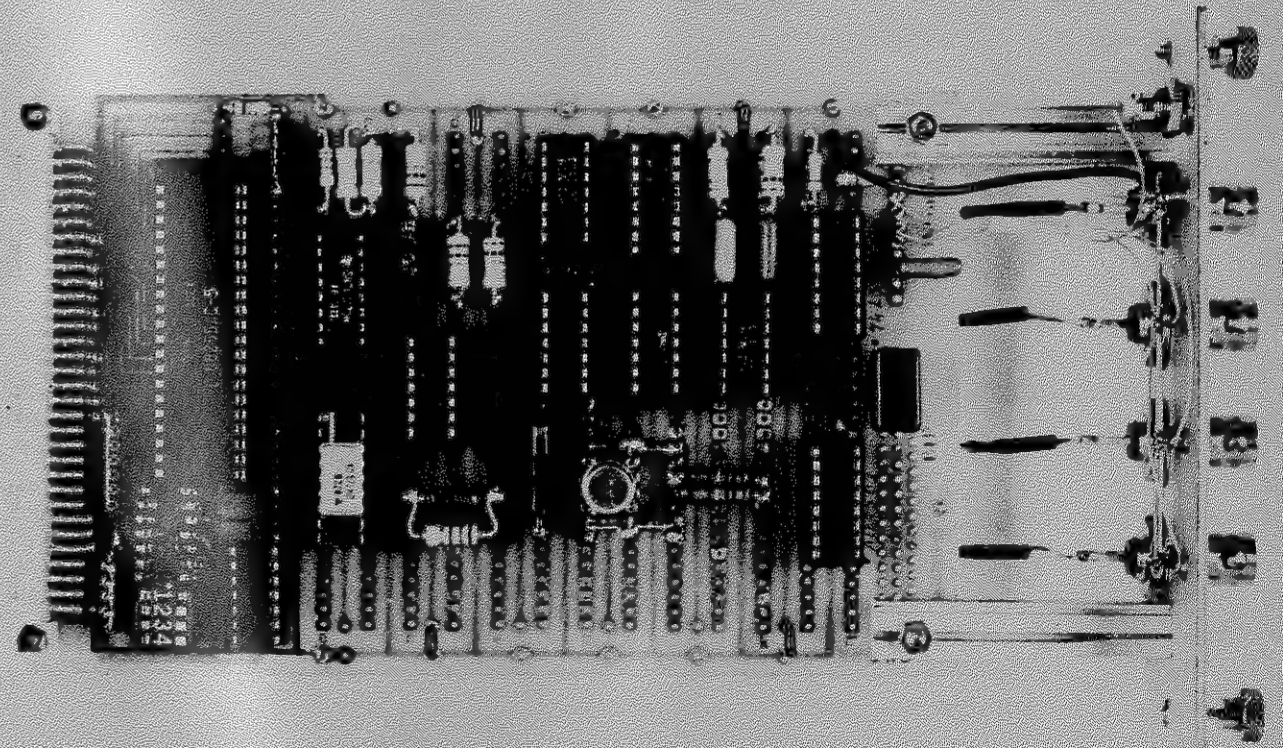


HARDWARE PROJECTS



personal computer users club

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

International

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

DAInamic redactie :

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

Vormgeving : Ludo Van Mechelen.

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

DAInamic verschijnt de pare maanden.
 Bijdragen zijn steeds welkom.

CORRESPONDENTIE ADRESSEN.

Redactie en software bibliotheek

Wilfried Hermans
 Mottaart 20 Kredietbank Herselt
 3170 Herselt nr. 401-1009701-46
 Tel. 014/54 59 74 BTW : 420.840.834

Lidgelden / Subscriptions

Bruno Van Rompaey
 Bovenbosstraat 4 Generale
 B 3044 Haasrode Bankmaatschappij
 België Leuven
 tel. : 016/46.10.85 nr. 230-0045353-74

Voor Nederland : Pour la France :

GIRO : 4083817
 t.n.v. J.F. van Dunne' DAInamic FRANCE
 Hoflaan 70 C. Dufour
 3062 JJ ROTTERDAM Rue Lavoisier 9
 Tel. : (010) 144802 59149 DUNKERQUE
 Tel. 02866 3339

Inzendingen : Games & Strategy

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



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	57344	E	3584	E	224	E	14
F	61440	F	3840	F	240	F	15

belangrijke ASCII-waarden in DAInpc

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

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

Beste leden,

Ons voorstel om in nummer 26 een aantal hardware projecten te bespreken heeft ruime weerklank gevonden. Volgende hardware- toestanden passeren de revue in deze uitgave : het eerste deel van een serie over de DCE-bus, een bespreking van de interface met de video-camera, een X-bus kaart met RAM, EPROM, klok & EPROM-programmer, 2 artikelen over D/A conversie (de ene met wat meer bits dan de andere), 2 maal de VC-1541 aan de DAI, notities over ROM & I/O, en nogmaals 80 character display.

DAInamic - MSX, wat is er aan de hand ?

Misschien eerst even een toelichting over het begrip "MSX": MSX is geen computermerk, wel een computerstandaard. Het bekende Amerikaanse softwarehuis MICROSOFT heeft de hard-en software specificaties vastgelegd en reeds meer dan 40 (vooral Japanse) fabrikanten produceren apparaten volgens deze normen. Dit betekent dat alle software en hardware tussen deze verschillende machines volledig uitwisselbaar is. De specificaties zijn niet revolutionair, eerder klassiek, wat beslist zijn voordelen heeft ivm support en verkrijgbaarheid van onderdelen. Globaal bekeken klinkt het als volgt : 280 3.58 Mhz, 32K ROM extended BASIC, minimaal 16K RAM, (meestal 64K), 16K aparte RAM voor video, sound chip AY 8913, 32 sprites, resolutie 256 x 192 in 16 kleuren. U merkt dat onze DAI (en ook andere micro's) in verschillende opzichten beter presteert, de kracht van het concept zit echter in de uitwisselbaarheid van software en periferie. Mede gezien de prijs van deze apparaten geloven wij dat MSX uiteindelijk de beslissende stap gaat maken voor de computer in de huiskamer en de scholen. De redactie heeft unaniem beslist deze ontwikkelingen te volgen en heeft de afdeling MSX-club opgericht. Dit betekent beslist niet het einde van onze DAI-activiteiten. Willy Coremans, de auteur van DBASIC, heeft er al voor gezorgd dat wij MSX-programma's kunnen inlezen in onze DAI en omgekeerd. Verdere conversie is ter studie, mogelijk komt er een extensie-file voor DBASIC die het mogelijk maakt om MSX-programma's te laten lopen op de DAI, dit opent voor alle DAI-gebruikers hoogst interessante perspectieven! Deze uitbreiding van de activiteiten betekent wel dat we onze beschikbare tijd en energie wat moeten verdelen. Alle voorstellen tot versterking zijn dan ook welkom, er zijn trouwens al contacten met onze Waalse DAI-zusterverenigingen om tot gezamenlijke initiatieven te komen. In april verschijnt onze eerste MSX-uitgave, geïnteresseerde leden kunnen dan een gratis nummer aanvragen, onze DAInamic uitgave loopt normaal door.

Dear members,

The long story above tells about our new MSX-activities. Maybe not so important for non-Dutch speaking members, because all our MSX-publications will be only in Dutch language (for the moment). If you are interested in MSX-standaard, please contact us, we will send you our first MSX-magazine in april.

New soft & hardware:

SUPERBASE : audio : 1600 Bfr DCR : 1750 Bfr
 ATARI - joystick interface : 1350 Bfr - 196 FF/169 FF
 (please allow 3 or 4 weeks for delivery, the interfaces are produced and mailed by DAInamic France.
 MEMODAI : audio : 1000 Bfr DCR : 1150 Bfr

We hope to see you on our next meeting on 30th March 1985 !

W.Hermans

3	REMARK	Redaction
4	BLADWIJZER - CONTENTS	
5	5th DAInamic MEETING	
6	Programmeertechnieken	F.Druijff
10	Contents DAInamic 1984	J.Boerrigter
13	DCE-bus part 1	A.Beuckelaers
22	Digitizing with DAI	N.Looije
24	GETC du DAI	OCTET/Doumont
27	X-bus Card	F.Choppinet
29	TIPS	div/J.Streep
30	SUPERBASE	U.Wienkop/F.De Raedt/ S.Van Hoecke
34	10 bits A/D convertor	F.Uytterhoeven
45	Interface joystick	C.Dufour
46	A/D D/A convertor	K.H.Kopp
49	DAI ROM & I/O	R.Benedik
53	DAI DOS 1541	J.Boerrigter/H.Kop H.Rison/P.Wiegers
55	KEN-DOS story	G.Wassermann
57	VC-1451 for DAI	H.Tegethoff/LMPG
61	MEMODAI	DAInamic France
63	80 characters display	M.Hauser

DAInamic subscription rates :

Benelux : 1000 Bfr
 Europe : 1100 Bfr
 Outside Europe 1500 Bfr
 (Air Mail)

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

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

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 vervaardigd of openbaar gemaakt door middel van druk, fotocoop, microfilm of op welke andere wijze ook zonder voorafgaande schriftelijke toestemming van de uitgever.



30 MARCH 85

TONGELSBOS
 WESTERLO

10.00 - 18.00
 WELCOME !

PROGRAMMEERTECHNIEKEN

Zoals beloofd zal ik deze keer doorgaan met het bespreken van de structuren, die in BASIC voorkomen. Veel van de gemaakte opmerkingen over het wezen van de structuren zullen ook in ruimer verband, d.w.z. bij andere programmeertalen en / of in het dagelijks leven van toepassing kunnen zijn.

Ik wil beginnen in normaal DAI-BASIC en de FOR-NEXT nogeens naar voren halen. Vorig maal beweerde ik tot tweemaal toe dat er geen verschil is (beter gezegd dat ik geen verschil heb kunnen vinden) tussen NEXT en NEXT I. Ik hoopte met deze opmerking sommige mensen te prikkelen voor mij een verschil te vinden, maar helaas, dat heeft niet zo mogen zijn. Wel kreeg ik een aantal reacties van mensen die een schijnbaar verschil lieten zien. Dit verschil was mij wel bekend maar werd niet bedoeld. Nogmaals ik heb totaal geen verschil kunnen ontdekken tussen de volgende twee mogelijkheden:

```
<1> 10 FOR I=0 TO 10:...(body).....:NEXT
en
<2> 10 FOR I=0 TO 10:...(body).....:NEXT I
```

Ook heb ik geen verschil kunnen vinden in de verwerkingssnelheid van de ene of de andere mogelijkheid. Nu wil ik het verschil bij een bepaalde toepassing bespreken. U begrijpt dat na alle opmerkingen over 'geen' verschil dit een niet normale toepassing betreft. Het verschil is natuurlijk dat de DAI als alleen 'NEXT' wordt gebruikt automatisch aanneemt dat hier de NEXT bedoeld wordt, die hoort bij de laatst gebruikte FOR. Maakt men er echter een gewoonte van uit de FOR-NEXT te springen kan de kale NEXT die dan aangetroffen wordt best wel eens bij een andere FOR horen dan bedoeld werd. We kunnen hier juist bewust gebruik van maken door een NEXT bij twee (of meer) FOR's te laten horen. Vaker zie ik echter het gebruik van meerdere NEXT'n bij een FOR. Wat wil de programmeur die dit doet bereiken? Meestal niets bijzonders, hij wil gewoon dat het programma werkt en vond daar een oplossing voor. Later als hij aan dit programma nog een keer iets wil veranderen zal hij leren inzien dat hij beter en andere oplossingen had kunnen zoeken. Het overzicht verliest men al gauw bij zo'n gekunstelde structuur. Ik geef een voorbeeld:

```
1200 FOR I=0 TO 20:IF ARRAY(I)>=0 THEN NEXT
1210 ARRAY(I)=0-ARRAY(I):NEXT
1220 .....
```

De negatieve array elementen worden hiermee positief gemaakt. Heeft men geluk is het laatste element negatief en komen er geen brokken. Is het laatste element echter positief zal de loop beëindigd worden in regel 1200; het programma gaat dan door met regel 1210 en zal melden: 'NEXT WITHOUT FOR IN LINE 1210'. Een en ander valt te voorkomen door regel 1200 aan het eind te voorzien van GOTO 1220. En zeg nu alstublieft niet dat we de gewenste verandering ook kunnen krijgen door ARRAY(I)=ARRAY(I)*SGN(ARRAY(I)) of ARRAY(I)=ABS(ARRAY(I)) want dat weet ik ook wel. Het gaat hier om de structuur en niet om wat er in de body gebeurt. Voor een juist overzicht (dus een NEXT bij en FOR) is het dus nodig de volgende oplossing te nemen.

```
1200 FOR I=0 TO 20:IF ARRAY(I)<0 THEN ARRAY(I)=0-ARRAY(I)
1210 NEXT
```

Achter de NEXT van regel 1210 kunnen best andere instructies komen. Is echter in beide gevallen een actie nodig dan zullen we, bij ontstentenis van een 'IF THEN ELSE' bij DAI-BASIC, moeten kiezen tussen een dubbele vraag of een GOTO.

1200	FOR I=0 TO 20		1200	FOR I=0 TO 20
1210	IF A<=B THEN body 1	of	1210	IF A<=B THEN body 1:GOTO 1230
1220	IF A>B THEN body 2		1220	body 2
1230	NEXT		1230	NEXT

Er is echter nog een reden denkbaar waarom iemand zo nodig een FOR bij meer NEXT'n en meer FOR's bij een NEXT laat horen. Absolute snelheid en / of compactheid van een programma. Deze belangen zijn bijna altijd strijdig met overzichtelijkheid en snelheid van programmeren (dat is iets heel anders dan de snelheid van het programma). Ik geef U een voorbeeld om de priemgetallen tot 1000 mee uit te rekenen.

```
10 GOSUB 40:FOR G=0 TO 14:ON PEEK(G+#32F0) GOTO 20:
FOR I=(G+G+6)*G+#32F3 TO #34E1 STEP G+G+3:POKE I,1:NEXT
20 NEXT:POKE #1C0,#FF:FOR I=#32F0 TO #34E1:ON PEEK(I) GOTO 20:
PRINT I+I-#65DD;:POKE #7B,0:NEXT
30 B=PEEK(#1BE):A=PEEK(#1BF):PRINT :PRINT (#FFFF-B-A*256)/50.0:
POKE #5F,#C4:END
40 WAIT TIME 1:POKE #1BF,#FF:POKE #1BE,#FF:POKE #5F,#B4:
CURSOR 1,22:POKE #BF61,#32:POKE #131,1:RETURN
```

In dit programma wordt de eerste NEXT in regel 20 gebruikt voor de eerste FOR uit regel 10 en ook door de FOR uit regel 20. Deze FOR uit regel 20 gebruikt dus zowel de eerste NEXT als de laatste NEXT van regel 20. De snelheid is de beweegreden geweest om zo'n gekunstelde (of is kunstige toepasselijker?) aanpak te kiezen.

Maar zoals reeds eerder vermeld door gebruik van NEXT I ipv NEXT kan men gevaarlozer (NIET gevaarloos!) uit een binnenloop springen.

```
100 CLEAR 1000:DIM A(20):MODE 2
110 FOR I=0 TO 20:A(I)=RND(100.0):NEXT
120 FOR X=0 TO 20
130 FOR Y=0 TO YMAX
140 IF A(X)>Y THEN DOT X,Y 23:NEXT Y
150 NEXT X
```

Ziet U overigens hoe ik uit de binnenloop met Y spring zonder een GOTO? De Y achteraan regel 140 is hier niet nodig maar de X in regel 150 wel.

Nu ga ik weer aandacht besteden aan de structuren uit D-BASIC. Eerst de REPEAT UNTIL-loop waarvan ik het vorige artikel al een paar voorbeelden heb gegeven. Maar weest gewaarschuwd: Pas als je er zelf mee gewerkt hebt leer je het te waarderen. Je moet jezelf er in het begin toe dwingen er mee te werken. Veel van de nieuwe mogelijkheden zijn ook in normaal DAI-BASIC te programmeren en je zoekt haast automatisch binnen dat kader naar een oplossing. Je maakt dan ook nogeens beginnersfouten. Mij overkwam het toen ik trachtte de REPEAT-UNTIL te nesten. Dit nesten d.w.z. een REPEAT-UNTIL-loop in een andere REPEAT-UNTIL-loop (evt een andere loop) wilde ik gewoon proberen. Ik tikte het volgende in de EDIT-mode in:

```
10 REPEAT
20 REPEAT H=BETC UNTIL H>0
30 PRINT H
40 UNTIL H>120
```

Dit leverde echter een foutmelding op. Na enig proberen vond ik uit dat het wel werkte met regel 10 REPEAT A=A. Dus in feite een NOP (No Operation) na de REPEAT. Nog eens goed de handleiding doornemen en nadenken en ik zag de gemaakte fout: Vergelijk de REPEAT uit de REPEAT-UNTIL-loop met de FOR uit de FOR-NEXT-loop, daar kan ook niet de FOR op de ene regel en de start I=0 en TO N op de volgende. Voor de FOR-NEXT geldt dat FOR start TO eind en eventueel STEP stapgrootte op dezelfde regel moeten staan; alleen de NEXT mag, maar dat hoeft niet, een aantal regels later komen. Voor de REPEAT-UNTIL geldt iets soortgelijks; na de REPEAT moet op dezelfde regel minimaal een instructie gegeven worden die gerepeat dient te worden. De oplossing met de A=A instructie die loos is, werkt dus. We kunnen echter ook de volgende REPEAT (uit regel 20) achter de REPEAT van regel 10 plaatsen. Pas echter op; er hoort geen dubbele punt tussen REPEAT en de volgende instructie. Het programmaatje wordt dus:

```
10 REPEAT REPEAT H=GETC UNTIL H>0 :PRINT H:UNTIL H>120
```

Kijk de regel maar goed na op de dubbele punten die er soms wel en soms niet tussen moeten worden geplaatst. Het is allemaal logisch maar wel even wennen. Nu zocht ik een kleine routine, die de REPEAT-UNTIL zinnig nestte in zichzelf. Ik vond deze in de controle op numerieke invoer. Zoals iedereen wel zal weten komt het vaak voor dat we alleen cijferwaarden als invoer willen krijgen. Dit is vanzelfsprekend best op te lossen met een paar IF-statements maar de volgende oplossing is stukken sneller en ikzelf vind hem veel fraaier.

```
10 REM CHECK ON NUMERIC INPUT / F.H. DRUIJFF - 1/85
20 REPEAT REPEAT H=GETC:UNTIL H>47:UNTIL H<58:PRINT CHR$(H),
30 GOTO 20
```

De werking is alleen te controleren door bezitters van D-BASIC, dus maar snel aanschaffen. De eindeloze loop zoals hier gemaakt met GOTO 20 in regel 30 kan ook met REPEAT ...body... UNTIL A<>A gemaakt worden, daar hier wel nooit aan de voorwaarde voldaan zal worden. De loze A=A instructie is toch niet geheel zonder betekenis; als we namelijk meerdere REPEAT-UNTIL-loops willen nesten kunnen we niet beginnen met REPEAT REPEAT REPEAT instructie als we zoveel REPEATS willen dat die niet meer op de regel passen. In dat geval kan de loze A=A instructie uitkomst bieden.

Ik wilde nog een fraai voorbeeld geven voor de geneste REPEAT-UNTIL. Na de bovenstaande controle op numerieke invoer ligt een controle op cursor invoer of op lettertekens voor de hand. Deze zijn echter simpel zelf te maken en ik laat het aan de lezer over. De controle op cursorinvoer zal ik wel geven maar dan met gebruikmaking van een geneste WHILE-WEND-loop.

Er is een nog onbesproken gebleven verschil aan te geven tussen enerzijds de FOR-NEXT-loop en anderzijds de REPEAT-UNTIL-loop. Dit verschil is de gesloten structuur van de REPEAT-UNTIL-loop en de open structuur van de FOR-NEXT-loop. Aan het begin van dit artikel liet ik zien dat het goed mogelijk is een NEXT voor meerdere FOR's te gebruiken of omgekeerd meerdere NEXT'n bij een FOR te laten horen. De structuur heet dan open of vrij. Is het zo dat begin en eind van de loop altijd bij elkaar horen spreken we van een gesloten of gebonden structuur. Dit laatste is het geval bij de REPEAT-UNTIL-loop. Als ik helemaal eerlijk ben zal ik moeten toegeven dat ik alleen weet dat de compilslag bij D-BASIC controleert of de REPEAT's en de UNTIL's goed bij elkaar passen. Dus dezelfde controle als voor haakjes bij expressies; REPEAT is dan haakje openen en de UNTIL is haakje sluiten. Ik heb niet gecontroleerd of het mogelijk is te beginnen met een REPEAT weer een REPEAT en dan de twee UNTIL-voorwaarden om dan in het programma na de tweede REPEAT te springen naar de tweede UNTIL (hij

hoort logischerwijs natuurlijk bij de eerste UNTIL, vergelijk met FOR-NEXT I) hierna een sprong naar de eerste UNTIL, die wordt gevolgd door een sprong over de tweede UNTIL. Ik denk dat D-BASIC het accepteert en waarom ook niet als iemand het zichzelf moeilijk wil maken mag dat best van mij.

Nu de beloofde WHILE-WEND voor controle op cursorinvoer.

```
10 CHECK ON CURSOR INPUT / F.H. DRUIJFF - 1/85
20 WHILE H<500 DO WHILE H<16 DO H=GETC *** zeven onderdelen en geen :
30 WEND
40 WHILE H<20 DO ON H-15 GOSUB 100,200,300,400
50 H=400
60 WEND
70 H=0
80 WEND
100 PRINT "naar boven" Hier in te vullen
110 RETURN
200 PRINT "naar beneden" de actie die voor
210 RETURN
300 PRINT "naar links" uw speciale probleem
310 RETURN
400 PRINT "naar rechts" van toepassing is
410 RETURN
```

De WEND van regel 80 hoort bij de eerste WHILE van regel 20 en die twee samen zorgen ervoor dat het een eindeloze loop wordt. De tweede WHILE uit regel 20 zorgt er samen met de WEND van regel 30 voor dat de H=GETC blijft worden herhaald totdat er een toets met minstens ASCII-code 16 wordt ingedrukt. De WHILE uit regel 40 zorgt samen met de WEND uit regel 60 dat alleen gereageerd wordt als er een cursortoets werd ingedrukt. Zonder de H=400 uit regel 50 blijft aan de voorwaarde voldaan worden en zitten we dus in een eindeloze loop. Zonder de H=0 uit regel 70 is loop nummer twee echter eindeloos. We zien dus dat dit wel een werkend programma is, maar om het nu goed werkend te noemen...NEE. De constructie WHILE-WEND is hier echt niet op de beste manier gebruikt.

De WHILE-WEND-loop is ook een gesloten structuur dus bij elke WHILE hoort slechts een WEND en omgekeerd. Eens bereikte mij de vraag van iemand waarom de DAI niet bij aanvang van een programma controleert of het aantal GOSUB's overeenstemt met het aantal RETURN's. Tweede mogelijkheid: gegeven het feit dat de DAI het niet doet kan DAI'nic dan niet zorgen voor een programma dat dit controleert. Onze arme DAI'er kreeg namelijk keer op keer de foutmelding 'RETURN WITHOUT GOSUB IN LINE' of STACK OVERFLOW en beide waren te wijten aan het niet overeenstemmende aantal GOSUB's en RETURN's. In mijn antwoord heb ik toen uitgelegd dat alleen programma's die om structurele redenen in subroutines zijn opgebouwd het aantal GOSUB's en RETURN's zullen overeenstemmen. Maar meestal zal een subroutine worden gebruikt om bepaalde programmadelen, die meerdere keren nodig zijn, te vervangen door een subroutine die dan op de gewenste plaatsen in het programma kunnen worden aangeroepen met een GOSUB. De programmeur, die zijn programma's overzichtelijk wenst te houden, zal normaal op maar een enkele plaats uit een subroutine springen: achteraan! Het is echter mogelijk een subroutine op vrijwel elke plaats te verlaten. Doe dit niet, als er eenmaal genoeg programmeerervaring is zult U dat wel inzien. Het begin van de subroutine moet liefst ook aangegeven worden. Er zijn bij DAI hiervoor twee mogelijkheden. Ten eerste de regelnummering: Ik laat zelf de hoofdsroutines op een duizendtal beginnen en de ondersroutines op honderdtallen. Is absolute snelheid gewenst kan ik van deze regel afwijken. De tweede methode is nog simpeler; eenvoudig beginnen met een REM met de nam/werking van de betreffende subroutine.

Frank H. Druijff

CODE	PAGE	SUBJECT	AUTHOR
calc	-----	-----	-----
calc	84/305	Calculus	Lemoine
calc	84/124	Construction calculations	Vingerling
calc	84/83	Conversion fractions - decimal/floating point	Herrmann
calc	84/367	DAIminiCALC	Wanet
calc	84/303	Fast multiply/division in m.l.p.	Electronics
calc	84/308	Graphics on 1/4 circle	Berkx
calc	84/26	Interest calculation	Collet
calc	84/135	Math. game	-
calc	84/214	Math. game - solution	-
disk	-----	-----	-----
disk	84/385	Dataflex: bugs and errata	Couwberghs
firmw	-----	-----	-----
firmw	84/260	BASIC: SAVEA/LOADA	Vandebergh
firmw	84/258	Contents 74S288 EPROM's	Meystre
firmw	84/257	Corrections firmware manual - 3	Boerrigter
firmw	84/179	DCE-bus interfacing	Broekman
firmw	84/409	DIM statement	Boerrigter
firmw	84/331	Memory map mode 1/2	Hermans
firmw	84/334	Print routines	Boerrigter
game	-----	-----	-----
game	84/332	Look-out	Gios
game	84/301	Numbers up	Maertens
game	84/399	Numbers up - errata	Maertens
game	84/424	Road finding	Schall
graph	-----	-----	-----
graph	84/386	Christmas wishes	Overvoorde
graph	84/422	Creating 16 colours in mode 0	Pedelaborde
graph	84/416	Creating new colours	Dufour
graph	84/75	Cycloids	Druijff
graph	84/364	DAI character set	Dufour
graph	84/378	Fireworks	Marcel
graph	84/294	Graph. demo	Overvoorde
graph	84/128	Graphic demo - 1	Overvoorde
graph	84/387	Graphic fractals	Berkx
graph	84/431	Grids for 16x16 FBT/SFBT designs	Hermans
graph	84/150	Logo DAI	Rossion
graph	84/392	Logo N.O.S.	Overvoorde
graph	84/120	Memory map mode 5/6	Hermans
graph	84/75	NATO stars	Druijff
graph	84/110	Rectangle snakes basic/mip/Pascal	Sigg
graph	84/130	Screen lay-out demo	Beyens
graph	84/309	Shapes: fast drawing in BASIC	De Boeck
graph	84/129	Sundown	Mikulic
graph	84/263	Telephone	Schall
graph	84/399	Windows	Schall
hardw	-----	-----	-----
hardw	84/22	Video amplifier	de Bauw
libr	-----	-----	-----
libr	84/398	Bookkeeping program	Chabot
libr	84/203	Catalog 1.5.84	-
libr	84/386	Catalog 12/84	-
libr	84/40	Catalog 2/84	-
libr	84/237	Catalog DAInamic/France	-
libr	84/397	DBASIC V2.2	Coremans
libr	84/200	DOS toolkit	Couwberghs
libr	84/270	Eagle	Gortz
libr	84/265	Education 6-7-8	DiDAIsoft
libr	84/266	Frogger	Roger
libr	84/282	Games collections 13,14	Druijff
libr	84/201	MIX 1 - Quest - Math'fun 2	Druijff/Hermans
libr	84/268	Phoenix	Janin
libr	84/397	Phys. Geography and climatology	Antrop
libr	84/11	Puzzly/Daylaxians/Duel/CLIQ	Dialogue-informatique

CONTENTS 1984

CODE	PAGE	SUBJECT	AUTHOR
libr	84/271	Software gallery: pictures	-
libr	84/269	Tangram	Bellekens
libr	84/267	Toolkit 6	div.
libr	84/270	Turtle Basic	Costa
pasc	-----	-----	-----
periph	-----	-----	-----
periph	84/41	DAI-SWISS-DOS	Meystre
periph	84/377	DCR function indication	Strobel/Marchand
periph	84/379	EPROM pack	Borst & Co
periph	84/135	Eprom programmer: errata p.83/386	Beuckelaers
periph	84/76	INDATA floppies - test report	Dupagne
periph	84/84	KEN-DOS test report	Couwberghs
periph	84/251	KEN-DOS: commands (fr)	Vandermeersch
periph	84/191	KEN-DOS: test report	Wassermann
progr	-----	-----	-----
progr	84/257	Arrays bigger than 255 subscripts	Boerrigter
progr	84/195	Cursor keys: Using routines	Looije
progr	84/24	Deleting parts of programs 2	Menier
progr	84/25	GETC: Short & fast routine	Looije
progr	84/23	IF-THEN-ELSE 2	Prop
progr	84/60	Logic operators	Druijff
progr	84/346	Loop structures - DBASIC	Druijff
progr	84/146	Mode 0 screen programming	Druijff
progr	84/209	Mode 0 screen programming - 2	Druijff
progr	84/73	My first machine language program	Hooijkaas
progr	84/277	Print routines	Druijff
progr	84/27	Printer on/off - 2	Druijff
read	-----	-----	-----
read	84/378	New DAI's	Micro Magazine
read	84/406	SPL: Description	LIST - Mariatte
read	84/6	Structured programming with DAI Basic	Schout
read	84/410	Tissue design	Micro-ordinateurs
rem	-----	-----	-----
rem	84/105	BASIM: simulation program	Offereins/Meerman
rem	84/29	Contents DAInamic 1980-1983	Boerrigter
rem	84/231	Info French DAI users club	Dufour
rem	84/250	Meeting DAI 20.10.84 - Nivelles	Dulvins/Poels
rem	84/312	The ultimate video game (RAF use of DAI)	Computer systems June/83
rem	84/134	Vaserelli	Sip
sound	-----	-----	-----
sound	84/387	Sound fractals	Berkx
sound	84/256	Tuning flute/guitar	Zahner
teach	-----	-----	-----
teach	84/189	Course DCE-microcomputer - timers	Dirksen
teach	84/170	Course microprocessors 3.3	Beuckelaers
teach	84/225	Course microprocessors 4	Beuckelaers
teach	84/283	Course microprocessors 5	Beuckelaers
teach	84/352	Course microprocessors 6	Beuckelaers
tool	-----	-----	-----
tool	84/400	80-columns text	Doornebal
tool	84/338	8080 Assembly tables	Couwberghs
tool	84/213	Array - edit	Looije
tool	84/88	BASIC variables in m.l.programs	Esveld
tool	84/114	Basicode 2 - Germany	WDR Computer club
tool	84/28	Baudrate in Utility	Boerrigter
tool	84/296	Crypto-writer	Verberckmoes
tool	84/215	DAI as terminal for mainframes	Maes
tool	84/405	Epson MX-80/2: Screen tabulator	Zahner
tool	84/91	Erase till end of screen	Esveld
tool	84/293	External interrupt: how to use	Boerrigter
tool	84/389	Heap organisation	Chabot
tool	84/100	Interrupt service routine in BASIC	Biekart
tool	84/420	LISTings with CHR\$() inserted	Vandermeersch
tool	84/188	Lower case characters in INPUT statement	Boerrigter

CODE	PAGE	SUBJECT	AUTHOR
tool	84/98	Print using	Mariatte
tool	84/414	Program protection	Guerard
tool	84/402	SAVEA/LOADA: fast routines	Poels
tool	84/92	Text in data generator	Boerrigter
tool	84/419	Trics: Merging - Deleting program parts	Vanderweersch
tool	84/333	Upper-to-lower case: demo	de Bont
tool	84/151	Write and Read in Utility	Kietzmann

transl	84/393	BASIC monitor - 1 (fr)	Boerrigter
transl	84/177	Cassette listing (Fr)	Assink
transl	84/319	Cassette routines SDK-B5 (eng)	v.Oel
transl	84/162	DAI video hardware	Hospers
transl	84/17	Defectuous power supply (Eng)	Verberkt
transl	84/14	FBT-DISK-PEEK-POKE 2 (Eng)	Couwenberghs
transl	84/320	INDATA news 83/249 (eng)	-
transl	84/64	KEN-DOS (Fr)	Dufour
transl	84/163	Negative cursor	de Bont
transl	84/15	Paint (Eng)	Druijff
transl	84/164	Power supply and overvoltage protection (Eng)	Corswandt
transl	84/153	Program speed (Eng)	Druijff
transl	84/19	Programming in machine language (Eng)	Druijff
transl	84/12	Programming techniques (Eng)	Druijff
transl	84/231	Programming techniques 83/154 (fr)	Druijff
transl	84/316	Programming techniques 83/224 (eng)	Druijff
transl	84/233	Programming techniques 83/285 (fr)	Druijff
transl	84/395	Programming techniques 83/301 (fr)	Druijff
transl	84/156	Rotating bungalow (Eng)	Vingerling
transl	84/236	Simulation GOTO X/RUN X 83/208 (fr)	v.Dunne
transl	84/159	Sorting demo (Eng)	Maertens
transl	84/153	Timing problems (Eng)	v.Lieshout
transl	84/320	Using AZERTY keyboard 83/250	Schepens
transl	84/315	Vidextext in Belgium (eng)	-

use			
use	84/7	D-Basic	Coremans
use	84/141	D-Basic	Coremans
use	84/242	D-Basic - 2	Coremans
use	84/323	D-Basic - 3	Coremans
use	84/5	DAIPacman	Meyer
use	84/427	DBASIC - 4	Coremans
use	84/396	DNA - FWP conversion	Menier
use	84/94	FWP - User hints	Druijff
use	84/388	FWP - patches 2	Gruiters
use	84/239	FWP: tips - 1	Gruiters
use	84/302	FWP: tips 2	-
use	84/8	Fast Word Processor	Gruiters

ut			
ut	84/350	Bank transfers	Beyens
ut	84/96	How to put furniture in a room	Druijff
ut	84/321	Info stargazers	Poels

viewd			

DCE-BUS

In een korte artikelenreeks wensen we de verschillende aspecten van de D.C.E. bus toe te lichten.

- De "parallel peripheral interface" bouwsteen (8255 van Intel) waarmee de D.C.E. bus op hardware gebied is uitgerust.
- Het gebruik van deze bouwsteen als "general purpose interface"
- Het gebruik als eigenlijke D.C.E. bus zoals voorzien in de DAI PC.

1. DE PARALLELE INTERFACE BOUWSTEEN 8255.

Het betreft een programmeerbare bouwsteen die 24 I/O bits ter beschikking stelt van de gebruiker. Deze bits kunnen gegroepeerd worden in 2x8 bits en in 2x4 bits, aangegeven onder de benamingen Kanaal A, kanaal B, meest beduidende bits van kanaal C en tenslotte de minst beduidende bits van kanaal C. Deze kanalen kunnen in verschillende modi geprogrammeerd worden.

1.1 Blokschema van de 8255.

Zoals bij elke programmeerbare bouwsteen bestaat het blokschema uit twee delen. Het gedeelte links van de inwendige databus dat interfacing met de microprocessor toelaat en het gedeelte rechts van deze bus dat verbinding met de buitenwereld toelaat.

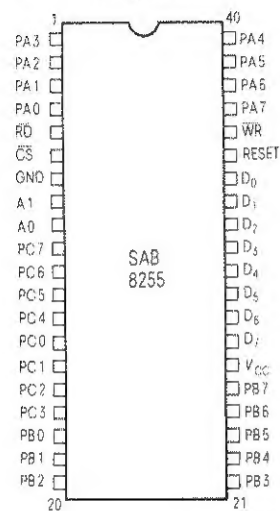
- Het interfacegedeelte met de microprocessor bestaat zelf uit twee delen. Een databusbuffer met tri-state logica die dataverkeer tussen microprocessor en de periferiebouwsteen toelaat. Ter herinnering tri-state logica betekent dat naast de twee logische toestanden 0 en 1 er een derde toestand bestaat waarbij de datalijnen in hoge impedantie staan en a.h.w. geïsoleerd worden van de eigenlijke databus. Opheffen van deze toestand gebeurt door het aanleggen van een selectiesignaal (CS chipselect) aan een van de klemmen van de besturingslogica.

- Het tweede gedeelte van de interfacelogica met de centrale verwerkingseenheid is de schrijf/lees en besturingslogica. Deze logica is verantwoordelijk voor de besturing van de bouwsteen (selectie, schrijftoestand, leestoestand, programmatie.)

Selectie van de bouwsteen wordt verkregen door het aanleggen van een laag niveau aan de CS (chip select) ingang. Het bepalen van de lees of schrijf richting wordt verkregen met de signalen RD(read) en WR(write) aan de overeenkomstige klemmen.

De adresseringsbits A0 en A1 worden gebruikt om de verschillende kanalen of het programmatieregister binnen de bouwsteen te selekteren.

Pin configuration



Dimensions page 398

Pin names

D ₀ -D ₇	Data bus (bi-directional)
RESET	Reset input
\overline{CS}	Chip select
\overline{RD}	Read input
\overline{WR}	Write input
A ₀ , A ₁	Port address
PA ₀ -PA ₇	Port A (Bit 0-7)
PB ₀ -PB ₇	Port B (Bit 0-7)
PC ₀ -PC ₇	Port C (Bit 0-7)
V _{cc}	Power (+5 V)
GND	Ground (0 V)

fig 2

Block diagram

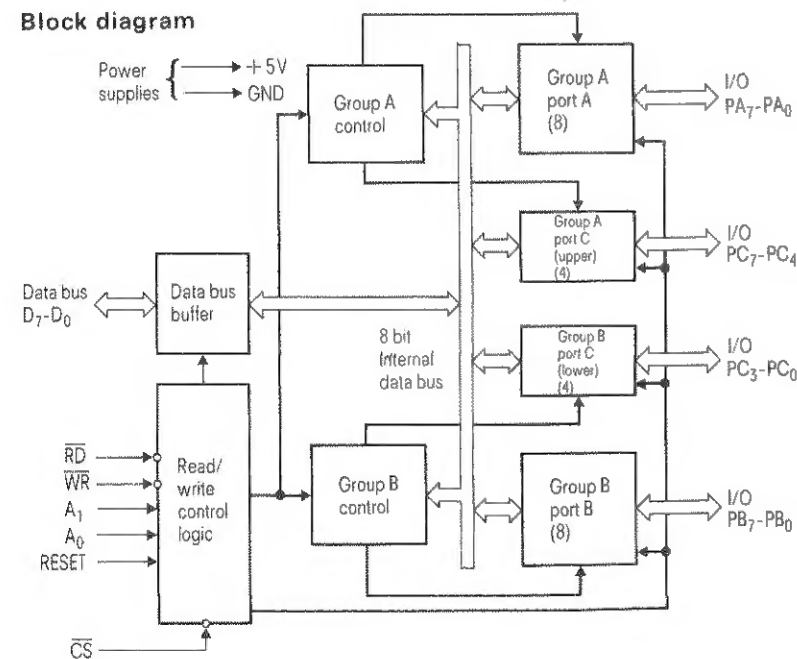


fig 1

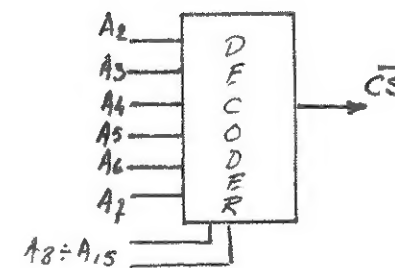
De RESET klem tenslotte laat toe alle programmatie van de bouwsteen ongedaan te maken. Door Het aanleggen van een laag niveau aan de \overline{CS} ingang komt de databusbuffer uit zijn derde toestand en is dataverkeer mogelijk van de microprocessor naar de bouwsteen (als WR laag is), of van de bouwsteen naar de microprocessor (Als RD laag is). Waar deze data belanden of van waar ze ingelezen worden is bepaald door het adres dat aan A₁, A₀ is aangelegd en wel volgens onderstaand tabelletje.

: CS	: A ₁	: A ₀	: RD=0	: WR=0	:
: 0	: 0	: 0	: kan. A naar CPU	: CPU naar kan A	:
: 0	: 0	: 1	: kan. B naar CPU	: CPU naar kan B	:
: 0	: 1	: 0	: kan. C naar CPU	: CPU naar kan C	:
: 0	: 1	: 1	: -----	: programmatie	:
: 1	: X	: X	: bouwsteen niet geselecteerd	:	:

Vermits bij de DAI PC een adres bestaat uit 16 bits volstaat het niet dat A₁ en A₀ een bepaalde waarde hebben maar dienen eveneens de bits A₂ tot A₁₅ in aanmerking genomen te worden.

De adressen voor de 8255 van de D.C.E. bus zijn:

- FE00 voor kanaal A
- FE01 voor kanaal B
- FE02 voor kanaal C
- FE03 programmatieregister.



De bits A₁ en A₀ van deze adressen liggen rechtstreeks aan de bouwsteen. De overige bits worden aan een zogenaamde adresdecoder gelegd. De adresdecoder geeft aan zijn uitgang een '0' niveau voor een welbepaalde combinatie of combinatiereeks van de ingangsbits. Dit nulniveau wordt dan gebruikt als \overline{CS} van de 8255.

Het gedeelte rechts van de interne bus vormt het interface-gedeelte met de buitenwereld en bestaat zoals hoger aangegeven uit twee 8-bits kanalen en twee 4-bits (halve) kanalen.

1.2 Programmatie van de 8255.

De 8255 kan in vier verschillende modi werken. Mode 0 laat toe A, B, C high en C low naar willekeur als input of als output te programmeren. In totaal zijn 16 combinaties mogelijk. Mode 1 laat unidirectionele gecontroleerde in- of output toe. Gecontroleerde I/O wil zeggen dat er besturingssignalen gebruikt worden die het datatransfert controleren. Deze besturingssignalen zijn verschillend alnaargelang het kanaal in input of output werkt. Het zijn:

Input: STB (strobe)
 IBF (input buffer full)

Output: OBF (output buffer full)
 ACK (acknowledgment)

Met het strobe signaal signaleert het periferieapparaat dat het data wenst over te maken aan de computer. Deze data worden ingelezen als STROBE laag is. Een hoog niveau op deze klem wil zeggen dat de data ingelezen zijn door de bouwsteen en dat door dit feit de inputbuffer vol is. Outputbuffer full signaleert aan het periferieapparaat dat data voor hem ter beschikking is. Het apparaat antwoordt met ACK als het deze data ingelezen heeft. Naast deze besturingssignalen wordt nog een interrupt signaal gebruikt. Hiermede kan de CPU verwittigd worden dat het betreffende kanaal data wenst uit te voeren of dat een periferieapparaat data wenst in te geven. Deze besturingssignalen ten getale van drie, worden ontleend aan kanaal C. In mode 2 kan alleen kanaal A gebruikt worden om bidirectioneel gecontroleerd data transfert door te voeren. Aan het kanaal worden 5 besturingsbits toegekend die weer ontleend worden aan kanaal C. Programmatie van deze modi die we verder nog in detail behandelen gebeurt aan de hand van een mo- dewoord dat als volgt is samengesteld.

Groep A		Groep B	
: 1	: M	: M	: A
: Ch	: M	: B	: Cl
0	0	mode 0	0 mode 0
0	1	mode 1	1 mode 0
1	X	mode 2	

Een 0 voor A,B,Cl,Ch programmeert het betreffende kanaal in output. Een 1 daarentegen zet het kanaal in input. Naast deze verschillende werkmodi kan de bouwsteen een willekeurig bit van kanaal C in input of output plaatsen. Daar- toe wordt een ander programmeerwoord gebruikt.

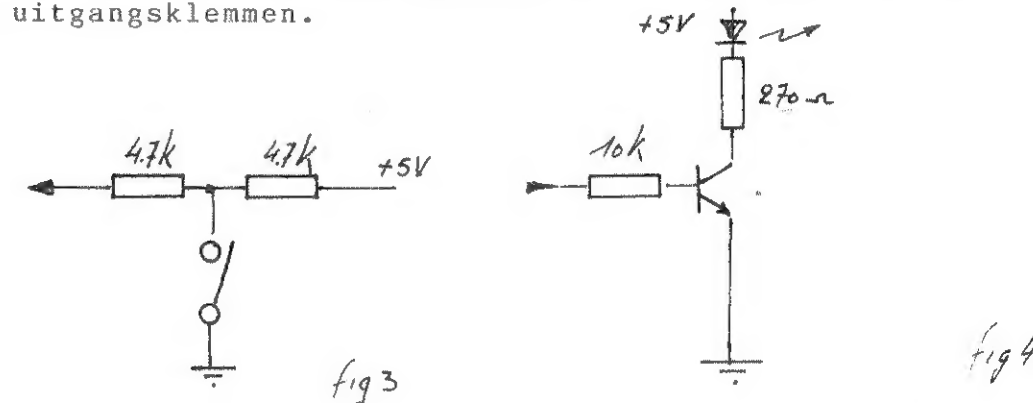
 : 0 : X : X : X : a : a : a : r/s :

a a a stelt binair het nummer van het bit van kanaal C voor
 r/s geeft aan dat dit bit als input (set=1) dan wel
 als output (reset=0) geschakeld wordt.

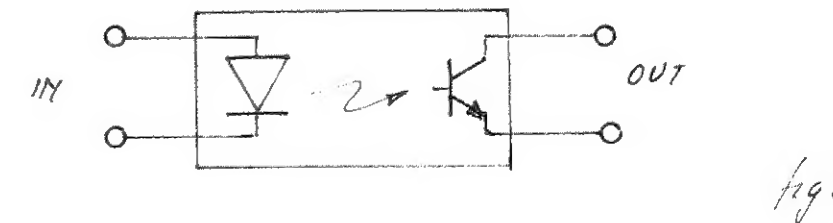
2. GEBRUIK VAN 8255 ALS GENERAL PURPOSE INTERFACE.

Bij het gebruik van de D.C.E. bus dient men er rekening me- de te houden dat de bouwsteen die de bus uitrust slechts een beperkte uitgangstroom kan leveren. Deze stroom is tevens af- hankelijk van het spanningsniveau dat op de uitgangspennen

aanwezig is. Bij een laag niveau is de maximale uitgangs- stroom 1,6 mA (deze stroom vloeit in de bouwsteen). Bij een hoog niveau is de maximale uitgangsstroom slechts 50 uA. Deze stroom vloeit uit de bouwsteen. Uitzondering mag op deze re- gel gemaakt worden voor slechts 8 op de 24 uitgangspennen die een stroom van 2mA mogen leveren dit met het oog op de sturing van darlington eindtransistoren. Rekening houdend met deze specificaties is het bijvoorbeeld niet mogelijk rechtstreeks LED diodes aan te sturen met de bouwsteen. Wenst men met de uitgangen te experimenteren of logica aan te sturen dan is het aangeraden een interface schakeling te gebruiken die in alle gevallen de D.C.E. bouwsteen beveiligd. Een voorbeeld van zulke schakeling wordt gegeven op fig 3 voor wat betreft de ingangen en op fig 4 voor wat betreft de uitgangsklemmen.



Wenst men daarenboven de D.C.E. bus te gebruiken om schake- lingen (apparaten) aan te sturen waarbij het gevaar bestaat dat op de verbinding een spanning kan geïnjecteerd worden (bijvoorbeeld de netspanning) dan is het wenselijk OPTO kop- pelaars te gebruiken. Een optokoppelaar is een digitaal cir- cuit dat in één behuizing een LED en een phototransistor be- vat. (fig 5) De galvanische isolatie die tussen deze elemen- ten bestaat, weerstaat aan 3000 tot 4000V. De te gebruiken schakeling kan bijvoorbeeld zijn zoals in fig 5.

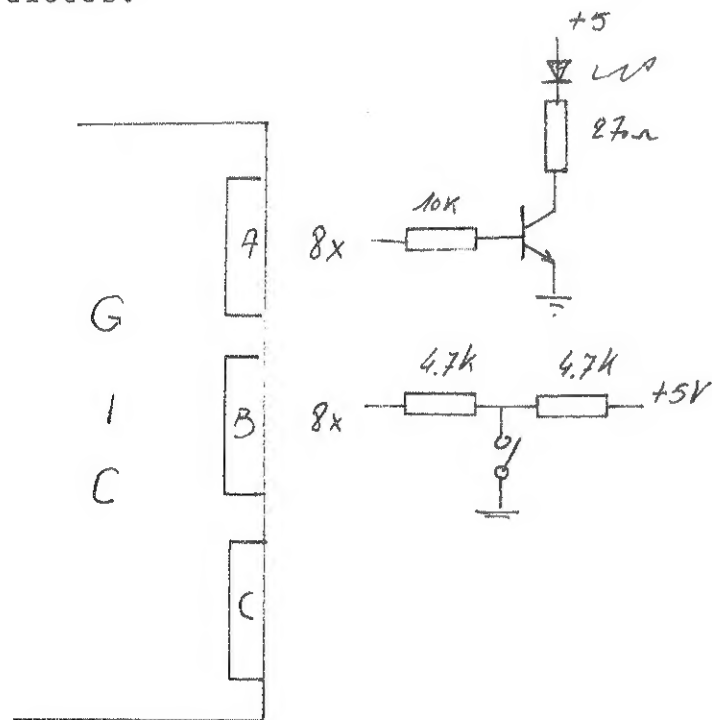


Wenst men een willekeurige toepassing te interfaceren met de D.C.E. bus dan dient men eveneens met de hogervermelde beper- kingen rekening te houden. Tussen de bus en de toepassing

dienen 'buffers' of 'latches'(grendelketens) opgenomen te worden. Beide schakelementen bestaan al dan niet in tri-state. Het verschil tussen een buffer en een latch bestaat in het feit dat een buffer de data versterkt (kan een hogere belastingsstroom leveren), terwijl een latch daarenboven deze data memoriseert. De data blijven onverandert in de latch tot een nieuwe inschrijving gebeurt.

2.1 Voorbeeld van gebruik van 8255 in mode 0.

Veronderstellen we dat kanaal A uitgerust is met 8 LED's en kanaal B met 8 schakelaars zoals aangegeven op fig 6. We wensen een programmatje te schrijven dat voortdurend de stand van de schakelaars inleest en deze uitgeeft naar de diodes.



Het programmatje zou als volgt kunnen zijn.

```

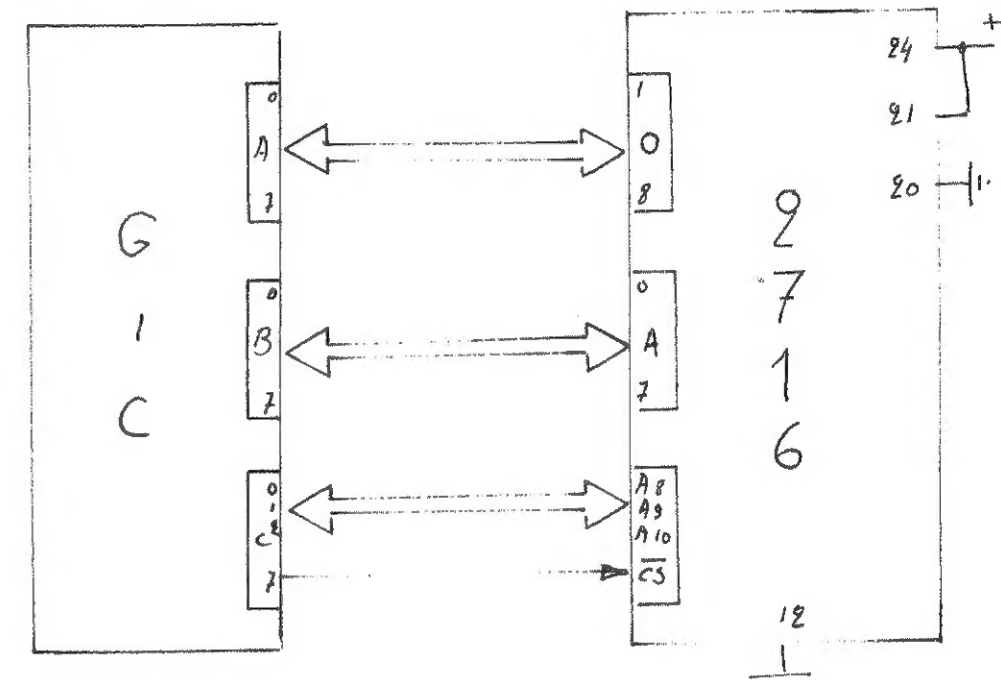
INIT   MVI  A, :82      Initialiseren 8255
        STA  :FE03      kan A output, kan B input
IN     LDA  :FE01      Inlezen kanaal B
OUT    STA  :FE00      Uitgeven naar kanaal A

```

2.2 Voorbeeld van gebruik van 8255 in mode 0.

Veronderstel dat we op de DCE bus een Eprom wensen aan te sluiten en dat we per programma de inhoud van deze EPROM in geheugen wensen te lezen. Zoals uit fig. 7 blijkt is kanaal A verbonden met de data lijnen 01 to 08 van de EPROM (pennen 9, 10, 11, 13, 14, 15, 16 en 17). Kanaal B is verbonden met de adreslijnen (pennen 1, 2, 3, 4, 5, 6, 7 en 8). Bit 0, C0, C1 en C2 van kanaal C zijn verbonden met de adreslijnen

A8, A9 en A10. Bit 7 van kanaal C tenslotte is verbonden met de selectieingang van de Eprom. We stellen ons voor gans de inhoud van de Eprom te lezen en onder te brengen in het geheugen van de DAI beginnend vanaf het adres BUFFER (in ons geval :9000). Vermits de Eprom van het type 2716 is, bedraagt de lengte van het in te lezen blok :7FF bytes. Het programma wordt in geheugen gebracht vanaf adres :300 en kan dus in Utility gestart worden met G300.



Het programma voor het lezen van de Eprom kan er als volgt uitzien.

```

BUFFER EQU :9000      Beginadres buffer.
LENGTE EQU :07FF     Lengte te lezen blok.
STRTAD EQU :0000     Bebinadres Eprom.

INIT   ORG :300
        MVI  A, :90      Initialisatie 8255
        STA  :FE03
        LXI  D, STRTAD   Laden adrespointers.
        LXI  H, BUFFER
        LXI  B, LENGTE
RDKAR  MOV  A, E         Uitgeven LSB adres.
        STA  :FE01
        MOV  A, D         Uitgeven MSB adres.
        STA  :FE02
        LDA  :FE00       Inlezen van een karakter.
        MOV  M, A        Wegschrijven naar buffer.
        INX  H           Ophogen adrespointers.

```

```

INX D
DCX B      Lengte = Lengte - 1
MOV A,C    Testen of lengte =0 ?
ORA B
JNZ RDKAR  Indien neen, volgende karakter
JMP :EA42  Indien ja terug naat Utility.

```

3. GEBRUIK VAN 8255 ALS GIP IN MODE 1.

Mode 1 laat toe kanaal A of B te programmeren als gecontroleerde input of output. Het datatransfert wordt dan zoals hoger gezien bestuurd door controlesignalen. Drie van deze controlesignalen, te weten STB, OBF en ACK zijn actief laag, het vierde signaal IBF is actief hoog. Herhalen we tevens dat twee van deze signalen (STB en IBF) gebruikt worden als het kanaal in output werkt, de twee andere (OBF en ACK) als het kanaal in input werkt. Deze signalen worden ontleend aan kanaal C en wel als volgt:

kan B	PC0	PC1	PC2	PC3	PC4	PC5	PC6	PC7	kan A
input	INTR	IBF	STB	INTR	STB	IBF	I/O	I/O	input
outp.	INTR	OBF	ACK	INTR	I/O	I/O	ACK	OBF	outp.
	groep B			groep A					

Om het mogelijk te maken onder interrupt te werken zijn naast de hogervermelde besturingssignalen nog interrupt aanvraagsignalen voor elk van de kanalen voorzien. Men kan zich op elk ogenblik van de toestand van de besturingssignalen vergewissen door het kanaal C te lezen. De informatie van kanaal C is als volgt samen te vatten.

	kanaal A input			kan. B input				

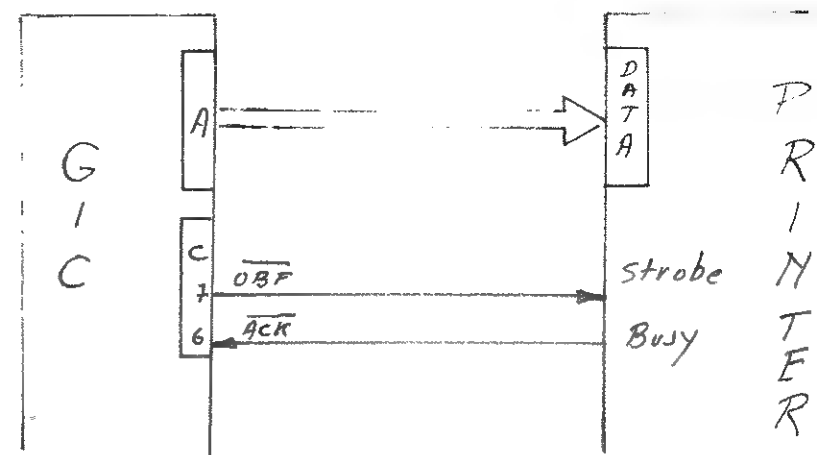
	:I/O	:I/O	:IBF	:INTE	:INTR	:INTE	:IBF	:INTR:

	kanaal A output			kan B output				

	: OBF	:INTE	:I/O	:I/O	:INTR	: INTE	:OBF	:INTR:

3.1 Voorbeeld gebruik van 8255 in mode 1.

Als voorbeeld in mode 1 wensen we een parallele printer te interfaceren met de DAI PC.



Vermits kanaal A in output geprogrammeerd wordt ziet het formaat van het statusregister er als volgt uit

```

-----
: OBF: INTE : X : X : INTR : X : X : X :
-----

```

De besturingssignalen OBF en ACK liggen respectievelijk aan de klemmen PC7 en PC6 van kanaal C.

De gang van zaken bij de besturingssignalen is als volgt. Nadat een karakter in de uitgangsbufer van de 8255 is geschreven wordt de DATA STROBE signaal van de printer gegenereerd door OBF laag te zetten. Het ontvangstmeldingssignaal BUSY van de printer geldt als ACK, het zet tevens het OBF terug op 1 zodat een volgende karakter kan uitgegeven worden.

De programmaroutine DOUT zou er als volgt kunnen uitzien.

```

ICW      EQU  :B0
CWR      EQU  :FE03
OBFON    EQU  1
OBFOF    EQU  0
KANA     EQU  :FE00
KANB     EQU  :FE01
KANC     EQU  :FE02
PRBUSY   EQU  :80

DOUT     ORG   :300
        LXI  H,BUFFER  Initialisatie pointers
        LXI  B,LENGTE
        MVI  A,ICW     Initialisatie 8255
        STA  CWR
        MVI  A,OBFON   OBF niet actief
        STA  CWR
KAROUT   LDA  KANC     Lezen status
        ANI  PRBUSY   Testen Busy van printer.
        JNZ  KAROUT
        MOV  A,M       Uitgeven van karakter
        STA  KANA
        MVI  A,OBFOF   OBF activeren
        STA  CWR
        INX  H         Pointer ophogen
        DCX  B         Lengte = lengte - 1
        MOV  A,C       Rest 0 ?
        ORA  B
        JNZ  KAROUT   Indien niet volgende karakter.
        RET

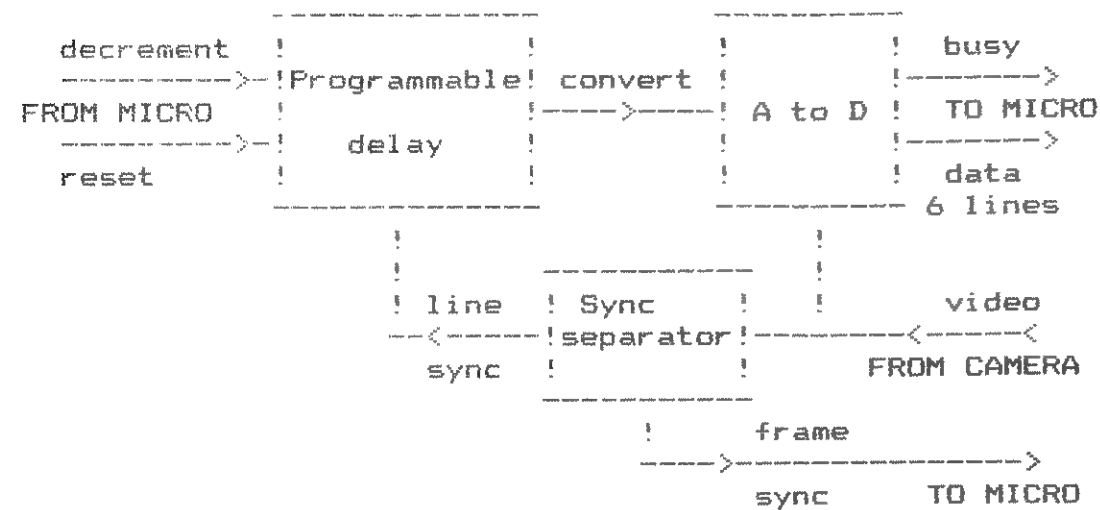
```

DIGITIZING

THE INTERFACE

DAInamic bought the videocamera interface from Educational Electronics through their BENELUX representative Entrac B.V. in Diemen, the Netherlands. The interface was delivered with its own powersupply and a flatcable with two 20 pin connectors. A videocable (with BNC connectors) was not provided. The interface needs a 1 Volt p/p composite videosignal from a camera or videorecorder and 10 lines to the computer. A User Guide (which was not very helpfull) completed the package.

BLOCK DIAGRAM



The video signal is split into data and timing signals. The timing of a frame (20ms) and of a line (64us) are output to the computer as well as 6 data lines which provide 64 levels of grey. The two inputs to the interface are used to reset and to decrement the programmable delay. A TV signal consists of 50 frame scans/second, each frame consists of 312 lines vertically. In the interlace mode alternate frames are interleaved giving the normal 625 lines. The programmable delay splits the line into 256 'pixels' horizontally of which the leftmost and rightmost are not useable. This gives a resolution of 220 pixels x 312 lines. The first and last lines of a frame are not visible on a normal TV which leaves approx. 256 lines visible (e.g. MODE 5 & 6).

THE SOFTWARE

1. Making a proportional picture.

The aspect ratio is thus 220 to 256. To obtain a proportional picture the ratio should be about 336 to 256. (MODE 5 & 6): This means stretching one pixel of the interface into one and a half MODE 5/6 pixel. Or doubling the width and to make the picture in mode 7/8. The first possibility is very unpractical especially in a 16 color

mode. The second possibility gives a good aspect ratio on the screen but is non square on a printer. The software now uses a 220x242 picture mapped on to a 440x242 screen (MODE 7/8) or a 220x242 picture mapped on to a 220x121 screen (by averaging two lines to one line). The software for different MODEs uses the same aspect ratio but lines are trimmed at the edges and lines from top and bottom are skipped or the picture is vertically and horizontally averaged.

2. Interfacing & timing

The lines FROM the interface have been wired on to a 74 pin connector for the DCE-bus. The 8 outputs to port A, the two inputs to PORT C. The routine to fetch the data was written in the stackRAM to ensure a proper timing, the conversion routines were written in RAM.

3. Digitizing

schematically:

- | | |
|-------------------------|---|
| 1. INITIALISE INTERFACE | reset delay to leftmost 'pixel'. |
| 2. WAIT FOR FRAMESYNC | digitizer is at the topline. |
| 3. SKIP LINES | skip invisible toplines by counting linesyncs. |
| 4. DIGITIZE 256 LINES | read 256 lines of data and store it in a buffer. The data is valid when the linesync is high. |
| 5. DECREMENT DELAY | delay one 'pixel' more to the right |
| 5. REPEAT | repeat from step 2. until 4 or 8 lines done. |
| 6. CONVERT DATA | the data in the buffer is converted and mapped onto the screen. |
| 7. REPEAT | repeat from step 2 until delay is at leftmost pixel. |

the 64 greylevels are converted to 4 colour mode by dividing the data by 16, for a 16 colour mode the data is divided by 4. Software is available for MODE 2 to MODE 8 together with two screencopy routines for EPSON printers. The software is (not yet) fully compatible with MDCR. Digitizing can be done with the MDCR connected but the interface has to be disconnected to work with the MDCR. A picture in MODE 8 takes about 12 seconds (5 seconds for digitizing and 7 to put the picture on to the screen).

4. Controls

The interface has two controls; one to adjust the SYNC to ensure correct timing and the other to adjust GAIN which increases the outputlevel.

Samples of digitized pictures from camera and videorecorder are published in this and previous magazines. The interface works Ok. The only disadvantage is that it is slightly overpriced: Dfl.1150,-- or 21000 Bfr. (ex VAT).

N.P. Looije

== The GETC of the DAI.==
 =====
 (Translated from "Octet"84/3)

Here is the reply to a question raised in the "TRICKS and TIPS" section of last month. It was asked to find a method for displacing a moving spot on the screen, using the cursor keys and without using the REPEAT key. The technique can of course be extended to the other keys. The present paper was sent to us by M.Fabien F.O.D.J.U.D from Creuzwald(France), and the program illustrating the first solution is by M.Walter COSTA :

The GETC of our dear DAI has indeed the inconvenience of not permitting the AUTOREPEAT without simultaneous use of the REPEAT-key and of the key in question. This is rather annoying when producing such programs as graphics, games, etc...

But there exists a solution to every problem, and everything can be arranged by means of a few PEEK's and POKE's. To my knowledge, two methods do exist(who can provide more?). Those two methods rely on the keyboard organization as described on page 34 of the DAI Personal Computer Reference Manual.

* FIRST SOLUTION *

Each of the addresses #2B1 up to #2B8 scan one line of 7 columns of the keyboard matrix. These lines are encoded as a 7 bits byte, the LSB (bit 0) being at the left. For instance, when the RETURN-key has been pressed, #2B1 contains #04.

In order to make this reading possible, one should not forget to POKE 0 in #2B9 because, should it contain #FF, then only the BREAK key will be scanned.

* SECOND SOLUTION *

It is somewhat more complex than the previous one, and it is generally used in machine language. Here also, thanks to the Reference Manual, we know that #FF07 is the address of the output port of the keyboard, and that #FF01 the address of its input port.

In practice, put the matrix line to be selected into #FF07, and read the result from #FF01. For instance, if the "N" key is to be tested, #FF07 has to be set to #40(byte corresponding to the line containing "N") and the test will be performed by means of:

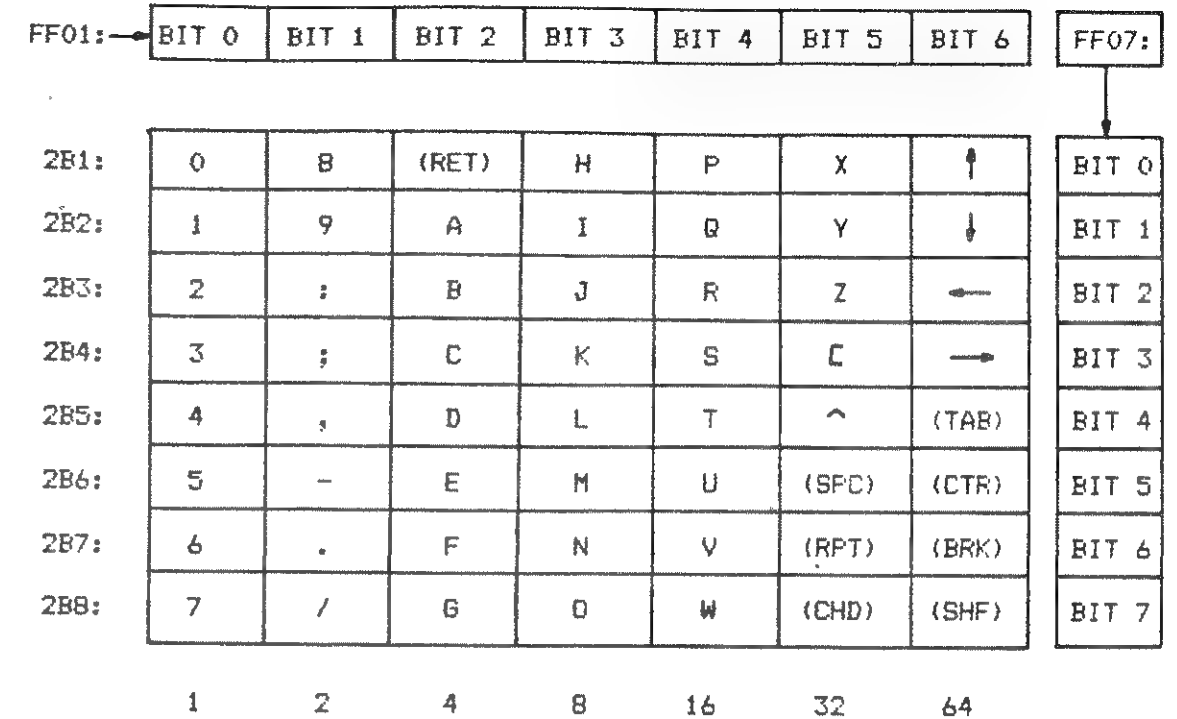
IF PEEK(#FF01)=#08 THEN....

(#08 is the byte corresponding to the column containing "N")

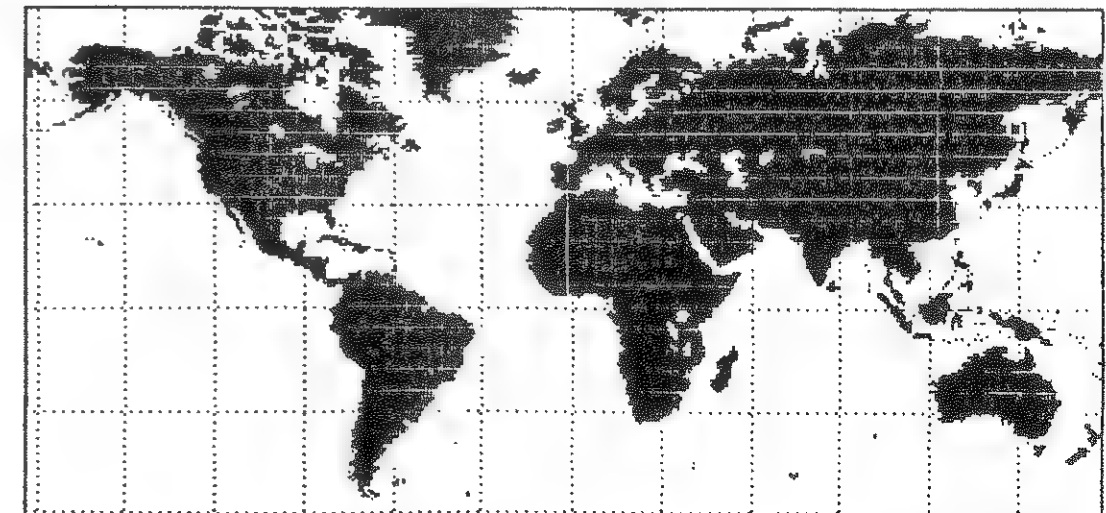
* REMARK *

It is to be noted that both these methods allow for pressing two keys simultaneously!

PORTS DU CLAVIER:



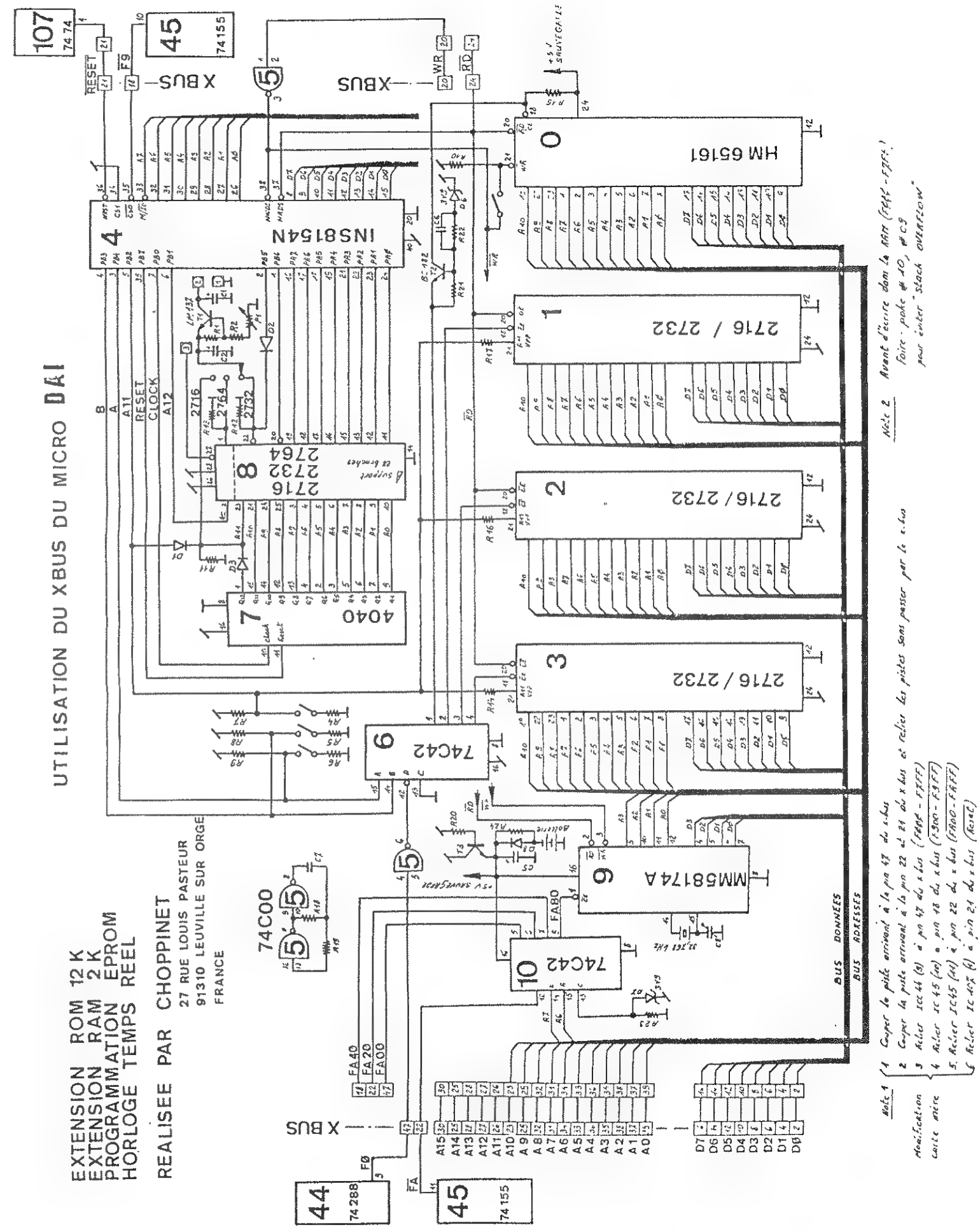
Fabien FOJUD



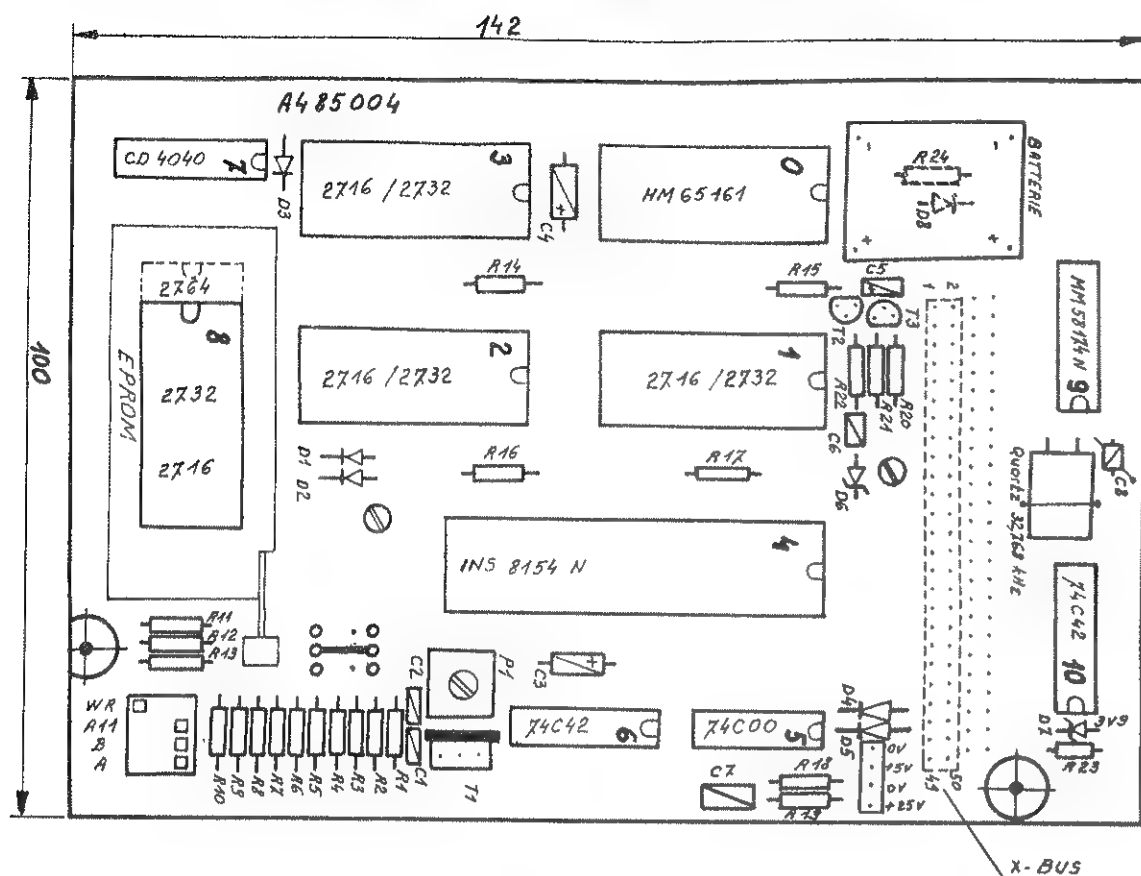
```

10 REM
20 REM *****
30 REM *** Substitute for GETC on the DAI ***
40 REM ***          COSTA          ***
50 REM *****
60 REM
70 REM All there is to do is to perform the corresponding PEEK into the
80 REM keyboard memory. Before that, do not forget to unlock the key-
90 REM board by means of POKE #2B9,0. See DAI Reference Manual, page 34.
100 REM The eight rows correspond to addresses #2B1 to #2B8. The columns
110 REM contain the powers of 2 (least significant bit on the left).
120 REM
130 REM EXAMPLES:
140 REM
150 REM When pressing on A, one obtains 4 in #2B2.
160 REM When pressing on ^ (exponentiation), one obtains 32 in #2B5
170 REM When pressing on A and on I, one has 12 (=4+8) in #2B2
180 REM If no key is pressed, the memories contain 0 or 128.
200 REM
210 REM APPLICATION:
220 REM
230 REM Displacement of a moving spot by means of the four cursor keys:
240 REM
250 POKE #2B9,0:MODE 3:X=XMAX/2:Y=YMAX/2:COLORG 0 1 2 3
260 XP=X:YP=Y
270 IF PEEK(#2B1)=64 THEN Y=Y+1
280 IF PEEK(#2B2)=64 THEN Y=Y-1
290 IF PEEK(#2B3)=64 THEN X=X-1
300 IF PEEK(#2B4)=64 THEN X=X+1
310 DOT XP,YP 0:REM Previous dot is erased.
320 DOT X,Y 1
330 GOTO 260
340 REM
350 REM One may press two keys at the same time (diagonal displacement)
360 REM
1000 REM In order to recover the ASCII code of the key that was pressed,
1005 REM do as follows:
1010 REM
1020 POKE #2B9,0:POKE #2BE,#BA
1030 P=PEEK(#2BA):CAR#=CHR$(P)
1040 PRINT P,CAR#
1050 REM POKE #2BE,#BA
1060 GOTO 1020
1070 REM
1080 REM Memory #2BA keeps the code of the last key being pressed. If
1085 REM it is not desired, add following line:
1100 REM 372 POKE #2BE,#BA

```



X-BUS CARD



X-BUS CARD by ERIC CHOPPINET

The X-bus extension-card developed by Eric Choppinet offers following features :

- * 2K RAM with battery - backup
- * 12K ROM with bankswitching on addresses HF000 - HF7FF
Bankswitching is done by a PIA 8154
- * REAL TIME CLOCK (on address FAXx)
- * EPROM PROGRAMMER (also supported by 8154)

This card has to be connected to the X-BUS (inside computer) in the same way as the TOS-card of DCR.

TIPS

THIS FEATURE HAS BEEN STARTED PREVIOUSLY, BUT UNFORTUNATLY IT HAS BEEN DISCONTINUED. SO IN THE MID 80'S WE DECIDED TO REVITALIZE THIS SECTION OF YOUR DAINamic MAGAZINE. THE TIPS WILL COME MOSTLY IN RESPON FROM READERS'S QUESTIONS OR RELEVANT PUBLICATIONS ELSEWHERE. ALL QUESTIONS AND TIPS WORTHWHILE, ELEMENTETRY FOR THE NOVICES AND HIGH-TECH FOR THE VERY SOPHISTICATED AND ANYTHING IN BETWEEN, WILL BE PUBLISHED FOR YOUR BENEFIT. SO PLEASE DON'T HESITATE,ASK ALL YOU WANTED TO KNOW ABOUT YOUR 8080 AND NEVER DARED TO ASK.

WIM BREMER DESK EDITOR; HANS STREEP ENGL.DEPT.ED. JAN 85

TIP 1
WHENEVER YOU SEE IN A PROGRAM; A WAIT FOR INSTRUCTION:
100 G=GETC: IF G&ç32 THEN GOTO 100
YOU MAY SUBSTITUTE:
100 CALLM ùD6DA
(TIP CONTRIBUTED BY THEO VERBERKT)

TIP 2
WHEN WORKING WITH A DCR (DIGITAL CASS. REC.) AND LOADING BASICODE II FROM AUDIOCASSETTE WE'LL GET AFTER A DCR COMMAND A SYNTAX ERROR.
SOLUTION: REMOVE THE CASSETTE FROM THE DCR AND TYP: ùF2F2

TIP 3
MEMORY BLACKOUT DUE TO RESET FAILURE?
REINHARDT DIECKMANN A CLUB MEMBER OF THE GERMAN DAI NOTICED A DESIGN FLAW IN THE PREVIOUS DAI MODELS THE LEAD 2 OF THE IC94 IS DURING A REGULAR RESET CONNECTED TO GROUND. HOWEVER DURING A NORMAL OPERATION THIS SAME LEAD IS ISOLATED, LEFT ONLY TO COLLECT LIKE AN ANTENNE STATIC DISCHARGES AND OTHER DISTURBANCES.ALSO SHORT POWER INTERRUPTIONS MAY EFFECT THE IC94 LEAD CAUSING A 'FATAL' MEMORY LOSS

SOLUTION: CONNECT THE CENTER LEAD OF THE RESET SWITCH WITH A 240 OHM RESISTOR(note 1. WITH THE WIRES PROTECTED AGAINST SHORTCIRCUITING TO A 5 VOLTS RED (TANTAL) CAPICITOR. THIS MINOR OPERATION PREVENTS ACCORDING TO REINHARDT MOST OF THOSE UNDESIED MEMORY BREAKDOWNS (TIP CONTRIBUTED ALSO BY INNO BROEKMAN, WHO NEVERTHELESS WISHES TO ACCEPT NO LIABILITY DUE TO POSSIBLE ERRORS.) THUS BE SURE TO GET OUT THE OLD VOLTMETER, AND PROCEED WIH DUE CAUTION.
note 1. THEO VERBERKT PREFERS STRONGER MEDICINE, HE CALLS FOR A 4K7 OHM RESISTOR.AND REPORTS THE DAI DESIGN ENGINEERS TO HAVE IMPROVED THE NEW SERIES ç=7.

TIP 4
A SIMULAR IF NOT IDENTICAL PROBLEM, ALWAYS WHEN LEAST EXPECTED AND WIPING OUT MUCH TIME AND EFFORT FROM YOUR PART. YOU GUESSED IT: THE FEARED GREEN RESET-DEMON!
BELOW YOU'LL FIND A LISTING WICH ACCORDING TO ROBERT SIP WILL REANIMATE UP TO 99,999% OF SUCH CASES.PROVIDED OF-COURSE YOUR PROGRAM HASN'T COMMITTED SUISCIDE IN ONE OF THOSE TIMELESS LOOPS

CURE:

1. PLEASE DON'T PANIC, AND DON'T CURSE;
2. RESET IN CASE IT IS NECESSARY;
3. UT
Z3
S29B 02-00 EC-A6
B
NEW;
4. LOAD PROGRAM "WHAT TO DO WHEN A CRASH OCCURS";
5. RUN;
6. 2EC FOR ANY REGULAR PROGRAM;
or G-00 FOR A ("LOST") MACHINE LANG. PROGRAM ie
G900 FOR FGT
7. LIGHT A CANDLE TO YOUR FAVORITE SAINT;
8. THE PROGRAM ADDRESSES SHOULD BE: HEAP
START OF TEXT
START OF SYMBOL TABLE
END OF SYMBOL TABLE
9. RETURN POINTERS 29B AND 2A4; AND TURN ABOUT LSB AND MSB
10. B BASIC;
11. LIST. YOU SHOULD BE IN COMPLETE CONTROL NOW.
12. WHEN NOTHING HAPPENS, BLOW OUT THE CANDLE AND SKIP THE
DONT'S AT 1.
13. RECONSIDER TIP 3. OR BETTER YET OBTAIN A COPY OF FWP
FAST WORD PROCESSOR FROM DAINamic .SERIOUSLY TO BE CON-
SIDERED FOR ANY EXTENSIVE WORD PROCESSING, AND HIGHLY
RECOMMENDED BY YOUR ENGL. TRANSL. ED.

COPY LISTING "WHAT TO DO WHEN etc." 1-240

TIP BY ROBERT SIP TRANSLATED BY W.B. FOR THE ORG. JAN/FEB
'82 DAINamic PUBL.

QUESTION: WHO WOULD LIKE TO ADOPT THE ROBERT HARVEY AND
THE ROBERT SARGENT VERSION OF THE SOFTWARE PROGRAM FOR THE
SPECTRUM SOLID STATE DRAM 4116 AS PUBL. IN THE ELECTRONICS
AND COMPUTING APR MAY '84 IN ORDER TO GIVE THE DAI SOLID
STATE VISION?

PART ONE OF THE APRIL ARTICLE IS MISSING DOES ANYONE HAVE
A COPY FOR THE LANG. ED.?

QUESTION: I LIKE TO SET UP A SEMINAR INVESTIGATING CD
(COMPACT DISK) AS A READ AND WRITE STORAGE MEDIUM FOR THE
COMPUTOR. I NEED THE SANYO DAD 8 CD SPECS. AND FEEDBACK
FROM YOU IF YOU ARE SERIOUSLY INTERESTED

ALL REMARKS ABOUT TIPS CAN BE DIRECTED TO THE DAINamic
EDITOR, WIM BREMER.

PLEASE MAIL YOUR RESPONSE REFERRING TO SOLID STATE AND CD
DIRECTLY TO:

J STREEP
CZ PETER ST 101 AMSTERDAM (C)
1018 PE THE NETHERLANDS
TEL 020-279862



SUPERBASE



100% MACHINE CODE

36000 BYTES FREE

WITH COLOR-EDITOR

SPECIAL QUICKSORT

AUDIO : 1600 BFR

DCR : 1750 BFR



From now on there is a data-administrationsprogram for the DAI. It is completely written in Machine Code, relatively short and very quick when sorting data. (Sorts 630 items in about 10 s) The program uses the complete available Memory space, in the Disk-version you have about 29500 Bytes storage capacity, and in the Cassette (or MDCR) version, remain around 36000 Bytes free to use.

One Data-item can have max. 20 Criteria with 40 characters max. each.

One Data-item equals one card-system, on which 20 Subgroups can be stored. The program is Menu-orientated, showing the commands available at one key-press.

For example the Main Menu:

Data-Administration V2.1 Disk

- R Add Data
- A Memorize
- B Define screen-layout
- E Adding
- K Korrekt
- S Sort
- D Printout
- L Delete
- C Change text
- U Reformat Data
- Z Back to Basic

SUPERBASE

The Print-Menu:

What Data must be printed

- B) All
- S) Data between to numbers
- T) Data between two textparagraphs
- I) Data with specified content
- E) Input of text for circular letter
- L) Load text
- A) Memorize text
- K) Erase text
- R) Initialize Printconditions
- W) Choose Printconditions
- B) Choose List/Text
- Z) Back to Main Menu

Input is done very conveniently in a special Editor. At input, the wanted Criterium is printed in color. On one single line, you can go left and right in text with the Cursors, and new text is automatically added at cursorposition.

With Cursor Up/Down, you can switch to another Criterium. With Shift-Cursor left/right you can go back or forward to another Data-Item. That means: The single Data-items are ordered like adjacent reference cards, which can always be easily accessed.

Input/Correction Lfd.Nr.: 200

Name Wienkop
First Name Uwe
Street Laerfeldstr. 54
Place 4630 BOCHUM 1

When printing your data, you can use the standard scheduled Form, or create a layout of your own. You can even create circular letters, in which you use your own Data.

Regardless the possibilities you choose, you always have complete Search Capability. For example: you could search for specified Data: all persons with the same Postal Number etc...

At Search, a special Quick-Sort routine in machine Code is used! You can compare your Data for max. 2 different Criteria, when Data are corresponding for the first Crit, then they are compared for the second.

I hope to have shown you in brief the possibilities of this program. Please contact me for further questions.

(Translation by Staf Van Hoecke.)

EXTRA - SUPERBASE FILE CONVERTOR - EXTRA

Many of us have already typed in a lot of data in other database-programs as MAILING LIST or selfmade programs, using ordinary string-arrays.

In order to save us from typing all this information again, Freddy Deraedt has created a conversion program to transfer stringarrays into SUPERBASE file format.

As SUPERBASE can hold much more information than MAILING LIST, the conversion routines offer the possibility to merge different stringarrays into one SUPERBASE file.

These conversion routines are included in the SUPERBASE package, together with complete documentation.

Please indicate with your order if you want the Dutch or the English documentation, French documentation will be available later.

10 bits A/D

Erps-Kwerps, 15 Januari 1985.

Uytterhoeven Fr.
Kammestraat 34,
3071 Erps-Kwerps.

Beste Redactieleden,

Op de eerste plaats stuur ik U mijn beste Kerst-en Nieuwjaarswensen. Tevens maak ik van de gelegenheid gebruik om U één van mijn ontwerpen voor te stellen nl. een 4 kanaals, 10 bits DCE-bus compatibel analoog-digitaal converter. Dit is een toestel dat programmeerbaar is vanuit machinetaal of BASIC.

Het doel van dit ontwerp was het ontwerpen van een 10 bits 4 kanaals data-logger voor het meten en verwerken van metingen op spierweefsels in het Laboratorium voor Fysiologie van de U.C.L. (Université Catholique de Louvain)

Het ontwerp is gebouwd op de universele interfacekaart die beschreven staat in DAInamic nr. 14. Een voordeel hiervan is dat de DCE-bus verder volledig kan benut worden voor andere of bestaande uitbreidingen omdat de analoog-digitaal converter slechts 1 van de 16 beschikbare adressen in beslag neemt.

De twee duurste elementen die in het ontwerp voorkomen zijn de converter AD571K en de multiplexer AD7502 beide van Analog-Devices. De data sheets hiervan zijn dan ook bij deze firma gemakkelijk verkrijgbaar.

Bovenaan links, boven de streeplijn ziet U op het schema (Fig.1) de verbinding tussen de DAI pc. en de universele interfacekaart. Op de interfacekaart moeten de volgende lijnen, CS, A0, A1, en RD met een scherp mes worden onderbroken en moet de flip flop 74LS123 met de poorten CD 4081, CD4011, CD4001 rechts bovenaan de streeplijn worden tussengelast. Dit is nodig opdat bij elke omschakeling van kanaal (wanneer in poort C van de 8255A op de interfacekaart bit 0 en bit 1 worden aangesproken) de input buffer van de 8255A en de outputbuffer van de AD571K geleidigd zouden worden. Hardware-matig wordt een nieuwe conversie cyclus gestart. Dit bespaart ons instructies en dus ook tijd.

De flip flop 7474 wordt gebruikt als latch voor D0 en D1 (de 2 minst beduidende bits van de AD571K). De niet gebruikte bits (bit 2 tot bit 7) van PB moeten worden met de massa verbonden aangezien PB evenals PA als input wordt gebruikt.

De dubbele monostabiele flip flop 74LS123 geplaatst tussen de uitgang DR (data ready) van de AD 571K en de ingang strobe van de 8255A worden gebruikt om een correcte timing te bekomen tussen beide. (zie timing diagram fig.2)

Bit 7 van poort C gebruiken we als synchronisatie ingang,

d.e. door middel van de 3 rand poorten en een monostabiele FF (3*1/4 CD4011 en 1/2 74LS123) wordt gestuurd. Zolang we de gate ingang laag sturen, halen wij doorlopend data naar binnen, (2000 samples/sec) wanneer we echter de sync. input gebruiken, krijgen we 1 data/sync.puls. Dit gedeelte van de schakeling is enkel nuttig wanneer we onze ADC uit machinetaal sturen en mag volledig worden weggelaten wanneer we enkel BASIC gebruiken (max. 400 samples/sec). Wanneer we geen synchronisatie ingangen nodig hebben naar toch de ADC zowel in machinetaal als in BASIC willen sturen, moeten wij van poort C bit 5 met bit 7 verbinden omdat we in machinetaal op bit 7 ons IBF signaal testen.

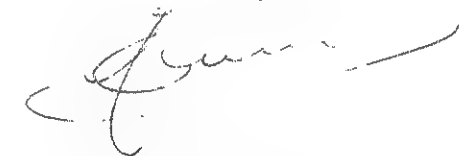
De 74LS156 wordt gebruikt als 2 to 4 decoder om d.m.v. 4 LED'S aan te duiden op welk ingangskanaal we ons bevinden.

Verder is er een betrekkelijk snelle operationele versterker LF 356 gebruikt, differentieel geschakeld met een bandbreedte van 50 kHz., dit omwille van de multiplexer AD 7502. Samen met een ingangsimpedantie van 100 kohm en een versterkingsfactor van 10X bekomen we dan 4 differentiele gemultiplexte ingangen met een gevoeligheid van 0 tot 1 Volt of van -0.5 tot +0.5 Volt per kanaal, dit naargelang men al dan niet bipolair werkt. Na het afregelen van de offsetspanning van de LF 356 met de 25 kohm pot. iken we onze ADC met de regelbare 200 ohm pot. aan de input (pin 13) van de AD 571 opdat we een gevoeligheid van 1mV/punt bekomen. (zie application note AD 571)

Alle IC's worden gevoed met een spanning van +5 Volt behalve de LF356 met +/-15 Volt, de AD571K en de AD7502 hebben bijkomende spanningen van respectievelijk -15 Volt en +/-15 Volt nodig. De voeding is weergegeven in fig.3. en is universeel voor alle andere interfacekaarten op de DCE-bus aanwezig.

Hierbij stuur ik U 5 verschillende BASIC routines, die elk naargelang van de toepassing als subroutine in een programma kunnen gebruikt worden, ook een assembler programma dat opgeroepen wordt vanuit het bijhorende BASIC programma. Om volledig te werken moet hieraan echter nog een screencopy programma toegevoegd worden dat start op adres #300 en past bij de gebruikte printer.

Hoogachtend,



Fr. Uytterhoeven.

```

5 REM 2 BITS CONVERSION
10 OUT #13,#BA:REM CARD ADD 1 #BA IN COMMAND REGISTER
20 OUT #12,0:REM CARD ADD 1 SELECT CHANNEL 0
30 AX=INP(#10):REM INPUT 8 MSB
40 PRINT AX,
50 GOTO 30
    
```

```

5 REM 10 BITS CONVERSION
10 OUT #13,#BA:REM CARD ADD 1 #BA IN COMMAND REGISTER
20 OUT #12,0:REM CARD ADD 1 SELECT CHANNEL 0
25 B%=INP(#11):REM INPUT 2 LSB
30 AX=INP(#10) SHL 2:REM INPUT 8 MSB
40 PRINT AX+B%:"mV",
50 GOTO 25
    
```

```

5 REM 8 BITS CONVERSION BIPOLAR
10 OUT #13,#BA:REM CARD ADD 1 #BA IN COMMAND REGISTER
20 OUT #12,0:REM CARD ADD 1 SELECT CHANNEL 0
30 AX=INP(#10)-128:REM INPUT 8 MSB
40 PRINT AX,
50 GOTO 30
    
```

```

5 REM 10 BITS CONVERSION BIPOLAR
10 OUT #13,#BA:REM CARD ADD 1 #BA IN COMMAND REGISTER
20 OUT #12,0:REM CARD ADD 1 SELECT CHANNEL 0
25 B%=INP(#11):REM INPUT 2 LSB
30 AX=INP(#10) SHL 2:REM INPUT 8 MSB
40 PRINT AX+B%-512.0:"mV",
50 GOTO 25
    
```

```

5 REM FAST 8 BITS CONVERSION
10 CLEAR 30000
20 DIM AX(80.0,80.0)
30 OUT #13,#BA:DUT #12,0
40 FOR I%=0.0 TO 80.0
50 FOR J%=0.0 TO 80.0
60 AX(I%,J%)=INP(#10)
70 NEXT J%:NEXT I%
80 FOR I%=0.0 TO 80.0
90 FOR J%=0.0 TO 80.0
100 PRINT AX(I%,J%),
110 NEXT J%:NEXT I%
    
```

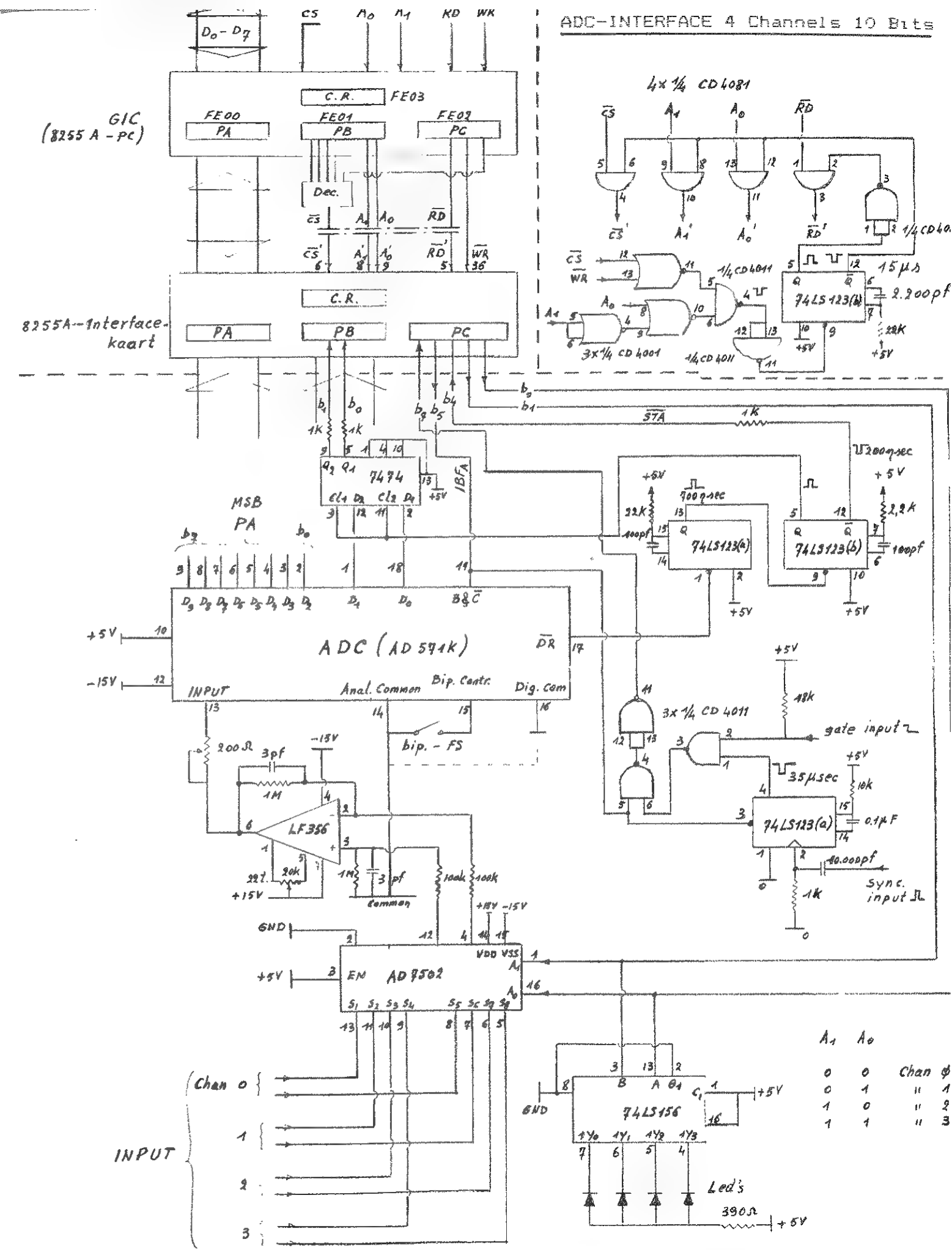


Fig. 1

Control word (8255A) : # BA
 - PA : Strobed input Mode 1
 - PB : Input Mode 0
 - PC : output

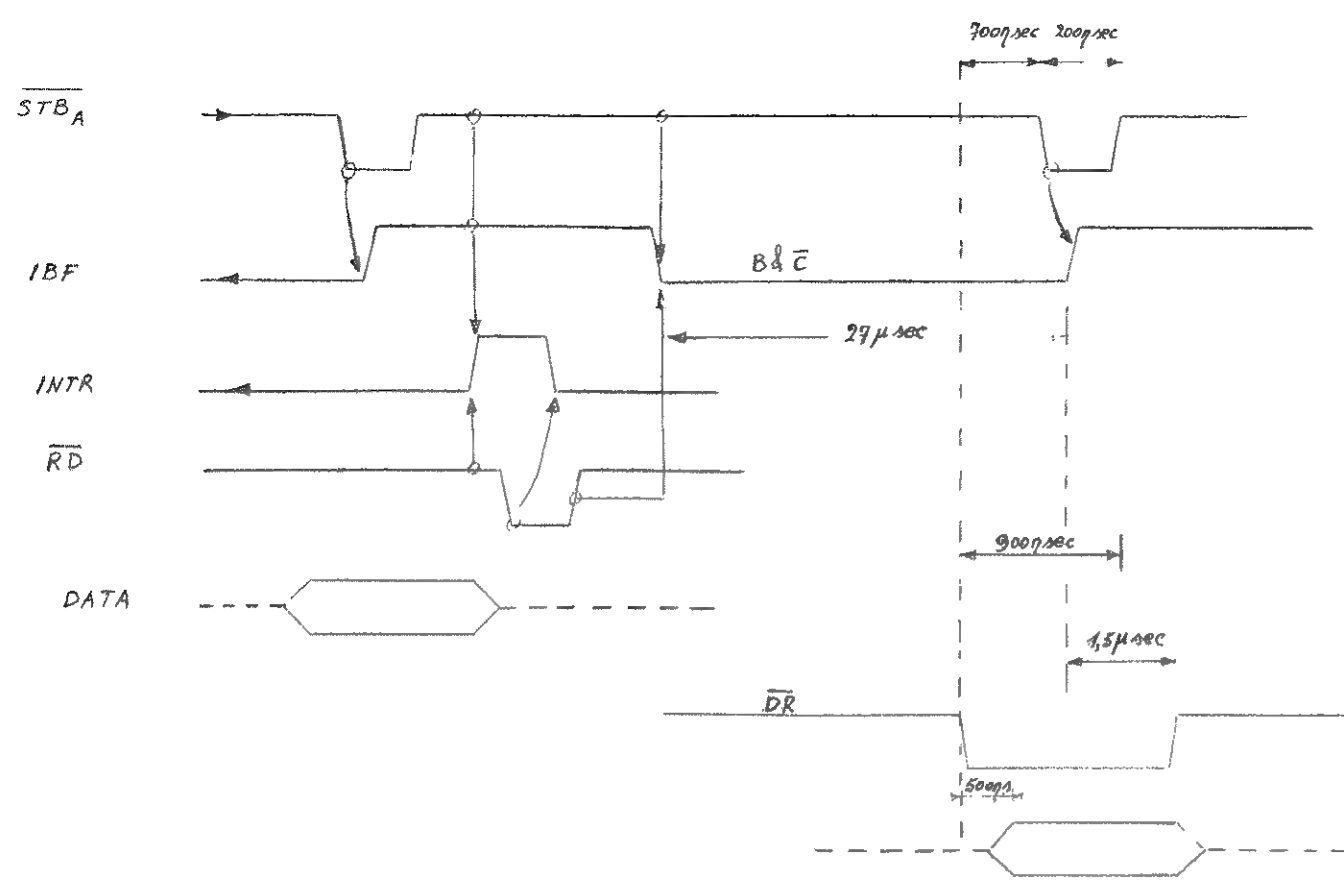


Fig. 2

```

10  MODE 0:PRINT CHR$(12):POKE #131,1
20  BEADD%=#570:REM ADRESSE DE DEPART
30  POKE #2F0,#70:POKE #2F1,#5:REM BEADD
40  CURSOR 9,21:PRINT "CONVERTISSEUR AD - 4 CANAUX 8 BITS"
50  CURSOR 9,20:PRINT "*****"
60  INPUT "NOMBRE DE CANAUX ";NR%:PRINT
70  IF NR%<1.0 OR NR%>4.0 GOTO 60
80  POKE #2F6,NR%
90  INPUT "NOMBRE D'ECHANTILLONS PAR CANAL ";T%:PRINT
100 COUNT%=T%*NR%:IF COUNT%>22160 GOTO 90:REM HEAP=5000
110 PRINT "NOMBRE DE CONVERSIONS =":COUNT%
120 ENADD%=BEADD%+COUNT%-1
130 PRINT "ADRESSE DE DEPART= #":HEX$(BEADD%)
140 PRINT "ADRESSE FINALE= #":HEX$(ENADD%)
150 V1%=PEEK (VARPTR (ENADD%)+3)
160 V2%=PEEK (VARPTR (ENADD%)+2)
170 POKE #2F2,V1%
180 POKE #2F3,V2%
190 CALLM #41C
200 COLORG 0 3 12 13
210 MODE 6A
220 CALLM #41B
230 IF GETC=32 GOTO 250
240 GOTO 210
250 CALLM #414
260 GOSUB 550
270 INPUT "NOM DE L'ENREGISTREMENT SUR CAS ";NOM$:PRINT
280 ENADD%=HEX$(ENADD%):BEADD%=HEX$(BEADD%):NR%=HEX$(NR%)
290 F1%=BEADD%+"-"+ENADD%+"":NR%
300 A%=NOM%+" "+F1%:A%=VARPTR (A%):POKE #2F7,PEEK (A%)
310 POKE #2F8,PEEK (A%+1)
320 IF ASC (NOM%)=0 THEN MODE 0:GOTO 10
330 CALLM #F000:REM DCR1
340 CALLM #410
350 GOSUB 550
360 PRINT "GRAPHIQUE 0/N ?":PRINT
370 G%=GETC:IF G%=0 GOTO 370
380 IF G%<>79 GOTO 10
390 MODE 6A
400 SP=335.0/T%:X=0.0:XCR%=335
410 FOR I%=BEADD% TO ENADD% STEP NR%
420 IF XCR%=INT (X) GOTO 470
430 FOR J%=0 TO NR%-1
440 C%=J%:IF J%=3 THEN C%=0
450 DOT X,PEEK (I%+J%) 21+C%
460 NEXT J%
470 XCR%=INT (X):X=X+SP
480 NEXT I%
490 GOSUB 550
500 PRINT "COPIE DE L'ECRAN 0/N ?"
510 G%=GETC:IF G%=0 GOTO 510
520 IF G%<>79 THEN END
530 POKE #131,0:COLORG 15 3 12 13
540 CALLM #300
550 ENVELOPE 0 15
560 SOUND 0 0 15 1 FREQ(660.0):SOUND 1 0 15 1 FREQ(651.0)
570 WAIT TIME 15:SOUND OFF
580 RETURN
590 END

```

POWER SUPPLY

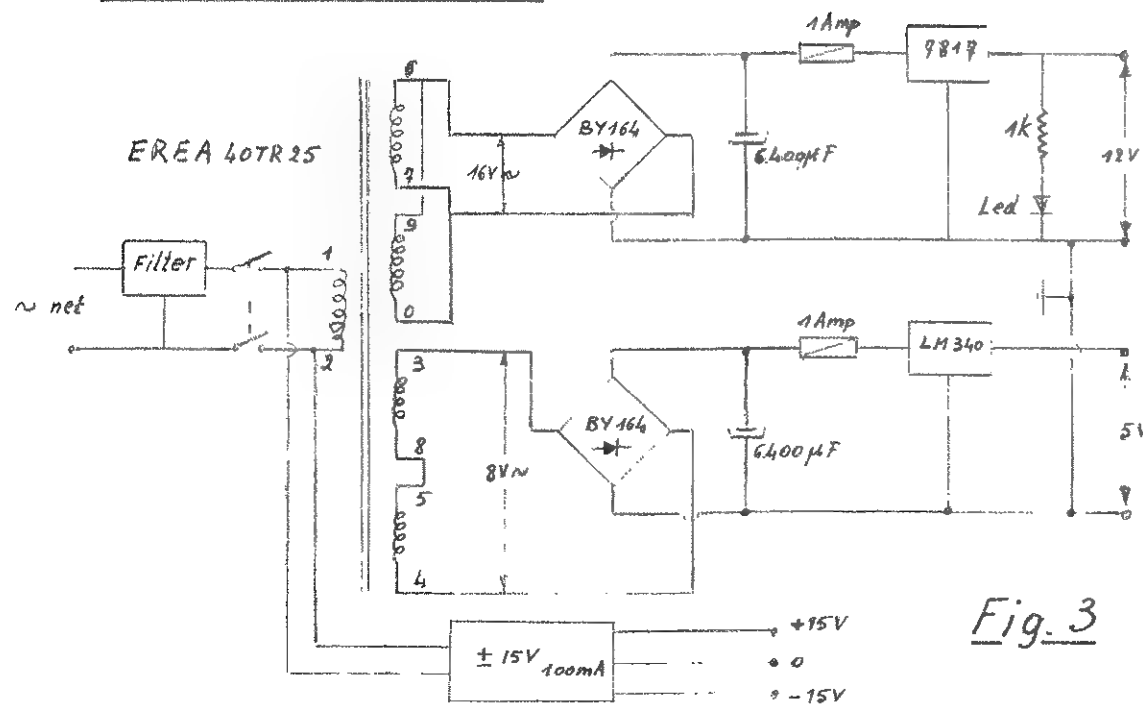


Fig. 3

```

001 *****
002 *AD CONVERTOR*
003 *****
004          DRG      :29B
005 029B 005C      DBL      :5C00
006          DRG      :410
007          BEADD    EQU      :2F0
008          ENADD    EQU      :2F2
009          CTADD    EQU      :2F4
010          CHANUM   EQU      :2F6
011          SYMADA   EQU      :2F7
012          SCANAD   EQU      :2F9
013          XCOORD   EQU      :2FB
014 0410 C5        PUSH     B
015 0411 C32805    JMP      WFILEA
016 0414 C5        PUSH     B
017 0415 C3F004    JMP      PROGRA
018 0418 C5        PUSH     B
019 0419 C32C04    JMP      INFSCR
020          *
021          *INIT
022          *
023 041C C5        PUSH     B
024 041D 2AF002    LHL     BEADD
025 0420 22F402    SHLD    CTADD
026 0423 1613      MVI     D, :13
027 0425 1EBA      MVI     E, :BA
028 0427 CDC8D8    CALL    :D8C8      OUTPUT COMMAND REG.
029 042A C1        POP      B
030 042B C9        RET
031          *
032          *INPUT-SCREEN
033          *
034 042C F3        INFSCR   DI
035 042D 210000    LXI     H, :00
036 0430 22FB02    SHLD    XCOORD      SET X
037 0433 216805    NEXTSC  LXI     H, SCRMEM
038 0436 22F902    SHLD    SCANAD
039 0439 AF        XRA     A
040 043A 4F        MOV     C, A          INIT CHANNEL 0
041 043B 1612      CONT1   MVI     D, :12
042 043D 5F        MOV     E, A
043 043E CDC8D8    CALL    :D8C8      SELECT CHANNEL AND CONVERT
044 0441 1610      MVI     D, :10      INPUT PA
045 0443 CDE0D8    CALL    :D8E0
046 0446 7B        MOV     A, E
047 0447 2AF902    LHL     SCANAD
048 044A 77        MOV     M, A          RESULT IN SCRMEM
049 044B 23        INX     H
050 044C 22F902    SHLD    SCANAD
051 044F CD1F05    CALL    CHTEST      CHAN. COUNTER IN REG. B
052 0452 C23B04    JNZ     CONT1
053 0455 216805    LXI     H, SCRMEM

```

```

054 0458 22F902    SHLD    SCANAD      RESET COUNTER
055 045B CD7304    CALL    DOTSLR
056 045E 2AFB02    LHL     XCOORD
057 0461 23        INX     H
058 0462 22FB02    SHLD    XCOORD
059 0465 7C        MOV     A, H
060 0466 3D        DCR     A
061 0467 C23304    JNZ     NEXTSC
062 046A 7D        MOV     A, L
063 046B FE4F      CPI     :4F
064 046D C23304    JNZ     NEXTSC
065 0470 FB        EI
066 0471 C1        POP     B
067 0472 C9        RET              RETURN TO BASIC
068          *
069          *THE MEMORY'S POSITION IN MODE 6=#6644
070          *      "      "      MODE 6A=#75BC
071          *      "      "      +(Y*#5A)-(X/4)IAND #7E
072          *
073 0473 AF        DOTSCR  XRA     A
074 0474 4F        MOV     C, A          RESET CHAN. COUNTER REG C
075 0475 2AF902    CONT2   LHL     SCANAD      POINTER SCRMEM
076 0478 7E        MOV     A, M          VALUE Y IN A
077 0479 115A00    LXI     D, :5A        90 BYTES/LINE
078 047C 210000    LXI     H, :00
079 047F 37        BCL2    STC
080 0480 3F        CMC
081 0481 1F        RAR
082 0482 D28604    JNC     BCL1
083 0485 19        DAD     D
084 0486 B7        BCL1    ORA     A
085 0487 CA9004    JZ      JPR1
086 048A EB        XCHG
087 048B 29        DAD     H
088 048C EB        XCHG
089 048D C37F04    JMP     BCL2
090 0490 11BC75    JPR1    LXI     D, :75BC      FIRST MEM. ADDR. IN MODE 6A
091 0493 19        DAD     D              HL=(Y*#5A)+#75BC
092 0494 0602      MVI     B, 2
093 0496 E5        PUSH    H
094 0497 2AFB02    LHL     XCOORD
095 049A 37        LOP1    STC
096 049B 3F        CMC
097 049C 7C        MOV     A, H
098 049D 1F        RAR
099 049E 67        MOV     H, A
100 049F 7D        MOV     A, L
101 04A0 1F        RAR
102 04A1 6F        MOV     L, A
103 04A2 05        DCR     B
104 04A3 C29A04    JNZ     LOP1
105 04A6 E67E      ANI     :7E           X=X SHR 2 IAND #7E
106 04A8 47        MOV     B, A
107 04A9 E1        POP     H
108 04AA 7D        MOV     A, L
109 04AB 90        SUB     B
110 04AC 6F        MOV     L, A
111 04AD 7C        MOV     A, H
112 04AE DE00      SBI     :00
113 04B0 67        MOV     H, A          POSITION OF MEMORY IN HL
114 04B1 3AFB02    LDA     XCOORD
115 04B4 E607      ANI     :07           POINT POSITION
116 04B6 47        MOV     B, A

```

```

117 04B7 04      INR   B
118 04B8 AF      XRA   A
119 04B9 37      STC                   SET CARRY=1
120 04BA 1F      FINPT  RAR
121 04BB 05      DCR   B
122 04BC C2BA04  JNZ   FINPT
123 04BF 47      MOV   B,A
124 04C0 AF      XRA   A
125 04C1 B9      CMP   C
126 04C2 CAD304  JZ    PLO12          (COL.REG.1)
127 04C5 3C      INR   A
128 04C6 B9      CMP   C
129 04C7 C2CE04  JNZ   PLO3
130 04CA 23      INX   H
131 04CB C3D204  JMP   PLO12          MEMORY ADD. +1 (COL.REG.2)
132 04CE 7E      PLO3  MOV   A,M
133 04CF B0      ORA   B
134 04D0 77      MOV   M,A           PLOT
135 04D1 23      INX   H
136 04D2 7E      PLO12 MOV   A,M
137 04D3 B0      ORA   B
138 04D4 77      MOV   M,A           PLOT POINT (COL.REG 1-2-3)
139 04D5 2AF902  LHLD  SCANAD
140 04D8 23      INX   H
141 04D9 22F902  SHLD  SCANAD
142 04DC CD1F05  CHT   CALL  CHTEST      DESTROY REG. A-E
143 04DF C27504  JNZ   CONT2
144 04E2 C9      RET
145              *
146 04E3 F5      IBF   PUSH  PSW
147 04E4 1612    MVI   D,:12         SELECT PC
148 04E6 CDE0D8  WAIT  CALL  :D8E0      INPUT PC
149 04E9 7B      MOV   A,E
150 04EA 17      RAL                   BIT 7=1?
151 04EB D2E604  JNC   WAIT
152 04EE F1      POP   PSW
153 04EF C9      RET
154              *
155              *PROGRAM
156              *
157 04F0 F3      PROGRA DI
158 04F1 AF      XRA   A
159 04F2 4F      MOV   C,A           SELECT CHANNEL 0
160 04F3 CDE304  CONT  CALL  IBF
161 04F6 1612    CNT   MVI   D,:12
162 04F8 59      MOV   E,C           SELECT CHANNEL 0 1 2 3
163 04F9 CDC8D8  CALL  :D8C8         SELECT CHANNEL AND CONVERT
164 04FC 1610    MVI   D,:10         INPUT PA
165 04FE CDE0D8  CALL  :D8E0
166 0501 7B      MOV   A,E
167 0502 2AF402  LHLD  CTADD
168 0505 77      MOV   M,A           RESULT IN MEMORY
169 0506 23      INX   H
170 0507 22F402  SHLD  CTADD
171 050A CD1F05  CALL  CHTEST
172 050D C2F604  JNZ   CNT
173 0510 0E00    MVI   C,:0
174 0512 EB      XCHG
175 0513 2AF202  LHLD  ENADD
176 0516 CD14DE  CALL  :DE14         COMPARE HL WITH DE (HL-DE)
177 0519 D2F304  JNC   CONT
178 051C FB      EI
179 051D C1      POP   B
    
```

```

180 051E C9      RET                   RETURN TO BASIC
181              *
182 051F 3AF602  CHTEST LDA  CHANUM       COMPARE CHANUM WITH REG C
183 0522 5F      MOV   E,A
184 0523 79      MOV   A,L
185 0524 3C      INR   H
186 0525 BB      CMP   E
187 0526 4F      MOV   C,A
188 0527 C9      RET
189              *
190 0528 00      WFILEA NOP
191 0529 F3      SWITCH DI
192 052A 3A4000  LDA   :40           PUDRM
193 052D F5      PUSH  PSW
194 052E F6C0    ORI   :F0           BANK 3
195 0530 3206FD  STA  :FD06         PORD
196 0533 324000  STA  :40           PUDRM
197 0536 FB      EI
198 0537 2AF702  LHLD  SYMADA       ADD. IN HEAP
199 053A 113E01  JUMPB LXI  D,:13E  EBUF
200 053D 46      MOV   B,M
201 053E 04      INR   B
202 053F 7E      MOVEBU MOV  A,M
203 0540 12      STAX  D
204 0541 23      INX   H
205 0542 13      INX   D
206 0543 05      DCR   B
207 0544 C23F05  JNZ   MOVEBU
208 0547 0E00    MVI   C,:00
209 0549 3E31    MVI   A,:31
210 054B 215D05  LXI   H,RETADD
211 054E E5      PUSH  H
212 054F 2AF002  LHLD  BEADD
213 0552 E5      PUSH  H
214 0553 2AF202  LHLD  ENADD
215 0556 E5      PUSH  H
216 0557 213E01  LXI   H,:13E       EBUF
217 055A C3F8EE  JMP   :EEF8         FILE 1 TO DCR
218 055D F3      RETADD DI
219 055E F1      POP   PSW           OLD BANK
220 055F 324000  STA  :40           PUDRM
221 0562 3206FD  STA  :FD06         PORD
222 0565 FB      EI
223 0566 C1      POP   B
224 0567 C9      RET                   TO BASIC
225              *
226 0568 00000000 SCRMEM DATA 00,00,00,00
227              *
228 056C      END

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

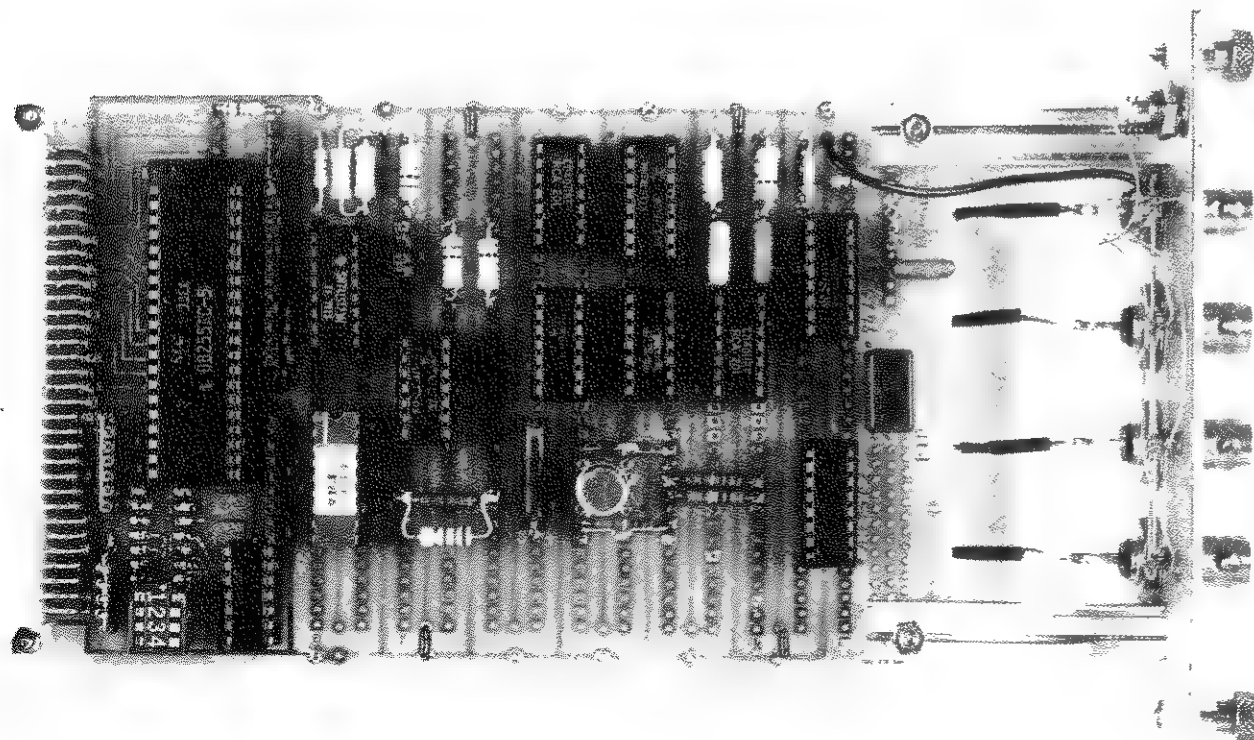
BCL1  0486  BCL2  047F  BEADD  02F0  CHANUM 02F6
CHT   04DC  CHTEST 051F  CNT    04F6  CONT   04F3
CONT1 043B  CONT2  0475  CTADD  02F4  DUTSCR 0473
ENADD 02F2  FINPT  04BA  IBF    04E3  INPSCR 042L
JPR1  0490  JUMPB  053A  LOP1  049A  MOVEBU 053F
NEXTSC 0433  PLO12  04D2  PLO3   04CE  PROGRA 04F0
RETADD 055D  SCANAD 02F9  SCRMEM 0568  SWITCH 0579
SYMADA 02F7  WAIT   04E6  WFILEA 0528  XCUURD 04FB
    
```

#P.
029H 00 5C

0410 C5 C3 28 05 L5 C2 F0 04 C5 C3 2C 04 C5 2A F0 02
0420 22 F4 02 16 17 1E BA CD CB DB C1 C9 F3 21 00 00
0430 22 FB 02 21 68 05 22 F9 02 AF 4F 16 12 5F CD CB
0440 DB 16 10 CD F0 DB 78 2A F9 02 77 23 22 F9 02 CD
0450 1F 05 C2 3B 04 21 68 05 22 F4 02 CD 73 04 2A FB
0460 02 23 22 FB 02 70 3D C2 33 04 7D FE 4F C2 33 04
0470 FB C1 09 AF 4F 2A F9 02 7E 11 5A 00 21 00 00 37
0480 3F 1F D2 8e 04 19 B7 CA 90 04 EB 29 EB C3 7F 04

0490 11 B0 75 19 06 02 E5 2A FB 02 37 3F 7C 1F 67 7D
04A0 1F 6F 05 C2 9A 04 E6 7E 47 E1 7D 90 6F 7C DE 00
04B0 67 3A FB 02 E6 07 47 04 AF 37 1F 05 C2 BA 04 47
04C0 AF B9 CA D2 04 3C B9 C2 CE 04 23 C3 D2 04 7E B0
04D0 77 23 7E B0 77 2A F9 02 23 22 F9 02 CD 1F 05 C2
04E0 75 04 C9 F5 16 12 CD E0 DB 7B 17 D2 E6 04 F1 C9
04F0 F3 AF 4F CD E3 04 16 12 59 CD CB DB 16 10 CD E0
0500 DB 7B 2A F4 02 77 23 22 F4 02 CD 1F 05 C2 F6 04

0510 0E 00 EB 2A F2 02 CD 14 DE D2 F3 04 FB C1 C9 3A
0520 F6 02 5F 79 3C BB 4F C9 00 F3 3A 40 00 F5 F6 C0
0530 32 06 FD 32 40 00 FB 2A F7 02 11 3E 01 46 04 7E
0540 12 23 13 05 C2 3F 05 0E 00 3E 31 21 5D 05 E5 2A
0550 F0 02 E5 2A F2 02 E5 21 3E 01 C3 FB EE F3 F1 32
0560 40 00 32 06 FD FB C1 C9 00 00 00 00



INTERFACE JOYSTICK POUR DAI

Une nouvelle interface destiné au DAI vient de voir le jour. Elle est munie d'une fiche DIN destiné au raccordement du DAI (Prise PDL) et d'une fiche 9 Points pour la fixation d'un JOYSTICK.

Pour ceux qui ne le saurait pas, la différence entre un PADDLE et un JOYSTICK est très importante: au niveau du toucher et de la précision.

Un Paddle contient des potentiometres à course progressive. Le Joystick lui donne les huit directions "cardinales" par un jeu de contacteurs. (Il faut avoir tenue les deux en mains pour sentir la différence.)

Les caractéristiques de l'interface sont impressionnantes lisez plutôt:

- 1) Elle est compatibles avec tout les JOYSTICKS de type 'ATARI'. Vous aurez donc un choix immense de ce côté.
- 2) Elle s'adapte très facilement sur le DAI. En effet, il n'y a aucune modification interne à effectuer. Vous brancher des deux cotés, et vous voilà aux commandes d'un engin de premiere classe.
- 3) Son utilisation dans vos programmes est SUPER-Simple !!! Pas de règles compliqués de conversions à effectuer. Ca marche tout seul !!!
- 4) Elle n'utilise qu'un seul port du DAI laissant libres les deux autres, de l'entré PADDLE ou elle est branché.
- 5) Un réglage fin à été prévue pour qu'elle s'adapte sans difficultés à toutes les révisions du DAI

Vous pourrez vous procurer cette interface au club DAInamic Pour l'instant en souscription uniquement. Elle devrait etre disponible à partir de fin Avril.

Interface DAI-Joystick
Souscriptions (Date limite :15/04/85)

196 Frs
169 Frs



A/D - D/A

Sehr geehrte Clubfreunde,

In DAInamic Nr.14 (Januar-Februar 1983) ist eine ADC-Schaltung von mir veröffentlicht worden. Nun habe ich eine verbesserte ADC-DAC-Schaltung die Analog-Digital-Conversion und Digital-Analog-Conversion durchführt und dabei DCE-bus kompatibel ist. Leider muß die ADC-Conversion über ein separates Maschinenprogramm durchgeführt werden weil das DCE-Bus Timing nicht genug Zeit für die ADC-Conversion läßt. Deshalb benutze ich ein Maschinenprogramm aus dem ROM-Listing (DCE-Bus Input-Routine) mit eingefügten NOP-Befehlen. Die DAC-Conversion wird mit dem Basic Befehl OUT Card-Adresse,X durchgeführt. Die ADC-Conversion wird mit dem Basic Befehl CALLM ... durchgeführt. Mit dem einfachen DCE-Bus Decoder kann man nun den ADC-DAC-Converter betreiben ohne das andere DCE-Bus Geräte gestört werden.

Mit freundlichen Grüßen
K.H. Kopp

```

5 REM ANALOG ZU DIGITAL
10 DATA #F5,#E5,#16,#66,#21,#03,#FE,#36,#90,#2B
20 DATA #36,#FE,#7A,#32,#01,#FE,#34,#00,#00,#00
30 DATA #00,#00,#36,#FB,#3A,#00,#FE,#5F,#32,#00
40 DATA #A0,#36,#FF,#35,#E1,#F1,#C9
50 FOR A=#A001 TO #A001 + 36
60 READ B%
70 POKE A,B%
80 NEXT A
90 POKE #A004,#66:REM POKE DIE CARD ADRESSE NACH #A004
100 CALLM #A001
110 PRINT PEEK(#A000)
120 GOTO 100
    
```

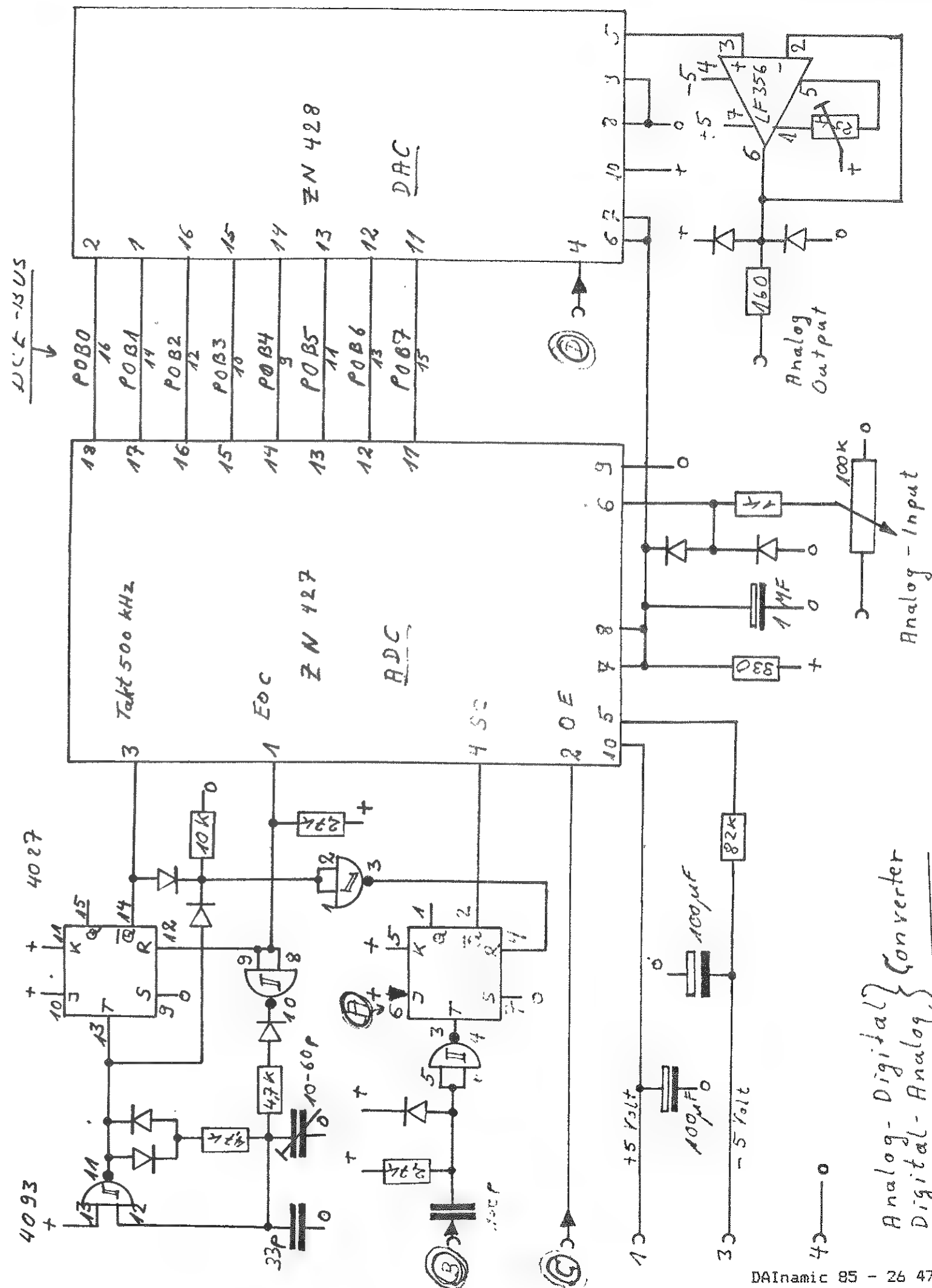
```

5 REM DIGITAL - ANALOGG - SAEGEZAHN
10 FOR A=0 TO 255
20 OUT #66,A:REM #66 = CARD ADRESSE
30 NEXT A
40 GOTO 10
    
```

Einfacher DCE-Bus Decoder

Mit den Codier-Schaltern wird die Card-Adresse eingestellt. An Punkt (A) sind bei richtiger Card-Adresse +5V. Mit diesem Signal wird das Start-Flip-Flop (PIN 6) freigegeben. An Punkt (B) ist immer das Bus Expand Signal vorhanden (Invertiert). Mit diesem signal wird das freigegebene Start-Flip-Flop gesetzt. An Punkt (C) ist bei richtiger Card-Adresse das Read-Signal. Mit diesem Signal wird der ADC-Ausgang aktiviert. An Punkt (D) ist bei richtiger Card-Adresse das Write-Signal. Mit diesem Signal wird der DAC-Eingang aktiviert.

In der Schaltung werden die Punkte
(A) mit (A)
(B) mit (B)
(C) mit (C)
(D) mit (D) verbunden.



Analog-Digital } Converter
Digital-Analog }

ROM I (111561φ) - I/O SELECT

z.B.

Adresse	Inhalt	Device	z.B.	
F9φφ φ1 φ2 φ3	FE	1	8255	Port A B C Control
φ4 φ5 φ6 φ7	FD	2	8255	A B C Control
φ8 φ9 φA φB	FB	3	8255	A B C Control
φC φD φE φF	F7	4	8253	Zähler φ 1 2 Steuerwort
1φ 11 12 13	EF	5	8253	Zähler φ 1 2 Steuerwort
14 15 16 17	DF	6	X	beliebig X
18 19 1A 1B	BF	7	X	X
1C 1D 1E F91F	FF FF FF 7F	gesperrt 8	Enable für Latch (74LS373)	

ROM II (111561φ) - Bank Routine

Adresse	Inhalt
F92φ	21
21	1F
22	F9
23	7φ
24	E1
25	E9
26	FF
27	φφ
28	φφ
29	φφ
2A	φφ
...	...
3D	φφ
3E	φφ
3F	φφ

} LXI H, F91F
} MOV M, B
} POP H
} PCHL

Ausstieg aus einer Bank:

- 1) nur mit einem Return C9 z.B (BASIC)
- 2) mit Umschaltung und Übergabe der neuen Adresse;



LXI H, XXXX Adresse nach dem Umschalten
 PUSH H Retten der Adresse
 MVI B, xx Neue Bank (Steuerwort Bank Select)
 JMP F92φ Umschalt-Routine (ROM II)

Adr. F91F Latch Enable (9415373)

Steuerwort Bank Select

I/O Adr.	Steuerwort	Bank
F91F	50	0
-"-	51	1
-"-	52	2
-"-	53	3
-"-	54	4
-"-	55	5
-"-	56	6
F91F	57	7
	DIP-Schalter	

Mit den DIP-Schaltern am Eingang von IC⁸ (DM81LS97)
kann jene Bank ausgewählt werden, die nach einem RESET
aktiviert sein soll, ohne das ein Steuerwort eingeschrieben werden muß

Vergleichstypen:

für IM5610 6331 745288 (wie im DAI) } 32x8

für DM81LS97 74LS244 od. 74LS241 (II)

DAI DOS 1541 - OPERATING SYSTEM

A cheap diskdrive for the DAI

INTRODUCTION:

At the moment, Commodore markets a diskdrive VC1541 for a very moderate price. This diskdrive is an intelligent system with the entire operating system in the drive itself. Reason enough to see if it could be used for the DAI.

The VC1541 is not a particularly fast diskdrive. But the advantages of direct access to files, in combination with a sophisticated file management system, are interesting enough to make it a very useful external memory device for the DAI; especially considering its price.

After some months designing the hardware and developing the software to adapt the VC1541 to the DAI, the 'DAI DOS 1541' is ready. It consists of:

- A Commodore VC1541 diskdrive.
- An interface card to connect it to the DCE-bus.
- An EPROM/RAM card to be installed on the X-bus.

The DAI DOS 1541 is entirely compatible with the DAI BASIC V1.0 and V1.1 versions. No DAI RAM space is used.

As well as 4 VC1541 diskdrives, up to 4 Memocom DCR's can be operated via the DOS. A High Speed Dataloader and a parallel printer can be connected too. All these peripherals are 100% supported by the DAI DOS 1541.

The only modification to be made inside the DAI is one wire to connect the reset to the X-bus.

FEATURES:

The DAI commands are extended with the following commands:

- All DCR commands as given in the DCR manual.

- For the diskdrive:

FDD x : Selects drive 0-3.
FORMAT : Formats a diskette.
SCRATCH : Deletes a file from the diskette.
RENAME : Changes the name of a file on diskette.
COPY : Merges files.
VALID : Reorganizes space on a diskette.
DIR : Reads the directory. Via the cursor keys, a file can be loaded and started.
FVER : Verifies a file on the diskette.

Files are automatically opened and closed.

- Other commands:

CAS x : Selects audio cassette 1-2.
RDL x : Reads data loader EPROM 0-3.
USR : Jumps to a m.l. routine with start address stored in the DAI I/O vector area.
BOOT : Loads a m.l. routine and a BASIC program.
UBL : As 'BOOT', but for files with identical

names for both m.l. and BASIC part.

LNON : Activates AUTO-LINENUMBER function.
LNOFF : Disables auto-line number.
<tab> : Clears screen.
/C : Sets default cursor.
/D : Output to screen only.
/E : Input from edit buffer.
/F : Sets implicit floating point numbers.
/H : Displays I/O directions, ext. memory device
and impl. number type.
/I : Sets implicit integer numbers.
/M : Selects mode 0.
/P : Selects parallel printer.
/S : Selects serial printer.
/T : Sets default text colours.
/O : Combination of /C, <tab>, /I, /M, and /T.

All commands can be used in the direct mode as well as from BASIC programs.

The set of error messages is extended with additional error reports by the DOS and by the disk drive itself. Two 'on error goto' features are available.

A comprehensive user manual describes all the features in detail. Together with the MDCR manual and the VC1541 manual, all possibilities are covered.

HOW TO GET IT:

At this moment, we are in contact with the industry to see if they are interested in producing this DAI-DOS. If these contacts are not successful, we will produce the DAI-DOS ourselves.

As soon as we know the results, we will publish them in order to keep you informed.

January, 1985

Jan Boerrigter
Hein Kop
Henk Rison
Peter Wiegers

DAI DOS 1451

KEN-DOS Abschlußbericht

Seit meinem letzten KEN-DOS Bericht (im Mai), hat die Software abermals einige Änderungen erfahren, aber nun ist sie endlich wirklich fertig, mein Rechner ist auch repariert, und KEN-DOS läuft seit einigen Wochen wieder bei mir. Die Probleme, die ich früher hatte, scheinen auch wirklich jetzt gelöst zu sein, und ich bin eigentlich sehr zufrieden mit dem System. Hier noch ein kurzer Überblick über die wesentlichen Neuerungen:

Es gibt zwei neue Befehle, um Dateien von einer Diskette zu einer anderen zu überschreiben: COPY kopiert eine einzelne Datei, und COMFAC kopiert *alle* Dateien, die kein *Delete*vermerk tragen (die Anbringung dieses Vermerks, eigentlich eine Vorstufe zum Löschen der Datei, kann mit dem RESTOR Befehl wieder rückgängig gemacht werden). In der allerletzten DOS-Version können *Delete*vermerke bequem aus der Directory angebracht werden, und RESTOR kann *alle* Dateien auf einmal wieder retten, so daß COMFAC doch eine sehr bequeme Möglichkeit bietet, mehrere, aber nur bestimmte Dateien zu kopieren. Im Gegensatz zum schon vorhandenen Befehl BACKUP (der eine genaue sektorenweise Kopie der Quelldiskette erstellt und den ursprünglichen Inhalt der Zieldiskette zerstört), kopieren die neuen Befehle dateienweise und fügen die kopierten Dateien hinten an, ohne etwas zu überschreiben. Sie können also benutzt werden, um eine Diskettensammlung umzuorganisieren, während BACKUP mehr zur Erstellung von Sicherheitskopien gedacht ist.

Die Möglichkeiten, DOS-Befehle in Programmen aufzurufen, sind wesentlich verbessert worden. Wie schon im letzten Bericht beschrieben, können Maschinenspracheprogramme das DOS ansprechen durch ein CALL :FOO6 mit Übergabe der nötigen Befehlsparameter in den CPU-Registern. In BASIC-Programmen kann das DOS auf zwei Weisen angesprochen werden: wie im DCR-TOS durch die Anweisung CALLM #FOO0; REM Dos-Befehl (Nachteil: alle Befehlsparameter müssen schon bei Programm-erstellung feststehen), oder, ähnlich zum Indata-DOS, durch die Folge: A\$="Dos-Befehl":PRINT A\$;CALLM #FOO3. Bei der zweiten Methode kann A\$ vom Programm errechnet und somit variabel sein. Allerdings, im Gegensatz zum Indata-DOS wird A\$ wirklich auf dem Bildschirm ausgegeben, ein Nachteil wenn man etwa Bilder abspeichern will. Das kann aber mit Hilfe eines POKE #135,1 (input from string!) umgangen werden-- vorher muß man die Adresse vom ersten Zeichen von A\$ bei #132-133 abspeichern und die Länge von A\$ bei #134; dann kann das PRINT entfallen.

Um Programme zu laden und zu starten (auch Maschinenspracheprogramme, da man ihnen jetzt eine Startadresse zuordnen kann) reicht es jetzt, den Programmnamen (auch wie üblich abgekürzt) einzutippen und abzuschließen mit *cursor-runter*, RETURN. Es gibt auch eine Autostartmöglichkeit: beim Einschalten oder Reset wird vom Laufwerk 0 ein etwa vorhandenes Programm, dessen Namen mit "AUTOEXEC..." beginnt, automatisch geladen und gestartet.

Es gibt jetzt einen Befehl (MANUAL), mit dem man einzelne Sektoren lesen und schreiben kann.

Die KEN-DOS Fehlerbehandlungsprogramme laufen jetzt über zugängliche und dokumentierte Vektoren im KEN-DOS RAM, so daß ein Maschinenprogramm sie jetzt abfangen und durch eine eigene Fehlerbehandlung ersetzen kann. Das ist sehr nützlich um zu vermeiden, daß kleine Fehler etwa bei der Eingabe eines Dateinamens zu einem Programmabbruch und Rückkehr zu BASIC führen müssen.

Schließlich ist die Betriebsanleitung ergänzt und verbessert worden. Die Beschreibung der Befehle ist zwar genauso knapp wie bisher (und ein neuer Befehl, COMP, ist überhaupt nicht erläutert!), aber die Schnittstellen zum DOS, die Speicherbelegung, und alle wichtigen Adressen im DOS sind genau beschrieben, so daß man doch sehr gut mit dem System zurecht kommen kann.

VC-1541 for the DAI

Insgesamt finde ich das nun fertige DOS sehr leistungsfähig und äußerst bequem, wobei mir besonders gefallen: die enorme Geschwindigkeit, der freie Zugriff zu den Dateien (ohne sie immer öffnen und schließen zu müssen, wie bei anderen DOS), und die Möglichkeit, viele Operationen mit nur einer Taste aufzurufen (z.B. das Laden von Programmen aus der Directory), so daß man oft noch nicht einmal Dateinamen eintippen muß!. Das einzige, was mir an der Software wirklich fehlt, ist, daß es keine gute (wenigstens keine dokumentierte) Möglichkeit gibt, die in der Directory gespeicherten Daten einem Programm zugänglich zu machen (etwa um eigene DOS-Nutzprogramme schreiben zu können).

Auch die Hardware scheint jetzt (endlich!) sehr gut zu funktionieren, obwohl ich früher sehr große Schwierigkeiten damit hatte. Als mein Rechner repariert war, konnte ich sogar eine Zeitlang keine Disketten formatieren, und Disketten lesen nur unter ständiger Justierung der Laufgeschwindigkeit. Danach hat Herr Gooswit den Controller neu justiert und einige kleine bauliche Veränderungen auf der Controllerkarte und der Epromkarte vorgenommen, und jetzt läuft das System besser (und sogar ein wenig schneller) als je zuvor. Ich habe vor allem keinerlei Rechnerausstiege mehr, und das System ist sehr zuverlässig (in mehreren Wochen Betrieb höchstens zwei oder drei Mal einen Schreib- oder Lesefehler). Allerdings bezieht sich diese Zuverlässigkeit nur auf den Betrieb mit ein und demselben System. Es können sehr wohl Probleme auftreten beim Austausch von Disketten zwischen verschiedenen Systemen: ein Kollege von mir, der auch KEN-DOS hat, kann von mir geschriebene Disketten nicht lesen! Vielleicht liegt es daran, daß seine Laufwerke von einer anderen Marke sind, denn ich habe keine Verträglichkeitsprobleme zwischen meinen eigenen beiden Laufwerken. Übrigens hat mein Kollege, genau wie ich, am Anfang überhaupt große Probleme gehabt mit seinem System, die erst behoben wurden, als Herr Gooswit ihm einen persönlichen Besuch abstattete. Jetzt scheint er aber auch voll zufrieden zu sein mit dem System. Ich sollte auch fairerweise betonen, daß Herr Gooswit voll hinter seinem Erzeugnis steht, und sich sehr große Mühe gibt, eventuell auftretende Probleme zu beheben. Man braucht also wegen Garantieansprüchen keine Bedenken zu haben.

Die Zuverlässigkeit meines Systems liegt sicher zum Teil an Herrn Gooswits Lötarbeiten auf den DOS-Platinen und auch auf der DAI-Platine! Aber die jetzt zum Verkauf kommenden DOS-Karten sind ja schon von vornherein verbessert, und die empfohlenen Änderungen auf der DAI-Platine (die zum Teil die Störanfälligkeit verringern sollen) sind zwar etwas umfangreicher als nur das Einlöten eines einzigen Drahts, sind aber sehr genau beschrieben in der neuen Bedienungsanleitung. Nur eine Warnung dazu: Herr Gooswit empfiehlt eine Änderung auf der Stack Overflow Leitung, und hat sie bei mir auch durchgeführt, aber als mein Gerät neulich bei Indata repariert wurde, wurde die Änderung rückgängig gemacht (im Reparaturbericht stand: 'Falsche Änderung der Stack Leitung durch Kunden'), und ich betreibe mein Gerät seitdem im Indata-Zustand ohne jegliche Probleme. Diese Änderung würde ich also nicht durchführen!.

Aber KEN-DOS würde ich ohne Einschränkung wieder kaufen.

Bochum, den 27. August 1984

Gordon Wassermann

Nowadays the cheapest floppy-disk system is the VC-1541 for the Commodore 64. It has a 16K byte operating system of its own (CBM DOS V2.6) and a 2K byte RAM. The drive uses 5 1/4 inch diskettes with a memory capacity of 170K/disk. The datatransfer is done through a serial IEC-bus with only 5 connections. The interface consists of 3 TTL-invertors with open collectors (7406 etc.) and a bit of software. The software resides in an EPROM on the X-bus, like the TOS of the MDCR.

The serial bus:

Because of the serial way of working of the bus the number of lines is only 5, of wich 2 are bidirectional. By the ATN-signal (attention) the buscontroller (in this case the DAI) tells the peripheral that the actually transmitted bytes are no data but a command. Transmission of data- and commandbytes is done using CLK (clock) and DAT (data). Those two lines are bidirectional and are moreover used for handshaking. The two other lines are mass and reset. Further information on the bussignals can be found in an article in the may issue of '64-er'.

The speed:

The serial way of working and the fact that the bus is widely in software are major slowdowns in data-transfer. The speed is only about 3000 baud. This is ridiculously small for a disk system, but still much faster as a cassette-recorder, even not considering access-time.

A small comparison of accesstime:

device	I	10K file	I	price/10K	I	form of commands	I	system price
audio-	I	ca. 170s	I	ca. 2Bfr	I		I	ca. 2000Bfr
cassette	I	C60 cas.	I		I		I	
MDCR	I	ca. 43s	I	ca. 20Bfr	I	CALLM#F000:REM com.	I	ca.15000Bfr
KEN-DOS	I	< 3s	I	ca. 8Bfr	I	idem	I	ca.50000Bfr
DAI-floppy	I	ca. 13s	I	ca. 16Bfr	I	POKE :?"com.": POKE	I	
VC-1541	I	ca. 27s	I	ca. 9Bfr	I	real BASIC-commands	I	ca.18000Bfr

The hardware:

Because of the small amount of lines one needs few parts to connect the drive to the DAI. For the bus one only needs 1 IC. Commodore uses the 7406. Because this IC costs at least 100Bfr nowadays it is advisable to use the cheaper 7401. The 7401 contains four NAND with open collectors instead of six invertors like the 7406. One of the two NAND-gates must be connected to 5V over a resistor of 1KOhm. One can therefor easily use resistor R4, wich is already used for such a purpose. Two other IC's (74LS00 and 74LS293) make the software autostart when switching on the DAI or pressing RESET. Instead of the 74LS293 one can use other counters, if only they count from 0 to 4. Of course one must be aware of the other pin connections.

Important notice: This interface can't be used together with the MDCR or a parallel printer.

The software of the Commodore floppy:

The software resides in an EPROM on address #F000-#FFFF as with MDCR. The software is no DOS because the VC1541 has a 16Kbyte DOS ROM and a 2Kbyte RAM. Anyway there are commands necessary (except from LOAD/SAVE) to drive the DOS. These commands are built up in a NCU-basis (New Command Unit), this means that the new commands are of equal value as the standard commands and can be used in BASIC-programs.

This description is in the first place to explain the new commands, because the possibilities of the DOS (sequential, relative and random-access files, block reading and writing, etc.) are fully described in the manual. There is also a "Floppy-book". Some tricks and programs will be explained later.

The software with the NCU-driven commands (It is only a minni-NCU vesion, that contains only the most important commands.) is immediately ready for use because of the RESET-logic connected to the DCE-bus. With RESET the NCU is initialized, the "DISK"-command is executed, the baud-rate is set (only interesting for people with a printer on 9600 baud with 1 stop-bit), and the "return to BASIC"-address is set. (By changing this it is possible to trap disk-errors.). The return-address must be put in #00CE/CF.

The new commands are partially copied from Commodore (and up to INPUT# even better), partially also they are completely new, to simplify working with the DOS.

CLS	OPEN	INPUT#	TEXT#
DISK	CLOSE	GET#	
CAS	STATUS	MGET#	
DIR	ON.EOF	PRINT# (shorter: P#)	
GMP	COM	PUT#	
IF		MPUT#	

a) CLS : CLear Screen

b) DISK : switches the LOAD/SAVE vectors to disk-use and sets the standard primary address to 8. This primary address is a device-number fixed by hardware, with which the different peripherals are distinguished. With other numbers two drives or a Commodore printer can be attached and addressed. This initialisation of the primary address makes it superfluous to indicate it with LOAD/SAVE etc., as described in the manual. Programs, UT-files or data are always read from the device whose number is in #00C9. The "DISK"-command also clears the ON EOF setting.

c) CAS : switches the vectors back to cassette.

d) DIR : lists the directory. Contrary to Commodore the directory must not be loaded as a BASIC-program and then listed, but will be immediately printed. The directoy list can be stopped with the space-bar or BREAK as a BASIC listing. The possibility given by Commodore to list only specific names, is used by adding a string to the command.

DIR "SG*" (everything that starts with SG. The "*" is a wildcard.)

The DIR-command gets the directory from the device with the standard primary address (default=8).

e) GMP : Go Machine language Program. Is an abbreviation of "CALLM#400", as most utilities start at #400.

f) IF : IF is no new command, but there is a new interpretation. In "IF..THEN" the THEN is superfluous. Of course the commands are only useable with the disk NCU in memory. Passing on programs with the new commands (except IF) to non-floppy users is not possible without a MLP-supplement. This can't be a real problem because they are all pure disk-commands except from CLS, GMP, PRINT#0, PUT#0,...

g) OPEN : The OPEN-command has similar to Commodore the following syntax :
OPEN channel PrimAddr SecAddr
OPEN channel PrimAddr SecAddr "..."

The meaning of the secondary address is explained in the manual (one can work with up to 6 different files at the time). The channelnumber is the number with which one indicates the wanted device/file in the commands mentioned in the third column. With OPEN the primary and secondary address is set and is not necessary anymore afterwards (except for the block write an read commands, see manual). In contrast with Commodore the channelnumber must be between 1 and 6 inclusive. This means one can work at 6 files at the time (furthermore still LOAD/SAVE and DIRectory). Each opened channel must be closed again with "CLOSE#". If not, because e.g. there was a BASIC-error, the floppy will show this with a LED (except from SecAddr=15 and STATUS). When trying to open the same channel again, "INVALID NUMBER" is printed and the old channel is closed for security reasons. With "OPEN" one can choose the PrimAddr and the SecAddr between 0-15. If that device does not exist or is not switched on (also when with LOAD/SAVE PrimAddr=8), the error-message "INVALID DEVICE" is given. (Commodore: "DEVICE NOT PRESENT").

h) CLOSE : with "CLOSE channel" the opened channels can be closed again. As described in the manual, the closing of the SecAddr 15 means closing all files in the floppy. For this reason 'commandchannel 15' must be closed last. This feature uses "CLOSE 0" : CLOSE 0 closes all channels in the computer and all channels in the standard PrimAddr device. So if all channels are opened to the floppy, then CLOSE 0 will do. As the other channel-depending commands, CLOSE of a channel which isn't opened will give a "INVALID NUMBER" error.

j) STATUS : is an abbreviation of the in the manual described procedure to get a previous errormessage printed. "STATUS" the errormessage (e.g. OK) on the screen and/or the printer.

k) ON.EOF : (and ON.EOF ln) are a possibility to pass control to a specific BASIC-line when reaching the end of file while reading. With Commodore a flag must be tested (on #0007), but here the INPUT#, GET#, MGET# or TEXT# command is automatically stopped and a GOTO is performed. This function must be switched off again with ON.EOF without a linenummer (also the "DISK"-command does this), because this is not automatically performed with the mini-NCU-version. If the linenummer does not exist or if the ON.EOF function is switched off (in other words, if one tries to read beyond the EOF), the BASIC-error "OUT OF DATA" is printed and the channel is closed. With "ON EOF ln" the channel remains open.

l) COM "c" : is a feature to skip the complicated Commodore method to pass easy commands like formatting, erase, rename, etc. to the floppy. With Commodore the command-SecAddr must be opened (OPEN x 8 15), the command printed, and the channel closed again. For more complicated commands this is still necessary, but commands that must be given 'once quickly', are a lot easier with COM.

m) TEXT# ch

With this command bytes can be read from the channel and immediately printed (depending on #0131). Printing is stopped when a CHR#(13) is reached. With this command one can quickly look wether a textfile is OK.

n) INPUT# ch;vars

This will perform an INPUT of numbers and strings from disk. The error "SOME INPUT IGNORED" is suppressed and when not enough items are entered they are loaded further automatically. With "SYNTAX ERROR" the program is stopped, the channels remain open. The disadvantage of this 'over the screen' method is that the errorchannel can't be read as easily as discribed in the manual (only usefull with GET#). The advantage : One can perform a normal INPUT with the keyboard with "INPUT# 0;..." and one can this way choose to give the data direct or from disk. In both cases one can see the data. Also I want to mention here that numbers can be saved much more secure and less place occupying with MPUT#/MGET#.

o) GET# ch;vars,...

Per variable one byte is read and presented as a number 0-255 or 1 letter when using stringvariables

p) MGET# ch,addr number

With MGET# 'number' bytes are read and stored from 'addr' on. When reaching EOF during reading, the number of bytes not read is put in #0048/49. This command is easy to read a large amount of data at the time because it is much faster as single GET#'s, in particular with block-reading and writing.

q) PRINT# ch;... (or P#) (data as with normal print)

The data (numbers and strings) are passed to the opened channel and put on disk. Options : positive numbers are printed without a space in front, a comma between the data means that a comma is printed and as a bonus : with channel 0 the data are not printed to disk but on the screen. So the 'without space' and the comma option can also be used on the screen.

r) PUT# ch;...

Put# prints strings as does the PRINT#-command, but numbers are seen as CHR#()-signs. Comma and semicolon are of no meaning.

s) MPUT# ch,Addr number

Writes 'number' memorybytes, from Addr on to, to disk. This is easy for large amounts of data like screens etc.

Notice:

The error 'TYPE x' (e.g. TYPE 1) occurs when filetypes do not match. I can forward the source-listing on request. The software is a coproduction by H.Steves (bus-control) and me (NCU,commands).

Heinrich Tegethoff
translation:
LMPG-soft.

MEMODAI

Où en est votre mémoire ? Le DAI vous offre la possibilité de l'entraîner avec ce nouveau logiciel :MEMODAI.

Le principe est simple : 27 paires de dessins à retrouver sur une grille de 9x6 . Vous n'en visionnez que deux de votre choix à la fois qui, s'ils ne constituent pas une paire, s'effaceront. Dans le cas contraire ceux ci resteront affichés sur la grille , vous marquerez un point et rejouerez !

Vous pourrez affronter le DAI (5 niveaux de jeu), jouer contre une personne de votre choix, ou laisser le DAI se battre contre lui même à son niveau maximum .

Pour vous familiariser avec les différents dessins, le DAI accepte, si vous lui demandez bien sur , de vous les présenter en musique .

Ne vous étonnez pas si le DAI répond INSTANTANEMENT: la totalité du graphisme (Mode 6), son affichage sur l'écran, la réflexion du jeu ont été traitées en langage machine .

Derniers détails: les dessins sont entièrement disposés au hasard à chaque partie et le DAI NE TRICHE JAMAIS !

Mémo-Dai



PS : Battre le DAI au niveau 5 , sans support écrit, relève de l'exploit !!! Les niveaux 1 et 2 restent par contre très accessibles .

80 CHARACTERS

Diese Schaltung habe ich über ein etwa 30 cm langes Kabel angeschlossen, es gibt dabei keine Probleme.

Der 330 Ohm Widerstand der im Dainamic Schaltplan eingezeichnet ist, sollte am Ende der Leitung bei IC 1 angebracht werden.

Außerdem sollten direkt an den Spannungs- und Masseanschlüssen der ICs einige Entkoppelkondensatoren (100 nF o.ä.) angebracht werden.

In der DAI-Hardware habe ich noch einen kleinen Fehler entdeckt. Bei 16-farbigen Zeichen wird die Hintergrundfarbe nicht richtig angezeigt. Die Farbe des alten Feldes wird auf das neue Feld ausgedehnt, wie es für Mode 1 etc. auch sinnvoll ist. Allerdings sollte dieser Effekt nur in den Grafikmodi auftreten und nicht wenn Zeichen dargestellt werden.

Dem läßt sich aber leicht abhelfen. Pin 11 und 12 von IC 24 werden voneinander getrennt und Pin 11 an Pin 15/IC 48 angeschlossen.

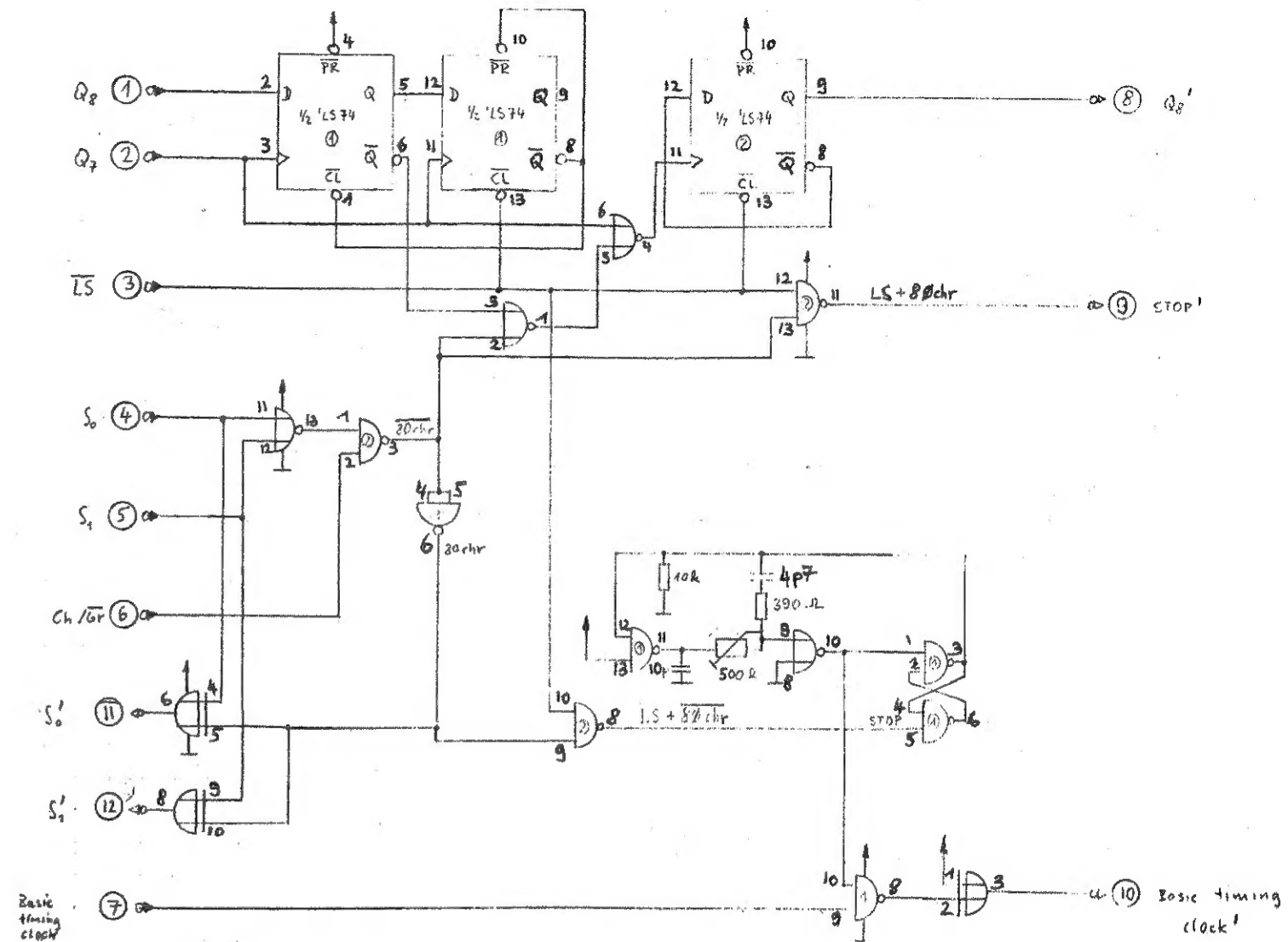
Mit dieser Änderung wird die neue Hintergrundfarbe sofort am Beginn eines Feldes angezeigt.

Ich hoffe, daß Sie mit meinen Anregungen etwas anfangen können und verbleibe

mit freundlichen Grüßen

Wolfgang Henning

3 Beilagen



- | | |
|------------|---------------|
| 2 x 74LS00 | 1 x 10 kΩ |
| 1 x 74LS02 | 1 x 390Ω |
| 2 x 74LS74 | 1 x 330Ω |
| 1 x 74LS86 | 1 x 500Ω trim |
| | 2 x 10 pF |

not used : $\frac{1}{2}$ LS74
 $\frac{1}{2}$ LS86

