ENTERface

Geerweg 2, 2611 VN Delft

DRUKWERK

PORT BETAALD
PORT PAYE
DELFT

JAN/FEB 89

ENTERface

INHOUDSOPGAVE

GEBRUIKERSDAG—ALGEMEEN 28 JANUARI

H.F. Witte dorpshuis, Henri Dunantplein 4, DE BILT
bus 57 vanaf CS, halte Nobellaan, Nobellaan in, 3e weg links

COLOFON

De *ENTERFACE* is het officiele orgaan van de *Dutch Enterprise User Group* en verschijnd eens per twee maanden.

REDAKTIE ..... Robin Ketelaars . *voor toesturen kopij* .....
Geerweg 2, 2611 VN DELFT, 015-126422

MEDEWERKENDEN .... Fons, Stan, Spock, Hans, Arjan .....
.....altijd maar dezelfden die schrijven!..

VOORPLAAT ..... Geen artistieke kopij, maar de adressering zie je hier

KOPIJ ..... Inzenden op DISK: of TAPE:
ZIE VOORWAARDEN IN HET APRIL/MEI NUMMER
72 tekens * 50 regels TAB's onder L(oad) en E(xit)
1e regel = titel die bovenaan bladzijde komt
Centreren titel niet nodig, gaat vanzelf, max 36 tekens
<ALT L> voor de zin geeft nieuwe titel & nieuwe bladzij
<ALT \> voor de zin geeft alleen een titel
PRINT met F3 <naam>.KPY voor automatische verwerking

OVERNEMEN ..... van de gehele of gedeeltelijke inhoud toegestaan

HAMER + PENNINGEN Stan Tuinder .... *voor geldkwesties en adviezen* .....
Willemstr 170, 2713 AJ ZOETERMEER, 079-169523

SECRETARIAAT ..... Fons Kraakman ... *voor aanmelden en opzeggen* .....
Bleekveld 8, 1852 JH HEILOO, 072-335131

SOFTWARE ..... Dik Fennema ..... *voor aanvragen/toesturen software*
v Duivenvoordeln 30, 2241 ST WASSENAAR, 01751-17249

KONTAKTEN ..... NOORD: Kees 05945-15626; Peter 050-775850
MIDDEN: Peter 08885-02186; BRABANT: Rob 01651-1245
ZEELAND: Ben 01185-2525; ZUID: Rene&Jos 043-617623
WEST: Hans 02510-46630; Erik 01803-17051

OPBELLEN ..... BEL ALLEEN BOVENSTAANDE NUMMERS TUSSEN 20 EN 22 UUR

DRUKWERK ..... DRUK.TAN HECK, Pluympot 1a, 2611 LX DELFT, 015-142981

VAN DE SECRETARIS

Hoewel het de gewoonte is een verslag van de jaarvergadering pas te publiceren voor de volgende jaarvergadering wil ik een keer van deze "gulden" regel afwijken. Op de jaarvergadering zijn namelijk een aantal belangrijke besluiten genomen. Het zou voor de mensen die niet aanwezig waren jammer zijn als zij dit pas na een jaar zouden vernemen. Vandaar mijn afwijking. Tenslotte, het verslag dat hieronder volgt is niet volledig, hierin zijn alleen de belangrijkste besluiten opgenomen. Het volledige jaarverslag zal gepubliceerd worden in de "Enterface" die verschijnt voor de jaarvergadering van 1989.

Ik denk dat we kunnen spreken van een zeer geslaagde jaarvergadering. De deelname van de leden aan de jaarvergadering was groot. Men had opbouwende kritiek op voorstellen vanuit het bestuur, daarnaast kwam men zelf met goede voorstellen.

Het voorstel van de voorzitter om de contributie van f 45,-- naar f 35,-- te verlagen en in te teren op de reserves werd afgewezen. Men was voor handhaving van het huidige contributiepeil. Met het geld dat overblijft, kunnen dan nieuwe hardware ontwikkelingen ondersteund worden. Daarnaast kan met dat geld interessante hardware uit het buitenland voor de club aanschaft worden. Deze hardware wordt dan op de gebruikersdagen geshowd. De gebruikersdagen zullen dus nog interessanter worden.

Daarnaast wilde de ledenvergadering dat het bestuur meer doet aan de werving van leden. Men denkt dan aan gericht adverteren. Een klacht van een nieuw lid was namelijk dat het voor hem heel moeilijk was om de DEUG op te sporen. Het bestuur dient ook een "visje" uit te werpen onder de potentiele leden. Vastgesteld zal moeten worden wie er ooit een Enterprise gekocht heeft en nog geen lid is. Deze mensen moeten dan aangeschreven worden. Niet alleen werving was aan de orde, maar ook behoud van leden. Leden die hun contributie voor 1987 nog niet hadden voldaan, wilde men niet meteen royeren, deze leden zullen eerst worden aangeschreven.

Een verzoek uit de ledenvergadering aan de leden was om voor de "ENTER EN RETURN" rubriek van de "ENTERFACE" ook problemen in te sturen die een softwarematige oplossing behoeven. Binnen de gebruikersgroep zijn namelijk leden die wel willen programmeren, en dat ook heel goed kunnen, maar geen ideeen meer hebben. Dus mensen, zit u met een probleem, kom er maar mee op, schrijf ze maar van uw af !!
Nu zullen bepaalde problemen best wel eens zo interessant zijn dat meerdere mensen hieraan gaan werken. Om nu al te veel dubbel werk te voorkomen, is het het beste dat men in de "ENTERFACE" meldt dat men met een bepaald probleem bezig is.

Tot zover het "voorlopig" verslag van de jaarvergadering.

IS-DOS EN MACHINETAAL

And now for something completely different (1).

Inhakend op de artikelen die zijn verschenen over het werken met GEN wil ik een aantal opmerkingen plaatsen t.a.v. het schrijven van programma`s die bestemd zijn om onder IS-DOS te draaien.

1. Bij het assembleren (het omzetten van assembler naar machinetaal) dient in GEN de opdracht "A" gegeven te worden, gevolgd door een aantal parameters. Een hiervan is de "R" commando. Deze geeft aan om wat voor soort programma het gaat. Geef je A,,R 5 op, dan betekent dit dat je de file wilt assembleren en dat het een "gewoon" machinetaal programma moeten worden. Dat geeft die vijf aan. Het gaat hier dan bijvoorbeeld om spelletjes in machinetaal. Geef je de opdracht A,,R 7 dan betekent dit dat het om "systeem relocatable" machinetaal gaat. Dit zijn programma`s die in het geheugen blijven staan en steeds weer oproepbaar zijn via het ":" commando. Een programma van dit soort is bijvoorbeeld "De Screen dump and Load utility VSL", uit de Software Library. Bij het assembleren van een programma dat moet werken onder IS-DOS geef je het commando: A,,R 0.

2. In mijn IS-DOS handleiding staat een fout. Volgens de handleiding zou een "absolute sector read" geschieden via de functie 2EH. Echter dit moet zijn 2FH. Hetzelfde geldt voor een "absolute sector write". Volgens de handleiding zou dit moeten geschieden via ook weer functie 2EH. Dit moet echter functie 30H zijn.

3. Zou u wat willen "rotzooien" met de "absolute sector read/write calls" vergeet dan niet daarvoor een aparte schijf te gebruiken. Wijzigingen op de schijf zouden wel eens kunnen leiden tot het opnieuw moeten formatteren van een schijf. Daarnaast moet het "disk transfer address" aangepast worden. Het "disk transfer address" geeft aan vanaf welke plaats in het geheugen de data staat indien men een "write-call" uitvoert of vanaf welke plaats de data moet komen te staan indien men een "read-call" uitvoert. Als een programma een "absolute sector read call" doet, wordt een blok van 256 bytes ingelezen. Standaard wordt dit blok geplaatst vanaf adres 128 decimaal. Een machinetaal programma begint echter op adres 256 decimaal. Dit betekent dus dat een deel van het programma overschreven wordt. Als men deze functie een keer wilt uitproberen, dan zal het "Disk transfer address" een andere waarde moeten hebben. Dit geschiedt via functie 26 decimaal. In de IS-DOS manual staat vermeld welke registers hierbij gebruikt moeten worden.

De mensen die hier mee willen gaan knutselen wens ik veel succes en mocht men tegen iets interessants aan lopen, meldt dit dan even in de Enterface. De overige gebruikers kunnen hiermee dan ook hun voordeel doen.

Fons Kraakman

ENTERPRISE SPEAKEASY

Ooit heeft AZTEC voor Enterprise een Speakeasy spraaksynthesizer gemaakt. Deze was tot nu toe niet in Nederland te koop. Het bedrijf Werner Linder in Duitsland verkoopt echter alle originele Enterprise produkten (alle software, hardware zoals Enterprise printer, en onderdelen, zoals de NICK en DAVE chip, maar ook keyboardmembramen !). Ook de Speakeasy is hier te bestellen, wat ik dus gedaan heb.

De spraaksynthesizer is een klein kastje, en haalt zijn voeding van de Enterprise transformator. De uitgang van de trafo gaat in de Speakeasy, waar weer een snoertje aan zit dat in de Enterprise gaat. Het voordeel van deze methode (i.p.v. 5 volt aftappen van bijv. de joystickport) is dat de spanningregelaars (die tegen het koelblok zitten geschroefd) niet extra worden belast. Ik heb zelf, met een diskcontroller en een geheugenuitbreiding deze regelaars moeten vervangen door zwaardere van 2 A.

De Speakeasy wordt via een mooie stekker aangesloten op de printerport. Als je echter een andere paralelle port op je Enterprise hebt gemaakt zal die daar ook wel op werken.

Aan de binnenkant zie het ding er bijzonder simpel uit. Hij is opgebouwd rond de GI SP0256 AL2 spraaksynthese chip. In feite is dat vrijwel het enige onderdeel. Verder zit er ook een speaker in met een versterkertje, waaruit een heel wat beter geluid komt dan uit dat prulding in onze Enterprise. Heel leuk is dat je het geluid van de Enterprise met een snoertje naar de versterker van de Speakeasy kunt sturen. Minder leuk is, dat er geen audio uitgang opzit, waardoor je het geluid naar je monitor of een HI-FI installatie kunt sturen. Met een soldeerbout en een schroeven draaier is dat natuurlijk eenvoudig te verhelpen.

De chip is een wel vaker gebruikte chip, onder andere in een spraakmodule voor de Commodore "he gatsie" 64. Vreemd genoeg schijnt deze module de mogelijkheid tot klemtoon en een hoge en een lage stem te hebben, en heeft de Speakeasy dit niet.

De spraak wordt opgebouwd uit zogenaamde allofonen, of gewoon allerlei klinker- en medeklinkerklanken. Er zijn 58 klanken beschikbaar. Jammer genoeg zit onze prachtige g klank daar niet bij. Begrijpelijk, de Engelsen zouden denken dat het ding keelontsteking had als ze zo'n klank zouden horen. Ook de ij zit er niet bij, maar meestal is er wel een klank te vinden waardoor het woord toch te verstaan is. En als je zo'n klank niet vind, gebruik je gewoon een ander woord.

De spraak is, vooral in zinnen, vrij goed verstaanbaar. Losse woorden zijn soms wat moeilijker, maar als je zo'n woord eenmaal weet is het voortaan ook prima te verstaan.

Bij het apparaat zit ook wat software. Dit zijn alle BASIC extensions, die de functie SAY$ aan de BASIC toevoegen. Je kunt iets laten zeggen door bijvoorbeeld te typen: LPRINT SAY$("HELLO"); De functie SAY$ zet de letters om in de (soms) juiste allofonen. Er is een Engelse, een Duitse

en hoera hoera... een SPAANSE versie. De Engelse versie is nog het best, maar ook deze verslikt zich nogal snel in de woorden, zodat je toch snel vervalt in het opbouwen van de woorden door het zelf achter elkaar zetten van de juiste klanken. Als je een nederlandse tekst wilt laten uitspreken moet je dit sowieso doen. Echt erg is dit niet, omdat dit vrij snel gaat. Behalve deze extensions is er ook nog een veel betere duitse versie, jammer genoeg wel in BASIC.

Als je een voeding hebt is het ding op iedere andere computer met een centronics printer aansluiting te gebruiken

Zoals ik al zei zit het ding simpel in elkaar en is het vrij makkelijk na te maken. Het is echter maar de vraag dit wel de moeite waard is. Het apparaat kost maar DM 69,-- Hier komt wel DM 20,-- verzendkosten bij. Als je twijfelt moet je maar naar een gebruikersdag komen. Misschien is de Speakeasy ook aanwezig op de HCC dagen op de Enterprise stand. Als je interesse hebt, kun je schrijven naar:

Werner Lindner
Hard- und Softwareideen
Landsberger StraBe 49
D-8913 Schondorf a.A.

Arjan van Gog, Oosteinde 149, 2611 VC DELFT

TE KOOP AANGEBODEN

Omdat ik ben overgestapt op een IBM-PC-kloon, houd ik weinig tijd over om eens lekker op mijn ENTERPRISE te programmeren etc. Daarom heb ik besloten om m'n ENTERPRISE te verkopen. Vandaar:
Te koop: ENTERPRISE 128K
+Philips Monochroom monitor
+Monitorkabel
+Parallel printerkabel
+2 jaargangen ENTERFACE

In koop 450,-

Mijn naam is: Hans Dekker
J.Ruysdaellaan 15
1741 KT Schagen
Tel: 02240 - 12990

MACHINETAAL IN BASIC

In het vorige nummer van de Enterface stond een programma van Rene van Ee, waarmeer je GEN machinecodefiles in je basicprogramma kon zetten. Een nadeel van de gebruikte methode was dat de machinetaal niet in je listing was te zien, en de routine niet langer kon zijn dan 255 bytes. Een mooiere manier is om de code om te zetten in CODE=HEX$("dd,dd,dd") regels.

Om zo`n programma te kunnen maken, moet je eerst weten hoe BASIC regels in het geheugen worden opgeslagen. Je hebt dus een voorbeeldprogramma nodig, en een een programma dat laat zien hoe het voorbeeldprogramma in het geheugen staat.

In PROGRAM 0 tik je onderstaande BASIC regel in:

```
100 CODE =HEX$("1,2,3,4,FF")
```

Daarna run je onderstaand programma in PROGRAM 1:

```
100 PROGRAM "ONTLEDER.BAS"
110 LET BEGIN=PEEK(540)+256*PEEK(541)
120 LET CHAN=0
130 LET EIND=ADRESS_NEW_LINE
140 FOR ADRES=BEGIN TO EIND
150   PRINT £CHAN:ADRES.PEEK(ADRES),
160   IF PEEK(ADRES)>31 AND PEEK(ADRES)<160 THEN
170     PRINT £CHAN:CHR$(PEEK(ADRES))
180   ELSE
190     PRINT £CHAN:""
200   END IF
210 NEXT
220 DEF ADRESS_NEW_LINE
230   LET ADRES=BEGIN
240   DO
250     !PRINT ADRES,PEEK(ADRES)
260     IF PEEK(ADRES)=0 THEN EXIT DO
270     LET ADRES=ADRES+PEEK(ADRES)
280   LOOP
290   LET ADRESS_NEW_LINE=ADRES
300 END DEF
```

Als je het programma runt op een 576 K machine, zou je de uitvoer op de volgende bladzij moeten krijgen (ik heb hier de getallen naast elkaar gezet om ruimte te besparen). Als je minder geheugen hebt, zal je BASIC programma op een lager adres beginnen, maar verder precies hetzelfde zijn.

4865	27		4879	10	
4866	100	d	4880	49	1
4867	0		4881	44	,
4868	0		4882	50	2
4869	96	`	4883	44	,
4870	9		4884	51	3
4871	19		4885	44	,
4872	68	D	4886	52	4
4873	72	H	4887	44	,
4874	69	E	4888	70	F
4875	88	X	4889	70	F
4876	36	$	4890	9	
4877	8		4891	0	
4878	128		4892	0	

Het is niet zo moeilijk om te bedenken dat het eerste byte het lengtebyte is. Als je dat natelt, blijkt dat dit de lengte (inclusief de lengtebyte) tot en met de eerste 0 na de regel aangeeft. Die 0 op 4895 hoort dus bij de regel, dus:iedere regel begint met een lengtebyte en wordt afgesloten met een 0.

Hierna komt natuurlijk het regelnummer (altijd 2 bytes), gevolgd door een byte dat aangeeft hoeveel er bij het listen moet worden ingesprongen. Zoals je ziet zou een regelnummer in principe ook groter dan 9999 mogen zijn (nl. 65535), maar de programmeurs van Intelligent Software vonden dat kennelijk niet netjes, zodat het LIST commando geen regelnummers groter dan 9999 afdrukt.

Hierna volgt meestal (ik ben eigenlijk nog niet anders tegengekomen) het getal 96, dat aangeeft dat het volgende getal een commandonummer is. Je kunt heel makkelijk een lijstje maken met de nummers en de commando`s daar achter. Het volgende programma doet dit:

```
10 PRINT
20 FOR CODE=0 TO 255
30   POKE 10,CODE
40   LIST 10
50 NEXT
```

Gewoon intypen van dit programma gaat natuurlijk niet, het LISTcommando moet je er met een truukje inzetten, en het adres in 30 moet worden vervangen door het echte adres. In een van de oude, groene Enterfaces stond het programa SUPERKEY van 'Spock' dat dit heel eenvoudig maakte. Omdat misschien een aantal mensen die Enterface en het programma niet hebben, zullen we het nu even met de hand doen. Superkey staat wel in de bibliotheek, en misschien heb ik de redactie op een idee gebracht, en staat het programma opnieuw ergens voor of achter mij.

Om het LISTcommando in de listing te krijgen, zetten we er eerst een ander commando voor in de plaats, dat wel in een programma mag worden gebruikt, bijvoorbeeld PRINT. Regel 40 wordt dus:

```
40   PRINT 10
```

Nu moeten we in het geheugen zoeken op welk adres de code van dit PRINT commando staat, en deze vervangen door de code van het LIST commando. We maken dus een dump van het programma:

4865	7		4866	10		4867	0		4868	0	
4869	96	`	4870	56	8	4871	0		4872	22	
4873	20		4874	0		4875	0		4876	96	`
4877	31		4878	36	$	4879	67	C	4880	79	O
4881	68	D	4882	69	E	4883	19		4884	194	
4885	0		4886	0		4887	34	"	4888	84	T
4889	79	O	4890	194		4891	255		4892	0	
4893	0		4894	16		4895	30		4896	0	
4897	1		4898	96	`	4899	54	6	4900	194	
4901	10		4902	0		4903	12		4904	36	$
4905	67	C	4906	79	O	4907	68	D	4908	69	E
4909	0		4910	10		4911	40	(	4912	0	
4913	1		4914	96	`	4915	56	8	4916	194	
4917	10		4918	0		4919	0		4920	7	
4921	50	2	4922	0		4923	0		4924	96	`
4925	47	/	4926	0		4927	0				

Deze dump is gemaakt op een 576k machine. Heb je zelf minder (of meer) geheugen, dan moet je zelf een dump maken.

Eerst veranderen we het PRINT commando in 40 door LIST. Regel 40 begint op adres 4910. We weten nog niet welke code LIST heeft, maar die vind je snel omdat de commando`s op alfabet staan. Als je die na wat proberen gevonden hebt, doe je dus:

```
POKE 4915,42
```

Dan moet nog het adres in regel 30 worden neergezet, van de code voor het PRINT commando in 10. Regel 10 begint natuurlijk op 4865. De code voor het PRINT commando staat op 4870. Regel 30 wordt dus:

```
30   POKE 4870,CODE
```

Als je dit programma runt, krijg je dus een lijst van commando's. Je kunt nu regel 25 PRINT CODE, tussenvoegen (wat ik zelf eerst was vergeten, je kunt hem er natuurlijk ook meteen inzetten), zodat je een lijst van commando's met hun codenummers krijgt. Voor diegenen die geen printer hebben, heb ik de lijst hieronder neergezet, exclusief het regelnummer dat het LIST commando er tussen gooit.

0	ALLOCATE	46	NEW
1	ASK	47	NEXT
2	AUTO	48	NUMERIC
3	CALL	49	OPEN
4	CAPTURE	50	OPTION
5	CASE	51	OK

6	CAUSE	52	OUT
7	CLEAR	53	PLOT
8	CLOSE	54	POKE
9	CODE	55	SPOKE
10	CONTINUE	56	PRINT
11	COPY	57	PROGRAM
12	DATA	58	RANDOMIZE
13	DEF	59	READ
14	DEF	60	REDIRECT
15	DELETE	61	REM
16	DIM	62	RENUMBER
17	DISPLAY	63	RESTORE
18	DO	64	RETRY
19	CHAIN	65	RETURN
20	EDIT	66	RUN
21	ELSE	67	SAVE
22	ELSE IF	68	SELECT
23	END	69	SET
24	END DEF	70	SOUND
25	END HANDLER	71	START
26	END IF	72	STOP
27	END SELECT	73	INFO
28	END WHEN	74	STRING
29	ENVELOPE	75	TEXT
30	EXIT	76	TOGGLE
31	FOR	77	TRACE
32	GOSUB	78	TYPE
33	GOTO	79	VERIFY
34	GRAPHICS	80	WHEN
35	HANDLER	81	!
36	IMAGE	82	LLIST
37	IF	83	LPRINT
38	IF	84	EXT
39	INPUT	85	GET
40	LET	86	FLUSH
41	LINE	87	LOOK
42	LIST	88	PING
43	LOAD	89	DATE
44	LOOP	90	TIME
45	MERGE	91	WAIT
		92	ON

Na deze zijsprong gaan we weer verder waar we mee bezig waren. We weten nu de code van het CODE commando, namelijk 9. Alleen aan die 9 hebben we nog niet veel. Een CODE=HEX$("dd,dd,dd"") regel zit wel wat ingewikkelder in elkaar dan 96,9. Wat we eersteens moeten doen is goed kijken hoe de CODE regel aan het begin van dit artikel in elkaar zit. De dump hiervan staat aan het begin van dit artikelje. Ik zal nu eerst per adres opnoemen wat het getal betekent. Als ik een adres oversla hoort dat adres bij het vorige.

4865 Lengte van de hele regel

4866 Regelnummer
4868 Hoeveelheid die ingesprongen wordt (indentering)
4869 'Volgende getal is een commando code'
4870 Code voor CODE
4871 operators e.d. worden gecodeerd met een getal tussen 0 en 32.
19 is de code van het "=" teken.
4872 namen van stringvariabelen worden voorafgegaan door een byte met de waarde 64 + 'de lengte van de naam'.
4873 De naam van de variabele
4877 8 is de code voor "("
4878 dit getal geeft aan, dat er een stringconstante volgt, voorafgegaan door
4879 een lengtebyte, dat niet in de lengte wordt meegeteld.
4880 De string die in basic tussen aanhalingstekens stond, inclusief komma's
4890 9 is de code voor ")"
4891 Het programma wordt afgesloten met twee nullen.

Om nu zelf regels in het geheugen te zetten, moet je opzoeken waar je nieuwe regel moet beginnen. Dat is niet moeilijk. Je kijkt bij de eerste regel hoe lang die is. Je gaat dat aantal bytes verder. Staat daar een 0, dan ben je aan het eind, zoniet dan ben je bij een nieuw lengtebyte, en herhaal je deze handeling. De routine ADDRESS_NEW_LINE in het programma MAKELINE.BAS hieronder doet precies hetzelfde.

Omdat je het lengtebyte nog niet weet, moet je eerst de hele regel in een string zetten (COMMAND$).

Daarna volgt het 2-bytes regelnummer. Dat zet je in COMMAND$ met COMMAND$=COMMAND$&WORD$(regelnummer). WORD$ zet een getal om in het formaat van CODE. Dus met CODE =WORD$(257) krijg je HEX$("01,01") in het geheugen.

Om het indenterings (inspring) byte hoef je je geen zorgen te maken. Door LIST wordt een routine (RST 10h, function: C1h, 193 decimaal) aangeroepen die deze bytes set.

Daarna volgt bij CODE =HEX$ altijd:

```
96,9, 19,68, 72,69,88,36, 8,128
          ^   ^  ^  ^  ^
          ! " H  E  X  $ "
         64  +1 +1 +1 +1
```

gevolgd door een lengtebyte. Om dat lengtebyte neer te kunnen zetten, moet je eerst weten hoe lang de stringconstante is. Je moet dus eerst de code inlezen en omzetten in de string dd,dd,dd,dd om te weten hoelang die string is. Deze string zet je apart in de variabele CODE$, om het lengte byte erin te kunnen zetten. Je begint dus met CODE$.

Daarna volgt dus een 9 als afsluitingshaakje.

Hieronder staat het programma. Voor zover ik weet is het bugfree, maar dat kan ik natuurlijk niet garanderen. Mocht het ooit niet goed werken, RESET dan de computer en probeer het nog eens. In regel 160 moet je de filename veranderen in de file die je wilt inlezen. Echt gebruikersvriendelijk is dat niet, maar als je er zin in hebt kun je dat zelf wel verhelpen. In de CALL in regel 160 kun je een ander regelnummer zetten (het staat zo slordig als een regelnummer twee keer voorkomt in een listing).

```
100 PROGRAM "MAKELINE.BAS"
110 !MAKELINE reads a gen objectcode file, converts it to CODE =HEX$
115 !          BASIC lines.
120 !                                    15-12-88
130 STRING CODE$,COMMAND$
140 LET BEGIN=PEEK(540)+256*PEEK(541) !                    > Save pointer
150 !
160 CALL MAKELINE(20,1,"FILENAME.OBJ") !    > (linenumber, channel, file)
170 !
180 DEF MAKELINE(LINE_NUMBER,CHANNEL,FILE$)
190   OPEN £CHANNEL:FILE$
200   !
210   LET ADRES=ADRESS_NEW_LINE !                          \ Find adres for new
220   !                                                    / basic lines
230   POKE 540,MOD(ADRES,256):POKE 541,INT(ADRES/256)
240   LET CODE$=""
250   !
260   FOR READ=1 TO 16 !                                        \Skip header
270     GET £CHANNEL:DATA$ !                                    \ and read
280     IF READ=3 THEN LET CODE_SIZE=ORD(DATA$) !                > code-
290     IF READ=4 THEN LET CODE_SIZE=CODE_SIZE+256*ORD(DATA$)!/ length
300   NEXT  !                                                  /
310   !
320   DO
330     !
340     IF CODE_SIZE<=0 THEN EXIT DEF !                  \  Check for end
350     !                                                /        of data
360     !
370     FOR READ=1 TO 15 !                          \
380       IF CODE_SIZE-READ<0 THEN EXIT FOR !        \
390       GET £CHANNEL:DATA$ !                       \
400       PRINT ORD(DATA$) !                          \
410       IF DATA$="" THEN EXIT FOR !                  \ Read and create
420       LET CODE$=CODE$&DEC_TO_HEX$(ORD(DATA$))&"," ! /  codestring for
430     NEXT  !                                       /     CODE command
440     LET CODE_SIZE=CODE_SIZE-15 !                 /
450     LET CODE$=CODE$(1:LEN(CODE$)-1) !           /
460     LET CODE$=CHR$(LEN(CODE$))&CODE$ !         /
470     !
480     LET COMMAND$=WORD$(LINE_NUMBER)&HEX$("00,60,09,13,44")&"HEX$"&
        HEX$("08,80")&CODE$&HEX$("9,0")
490     !
500     LET COMMAND$=CHR$(LEN(COMMAND$)+1)&COMMAND$ !>    Add length-byte
```

```
510     !
520     CODE =COMMAND$ !                                  > Put line in memory
530     !
540     LET CODE$,COMMAND$=""
550     !
560     LET LINE_NUMBER=LINE_NUMBER+10
570     !
580   LOOP
590   !
600   CODE =HEX$("0,0") !                                 \ Mark end of basic
610   !                                                   / pointer
620   !
630   POKE 540,MOD(BEGIN,256):POKE 541,INT(BEGIN/256) !\ Reset allocate
640   !                                                   / pointer
650   CLOSE £CHANNEL
660 END DEF
670 DEF DEC_TO_HEX$(DEC)
680   !
690   ! This functions returns any decimal value as a string.
700   !
710   !
720   STRING HEX$
730   LET HEX$=""
740   !
750   DO
760     LET HEX$=CHR$(MOD(DEC,16)+48-7*(MOD(DEC,16)>9))&HEX$
770     LET DEC=INT(DEC/16)
780   LOOP WHILE DEC>0
790   LET DEC_TO_HEX$=HEX$
800   !IF LEN(HEX$)=1 THEN LET DEC_TO_HEX$="0"&HEX$
810 END DEF
820 DEF ADRESS_NEW_LINE
830   LET ADRES=BEGIN
840   DO
850     PRINT ADRES,PEEK(ADRES)
860     IF PEEK(ADRES)=0 THEN EXIT DO
870     LET ADRES=ADRES+PEEK(ADRES)
880   LOOP
890   LET ADRESS_NEW_LINE=ADRES
900   PRINT :PRINT
910 END DEF
```

Arjan van Gog, Oosteinde 149, 2611 VC DELFT, tel.:015-124004

```
  100 PROGRAM "SUPERKEY"(REGELNUMMER,NEW_KEYWORD$)
  110 !--- LOAD dit programma in PROGRAM 1 -- ga terug naar PROGRAM 0
  120 !--- alleen PRINT wordt vervangen door dit programma
  130 !--- gebruik: b.v. CHAIN "superkey"(320,"edit")
  140 !--- zet aan het begin van je programma --- 1 END anders start het
  150 LET ADRES=4809:LET MEER=1
  160 LET KEYWORD$="":LET NEW_KEYWORD$=UCASE$(NEW_KEYWORD$)
  170 DO
  180   LET JUMP=PEEK(ADRES)
  190   IF REGELNUMMER=PEEK(ADRES+1)+PEEK(ADRES+2)*256 THEN
  200     FOR J=ADRES TO ADRES+JUMP
  210       IF PEEK(J)=96 AND PEEK(J+1)=56 THEN
  220         LET NEW_KEY=VERVANGING
  230         IF NEW_KEY THEN
  240           POKE J+1,NEW_KEY
  250           PRINT CHR$(25):PRINT CHR$(25);"poke "&STR$(J+1)&","&STR$(NEW_
KEY)
  260           PRINT CHR$(25);"dus op regel";REGELNUMMER;"is PRINT gewijzigd
in ";KEYWORD$
  270         END IF
  280         EXIT FOR
  290       END IF
  300     NEXT
  310   END IF
  320   LET ADRES=ADRES+JUMP
  330 LOOP WHILE JUMP
  340 CHAIN 0
  350 END
  360 DATA 02=AUTO,15=DELETE,20=EDIT,42=LIST,43=LOAD,45=MERGE
  370 DATA 46=NEW,51=OK,67=SAVE,71=START,73=INFO,79=VERIFY,00=NIET
  380 DEF VERVANGING
  390   RESTORE 360
  400   DO
  410     READ KEY$
  420     LET KEYWORD$=KEY$(4:)
  430   LOOP UNTIL KEYWORD$=NEW_KEYWORD$ OR KEYWORD$="NIET"
  440   LET VERVANGING=VAL(KEY$)
  450 END DEF
  460 !----- spock data '86 -----------------for ENTERPRISE 128 ----
```

De ROUTINE-tuin
Alhoewel het niet verplicht is, is het daarintegen wel overzichtlijk de routines in volgorde van de device-table te rangschikken. Zoals vroeger al was aangegeven, zet U de routines van enige lengte, die door meerdere device-functies worden benut, het liefst in een blok apart (General routines).

De INTERRUPT-routine.
Zodra de Z80-processor een interrupt krijgt, door video (lees NICK), 1 Hz (ook NICK), sound (DAVE) of "external" (lees NET:), dan wordt naar de zogeheten "INTERRUPT"-vector op adres 038h gesprongen. Exos heeft deze daar neergezet om later zelf de touwtjes in handen te krijgen. Is EXOS eenmaal meester over het systeem dan kijkt het snel alle devices na of ze het allemaal naar de zin hebben. De meeste vragen geen extra aandacht maar er zijn zo van die probleem gevallen die elke keer gehoord willen worden, zoals "KEYBOARD:". We kunnen het dit device niet kwalijk nemen omdat het maar een buffer van 1 byte heeft. (De snelle typisten onder U, die meer dan 50 tekens per seconde op dit wonderschone apparaat kunnen aanslaan, weten er van mee te praten.) Andere devices wisselen nog wel eens, zoals het printer device in de Printer_Spooler, die Patric van Voorn heeft ge-schreven. Met het commando :SPOOL wordt de printer ge-toggled tussen normaal en de Spooler-mode met 50 Hz interrupt. Aangezien ik mede schuldig ben aan deze constructie wil ik de gebruikers er op wijzen dat elke interrupt-routine tijd kost, ook al stuit dit meteen op RET. Een aardige verbetering kunt U krijgen door het interrupt-byte pas op £20 te zetten zodra er wat naar de printer gestuurd wordt en op £08 (= 1 Hz) of £00 (= geen) te resetten wanneer de buffer leeg is. Hierdoor bent U heel wat onnodige overhead kwijt.

Wat U uiteindelijk in de interrupt-routine zet is afhankelijk van het device dat U aan het bouwen bent. Zodra ik aan de suggesties voor nieuwe devices ben, kom ik daarop terug.

De OPEN-routine, CREATE-routine
Ver buiten dit device (basic of een ander stukje machinetaal),wordt een verzoek tot communucatie voorbereid. Netjes, via de postbode EXOS, wordt dit verzoek bij deze open of create routine afgeleverd.
U kunt dit simpel weigeren met

```
LD A,E7          ;call not supported
RET
```

of toestaan maar dan hebben deze routines, zoals eerder reeds gezegd, een verplicht nummerje af te draaien. Exos kan niet zomaar weten hoe groot de channel-buffer zou moeten zijn. Het is de taak van een open-routine de channel_buffer bij EXOS aan te vragen. Hoe dit moet is bij "LAZY:" al te zien. De hoeveelheid buffer die U nodig hebt, zet U in registers DE met dien verstande dat alleen een device met DD_FLAG = £01 (zoals Video:) meer dat 16K mag aanvragen of doorgeven.

Voor de volledigheid dus

```
(register A bevat nog steeds het aangevraagde channel)
OPEN:
```

```
CREATE:
   LD DE,channel_grootte       (zie verderop)
   EXOS _ALL_BUF
```

Deze actie is nodig omdat hierbij een channel-descriptor wordt aangemaakt en het adres van dit channel in het IX register (pag 1 adres) wordt doorgegeven. Indien U daar behoefte aan heeft kunt U deze pointers binnen het device bewaren.

Als U zeker weet dat aan dit device om een communicatie channel wordt gevraagd dan stelt U de channel_grootte bij de _ALL_BUF call op 0 en reserveert op dit segment het gewenste stuk buffer. In de read en write routines zal ik U laten zien dat U daar eenvoudiger uit kunt lezen en in schrijven.

```
PUSH IX           of niets
POP DE            en LD DE,DEV_BUFFER
LD HL,BUF_POINTER   ;in data section
LD (HL),E
INC HL
LD (HL),D
INC HL
IN A,(£B1)        of IN A,(£B3) ;segment met buffer
LD (HL),A
INC HL               ;HL wijst nu naar NEXT_CHAR_POINTER
LD (HL),E
INC HL
LD (HL),D
INC HL
LD BC,channel_grootte of buffer_grootte
EX DE,HL
ADD HL,BC            ;begin + lengte = eind-adres
EX DE,HL             ;HL wijst naar END_BUF_POINTER
LD (HL),E            ;DE bevat END_BUFFER
INC HL
LD (HL),D
RET
```

Mischien heb ik een volgende keer een echte Channel-descriptor aan mijn buffer geplakt en in de channel_chain ingelinkt.

Heeft U dingen gedaan die U bij het sluiten van een channel weer terug moet draaien dan kunt U daar twee vormen voor kiezen. U kunt de oude waarde van een variabele opslaan op een eigen locatie in de data sectie of deze waarde direct binnen de close-routine dumpen.

Naar U allicht weet, kunt U aan een device vragen een channel met een filenaam te openen. Heeft U zo'n file handling device gebouwd dan wijst DE naar een filenaam op het system segment. Deze naam voldoet reeds aan de eisen van een goede naam. Hoe U dan vervolgens de harddisk moet aanspreken zal een beetje afhankelijk zijn van de hardware die er om heen hangt. Misschien kom ik er later nog eens op terug.

De CLOSE-routine, DESTROY-routine.
Door deze device routine wordt U in de gelegenheid gesteld om het onheil dat U over Uzelf heeft aangeroepen, efficient van U af te weren. Of U er werkelijk verstandig aan doet om het bericht :"AGENDA:------MORGEN TENTAMEN GALACTISCHE-QUANTUMMEACHNICA-------"ongelezen te vernietigen laat ik aan Uzelf over. In ieder geval, als U channels heeft geopend boven de 100, kunt U ze beter sluiten. Blijf vooral met je po..tengels van het Keyboard channel af want je houdt gegarandeerd een dove machine over.

Gaat alles gladjes dan is

```
CLOSE:
DESTROY:
    XOR A
    RET
```

meestal voldoende.

Mocht Uw eerste device toch denken dat hij (of zij) nu de macht in handen heeft, roep dan de rode knop te hulp!

de READ_CHAR-routine.
De meeste devices weren de vraag af met LD A,£E7 = call not supported. Enkele andere spelen de vraag door naar KEYBOARD:. Het moet echter ook mogelijk zijn om van een input_disk-channel of video-channel te lezen en dat dan als informatie door te sluizen aan het stuk program dat dit device om een character had gevraagd. Heeft Uw device een legale channel-buffer en er al wat in weggeschreven dan kan het er ook weer worden uit gelezen. Zodra U het channel van buiten benadert met een Read_char vraag, komt U het device binnen met het channel segment in pag 1 en het channel adres in IX. U kunt deze IX gebruiken en daar dan wat bij optellen om aan het gewenste adres te komen of U neemt de next_char_pointer die U bij het openen van het channel naar het eerste vrije byte laat wijzen. Ik koos voor het laatste zodat:

```
READ_CHAR:
  CALL LOOK_FOR_SPACE  (mischien goed als GENERAL ROUTINE??
                        maar staat nu bij de write routines)
  LD HL,(NEXT_CHAR_POINTER)
  LD B,(HL)          ;char in B
  INC HL
  LD (NEXT_CHAR_POINTER),HL
  XOR A              ;status OK
  RET
```

Het "PIO:"-device zal waarschijnlijk een input poort lezen

```
  IN A,(£??)
  LD B,A                ;char in B
  XOR A                 ;status OK
  RET
```

De READ_BLOCK routines.
De meeste build-in devices hebben de (on)aardige gewoonte om read_block uit te voeren als een herhalende, zeer ingewikkelde en tijdrovende read_char actie tussen twee stukken RAM die allebei op pag 1 worden geadresseerd maar in verschillende segmenten. Daarbij komt nog dat voor beide stukken bijgehouden moet worden of de segment grenzen niet worden overschreden. Met de eigen device-buffer op pag 3 is het leven veel eenvoudiger geworden. EXOS komt naar deze routine toe met in DE het adres van de buffer waar het naar toe moet, in BC het aantal bytes en in A het channel dat aan dit device was toegewezen. Op pag-0 staat het zero-segment, pag-1 is het segment met Uw channel met N? bytes buffer (IX wijst naar het startadres), pag-2 is sys-segment geworden en op pag-3 staat nu het device met de device-buffer en deze routine die tot actie wil over gaan! Heb ik met een aangeplakte channel_descriptor mijn dev_buffer tot channel verheven dan staat het zelfde segment zowel op pag-1 als pag-3. Een ongevaarlijke vorm van navelstaren dus. Zolang de registers DE buiten de dev_buffer wijzen zult U niet in Uw eigen staart bijten. U destileert uit het DE register op welke oorspronkelijke pagina de buffer waar het naar toe moet heeft gestaan.

```
SET_DE_PAG_1:     (ook een GENERAL ROUTINE ????)
   PUSH AF        ;save AF
   PUSH HL        ;en HL
   LD A,D         ;A=D= XY?? ????
   RLCA           ;A= ???? ???X
   RLCA           ;A= ???? ??XY
   AND £03        ;A= 0000 00XY  =£00 of £01 of £02 of £03
```

Vanaf het sys-segment adres £BFFC staan de 4 segmenten die vlak voor deze device-call de Z80-pagina's zijn geweest. Het is daarom een zaak van aftellen om het segment te vinden.

```
   LD HL,£BFFC
   ADD A,L        ;A= £FC of £FD of £FE of £FF
   LD L,A
   LD A,(HL)      ;USR_P0, USR_P1, USR_P2, USR_P3
   OUT (£B1),A    ;zet dit segment op pag 1
   RES 7,D        ;pas DE adres aan
   SET 6,D
   POP HL
   POP AF
   RET
```

Nu is het geen probleem meer om bij die dump-buffer te komen en de data er in te plonsen.

```
READ_BLOCK:
   IN A,(£B1)        ;haal pag-1 op
   PUSH AF           ;en save op stack
   CALL SET_DE_PAG_1
   LD HL,(NEXT_CHAR_POINTER)
   LDIR              ;tjoep, daar staat het
```

```
        LD (NEXT_CHAR_POINTER),HL
        POP AF
        OUT (£B1),A      ;pag-1 terug
        XOR A            ;status OK
        RET
```

DEVICE DRIVERS
De WRITE routines
In principe geldt voor de write-routines het zelfde als voor de read-routines. Er kan naar een channel buffer of naar DEV_BUFFER worden geschreven. Ook nu kunt U al dan niet gebruik maken van een tellertje die direct, of met een optelsommetje aanwijst waar U naar kunt schrijven. Let er evenwel op dat U de buffer grenzen niet overschrijd. Voor de dev_ buffer is de test methode:

```
LOOK_FOR_SPACE:
  PUSH HL
  PUSH DE
  PUSH BC
  PUSH AF
  LD HL,(NEXT_CHAR_POINTER) ;daar waren we gebleven
  EX DE,HL
  LD HL,END_BUFFER
  SBC HL,DE          ;HL is nul als we aan het eind staan
                     ;en >0 als er nog ruimte is.
  JR Z,NO_SPACE      ;
```

In de programmeerfase zult U moeten beslissen hoe het systeem zal reageren op het volraken van de buffer. Beschadigt U de channel descriptor dan crasht het systeem. De prettigste methode is die van de editor nl. een rond lopende buffer. Zodra U aan het eind staat zult U weer naar het begin moeten springen. Via output van ^G naar "SOUND:" kunt U op deze situatie opmerkzaam gemaakt worden. We gaan echter nog even verder,

```
  SBC HL,BC          ;geeft carry als er meer zal worden
                     ;geschreven dan er plaats is.
  JR NC,EXIT_L_F_SP
NO_SPACE:
  LD B,£07           ;^G
  LD A,SOUND_CHAN
  EXOS _WRITE
  LD DE,DEV_BUFFER
  LD HL,NEXT_CHAR_POINTER :zet de read/write pointer op start
  LD (HL),E
  INC HL
  LD (HL),D
EXIT_L_F_SP:
  POP AF
  POP BC
  POP DE
  POP HL
```

```
RET
```

De WRITE_CHAR routine schrijft echter 1 byte weg en dat staat bovendien nog in B. Voor U de LOOK_FOR_SPACE call doet moet dit char dus veilig zijn en BC op 1 staan.

```
WRITE_CHAR:
  PUSH BC                       ;save char op stack
  LD BC,£0001                   ;slechts 1 byte
  CALL LOOK_FOR_SPACE
  POP BC
  LD HL,(NEXT_CHAR_POINTER)
  LD (HL),B
  INC HL                        ;werk pointer bij
  LD (NEXT_CHAR_POINTER),HL
  XOR A
  RET
```

De WRITE_BLOCK routine krijgt nu vanaf adres DE data te verwerken met een lengte van BC-bytes. Dat onhandige gehannes met voor elk over te bregen byte telkens twee segmenten uit te wisselen zie ik niet zo zitten. Een keer is genoeg, dus:

```
WRITE_BLOCK:
   CALL LOOK_FOR_SPACE
   IN A,(£B1)
   PUSH AF
   CALL SET_DE_PAG_1            ;DE-SEGM is bron segment
   LD HL,(NEXT_CHAR_POINTER)
   EX DE,HL                     ;wissel bron en doel
   LDIR                         ;klaar
   LD (NEXT_CHAR_POINTER),DE
   POP AF
   OUT (£B1),A                  ;herstel pagina 1
   XOR A
   RET
```

Heeft U een output device zoals printer, 8-kanaals lichtorgel e.d. dan kunnen de data daarna naar £B6, de paralelle output-poort gestuurd worden.

```
OUT (£B6),A
```

De seriele poort is £B7. Ook hier kunt U op aansluiten wat U wilt, alleen het is maar 1-kanaals. Het normale seriele protocol geeft voor een 1

```
LD A,£03        ;0000 0011
OUT (£B7),A
```

en voor een 0

```
LD A,£02        ;0000 0010
OUT (£B7),A
```

Met diverse wacht loops kunt U het tempo van de opeenvolging van nullen en enen naar eigen hand zetten.

Een strobe puls op printer connector A2 maakt U door bit 4 van poort £B5 te wisselen.

```
LD A,(£B5)
OR £10          ;???1 ????
OUT (£B5),A
  .
  .
AND £EF         ;???0 ????
OUT (£B5),A
```

Wilt U de data via de remote-control relais uitvoeren dan heeft U bit 7 voor remote-1 en bit 6 voor remote-2 nodig van poort £B5. Houdt er rekening mee dat deze relais een mechanische component in zich hebben en dus kunnen slijten.

De READ_CHAN_stat routines.
Volgens het Technisch Manual kan deze functie worden uitgevoerd om te weten of een read call wel zin heeft. Waar in z'n algemeen-heid naar gekeken wordt is mij nog niet duidelijk. Bij keyboard wordt naar een plaats in het geheugen gekeken waar een kopie van het ingevoerde teken staat. Bij de aanwezigheid van een teken is register C = £00 en bij EOF is C = £FF. In alle andere gevallen is C =£01. Vrij vertaald: C= £00 als de lees-pointer in een chan de schrijf-pointer nog niet gepasseerd is. Bij volle channel buffer C =£FF. Lees-pointer tussen schrijf-pointer en end_of_chan dan C =£01.

De SET_CHAN_STAT routine.
Deze routine is voor een file handling device bijzonder belangrijk. Ik werd er pas op de gebruikersdag van juni j.l. door Philip Homburg op attent gemaakt. Veel ervaring heb ik dus niet. Volgens het Technisch Manual is het de bedoeling dat hiermee random access op files kan worden uitgevoerd. Buiten het gebruikelijke chan_nr in A is er ook nog de write_ flag in C met alleen bit 0 en bit 2 als geldige vlaggen. De registers DE moeten naar een 16 bytes parameter blok wijzen met (byte 0-3)een 32-bits file-pointer, (byte 4-7) een 32-bits file-size, (byte 8) protect-byte en (byte 9-15) de rest zijn nullen. Het is alleen wachten op iemand die een 4096-K file weet te maken die dan de volle 32-bits file-pointer nodig heeft. Wie het wat bescheidener wil doen kan eens wat experimenteren met file van 1-K. De pointer die dan toch nog moet worden ingevuld is (00000000 00000000 00000100 00000000) ofwel £00,£00,£04,£00 of minder. Wie er wat meer ervaring mee heeft moet dat me tegen die tijd maar eens zeggen.

De SPEC_FUNC routine.
Onder dit hoofdstuk kan eenieder zijn of haar fantasie uitleven. Zo zou ik me voor kunnen stellen dat "PLOTTER:" met deze functie de penkleur laat veranderen of van A4 formaat naar A5 kan stapt. De "ROBOT:" kan de

grijper openen of sluiten. "CLIMAT:" leest de buitentemperatuur en " SECURIT:" start een bandrecorder met Bouvier geblaf zodra er op de bel wordt gedrukt. Als er bij Uw thuiskomst een aanslag voor de hondenbelasting in de bus ligt, kunt U er zeker van zijn dat een van de aanbellers diep onder de indruk was van dit device.
Op verzoek van EXOS 2.1 zijn de sub_functions tot 63 voor IS/Enterprise gereserveerd, daarboven is alles vrij. Mocht U plannen hebben voor Uw device een paar functie nummers te reserveren laat het me dan weten.

De INIT routine.
Een device moet over het algemeen geinitialiseerd worden. Dit betekent dat voor het device echt kan werken, eerst het een en ander klaar gezet moet worden. Wilt U alle interrupts eerst door dit device laten afhandelen? zet dan de soft_interrupt_vector op segm 255 £BFED,£BFEE. Volgens het Technical Manual moet dit wijzen naar een adres op page 0. Helaas zijn er tussen £0000 en £0030 niet voldoende bytes die zowel onder IS-BASIC als onder ISDOS ongebruikt blijven. Wat wel is gegarandeerd, is dat het sys-segment op page 2 zit. Bouw de GO_ON inlink routine iets anders (zie Enterface sep-okt-nov 88 pag 23) en vraag 17 bytes dev_ram aan.

```
            GO_ON:
              POP AF
              LD A,(DD_TAB+£01)
              SET 6,A
              RES 7,A
              LD (DD_TAB+£01),A
              IN A,(£B3)
-->           LD (SPEC_INTER_SEG+£01),A
              LD (DD_TAB_SEG),A
              LD DE,DD_TYPE
-->           LD BC,£0011
-->           PUSH BC
              EXOS _LINK
-->           POP BC
              RET NZ            ;wie is er foutloos??
-->           PUSH IX           ;bevat adres hoogste dev_ram_byte
-->           POP HL
-->           SBC HL,BC         ;HL is nu het laagste adres
-->           EX DE,HL
-->           LD HL,£BFED       ;$USER_ISR  user_interrupt_vector
-->           LD (HL),E
-->           INC HL
-->           LD (HL),D
-->           LD HL,SPEC_INTERRUPT
-->           LDIR              ;zet spec_interrupt routine in sys_segm
              XOR A
              RET
```

De SPEC_INTERRUPT routine is dan:

```
            SPEC_INTERRUPT:
```

```
    IN A,(£B3)      ;SAVE PAG-3 SEGM
    LD (EXIT_SPEC_INTER+£01),A
SPEC_INTER_SEG:
    LD A,£00        ;TO BE FILLED IN BY DEVICE
    OUT (£B3),A     ;DEVICE IN PAGE 3
    JP your_interrupt_routine_on_dev_segm
EXIT_SPEC_INTER:
    LD A,£00        ;TO BE FILLED IN
    OUT (£B3),A
    RET
```

De SOFT_INTERRUPT_VECTOR wijst dan naar dit stukje device-ram.
Verdere initialisatie acties zouden kunnen zijn: "PIO:" zet alle array's op nul en "ROBOT:" wil samen met "PLOTTER:" de apparatuur naar de uitgangs stand dirigeren, "SECURIT:" checkt of alle sloten in het veiligheids circuit de vereiste stroom door laten. "MODEM:" test of er inderdaad een modem aan de serial hangt. "AGENDA:" leest de file " HUISWERK.AGD". "HOROSCOOP:" en "BIO_RITHM:" willen een datum van U hebben. Het lijkt me niet erg handig als "CLIMAT:" eerst de verwarming uit gooit em daarna vanuit de nul-stand een stookprofiel oplegt aan de cv-ketel.
U ziet, de init fase kan ook bestaan uit het opnemen van de gevonden situatie. U moet evenwel goed nadenken of U dit allemaal wel bij iedere warme system reset aan Uw hoofd wilt hebben. Een goed alternatief is om, tijdens de eerste init, dit soort data binnen het device in een buffer op te slaan en vervolgens de entry naar deze routine te wijzigen in bv. JP EXIT_ACA. Een apart :INIT_DEV commando dat U in de command_table heeft staan biedt U dan ten alle tijden opnieuw de mogelijkheid met een schone lij verder te gaan.

In de init-routine opent U ook de channels, buffers en alle andere atributen, toeters en bellen die Uw device als default aanneemt. Dit zult U wel bij elke warme reset moeten herhalen omdat EXOS alle device channels sluit, zelfs video channels boven de 100!

De BUFFER_MOVE routine.
De buffer_move routine wordt zelfstandig door EXOS uitgevoerd. Het Systeem is echter zo aardig om U daarvan in kennis te stellen en deze routine aan te roepen. Indien U er belangstelling voor heeft en wat pointers moet aanpassen aan de nieuwe situatie dan is de hoeveelheid verschuiving door BC aangegeven en het segment waar de channel buffer nu in ligt vindt U met:

```
    IN A,(£B1)
```

U schuift dichter op het systeem zodra er een eerder geopend video channel wordt geclosed. Het omgekeerde vindt plaats als een video channel wordt geopend.

Tot slot ook hier weer een data-sectie

```
BUF_POINTER:
  DEFW £0000
BUF_SEGM:
```

```
  DEFB £00
NEXT_CHAR_POINTER:
  DEFW £0000
END_BUF_POINTER:
  DEFW £0000
DEV_BUFFER:
  DEFS hoeveel_had_U_g'at_willen_hebben?
END_BUFFER:
```

PERMANENTE KLOK

We kennen allemaal (ja toch?) de klok van fabian voor in de statuslijn. Het jammere van dit programma is dat de klok verdwijnt, wanneer je naar de WP gaat of je roept een andere extensie aan. (IS-DOS, BASIC, LISP, ETC)

Het onderstaande programma heeft stan tuinder geschreven om een permanente klok te maken. Je kunt het aanroepen in je EXDOS.INI file dat je op je opstartschijf hebt zitten. Zolang je niet reset blijft de klok in de statuslijn rechtsbovenin staan.

De kloktijd wordt een maal per seconde 'geupdate'. ('bijgesteld' in het nederlands)

Laad onderstaand programma en laat het 'runnen'. (werken)

```
100 PROGRAM "KLOK_DEV.BAS"
110 OPEN £1:"KLOK.XR" ACCESS OUTPUT
120 FOR ITEL=1 TO 16
130   READ HDR
140   PRINT £1:CHR$(HDR);
150 NEXT ITEL
160 DATA 0,7,84,0,0,0,0,0,0,0,0,0,0,0,0,0
170 ! 7=RELOC EXTENSION
180 ! 84=LENGTE VAN EXTENSION
190 FOR ITEL=1 TO 98
200   READ RSX
210   PRINT £1:CHR$(RSX);
220 NEXT ITEL
230 DATA 60,191,129,12,6,171,21,182,179,25,65,80,0
240 DATA 140,13,0,0,128,0,15,112,171,5,162,201
250 DATA 0,0,0,0,0,64,2,4,192,0,0,0,66,89,48,158,75,128
260 DATA 64,2,19
270 DATA 162 !of voor EXOS 2.0: 194
280 DATA 252,34
290 DATA 216 !of voor EXOS 2.0: 218
300 DATA 95,1,128
310 DATA 71 !of ook de seconden: 103
320 DATA 224,56,28,14,7
330 DATA 115,3,216,203,0,144,76,252,24,2,65,96,8,40,0
340 DATA 230,7,177,150,1,32,153,225,252,1,20,0
350 DATA 199,203,160,144,76,86,16,110,50,113,0,10
360 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0
370 CLOSE £1:!
```

Je hebt nu KLOK.XR, doe :LOAD KLOK.XR en stel de tijd in. Klaar is ...

---

28 januari 1988 ---- 28 januari 1988 ---- 28 januari 1988

HF Het Witte Dorpshuis
Henri Dunantlaan 4
De Bilt

bus 57 vanaf CS utrecht, uitstappen halte nobellaan, nobellaan in, 3e weg links schuin over parkeerplaats

---

--- 18 maart --- 18 maart ---- 18 maart --- 18 maart ---

de bilt

--- 20 mei --- 20 mei --- 20 mei --- 20 mei --- 20 mei --

de bilt

HOEWEL NIEMAND ZICH HIERAAN HOUDT TOCH MAAR WEER EEN EEN OPROEP:

SLUITINGSDATA KOPIJ: --- 1 WEEK NA DE GEBRUIKERSDAG ---

Het boekje is vaak niet optijd omdat er te weinig kopij is om een nummer te maken. Wanneer men een nummer verwacht komt er opeens wel kopij, maar dat is het boekje wat later.

Hou je aan de regels betreffende insturen van kopij, voorin het blad zie je ze staan.
Centreren van de titel hoeft niet , gaat vanzelf met het redaktieprogramma LAY-OUT(device,file$,blz_nr)

ADRESSEN OM TE WETEN

Gelieve zoveel mogelijk op de aangegeven tijden te bellen!!!!!

VOORZITTER	*Stan Tuinder.*	tevens tijdelijk penningmeester
	Willemstraat 170,	
	2713 AJ ZOETERMEER	
	079-169523	ma-vr 18.00-22.00 uur
SECRETARIS	*Fons Kraakman.*	ledenadministratie enz.
	Bleekveld 8.	AANMELDEN & OPZEGGEN
	1852 JH HEILOO	
	072-335131	
PENNINGMEESTER .	*Stan Tuinder*	zie voorzitter
LID-A	*PLAATSJE*	
	IS	
	VRIJ	
	VOOR JOU !	ma-vr 20.30-23.00 uur
LID-B	*Hans Roelofs,*	expert educative software
	Calsplantsoen 47,	
	1945 SL BEVERWIJK	
	02510-46630	ma-vr 20.00-22.00 uur
BIBLIOTHEEK	*Dik Fennema*	beheer en aanvraag software
	v Duivenvoordelaan 30,	
	2241 ST WASSENAAR	
	079-169523	ma-vr 19.00-22.00 uur
REDAKTIE	*Robin Ketelaars.*	adviseur/redacteur
	Geerweg 2,	ALLEEN KOPY ZENDEN
	2611 VN DELFT	
	015-126422	tussen 19.00 en 23.00 uur

De regio's kennen de volgende kontaktpersonen:

NOORD-NEDERLAND	*Kees van der Dong*	*05945-15626*
	Peter Hofstee	*050-775850*
OOST-NEDERLAND	*??????*	
WEST-NEDERLAND	*Zie bestuur!*	
GELDERLAND	*Peter van Helvoirt*	*08885-2186*
ZEELAND	*Ben Sinke*	*01185-2525*
NOORD-BRABANT	*Rob Hanssens*	*01651-1245*
LIMBURG	*Stichting Synthese Studio*	*043-617623*
	vraag naar Rene terHorst of Jos Mulders	

PRIJSLIJST ENTERPRISE HARDWARE EN SOFTWARE

Enterprise 64K	DM 198,--
Enterprise 128K	DM 298,--
Diskdrive 3.5". ds/dd 720Kbyte met ingebouwde diskcontroller	DM 498,--
Diskcontroller	DM 148,--
Diskdrive 5.25" ss/sd 180Kbyte (geen controller)	DM 198,-- *
Pakket 5.25" ss/sd drive+controller	DM 298,-- *
Kleurmonitor 40x25 tekens, incl. aansluitkabel	DM 348,-- *
Matrix printer EP 80+, 100 CPS, incl. aansluitkabel	DM 348,-- *
Inktlint voor EP 80+	DM 19,80
Speakeasy spraaksynthesizer	DM 69,--
Cassetterecorder	DM 45,--
Monitorkabel:SCART, chinch of open-end voor andere monitors incl. dokumentatie (aansluitgegevens)	DM 24,--
Centronics (printerkabel)	DM 24,--
Joystickkabel- en adapter	DM 24,--
Serial/netwerk (zonder stekker, incl dokumentatie)	DM 24,--

Ook alle Enterprise onderdelen, zoals Nick/Dave chips, keyboardmembramen etc. zijn leverbaar. Prijs op aanvraag.

De met "*" aangeduide artikelen zo lang de voorraad strekt.

		module	Cassette	Diskette
Lisp		89,--		
Forth			69,--	69,--
Spanish Gold	(tekstadventure met animaties)		29,--	29,--
Space Pirate	(duitstalige tekstadventure)		29,--	29,--
Adenture Disk I	(twee duitse tekstadventures: Billy the Kid en Drachenwelten)			29,--
Tombs of Doom	(zoek de weg door 256 ruimtes)		29,--	29,--
Wriggler	(bescherm een worm tegen kevers, spinnen en andere gevaren)		29,--	29,--
Gridtrouble	(bekende arcadehit; 32 levels)		29,--	29,--
BATMAN	(naar de wereldbekende strip)		29,--	29,--
Jammin	(Verzamel muziekinstrumenten)		29,--	29,--
Superpipeline II	(bescherm een pijlleiding)		29,--	29,--
Jack's House of Cards	(Verzamel speelkaarten)			
Wizard's Lair	(NIET hetzelfde als Devil's Lair)		29,--	29,--
Dot-Collector	(als 'Arkanoid' en 'Breakout')		29,--	29,--
Dot-Breaker	(als 'Pacman' maar meer schermen)		29,--	29,--
Spelpakket II	Dot Breaker, Dot Collector, BATMAN, Superpipeline, 5 in a row, U-Boot.			98,--

Werner Lindner Hard- und Software ideen
Landsberger Str.49 8913 Schondorf a. A.