijn er wel toepassingen te bedenken waar deze facilteit in DAI-basic erg van pas kan komen. J. Visser gebruikt het bv zeer zinvol in zijn 3-dim doolhof. De situatie die de speler ziet vanuit zijn standpunt wordt op het scherm getoond. De speler kiest hierop een actie voor beweging het programma tekent dan voorlopig onzichtbaar de nieuwe situatie. Dan wordt door een COLORG het beeld zeer snel aangepast.

Ik heb zelf ook een doolhofprogramma geschreven dat mbv de kleuren 16-19 een bepaald deel van het scherm zichtbaar maakt. Je overziet dus altijd maar een klein deel van het doolhof.
Dit wordt als volgt gedaan. De tekening van het doolhof wordt gemaakt in kleur 17, vervolgens zetten we een blok (FILL A,B C,D 19) neer in kleur 19.
Na een COLORG 0 0 14 9 is nu het doolhof onzichtbaar want kleur 17 betekent even bytes en dat bij oneven bytes gelijk nul kleur 21 dus 0=zwart. Op de plaats echter waar het blok staat is de achtergrond geel en daar waar doolhof en blok beide staan wordt de resulterende kleur 23 dus 9=blauw.
Het blok kunnen we dan laten bewegen door steeds aan de ene zijde er een lijn in kleur 19 bij te tekenen en aan de andere zijde er een te wissen mer kleur 18.
Een andere aardige bijkomstigheid is de mogelijkheid een bepaald deel van het beeld een geinverteerde kleur te geven.
Denk maar met mij na: tekening maken in kleur 17, bepaald deel (FILL) kleuren in kleur 19. Geef dan COLORG 0 12 12 0 en de tekening is blauw met zwarte achtergrond. Maar op de plaats van de FILL is dit juist omgekeerd. Zo kunnen we bij bepaalde tekeningen delen extra laten opvallen of aanwijzen zonder dat dit de rest van de tekening verstoort.

Bij dit artikel zijn nog een aantal programma's gevoegd die laten zien hoe we op een slimme manier gebruik kunnen maken van de kleuren 16-19.

Frank H. Druijff

```
10    REM COLORS 16-19 DEMO  /  F.H. DRUIJFF  2/82
20    COLORG 0 0 0 0:MODE 6
30    FOR X=0 TO XMAX-24 STEP 2
40    DRAW X,3 X+12,50 18:DRAW X+2,3 X+14,50 19
50    DRAW X+12,50 X+21,3 18:DRAW X+14,50 X+23,3 19
60    DRAW X,3 X+21,3 18:DRAW X+2,3 X+23,3 19
70    FILL X+10,0 X+12,2 18:FILL X+12,0 X+14,2 19
80    COLORG 0 0 14 14
90    DRAW X+1,3 X+13,50 16:DRAW X+3,3 X+15,50 17
100   DRAW X+13,50 X+22,3 16:DRAW X+15,50 X+24,3 17
110   DRAW X+1,3 X+22,3 16:DRAW X+3,3 X+24,3 17
120   FILL X+11,0 X+13,2 16:FILL X+13,0 X+15,2 17
130   COLORG 0 14 0 14
140   NEXT
```

```
10    REM COLORS 16-19 DEMO  /  F.H. DRUIJFF  2/82
20    COLORG 0 0 0 0:MODE 6:DRAW 0,2 XMAX,2 23
30    FOR X=0 TO XMAX-40 STEP 2
40    DRAW X+2,3 X,9 18:DRAW X,9 X+30,9 18
50    DRAW X+30,9 X+24,3 18:DRAW X+15,10 X+15,30 18
60    DRAW X+4,3 X+2,9 19:DRAW X+2,9 X+32,9 19
70    DRAW X+32,9 X+26,3 19:DRAW X+17,10 X+17,30 19
80    COLORG 0 0 12 12
90    DRAW X+3,3 X+1,9 16:DRAW X+1,9 X+31,9 16
100   DRAW X+31,9 X+25,3 16:DRAW X+16,10 X+16,30 16
110   DRAW X+5,3 X+3,9 17:DRAW X+3,9 X+33,9 17
120   DRAW X+33,9 X+27,3 17:DRAW X+18,10 X+18,30 17
130   COLORG 0 12 0 12:NEXT
```

```
10    REM COLORS'16-19 DEMO  /  F.H. DRUIJFF  2/82
20    COLORG 0 0 0 0:MODE 4:D=1
30    FOR X=0 TO XMAX-25:D=1-D:E=18-D-D:F=E+1:I=X+D
40    I2=I+2:I11=I+11:I13=I+13:I22=I+22:I24=I+24
50    DRAW I,3 I11,50 E:DRAW I11,50 I22,3 E:DRAW I,3 I22,3 E:FILL I+10,0 I11,2 E
60    DRAW I2,3 I13,50 F:DRAW I13,50 I24,3 F:DRAW I2,3 I24,3 F:FILL I+12,0 I13,2 F
70    COLORG 0 5*D 5-5*D 5:NEXT:GOTO 20
```

```
10    REM COLORS 16-19 DEMO  /  F.H. DRUIJFF  2/82
20    COLORG 0 0 0 0:MODE 4
30    FOR X=0 TO XMAX-25 STEP 2:FOR J=0 TO 1:FOR L=0 TO 1
40    K=16+J+J+L:I=X+J+L+L:REM PICTURE MOVES FROM LEFT TO RIGHT
50    I3=I+3:I20=I+20:DRAW I,3 I3,6 K:DRAW I,9 I3,6 K
60    DRAW I3,6 I20,6 K:DRAW I20,6 I+17,3 K:DRAW I20,6 I+17,9 K
70    NEXT:COLORG 0 5-5*J 5*J 5:NEXT:NEXT:GOTO 20
```

****
* *
* ERRATA DAI pc SCHEMATICS *
* *
****

The following failures have been found in the diagrams of the DAI.
Please update your copies accordingly.

Sheet 5: The pin lay-out of the ROM's MK36000 are reversed.
The correct lay-out is:

A0	pin 8	A8	pin 23	D0	pin 9
A1	pin 7	A9	pin 22	D1	pin 10
A2	pin 6	A10	pin 19	D2	pin 11
A3	pin 5	A11	pin 18	D3	pin 13
A4	pin 4	A12	pin 21	D4	pin 14
A5	pin 3			D5	pin 15
A6	pin 2			D6	pin 16
A7	pin 1			D7	pin 17

Sheet 7: The polarisation of the diodes D43, D44, D45 has to be reversed.

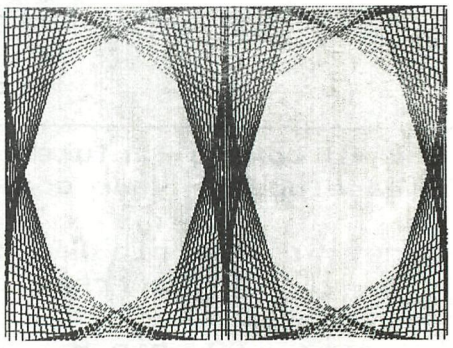
Sheet 10: R101 has a value of 5,6 kohm.

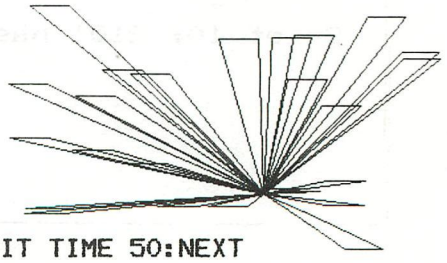
Jan Boerrigter

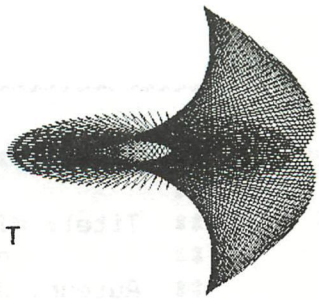
program identification

title	:	DEMO POELS
author	:	Christian POELS
purpose	:	demonstrates graphic capabilities of DAIpc
comment	:	BREAK-key is disabled in line 53846

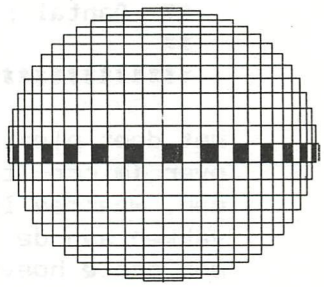
```
5     MODE 0:PRINT CHR$(12):COLORT 9 15 9 9:POKE #BA2D,90:CURSOR 0,12
6     PRINT " = DEMO POELS =":CURSOR 30,5:PRINT "Christian POELS / 29-7-1982"
7     IF GETC<>32.0 THEN 7
8     COLORT 13 1 0 0
9     GOSUB 50000
10    MODE 6:COLORG 8 3 11 0
20    FOR X=0.0 TO XMAX/2.0 STEP 5.0:C=X*6.0/XMAX+21.0
30    DRAW X,0 XMAX/2,2*X*YMAX/XMAX C
40    DRAW XMAX/2-X,0 0,2*X*YMAX/XMAX C
50    DRAW X,YMAX XMAX/2,YMAX-2*X*YMAX/XMAX C
60    DRAW XMAX/2-X,YMAX 0,YMAX-2*X*YMAX/XMAX C
70    DRAW X+XMAX/2,0 XMAX,2*X*YMAX/XMAX C
80    DRAW XMAX-X,0 XMAX/2,2*X*YMAX/XMAX C
90    DRAW X+XMAX/2,YMAX XMAX,YMAX-2*X*YMAX/XMAX C
100   DRAW XMAX-X,YMAX XMAX/2,YMAX-2*X*YMAX/XMAX C
110   NEXT:WAIT TIME 250
150   MODE 6
170   COLORG 13 0 0 0
180   FOR BOU=1.0 TO 4.0:MODE 6:XX=RND(XMAX):YY=RND(YMAX):FOR O=1.0 TO 25.0:X=RN
D(XMAX-30.0):Y=RND(YMAX)
190   DRAW X,Y X+30,Y 0:DRAW XX,YY X,Y 0:DRAW XX,YY X+30,Y 0
200   NEXT
210   WAIT TIME 100:NEXT
320   REM ALEATOIRE
330   MODE 0:PRINT CHR$(12):PRINT "TEST D'HOMOGENEITE DE LA FONCTION 'RND'":WAIT
 TIME 150
340   CLEAR 1000:DIM C%(15.0):COLORG 0 0 0 0:MODE 1:MODE 1
350   X%=RND(15.0)+1.0:C%(X%)=C%(X%)+1.0:IF C%(X%)=46.0 THEN 370
360   FILL C%(X%)-1.0,X%*4 C%(X%),(X%+1)*4-1 X%:GOTO 350
370   REM DESSIN
380   MODE 6:COLORG 8 1 0 0:H1=240.0:V1=180.0
390   FOR I=0.0 TO 9540.0 STEP 53.0
400   X=I*PI/180.0
410   H=130.0+110.0*COS(2.0*X)*COS(X)
420   V=180.0+150.0*SIN(3.0*X)*SIN(X)
430   DRAW V,H V1,H1 1:H1=H:V1=V:NEXT I
440   FOR O=1.0 TO 20.0:COLORG RND(16.0) RND(16.0) 0 0:WAIT TIME 50:NEXT
450   MODE 6:COLORG 0 1 3 14:FILL XMAX/2-3,YMAX/2-3 XMAX/2+3,YMAX/2+3 23
460   FOR Y=0.0 TO YMAX/2.0 STEP 3.0
470   DRAW 0,Y XMAX,0 21:DRAW 0,0 XMAX,Y 22
480   DRAW 0,YMAX XMAX,YMAX-Y 21:DRAW XMAX,YMAX 0,YMAX-Y 23
490   DRAW Y*XMAX/YMAX,0 0,YMAX 21:DRAW 0,0 Y*XMAX/YMAX,YMAX 22
500   DRAW XMAX,0 XMAX-Y*XMAX/YMAX,YMAX 21:DRAW XMAX,YMAX XMAX-Y*XMAX/YMAX,0 23
510   NEXT
```

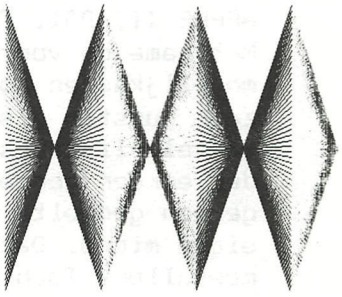






```
520   FOR T=100.0 TO 0.0 STEP -10.0:GOSUB 550:NEXT
530   T=2.0
540   FOR M=1.0 TO 90.0:GOSUB 550:NEXT:GOTO 570
550   COLORG 0 3 14 1:WAIT TIME T:COLORG 0 14 1 3:WAIT TIME T
560   COLORG 0 1 3 14:WAIT TIME T:RETURN
570   MODE 6:N=9.0:COLORG 0 N 5 3
580   A=287.0:B=112.0:C=47.0:D=142.0:J=15.0:F=127.0:H=167.0
590   DIM CO(2.0)
600   CO(1.0)=5.0:CO(2.0)=3.0:Z3=1.0
610   FOR I=1.0 TO J
620   DRAW A-I,F A-I,B+J N:DRAW H,B+J A-I,B+J N
630   DRAW C+I,F C+I,B+J N:DRAW H,B+J C+I,B+J N
640   DRAW C+I,F C+I,D-J N:DRAW H,D-J C+I,D-J N
650   DRAW A-I,F A-I,D-J N:DRAW H,D-J A-I,D-J N
660   A=A-I:B=B+(J-I):C=C+I:D=D-(J-I)
670   NEXT I
680   FOR NB=1.0 TO 60.0:Z3=2.0
690   IF NB/2.0=INT(NB/2.0) THEN Z3=1.0
695   FOR BOUCLE=1.0 TO 8.0
700   A=287.0:I1=1.0:I2=15.0:I3=1.0:IF Z1>1000.0 THEN Z1=0.0
710   Z1=Z1+1.0:GOSUB 760
720   I1=15.0:I2=1.0:I3=-1.0
730   Z1=Z1+10.0:GOSUB 760
740   NEXT BOUCLE:GOTO 840
760   FOR I=I1 TO I2 STEP I3
770   FILL (A+1)-I,128 (A-1),140 CO(Z3)
780   Z1=Z1+1.0:Z2=Z1/2.0:IF INT(Z2)<>Z2 GOTO 800
790   GOTO 810
800   FILL (A+1)-I,128 (A-1),140 0
810   A=A-I
820   NEXT I
830   RETURN
840   MODE 6
850   COLORG 9 2 14 0:FOR Y%=0 TO YMAX STEP 5:DRAW 0,Y% XMAX/4,YMAX-Y% 21
860   DRAW XMAX/4,YMAX-Y% XMAX/2,Y% 22:DRAW XMAX/2,Y% 3*(XMAX/4),YMAX-Y% 21:DRAW
 3*(XMAX/4),YMAX-Y% XMAX,Y% 22
870   NEXT
880   WAIT TIME 200
40000 REM KALEIDOSCOPE
40010 COLORG 8 1 3 5:MODE 6
40020 FOR I%=1 TO 20:X1%=RND(XMAX/2.0):Y1%=RND(YMAX/2.0):XX1%=RND(XMAX/2.0):YY1%
=RND(YMAX/2.0)
40030 X2%=Y1%*XMAX/YMAX:Y2%=X1%*YMAX/XMAX:X3%=XMAX-X2%:Y3%=Y2%:X4%=XMAX-X1%:Y4%=
Y1%
40040 XX2%=YY1%*XMAX/YMAX:YY2%=XX1%*YMAX/XMAX:XX3%=XMAX-XX2%:YY3%=YY2%:XX4%=XMAX
-XX1%:YY4%=YY1%
40050 X5%=X4%:Y5%=YMAX-Y1%:X6%=X3%:Y6%=YMAX-Y3%:X7%=X2%:Y7%=Y6%
40060 XX5%=XX4%:YY5%=YMAX-YY1%:XX6%=XX3%:YY6%=YMAX-YY3%:XX7%=XX2%:YY7%=YY6%
40070 X8%=X1%:Y8%=Y5%
40080 XX8%=XX1%:YY8%=YY5%
40090 C%=RND(4.0)+16.0:FILL X1%,Y1% XX1%,YY1% C%:FILL X2%,Y2% XX2%,YY2% C%:FILL
X3%,Y3% XX3%,YY3% C%:FILL X4%,Y4% XX4%,YY4% C%
40100 FILL X5%,Y5% XX5%,YY5% C%:FILL X6%,Y6% XX6%,YY6% C%:FILL X7%,Y7% XX7%,YY7%
 C%:FILL X8%,Y8% XX8%,YY8% C%
40110 NEXT:COLORG RND(16.0) RND(16.0) RND(16.0) RND(16.0):GOTO 40020
50000 REM Ou est le POKE?!?...
53846 POKE #5F,#84:RETURN:REM Bien vu!
60000 REM
```





```
**************************************************************
**                                                          **
**  Titel: MICROCOMPUTERS IN HET ONDERWIJS                  **
**         natuurwetenschap en techniek                     **
**  Auteur: Jan Roelants,ing.                               **
**  Uitgave: 1982, Van In - Lier, Malmberg - Den Bosch      **
**  ISBN: 90 306 10441                                      **
**  Prijs: Bfr 590,-                                        **
**  Aantal pagina's: 309                                    **
**                                                          **
**************************************************************
```

Het doet plezierig aan om in de groeiende berg van literatuur over (micro-)computers in het onderwijs een uitgave tegen te komen, waarmee leraren in de technische en natuurwetenschapelijke vakken aan de slag kunnen.Het boek van Jan Roelants biedt daartoe een grote hoeveelheid software.

De inhoud valt uiteen in een algemeen gedeelte (~100 pagina's) en een software gedeelte. Het algemene gedeelte bestaat grotendeels uit een vergelijking van de basic dialecten van CBM 3032 APPLE II, DAI, TRS 80 LEVEL II, P2000T PHILIPS en SORCERER. Met name de voor het onderwijs zo belangrijke edit en grafics mogelijkheden komen uitvoerig aan de orde. Dat de DAI daarbij zeer gunstig afsteekt, zal U nauwelijks meer verbazen.
De vele listings die daarna volgen zijn steeds afkomstig van een der eerder genoemde microcomputers. Met de informatie uit het algemeen gedeelte kan de lezer de programma's aanpassen aan zijn eigen micro. Dat zal soms voor een beginnend programmeur niet meevallen. Toch lijkt mij deze opzet te verkiezen boven een weergave van alle listings in b.v. de BASICODE-norm.
Ook voor niet onderwijsgevenden kan het algemeen gedeelte zeer nuttig zijn.

In het tweede gedeelte van "Microcomputers in het onderwijs" komen aan de orde:
- natuurkunde (4 programma's)
- mechanica (4 programma's)
- sterkteleer (1 programma)
- elektriciteit (4 programma's)
- regel techniek (2 programma's)
- digitale techniek (1 programma)
- wiskunde (3 programma's)
- tekenen van de ruimte (1 programma)

Steeds wordt de eigenlijke listing vooraf gegaan door een analyse van het probleem en een bespreking van het programma (flow charts). Van de grafische gedeelten zijn een aantal screencopies opgenomen.

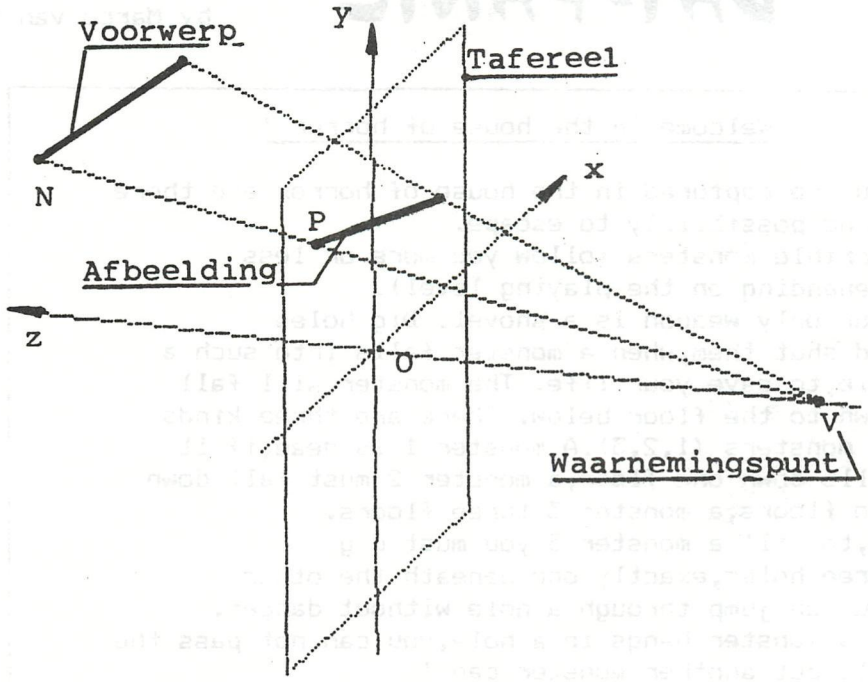
This book, together with the programs on tape, is available from DAInamic.
- BOOK + programs on audio : 990 Bfr
- BOOK + programs on DCR :1100 Bfr

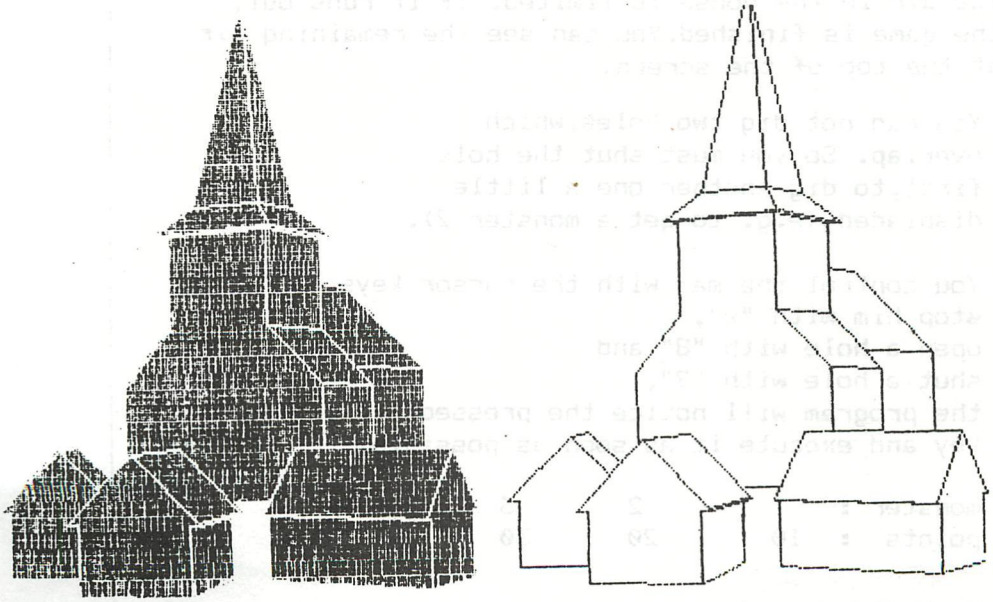
Het merendeel van de programma's is volgens eenzelfde stramien opgebouwd. Centraal staan een of meer formules behorende bij een natuurkundige of technische situatie. In een invoergedeelte worden aan de gebruiker van het programma de parameters behorende bij de situatie gevraagd. De fysische variabelen (kracht, snelheid spanning,...) en hun onderlinge samenhang behorende bij de gegeven situatie wordt door de computer berekend en zo mogelijk grafisch weergegeven. De computer wordt dus primair gebruikt als rekenmachiene, waarmee de samenhang tussen verschillende variabelen snel kan worden doorgerekend en gepresenteerd.

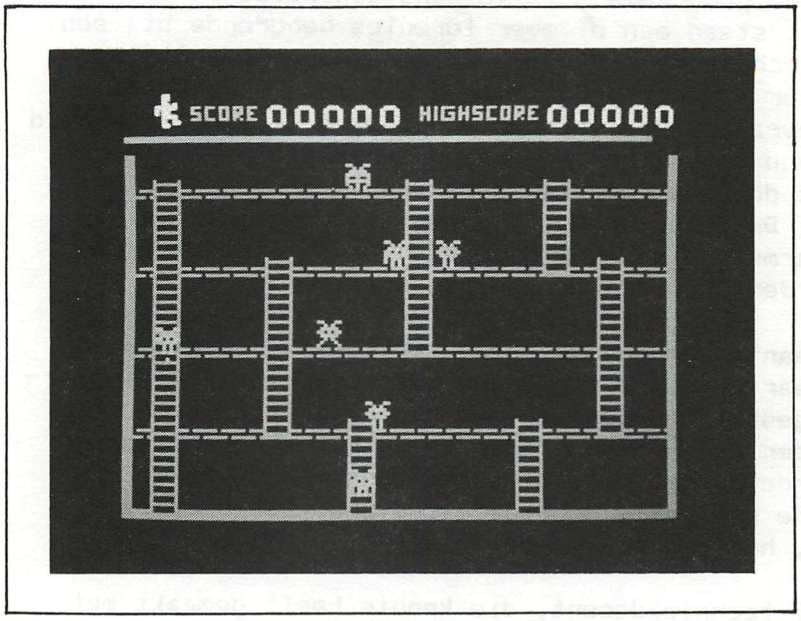
Wie verwacht een aantal programma's aan te treffen die direkt in de klas bruikbaar zijn (de titel doet dit vermoeden), zal waarschijnlijk teleurgesteld worden. Tussen het didaktisch gebruik van de microcomputer bij het onderwijs in de natuurwetenschappen en de gepresenteerde programma's ligt een lange weg.
Terecht staat in de motivatie door de uitgevers:"Dit boek is slechts een begin, het openen van een poort".

De natuurkunde- of techniekdocent, die kennis heeft gemaakt met de microcomputer en die kennis bruikbaar wil maken voor zijn vak, vindt in dit boek de helpende hand van Jan Roelants en kan zo de eerste stap zetten op een lange moeizame weg.

T.BERCKX







NEW SOFTWARE

DAI-PANIC

by Marco van Meegen

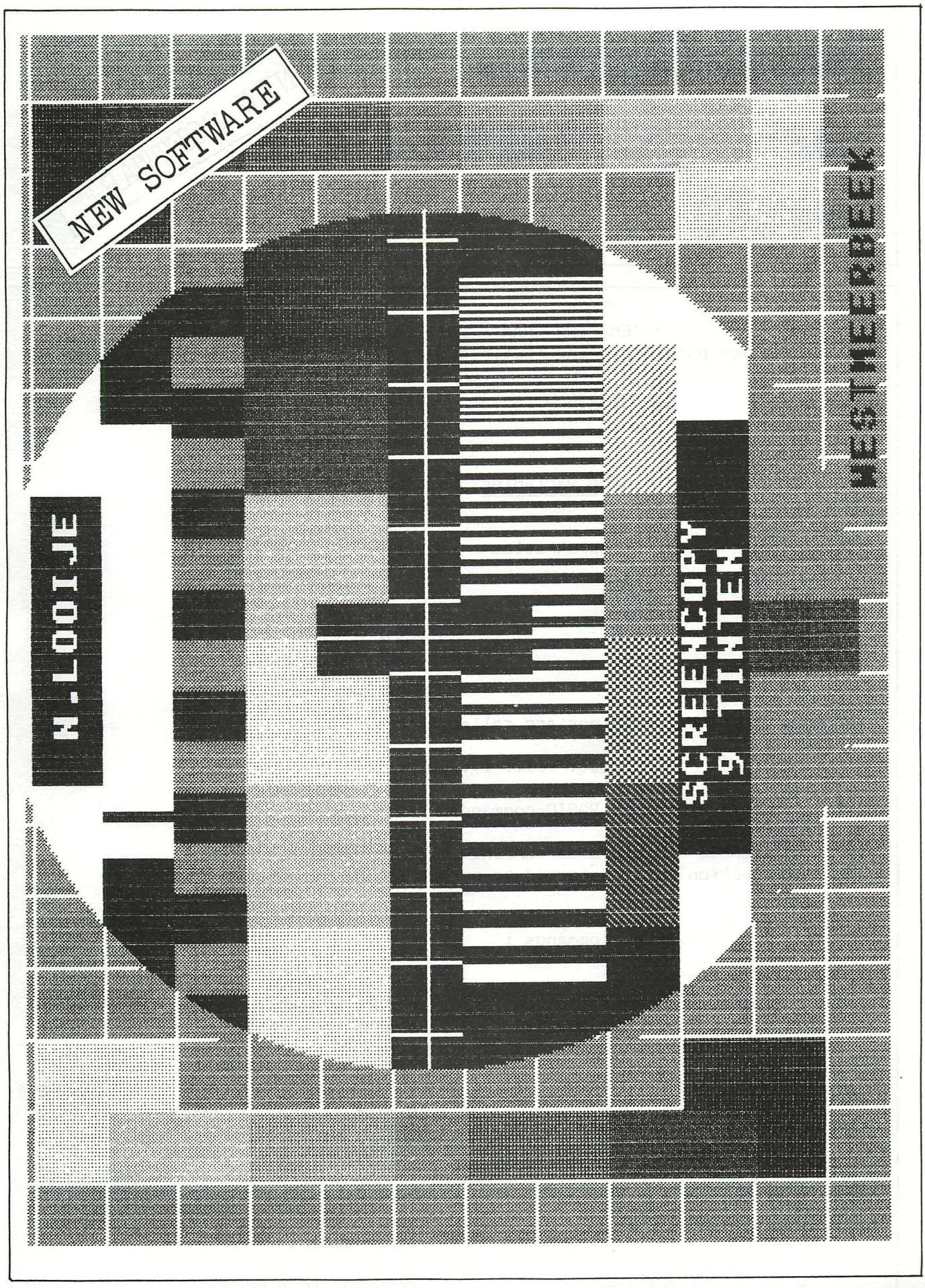
Welcome in the house of horror !

You are captured in the house of horror and there is no possibility to escape.
Horrible monsters follow you more or less (depending on the playing level).
Your only weapon is a shovel. Dig holes and shut them,when a monster falls into such a hole,to save your life. The monster will fall down to the floor below. There are three kinds of monsters (1,2,3).A monster 1 is dead,if it falls down one floor,a monster 2 must fall down two floors,a monster 3 three floors.
So,to kill a monster 3 you must dig three holes,exactly one beneath the other.
You can jump through a hole without danger.
If a monster hangs in a hole,you can not pass the hole,but another monster can !
The air in the house is limited. If it runs out, the game is finished.You can see the remaining air at the top of the screen.

You can not dig two holes,which overlap. So you must shut the hole first,to dig another one a little displaced (e.g. to get a monster 2).

You control the man with the cursor keys,
stop him with "0",
open a hole with "8" and
shut a hole with "9",
the program will notice the pressed key and execute it as soon as possible.

monster :	1	2	3
points :	10	20	30



IF YOUR EPSON-PRINTER HAS GRAPHIC CAPABILITIES,
MX-80 WITH GRAFTRAX,MX-82 OR MX-100,
THEN YOU CAN PRINT ALL THOSE BEAUTIFUL DAI-COLORS
-- IN 9 GREY-SCALES !!
available on TOOLKIT 4

system-program-unit

NEW SOFTWARE

SYSTEM - PROGRAM - UNIT contains in one machine language part following routines :

* renumber
  - step - mode
  - add - mode
  - sub - mode

* rename variables
  - global
  - with line-restrictions

* variablen-atlas
  - V list
  - Q list (jump-table)

* check
  - number of lines
  - BASIC : size
  - number of symbols
  - size of symbol table

The different routines are called with :

CALLM2000: ..... : ..... : .....

followed by normal BASIC-commands.

If a BASIC-command after CALLM2000 is not recognised by SPU, action is transferred to normal BASIC-execution.

SPU - commands :

LET OLD = NEW	rename variable
LET OLD = NEW : OUT 100,500	rename only in lines 100-500
RESTORE	CLEAR SYMBOL-TABLE
LIST	Q-list of total program
LIST 100	Q-list of line 100
LIST 100-500	Q-list from 100-500
MODE 0	variablen-atlas
MODE 1	variablen-atlas with line-numbers
WAIT start,end,new	renumber in STEP-mode
WAITMEM start,end,add	renumber in ADD-mode
DOT start,end,sub	renumber in SUB-mode
CHECK	programm-check-list

system-program-unit

1

```
10    REM ******************************
11    REM *  SYSTEM - PROGRAM - UNIT  *
12    REM *                           *
13    REM *                      DEMO  *
14    REM ******************************
15    MODE 6A:COLORG 0 5 10 15
20    W=RND(5.0)
21    ON W GOSUB 100,101,102,103
23    PRINT A,B,C," SUM : ";A+B+C
25    DRAW 50,0 50,A 21
26    DRAW 100,0 100,B 22
27    DRAW 150,0 150,C 23
30    IF A>200 OR B>200 OR C>200 THEN 500
36    WAIT TIME 20:GOTO 20
100   A=A+0.5:RETURN
101   B=B+0.5:RETURN
102   C=C+0.5:RETURN
103   D=D+0.5:RETURN
500   FILL 0,0 XMAX,YMAX 20:A=0.0:B=0.0:C=0.0:GOTO 20
```

2

```
CALLM2000:WAIT 10,500,10
```

```
10    REM ******************************
20    REM *  SYSTEM - PROGRAM - UNIT  *
30    REM *                           *
40    REM *                      DEMO  *
50    REM ******************************
60    MODE 6A:COLORG 0 5 10 15
70    W=RND(5.0)
80    ON W GOSUB 150,160,170,180
90    PRINT A,B,C," SUM : ";A+B+C
100   DRAW 50,0 50,A 21
110   DRAW 100,0 100,B 22
120   DRAW 150,0 150,C 23
130   IF A>200 OR B>200 OR C>200 THEN 190
140   WAIT TIME 20:GOTO 70
150   A=A+0.5:RETURN
160   B=B+0.5:RETURN
170   C=C+0.5:RETURN
180   D=D+0.5:RETURN
190   FILL 0,0 XMAX,YMAX 20:A=0.0:B=0.0:C=0.0:GOTO 70
```

3

```
*CALLM2000:CHECK
```

```
Datum :          Programm :

Lines   : 19
BASIC   : 502   Bytes
Symbols : 6
Table   : 43   Bytes
```

```
1 - the original dummy program
2 - renumber in step mode
3 - CHECK : programm information
```

4 `CALLM2000:WAITMEM 150,190,1000`

s.p.u.

```
10    REM ******************************
20    REM *  SYSTEM - PROGRAM - UNIT  *
30    REM *                           *
40    REM *                      DEMO *
50    REM ******************************
60    MODE 6A:COLORG 0 5 10 15
70    W=RND(5.0)
80    ON W GOSUB 1150,1160,1170,1180
90    PRINT A,B,C," SUM : ";A+B+C
100   DRAW 50,0 50,A 21
110   DRAW 100,0 100,B 22
120   DRAW 150,0 150,C 23
130   IF A>200 OR B>200 OR C>200 THEN 1190
140   WAIT TIME 20:GOTO 70
1150  A=A+0.5:RETURN
1160  B=B+0.5:RETURN
1170  C=C+0.5:RETURN
1180  D=D+0.5:RETURN
1190  FILL 0,0 XMAX,YMAX 20:A=0.0:B=0.0:C=0.0:GOTO 70
```

---

5 `*CALLM2000:MODE 0`

```
Datum :          Programm :

A !
B !
C !
D !
W !
X %
```

---

6 `*CALLM2000:MODE 1`

```
Datum :          Programm :

A !
                 90,100,130,1150,1190,
B !
                 90,110,130,1160,1190,
C !
                 90,120,130,1170,1190,
D !
                 1180,
W !
                 70,80,
X %
```

4 - renumber in ADD-mode
5 - V-list : variables-atlas
6 - Q-LIST : cross-reference

7
```
*CALLM2000:LIST
```

s.p.u.

```
Datum :          Programm :

  70:  140,1190,
1150:  80,
1160:  80,
1170:  80,
1180:  80,
1190:  130,
```

---

7 - JUMP-TABLE of entire programm
8 - where is a jump/call to 1190?
9 - rename 'W' to 'RANDOMVALUE'
10 - rename 'C' to 'THIRDBAR'

8
```
*CALLM2000:LIST1190
```

```
Datum :          Programm :

1190:  130,
```

9
```
*CALLM2000:LET W = RANDOMVALUE
*LIST
10    REM ******************************
20    REM *  SYSTEM - PROGRAM - UNIT  *
30    REM *                           *
40    REM *                      DEMO  *
50    REM ******************************
60    MODE 6A:COLORG 0 5 10 15
70    RANDOMVALUE=RND(5.0)
80    ON RANDOMVALUE GOSUB 1150,1160,1170,1180
90    PRINT A,B,C," SUM : ";A+B+C
100   DRAW 50,0 50,A 21
110   DRAW 100,0 100,B 22
120   DRAW 150,0 150,C 23
130   IF A>200 OR B>200 OR C>200 THEN 1190
140   WAIT TIME 20:GOTO 70
1150  A=A+0.5:RETURN
1160  B=B+0.5:RETURN
1170  C=C+0.5:RETURN
1180  D=D+0.5:RETURN
1190  FILL 0,0 XMAX,YMAX 20:A=0.0:B=0.0:C=0.0:GOTO 70
```

10
```
*CALLM2000:LET C=THIRDBAR
*LIST
10    REM ******************************
20    REM *  SYSTEM - PROGRAM - UNIT  *
30    REM *                           *
40    REM *                      DEMO  *
50    REM ******************************
60    MODE 6A:COLORG 0 5 10 15
70    RANDOMVALUE=RND(5.0)
80    ON RANDOMVALUE GOSUB 1150,1160,1170,1180
90    PRINT A,B,THIRDBAR," SUM : ";A+B+THIRDBAR
100   DRAW 50,0 50,A 21
110   DRAW 100,0 100,B 22
120   DRAW 150,0 150,THIRDBAR 23
130   IF A>200 OR B>200 OR THIRDBAR>200 THEN 1190
140   WAIT TIME 20:GOTO 70
1150  A=A+0.5:RETURN
1160  B=B+0.5:RETURN
1170  THIRDBAR=THIRDBAR+0.5:RETURN
1180  D=D+0.5:RETURN
1190  FILL 0,0 XMAX,YMAX 20:A=0.0:B=0.0:THIRDBAR=0.0:GOTO 70
```

11
```
*CALLM2000:MODE 1

Datum :          Programm :

A !
          ------------------------------------------------
          90,100,130,1150,1190,
B !
          ------------------------------------------------
          90,110,130,1160,1190,
C !
          ------------------------------------------------
D !
          ------------------------------------------------
          1180,
RANDOMVALUE !
          ------------------------------------------------
          70,80,
THIRDBAR !
          ------------------------------------------------
          90,120,130,1170,1190,
W !
          ------------------------------------------------
X %
          ------------------------------------------------
*
*
```

12
```
*CALLM2000:RESTORE

Datum :          Programm :

A !
          ------------------------------------------------
          90,100,130,1150,1190,
B !
          ------------------------------------------------
          90,110,130,1160,1190,
D !
          ------------------------------------------------
          1180,
RANDOMVALUE !
          ------------------------------------------------
          70,80,
THIRDBAR !
          ------------------------------------------------
          90,120,130,1170,1190,
*
```

13
```
*CALLM2000:WAIT10,1190,2000
RENUMBER RANGE  : 10-1190
NEW LINE NUMBER : 2000-2180
*
*LIST
2000  REM ******************************
2010  REM *  SYSTEM - PROGRAM - UNIT  *
2020  REM *                           *
2030  REM *                      DEMO  *
2040  REM ******************************
2050  MODE 6A:COLORG 0 5 10 15
2060  RANDOMVALUE=RND(5.0)
2070  ON RANDOMVALUE GOSUB 2140,2150,2160,2170
2080  PRINT A,B,THIRDBAR," SUM : ";A+B+THIRDBAR
2090  DRAW 50,0 50,A 21
2100  DRAW 100,0 100,B 22
2110  DRAW 150,0 150,THIRDBAR 23
2120  IF A>200 OR B>200 OR THIRDBAR>200 THEN 2180
2130  WAIT TIME 20:GOTO 2060
2140  A=A+0.5:RETURN
2150  B=B+0.5:RETURN
2160  THIRDBAR=THIRDBAR+0.5:RETURN
2170  D=D+0.5:RETURN
2180  FILL 0,0 XMAX,YMAX 20:A=0.0:B=0.0:THIRDBAR=0.0:GOTO 2060
```

---

11 - 'W' & 'RANDOMVALUE' are in the symbol table
'C' & 'THIRDBAR' are in the symbol table
12 - RESTORE : only actual symbols are preserved in the symbol table
13 - again RENUMBER in step-mode

CATALOG

NEW SOFTWARE :

DAI-PANIC : A super arcade game in machine language by Marco van Meegen.
Dig holes, kill monsters, run around... action and fun for hours.

price : audio : 800 Bfr
DCR : 950 Bfr

TOOLKIT 4 : - SYSTEM-PROGRAM-UNIT : the ultimate tool for serious programmers.
- EPSON screen copies in 5 grey-scales. (see cover of Newsletter 12).
- EPSON screen copies in 9 grey-scales. (see TV-test pattern in this issue)
- ARRAY to EDIT & EDIT to ARRAY

price : audio : 1000 Bfr
DCR : 1150 Bfr

Microcomputers in het onderwijs : BOOK (Dutch) + programs on tape

price : BOOK + audio : 990 Bfr
BOOK + DCR :1100 Bfr

SOON AVAILABLE :

SPL : a new MACRO-ASSEMBLER with many extra features. (a SPHYNX-production)

DCE-INTERFACE-CARD
: A low-cost solution to have many peripherals connected together to the DCE-bus.
Each card is addressable,following the DCE-concept, double-sided,metallised and has a prototyping area.
A universal motherboard from ELEKTUUR will be used to connect all cards.(EPS 80024)
The price of a card,including components will be around 2500 Bfr.
Please contact us if you have interest, ordering the components in larger quantities could bring the price down !

OLD SOFTWARE :

VIDITEL : H.DE VRIES has sold the copyrights of his VIDITEL program to INDATA.
This program will be no longer available from DAInamic.

---

program identification

title : HISTOIRES ALEATOIRES

author : J.MOENS

purpose : generates random messages

comment :

```
1     REM
2     REM HISTOIRES ALEATOIRES.
3     REM ---------------------
4     REM MOENS JACQUES - 09/81
5     REM ---------------------
10    CLEAR 2000:DIM A$(20.0),B$(20.0),C$(20.0),D$(20.0),E$(20.0),F$(20.0),G$(20
.0)
20    C=RND(14.99):IF C<10.0 THEN 20
21    COLORT C 0 0 0
30    PRINT CHR$(12):FOR I=1.0 TO 5.0
32    CURSOR 20,15:PRINT "H I S T O I R E S":CURSOR 20,13:PRINT "---------------
--":WAIT TIME 40
34    CURSOR 20,15:PRINT "                    ":CURSOR 20,13:PRINT "
  ":WAIT TIME 30:NEXT
40    FOR I=1.0 TO 10.0
42    READ A$(I),B$(I),C$(I),D$(I),E$(I),F$(I),G$(I)
44    NEXT I
50    PRINT :INPUT "VOULEZ-VOUS INTRODUIRE DES DONNEES (O/N) ";R$:IF R$="O" THEN
 65
52    IF R$="N" THEN 150
54    GOTO 60
60    PRINT CHR$(12):PRINT :INPUT "NOMBRE DE JOUEURS (2 A 10)";NJ
62    IF NJ<2 OR NJ>10 THEN 50
65    FOR I=11.0 TO 10.0+NJ
70    PRINT CHR$(12):PRINT :INPUT "UN PRENOM MASCULIN ";A$(I)
80    PRINT CHR$(12):PRINT :INPUT "UN PRENOM FEMININ ";B$(I)
90    PRINT CHR$(12):PRINT :INPUT "L'HEURE OU LE MOMENT ";C$(I)
100   PRINT CHR$(12):PRINT :INPUT "EN QUEL LIEU ";D$(I)
110   PRINT CHR$(12):PRINT :INPUT "QUE DIT LE MONSIEUR ";E$(I)
120   PRINT CHR$(12):PRINT :INPUT "QUE REPOND LA DAME ";F$(I)
130   PRINT CHR$(12):PRINT :INPUT "NOTRE CONCLUSION EST ";G$(I)
140   NEXT
150   L$=" RENCONTRE ":M$=" IL LUI DIT  : ":N$=" ELLE REPOND : ":P$=" CONCLUSION
  : "
165   PRINT CHR$(12):PRINT :INPUT "SI VOUS VOULEZ UNE HISTOIRE , APPUYEZ SUR 'RE
TURN' ";Y$:PRINT CHR$(12)
170   PRINT CHR$(12)
172   C=RND(14.99):IF C<10.0 THEN 172
174   COLORT C 0 0 0
180   GOSUB 390:A$=A$(I):GOSUB 390:B$=B$(I):GOSUB 390:C$=C$(I):GOSUB 390:D$=D$(I)
```

```
220   GOSUB 390:E$=E$(I):GOSUB 390:F$=F$(I):GOSUB 390:G$=G$(I)
260   PRINT :PRINT :PRINT " ";:PRINT A$;:PRINT L$;:PRINT B$;:PRINT " ";:PRINT C$
;:PRINT " ";:PRINT D$
262   PRINT :PRINT M$;:PRINT E$
264   PRINT :PRINT N$;:PRINT F$
266   PRINT :PRINT P$;:PRINT G$
270   PRINT :PRINT :INPUT " POUR CONTINUER APPUYEZ SUR 'RETURN'";Y$:GOTO 170
390   I=INT(RND(10.0+NJ)+1.0):RETURN
1000  DATA "LUC","CHANTAL","A LA PISCINE","A MIDI","IL FAIT FROID AUJOURD'HUI","
JE VOUS CROIS","EN AVRIL NE TE DECOUVRE PAS D'UN FIL"
1010  DATA "ERIC","CORINE","AU BAL","A MINUIT","JE VOUDRAIS DANSER AVEC VOUS","A
H! BON!","VIVE LES VACANCES !"
1020  DATA "JACQUES","RAPHAELLE","AU MARCHE","LE MATIN","JE CHERHE DES PETITS PO
IS","ALLEZ VOIR AU SUPERMARCHE"
1022  DATA "ON A TOUJOURS BESOIN DE PETITS POIS CHEZ SOI"
1030  DATA "CHRISTOPHE","BRIGITTE","DANS LE PETIT BOIS","L'APRES MIDI","J'AIME L
A NATURE","MOI,JE VOUDRAIS JOUER"
1032  DATA "RIEN NE SERT DE COURIR ..."
1040  DATA "CEDRIC","SARAH","DANS LA RUE","A 10 HEURES","J'AI MAL AUX PIEDS","AL
LEZ DONC VOIR LE MEDECIN !","AH ! LES FEMMES !"
1050  DATA "GUY","CHRISTIANE","AU CARNAVAL","A 3 HEURES","J'AIME LA FETE","MOI,J
E VOUDRAIS ALLER AU CIRQUE","VIVE LE CIRQUE"
1060  DATA "MICHEL","ISABELLE","AU CIRQUE","LE SOIR","JE VOUS AIME A LA FOLIE","
MOI,JE DOIS ALLER ETUDIER","VIVE LA NATURE"
1070  DATA "JEAN","ANNE","AU CINEMA","L'APRES-MIDI","IL FAIT CHAUD ICI","VOUS ET
ES UN SOT !","VIVEMENT LES VACANCES !"
1080  DATA "ALBERT","LILIANE","A LA POSTE","A 5 HEURES DU SOIR","JE NE SAIS PLUS
 QUEL JOUR NOUS SOMMES","EH BIEN ! VOUS ALORS !"
1082  DATA "QUAND ON EST BETE,C'EST POUR TOUTE SA VIE"
1090  DATA "PIERRE","SABINE","A L'ECOLE","A 8 HEURES","J'AI EU UN ACCIDENT HIER"
,"JE NE VOUS CROIS PAS !"
1092  DATA "SOYONS TOUJOURS PRUDENTS"
```

GRAPHICS M.DIERCKX

```
5     MODE 6
10    COLORG 0 0 0 0
20    FOR Q=1 TO XMAX STEP 3:DRAW 0,0 Q,YMAX 17+2*A:COLORG 0 15-15*A 15*A 15:DRA
W 0,0 Q-1,YMAX 18-2*A:A=1-A:NEXT
30    FOR Q=XMAX TO 1 STEP -3:DRAW XMAX,0 Q,YMAX 17+2*A:COLORG 0 15-15*A 15*A 15
:DRAW XMAX,0 Q-1,YMAX 18-2*A:A=1-A:NEXT
40    FOR Q=1 TO XMAX STEP 3:DRAW 0,YMAX Q,0 17+2*A:COLORG 0 15-15*A 15*A 15:DRA
W 0,YMAX Q-1,0 18-2*A:A=1-A:NEXT
50    FOR Q=XMAX TO 1 STEP -3:DRAW XMAX,YMAX Q,0 17+2*A:COLORG 0 15-15*A 15*A 15
:DRAW XMAX,YMAX Q-1,0 18-2*A:A=1-A:NEXT
70    FOR L=0 TO 15
80    FOR K=0 TO 15
90    FOR J=0 TO 15
100   FOR I=0 TO 15
110   WAIT TIME 5:COLORG I J K L
130   NEXT:NEXT:NEXT:NEXT:GOTO 70
```

```
PAGE 01    MODE 8 V0.1

002                     *       ************************************
003                     *       A PROGRAM BY N.P. LOOIJE
004                     *       PALUDANUSHOF 22
005                     *       3151 CM HOEK VAN HOLLAND
006                     *       ************************************
007                     *       244*512(528) RESOLUTIE
008                     *       4 KLEUREN ALS MODE 6,4,2
009                     *       XMAX,YMAX,DOT,DRAW,FILL,SCRN( )
010                     *         OFF SCREEN,COLOR NOT AVAILABLE
011                     *       EN ANIMATIE WORDEN GEINITIALISEERD
012                     *       HET SCHERM NEEM 32728 BYTES IN
013                     *       MODE 8A BESTAAT NIET DUS AAN HET
014                     *       EIND VAN HET PROGRAMMA BIJV.
015                     *       65355 IF GETC=0 THEN 65355:MODE 0
016                     *       ************************************
017                     DABYT   EQU   0         data
018                     LCBYT   EQU   :30       lijncontrolebyte
019                     CCBYT   EQU   :40       kleurcontrolebyte
020                     MODE6   EQU   :A
021                     LINES   EQU   :F4       244 lijnen
022                     SCRTOP  EQU   :BFEF     top scherm na header
023                             ORG   :300
024 0300 F5                     PUSH  PSW
025 0301 C5                     PUSH  B
026 0302 D5                     PUSH  D
027 0303 E5                     PUSH  H
028 0304 3E0A                   MVI   A,MODE6
029 0306 EF                     RST   5
030 0307 18                     DATA  :18       initialiseer mode 6
031 0308 21EFBF                 LXI   H,SCRTOP
032 030B 1EF4                   MVI   E,LINES
033 030D 1684           LOOP0   MVI   D,:84     vul controlebytes
034 030F 3630                   MVI   M,LCBYT
035 0311 2B                     DCX   H
036 0312 3640                   MVI   M,CCBYT
037 0314 2B             LOOP1   DCX   H         vul databytes
038 0315 3600                   MVI   M,DABYT
039 0317 15                     DCR   D
040 0318 C21403                 JNZ   LOOP1
041 031B 2B                     DCX   H
042 031C 1D                     DCR   E
043 031D C20D03                 JNZ   LOOP0
044 0320 215903         CTRL    LXI   H,MDETBL  init. current state screen
045 0323 118400                 LXI   D,:84
046 0326 0615                   MVI   B,:15
047 0328 7E             LOOP3   MOV   A,M       verplaats 21 bytes
048 0329 12                     STAX  D
049 032A 13                     INX   D
050 032B 23                     INX   H
051 032C 05                     DCR   B
052 032D C22803                 JNZ   LOOP3
053 0330 212740                 LXI   H,:4027
054 0333 22A502                 SHLD  :2A5      SCRBOT
```

```
055 0336 0610              MVI   B,:10
056 0338 21F0BF            LXI   H,:BFF0
057 033B 112840            LXI   D,:4028
058 033E 7E          LOOP4 MOV   A,M       kopieer header in trailer
059 033F 12                STAX  D
060 0340 13                INX   D
061 0341 23                INX   H
062 0342 05                DCR   B
063 0343 C23E03            JNZ   LOOP4
064 0346 3E3F              MVI   A,:3F      maak trailer af
065 0348 322B40            STA   :402B
066 034B 322F40            STA   :402F
067 034E 323340            STA   :4033
068 0351 323740            STA   :4037
069 0354 E1                POP   H
070 0355 D1                POP   D
071 0356 C1                POP   B
072 0357 F1                POP   PSW
073 0358 C9                RET
074 0359 2740        MDETBL DATA  :27,:40   *FFB
075 035B 77B0                 DATA  :77,:B0   *GRR
076 035D 3740                 DATA  :37,:40   *GRE
077 035F 2740                 DATA  :27,:40   *CHS
078 0361 3740                 DATA  :37,:40   *GAE
079 0363 2740                 DATA  :27,:40   *SCE
080 0365 7756                 DATA  :77,:56   *GTE
081 0367 EF56                 DATA  :EF,:56   *GTS
082 0369 0002                 DATA  :00,:02   *GRC
083 036B F4                   DATA  :F4       *GRL
084 036C 2C                   DATA  :2C       *GAL
085 036D 86                   DATA  :86       *GXB
086 036E                      END

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

CCBYT  0040   CTRL   0320   DABYT  0000   LCBYT  0030
LINES  00F4   LOOP0  030D   LOOP1  0314   LOOP3  0328
LOOP4  033E   MDETBL 0359   MODE6  000A   SCRTOP BFEF
```

```
0300 F5 C5 D5 E5 3E 0A EF 18 21 EF BF 1E F4 16 84 36
0310 30 2B 36 40 2B 36 00 15 C2 14 03 2B 1D C2 0D 03
0320 21 59 03 11 84 00 06 15 7E 12 13 23 05 C2 28 03
0330 21 27 40 22 A5 02 06 10 21 F0 BF 11 28 40 7E 12
0340 13 23 05 C2 3E 03 3E 3F 32 2B 40 32 2F 40 32 33
0350 40 32 37 40 E1 D1 C1 F1 C9 27 40 77 B0 37 40 27
0360 40 37 40 27 40 77 56 EF 56 00 02 F4 2C 86
```

OBJECT CODE MODE 8

```
0D00 F5 C5 D5 E5 3E 0A EF 18 21 EF BF 1E F4 16 84 36
0D10 30 2B 36 40 2B 36 00 15 C2 14 0D 2B 1D C2 0D 0D
0D20 21 59 0D 11 84 00 06 15 7E 12 13 23 05 C2 28 0D
0D30 21 27 40 22 A5 02 06 10 21 F0 BF 11 28 40 7E 12
0D40 13 23 05 C2 3E 0D 3E 3F 32 2B 40 32 2F 40 32 33
0D50 40 32 37 40 E1 D1 C1 F1 C9 27 40 77 B0 37 40 27
0D60 40 37 40 27 40 77 56 EF 56 00 02 F4 2C 86
```

OBJECT CODE MODE 8, RELOCATED ABOVE FBT

```
PAGE 01     MODE 7 V0.0

002                     *       **********************************
003                     *       A PROGRAM BY N.P. LOOIJE
004                     *       PALUDANUSHOF 22
005                     *       3151 CM HOEK VAN HOLLAND
006                     *       **********************************
007                     *       244*512(528) RESOLUTIE
008                     *       16 KLEUREN ALS MODE 5,3 EN 1
009                     *       XMAX,YMAX,DOT,DRAW,FILL,SCRN( )
010                     *         OFF SCREEN,COLOR NOT AVAILABLE
011                     *       WORDEN GEINITIALISEERD
012                     *       HET SCHERM NEEMT 32728 BYTES IN
013                     *       MODE 7A BESTAAT NIET DUS AAN HET
014                     *       EINDE VAN HET PROGRAMMA BIJV.
015                     *       65355 IF GETC=0 THEN 65355:MODE 0
016                     *       **********************************
017                     DBYTPL  EQU   :42       databytes per lijn
018                     DABYT   EQU   :FF       data
019                     LCBYT   EQU   :B0       lijncontrolebyte
020                     CCBYT   EQU   :40       kleurcontrolebyte
021                     MODE5   EQU   :8
022                     LINES   EQU   :F4       244 lijnen
023                     SCRTOP  EQU   :BFEF     top scherm na header
024                             ORG   :300
025 0300 F5                     PUSH  PSW
026 0301 C5                     PUSH  B
027 0302 D5                     PUSH  D
028 0303 E5                     PUSH  H
029 0304 3E08                   MVI   A,MODE5
030 0306 EF                     RST   5
031 0307 18                     DATA  :18       initialiseer mode 5
032 0308 2A9E00                 LHLD  :9E       COLORGregisters 0 en 1
033 030B 3E0F                   MVI   A,:F
034 030D A5                     ANA   L
035 030E 87                     ADD   A
036 030F 87                     ADD   A
037 0310 87                     ADD   A
038 0311 87                     ADD   A
039 0312 6F                     MOV   L,A
040 0313 7C                     MOV   A,H
041 0314 E60F                   ANI   :F
042 0316 85                     ADD   L         A bevat kleurbyte
043 0317 0EFF                   MVI   C,DABYT
044 0319 21EFBF                 LXI   H,SCRTOP
045 031C 1EF4                   MVI   E,LINES
046 031E 1642           LOOP0   MVI   D,DBYTPL  vul controlebytes
047 0320 36B0                   MVI   M,LCBYT
048 0322 2B                     DCX   H
049 0323 3640                   MVI   M,CCBYT
050 0325 2B             LOOP1   DCX   H         vul data & kleurbytes
051 0326 71                     MOV   M,C
052 0327 2B                     DCX   H
053 0328 77                     MOV   M,A
054 0329 15                     DCR   D
```

```
055 032A C22503             JNZ   LOOP1
056 032D 2B                 DCX   H
057 032E 1D                 DCR   E
058 032F C21E03             JNZ   LOOP0
059 0332 216B03     CTRL    LXI   H,MDETBL  init. current state screen
060 0335 118400             LXI   D,:84
061 0338 0615               MVI   B,:15
062 033A 7E         LOOP3   MOV   A,M       verplaats 21 bytes
063 033B 12                 STAX  D
064 033C 13                 INX   D
065 033D 23                 INX   H
066 033E 05                 DCR   B
067 033F C23A03             JNZ   LOOP3
068 0342 212740             LXI   H,:4027
069 0345 22A502             SHLD  :2A5      SCRBOT
070 0348 0610               MVI   B,:10
071 034A 21F0BF             LXI   H,:BFF0
072 034D 112840             LXI   D,:4028
073 0350 7E         LOOP4   MOV   A,M       kopieer header in trailer
074 0351 12                 STAX  D
075 0352 13                 INX   D
076 0353 23                 INX   H
077 0354 05                 DCR   B
078 0355 C25003             JNZ   LOOP4
079 0358 3EBF               MVI   A,:BF     maak de trailer af
080 035A 322B40             STA   :402B
081 035D 322F40             STA   :402F
082 0360 323340             STA   :4033
083 0363 323740             STA   :4037
084 0366 E1                 POP   H
085 0367 D1                 POP   D
086 0368 C1                 POP   B
087 0369 F1                 POP   PSW
088 036A C9                 RET
089 036B 2740       MDETBL  DATA  :27,:40   *FFB
090 036D 77B0               DATA  :77,:B0   *GRR
091 036F 3740               DATA  :37,:40   *GRE
092 0371 2740               DATA  :27,:40   *CHS
093 0373 3740               DATA  :37,:40   *GAE
094 0375 2740               DATA  :27,:40   *SCE
095 0377 7756               DATA  :77,:56   *GTE
096 0379 EF56               DATA  :EF,:56   *GTS
097 037B 0002               DATA  :00,:02   *GRC
098 037D F4                 DATA  :F4       *GRL
099 037E 2C                 DATA  :2C       *GAL
100 037F 86                 DATA  :86       *GXB
101 0380                    END

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

CCBYT  0040   CTRL   0332   DABYT  00FF   DBYTPL 0042
LCBYT  00B0   LINES  00F4   LOOP0  031E   LOOP1  0325
LOOP3  033A   LOOP4  0350   MDETBL 036B   MODE5  0008
SCRTOP BFEF
```

continued from page 378

In Equation 5, ROS is the One-Shot Control Resistor (Pin 24) and COS is the One-Shot Control Capacitor (Pin 23). Maximum duration of the One-Shot is approximately 10.0 seconds. When the One-Shot is controlled by external logic, the One-Shot Control Resistor and Capacitor may be eliminated. Simply begin One-Shot with Pin 9 (system enable) and end cycle by taking Pin 23 (One-Shot Capacitor) high.

8. ENVELOPE SELECT LOGIC

The Envelope Select Logic determines the envelope that is applied to the mixer output according to the following table

Envelope Select 1	Envelope Select 2	Selected Function
Pin 1	Pin 28	
0	0	VCO
0	1	Mixer Only
1	0	One-Shot
1	1	VCO with Alternating Polarity

Table 4: Envelope Select Logic Output

9. ATTACK AND DECAY CONTROL

The Attack/Decay circuitry alters the rise and fall of the envelope. An example of a noise waveform utilizing the envelope generator under one-shot control is:

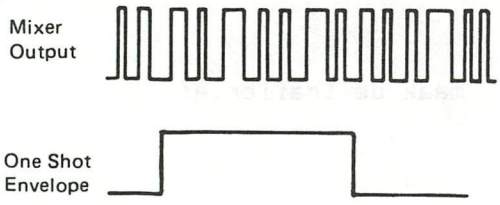


Figure 4: One-Shot Controlled Noise Waveform

By utilizing the Attack and Decay Control Inputs (Pin 7,10), the waveform may be affected in the following manner:

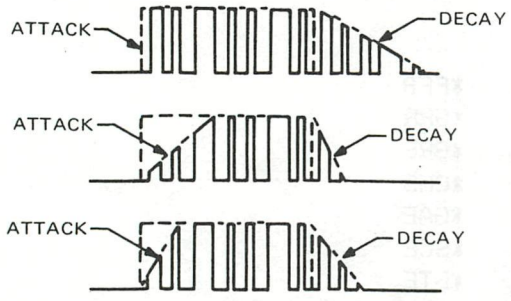


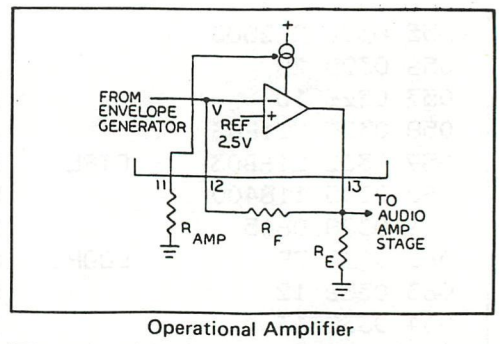
Figure 5: Examples of Varying Degrees of Attack and Decay on a Waveform

The amount of Attack and Decay is determined by the Attack Control Resistor (RA) (Pin 10) and the Decay Control Resistor (RD) (Pin 7) and the Attack Decay Timing Capacitor (CA-D) (Pin 18) According to the following equations:

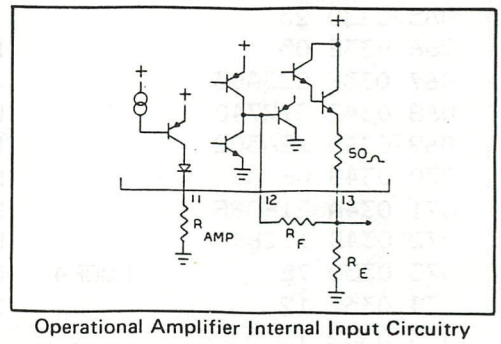
Attack Time (seconds) ≈ RA CA—D
Decay Time (seconds) ≈ RA CA—D
@ VREG = 5.0V

10. OUTPUT AMPLIFIER

The output amplifier is a gain section designed to interface with external sound modulators or additional amplifier stages. The output is an operational amplifier operating as a summer and invertor, as illustrated. The output is an emitter-follower without a load resistor. Therefore, pin 13 should have a pull-down resistor, RE, with a value ranging from 2.7K to 10K ohms. The equivalent of the input circuitry for the amplifier section is shown in the next column.



Operational Amplifier



Operational Amplifier Internal Input Circuitry

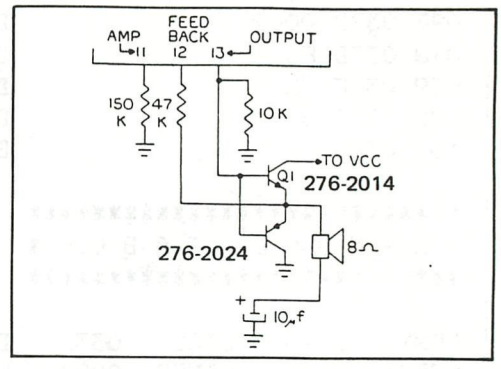
The resistor value RAMP sets the operating currents for the operational amplifier's internal circuitry and is the main adjustment to control the amplifier's output amplitude. The value of this resistor is normally between 47K and 220K ohms. Any lower resistance will typically begin to saturate the operational amplifier and is especially noticeable on the decay portion of the sound envelope.

The feedback resistor, RE, is used to compensate for external variations and also any chip-to-chip variations. This is accomplished by connecting the feedback resistor between the last amplifier stage and the input Pin 12, as shown below. The feedback resistor is connected to the last stage at a point where the signal is in-phase with the operational amplifier's output. The peak output voltage is determined by the following equation:

$$V_{OUT} \simeq \frac{3.4R_F}{RAMP} \qquad @ V_{REG} = 5.0V$$

Where VOUT = volts
R = ohms

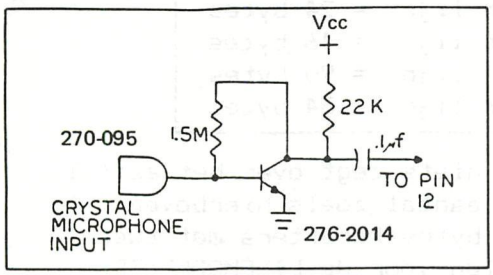
The dynamic output range is limited to 2.5 volts peak-to-peak before clipping occurs. The example shown is ideally suited for most applications. The amplifier is in a push-pull configuration and will draw current only when a signal is present. Depending on the voltage applied to the collector of Q1, this circuit will provide approximately 300-400mW of power into an 8-ohm speaker.



If the amplitude of the sound output is to be varied for particular sounds, the resistance RAMP can be varied by logic control lines. This can be done (as described earlier) by using the logic control line to switch a logic gate that will put a resistor in parallel with RAMP.

Special filtering can be added to the output of the amplifier or can be included in the feedback section of the operational amplifier. Since the output of the amplifier always contains square waves, filtering will change the timbre (harmonic content) of the output signal.

Other external sounds may be added to the input of the amplifier at Pin 12. This input can be made either directly or through a series resistor. An example of an input configuration to add a person's voice to the system is shown below. This could be used to sing along or talk along with the sounds being generated by the chip.



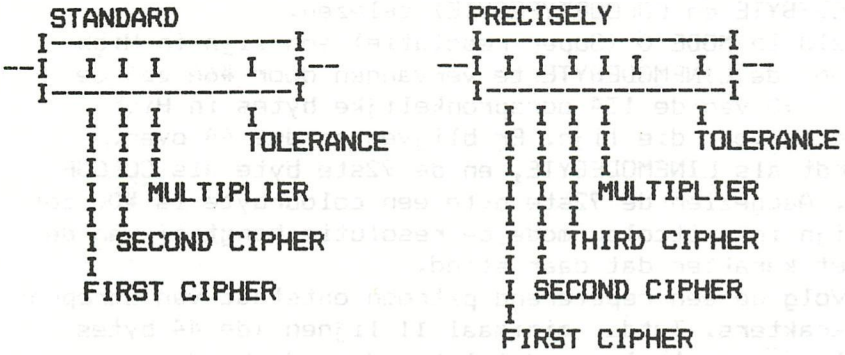
11. REGULATOR

Either a 5-volt regulated supply may be applied to Pin 15 (VREG) or a 7.5-volt min/9.0 volt max regulated supply may be applied to Pin 14 (VCC). Pin 15 (VREG) can be used as a 5-volt regulated supply for the rest of the system with a current supply of up to 10MA out of the IC.

12. NOTE:

Control resistors and capacitors may be eliminated if the desired sound does not require that generator or logic section. For dedicated sound, the logic inputs (Pins 1, 9, 22, 25, 26, 27, 28) may be hard-wired for high or low logic levels. Individual sounds (single or multiple) will determine which of the other components are required.

COLOR-CODE FOR RESISTORS



COLOR	FIRST CIPHER	SECOND CIPHER	THIRD CIPHER	MULTIPLIER	TOLERANCE
BLACK	0	0	0	1	-
BROWN	1	1	1	10	1%
RED	2	2	2	100	2%
ORANGE	3	3	3	1000	-
YELLOW	4	4	4	10,000	-
GREEN	5	5	5	100,000	0.5%
BLUE	6	6	6	1,000,000	-
VIOLET	7	7	7	10,000,000	-
GREY	8	8	8	-	-
WHITE	9	9	9	-	-
GOLD	-	-	-	0.1	5%
SILVER	-	-	-	0.01	10%

TABLE 4

Hoek van Holland 17 November 1982

Beste DAI vrienden,

Sinds het laatste nummer van Dainamic heb ik een aantal reacties gehad van leden o.a. over de unitcolourmode en tekstmodes. Het blijkt dat het handboek toch niet duidelijk en volledig genoeg is.
Daarom als aanvulling op voorgaand artikel deze brief en een demoprogramma

Aan de hand van het handboek kunnen we zijn dat de DAI acht resoluties kent nl.

> Unitcolourmode	4 resoluties	= 4 bytes
> Low resolution	88 blobs per lijn	= 24 bytes
> Medium resolution	176 blobs per lijn	= 46 bytes
> High resolution	352 blobs per lijn	= 90 bytes
> Super resolution	528 blobs per lijn	=134 bytes

De eerste onvolledigheid is dat het handboek niets zegt over het aantal bytes per lijn. Normaal gesproken is dit het aantal zoals hierboven vermeld. Dit is te berekenen door het aantal bytes/karakters met twee te vermenigvuldigen en er twee bij op te tellen voor de LINEMODEBYTE en de COLOURTYPEBYTE. Over de uitzonderingen verderop meer.
De uitlezing door de videohardware geschiedt (vereenvoudigd) als volgt;
Eerst wordt de uitlezing geladen met het hoogste RAM adres en twee instructies gelezen (de LINEMODE en COLOURTYPEBYTE).
Aan de hand hiervan wordt een teller geladen met het aantal nog te lezen bytes tot de volgende instructies. Tevens wordt de uitlees snelheid, het aantal herhalingen, de kleurmode, en karakter of graphische mode vastgesteld.Als de teller afgelopen is wordt op het volgende RAM adres (Top RAM - lijnlengte) worden dan de volgende instructies (LINEMODEBYTE en COLOURTYPEBYTE) gelezen.
Als je bijvoorbeeld in MODE 0 (Super resolutie) een lijn in High resolution zet door de LINEMODEBYTE te vervangen door #6A zal de videohardware maar 90 van de 134 oorspronkelijke bytes in High resolution uit lezen voor die lijn. Er blijven er dus 44 over.
De 91ste byte wordt als LINEMODEBYTE, en de 92ste byte als COLOURTYPEBYTE gelezen. Aangezien de 92ste byte een colourbyte is #00 zet de hardware de lijn in unitcolourmode de resolutie hangt af van de 91ste byte nl. het karakter dat daar stond.
Dit heeft als gevolg de een repeterend patroon ontstaat van strepen of stukken van karakters. Totdat minimaal 11 lijnen (de 44 bytes opgedeeld in 11 lijnen unitcolourmode) later de videohardware weer de LINEMODEBYTE van de volgende regel tegenkomt.
Deze patronen zijn dan te wissen door COLORT x y x x, maar het is niet weg. Dit is te merken doordat alle regels naar onder geschoven zijn. In alle DEMO's van grote karakters wordt deze techniek gebruikt dit heeft als voordeel dat de screendriver netjes op de volgende regel verder gaat. Het nadeel is dat het aantal regels wordt beperkt en meestal maar twee kleuren mogelijk zijn.
Om meer regels grote letters te krijgen moet je het scherm opnieuw in delen. Hier blijkt de tweede onvolledigheid van het handboek er zijn nogal wat uitzonderingen op de hierboven gemaakte rekensommetjes voor het aantal bytes per lijn. Ook de unitcolourmode wordt niet uitvoerig behandeld daarom onderstaande opmerkingen en tabel.

UITZONDERINGEN & BIJZONDERHEDEN

1. COLOURBYTE
   Er is een belangrijk verschil tussen karakter en graphicmodes nl. de plaats van de COLOURBYTE.
   GRAPHICS >ADRES COLOURBYTE = ADRES DATABYTE-1
   KARAKTERS >ADRES COLOURBYTE = ADRES DATABYTE-3
   dit houdt in dat het laatste karakter van een regel de kleurin-

formatie haalt uit de COLOURTYPEBYTE van de volgende regel.

2. LOW en MEDIUM resolutie
   KARAKTERS >LOW res. 12 databytes totaal 26 bytes per lijn
   MEDIUM res. 23 databytes totaal 48 bytes per lijn
   GRAPHICS >LOW res. 11 databytes totaal 24 bytes per lijn
   MEDIUM res. 22 databytes totaal 46 bytes per lijn
   dit is dus afwijkend voor de karakters.

3. ONZICHTBARE KARAKTERS
   In alle graphicmodes en Unitcolourmodes is de gehele lijn zichtbaar. In de charactermodes BEHALVE High res. is het op een na laatste karakter gedeeltelijk zichtbaar en het laatste karakter onzichtbaar. In de High res. is aan het einde het laatste karakter half zichtbaar, deze haalt zijn kleurinformatie uit de COLOURTYPEBYTE van de volgende regel. Dit is bij een totaal plaatje zichtbaar als een blokje in kleur 2 van het COLORT commando. (Op de meeste TV's zijn de laatste karakters niet zichtbaar.)

4. 16 KLEUREN KARAKTERS
   Vanaf rev. (5 of 6) computers is deze modificatie standaard ingebouwd. Deze modificatie heeft dezelfde beperking als de 16 kleuren graphics nl. de videohardware moet in een karakterbyte eerst een logische "1" tegen komen om de achtergrondkleur te veranderen. bij POKE's moet men hier rekening mee houden.
   Bijv. CHR$(#7F)(volle blok)als eerste karakter op de lijn en de COLOURBYTE 17 x achtergrondkleur.
   Bijv. FA 40 7F 88 44 xx 41 yx
   ^ ^ ^ ^ ^ ^ ^
   LCB CTB DATA1 DATA2 COL1 DATA3 COL2
   Dit zet de lijn in kleur x en het karakter op kleur y.

5. UNITCOLOURMODE
   De unitcolourmode kent twee manieren van uitlezing door de hardware nl. voor MEDIUM/LOW en HIGH/SUPER resolutie.
   In beide gevallen zijn 4 bytes per lijn nodig nl. LINEMODE- en COLOURTYPEBYTE + DATA + COLOURBYTE.
   In LOW/MEDIUM res. is afstand tussen de LINEMODEBYTEs twee bytes, in de HIGH/SUPER is de afstand 4 bytes. In het eerste geval is dus de DATA- & COLOURBYTE van de eerste regel resp. de LINEMODE- & COLOURTYPEBYTE van de tweede regel dus altijd een vast patroon. In het tweede geval zijn de DATA en de COLOURBYTE naar keuze in te vullen.

Hieronder een overzicht van het effect dat CONTROLEBYTES te weeg brengen.

CONTROLE BYTES	DISPLAY	COLORS	RESOLUTIE	BYTES TOT NIEUW CONTROLEBYTE	DATA & COLOUR BYTES	ZICHTBARE FIELDS & BLOBS
0x 40	GRAPHICS	4	LOW	24	11	11 - 88
1x 40	GRAPHICS	4	MEDIUM	46	22	22 -176
2x 40	GRAPHICS	4	HIGH	90	44	44 -352
3x 40	GRAPHICS	4	SUPER	134	66	66 -528
4x 40	CHAR	4	LOW	26	12	10 - 87
5x 40	CHAR	4	MEDIUM	48	23	21 -173
6x 40	CHAR	4	HIGH	90	44	43 -345
7x 40	CHAR	4	SUPER	134	66	64 -519
8x 40	GRAPHICS	16	LOW	24	11	11 - 88
9x 40	GRAPHICS	16	MEDIUM	46	22	22 -176
Ax 40	GRAPHICS	16	HIGH	90	44	44 -352
Bx 40	GRAPHICS	16	SUPER	134	66	66 -528
Cx 40	CHAR	16	LOW	26	12	10 - 87
Dx 40	CHAR	16	MEDIUM	48	23	21 -173

```
Ex 40     CHAR      16        HIGH      90        44          43 -345
Fx 40     CHAR      16        SUPER     134       66          64 -519
************************UNIT COLOUR MODE ****************************
0x 00     GRAPHICS  4         LOW       2         1           11 - 88
1x 00     GRAPHICS  4         MEDIUM    2         1           22 -176
2x 00     GRAPHICS  4         HIGH      4         1           44 -352
3x 00     GRAPHICS  4         SUPER     4         1           66 -528
4x 00     CHAR      4         LOW       2         1           11 - 88
5x 00     CHAR      4         MEDIUM    2         1           22 -176
6x 00     CHAR      4         HIGH      4         1           44 -352
7x 00     CHAR      4         SUPER     4         1           66 -528
8x 00     GRAPHICS  16        LOW       2         1           11 - 88
9x 00     GRAPHICS  16        MEDIUM    2         1           22 -176
Ax 00     GRAPHICS  16        HIGH      4         1           44 -352
Bx 00     GRAPHICS  16        SUPER     4         1           66 -528
Cx 00     CHAR      16        LOW       2         1           11 - 88
Dx 00     CHAR      16        MEDIUM    2         1           22 -176
Ex 00     CHAR      16        HIGH      4         1           44 -352
Fx 00     CHAR      16        SUPER     4         1           66 -528
```

Dit zijn de juiste waarden voor een correct opbouwen van het scherm. Bijgaand ook een demonstatieprogramma met alle mogelijkheden.

met vriendelijke groeten
N.P. Looije
Paludanushof 22
3151 CM Hoek van Holland

```
1     REM LINEMODEBYTES EN COLOURTYPEBYTES/N.P.LOOIJE
10    MODE 0:PRINT CHR$(12):COLORT 5 15 5 5
20    CURSOR 0,17:PRINT " DAI PERSONAL                    COMPUTER"
30    POKE #BCCB,#5F:POKE #BC9B,#5F:POKE #BC9A,#40
40    IF GETC=0 THEN 40
50    COLORT 8 0 3 9:PRINT CHR$(12);"BASIC V6.0":POKE #75,#5F
60    PRINT "*";
70    IF GETC=0 THEN 70
80    P=0:GOSUB 8000
100   A$="****************************************"
110   BYTE=#BF4F:S=0:B$=A$+"**":S$=B$:GOSUB 9000
120   BYTE=BYTE-#88:S$=A$:S=1:GOSUB 9000
130   BYTE=BYTE-#86:S$="*      24  LINES   OF  CHARACTERS      *":GOSUB 9000
140   BYTE=BYTE-#86:S$=A$:GOSUB 9000
160   BYTE=BYTE-#84:S$=B$:S=0:GOSUB 9000:S=1
200   A$="****************************"
210   BYTE=#BC5F:S$=A$:GOSUB 9000
220   BYTE=BYTE-#5A:S$="*    IN  4  RESOLUTIONS    *":GOSUB 9000
230   IF P=1 THEN BYTE=BYTE-#5A:S$="*      AND 16 COLOURS      *":GOSUB 9000
240   BYTE=BYTE-#5A:S$=A$:GOSUB 9000
250   BYTE=BYTE-#B4+P*#5A:S=0:S$=" PRESS SPACE TO CONTINUE":GOSUB 9000:S=1
300   BYTE=#BA75:S$="**************":GOSUB 9000
310   BYTE=BYTE-#C0:GOSUB 9000
320   BYTE=BYTE+#90:S$="*PROGRAMMABLE*":GOSUB 9000
```

```
330   BYTE=BYTE-#30:S$="* GRAPHICS   *":GOSUB 9000
340   BYTE=BYTE-#30:S$="*  GENERATOR *":GOSUB 9000
400   BYTE=#B96F:S$="*******":GOSUB 9000
410   BYTE=BYTE-#34:GOSUB 9000
420   BYTE=BYTE+#1A:S$="*     *":GOSUB 9000
430   BYTE=BYTE-4:S=0:S$="DAI":GOSUB 9000:S=1
440   BYTE=BYTE-#54:S$="NPL":GOSUB 9000
500   IF GETC<>0 THEN 530
510   IF P=0 THEN SS=1-SS:SSS=1-SS:COLORT SS*3+SSS*8 SS*9 SS*8+SSS*3 SSS*9
520   WAIT TIME 100:GOTO 500
530   IF P=1 THEN 1000
600   COLORT 15 0 0 0:PRINT CHR$(12)
610   P=1:POKE #75,32:GOSUB 8000:GOTO 100
1000  CLEAR 5000:MODE 0:PRINT CHR$(12):COLORT 0 14 5 15
1010  DIM DISPLAYMODE$(3),COLOURMODE(1),RESOLUTION$(3),UCM$(1)
1020  FOR A=0 TO 1:READ DISPLAYMODE$(A),COLOURMODE(A),UCM$(A):NEXT
1030  FOR A=0 TO 3:READ RESOLUTION$(A):NEXT
1040  DATA GRAPHICS,4,UNITCOLOURMODE,CHARACTERS,16,,LOW,MEDIUM,HIGH,SUPER
1050  FOR A=#0 TO #1F0 STEP #10:LINEMODEBYTE=(A+#A) IAND #FF:COLOURTYPEBYTE=#40-
A/#100*#40
1060  READ BYTESPERLINE,COLOURBYTE,BLOBS,OPM$
1070  PRINT CHR$(12);
1080  PRINT RESOLUTION$((LINEMODEBYTE IAND #30) SHR 4);" RESOLUTION ";
1090  PRINT COLOURMODE((LINEMODEBYTE IAND #80) SHR 7);" COLOUR ";
1100  PRINT DISPLAYMODE$((LINEMODEBYTE IAND #40) SHR 6);" ";
1110  PRINT UCM$((COLOURTYPEBYTE IAND #40) SHR 6)
1120  PRINT "LINEMODEBYTE      =#";HEX$(LINEMODEBYTE)
1130  PRINT "COLOURTYPEBYTE    =#";HEX$(COLOURTYPEBYTE)
1140  PRINT "BYTES PER LINE    =#";HEX$(BYTESPERLINE);" =";BYTESPERLINE
1145  PRINT "VISIBLE BLOBS     =";BLOBS
1150  COLOUROFFSET=-1:IF LINEMODEBYTE IAND #40=#40 THEN COLOUROFFSET=-3
1170  PRINT "ADRES COLOURBYTE   = ADRES DATABYTE";COLOUROFFSET
1180  PRINT OPM$:GOSUB 10000
1190  NEXT
1200  PRINT CHR$(12):END
8000  FOR A=#BFEF TO #BD51 STEP -#86:POKE A,#7A+P*#80:POKE A-1,#40:POKE A-2,#20+
#5F*P:POKE A-5,#FF*P:NEXT
8010  FOR A=#BCCB TO #BB09 STEP -#5A:POKE A,#6A+P*#80:POKE A-1,#40:POKE A-2,#20+
#5F*P:POKE A-5,#FF*P:NEXT
8020  FOR A=#BAAF TO #B9BF STEP -#30:POKE A,#5A+P*#80:POKE A-1,#40:POKE A-2,#20+
#5F*P:POKE A-5,#FF*P:NEXT
8030  FOR A=#B98F TO #B8F3 STEP -#1A:POKE A,#4A+P*#80:POKE A-1,#40:POKE A-2,#20+
#5F*P:POKE A-5,#FF*P:NEXT
8040  FOR A=#BC45 TO #B921 STEP -#86:POKE A,#20:POKE A-1,0:NEXT
8050  RETURN
9000  ADRES=BYTE-2*LEN(S$)+2:FOR A=LEN(S$)-1 TO 0 STEP -1
9010  COLOR=#FF:IF P=1 THEN S=1:COLOR=INT(RND(16))*#10+#F:IF COLOR=#FF THEN 9010
9020  POKE ADRES,ASC(MID$(S$,A,1)):POKE ADRES-3,S*COLOR
9030  ADRES=ADRES+2:NEXT:RETURN
10000 CURSOR 0,0
10010 FOR LINESTART=#B9A7 TO #B9A7-12*BYTESPERLINE STEP -BYTESPERLINE
10020 POKE LINESTART,LINEMODEBYTE
10030 POKE LINESTART-1,COLOURTYPEBYTE
10040 FOR CHARBYTE=LINESTART-BYTESPERLINE+2 TO LINESTART-2 STEP 2
10050 IF COLOURTYPEBYTE IAND #40=0 AND LINEMODEBYTE IAND #30<#20 THEN CHARACTER=
LINEMODEBYTE:GOTO 10100
10060 IF LINEMODEBYTE IAND #40=0 THEN CHARACTER=RND(256):COLOURBYTE=RND(256):GOT
O 10110
10070 CHARACTER=RND(128)
10080 IF LINEMODEBYTE IAND #80=0 THEN COLOURBYTE=COLOURBYTE IXOR #FF:GOTO 10110
10090 COLOURBYTE=RND(16) SHL 4:IF COLOURBYTE=0 THEN 10090
10100 IF COLOURTYPEBYTE IAND #40=0 THEN COLOURBYTE=COLOURBYTE IAND #BF:POKE LINE
START-1,COLOURBYTE
10110 POKE CHARBYTE,CHARACTER
10120 POKE CHARBYTE-1,COLOURBYTE
10130 NEXT CHARBYTE
```

```
10140 NEXT LINESTART
10150 C0=0:C1=14:C2=5:C3=15
10160 IF GETC<>0 THEN RETURN
10170 C=C1:C1=C2:C2=C3:C3=C:COLORT C0 C1 C2 C3
10180 WAIT TIME 20:GOTO 10160
60000 DATA #18,#FF,88,
60010 DATA #2E,#FF,176,
60020 DATA #5A,#FF,352,
60030 DATA #86,#FF,528,
60040 DATA #1A,#FF,87,12! CHARACTERS LAST ONE NOT VISIBLE
60050 DATA #30,#FF,173,23! CHARACTERS LAST ONE NOT VISIBLE
60060 DATA #5A,#FF,345,LAST COLOURBYTE IS COLOURTYPEBYTE OF NEXT LINE
60070 DATA #86,#FF,519,66 CHARACTERS LAST ONE NOT VISIBLE
60080 DATA #18,#FF,88,
60090 DATA #2E,#FF,176,
60100 DATA #5A,#FF,352,
60110 DATA #86,#FF,528,
60120 DATA #1A,#FF,87,12! CHARACTERS LAST ONE NOT VISIBLE
60130 DATA #30,#FF,173,23! CHARACTERS LAST ONE NOT VISIBLE
60140 DATA #5A,#FF,345,LAST COLOURBYTE IS COLOURTYPEBYTE OF NEXT LINE
60150 DATA #86,#FF,519,66 CHARACTERS LAST ONE NOT VISIBLE
60160 DATA #4,#0,88,NEXT LINE STARTS AT LINEMODEBYTE -2
60170 DATA #4,#0,176,NEXT LINE STARTS AT LINEMODEBYTE -2
60180 DATA #4,#0,352,
60190 DATA #4,#0,528,
60200 DATA #4,#0,88,NEXT LINE STARTS AT LINEMODEBYTE -2
60210 DATA #4,#0,176,NEXT LINE STARTS AT LINEMODEBYTE -2
60220 DATA #4,#0,352,
60230 DATA #4,#0,528,
60240 DATA #4,#F0,88,NEXT LINE STARTS AT LINEMODEBYTE -2
60250 DATA #4,#F0,176,NEXT LINE STARTS AT LINEMODEBYTE -2
60260 DATA #4,#F0,352,
60270 DATA #4,#F0,528,
60280 DATA #4,#F0,88,NEXT LINE STARTS AT LINEMODEBYTE -2
60290 DATA #4,#F0,176,NEXT LINE STARTS AT LINEMODEBYTE -2
60300 DATA #4,#F0,352,
60310 DATA #4,#F0,528,
```

OBJECT CODE MODE 7

```
0300 F5 C5 D5 E5 3E 08 EF 18 2A 9E 00 3E 0F A5 87 87
0310 87 87 6F 7C E6 0F 85 0E FF 21 EF BF 1E F4 16 42
0320 36 B0 2B 36 40 2B 71 2B 77 15 C2 25 03 2B 1D C2
0330 1E 03 21 6B 03 11 84 00 06 15 7E 12 13 23 05 C2
0340 3A 03 21 27 40 22 A5 02 06 10 21 F0 BF 11 28 40
0350 7E 12 13 23 05 C2 50 03 3E BF 32 2B 40 32 2F 40
0360 32 33 40 32 37 40 E1 D1 C1 F1 C9 27 40 77 B0 37
0370 40 27 40 37 40 27 40 77 56 EF 56 00 02 F4 2C 86
```

OBJECT CODE MODE 7, RELOCATED ABOVE FGT

```
0D00 F5 C5 D5 E5 3E 08 EF 18 2A 9E 00 3E 0F A5 87 87
0D10 87 87 6F 7C E6 0F 85 0E FF 21 EF BF 1E F4 16 42
0D20 36 B0 2B 36 40 2B 71 2B 77 15 C2 25 0D 2B 1D C2
0D30 1E 0D 21 6B 0D 11 84 00 06 15 7E 12 13 23 05 C2
0D40 3A 0D 21 27 40 22 A5 02 06 10 21 F0 BF 11 28 40
0D50 7E 12 13 23 05 C2 50 0D 3E BF 32 2B 40 32 2F 40
0D60 32 33 40 32 37 40 E1 D1 C1 F1 C9 27 40 77 B0 37
0D70 40 27 40 37 40 27 40 77 56 EF 56 00 02 F4 2C 86
```

program identification

title	:	DEMO FGT -how to use-
author	:	Bruno Van Rompaey
purpose	:	
comment	:	

FGT - TOELICHTINGEN
Presentatie van een programmapakket met FGT :
dit pakket bestaat steeds uit 3 delen:
1. BOOTSTRAP LOADER
2. FGT PROGRAMMA+TABEL
3. UW BASIC-PROGRAMMA DAT FGT GEBRUIKT
BOOTSTRAP-LOADER: dit is een BASIC-programma.
De uitvoering van dit programma heeft tot gevolg dat
het erop volgende FGT-programma wordt geladen, gevolgd
door het laden en uitvoeren van uw BASIC-programma.
Meer info over de DAInamic Bootstrap Loader: DBL
DAInamic nr 9: blz.43 tot en met 45.
We illustreren met een voorbeeld:
Tik : LOAD:RUN
Het programma dat geladen wordt is de DBL,gevolgd
door het FGT-programma en een BASIC-programma.
Dit BASIC-programma demonstreert het gebruik van de
FGT-parameters: meer informatie in de FGT-tekst.

```
10    MODE 0:PRINT CHR$(12)
20    COLORG 8 3 5 0:COLORT 8 0 8 8
25    POKE #BF69,#6A:CURSOR 3,22:PRINT "FGT - TOELICHTINGEN - diDAIsoft"
26    CURSOR 4,21:PRINT "LEES ONDERSTAANDE TEKST, DRUK SPACE, KIJK EN WACHT"
30    CURSOR 5,19:PRINT "Het programmadeel dat uitgevoerd wordt is :"
40    PRINT :PRINT
50    LIST 100-180
60    GOSUB 12000
100   MODE 6
110   A$="diDAIsoft":REM tekst die geschreven wordt
120   X%=50:REM x-coordinaat : begin tekst
130   Y%=70:REM y-coordinaat : begin tekst
140   CC%=21:REM kleur waarin de tekst geschreven wordt
150   DF%=1:REM vergrotingsfactor van de tekst
160   SP%=6:REM factor voor de ruimte tussen de karakters
170   GOSUB 10000:REM routine 10000-10050 maakt uitvoering
180   REM van het FGT-machinetaalprogramma mogelijk
190   GOSUB 11000
210   CURSOR 10,15:PRINT "We wijzigen X%=50 in X%=200."
220   CURSOR 10,14:PRINT "Alle andere FGT-parameters blijven dezelfde"
230   CURSOR 10,13:PRINT "en moeten dus niet meer worden opgegeven."
240   CURSOR 10,12:PRINT "Het uitgevoerde programmadeel is :"
245   PRINT :PRINT
250   LIST 300-310
260   GOSUB 12000
```

```
300   X%=200:REM wijziging x-coordinaat
310   GOSUB 10000
320   GOSUB 11000
340   CURSOR 10,15:PRINT "We wijzigen Y%=70 in Y%=200 en"
345   CURSOR 10,14:PRINT "de kleur CC%=21 in CC%=22."
350   CURSOR 10,13:PRINT "Het uitgevoerde programmadeel is :"
355   PRINT :PRINT
360   LIST 400-420
370   GOSUB 12000
400   Y%=200:REM nieuwe y-coordinaat : begin tekst
410   CC%=22:REM nieuwe kleur (groen)
420   GOSUB 10000
430   GOSUB 11000
450   CURSOR 10,17:PRINT "We wijzigen nu DF%=1 in DF%=0"
455   PRINT :PRINT
456   LIST 480-490
460   GOSUB 12000
480   DF%=0:REM Het letterformaat wordt kleiner
490   GOSUB 10000
495   GOSUB 11000
510   CURSOR 10,17:PRINT "We wijzigen A$=diDAIsoft in A$=DAInamic"
515   PRINT :PRINT
520   LIST 570-580
530   GOSUB 12000
570   A$="DAInamic":REM Verandering van de geschreven tekst
580   GOSUB 10000
585   GOSUB 11000
590   CURSOR 5,17:PRINT "We veranderen DF%,X%,Y%,CC% in "
600   CURSOR 15,15:PRINT "DF%=2,X%=50,Y%=70 en CC%=23"
610   CURSOR 5,13:PRINT "en bestuderen het effect van SP%"
620   CURSOR 5,11:PRINT "De waarde van SP% is in instr.160 op 6 gebracht."
630   CURSOR 5,9:PRINT "We voeren bijgevolg volgend programmadeel uit :"
640   PRINT :PRINT
650   LIST 700-720
660   GOSUB 12000
700   DF%=2:X%=50:Y%=70:CC%=23
710   SP%=6
720   GOSUB 10000
730   GOSUB 11000
750   CURSOR 5,17:PRINT "We veranderen de waarde van SP%=6 in SP%=11"
755   PRINT :PRINT
756   LIST 800-810
760   GOSUB 12000
800   SP%=11:REM Wijziging van de ruimte tussen de karakters
810   GOSUB 10000
815   GOSUB 11000
820   CURSOR 5,17:PRINT "We laten nu SP% veranderen van 8 tot 0"
830   PRINT :PRINT
840   LIST 900-950
850   GOSUB 12000
900   FOR SP%=8 TO 0 STEP -1
910   CC%=21:GOSUB 10000
920   WAIT TIME 2
930   CC%=20:GOSUB 10000
950   NEXT
960   GOSUB 11000
970   CURSOR 5,19:PRINT "De parameters X%,Y%,DF%,CC% en SP% zijn"
980   CURSOR 5,18:PRINT "duidelijk de belangrijkste in de FGT-routine."
990   CURSOR 5,17:PRINT "De overige worden verder besproken."
1000  CURSOR 5,15:PRINT "DE PARAMETER : VF% (vertical flag)"
1010  CURSOR 5,13:PRINT "We initieren SP% terug op 5 en CC% op 22."
1020  CURSOR 5,12:PRINT "De waarde van VF% werd nog niet gebruikt."
1025  CURSOR 5,11:PRINT "In alle voorgaande programmadelen was die waarde"
1026  CURSOR 5,10:PRINT "bijgevolg 0. We wijzigen die nu in de alternatieve"
1027  CURSOR 5,9:PRINT "waarde 1."
```

```
1028  PRINT :PRINT
1030  LIST 1050-1090
1040  GOSUB 12000
1050  SP%=5:CC%=22
1060  VF%=0:REM default-waarde voor de parameter VF%
1070  GOSUB 10000:WAIT TIME 10
1080  VF%=1:REM schrijfrichting wordt over 90 gr gedraaid
1090  GOSUB 10000
1092  GOSUB 11000
1094  CURSOR 5,17:PRINT "DE PARAMETER ZF% (zinflag)"
1096  CURSOR 5,15:PRINT "We wijzigen de DEFAULT-waarde 0 in 1."
1100  CURSOR 5,14:PRINT "We zetten VF% opnieuw op zijn DEFAULT-waarde 0."
1110  CURSOR 5,13:PRINT "We wijzigen eveneens X%=40 in X%=160."
1120  CURSOR 5,12:PRINT "Je merkt wel waarom we dit doen."
1130  PRINT :PRINT
1140  LIST 1200-1220
1150  GOSUB 12000
1200  VF%=0:X%=160
1205  ZF%=0:REM default-waarde
1210  GOSUB 10000:WAIT TIME 10
1215  ZF%=1:REM schrijfrichting wordt over 180 gr gedraaid
1220  GOSUB 10000
1225  GOSUB 11000
1230  CURSOR 5,17:PRINT "DE PARAMETERS FC%, FF% EN ID%"
1240  CURSOR 5,15:PRINT "In FC% staat 0,1,2 of 3 en wijst hiermee naar de "
1250  CURSOR 5,14:PRINT "kleuren uit het laatste COLORG-statement."
1260  CURSOR 5,13:PRINT "FF% bevat 0 of 1 : indien FF%=1 wordt,voor-"
1270  CURSOR 5,12:PRINT "dat A$ geschreven wordt, de achtergrond van deze"
1280  CURSOR 5,11:PRINT "tekst KARAKTER PER KARAKTER in de kleur gezet,"
1282  CURSOR 5,10:PRINT "waarnaar FC% verwijst. We wijzigen SP% van 5 in 7."
1283  CURSOR 5,9:PRINT "WAAROM ?"
1284  PRINT :PRINT
1285  LIST 1300-1310
1290  GOSUB 12000
1300  SP%=7:ZF%=0:X%=40:FC%=1:FF%=1:ID%=10
1310  GOSUB 10000
1340  GOSUB 11000
1350  CURSOR 5,17:PRINT "COLORG-statement is COLORG 8 3 5 0"
1360  CURSOR 5,16:PRINT "Met FC%=1 wordt de achtergrond rood (3)"
1370  CURSOR 5,15:PRINT "Met FC%=2 wordt de achtergrond groen (5)"
1380  CURSOR 5,14:PRINT "Kijk maar..."
1390  PRINT :PRINT
1395  LIST 1400
1396  GOSUB 12000
1400  FC%=2:GOSUB 10000
1410  GOSUB 11000
1420  CURSOR 5,17:PRINT "Inderdaad, omdat ook CC% nog naar groen"
1430  CURSOR 5,16:PRINT "verwees, was de tekst in vorig programmadeel"
1440  CURSOR 5,15:PRINT "onleesbaar."
1460  CURSOR 5,14:PRINT "We wijzigen tot slot de parameter ID% van 10 in 5."
1470  PRINT :PRINT
1480  LIST 1500-1505
1490  GOSUB 12000
1500  ID%=5:REM ID% factor voor hoogte gekleurde achtergrond
1505  GOSUB 10000
1510  GOSUB 11000
1520  CURSOR 5,20:PRINT "DE LAATSTE FGT-PARAMETER IS PF% (positionflag)"
1530  CURSOR 5,19:PRINT "PF% neemt alleen 0 en 1 als bruikbare waarden aan"
1540  CURSOR 5,18:PRINT "PF% bepaalt de plaats waar een tweede tekst"
1560  CURSOR 5,17:PRINT "geschreven wordt.We illustreren als volgt:"
1570  CURSOR 5,16:PRINT "We schrijven eerst de tekst A$=diDAIsoft en daarna "
1580  CURSOR 5,15:PRINT "de tekst A$=DAInamic op scherm."
1590  CURSOR 5,14:PRINT "De eerste maal staat PF% op 1,de tweede maal op 0."
1595  LIST 1600-1710:GOSUB 12000
```

```
1600  ID%=0:FC%=0:FF%=0:REM default-waarden
1610  X%=40:Y%=180:DF%=0:CC%=23:SP%=6
1620  PF%=1
1630  A$="DAInamic":GOSUB 10000
1640  WAIT TIME 50
1650  A$="diDAIsoft":GOSUB 10000
1660  WAIT TIME 50
1670  FILL X%,Y% XMAX,YMAX 20:REM BLANCO SCH ERM
1680  PF%=0:REM WIJZIGING PF%-PARAMETER
1685  CC%=21:DF%=1:REM WIJZIGING KLEUR EN FORMAAT
1690  A$="DAInamic":GOSUB 10000
1700  WAIT TIME 50
1710  A$="diDAIsoft":GOSUB 10000
1720  GOSUB 11000
1730  CURSOR 5,17:PRINT "Merk op dat met PF%=1 de twee teksten naast el-"
1740  CURSOR 5,16:PRINT "kaar worden geschreven,d.w.z. het einde vam de"
1750  CURSOR 5,15:PRINT "vorige tekst wordt als startcoordinaat van de nieuwe"
1760  CURSOR 5,14:PRINT "tekst genomen. Merk eveneens op dat indien deze vlag"
1770  CURSOR 5,13:PRINT "gezet is, zelfs het expliciet opgeven van een X% en een
"
1780  CURSOR 5,12:PRINT "Y% (instr. 1610) geen invloed heeft op de plaats waar"
1790  CURSOR 5,11:PRINT "de tekst geschreven wordt: dit verklaart waarom "
1800  CURSOR 5,10:PRINT "de tekst DAInamicdiDAIsoft (in formaat DF%=0) geheel"
1810  CURSOR 5,9:PRINT "rechts op het scherm staat. Deze tekst sluit aan op "
1820  CURSOR 5,8:PRINT "het einde van de tekst uit het vorige programmadeel"
1830  CURSOR 5,7:PRINT "De tekst in formaat DF%=1 en onder de vlag"
1840  CURSOR 5,6:PRINT "PF%=0 staat uiteraard wel op de nieuwe X% en"
1850  CURSOR 5,5:PRINT "Y% positie. Omdat tussen het wegschrijven van"
1860  CURSOR 5,4:PRINT "A$=DAInamic en A$=diDAIsoft de parameters X%"
1870  CURSOR 5,3:PRINT "en Y% niet veranderd worden, is het logisch"
1880  CURSOR 5,2:PRINT "dat beide teksten over elkaar worden geschreven."
1890  G%=GETC:IF G%<>32 THEN 1890
1900  PRINT CHR$(12)
1910  CURSOR 5,17:PRINT "De teksten die tot nu toe op scherm wer-"
1920  CURSOR 5,16:PRINT "den geschreven werden aan A$ toegekend"
1930  CURSOR 5,15:PRINT "vanop het toetsenbord. De ingelezen FGT-"
1940  CURSOR 5,14:PRINT "tabel bevat ook een aantal ontworpen kar-"
1950  CURSOR 5,13:PRINT "rakters. Deze roep je op met de ASCII-"
1960  CURSOR 5,12:PRINT "code die tijdens het ontwerpen aan elk"
1970  CURSOR 5,11:PRINT "karakter werd toegekend"
1980  PRINT :PRINT
1990  LIST 2000-2030
1995  GOSUB 12000
2000  X%=70:Y%=200:DF%=1:SP%=8:CC%=21:REM rood
2010  A$=CHR$(13)+CHR$(0)+","+CHR$(1)+","+CHR$(2)+CHR$(14)
2020  REM op 13 en 14 staan de accolades
2025  REM op 0,1 en 2 staan Griekse letters
2030  GOSUB 10000
2040  GOSUB 11000
2050  CURSOR 5,17:PRINT "Volgend programmadeel demonstreert de karak-"
2060  CURSOR 5,16:PRINT "ters die in de ingelezen tabel beschikbaar zijn."
2070  CURSOR 5,15:PRINT "Meerdere karakters zijn beschikbaar in het program-"
2080  CURSOR 5,14:PRINT "ma grafische hulp. De getallen zijn de oproepcodes"
2090  CURSOR 5,13:PRINT "in de instructie CHR$(code)."
2100  GOSUB 12000
2110  DF%=1:CC%=21:X%=30:FOR J%=0 TO 2:Y%=230
2120  FOR I%=0 TO 9:B$=STR$(J%*10.0+I%):C$=MID$(B$,1,LEN(B$)-3)
2130  A$=C$+"="+CHR$(J%*10+I%):GOSUB 10000:Y%=Y%-25:NEXT:X%=X%+75:NEXT
2140  Y%=230:FOR I%=30 TO 31:A$=MID$(STR$(I%),1,LEN(STR$(I%))-3.0)+"="+CHR$(I%):
GOSUB 10000:Y%=Y%-25:NEXT
2145  WAIT TIME 100
2150  GOSUB 11000
```

```
2160  CURSOR 5,20:PRINT "SAMENSTELLEN VAN EEN FGT-PAKKET"
2170  CURSOR 5,18:PRINT "1. Begin met de BOOTSTRAP LOADER"
2180  CURSOR 5,16:PRINT "2. Plaats hierachter het FGT-deel: dit doe je met:"
2190  CURSOR 5,15:PRINT "   UT (RETURNTOETS) W29B BFF FGT 29B BFF WIE "
2200  CURSOR 5,14:PRINT "   indien je de FGT-tabel uit dit programma gebruikt."
2210  CURSOR 5,13:PRINT "   Voor een andere FGT-tabel moet de bovengrens BFF "
2220  CURSOR 5,12:PRINT "   aangepast worden."
2225  CURSOR 5,10:PRINT "3. Zet hierachter je BASIC-programma "
2230  CURSOR 10,8:PRINT "    SUCCES MET JE FGT-EXPERIMENTEN"
2240  CURSOR 30,4:PRINT "  Bruno van Rompaey"
9000  END
10000 REM ***SUBROUTINE FGT
10010 POKE #2F2,X% MOD 256:POKE #2F3,X%/256:POKE #2F4,Y%
10020 POKE #2F5,SP%:POKE #2F6,ID%
10030 C%=FC%*#40+CC%:F%=FF%*#80+PF%*#40+ZF%*#20+VF%*#10+DF%:POKE #2F0,C%:POKE #2
F1,F%
10040 CALLM #300,A$
10050 RETURN
11000 WAIT TIME 30:FILL 0,0 XMAX,YMAX 20:MODE 0
11010 POKE #BF69,#6A:CURSOR 3,22:PRINT "FGT - TOELICHTINGEN - diDAIsoft"
11020 CURSOR 4,21:PRINT "LEES ONDERSTAANDE TEKST, DRUK SPACE, KIJK EN WACHT":RET
URN
12000 G%=GETC:G%=GETC:G%=GETC
12010 G%=GETC:IF G%<>32 THEN 12010
12015 PRINT CHR$(12)
12020 MODE 6:RETURN
```

MICRO · DICO

DCE Data Comunication Equipment (equipement pour la transmission de données
TIC Timer Interupt Controler (controleur de temporisation des interruptions)
RAM Random Access Memory (Memoire vive)
ROM Read Only Memory (Memoire morte)
PROM Programable Read Only Memory(Memoire morte programmable)
EPROM Erasable & Programable Read Only Memory (Memoire morte programmable et effacable)
µP Micro-processor (Microprocesseur)
DOS Disk Operating System (Controleur pour disquettes)
TOS Tape Operating System (Controleur pour cassettes)
I/O Input output (Entrées Sorties)
BIT Binary unit-digit (Unitée Binaire)
PSW Program Status Word (Mot d'état)
LSB Less Significant Bit (Bit le moins signifiant)
LSD ' ' Digit (chiffre ... ')
MSB Most ' ' Bit (Bit le plus significatif)
CP/M Control Program for µP (Programme de gestion pour microprocesseur)
LSI Large Scale Integration (Integration à Haute echelle)
MDS Micro Device System

```
100 REM Benchmark 1
110 PRINT "S"
120 FOR K=1 TO 1000
130 NEXT K
140 PRINT "E"
150 END
```

```
100 REM Benchmark 2
110 PRINT "S"
120 K=0
130 K=K+1
140 IF K<1000 THEN 130
150 PRINT "E"
160 END
```

```
100 REM Benchmark 3
110 PRINT "S"
120 K=0
130 K=K+1
140 A=K/K*K+K-K
150 IF K<1000 THEN 130
160 PRINT "E"
170 END
```

```
100 REM Benchmark 4
110 PRINT "S"
120 K=0
130 K=K+1
140 A=K/2*3+4-5
150 IF K<1000 THEN 130
160 PRINT "E"
170 END
```

```
100 REM Benchmark 5
110 PRINT "S"
120 K=0
130 K=K+1
140 A=K/2*3+4-5
150 GOSUB 190
160 IF K<1000 THEN 130
170 PRINT "E"
180 END
190 RETURN
```

```
100 REM Benchmark 6
110 PRINT "S"
120 K=0
130 DIM M(5)
140 K=K+1
150 A=K/2*3+4-5
160 GOSUB 220
170 FOR L=1 TO 5
180 NEXT L
190 IF K<1000 THEN 140
200 PRINT "E"
210 END
220 RETURN
```

```
100 REM Benchmark 7
110 PRINT "S"
120 K=0
130 DIM M(5)
140 K=K+1
150 A=K/2*3+4-5
160 GOSUB 230
170 FOR L=1 TO 5
180 M(L)=A
190 NEXT L
200 IF K<1000 THEN 140
210 PRINT "E"
220 END
230 RETURN
```

```
100 REM Benchmark 8
110 PRINT "S"
120 K=0
130 K=K+1
140 A=K↑2
150 B=LOG(K)
160 C=SIN(K)
170 IF K<1000 THEN 130
180 PRINT "E"
190 END
```

pcw

```
IMPINT
10 MODE 6:COLORG 0 10 0 0
20 FOR I=1 TO 44
30 DRAW 50+100x(SIN(I)+1),110 150,10+100x(COS(I)+1) 21
40 NEXT
```

Benchmark timings

MACHINE	BM1	BM2	BM3	BM4	BM5	BM6	BM7	BM8	AVERAGE
Olivetti M20	1.3	4.0	8.1	8.5	9.6	17.4	26.7	1.6	11.5
DAI	0.9	4.8	10.1	9.8	11.2	18.1	30.1	2.1	13.3
BBC Micro	1.0	3.1	8.2	8.7	9.1	13.9	21.4	5.1	14.6
Monroe 8820	2.1	4.2	9.9	10.5	11.0	20.1	32.0	3.3	15.4
Altos ACS 800-2	1.4	4.3	11.3	11.3	12.0	21.2	34.9	2.7	15.4
Vector Graphic VIP	1.0	3.8	10.9	10.7	11.6	20.5	32.7	3.4	15.7
ACT 800	0.9	4.6	8.5	9.4	10.1	14.9	23.4	5.6	16.0
Sharp MZ80B	0.6	4.5	8.5	11.5	13.0	19.0	27.5	5.0	16.8
Micromation I Plus	1.4	4.4	11.2	11.3	11.5	21.2	34.9	3.9	16.9
Mini 801 (44k CP/M)	1.2	3.7	9.9	9.8	10.5	18.6	29.6	5.4	17.2
IBM Personal Computer	1.5	5.2	12.1	12.6	13.6	23.5	37.4	3.5	17.6
* Exleigh Expert (compiled)	2.5	2.5	8.0	8.0	8.0	21.0	25.0	7.0	18.1
Osborne 01	1.4	4.4	11.7	11.6	12.3	21.9	34.9	6.1	19.9
Tandy TRS-80 Model II	1.0	5.0	13.0	13.0	14.0	23.0	35.0	6.0	20.5
Hewlett Packard HP125	1.7	5.0	12.5	12.5	14.0	26.0	40.0	6.0	21.5
Mini 801 (62k CP/M)	1.7	4.7	12.4	12.2	13.1	24.3	38.6	6.6	21.6
Intertec Superbrain	1.6	5.2	14.0	13.9	14.8	26.3	43.2	5.6	21.9
Positron 9000	1.1	2.1	5.4	6.8	7.2	14.9	20.2	12.0	22.2
DDE SPC/1	4.8	6.2	14.7	13.9	14.7	41.1	58.1	2.6	22.4
ABC 24	1.2	4.0	16.0	15.0	16.0	25.0	38.0	8.0	24.4
Apple III	1.7	7.2	13.5	14.5	16.0	27.0	42.5	7.5	24.7
ACT Sirius 1	2.0	7.4	17.0	17.5	19.8	35.4	55.9	4.3	24.8
Oki if8000	2.2	6.4	16.8	16.8	17.9	31.8	50.7	5.7	25.0
Ohio Scientific Challenger C2 4P	1.4	7.8	15.0	16.5	17.8	27.0	39.5	7.5	25.0
Xerox 820	1.7	5.5	15.5	15.1	16.2	28.9	46.1	8.0	26.1
NEC PC 8001	1.7	8.3	18.1	17.8	18.6	29.5	49.2	7.0	26.7
Newbrain	2.0	5.8	19.2	17.5	19.2	32.0	48.8	7.0	26.8
ABC 80	1.1	2.3	11.1	12.1	12.6	17.7	23.9	13.6	27.1
Philips P2000	1.9	5.9	15.8	15.7	16.7	29.8	47.2	8.5	27.3
Commodore VIC 20	1.4	8.3	15.5	17.1	18.3	27.2	42.7	9.9	28.7
* Exleigh Expert (interpreted)	2.5	7.2	18.5	18.5	19.3	35.0	52.0	8.5	29.8
Apple II	1.3	8.5	16.0	17.8	19.1	28.6	44.8	10.7	30.4
Hewlett Packard HP85	1.8	3.8	16.3	16.5	17.7	30.0	44.8	12.7	32.2
Pasca 640	2.0	7.0	19.0	18.0	20.0	36.0	57.0	10.0	32.4
Sharp MZ80K	1.4	9.4	16.3	22.5	25.4	36.8	51.1	10.2	33.1
Exidy Sorcerer	1.8	10.0	20.7	22.2	24.3	37.6	53.7	9.6	33.3
Sharp MZ80A	1.5	9.2	16.4	22.8	25.6	37.7	55.0	10.1	33.7
Commodore CBM 8032	1.7	10.0	18.4	20.3	21.9	32.4	51.0	11.9	34.3
Transam Tuscan	2.3	13.0	26.0	27.0	32.0	48.0	68.0	6.0	34.5
Commodore PET 2001	1.7	9.9	18.4	20.4	21.0	32.5	50.9	12.3	34.7
Compucolor II	2.0	10.9	22.4	23.9	25.7	38.7	55.2	10.2	35.1
Dragon 32	1.6	10.2	19.7	21.6	23.3	34.3	50.0	12.9	36.2
Hewlett Packard HP86	3.0	5.2	19.4	18.8	20.4	36.5	56.5	13.4	36.7
Hitachi Peach	2.0	11.0	26.0	26.0	27.0	46.0	78.0	10.0	39.5
Tandy TRS-80 Color Computer	2.0	11.3	22.2	23.9	27.0	41.5	61.1	13.0	39.9
Panasonic JD700	2.8	9.1	24.6	24.7	26.2	43.9	69.7	11.8	39.9
SBS 8000	1.8	9.4	29.0	29.0	31.6	44.0	82.5	11.2	42.4
Heath H89 (Mbasic)	2.5	9.2	25.8	26.0	27.0	46.6	73.2	13.0	42.5
Tandy TRS-80 Model I Level II	2.7	11.6	28.0	28.5	31.3	51.9	81.0	11.7	44.0
Video Genie	2.7	11.6	28.0	28.5	31.3	51.9	81.0	11.7	44.0
Cromemco System Three	1.7	4.6	14.9	17.8	19.4	30.2	41.9	22.9	44.9
Ohio Scientific Challenger C3 S1	1.7	13.1	21.6	23.7	29.2	39.6	58.3	17.6	45.4
Sinclair ZX81 (fast mode)	4.5	6.9	16.4	15.8	18.6	49.7	68.5	22.9	51.2
Sinclair Spectrum	4.8	8.7	21.1	20.4	24.0	55.3	80.7	25.3	58.5
Sharp PC3201	4.0	13.5	35.5	35.5	38.5	67.0	108.0	25.0	69.0
Casio fx9000	2.5	9.0	24.0	24.0	26.0	42.0	60.0	36.5	69.1
Atari 400/800	2.3	7.4	19.9	23.2	26.8	40.7	61.5	43.1	76.6
Texas TI 99/4	2.9	8.8	22.8	24.5	26.1	61.6	84.4	38.2	76.6
Texas TI99/4A (standard)	3.0	9.0	24.0	24.8	26.2	61.9	84.6	38.4	77.2
Texas TI99/4A (extended)	6.5	18.5	40.0	40.1	42.0	98.4	140.3	24.0	78.2
Periflex 630/48	4.5	10.5	27.5	28.5	31.5	59.0	79.5	60.0	105.1
BASF 7120	2.4	7.0	35.0	36.5	39.0	50.0	63.0	114.0	171.6

*SORD M23

program identification

title	:	MULTIPUZZLE
author	:	Christian POELS
purpose	:	a game of calculation & strategie
comment	:	

```
+0----+----1----+----2----+----3----+----4----+----5----+----+
!                                                           !
!                    == MULTIPUZZLE ==                      !
!                    ================                       !
2                                                           2
!Christian POELS - 27/5/1981                                !
!                                                           !
!          * 5 *              ENTREES PRECEDENTES :         !
!            5 4                                            !
+      ----------          Colonne 1 : 3 4                  +
!        * * * 4           Colonne 2 : 5                    !
!      * * * 5             Colonne 3 : 0                    !
!      ----------          Colonne 4 : 7                    !
!      * * * 5 4           Colonne 5 : 1 2 6                !
1                                                           1
!Nombre de coups joues : 8                                  !
!Nombre de coups manques : 6                                !
!Nombre de chiffres trouves : 7                             !
!                                                           !
+ESSAI No. 9                                                +
!                                                           !
!Quel est votre chiffre?6                                   !
!Quelle est la colonne?5                                    !
!                                                           !
0                                                           0
+0----+----1----+----2----+----3----+----4----+----5----+----+
```

```
  1     REM .........Copyright Christian POELS
  2     REM ..........10, rue des Bas-Sarts
  3     REM ..........B-4100 Seraing - Belgique
  4     REM ..........Tel.: (041) 37.16.06
  5     POKE #FF05,10:CLEAR 1000
  7     MODE 0:COLORT 12 1 0 0
  10    REM
  20    GOSUB 100
  30    POKE #FF05,255
  40    GOSUB 200
  50    GOSUB 300
  60    PRINT :PRINT :PRINT "Voulez-vous jouer une autre partie (O/N)?";
  70    RE=GETC:IF RE=79.0 THEN 30
  71    IF RE<>78.0 THEN 70
  80    POKE #FF05,255
  90    END
  100   GOSUB 11000
```

```
106   PRINT "Voulez-vous les instructions (O/N)?";
107   RE=GETC:IF RE=79.0 THEN GOTO 10000
108   IF RE<>78.0 THEN 107
109   PRINT CHR$(12)
110   POKE #FF05,255:DIM R$(8.0),F(9.0,5.0),A(5.0,5.0),P$(5.0,5.0),N(5.0)
120   FOR I%=1 TO 8:READ R$(I%):NEXT
130   DATA score excellent,tres bon score,bon score,score au dessus de la m
oyenne
140   DATA score moyen,score en dessous de la moyenne,score passable,mauvai
s score
150   REM
199   RETURN
200   REM
210   FOR I%=1 TO 5:FOR J%=1 TO 5:A(I%,J%)=-1.0:NEXT J%:NEXT I%
220   FOR I%=0 TO 9:FOR J%=1.0 TO 5.0:F(I%,J%)=0.0:NEXT J%:NEXT I%
230   A(1.0,1.0)=INT(RND(10.0)):A(2.0,1.0)=INT(RND(10.0)):A(3.0,1.0)=INT(RN
D(9.0)+1.0)
240   A(1.0,2.0)=INT(RND(10.0)):A(2.0,2.0)=INT(RND(9.0)+1.0)
250   N(1.0)=100.0*A(3.0,1.0)+10.0*A(2.0,1.0)+A(1.0,1.0):N(2.0)=10.0*A(2.0,
2.0)+A(1.0,2.0)
260   N(3.0)=A(1.0,2.0)*N(1.0):N(4.0)=A(2.0,2.0)*N(1.0)*10.0:N(5.0)=N(1.0)*
N(2.0)
270   M=10.0:FOR I%=1 TO 5
280   FOR J%=3 TO 5:N1%=INT(N(J%)/M+1E-2):A(I%,J%)=INT(N(J%)-N1%*M+1E-2):N(
J%)=N1%:NEXT J%
290   NEXT I%:A(1.0,4.0)=-1.0:A(5.0,3.0)=-1.0:N1%=0.0:N2%=0:N5%=0.0
292   FOR I%=1 TO 5:FOR J%=1 TO 5:IF A(I%,J%)=(-1.0) THEN P$(I%,J%)="  "
293   IF A(I%,J%)<>(-1.0) THEN P$(I%,J%)=" *"
294   NEXT J%:NEXT I%
299   RETURN
300   GOSUB 11000
305   POKE #BCCA,#D0:POKE #BFEE,#CB
306   POKE #BD50,#CC:POKE #B920,#CB:POKE #B89A,#D2:POKE #B708,#CC:POKE #B68
2,#D1:POKE #B576,#D3
310   GOSUB 1000
320   IF N5%=18.0 THEN 340
330   GOSUB 2000:GOTO 310
340   G=G+1.0:T=T+N1%:V=T/G
350   FOR I=1.0 TO 8.0:CURSOR 0,I:PRINT "
                              ":NEXT:CURSOR 0,9:G%=G
355   POKE #FF05,4:PRINT "Nombre moyen de coups rates apres";G%;" partie";:
IF G>1.0 THEN PRINT "s";
360   V%=V:PRINT " :";V%;:PRINT ".":Q=INT(V/2.0):IF Q>=1.0 THEN 367
365   Q=1.0:IF Q>8.0 THEN Q=8.0:GOTO 370
367   IF Q>8.0 THEN Q=8.0
370   PRINT "Voila un ";R$(Q);"."
899   RETURN
1000  REM
1010  CURSOR 0,17
1020  FOR I%=1 TO 5
1025  CURSOR 5,CURY
1030  FOR J%=5 TO 1 STEP -1:PRINT P$(J%,I%);:NEXT J%
1040  PRINT :IF I%<>2.0 AND I%<>4.0 THEN 1050:PRINT TAB(5);" ";:FOR I=1.0 T
O 11.0:PRINT CHR$(95);:NEXT:PRINT
1050  NEXT I%
1060  PRINT :PRINT "Nombre de coups joues :";N2%:PRINT "Nombre de coups man
ques :";N1%
1070  PRINT "Nombre de chiffres trouves :";N5%
1999  RETURN
```

```
2000  N2%=N2%+1.0
2010  CURSOR 0,5:PRINT "ESSAI No.";N2%
2020  PRINT :PRINT "Quel est votre chiffre?";
2021  RE=GETC:IF RE<48.0 OR RE>57.0 THEN 2021
2022  D=RE-48.0:PRINT CHR$(RE)
2023  PRINT "Quelle est la colonne?";
2024  RE=GETC:IF RE<49.0 OR RE>53.0 THEN 2024
2025  C=RE-48.0:PRINT CHR$(RE)
2026  REM
2030  REM
2040  IF F(D,C)=1.0 THEN 2010
2050  F(D,C)=1.0
2060  N9%=N5%
2070  FOR I%=1 TO 5:IF A(C,I%)<>D THEN 2080:P$(C,I%)=" "+MID$(STR$(D),1,1):
N5%=N5%+1.0
2080  NEXT:IF N9%=N5% THEN N1%=N1%+1.0
2090  GOSUB 3000
2999  RETURN
3000  CURSOR 25,17
3010  PRINT "     ENTREES PRECEDENTES :":PRINT
3020  FOR I%=1 TO 5:CURSOR 25,CURY:PRINT "Colonne";I%;" :";:FOR J%=0 TO 9:I
F F(J%,I%)=1.0 THEN PRINT J%;
3030  NEXT:PRINT :NEXT:PRINT
3999  RETURN
9000  INPUT R$:R$=LEFT$(R$,1):IF R$<>"O" AND R$<>"N" THEN PRINT "O OU N";:G
OTO 9000
9010  RETURN
10000 PRINT CHR$(12):POKE #BDD6,#D5:POKE #BAB2,#D2:POKE #B78E,#D0:POKE #B4F
0,#D3:POKE #FF05,3:PRINT "INSTRUCTIONS :"
10005 PRINT "==============":PRINT
10010 PRINT "     Multipuzzle est un jeu d'adresse, de chance et de"
10020 PRINT "deduction dont le but est de retrouver les chiffres d'une"
10030 PRINT "multiplication (d'un nombre de 3 chiffres par un nombre de 2"
10040 PRINT "chiffres), calculee secretement et au hasard par"
10050 PRINT "l'ordinateur.":PRINT
10060 PRINT "     Une fois lance, le programme affiche sur l'ecran le"
10070 PRINT "squelette bien connu d'une multiplication, dont la"
10080 PRINT "particularite est que tous ses chiffres sont remplaces par"
10090 PRINT "des asterisques (*). Vous demandez alors, ";CHR$(34);"Y a-t-il
 tel"
10095 PRINT "chiffre dans telle colonne?";CHR$(34);"."
10100 PRINT :PRINT "     Si le chiffre existe, il vient remplacer le ou les
"
10110 PRINT "asterisques correspondants. Dans le cas contraire, le"
10120 PRINT "programme comptabilise imperturbablement le nombre de coups"
10130 PRINT "rates.":PRINT :PRINT "BONNE CHANCE!":POKE #FF05,255:WAIT TIME
200:GOTO 110
11000 PRINT CHR$(12):POKE #BDD6,#D0:POKE #BCCA,#D3:PRINT TAB(21);"== MULTIP
UZZLE ==":PRINT TAB(21);"================":PRINT
11010 PRINT "Christian POELS - 27/5/1981":PRINT :RETURN
```

FGT-DISK-PEEK-POKE

Programma 1

```
100   REM : FGT IN MATRIX
110   CLEAR 20000
120   DIM A%(#90,#F)
130   FOR I%=0 TO #90
140   FOR J%=0 TO #F
150   S%=#1C00+I%*16+J%
160   A%(I%,J%)=PEEK(S%)
170   NEXT:NEXT
180   N$="FGT-MATH:0"
190   SAVEA A% N$
```

Programma 2

```
1000  REM : FGT INLADEN ALS MATRIX
1010  CLEAR 20000
1020  DIM A%(#90,#F)
1030  N$="FGT-MATH:0"
1040  LOADA A% N$
1050  FOR I%=0 TO #90
1060  FOR J%=0 TO #F
1070  S%=#1C00+I%*16+J%
1080  POKE S%,A%(I%,J%)
1090  NEXT
1100  NEXT
1110  CLEAR 1000:STOP
```

Programma 3

```
10    REM : POINTERS
20    POKE #29B,#0:POKE #29C,#25
30    N$="FGT-LOAD:0"
40    CALLM #300,N$
```

SPEED UP THE BITMANIPULATION PROGRAM FROM DAInamic 12

2050 B=P IAND (2^(X+1)) should become :
2050 B=P IAND((2 SHL X)-1)

- Henk Stokhorst

'n FGT - compatibel met de Diskette-driver

In het DAInamic-nummer van 4 april 1981 - blz.67 tot 77 - vinden we 'n zeer interessant artikel over het Fast Graf Text (FGT) programma. De daar beschreven FGT-routine is echter incompatibel met 'n diskette-driver. Waarom?
Dit programma wordt gepookt op #300 tot +/- #900.
Om niet gestoord te worden door het inladen van 'n Basic-programma gaat men de adressen voor Start of Heap wijzigen, zodat Heap en Basic-programma na de FGT komt . Het FGT-programma wordt dan gestart met 'n CALLM#300,A$.(Wel bekend aan DOS-gebruikers, maar dan voor het inladen van 'n nieuw programma!).
Wanneer we echter over 'n diskette-driver beschikken, wordt de DOS (Diskette Operating System) automatisch ingeladen daar waar normaal de Heap begint, en worden de adressen aangepast.
Vergelijken we nu de begintoestand:

Zonder diskette	Met diskette
*UT	
>D29B 2A6 (R)	
029B EC 02 00 01 EC	029B CC 19 00 01 CC
02A0 03 ED 03 EE 03 50 B3	02A0 1A CD 1A CE 1A 50 B3

Wat we interpreteren als:

029B 02EC	Start of Heap	19CC
029D 0100	Lenght of Heap	0100 (256 bytes)
029F 03EC	Start of Text	1ACC (natuurlijk: tel op!)
02A1 03ED	Start of Symbol	1ACD (geen programma!)
02A3 03EE	End of Symbol	1ACE (geen variabelen!)
02A5 B350	Bottom of Screen	B350

De FGT mag dus niet langer ingeladen worden op #300-#900 , wat dan zou onze DOS vernietigd worden, wat fataal zou zijn.
De FGT moet dus van nieuwe adressen voorzien worden , aangepast aan de plaats van de DOS . Dat werk is nu gebeurd, en er zijn aangepaste FGT-routines te verkrijgen.
Deze worden gepookt vanaf 1C00 tot'n adres afhankelijk van de gebruikte FGT . Als richtsnoer zullen we de Math-FGT gebruiken , die eindigt rond 2430. Het begin, 1C00, is ruim veilig achter de DOS, aangezien de computer zelf de Start of Heap normaal voorziet op 19CC:'n reserve van 'n halve K-byte . Hopelijk zal deze veiligheidsmarge volstaan voor de "nieuwe DOS".
Hoe gaan we nu praktisch te werk?
Bij het aanzetten wordt automatisch de DOS ingeladen.
We plaatsen onze diskette met onze FGT in Driver 0.
Desnoods gaan we met 'n DIR0 zien naar de juiste naam van de FGT.

```
*UT (R)
(>F1C00 3000 0 (R): veiligheidshalve sporen van 'n vroegere programma
                    wissen: facultatief.)
>R FGT:0 (R)      : inlezen van het FGT-machinetaal programma.
>D1C00 2500 (R)   : om het einde van de FGT vast te stellen.
                    desnoods het ladr en hadr in D aanpassen.
                    (begin en eindadres van het display.)
                    Men noteert het einde van de FGT , b.v. 2439, en
                    rond af naar omhoog: b.v. 2500.
>S29B (Space Bar) CC- 00 (SP) 19- 25 (R) : CC- en 19- worden door de
                    computer zelf gemeld.
                    Daarmee is de Start of Heap (op adres 29B en 29C)
                    ingesteld van 19CC naar 2500.
>B                : om terug naar Basic te gaan.
```

Door 'n NEW of 'n CLEAR 4 kan men de andere pointers aanpassen.
Natuurlijk moet uw Basic - programma voorzien zijn van 'n subroutine voor het aanroepen van de FGT . De instrukties daarvoor kunt U overnemen uit het meestal bijgeleverde demo-programma van uw FGT.

Automatisch inladen van FGT, en instellen van de pointers.

Toch blijft dit nog 'n omslachtig werkje , vooral als het dient uitgevoerd gedurende, of zelfs bij de aanvang van 'n les.
Er bestaat echter 'n methode om dit volledig automatisch te laten gebeuren. Daartoe gebruiken we drie kleine programma's,die elders afgedrukt staan.

Programma 1: hoeft maar eens gebruikt, en mag daarna vernietigd worden.
We laden onze FGT in als boven , en passen daarna de pointers aan (want we gaan 'n Basic-programma inladen of intypen).
Dan laden we (of typen) het eerste programma in, en doen 'n RUN.
De FGT wordt nu in 'n matrix (array) geplaatst. De dimensie (#90,#F) is ruim voor onze Math-FGT, en kan voor andere aangepast worden(desnoods) (vergroot), dit om plaatsruimte op de schijf te besparen.(73 files van) (de 580 beschikbare!).Vergeet niet bij wijziging dezelfde wijziging aan te brengen in programma 2. De matrix wordt door het programma ook automatisch weggeladen. Door 'n COPY FGT-MATH.INT:0 FGT-MATH.INT:1 kan de matrix op andere schijven gereproduceerd worden , waar programma's op staan die er gebruik van gaan maken.

Programma 2: dient om de matrix terug in te laden , en weg te poken.
CLEAR 20000 is nodig om de matrix in ontvangst te nemen.
Het programma is kort genoeg om toe te laten de matrix in het programma zelf op te nemen (wat'n bekend nadeel is van matrix-inlezen.)
Maar: de pointers moeten vooraf aangepast worden!
Immers: de matrix wordt weggeladen in de Heap . Door CLEAR 20000 is de Size of Heap 4E20 , en gaat dan van 19CC tot 67EC . Wanneer wij op 1C00 beginnen te poken , vernietigen we zelf onze gegevens , wat blokkering veroorzaakt.(invalid number in line 1080).
'n Aanpassen van de pointers in het begin van het programma zou al even erg zijn. Dus schakelen we programma 3 in.
(Is dat nu eenvoudiger? - Nog even geduld!).

Programma 3: Dit uiterst kort programma doet niets anders dan de Start of Heap bepalen. De uitvoer van het programmaatje dat zich bevindt op 'n slechts klein aantal adressen na 19CC,wordt daardoor niet beinvloed. Met 'n CALLM #300, N$ wordt dan programma 2 ingeladen, op 'n veilige plaats(want de andere pointers worden door de CALLM aangepast). De uitvoer van programma 2 vergt wel even tijd (matrix inladen) , en de computer verwittigt dat hij klaar is met 'n STOPPED IN LINE 1120.De DAI is nu klaar om gelijk welk programma in te laden , met gebruik van onze FGT. De CLEAR 2000 op het einde van programma 2 is nodig om de Heap,die voldoende groot moest zijn om de matrix te kunnen bevatten,terug te reduceren, en zo het adres van Start of Text aan te passen . In onze grafische programma's gebruiken wij vaak MODE 5/6, wat de Bottom of Screen naar voren brengt, zodat botsing met het BASIC-programma(out of memory) niet uitgesloten is.

Eindelijk eenvoudiger: aan ons programma 3 geven we de naam "$USER". Daartoe hebben we door 'n DELETE het DAI-$USER gewist (of liefst bij het vormen van onze schijf niet overgenomen). Bij het inschakelen van de computer wordt programma 3 automatisch ingeladen en daarna door de CALLM #300,N$ de FGT. Tussen inschakelen en bedrijfsklaarheid verloopt minder dan een minuut.(1 op 10 keer is'n Hart-Reset) (vereist: CALLM#300 hapert soms.)

Allereenvoudigst: Voor de les wijzigt men Programma 2 als volgt:

```
1110 CLEAR 1000
1120 N$="Programmatitel": CALLM #300, N$
```

Met 'n SAVE wordt dit gewijzigd programma terug (onder dezelfde naam:) (LOAD-FGT) weggeladen. Bij aanzetten van de computer in de les wordt nu niet alleen de FGT, maar ook uw lesprogramma ingeladen en gestart.

Veel succes!

J.Gesp - Asse.

Een eenvoudige goedkope en veelzijdige EPROM-Programmer

De DAI personal computer heeft standaard vele uitgebreide I/O mogelijkheden. Hiermee zijn vele applicaties te realiseren. Dit artikel behandelt een van de toepassingen.

Wat is een EPROM?

Een Erasable Programmable Read Only Memory is een geheugenelement dat, zoals de naam al zegt, alleen door de computer kan worden uitgelezen. Het kenmerkende van Read Only Memories is dat ze hun inhoud bij spanningsuitval niet verliezen. Ze zijn dus bij uitstek geschikt om in microprocessor-systemen toe te passen die direkt bij inschakelen moeten werken.
Het vervelende van ROM's is dat de inhoud al in de fabriek moet worden aangebracht, dit is alleen aantrekkelijk voor grote aantallen.
Een EPROM biedt de mogelijkheid om herhaald, door de gebruiker, geprogrammeerd te worden. Na programmeren kan de inhoud nl. met Ultra-violet licht gewist worden. Een ideale bouwsteen dus voor gebruik in de ontwikkeling, in kleine series of door de hobbyist.

Het programmeren van een EPROM

De huidige meest gangbare EPROM's zijn opgebouwd als een matrix van 1024x8, 2048x8 of 4096x8 bits. Respectievelijk de 1K, 2K en 4K EPROM. Deze kunnen heel gemakkelijk in een systeem met een woordbreedte van 8 bits worden opgenomen.
In fig. 1 zijn de EPROM's met typenummer en aansluitgegevens opgenomen.
De diverse aansluitingen (fig. 1a):

Ao..An : adreslijnen
Oo..O7 : data- of outputlijnen
$\overline{CE}$: chip enable (active low)
$\overline{OE}$: output enable (active low)
Gnd, V_{cc}, V_{pp} : spanningsaansluitingen

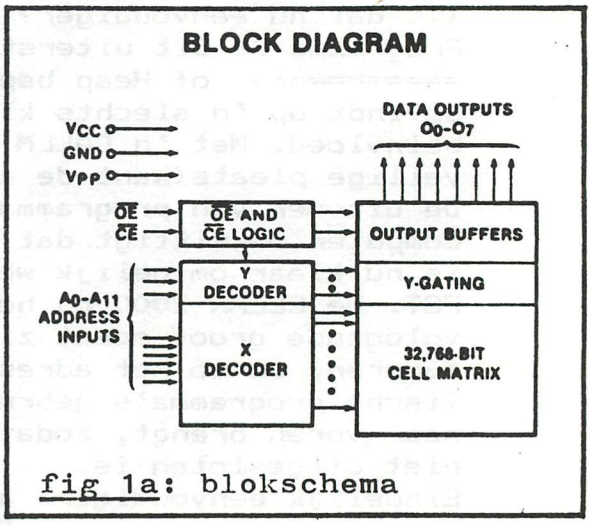


fig 1a: blokschema

Onder normale omstandigheden is de EPROM opgenomen in een µP-systeem. Biedt de microprocessor dan een adres, chipselect en leessignaal aan, dan wordt een geheugenelement geselecteerd en via de uitgangen wordt de waarde naar de µP geschreven. Deze data kan bijvoorbeeld de volgende instructie voor het systeem zijn.
De EPROM moet echter wel geprogrammeerd worden d.w.z. de inhoud moet aangebracht worden.
Uitgangspunt hierbij is een gewist ic, hierin zijn alle bits "1". Door het selectief "0" maken van een aantal bits wordt de EPROM geprogrammeerd. Dit "0" maken gebeurt door op de V_{pp} aansluiting de zogenaamde programmeerspanning van +25V aan te bieden. Vervolgens wordt een adres en datawoord aangeboden, door dan een 50 msec. TTL-puls op de PROG-ingang te zetten wordt de EPROM geprogrammeerd. De data wordt als het ware in de geselecteerde geheugenlocatie "gebrand".
Door deze handeling herhaald uit te voeren kan de gehele EPROM geprogrammeerd worden. Meestal zal men dit doen door iedere cyclus het adres met 1 op te hogen en een nieuw datawoord aan te bieden.

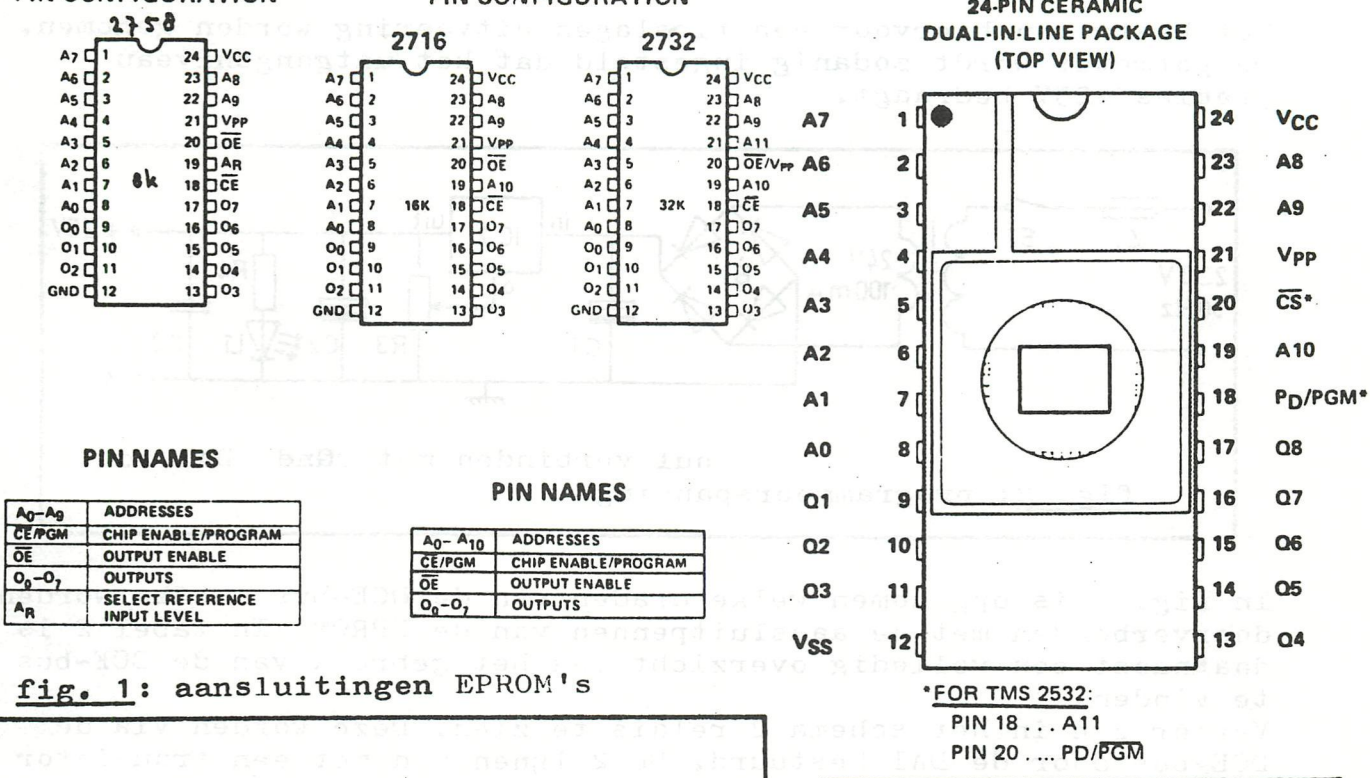


fig. 1: aansluitingen EPROM's

PIN NOMENCLATURE	
A(N)	Address inputs
$\overline{CS}$	Chip Select
PD/PGM, PD/$\overline{PGM}$	Power Down/Program
Q(N)	Input/Output
V_{CC}	+5 V Power Supply
V_{PP}	+25 V Power Supply
V_{SS}	0 V Ground

In tabel 1 zijn de preciese gegevens betreffende het gebruik van de EPROM's opgenomen.
Doordat het programmeren van iedere plaats 50 msec. in beslag neemt, duurt het volledig programmeren van de 2758 1020x0,05=50 sec.
Voor de 2716 en 2516 is dit 100 sec., voor de 2732 en 2532 200 sec.
Na geprogrammeerd te zijn kan de EPROM gewist worden. Hiervoor zijn professionele apparaten, deze zijn echter duur. Het wissen gaat ook goed met een speciale UV-lamp van Philips, TUV 6W-E, deze kost ongeveer 50 gulden. De EPROM's kunnen hier met een elastiekje op bevestigd worden, na 15 à 20 min. zijn ze dan "schoon".

Opbouw Programmer

We hebben nu gezien dat, willen we een EPROM programmeren, we een adres, datawoord plus de nodige controlesignalen aan het ic moeten aanbieden. Deze signalen kunnen geheel door de DAI gegenereerd en via de DCE-bus naar de programmer gestuurd worden. Verder is het nog nodig om de EPROM van spanning te voorzien. De voedingsspanning van +5V is beschikbaar van de DAI, de +25V programmeerspanning moet echter opgewekt worden.

De Hardware

De hardware van de programmer komt neer op de verbinding met de DAI en het maken van de +25V spanning (fig. 2 en fig. 3). De programmeerspanning wordt gemaakt door een 24V wisselspanning gelijk te richten en af te vlakken. Daarna wordt de spanning gestabiliseerd met een 7812 (+12V) spanningsstabilisator die op het juiste uitgangsniveau wordt gebracht door een instelpotmeter in de nulleiding.

Het beste kan hiervoor een tiental uitvoering worden genomen. De potmeter wordt zodanig ingesteld dat het uitgangsniveau precies +25V bedraagt.

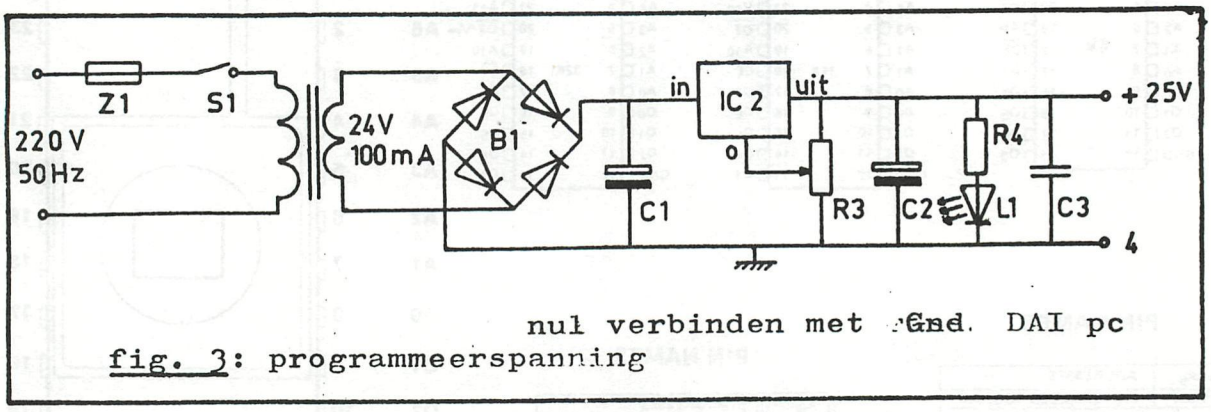


fig. 3: programmeerspanning

In fig. 2 is opgenomen welke draden van de DCE-bus moeten worden doorverbonden met de aansluitpennen van de EPROM. In tabel 2 is daarnaast een volledig overzicht van het gebruik van de DCE-bus te vinden.
Verder zijn in het schema 2 relais te zien. Deze worden via de DCE-bus door de DAI bestuurd. De 2 lijnen zijn met een transistor gebufferd. In het ontwerp is gekozen voor 12V relais die vanuit de DAI gevoed worden. Dit kan zonder problemen omdat de DAI een 12V 1A voeding heeft en hier slechts ongeveer een 0.5A van gebruikt. Het is echter ook mogelijk om 24V relais te gebruiken die dan door de programmeerspanning gevoed worden. Het is dan wel nodig een zwaardere trafo te nemen die in de grotere stroombehoefte kan voorzien.
De diodes die over de relais zijn geschakeld dienen om de inductiespanning, die bij uitschakelen ontstaat, op te vangen. Deze spanning kan nl. de transistors beschadigen.
Het ene relais verzorgt de keuze tussen de 2732 en de andere EPROM's. Dit omdat de 2732 iets andere aansluitingen heeft. In totaal moeten 3 aansluitingen verwisseld worden. Bij de meeste programmers gebeurt dit met een handbediende schakelaar waarbij vergissen nogal eens voorkomt.
Het tweede relais zorgt ervoor dat de programmeerspanning wordt aangesloten bij het programmeren van de ic's.
Het geheel kan in een kastje gemonteerd worden. Als voetje voor de EPROM's is een gewoon ic-voetje bruikbaar mits men over wat handigheid beschikt bij het aanbrengen en eruit halen van ic's. Het is daarom aan te bevelen een zero insertion force socket te gebruiken.

Poort 0	bit 0-7:	datalijnen (Oo t/m O7)
Poort 1	bit 0-7:	adreslijnen (Ao t/m A7)
Poort 2	bit 0-2:	adreslijnen (A8 t/m A10)
	bit 3 :	$\overline{CE}$/PROG (2716,2516,2758)
	:	A11 (2732,2532)
	bit 4 :	$\overline{OE}$ (2716,2516,2758)
	:	$\overline{CE}$/$\overline{PROG}$ (2732)
	:	$\overline{OE}$/$\overline{PROG}$ (2532)
	bit 5 :	"1" (2716,2516,2758,2532)
	:	$\overline{OE}$/V_{pp} (2732)
	bit 6 :	programmeerrelais
	bit 7 :	keuzerelais

TABEL 2: gebruik van de DCE-bus

MODE SELECTION 2716/2758

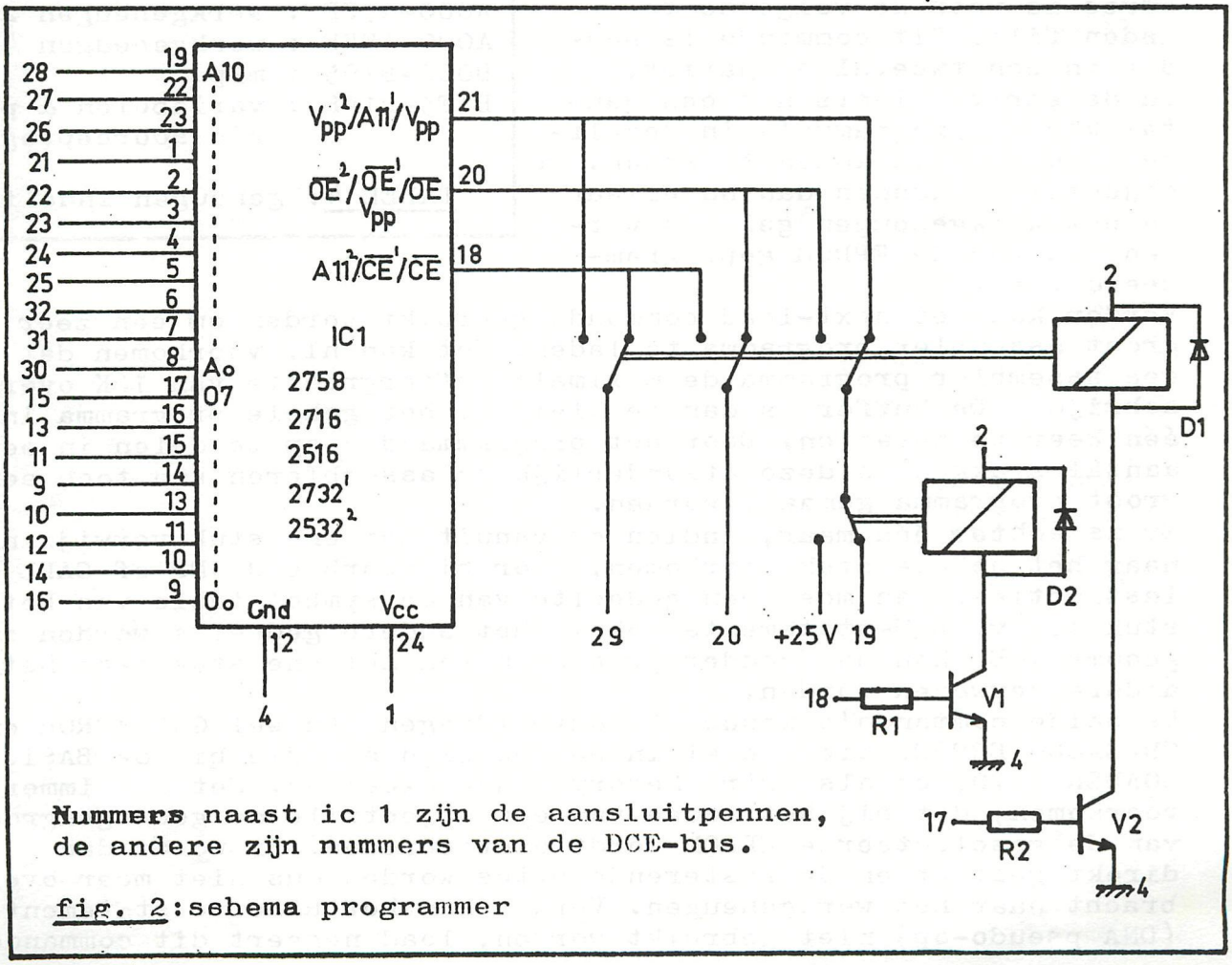
MODE \ PINS	$\overline{CE}$/PGM (18)	A_R (19)	$\overline{OE}$ (20)	V_{PP} (21)	V_{CC} (24)	OUTPUTS (9-11, 13-17)
Read	V_{IL}	V_{IL}	V_{IL}	+5	+5	D_{OUT}
Standby	V_{IH}	V_{IL}	Don't Care	+5	+5	High Z
Program	Pulsed V_{IL} to V_{IH}	V_{IL}	V_{IH}	+25	+5	D_{IN}
Program Verify	V_{IL}	V_{IL}	V_{IL}	+25	+5	D_{OUT}
Program Inhibit	V_{IL}	V_{IL}	V_{IH}	+25	+5	High Z

MODE SELECTION 2732

MODE \ PINS	$\overline{CE}$ (18)	$\overline{OE}/V_{PP}$ (20)	V_{CC} (24)	OUTPUTS (9-11,13-17)
Read	V_{IL}	V_{IL}	+5	D_{OUT}
Standby	V_{IH}	Don't Care	+5	High Z
Program	V_{IL}	V_{PP}	+5	D_{IN}
Program Verify	V_{IL}	V_{IL}	+5	D_{OUT}
Program Inhibit	V_{IH}	V_{PP}	+5	High Z

DEVICE		MODE										
FUNCTION (PINS)		Read		Output Disable		Power Down		Start Programming		Inhibit Programming		Program Verification
TMS 2516	TMS 2532	TMS 2516	TMS 2532	TMS 2516	TMS 2532	TMS 2516	TMS 2532	TMS 2516	TMS 2532	TMS 2516	TMS 2532	TMS 2516
PD/PGM (18)	PD/$\overline{PGM}$ (20)	V_{IL}	V_{IL}	Don't Care	V_{IH}	V_{IH}	V_{IH}	Pulsed V_{IL} to V_{IH}	Pulsed V_{IH} to V_{IL}	V_{IL}	V_{IH}	V_{IL}
$\overline{CS}$ (20)	Use PD/$\overline{PGM}$ as chip select	V_{IL}	N/A	V_{IH}	N/A	Don't Care	N/A	V_{IH}	N/A	V_{IH}	N/A	V_{IL}
V_{PP} (21)	V_{PP} (21)	+5	+5	+5	+5	+5	+ 5	+25	+25	+25	+25	+25 (or + 5)
V_{CC} (24)	V_{CC} (24)	+5	+5	+5	+5	+5	+5	+5	+5	+5	+5	+5
Q (9 to 11, 13 to 17)	Q (9 to 11, 13 to 17)	Q	Q	HI-Z	HI-Z	HI-Z	HI-Z	D	D	HI-Z	HI-Z	Q

TABEL 1: gegevens over gebruik EPROM's Vil="0" Vih="1"



Nummers naast ic 1 zijn de aansluitpennen, de andere zijn nummers van de DCE-bus.

fig. 2: schema programmer

De Software

Het programma zorgt ervoor dat de 8255 (DCE-bus) in de juiste mode staat en dat De juiste signalen verzonden worden.
De software bestaat uit een BASIC en een machinetaal gedeelte. Het machinetaal gedeelte is ondergebracht in de Symbol Table en wordt bij de eerste RUN door het BASIC-programma naar de work-area geladen.
Het programma voorziet in 8 eentoets-commando's, voor een overzicht zie tabel 3.

C	: Check
L	: Load
N	: Next load
P	: Program
R	: Read
T	: Type
U	: Utility
V	: Verify

TABEL 3: commando's

Check: controleert of de EPROM gewist is. De computer leest alle geheugenlocaties uit en verifieert of deze FF_H bevatten. D.w.z. of alle bits "1" zijn. Al naar gelang komt de melding "OK" of "BAD".

Load/Next load: deze 2 commando's zijn speciaal bedoeld voor gebruik met DNA. (DAInamic Assembler). Het load commando laadt een objectfile van adres 3000_H naar het werkgeheugen. Het werkgeheugen is een geheugenblok van 4K in de RAM van de DAI waar de data, die in de EPROM moet komen, wordt opgeslagen, tabel 4.
Het load commando plaatst de objectfile steeds vanaf het beginadres $A000_H$ in het werkgeheugen.
Met het next-load commando is het mogelijk verschillende files achter elkaar in het werkgeheugen te laden. In een pointer wordt bijgehouden wat het beginadres is voor de volgende te laden file. Dit commando is handig in een tweetal situaties.
In de eerste plaats als een aantal kleine programma's in dezelfde EPROM moeten komen te staan. De objectfiles kunnen dan na elkaar in het werkgeheugen geladen worden, waarna de EPROM geprogrammeerd wordt.

2EC- E75	: BASICprogramma
2EC-100B	: BASICprogramma & mlp in Symbol Table
A000-A3FF	: werkgeheugen 1K
A000-A7FF	: werkgeheugen 2K
A000-AFFF	: werkgeheugen 4K
B000-B195	: mlp
B1F5-B1FF	: variabelen & pointers zie sourceprogramma

TABEL 4: geheugen indeling

Verder kan het next-load commando gebruikt worden om een zeer groot assembler programma te laden. Het kan nl. voorkomen dat een assembler programma de maximale buffergrootte van 16K overschrijdt. De buffer is dan te klein om het gehele programma in één keer te bevatten. Door het programma dan op te delen in een aantal stukken en deze afzonderlijk te assembleren kan toch een groot programma gemaakt worden.
Er is echter één maar, indien er vanuit het ene stuk verwijzingen naar het andere stuk voorkomen, door bijvoorbeeld JMP of CALL instructies, dan moet een gedeelte van de symbol table van het ene stuk d.m.v. EQU-statements ook in het andere gedeelte worden opgenomen. Er kan dan zonder problemen van het ene stuk naar het andere verwezen worden.
De beide commando's kennen 3 foutmeldingen. En wel GAP ERROR en CHECKSUM ERROR, die geheel indentiek zijn aan die bij de BASIC LOADER V1.0, en als extra Memory range exceeded. Het kan immers voorkomen, dat bij het laden van een objectfile de geheugengrootte van de geselecteerde EPROM wordt overschreden. Er wordt dan direkt gestopt en de resterende bytes worden dus niet meer overgebracht naar het werkgeheugen. Verder kan het REServe statement (DNA pseudo-op) niet gebruikt worden, load negeert dit commando.

Resumerend is het gebruik van DNA met de programmer als volgt samen te vatten:
Er wordt een assembler programma ingetikt dat op een willekeurige plaats kan beginnen d.m.v. het ORG-statement. Het programma wordt dan geassembleerd en weggeschreven naar tape als objectfile (#0). Dan kan de EPROM-Programmer geladen worden en in utility wordt vervolgens de objectfile ingelezen (>R). Met het Load of Next-load commando kan dan de file naar het werkgeheugen overgebracht worden waarna de laatste stap het programmeren van de EPROM is.

Program: programmeert een EPROM met de data die in het werkgeheugen opgenomen is.

Read: leest een EPROM uit en copieert de inhoud in het werkgeheugen.

Type: wordt gebruikt om één van de 5 types EPROM te selecteren.

Utility: vanuit het programma wordt naar utility gesprongen. Gemakkelijk om het werkgeheugen te bekijken of om een objectfile in te lezen.

Verify: leest de inhoud van een EPROM en vergelijkt deze met de data in het werkgeheugen. Verify dient ter controle van het programmeren. Melding "OK" of "BAD".

De gehele software bestaat uit een sourceprogramma, bijbehorend machinetaal programma, BASIC-programma en de EPROM-programmer. Dit is het machinetaal en BASIC-gedeelte, samengevoegd tot een geheel, in DAInamic nummer 6 blz. 135 is beschreven hoe dit moet gebeuren.

Toepassingen:

De programmer kan gebruikt worden als hulpmiddel bij de ontwikkeling van een microprocessor systeem. Vooral in combinatie met DNA zijn er leuke mogelijkheden om op de DAI software ontwikkeling voor een 8080/8085 µP-systeem te doen. Voor mij was dat de direkte aanleiding tot het ontwerp van de programmer.
Het gebruik van de programmer hoeft niet beperkt te blijven tot de 5 genoemde types. Ombouw naar bijv. de 2708 is eenvoudig mogelijk, terwijl het met een aanpassings-printje mogelijk is een 1 chip processor zoals de 8748 van Intel te programmeren. Hierover in de toekomst, bij voldoende belangstelling, meer.

IC 1	: EPROM
IC 2	: 7812
V1,V2	: BC 548
D1,D2	: 1N4148
B1	: brugcel B40/C1000
R1,R2	: 10k
R3	: potmeter 5k
R4	: 1k5
C1	: 2200 µF 40V
C2	: 4,7 µF 30V tantaal
C3	: 100 nF
L1	: goene LED
Z1	: zekering 100 mA
S1	: miniswitch

Verder: 2x 12V relais, zekeringhouder, trafo 24V 100 mA, flatcable met 34 polige connector, zero insertion force socket, kastje, netsnoer, stukje montageprint.

TABEL 5: benodigdheden

Vragen en opmerkingen
Questions and remarks
Fragen und bemerkungen

Guus Knoops
Siebenstraat 2B
6035 bd Ospel
Nederland
04951-31286

De programmer kan ook bij mij besteld worden.

program identification

title	:	EPROM-PROGRAMMER-DRIVER
author	:	G.KNOOPS
purpose	:	
comment	:	machine language part should be in RAM, hardware should be connected.

```
3     REM ******************************************************
4     REM *  Program for controling the EPROM-PROGRAMMER   *
5     REM *         Copyright GUKNO Software 1981          *
6     REM *                  21 april 1982          V3.0   *
7     REM ******************************************************
8     REM GUKNO Technics, Siebenstr. 2B, 6035 BD Ospel
9     REM Nederland; tel:04951-31286
10    MODE 0:PRINT CHR$(12):PRINT
20    PRINT TAB(40);"EPROM-Programmer"
30    DIM T$(5.0),MEMR%(5.0):RESTORE:FOR X%=1.0 TO 5.0:READ T$(X%),MEMR%(X%):NEXT
40    IF PEEK(#B000)=245 THEN 70:ES%=PEEK(#2A3)+PEEK(#2A4)*256-#B195-1
50    FOR I%=#B000 TO #B195:POKE I%,PEEK(ES%+I%):NEXT
51    REM Move MachineLanguageProgram to workarea
60    POKE #B1F4,1:POKE #B1FD,#A8:POKE #B1FA,0:POKE #B1FB,#A0
61    REM Set: TYPE Number,MEMoryRange
62    REM .    DestinationStartAddresObjectFile
70    TYPE$=T$(PEEK(#B1F4))
80    POKE #FE02,(PEEK(#FE02) IAND #BF):WAIT TIME 5:POKE #FE03,#9B:POKE #FE03,#80:PRINT TYPE$;"
";
81    REM 1:Programmingrelais off, programmingvoltage down
82    REM 2:GIC is cleared and all ports are set to output
90    A%=GETC:WAIT TIME 5
100   REM Commands: Check,Load,Next load,Program,Read,Type
101   REM ========  Utility,Verify
110   A%=GETC:IF A%=ASC("C") THEN 200:IF A%=ASC("L") THEN 300:IF A%=ASC("N") THEN 400:IF A%=ASC
("P") THEN 500:IF A%=ASC("R") THEN 600:IF A%=ASC("T") THEN 700:IF A%=ASC("U") THEN 800:IF A%=ASC
("V") THEN 900
120   GOTO 110
200   REM Check if the EPROM is erased
210   PRINT "Check"
220   GOSUB 1000
230   CALLM #B000
240   A%=PEEK(#B1FF):IF A%=0 THEN CURSOR 16,CURY:PRINT "OK":GOTO 80
250   CURSOR 15,CURY:PRINT "Not erased !":GOTO 80
300   REM Loading DNA objectfile
310   PRINT "Load"
320   POKE #B1FA,0:POKE #B1FB,#A0
321   REM Set DSAOF to #A000
330   GOTO 420
400   REM Loading Next DNA objectfile
401   REM DSAOF is end addres of previous file
410   PRINT "Next load"
420   CALLM #B037
430   A%=PEEK(#B1FF):IF A%=15 THEN CURSOR 15,CURY:PRINT "GAP ERROR":GOTO 80
440   IF A%=240.0 THEN CURSOR 15,CURY:PRINT "Memory Range exceeded":GOTO 80
450   IF A%=255.0 THEN CURSOR 15,CURY:PRINT "CHECKSUM ERROR":GOTO 80
470   GOTO 80
```

```
500   REM Programming the EPROM
510   PRINT "Program":CURSOR 15,CURY:PRINT "Busy !!"
520   GOSUB 1050
530   CALLM #B0E5
540   GOTO 80
600   REM Reading the EPROM
610   PRINT "Read"
620   GOSUB 1000
630   CALLM #B138
640   GOTO 80
700   REM Type EPROM can be changed
710   PRINT "Type:":FOR X%=1.0 TO 5.0:CURSOR 4*X%+6,CURY:PRINT T$(X%):NEXT
720   A%=GETC:WAIT TIME 5:INPUT "Select a type";IN$
730   Y%=0.0:FOR X%=1.0 TO 5.0:IF IN$=T$(X%) THEN Y%=X%
740   NEXT
750   IF Y%=0.0 THEN PRINT :PRINT TAB(14);"Not known type!!":GOTO 80
760   TYPE$=IN$:POKE #B1F4,Y%:POKE #B1FD,(MEMR%(Y%) IOR #A000)/255.0
761   REM Set new TYPE Number and MEMoryRange
770   PRINT :GOTO 80
800   CALLM #B194
900   REM Verifying if the EPROM is programmed correctly
910   PRINT "Verify"
920   GOSUB 1000
930   CALLM #B15E
940   A%=PEEK(#B1FF):IF A%=0 THEN CURSOR 16,CURY:PRINT "OK":GOTO 80
950   CURSOR 15,CURY:PRINT "BAD !":GOTO 80
1000  REM Initiate 8255 (GIC) for Check,Read,Verify
1001  REM GICB=GIC ControlByte;ECB=EPROM ControlByte
1002  REM RB=RelaisByte
1010  GICB%=#90
1020  IF TYPE$="2732" THEN ECB%=#80:RB%=#B0:GOTO 1100
1030  ECB%=#20:RB%=#38:GOTO 1100
1050  REM Initiate 8255 for Programming
1060  GICB%=#80
1070  IF TYPE$="2732" THEN ECB%=#C0:RB%=#D0:GOTO 1100
1080  IF TYPE$="2532" THEN ECB%=#40:RB%=#50:GOTO 1100
1090  ECB%=#58:RB%=#50
1100  POKE #FE03,GICB%:POKE #FE02,RB%:POKE #B1FE,ECB%:WAIT TIME 25:RETURN
10000 DATA 2716,2048,2516,2048,2732,4096,2532,4096,2758,1024
10001 REM Type,MEMoryRange
```

mlp

```
B000 F5 C5 D5 E5 21 01 FE 3A FE B1 57 1E 00 01 00 A0
B010 73 23 72 2B 2B 7E FE FF C2 2D B0 23 13 03 3A FD
B020 B1 B8 C2 10 B0 3E 00 32 FF B1 C3 32 B0 3E FF 32
B030 FF B1 E1 D1 C1 F1 C9 F5 C5 D5 E5 01 02 30 3A 00
B040 30 67 3A 01 30 6F 09 2B 2B 22 F8 B1 2A FA B1 EB
B050 0A FE 00 C2 CB B0 03 0A FE 00 C2 CB B0 03 0A FE
B060 FF C2 C3 B0 2A F8 B1 7C B8 DA C3 B0 C2 74 B0 79
B070 BD D2 C3 B0 21 F5 B1 36 FF 03 0A 86 77 03 0A 86
B080 77 03 0A FE 00 CA 91 B0 03 03 03 03 03 03 C3 50
B090 B0 03 0A 86 77 23 36 00 0A 23 77 2B 2B 03 0A 12
B0A0 13 86 77 3A FD B1 BA CA D3 B0 23 34 7E 23 BE C2
B0B0 9B B0 03 0A 2B 2B BE C2 DB B0 6B 62 22 FA B1 03
B0C0 C3 50 B0 3E 00 32 FF B1 C3 E0 B0 3E 0F 32 FF B1
B0D0 C3 E0 B0 3E F0 32 FF B1 C3 E0 B0 3E FF 32 FF B1
B0E0 E1 D1 C1 F1 C9 F5 C5 D5 E5 21 00 FE 3A FE B1 57
B0F0 1E 00 FE 58 CA FF B0 3E 10 32 FC B1 C3 04 B1 3E
B100 08 32 FC B1 01 00 A0 0A FE FF CA 2A B1 77 23 73
B110 23 72 C5 06 07 0E EB 00 00 00 0D C2 17 B1 05 C2
B120 15 B1 C1 3A FC B1 AA 77 2B 2B 03 13 3A FD B1 B8
B130 C2 07 B1 E1 D1 C1 F1 C9 F5 C5 D5 E5 21 01 FE 3A
B140 FE B1 57 1E 00 01 00 A0 73 23 72 2B 2B 7E 02 03
B150 13 23 3A FD B1 B8 C2 48 B1 E1 D1 C1 F1 C9 F5 C5
B160 D5 E5 21 01 FE 3A FE B1 57 1E 00 01 00 A0 0A 73
B170 23 72 2B 2B BE C2 8A B1 03 13 23 3A FD B1 B8 C2
B180 6E B1 3E 00 32 FF B1 C3 8F B1 3E FF 32 FF B1 E1
B190 D1 C1 F1 C9 CF 04
```

```
PAGE 01    GUKNO Software    21 april 1982  V3.0

002                          ORG   :B000     *** EPROM PROGRAMMER ***
003                 PORT1    EQU   :FE01     PORT 1 OF 8255 (GIC)
004                 *8255 PORT 0: USED AS DATALINES
005                 *      PORT 1: USED AS LOW ADDRESLINES
006                 *      PORT 2: USED AS ADDRES AND CONTROL-SIGNALS
007                 SAWA     EQU   :A000     STARTADDRES WORK AREA
008                 *WORK AREA: A000-A3FF FOR 1K EPROM
009                 *           A000-A7FF FOR 2K EPROM
010                 *           A000-AFFF FOR 4K EPROM
011                 FADR     EQU   :3002     STARTADDRES OBJECTFILE
012                 CHB      EQU   :B1FF     CHECKBYTE
013                 *TESTED BY BASIC PROGRAM
014                 ECB      EQU   :B1FE     EPROM CONTROL BYTE
015                 *SET BY BASIC PROGRAM
016                 MEMR     EQU   :B1FD     MEMORY RANGE
017                 *SET BY BASIC PROGRAM
018                 *INDICATES END OF WORK AREA
019                 EXOR     EQU   :B1FC     EXOR BYTE USED IN PROGRAM
020                 DSAOF    EQU   :B1FA     DSAOF
021                 *DESTINATION START ADDRES OBJECT FILE
022                 *OBJECT FILE IS LOADED FROM THIS ADDRES IN WORK AREA
023                 FLEA     EQU   :B1F8     FILE LENGTH ADDRES
024                 EXREG    EQU   :B1F5     EXTRA REGISTERS
025                 *
026 B000 F5         CHECK    PUSH  PSW       CHECK IF EPROM ERASED
027 B001 C5                  PUSH  B
028 B002 D5                  PUSH  D
029 B003 E5                  PUSH  H
030 B004 2101FE              LXI   H,PORT1
031 B007 3AFEB1              LDA   ECB
032 B00A 57                  MOV   D,A
033 B00B 1E00                MVI   E,0
034 B00D 0100A0              LXI   B,SAWA
035 B010 73          NEXTC   MOV   M,E       LOW ADDRES IS APPLIED
036 B011 23                  INX   H
037 B012 72                  MOV   M,D       ADDRES & CONTROL IS APPLIED
038 B013 2B                  DCX   H
039 B014 2B                  DCX   H
040 B015 7E                  MOV   A,M       DATA IS READ FROM EPROM
041 B016 FEFF                CPI   :FF
042 B018 C22DB0              JNZ   NOTER     NOT ERASED
043 B01B 23                  INX   H
044 B01C 13                  INX   D
045 B01D 03                  INX   B
046 B01E 3AFDB1              LDA   MEMR
047 B021 B8                  CMP   B
048 B022 C210B0              JNZ   NEXTC
049 B025 3E00                MVI   A,0
050 B027 32FFB1              STA   CHB       ERASED!
051 B02A C332B0              JMP   ERASED
052 B02D 3EFF        NOTER   MVI   A,:FF
053 B02F 32FFB1              STA   CHB
054 B032 E1          ERASED  POP   H
```

```
PAGE 02    GUKNO Software    21 april 1982  V3.0

055 B033 D1                 POP   D
056 B034 C1                 POP   B
057 B035 F1                 POP   PSW
058 B036 C9                 RET
059 B037 F5          LOAD   PUSH  PSW       LOAD OBJECTFILE IN WORK AREA
060 B038 C5                 PUSH  B
061 B039 D5                 PUSH  D
062 B03A E5                 PUSH  H
063 B03B 010230             LXI   B,FADR
064 B03E 3A0030             LDA   FADR-2    HIGH ORDER FILE LENGTH
065 B041 67                 MOV   H,A
066 B042 3A0130             LDA   FADR-1    LOW ORDER FILE LENGTH
067 B045 6F                 MOV   L,A
068 B046 09                 DAD   B
069 B047 2B                 DCX   H
070 B048 2B                 DCX   H
071 B049 22F8B1             SHLD  FLEA
072 B04C 2AFAB1             LHLD  DSAOF
073 B04F EB                 XCHG
074 B050 0A          NEXTL  LDAX  B
075 B051 FE00               CPI   :0
076 B053 C2CBB0             JNZ   GAPE      GAP ERROR
077 B056 03                 INX   B
078 B057 0A                 LDAX  B
079 B058 FE00               CPI   :0
080 B05A C2CBB0             JNZ   GAPE
081 B05D 03                 INX   B
082 B05E 0A                 LDAX  B
083 B05F FEFF               CPI   :FF
084 B061 C2C3B0             JNZ   FL        FILE LOADED
085                  *CONTROL IF END FILE IS REACHED
086 B064 2AF8B1             LHLD  FLEA
087 B067 7C                 MOV   A,H
088 B068 B8                 CMP   B
089 B069 DAC3B0             JC    FL        FILE LOADED
090 B06C C274B0             JNZ   NOTJET
091 B06F 79                 MOV   A,C
092 B070 BD                 CMP   L
093 B071 D2C3B0             JNC   FL
094 B074 21F5B1      NOTJET LXI   H,EXREG
095                  *EXREG CHECKSUM/EXR.+1 COUNTER/EXR.+2 LENGTH RECORD
096 B077 36FF               MVI   M,:FF     CHECKSUM BYTE
097 B079 03                 INX   B
098 B07A 0A                 LDAX  B
099 B07B 86                 ADD   M
100 B07C 77                 MOV   M,A
101 B07D 03                 INX   B
102 B07E 0A                 LDAX  B
103 B07F 86                 ADD   M
104 B080 77                 MOV   M,A
105 B081 03                 INX   B
106 B082 0A                 LDAX  B
107 B083 FE00               CPI   0
```

```
PAGE 03    GUKNO Software    21 april 1982  V3.0

108 B085 CA91B0              JZ    NORES     RESERVE TYPE IS NOT USED
109 B088 03                  INX   B         RESERVE TYPE IS IGNORED
110 B089 03                  INX   B
111 B08A 03                  INX   B
112 B08B 03                  INX   B
113 B08C 03                  INX   B
114 B08D 03                  INX   B
115 B08E C350B0              JMP   NEXTL
116 B091 03          NORES   INX   B
117 B092 0A                  LDAX  B
118 B093 86                  ADD   M
119 B094 77                  MOV   M,A
120 B095 23                  INX   H
121 B096 3600                MVI   M,0       CLEAR COUNTER
122 B098 0A                  LDAX  B         LENGTH RECORD
123 B099 23                  INX   H
124 B09A 77                  MOV   M,A
125 B09B 2B          NEXTB   DCX   H
126 B09C 2B                  DCX   H
127 B09D 03                  INX   B
128 B09E 0A                  LDAX  B
129 B09F 12                  STAX  D
130 B0A0 13                  INX   D
131 B0A1 86                  ADD   M
132 B0A2 77                  MOV   M,A
133 B0A3 3AFDB1              LDA   MEMR
134 B0A6 BA                  CMP   D
135 B0A7 CAD3B0              JZ    MEMREX    MEMORY RANGE EXCEEDED
136 B0AA 23                  INX   H
137 B0AB 34                  INR   M
138 B0AC 7E                  MOV   A,M
139 B0AD 23                  INX   H
140 B0AE BE                  CMP   M
141 B0AF C29BB0              JNZ   NEXTB     NEXT BYTE
142 B0B2 03                  INX   B         CONTROL OF CHECKSUM
143 B0B3 0A                  LDAX  B
144 B0B4 2B                  DCX   H
145 B0B5 2B                  DCX   H
146 B0B6 BE                  CMP   M
147 B0B7 C2DBB0              JNZ   CHECKE    CHECKSUM ERROR
148 B0BA 6B                  MOV   L,E
149 B0BB 62                  MOV   H,D
150 B0BC 22FAB1              SHLD  DSAOF     ADJUST DSAOF FOR NEXT LOAD
151 B0BF 03                  INX   B
152 B0C0 C350B0              JMP   NEXTL
153 B0C3 3E00        FL      MVI   A,0
154 B0C5 32FFB1              STA   CHB
155 B0C8 C3E0B0              JMP   END
156 B0CB 3E0F        GAPE    MVI   A,:F
157 B0CD 32FFB1              STA   CHB
158 B0D0 C3E0B0              JMP   END
159 B0D3 3EF0        MEMREX  MVI   A,:F0
160 B0D5 32FFB1              STA   CHB
```

```
PAGE 04    GUKNO Software    21 april 1982  V3.0

161 B0D8 C3E0B0             JMP   END
162 B0DB 3EFF       CHECKE  MVI   A,:FF
163 B0DD 32FFB1             STA   CHB
164 B0E0 E1         END     POP   H
165 B0E1 D1                 POP   D
166 B0E2 C1                 POP   B
167 B0E3 F1                 POP   PSW
168 B0E4 C9                 RET
169 B0E5 F5         PROG    PUSH  PSW       PROGRAMMING OF THE EPROM
170 B0E6 C5                 PUSH  B
171 B0E7 D5                 PUSH  D
172 B0E8 E5                 PUSH  H
173 B0E9 2100FE             LXI   H,PORT1-1
174 B0EC 3AFEB1             LDA   ECB
175 B0EF 57                 MOV   D,A
176 B0F0 1E00               MVI   E,0
177 B0F2 FE58               CPI   :58
178 B0F4 CAFFB0             JZ    E16
179 B0F7 3E10               MVI   A,:10
180 B0F9 32FCB1             STA   EXOR
181 B0FC C304B1             JMP   E32
182 B0FF 3E08       E16     MVI   A,:8
183 B101 32FCB1             STA   EXOR
184 B104 0100A0     E32     LXI   B,SAWA
185 B107 0A         NEXTP   LDAX  B
186 B108 FEFF               CPI   :FF
187 B10A CA2AB1             JZ    NOPROG    NOT NECESSARY TO PROGRAM
188 B10D 77                 MOV   M,A       DATA IS APPLIED
189 B10E 23                 INX   H
190 B10F 73                 MOV   M,E       LOW ADDRES IS APPLIED
191 B110 23                 INX   H
192 B111 72                 MOV   M,D       PROG-SIGNAL IS APPLIED
193 B112 C5                 PUSH  B
194 B113 0607               MVI   B,7       LOOP OF 50 MSEC.
195 B115 0EEB       LOOP2   MVI   C,235
196 B117 00         LOOP1   NOP
197 B118 00                 NOP
198 B119 00                 NOP
199 B11A 0D                 DCR   C
200 B11B C217B1             JNZ   LOOP1
201 B11E 05                 DCR   B
202 B11F C215B1             JNZ   LOOP2
203 B122 C1                 POP   B
204 B123 3AFCB1             LDA   EXOR
205 B126 AA                 XRA   D         STOP PROGRAMMING
206 B127 77                 MOV   M,A       PROG-SIGNAL IS INVERTED
207 B128 2B                 DCX   H
208 B129 2B                 DCX   H
209 B12A 03         NOPROG  INX   B
210 B12B 13                 INX   D
211 B12C 3AFDB1             LDA   MEMR
212 B12F B8                 CMP   B
213 B130 C207B1             JNZ   NEXTP
```

```
214 B133 E1                POP   H
215 B134 D1                POP   D
216 B135 C1                POP   B
217 B136 F1                POP   PSW
218 B137 C9                RET
219 B138 F5         READ   PUSH  PSW       READING THE EPROM
220 B139 C5                PUSH  B
221 B13A D5                PUSH  D
222 B13B E5                PUSH  H
223 B13C 2101FE            LXI   H,PORT1
224 B13F 3AFEB1            LDA   ECB
225 B142 57                MOV   D,A
226 B143 1E00              MVI   E,0
227 B145 0100A0            LXI   B,SAWA
228 B148 73         NEXTR  MOV   M,E
229 B149 23                INX   H
230 B14A 72                MOV   M,D
231 B14B 2B                DCX   H
232 B14C 2B                DCX   H
233 B14D 7E                MOV   A,M
234 B14E 02                STAX  B
235 B14F 03                INX   B
236 B150 13                INX   D
237 B151 23                INX   H
238 B152 3AFDB1            LDA   MEMR
239 B155 B8                CMP   B
240 B156 C248B1            JNZ   NEXTR
241 B159 E1                POP   H
242 B15A D1                POP   D
243 B15B C1                POP   B
244 B15C F1                POP   PSW
245 B15D C9                RET
246 B15E F5         VERIFY PUSH  PSW       VERIFY OF PROGRAMMING
247 B15F C5                PUSH  B
248 B160 D5                PUSH  D
249 B161 E5                PUSH  H
250 B162 2101FE            LXI   H,PORT1
251 B165 3AFEB1            LDA   ECB
252 B168 57                MOV   D,A
253 B169 1E00              MVI   E,0
254 B16B 0100A0            LXI   B,SAWA
255 B16E 0A         NEXTV  LDAX  B
256 B16F 73                MOV   M,E
257 B170 23                INX   H
258 B171 72                MOV   M,D
259 B172 2B                DCX   H
260 B173 2B                DCX   H
261 B174 BE                CMP   M
262 B175 C28AB1            JNZ   BAD
263 B178 03                INX   B
264 B179 13                INX   D
265 B17A 23                INX   H
266 B17B 3AFDB1            LDA   MEMR
```

```
PAGE 06    GUKNO Software    21 april 1982  V3.0

267 B17E B8                  CMP   B
268 B17F C26EB1              JNZ   NEXTV
269 B182 3E00                MVI   A,0        VERIFY OK
270 B184 32FFB1              STA   CHB
271 B187 C38FB1              JMP   OK
272 B18A 3EFF          BAD   MVI   A,:FF      VERIFY BAD
273 B18C 32FFB1              STA   CHB
274 B18F E1            OK    POP   H
275 B190 D1                  POP   D
276 B191 C1                  POP   B
277 B192 F1                  POP   PSW
278 B193 C9                  RET
279 B194 CF            UT    RST   1          JUMP TO UTILITY
280 B195 04                  DATA  4
281 B196                     END

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

BAD     B18A   CHB     B1FF   CHECK   B000   CHECKE B0DB
DSAOF   B1FA   E16     B0FF   E32     B104   ECB    B1FE
END     B0E0   ERASED  B032   EXOR    B1FC   EXREG  B1F5
FADR    3002   FL      B0C3   FLEA    B1F8   GAPE   B0CB
LOAD    B037   LOOP1   B117   LOOP2   B115   MEMR   B1FD
MEMREX  B0D3   NEXTB   B09B   NEXTC   B010   NEXTL  B050
NEXTP   B107   NEXTR   B148   NEXTV   B16E   NOPROG B12A
NORES   B091   NOTER   B02D   NOTJET  B074   OK     B18F
PORT1   FE01   PROG    B0E5   READ    B138   SAWA   A000
UT      B194   VERIFY  B15E
```

This EPROM-programmer is available from Mr. KNOOPS at the price of 258 Gld. This includes : - assembled circuit
- housing
- flatcable + DCE-connector
- zero force insertion socket

comments

We ordered one unit , but at this time there are not yet any test results. As Mr. KNOOPS can programm EPROM's on his DAIpc, I think anyone should succeed with this unit.
While the programmer is not cheap, one should consider that there are a lot of expensive items in the circuit.(zero force,flatcable..)

disadvantages : - the programmer is not interfaced following the DAI-DCE-concept.
- the programmer software can not be loaded from DCR or DISC.
- in the near future, the unit will not be available from stock.

advantages : - the unit will be delivered totally finished, including software on cassette.
- there is no need to connect any hardware inside the housing of the computer.

****

```
10    REM ********************************
11    REM EXEMPLE DE PROGRAMME PRINCIPAL
12    REM ********************************
20    PRINT CHR$(12):MODE 0:POKE #FF05,255:POKE #74,1:POKE #75,95:COLORG 12 0 0
0:COLORT 12 0 0 0
30    GOSUB 60000:REM CHARGEMENT ROUTINE MACHINE
40    PRINT CHR$(12):MODE 0:PRINT "DEFINITION DE LA FENETRE DE TRAVAIL PAR LES C
OORDONNEES DE"
50    PRINT "2 COINS EXTREMES (X1,Y1) ET (X2,Y2):"
60    PRINT :PRINT "NB : X1 doit etre un multiple de 8 et X2 un multiple"
70    PRINT "de 8, moins 1.":PRINT
80    INPUT "Fenetre (X1,Y1,X2,Y2)";X1%,Y1%,X2%,Y2%:PRINT
90    X1%=8*INT(X1%/8):X2%=8*INT((X2%+1)/8)-1
100   INPUT "Nombre de rotations";NROT%:PRINT
110   INPUT "Mode (1/2/3/4/5/6)";M%
120   GOSUB 1000:REM CALCUL DES PARAMETRES
500   REM ****************************************
501   REM FAITES VOTRE DESSIN : LIGNES 5001000
502   REM ****************************************
510   DRAW 0,YMAX/2 XMAX,YMAX/2 0
520   FOR O%=1 TO 200:DOT RND(XMAX),RND(YMAX) 0:NEXT
530   DRAW X1%,Y2% X1%,Y1% 21:DRAW X2%,Y1% X2%,Y2% 21
540   FOR O%=1 TO 50:DOT X1%+RND(X2%-X1%),Y2%+RND(Y1%-Y2%) 21:NEXT
550   CALLM #3009
560   IF GETC=0 THEN 560
570   GOTO 40
999   END
1000  REM **********************
1001  REM CALCUL DES PARAMETRES
1002  REM **********************
1003  REM DONNEES : FENETRE: X1,Y1,X2,Y2
1004  REM ..........MODE: M
1005  REM ..........NOMBRE DE ROTATIONS: NROT
1010  IF M%=1 THEN MODE 1:DE%=#B9EB:LL%=24
1020  IF M%=2 THEN MODE 2:DE%=#B9EB:LL%=24
1030  IF M%=3 THEN MODE 3:DE%=#A8BD:LL%=46
1040  IF M%=4 THEN MODE 4:DE%=#A8BD:LL%=46
1050  IF M%=5 THEN MODE 5:DE%=#6645:LL%=90
1060  IF M%=6 THEN MODE 6:DE%=#6645:LL%=90
1070  IF X1%>X2% THEN T%=X2%:X2%=X1%:X1%=T%
1080  IF Y2%>Y1% THEN T%=Y2%:Y2%=Y1%:Y1%=T%
1090  DEB%=DE%+Y1%*LL%-X1%/4:POKE #3000,DEB% MOD 256:POKE #3001,DEB%/256
1100  DEB2%=DEB%-LL%:POKE #3002,DEB2% MOD 256:POKE #3003,DEB2%/256
1200  NBYT%=(X2%-X1%+1)/4:POKE #3005,NBYT%
1210  NLIGN%=Y1%-Y2%:POKE #3004,NLIGN%
1220  FIN%=DE%+Y2%*LL%-X1%/4:POKE #3006,FIN% MOD 256:POKE #3007,FIN%/256
1230  SAUT%=LL%-NBYT%:POKE #3008,SAUT%
1240  POKE #30A0,NROT% MOD 256:POKE #30A1,NROT%/256
1999  RETURN
60000 DATA #C5,#D5,#E5,#F5,#3A,#5,#30,#47,#11,#0,#31,#2A,#0,#30,#7E,#12
60001 DATA #2B,#13,#5,#C2,#17,#30,#2A,#0,#30,#EB,#2A,#2,#30,#3A,#4,#30
60002 DATA #4F,#3A,#5,#30,#47,#7E,#12,#2B,#1B,#5,#C2,#2E,#30,#3A,#8,#30
60003 DATA #47,#7B,#90,#D2,#40,#30,#15,#5F,#7D,#90,#D2,#47,#30,#25,#6F,#D
60004 DATA #C2,#2A,#30,#2A,#6,#30,#EB,#21,#0,#31,#3A,#5,#30,#47,#7E,#12
60005 DATA #1B,#23,#5,#C2,#57,#30,#2A,#A0,#30,#2B,#22,#A0,#30,#7C,#B7,#C2
60006 DATA #0D,#30,#7D,#B7,#C2,#0D,#30,#F1,#E1,#D1,#C1,#C9
60010 FOR M%=#3009 TO #3074
60020 READ B%:POKE M%,B%
60030 NEXT:RETURN
61001 REM
```

ROUTINE MACHINE ROUTINE MACHINE ROUTINE MACHINE ROUTINE MACHINE ROUTINE MACHINE

TITRE :.....ROTATION D'UNE ZONE D'ECRAN (EN PAGE GRAPHIQUE)
AUTEUR :.....C. POELS
INTERET :.....Peut servir comme sous-routine dans un programme graphique faisant appel a l'animation.

COMMENTAIRES : Le programme suivant fait effectuer a toute zone graphique d'ecran, une rotation vers le haut. La vitesse de cette rotation varie beaucoup suivant la taille de la fenetre definie.

Le programme peut se decomposer comme suit :
Lignes 10 a 110 : Presentation et entree des parametres :
- Coordonnees de deux coins opposes de la fenetre. (X1,Y1,X2,Y2)
- Nombre de rotations vers le haut. (NROT)
- MODE graphique desire. (M)

La ligne 30 fait appel au sous-programme de chargement en memoire de la sous-routine machine.
La ligne 1000 appelle le traitement des parametres et leur stockage en memoire.
Lignes 500 a 1000 : Dessin se trouvant sur l'ecran au moment de la rotation.
Lignes 1000 a 1999 : Traitement des parametres.
Lignes 60000 a 60999 : Routine machine (occupant la zone #3009 a #3074).

****
* DAI pc FIRMWARE MANUAL *
****

The 'DAI pc FIRMWARE MANUAL' can be obtained from: 'Micro Service', Fabritiusstr.15, 6174 RG Sweikhuizen, The Netherlands.

It can be ordered by transferring Hfl. 68.- to bankaccount 13.05.78.754 of 'Micro Service' with the RABO-bank, Geleen, NL or by sending an Eurocheque to the above address.

When other cheques than Eurocheques are used, Hfl.75.- has to be paid due to transfer provision.

Nederlandse clubleden kunnen ook bestellen door overmaking van Hfl.68.- via postgirorekening 1037671 van de RABO-bank te Geleen ten gunste van bovenvermeld bankrekeningnummer.

super noise generator

BLADWIJZER

COPYRIGHT 1982 BY DAInamic Belgium.

--- with many thanks to R.Rens---

W.De Winter R.Corswandt

```
**************************************************************
***                                                        ***
***    S U P E R   N O I S E   G E N E R A T O R           ***
***                                                        ***
***                 copyright DAInamic                     ***
**************************************************************
```

OPMERKINGEN.

ALGEMEEN ADVIES : KONTROLEER STEEDS UW WERK VOORDAT U
EEN VOLGENDE STAP DOET.

Noch DAInamic, noch de heren De Winter & Corswandt kunnen verantwoordelijk gesteld worden voor beschadigingen die zouden voorkomen tijdens het installeren of het gebruiken van de Super Noise Generator.

1/ PRINT BESTUKKEN.

Kontroleer de print op eventuele onderbroken of kortgesloten banen (vergelijk met koper-layout, figure 2B)
Bestuk de print volgens het komponentenschema (zie figure 2A)
De onderdelen plaats je aan de niet-verkoperde kant
Gebruik een soldeerbout met fijne punt en een vermogen kleiner dan 40 watt.
Indien nodig plooi dan de komponenten met een tangetje zorgvuldig op maat.

Plaats en soldeer de komponenten volgens grootte :

eerst de vier draad bruggen (W1 tot W4), ontmantel het stijve draadje, knip en plooi op maat.
alle weerstanden (R1-R17) insteken (voor de waarde, gebruik TABLE 4).
print omdraaien en solderen
-hierdoor zitten deze mooi op gelijke hoogte
dan de IC-voetjes,soldeer eerst twee pootjes en controleer of ze volledig tegen de print aanzitten, eventueel corrigeren
-let op of hun inkeping aan de juiste kant zit (zoals de inkeping op het ic).
en als volgende de kondensatoren (C1-C9)
-bij de elektrolytische geeft een zwarte band de minzijde aan
-bij de druppel-tantaalkondensator C9 geeft een +teken de pluskant aan
-de andere kondensatoren hebben geen richtingsvoorkeur, voorzichtig aanbrengen, zodat de aansluitdraden niet loskomen .
soldeer als laatste de trimmer P1.

Knip de uitstekende draadjes af (lengte maximum 3mm).

DE IC's NOG NIET AANBRENGEN !!!!!!

Kontroleer of dat alle komponenten op de juiste plaats zitten.
Kijk het soldeerwerk na op eventuele fouten.

2/ INBOUWEN VAN DE PRINT.

Verbreek eerst alle verbindingen met de DAI-pc.
Trek de vier zwarte pluggen uit de zijkant van de kast.
Hef het bovendeel aan de voorkant op,en duw het naar achter, verwijder voorzichtig een eventueel aardingsdraadje tussen de metalen plaat rond het toetsenbord en de 0V.

Het inbouwen doen we aan de voorzijde van de voeding (zie FIGUUR 6).
Indien u een Digitale Cassette-Recorder bezit trek dan eerst het eprom-kaartje uit de X-bus.

Schema 2

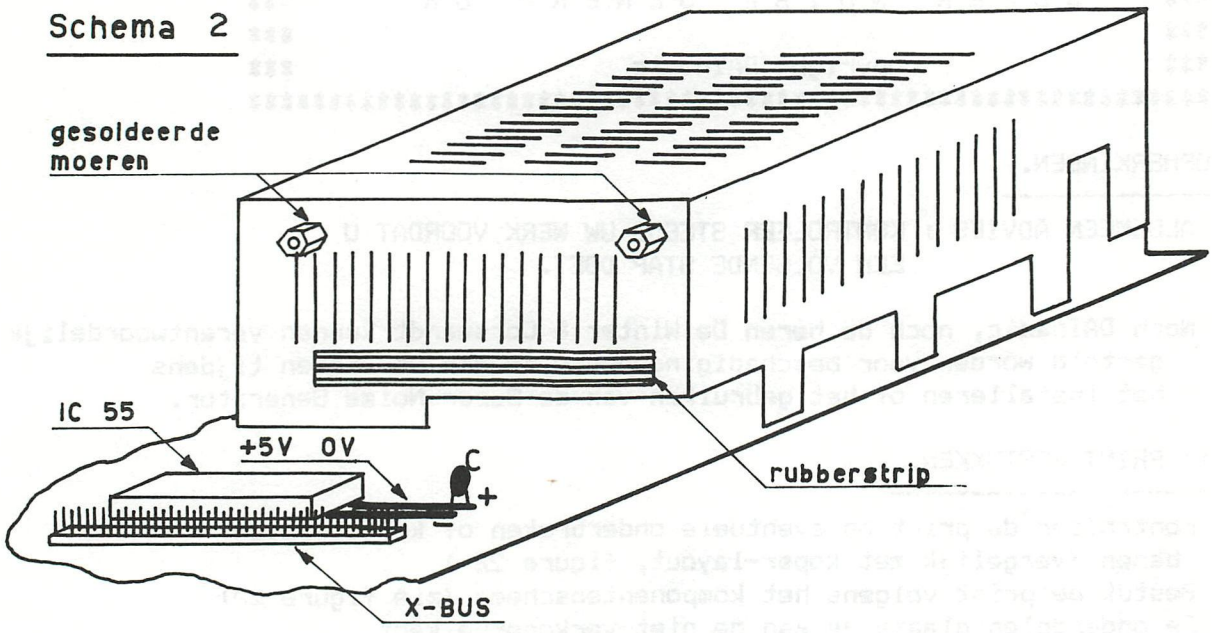


FIGURE 6

2.1/ Aanbrengen van de boutjes.

Lijm de rubber strip (zelfklevend) onderaan de voorzijde van de voeding.
Dit dient als isolerend afstandsstuk.

Verwijder het deksel van de voeding (dit om gemakkelijk te werken).

Schroef de moeren in de hiervoor voorzienen gaten op de SNG-print.
Alzo krijgen we de juiste maat om de voorzijde van de voeding te vertinnen.

Gebruik hiervoor liefst een zwaardere soldeerbout dan deze waarmee u
de komponenten soldeert (vermogen >40 watt).

Plaats de DAI-print vertikaal, met de achterzijde op de tafel,
plaats de SNG-print horizontaal (met de komponenten boven), op de
voorzijde van de voeding (VGL. FIGUUR 6).
Soldeer de moeren vast (zorg voor een goede vloeiing van het soldeer).
Kontroleer of er geen uitstekende draadjes kontakt maken met de voeding.

Schroef nu de print terug los.

NOTA !!

Wanneer er een koelplaatje gemonteerd is op de voorzijde van de voeding,
dan zijn er volgende oplossingen mogelijk:

- verwijder het koelplaatje (bij de meeste versies niet gebruikt).
- soldeer een metalen stripje aan de rechterkant van de voeding,
zodat het SNG-printje juist naast het koelplaatje komt en iets
voorbij de DAI-print.

2.2/ Aansluiten van de print.

a) verbinden van de kabel aan de S.N.G.-print.

Ontmantel nu de meegeleverde kabel zodat een 10 cm buitenmantel overblijft, dit stuk kan gepast heen en weer geschoven worden.

Trek een rode en een zwarte draad uit de buitenmantel en soldeer deze respektievelijk aan de +5 volt en de 0 volt van de print (zie FIGURE 2A).

Er zijn twee rijen aansluitpunten,deze zijn aangeduid door de letters A tot en met K (zie FIGURE 2A).
Soldeer de overige draadjes in de kabel, aan deze punten van de SNG-print. Gebruik hiervoor TABLE 1 als standaard.
Verwijder de eventueel overtollige draadjes nog niet uit de mantel, ze kunnen nog als reserve dienen bij een eventuele verkeerde handeling.

b) verbinden van de kabel aan de DAI-print.

Gebruik hiervoor FIGURE 5 om de punten terug te vinden.
Eerst verbindt je de rode en de zwarte draad. Dit kan je best doen rechts van het IC (IC 55 op FIGURE 5), tussen de voeding en de X-bus. Hier bevindt zich een condensatortje met aanduiding + en - (dwz resp. +5V en 0V).Krab zorgvuldig de lak van de banen en soldeer de rode (+5V) en de zwarte (0V) draad erop.

Schroef het printje op de moertjes en soldeer de overige draadjes op de aangeduide poten van de komponenten (zie FIGURE 3,4,5).
Daar de DAI-firma niet altijd dezelfde componenten gebruikt is soms een tweede plaats aangeduid als alternatief. U soldeert daar waar u het beste bijkan.
Om punt J te verbinden dient u de TV-kaart (print op steuntjes) weg te nemen.Let er bij het terug monteren van deze kaart op dat er geen kontakten verbogen zijn in de konnektor .

* CONTROLEER NOGMAALS OF ALLES JUIST AANGESLOTEN IS, EN OF ER GEEN KORTSLUITINGEN GEMAAKT ZIJN !!!

* PLAATS NU DE IC's IN HUN RESPECTIVELIJKE VOETJES, LET OP DE POLARITEIT !! OPPASSEN VOOR STATISCHE LADINGEN !!!

Als er geen fouten gemaakt zijn moet de SUPER NOISE GENERATOR nu volledig in orde zijn, tijd nu voor een korte test.

3/ TESTEN VAN DE SUPER NOISE GENERATOR.

Plaats het deksel van de voeding terug op zijn plaats. Zet het eventuele DCR eprom-kaartje op de X-bus. Maak de overige verbindingen in orde (KAST NIET SLUITEN !). Schakel de TV en de DAI-PC aan.

3.1/ AFREGELEN VAN TRIMMER P1 (cassette-geluid).

Sluit een cassette-recorder op CAS1 aan, plaats P1 in de middenstand.
Start een normale muziek-cassette, en tik in POKE#40,0 : de cassette start en er moet muziek hoorbaar zijn op de TV.
Regel nu trimmer P1 af op een normaal geluidsniveau.

* NOTA: als er veel vervorming optreed, regel dan het geluidsvolume op de recorder.

3.2/ EEN KORTE TEST (new noise).

geef NEW en tik in : * 10 ENVELOPE 0 16
* 20 NOISE OFF:WAIT TIME 1
* 30 NOISE 0 7:WAIT TIME 200
* 100 GOTO 20
* RUN

En nu hoort u een wegstervende lage ruisexplosie

3.3/ VOLLEDIGE TEST VAN DE SUPER NOISE GENERATOR.

Start de meegeleverde demonstratie-cassette.
Gebruik bij voorkeur een recorder met aangesloten remote-control,
alle programma's laden automatisch het volgende.
De band bevat volgende programma's :

0 SNG START
0 SNG DEMO+TEST : test alle bijkomende noise commando's
0 VIERTAKT MOTOR BOOTSTRAP LOADER
1ML VIERTAKT MOTOR 300 350
0 BASIC VIERTAKT MOTOR MET SNG : programma voorbeeld noise
0 CASSETTE SOUND DEMO : programma voorbeeld cassette sound
MUZIEK

4/ SOFTWARE STURING VAN DE UITBREIDING.

De uitbreiding met de S.N.G. is praktisch altijd kompatibel met vroegere muziekprogramma 's.
Dit doordat de bijkomende hardware gebruik maakt van de onderste volumeniveaus (1-7) van de normale DAI-noise.

4.1/ EXTRA NOISE COMMANDO'S

Het uiteindelijke DAI-volume is een kombinatie van NOISE-volume en ENVELOPE-volume (zie tabel 1).
Deze kombinatie kunt u te weten komen via (enkel voor noise commando):

```
ENVELOPE X Y:NOISE X Z:WAIT TIME 10:?HEX$(PEEK(#295) SHR 4.
```

Voor de besturing van de SNG is het eenvoudigste het envelope-volume op 16 te zetten en het noise-volume te gebruiken van 0 tot 7 (zie programma voorbeelden).

Bijkomende effecten verkregen door de S.N.G.

NOISE X 0 : S.N.G. reset, NOISE OFF; dit kommando of NOISE OFF dient u steeds te gebruiken voor u de niveaus 4, 5, 6 of 7 gebruikt.
NOISE X 1 : kontinu lage ruis.
NOISE X 2 : kontinue ruis met regelbare frequentie(F) dit gebeurt via SOUND 2 X 0 K FREQ(F) (X=ENVELOPE-NR;K=Kies 0,1,2 of 3)
NOISE X 3 : kontinu hoge ruis.
NOISE X 4 : hoge, snel wegstervende ruis
NOISE X 5 : hoge, traag uitstervende ruis
NOISE X 6 : lage, snel wegstervende ruis
NOISE X 7 : lage, traag uitstervende ruis

NOISE X 8 tot NOISE X 15 :normale DAI-ruis.

4.2/ CASSETTE SOUND STURING.

Een extra mogelijkheid van SNG bestaat erin om het geluidssignaal,komende van CAS1 door te schakelen naar de T.V. en stereo output.Dit onder software controle, doormiddel van POKE's.Dit zowel vanuit command-mode als uit BASIC.

-1) POKE#40,0 : start cassette 1 + 2 ,
+ doorgeven van cassettegeluid naar TV en stereo output.

POKE#40,#10 : stop cassette 2, cassette 1 op ; geen doorgeven.

POKE#40,#20 : stop cassette 1, cassette 2 op ; geen doorgeven.

POKE#40,#30 : stop cassette 1 + 2 ; geen doorgeven.

-2) Wanneer een van volgende POKE's gegeven is , wordt er, bij een LOAD opdracht, de aangeduide cassette geselecteerd.

POKE#13D,#10 : selecteer cassette 1

POKE#13D,#20 : selecteer cassette 2

POKE#13D,#30 : selecteer beide cassettes; doorgeven van geluid.

-3) DCR gebruikers kunnen, inplaats van bovenstaande POKE's, gebruik maken van de volgende commando's:

CAS / CAS1 /CASSETTE / CASSETTE 1 :selecteer cassette 1

CAS2 / CASSETTE 2 :selecteer cassette 2

CAS3 / CASSETTE 3 :selecteer cassette 1 +2 ;
doorgeven geluid.

5/PRINCIEPE WERKING.

Gebruik hiervoor FIGURE 1,3,4 en TABLE 4 .

5.1/ WERKING VAN DE NOISE UITBREIDING.

De eigenlijke schakeling bestaat uit 3 delen :

- de adres-decodering + besturing ; gevormd door de prom IC1.
- de wisselschakeling (voor digitale niveau's) en signaal oppoetser IC3 (N1...N4).
- en het eigenlijke 'hart' van de schakeling, de 'COMPLEX SOUND GENERATOR' IC2 met toebehoren.

A) adres-decodering :

Op de adress select ingangen van IC1 komen de vier data-bits toe die het volume van de normale DAI-noise regelen.Elk adres levert een bepaald bitpatroon aan de acht data uitgangen, die elk een bepaalde functie sturen van het sound-IC (IC2).Bestudeer hiervoor TABLE 2.
Het adres valid signaal (pin 15) word steeds op grond potentiaal gehouden, zodat elke verandering aan de adres ingangen een ander datawoord doet verschijnen aan de data uitgangen.

B) de schakeling rond IC3.

De schakeling selecteert een signaal,ofwel opgewekt door SOUND 2,ofwel van de VCO oscillator gevormd door C5,R17 + een deel van IC2 en de stuurspanning gekreerd door R10,R11.De selectie gebeurd door data-bit D1 van IC1. Het uitgangssignaal wordt gebruik om de noise oscillator in IC2 te sturen. Om de signalen terug op TTL niveau te krijgen is er gebruik gemaakt van smith-trigger inputs.

C) de 'COMPLEX SOUND GENERATOR'.

Gebruik hiervoor ook de DATA-sheets! (vooral het blokschema PAGE IX)

Via pin 3 van IC2 wordt het clock-signaal (verkregen via IC3) toegevoerd aan de noise generator.

Dit pseudo random signaal wordt vervolgens door de noise fiter gevoerd.De kantel-frequentie word bepaald door R12,C4 ,waarvan C4 m.b.v. D4 (IC1) wel of niet aangesloten wordt, (de datauitgangen van IC1 zijn open collectoruitgangen), wat resulteerd in een hoge of lage ruis.

Vervolgens wordt dit signaal aangeboden aan de mixer, waarvan enkel het noise kanaal gebruikt is (vast geselecteerd via pin 25,26,27).

Nu wordt in de envelope generator de stijg- en daaltijd van het noisesignaal bepaald, de volledige duurtijd gebeurd door het one shot circuit (zie hieronder).Door middel van de envelope select logic (pin 1,28 IC2) kan men verschillende vormen selecteren,de selectie gebeurd door D5,D7 IC1

Als men de one shot logic selecteerd, is de totale duurtijd van het ruissignaal vast bepaalt door R13,C8, evenals de stijgtijd ,omdat R15 een zeer lage weerstands waarde heeft, is de verandering van de capaciteit aan pin 8 van weinig invloed.Anders is dit bij de afvaltijd,dan wordt R16 gebruikt en dan is de verandering van de capaciteit aan pin 8 wel van belang. De waarde van deze capaciteit is te veranderen door D3 (IC1),als D3 laag is wordt C6 parallel geplaatst met C7, wat resulteerd in een langdurige ruis.

Ook moet de one shot na elke cyclus opnieuw gestart worden,dit gebeurd door pin 9 'even' hoog te maken (gestuurd door D6 IC1).

Dit 'behandelde' ruissignaal wordt nu op niveau gebracht door de amplifier en zijn instel weerstanden R8,R9,R14.

Om de originele DAI-ruis zomin mogelijk te storen is er S1 toegevoegd. Deze 'solid state' schakelaar word enkel gesloten als er een van de nieuwe noise mogelijkheden aangevraagd is (gestuurd door D2 IC1).

5.2/ CASSETTE STURING.

De schakeling wordt gevormd door de volgende onderdelen:

C1,C2 ;ontkoppel condensators
R1,R2 ;instelling
P1 ;volume regeling
S2,S3,S4;solid state schakelaars

Als beide cassette motors afstaan, zijn de de punten H en I laag. Hierdoor is de stuurspanning van S4 ook laag (pin 12). S4 wordt pas geleidend als beide motors gestuurd worden, in dit geval wordt het geluid komende van de cassette recorder toegevoerd aan IC14 (zie hiervoor FIGURE 3),en meegemoduleerd op het tv kanaal (ook naar de stereo plug).

6/ VOORBEELDEN.

In dit hoofdstuk worden alle besturingen nog eens op een rijtje gezet,en tevens enkele voorbeelden gegeven.

6.1/ DE COMMANDO'S.

A) super noise generator.

```
NOISE X 0 : SNG reset,noise off.
NOISE X 1 : continuous, low noise.
NOISE X 2 : continuous, variable noise (sweep).
NOISE X 3 : continuous, high noise.
NOISE X 4 : high, short decay noise.
NOISE X 5 : high, long  decay noise.
NOISE X 6 : low , short decay noise.
NOISE X 7 : low , long  decay noise.

NOISE X 8 to NOISE X 15 : normal DAI noise.
```

RESET : NOISE OFF of NOISE 0 0 :WAIT TIME 1

B) cassette sturing.

```
-direct command: POKE#40,0     ; CAS1 + CAS2 + SOUND
                 POKE#40,#10   ; CAS2 OFF; CAS1 ON ; SOUND OFF
                 POKE#40,#20   ; CAS2 ON ; CAS1 OFF; SOUND OFF
                 POKE#40,#30   ; CAS2 OFF; CAS1 OFF; SOUND OFF
-cassette mask : POKE#13D,#10 ; CAS1 SELECTED ; SOUND OFF
                 POKE#13D,#20 ; CAS2 SELECTED ; SOUND OFF
                 POKE#13D,#30 ; CAS1 + CAS2 SELECTED ; SOUND ON
```

6.2/ PROGRAMMA VOORBEELDEN.

- STENGUN 1.

```
10   ENVELOPE 0 6,10;0,2;3,24;0,2;
20   NOISE 0 15
```

*** met de 'envelope' selecteerd men hier een lage, korte ruis gedurende 32mS gevold door 6,4mS reset.Dan een doorlopende, hoge ruis gedurende 76,8mS terug gevolgd door 6,4mS reset, waarna men terug begint.

- STENGUN 2.

```
10   0 3,13;0,6;
20   NOISE 0 15
```

- EXPLOSION.

```
10   ENVELOPE 0 7,200;0
20   NOISE 0 15
```

*** daar de 'envelope' hier niet afgesloten is met ';' zal de explosin niet herhaald worden.

- PISTOL AND RIFLE

```
10   ENVELOPE 0 4,120:0,3;6,20;0,20;6,20;0,25;
20   NOISE 0 15
```

*** merk op dat elke nieuwe noise aanvraag voorafgegaan is van een reset gedurende minstens 6,4mS.Dit is noodzakelijk daar de interpreter geen volume verandering toelaat groter dan de helft van het bestaande volume.

- FLAK-KANON.

```
10    ENVELOPE 0 6,23;0,2;5,10;0,4;
20    NOISE 0 15
```

- ????????.

```
10    ENVELOPE 0 6,3;0,2;3,9;0,2;
20    NOISE 0 15:WAIT TIME 40:NOISE OFF:WAIT TIME 1
30    ENVELOPE 0 3,2;0,2;6,6;0,2;
40    NOISE 0 15:WAIT TIME 80:NOISE OFF:WAIT TIME 1
50    GOTO 10
```

*** Het is ook mogelijk om de 'envelope' te veranderen.

- DRUM'S.

```
10    ENVELOPE 0 3,6;0,7;3,6;0,9;3,5;0,8;3,6;0,30;
20    NOISE 0 15
```

- ROCKET.

```
10    ENVELOPE 0 16:NOISE 0 2
20    FOR X=100 TO 25000 STEP 100:SOUND 2 0 0 0 FREQ(X):NEXT
30    SOUND 2 OFF:WAIT TIME 15:GOTO 20
```

*** Met behulp van het commando NOISE X 2 selecteerd men de mogelijkheid om met SOUND 2 de clock frequentie te regelen, wat resulteerd in een variabele noise.(SOUND 2 wordt dan volledig onderdrukt door de SNG)

- THE OCEAN.

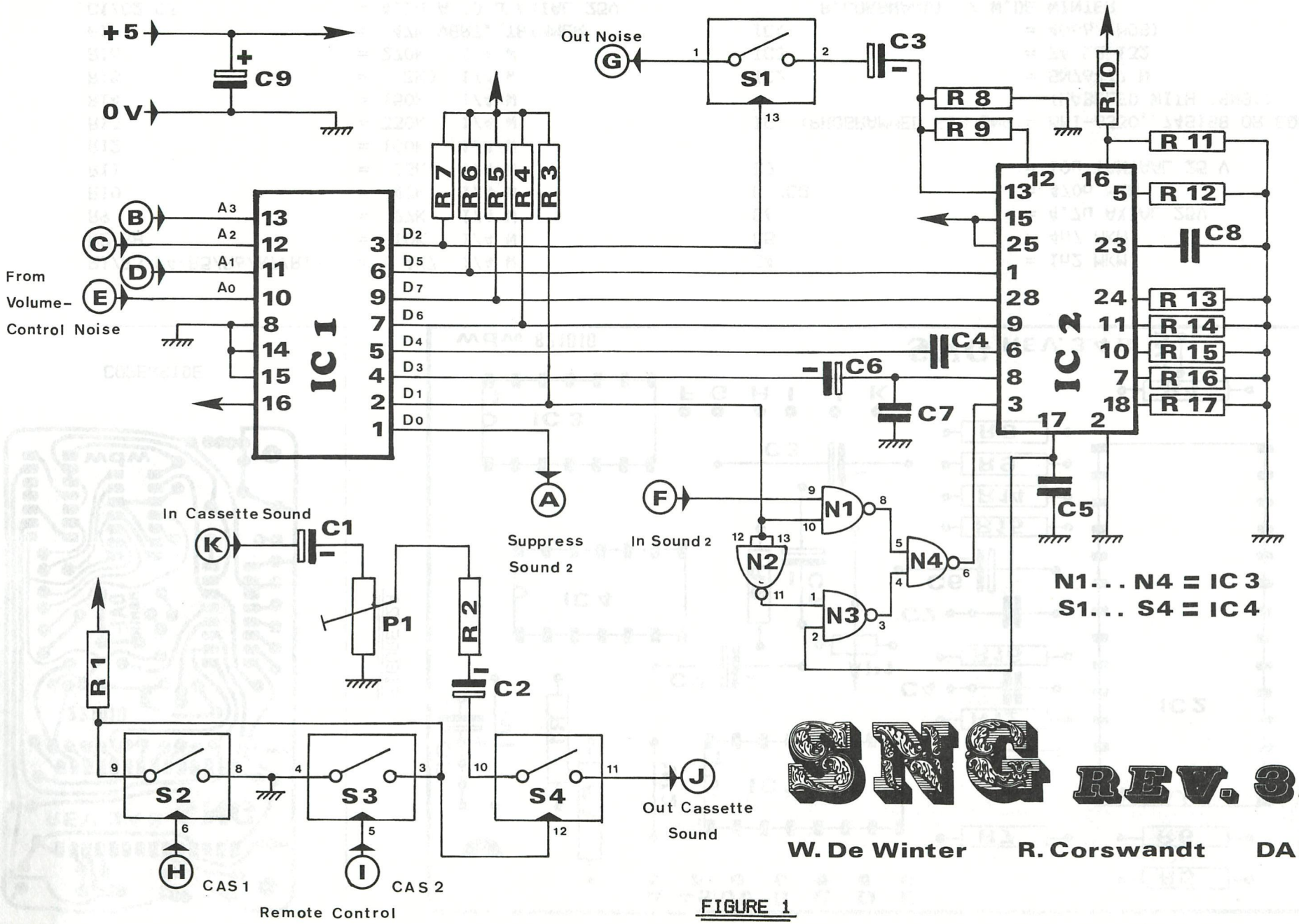
```
10    ENVELOPE 0 16:NOISE 0 2
20    FOR X=5 TO 200:SOUND 2 0 0 0 X:WAIT TIME 1:NEXT
30    FOR X=200 TO 5 STEP -1:SOUND 2 0 0 0 X:WAIT TIME 1:NEXT
40    GOTO 20
```

*** Als men de functie 'FREQ' niet gebruikt dan wordt de waarde van de variabele rechtstreeks gebruikt om het sound ic te sturen. De waarde van de variabele komt dan wel niet mee overeen met de frequentie (voor de juiste berekening zie het handboek)

!!!!!!! OOK IS HET MOGELIJK OM BIJ GEBRUIK VAN DE CASSETTE UITBREIDING TE 'KIJKEN' OF ER NOG GELUID OP DE BAND STAAT, EN DIT DOOR DE STATUS VAN DE CAS.BIT TE CHECKEN, DIE GEBRUIKT WORDT OM EEN PROGRAM TE LADEN.

Een manier is de hier onder vermelde:

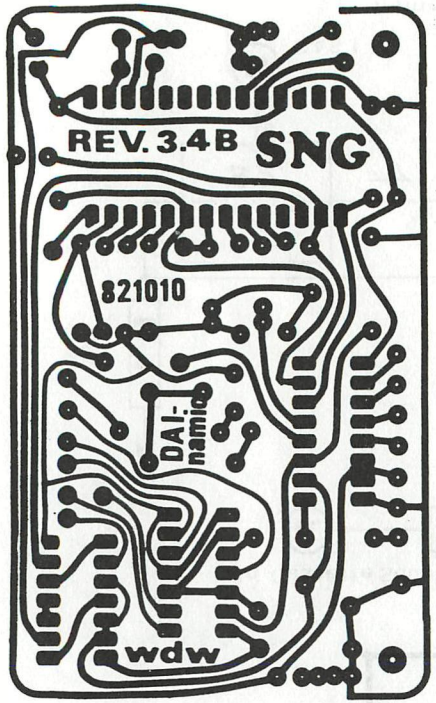
```
10    B%=PEEK(#FD00) IAND #A0    ; READ BEGIN STATUS OF CAS.BIT
20    A%=PEEK(#FD00) IAND #A0    ; READ STATUS OF CAS.BIT
30    C%=C%+1                    ; INCREMENT COUNTER
40    IF A%<>B% THEN C%=0        ; WANNEER VERANDERING COUNTER=0
50    IF C%>300 THEN 70          ; EXIT PROG. WANNEER GEDURENDE ONGEVEER
                                 ;  5 sec.GEEN VERANDERING OPGETREDEN IS
60    B%=A%:GOTO 20              ; ONTHOU DE OUDE TOESTAND EN CHECK NOGMAALS
70    ????                       ; EXIT
```



SNG REV. 3.4

W. De Winter R. Corswandt DAInamic

FIGURE 1



COPERSIDE

FIGURE 2B

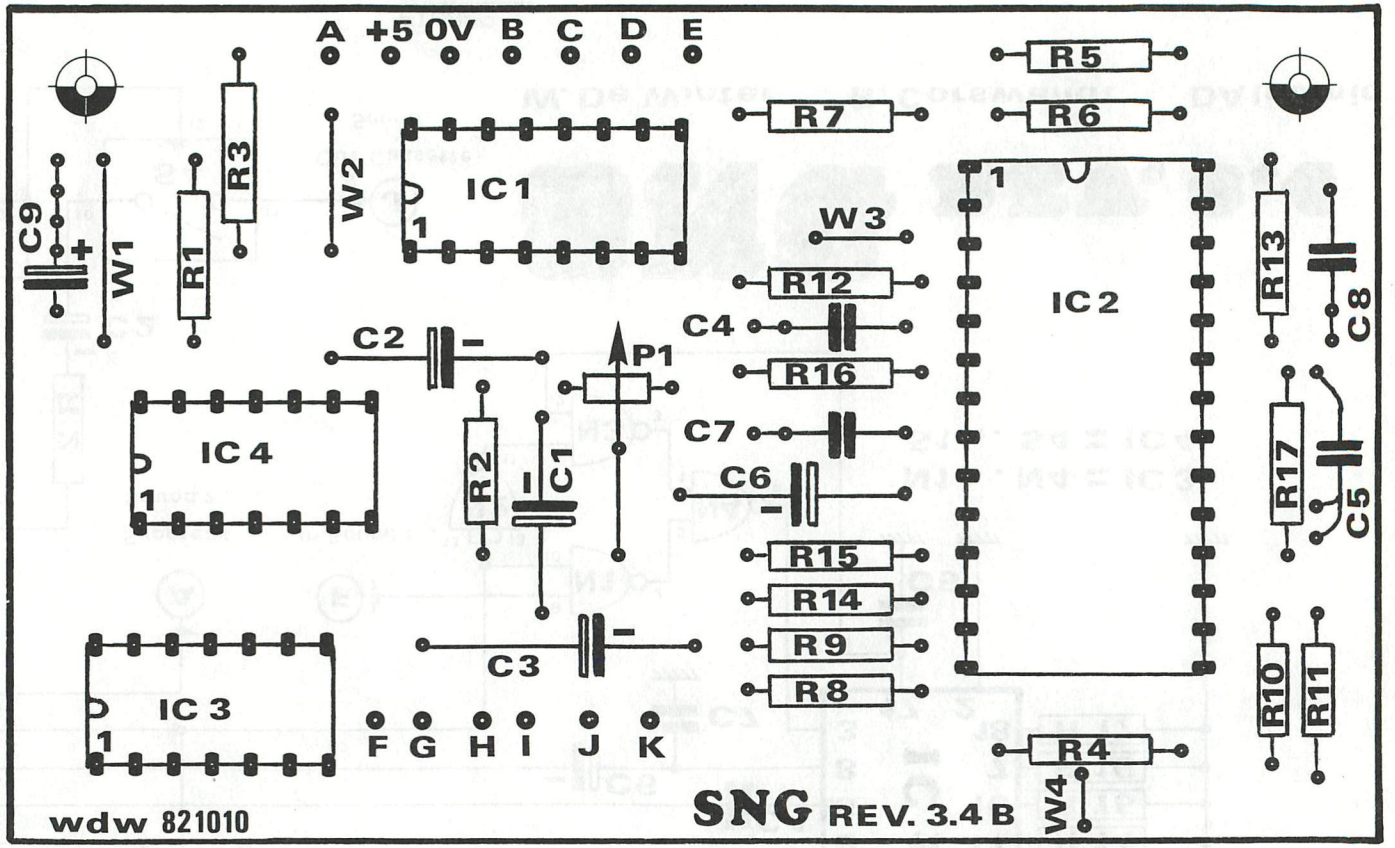


FIGURE 2A

R1/R3/R4/R5/R6/R7/R17	=	4K7	1/4 W	C4	= 1n2 MKM
R2/R8	=	10K	1/4 W	C5	= 4n7 MKM
R9	=	27K	1/4 W	C6	= 4.7u AXIAL 25V
R10	=	47K	1/4 W	C7/C8	= 470n MKM
R11	=	33K	1/4 W	C9	= 10u TANTAAL 25 V
R12	=	100K	1/4 W		
R13	=	330K	1/4 W	IC1 (PROGRAMMED FOR SNG)	= MMI-6330, 74S188 OR EQUI. (LABELED WITH 'SNG')
R14	=	150K	1/4 W		
R15	=	3K3	1/4 W	IC2	= SN76477 N
R16	=	270K	1/4 W	IC3	= 74 LS 132
P1	=	47K VERT. TRIMMER		IC4	= 4066 (MOS)
C1/C2/C3	=	4.7u A 10 u AXIAL 25V			

R.CORSWANDT / W.DE WINTER

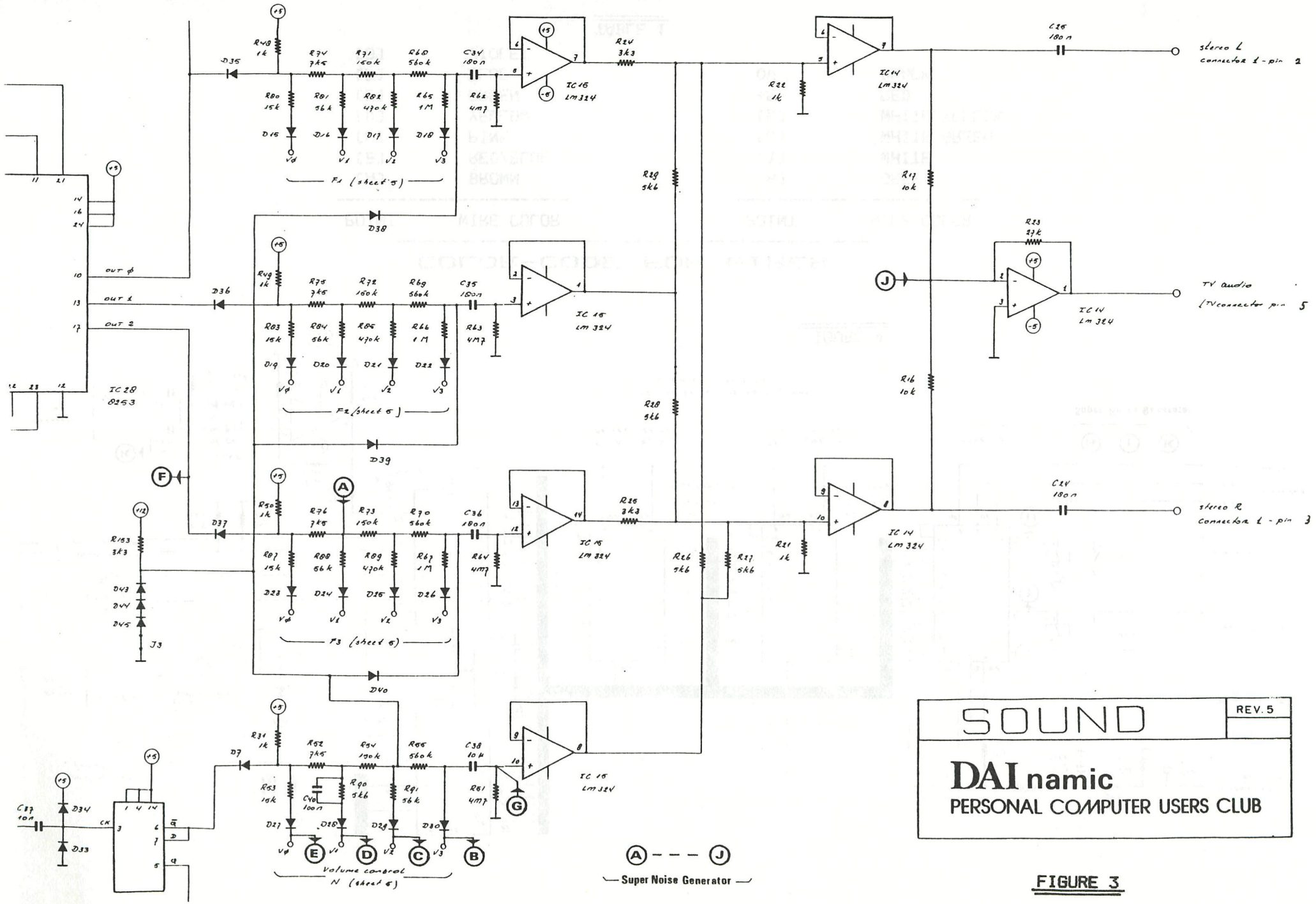

SOUND
REV. 5
DAInamic
PERSONAL COMPUTER USERS CLUB
IC 28 8253
IC 15 LM 324
IC 16 LM 324
IC 14 LM 324
stereo L connector 1 - pin 2
TV audio (TV connector pin 5
stereo R connector 1 - pin 3
F1 (sheet 5)
F2 (sheet 5)
F3 (sheet 5)
Volume control N (sheet 5)
A - - - J
Super Noise Generator

FIGURE 3

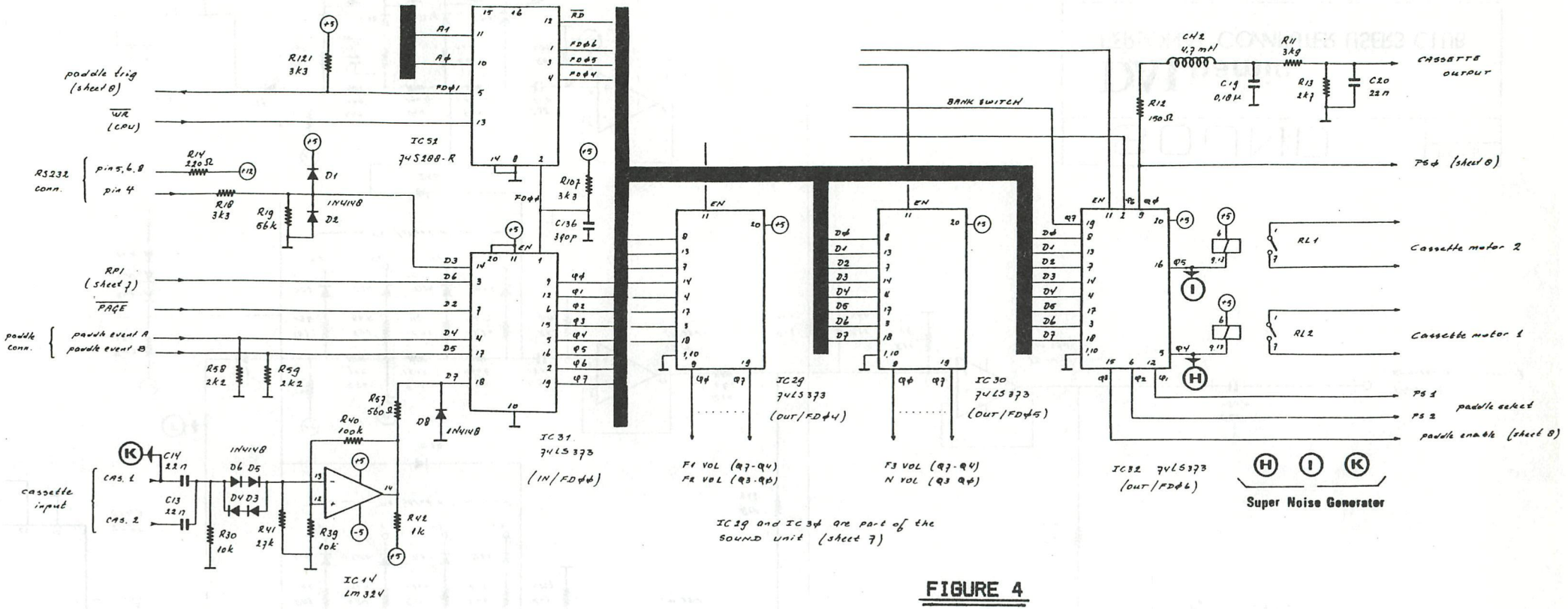
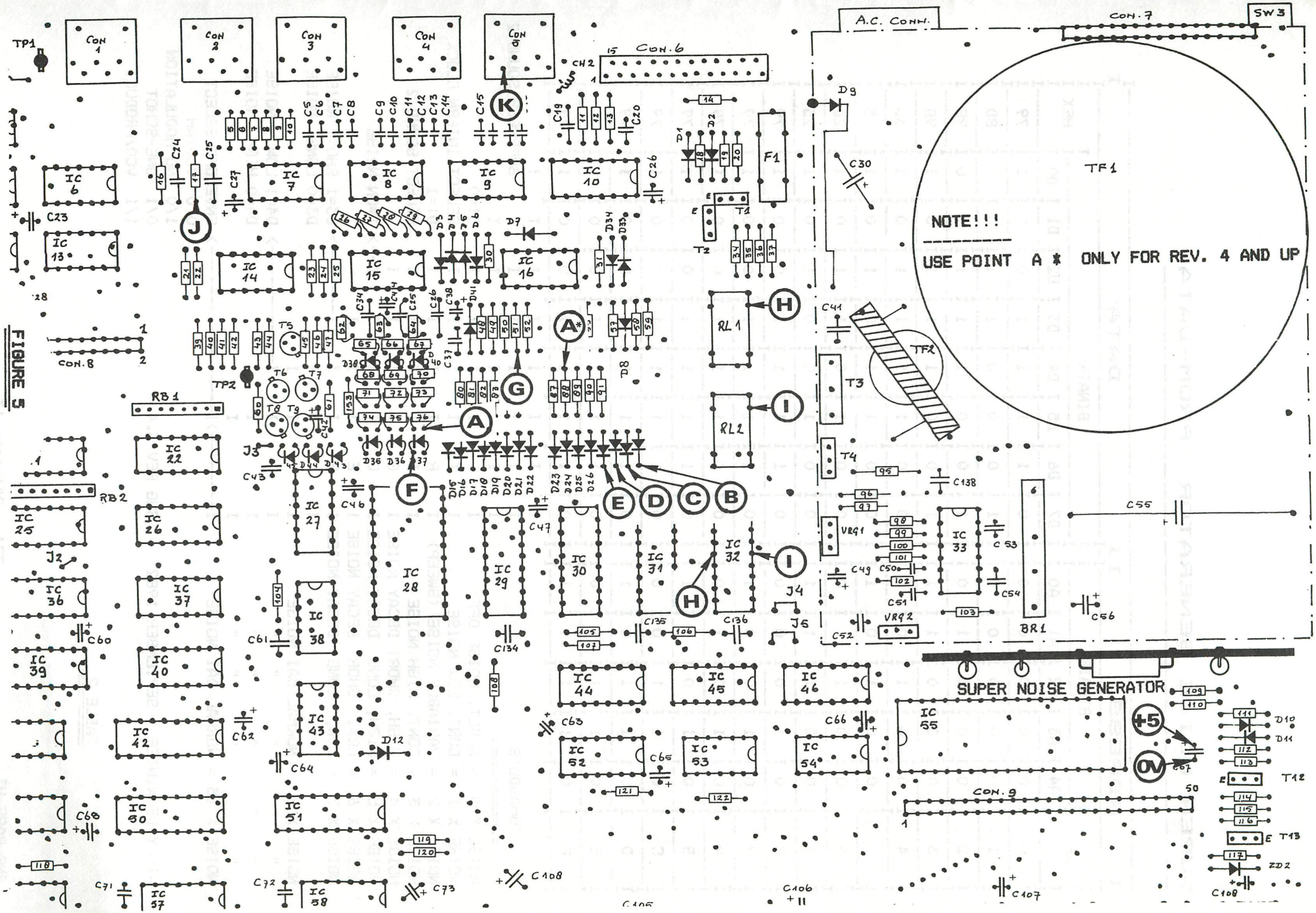


FIGURE 4

COLOR-CODE FOR WIRES

POINT	WIRE COLOR	POINT	WIRE COLOR
[A]	BROWN	[H]	GREY
[B]	RED/BLUE	[I]	WHITE
[C]	PINK	[J]	WHITE/GREEN
[D]	YELLOW	[K]	WHITE/YELLOW
[E]	GREEN	+5	RED
[F]	BLUE	0V	BLACK
[G]	VIOLET		

TABLE 1


A.C. CONN.
CON.7
SW3
TF1
NOTE!!!
USE POINT A * ONLY FOR REV. 4 AND UP
SUPER NOISE GENERATOR
CON.6
CON.8
CON.9
RL1
RL2
T1
T2
T3
T4
F1
D9
VR1
VR2
BR1
+5
0V
FIGURE 5
373
SNG PAGE V

SUPER NOISE GENERATOR PROM-DATA

ADRESS							DATA								
	BINAIR						BINAIR								
HEX	A4	A3	A2	A1	A0		D7	D6	D5	D4	D3	D2	D1	D0	HEX
0	0	0	0	0	0		0	1	1	1	1	0	0	1	79
1	0	0	0	0	1		1	0	0	0	1	1	0	1	8D
2	0	0	0	1	0		1	0	0	1	1	1	1	0	9E
3	0	0	0	1	1		1	0	0	1	1	1	0	1	9D
4	0	0	1	0	0		0	0	1	1	1	1	0	1	3D
5	0	0	1	0	1		0	0	1	1	0	1	0	1	35
6	0	0	1	1	0		0	0	1	1	0	1	0	1	2D
7	0	0	1	1	1		0	0	1	0	0	1	0	1	25
8	0	1	0	0	0		0	1	1	1	1	0	0	1	79
9	0	1	0	0	1		0	1	1	1	1	0	0	1	79
A	0	1	0	1	0		0	1	1	1	1	0	0	1	79
B	0	1	0	1	1		0	1	1	1	1	0	0	1	79
C	0	1	1	0	0		0	1	1	1	1	0	0	1	79
D	0	1	1	0	1		0	1	1	1	1	0	0	1	79
E	0	1	1	1	0		0	1	1	1	1	0	0	1	79
F	0	1	1	1	1		0	1	1	1	1	0	0	1	79

COMANDO'S
==========

NOISE X 0 = RESET, NOISE OFF
NOISE X 1 = CONT. LOW NOISE
NOISE X 2 = VARIABLE NOISE (SWEEP)
NOISE X 3 = CONT. HIGH NOISE
NOISE X 4 = HIGH, SHORT DECAY NOISE
NOISE X 5 = HIGH, LONG DECAY NOISE
NOISE X 6 = LOW, SHORT DECAY NOISE
NOISE X 7 = LOW, LONG DECAY NOISE

NOISE X 8 = NORMAL DAI NOISE
" " " " "
" " " " "
" " " " "
NOISE X 15 = NORMAL DAI NOISE

WILLY DE WINTER SEPTEMBER 1982

SNG REV 3.4

D7 = SUPPRESS SOUND 2
D6 = SELECT INTERN CLOCK
D5 = SELECT SOUND 2
D4 --> 1=ON NOISE
D3 -------> D3=1 SHORT NOISE
D3=0 LONG NOISE
D4 ----------> D4=1 LOW NOISE
D4=0 HIGH NOISE
D1 D0 -----------------> ENVELOPE SELECT
0/0 VCO/AM
1/0 NO MODULATION
0/1 ONE-SCHOT
1/1 VCO/PRODUCT

TABLE 2

THIS TABLE GIVES THE RESULT OF ALL THE COMBINATIONS OF VOLUME SELECTS, WITH ENVELOPE AND THE SOUND OF NOISE STATEMENTS

RESULT	ENV X	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
NOISE	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
NOISE	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
NOISE	2	0	0	0	0	1	1	1	1	1	1	1	1	2	2	2	2	2
NOISE	3	0	0	0	0	1	1	1	1	1	2	2	2	2	2	2	3	3
NOISE	4	0	0	0	1	1	1	1	2	2	2	2	3	3	3	3	4	4
NOISE	5	0	0	0	1	1	1	2	2	2	3	3	3	3	4	4	4	5
NOISE	6	0	0	0	1	1	2	2	2	3	3	3	4	4	5	5	5	6
NOISE	7	0	0	0	1	1	2	2	3	3	4	4	4	5	5	6	6	7
NOISE	8	0	0	1	1	2	2	3	3	4	4	5	5	6	6	7	7	8
NOISE	9	0	0	1	1	2	2	3	3	4	5	5	6	6	7	7	8	8	9
NOISE	10	0	0	1	1	2	3	3	4	4	5	6	6	7	8	8	9	9	10
NOISE	11	0	0	1	1	2	3	3	4	5	6	6	7	8	8	9	10	10	11
NOISE	12	0	0	1	2	2	3	4	5	5	6	7	8	8	9	10	11	11	12
NOISE	13	0	0	1	2	2	3	4	5	6	7	7	8	9	10	11	11	12	13
NOISE	14	0	0	1	2	3	4	4	5	6	7	8	9	10	11	11	12	13	14
NOISE	15	0	0	1	2	3	4	5	6	7	8	8	9	10	11	12	13	14	15

NOTE :

- Read Your handbook, chapter 'PROGRAMMABLE SOUND FACILITY' (page 89...). Attention !!!, the maximum number for volume in the ENVELOPE statement is not 15 but 16 (see table above).

- X is one of the two posible ENVELOPE numbers (0 or 1).

TABLE 3

TECHNICAL DATA

AN EXCLUSIVE RADIO SHACK SERVICE TO THE EXPERIMENTER

SN76477N COMPLEX SOUND GENERATOR

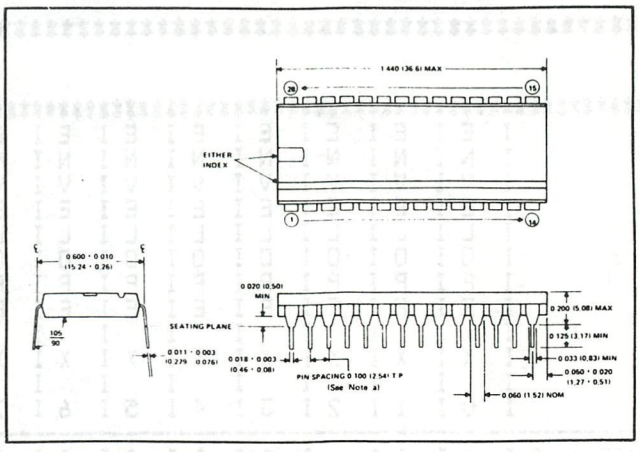
DESCRIPTION:

The SN76477N Complex Sound Generator is a linear/I^2L device which provides noise-, tone- or low-frequency - (or a combination of these) based complex sounds. Programming is via external components, (user-selected), which allows a wide variety of sounds to be created. The SN76477N is designed for ultimate flexibility in user-defined sounds, and may be used in any application requiring audio feedback to the operator (i.e. arcade/home video games, pinball games, toys, etc.; consumer oriented equipment, such as timers, alarms, controls, etc.; industrial equipment for indicators, alarms, feedback controls, etc.).

FEATURES

- Generates Noise, Tone or Low-Frequency-Based Sounds, or Combination of These
- Allows Custom Sounds to be Created Easily
- Low Power Requirements
- Allows Multiple-Sound Systems
- Compatible With Microprocessor Systems

OUTLINE DIMENSIONS



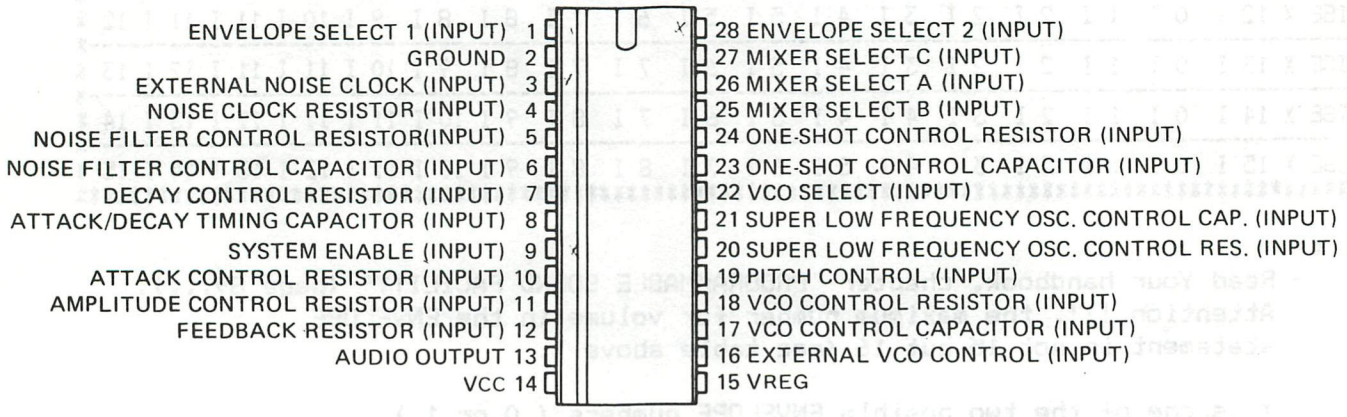
ABSOLUTE MAXIMUM RATINGS AT T_A = 25°C (UNLESS OTHERWISE SPECIFIED)

Supply Voltage, V_{REG}, Pin 15 . . . . . 6.0V
Supply Voltage, V_{CC}, Pin 14 . . . . . 12.0V
Input Voltage Applied to any Device Terminal 6.0V
Operating Temperature Range . . . . . −55°C to +120°C
Lead Temperature 1/16 Inch From Case For 10 Seconds . . . +260°C

RECOMMENDED OPERATING CONDITIONS

	MIN.	TYP	MAX	UNITS
Supply Voltage, V_{REG}, Pin 15	4.5	5.0	5.5	V
Supply Voltage, V_{CC}, Pin 14	7.5		9.0	V
Operating Free-Air Temperature	0	25	70	°C

DUAL-IN-LINE PACKAGE (TOP VIEW)

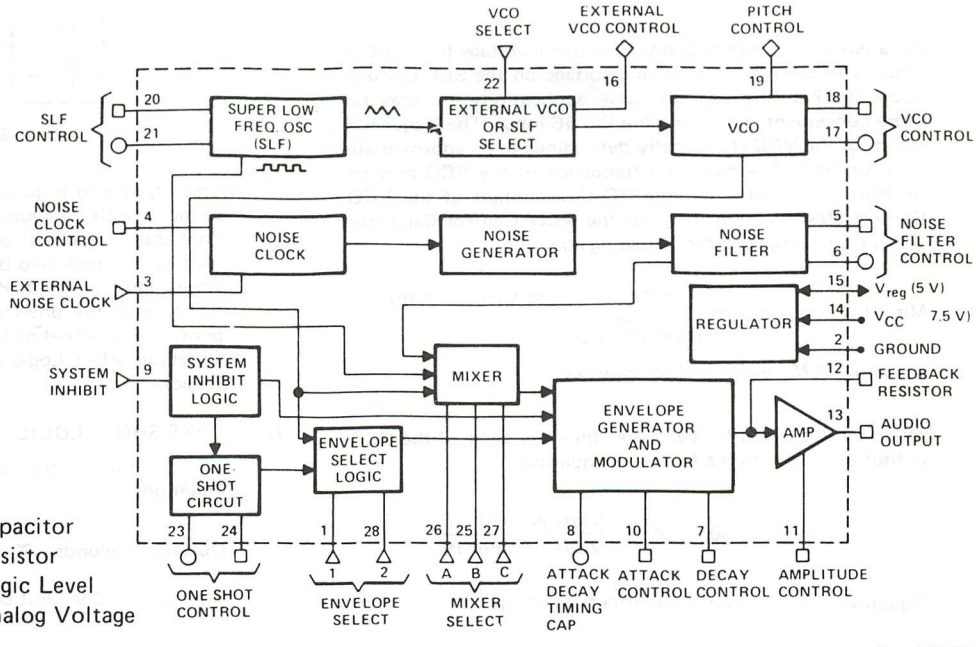


CUSTOM PACKAGED IN U.S.A. BY RADIO SHACK A DIVISION OF TANDY CORPORATION

OPERATING CHARACTERISTICS AT T_A = 25°C AND V_{REG} = 5.0V

PARAMETER	PIN	CONDITIONS	MIN	TYP	MAX	UNITS
I_{CC}	14	V_{REG} = 5.0V; NO EXT. LOAD		15	40	MA
V_{REG}	15	V_{CC} = 8.25V; I_{LOAD} = 10MA	4.5		5.5	V
INPUT REGULATION	15	I_{LOAD} = 10MA V_{CC} = 7.5V TO 9.0V		150		MV
CONTROL INPUT CURRENT RANGE			1		200	uA
NOISE CLOCK	4					
NOISE FILTER	5					
DECAY	7					
ATTACK	10					
AMPLITUDE	11					
VCO	18					
ONE SHOT	24					
LOGICAL "1" INPUT CURRENT						
ENVELOPE SELECT 1 & 2	1, 28	@ 2.0V		40	52	uA
MIXER SELECT A, B, C	25, 26, 27	@ 2.0V		40	52	uA
V_{CO} SELECT	22	@ 2.0V		40	52	uA
EXTERNAL NOISE	3	@ 2.0V		40	52	uA
SYSTEM ENABLE	9	@ 2.0V			100	uA
LOGICAL "1" INPUT VOLTAGE			2.0			V
ENVELOPE SELECT 1 & 2	1, 28					
MIXER SELECT A, B, C	25, 26, 27					
V_{CO} SELECT	22					
EXTERNAL NOISE	3					
SYSTEM ENABLE	9					
LOGICAL "0" INPUT VOLTAGE					0.8	V
ENVELOPE SELECT 1 & 2	1, 28					
MIXER SELECT A, B, C	25, 26, 27					
V_{CO} SELECT	22					
EXTERNAL NOISE	3					
SYSTEM ENABLE	9					
EXTERNAL V_{CO} CUTOFF	16		2.5			V
TRIP POINTS				2.5		V
ONE-SHOT CAP	23					
V_{CO} CAP	17					
NOISE FILTER CAP	6					
SLF CAP	21					
MAXIMUM PEAK-TO-PEAK OUTPUT VOLTAGE SWING	13	R_{LOAD} = 10K R_{FDBK} = 10K I_{11} = 200uA	2.5			V
DYNAMIC OUTPUT IMPEDANCE	13			100		OHMS

BLOCK DIAGRAM:



○Denotes Programming Via Capacitor
□Denotes Programming Via Resistor
△Denotes Programming Via Logic Level
◇Denotes Programming Via Analog Voltage

OPERATION

1. SLF (SUPER LOW FREQUENCY OSCILLATOR)

The SLF is normally operated in the range of 0.1 – 30 Hz, but will operate up to 20 kHz. The frequency is determined by the SLF control resistor (Pin 20) and capacitor (pin 21) according to the following equation:

$$\text{SLF Frequency (Hz)} \approx \frac{0.64}{R_{SLF}\,C_{SLF}} \qquad @\ V_{REG} = 5.0V$$

Equation 1: SLF Frequency Equation

The SLF feeds a 50% duty cycle square wave to the "mixer"; it also feeds a triangular wave to the "ext. VCO/SLF Select" logic, which is fed through to control the VCO when Pin 22 is high (see further explanation below).

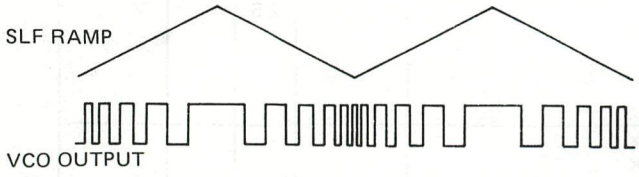
2. VCO (VOLTAGE CONTROLLED OSCILLATOR)

The VCO circuitry produces a tone output whose frequency is dependent upon the voltage at the input of the VCO. The higher Pin 16 voltage is, the lower the frequency. The controlling voltage may be either the SLF output, or it may be an externally applied signal on Pin 16. The selection of control modes (external – Pin 16; internal – SLF) is via the binary logic level on Pin 22, VCO Select, according to the following table:

Pin 22	Control Mode
0	External (Pin 16)
1	Internal (SLF)

Table 1: VCO Control Mode Selection

The input at the External VCO Control, Pin 16, may be a DC voltage, (producing a constant tone at the output of the VCO), or any waveform, producing a frequency modulated output from the VCO. A frequency modulated waveform also results when the SLF ramp controls the VCO (Pin 22 = high), as shown below:



VCO FM output

An alternate method to apply an external voltage to the VCO input is to place the controlling voltage on the SLF Control Capacitor Pin (Pin 21). In some applications this may be more convenient than using the Pin 16 input. The frequency Range of the VCO is internally determined at an approximate ratio of 10:1. The minimum frequency of the VCO may be determined by adjusting the RC time-contant of the VCO Control Resistor (Pin 18) and the VCO Control Capacitor (Pin 17), according to the following equation:

$$\text{Min VCO Freq. (Hz)} \approx \frac{0.64}{R_{VCO}\,C_{VCO}} \qquad @\ V_{REG} = 5.0V$$

Equation 2: Minimum VCO Frequency

The Pitch Control (Pin 19) varies the duty cycle of the VCO output according to the following equation:

$$\text{VCO Duty Cycle} \approx 50 \times \frac{\text{Voltage at Pin 16}}{\text{Voltage at Pin 19}} \%$$

Equation 3: Pitch Control of VCO Duty Cycle

By leaving Pin 19 high, a constant 50% duty cycle may be achieved. The specific % duty cycle, applies to constant tones produced by applying a constant DC voltage at the External VCO Control Pin (Pin 16). However, the Pitch Control may still be used to aesthetically alter the pitch of any frequency-modulated VCO output signals.

3. NOISE CLOCK

The Noise Clock clocks the Noise Generator. This circuit requires a 43K resistor to ground at Pin 4 to set an internal current level. An external noise clock may be supplied at Pin 3 to allow generation of lower frequency noise. This external clock should be a maximum 5 volt peak-to-peak square wave.

4. NOISE GENERATOR/FILTER

The Noise Generator is a binary psuedo random white noise generator whose output passes through the Noise Filter before being inputed to the mixer. The filter is a variable band width low-pass filter whose 3dB point is defined by the following equation:

$$\text{3dB Frequency (Hz)} \approx \frac{1.28}{R_{NF}\,C_{NF}} \qquad @\ V_{REG} = 5.0V$$

Equation 4: Noise Waveform 3 dB Frequency

5. The Mixer Logic selects one, (or a combination), of the inputs from the generators and feeds the output to the Envelope Generator and Modulator.

Mixer Select			Mixer Output
C (Pin 27)	B (Pin 25)	A (Pin 26)	
0	0	0	VCO
0	0	1	SLF
0	1	0	Noise
0	1	1	VCO/Noise
1	0	0	SLF/Noise
1	0	1	SLF/VCO/Noise
1	1	0	SLF/VCO
1	1	1	Inhibit

Table 2: Mixer Select Logic

6. SYSTEM ENABLE LOGIC

The System Enable Logic provides an enable/inhibit for the system output. The sound output is controlled according to the following table:

Pin 9	Output
0	Enabled
1	Inhibited

Table 3: System Enable Logic

This input also triggers the "one-shot" logic for momentary sounds, such as gunshots, bells and/or explosions. The "one-shot" logic is triggered by the negative-going edge. This may be accomplished by a momentary switch, or by a square wave input at Pin 9. Pin 9 must be held low for the entire duration of the one-shot sound (including attack and decay period). The one-shot logic is operable only when the proper Envelope Select Logic selection is made. (see Envelope Select Logic).

7. "ONE-SHOT" LOGIC

The duration of the "one-shot" is defined by the following equation:

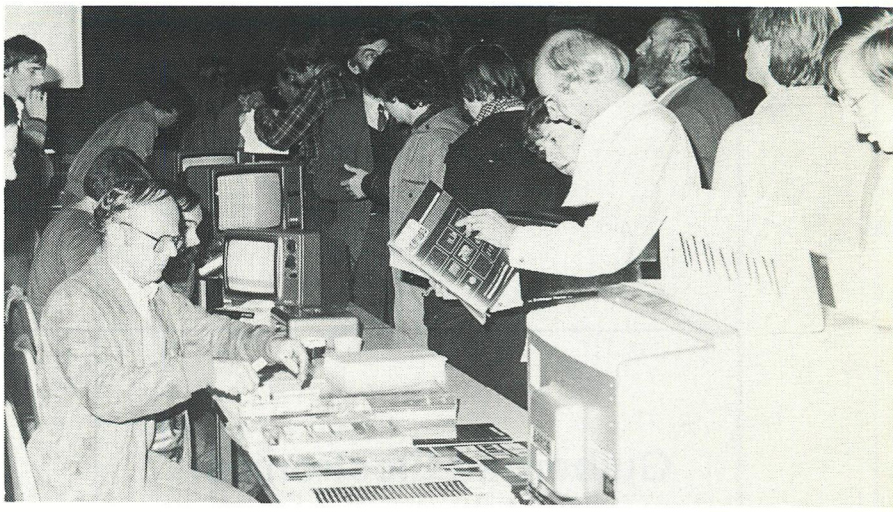
$$\text{Duration (seconds)} \approx 0.8\ R_{OS}\ C_{OS} \qquad @\ V_{REG} = 5.0V$$

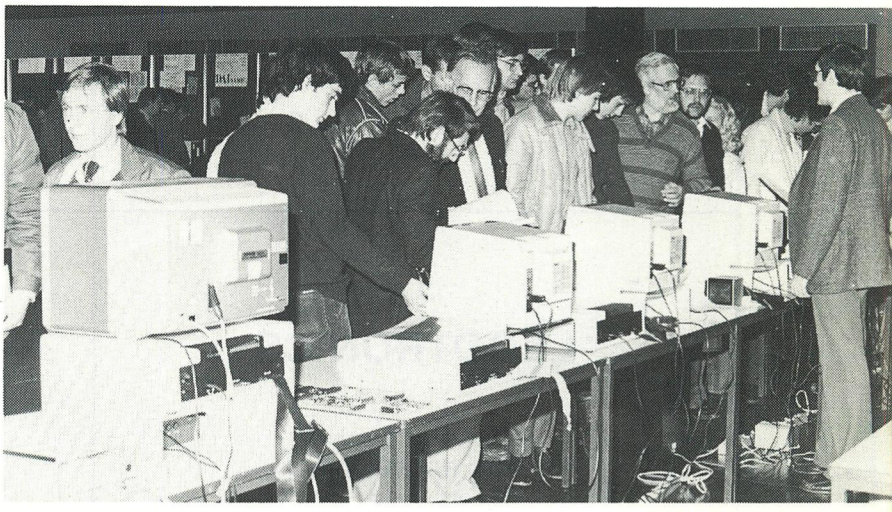
Equation 5: One-Shot Duration

cont. on page 324

SOUVENIR MIKROBEL 82

PRETTIGE FEESTEN !

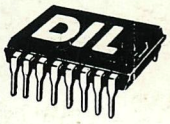




p.v.b.a. A.C.S.

Meensesteenweg 49
8800 ROESELARE
Tel. 051/21 30 89

Beenhouwersstraat 87
8000 BRUGGE
050/33 08 01



D.I.L.-ELEKTRONIKA,
MIJNSHERENLAAN 108,
3081 CH ROTTERDAM
Nederland
TELEFOON: 010 - 854213

Guibernau Electronica, s.a.

Sepulveda 104
BARCELONA-15 (SPAIN)
Tel. 243-34-32

IDS 2000

Rue de la Bonne Femme 11
4030 GRIVEGNEE
Tel. 041/41 32 20

LEGOTRONICS

Kon. Albert I laan 97
8800 ROESELARE
Tel. 051/22 01 03

MEMOCOM
Mini-digitale cassetterecorder

Postbus 2924
3000 CX ROTTERDAM - Nederland
Tel. 010-148284

MICRO SELECT

Toutes applications micro-électroniques
Vente de systèmes et composants micro-processeur

3, rue Delcloche
4020 LIÈGE
Tel. 041/41 28 10

Bennenbergweg 1
3221 NIEUWRODE (bij AARSCHOT)
Tel. 016/56 87 70
DAI - Epson Printers - Memocom digitale cassette recorder - Barco kleurenmonitor - Software - Microlectuur - Service

Publishing House J. VAN IN
att. : L. CAMPS

Educational Software
primary - secondary schools

Grote Markt 39
2500 LIER
Tel. 031/80 55 11

TEVETRONIC

Avenue Milcamps 57
1040 BRUSSEL
Tel. 02/736 61 24