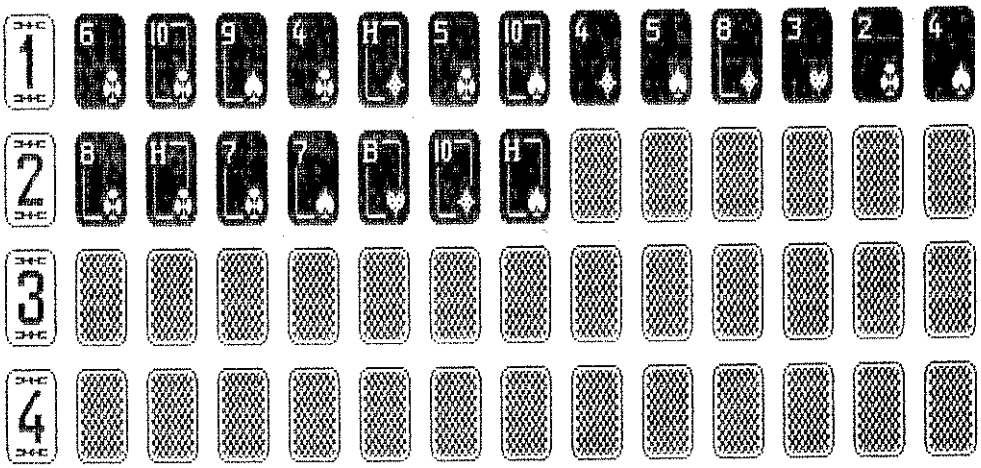
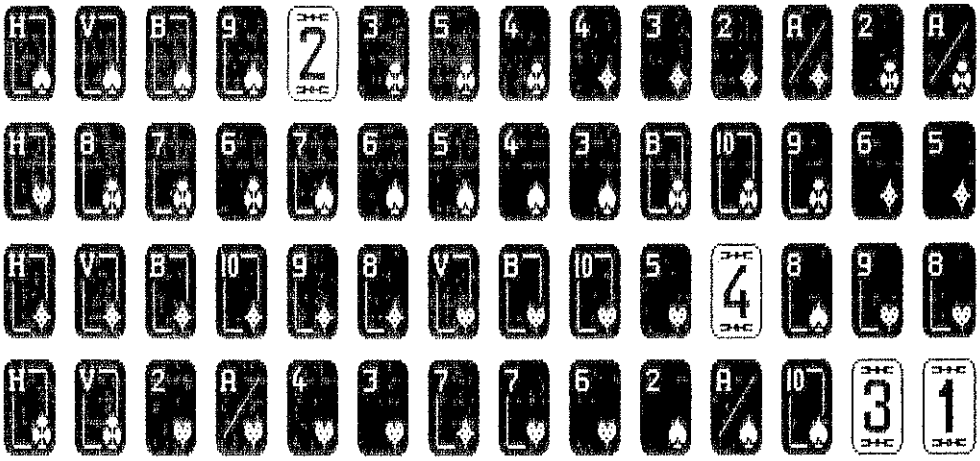
DATA 14





UITGAVE VAN DE HCC – DAI GEBRUIKERSGROEP

INHOUD DAITA 14

****

****

ADRESSEN:

Voorzitter : Kees Jagerman, Ilperveldstraat 77
1024 FJ Amsterdam
Tel. 020 - 367156

Secretaris a.i.
+ Hardware en Software : Theo Verberkt, Van Buerenstraat 13
5256 KL Oudheusden
Tel. 04162-2667

Penningmeester
+ Redaktie DAITA a.i. : Robert van den Broek, Melde 19
8265 CP Kampen
Tel. 05202 - 17131

VAN DE REDAKTIE

Wanneer ik in mijn vakbladen het redactionele stukje lees dan denk ik bij me zelf simpel eigelijk hoe krijgen ze het voor mekaar of zitten ze soms ook dagen lang in de auto naar hun werk over de tekst na te denken. Het zal wel niet maar bij mij staat het zweet weer op het voorhoofd en reken maar dit maal van angst.
Als het lukt en je leest dit verhaal nog voor de volgende bijeenkomst dan heb ik gewoon weer geluk. Maar in onze hobby houd je pas op als moeders snachts de stekker er uit trekt maar bij de vrijgezellen onder ons als de volgels weer fluiten. Maar wij blijven het een hobby vinden zoals een ander helemaal maf is van de elfstedentocht.
Maar nu dus even serieus DAITA 14 is weer in elkaar gezet. Maar hij is wel erg magertjes. DAIers wat is er aan de hand waar blijft de copy. Zullen we maar aannemen dat ook de vorst in ons brains is genesteld en dat de volledige inspiratie verdwenen is. Maar goed het wordt al weer beter weer de temperatuur stijgt.

Buiten alle mogelijke zonnige dingen moet ik nu eerst even een wat ernstige probleem aansnijden.
Ik denk dat wij toch langzaam moeten gaan geloven dat de kaars van de DAI langzaam maar zeker aan het uitgaan is. Het is de laatse tijd ook wel bijzonder stil om ons heen. Zelfs de nieuw aangekondigde grote DAI wordt nog niet wereld kundig gemaakt.
Deze geluiden zijn dan ook niet alleen uit nederland maar wij weten al dat Belgie zich langzamerhand naar de MSX aan het ombuigen is. En nu heb ik een brief van de DAI club uit Duitsland onder ogen gekregen waarin men zich deze zelfde dingen gaat afvragen. Letterlijk schrijven ze "Um den club uber die jaherswende '86/87 hinaus zu retten".
Spreekt dus duidelijke taal. Maar spoedig wil ik de strijd niet opgeven want al eens eerder heb ik gezegd zolang hij het doet blijf ik door gaan.

We hebben nog een afspraak met de fabriek die wij een dezer dagen hopen in te lossen. Mogelijk ben ik dan weer positief.

Op korte termijn maken wij een afspraak met Belgie en Duitsland voor een een goed gesprek en te horen wat we toch gezamelijk nog kunnen ondernemen. Dus tot zover nog maar even gebogen over ons machientje met alle plezier van dien. Wij hebben wel eens moeilijker verhalen gehoord en overleefd.

Genoeg gezeurd wij hebben toch nog intersante artikelen in dit nummer.

Het artikel van Jan Wubben, die heeft ware monnikenwerk gedaan door een index van het firmware manual te maken. Een kort voorbeeld is hier afgedrukt. Op verzoek te bestellen tegen porto en copy kosten.

Ruud Muller heeft een prachtig artikel met programma gemaakt om de printer uit te schakelen. Eindelijk dat valse ratelen op het moment dat je hem niet nodig hebt voorbij.

Dan heb ik zelf nog een oud artikel uit de kast gehaald over sorteren. In de basic kursus van de afdeling Zwolle is het een aardige les. Probeer maar eens je krijgt er nog wat losse andere types bij kado.

IK wil maar zeggen veel plezier weer met onze GRANNY en de hobby

ROBERT.

F I D O

Allemaal wel eens van het HCC FIDO-systeem gehoord.

Een FIDO-systeem bestaat uit een IBM achtige micro van 256 KB en een floppy en een hard disk van 20 MB plus een modem die zelf bepaalt wat er aan de andere kant van de lijn zit.

Als je zelf in bezit bent van een modem die 300/300 of 1200/1200 aan kan dan heb je slechts een datacom programma nodig een de zaak is voor elkaar. GEEN viditel programma of modem kun je hier gebruiken.

Waar staat zo'n systeem opgesteld. Voorlopig bij een van de bestuursleden in een afdeling thuis. Dus mogelijk vlak bij je. In de HCC-Nieuwsbrief wordt er elke keer aandacht aan geschonken.

Nu treft het dat voor de afdeling zwolle de fido-node bij mij thuis staat. En het is de bedoeling dat de DAI-gg op deze node zijn thuis haven krijgt dus mooier kan het niet.

Het is nu onze bedoeling om de aktiviteiten zoals we die op viewdata deden nu naar fido te verleggen máár let op het is geen viditel met mooie plaatjes. Het is zuiver een berichten verkeer en om programma,s uit te wisselen.

Wij ,Zwolle draait nu ongeveer voor 70% , volgende week ben ik vrij van de kursus en dan ga ik hurry up aan de bak om het systeem piekfijn in de lucht te krijgen. Zelfs wil ik hierbij onze Belgische vrienden uit nodigen om ook eens een kijkje te komen nemen.

FIDO Kampen-1 is 24 uur per dag bereikbaar. Tussen 4 uur en 6 uur wordt het systeem opnieuw gestart en komt de hoofd node binnen om de eventuele te verzenden berichten over te nemen of achter te laten.

Ooksmorgens tussen 9 uur en 11 uur loop je de kans er niet bij te kunnen omdat ik aan het updaten ben.

Het telefoon nummer is 05202 - 24380.

In de volgende DAITA gaan we er dieper op in dan weten wij vast meer over programma,s en modems.

Toch eens bellen. Succes. robert.

DAI-gg ¥¥ S O F T W A R E B I B L I O T H E E K ¥¥ DAI-gg

NAAM	PRIJS CASS.	PRIJS DCR
¥ Viditel-pakket monitor-mlp/monitor-basic telesoftware tranciever	F 40,--	F 45,--
¥ DAI-dres	F 30,--	F 35,--
¥ Perfect Editor II	F 45,--	F 50,--
op KEN-DOS	F 55,--	
¥ Boekhouding (huishoud) boodschappenlijst	F 40,--	F 45,--
¥ Logic simulator 3D mathematic graphics	F 40,--	F 45,--
¥ DBMS (data base management)	F 30,--	F 35,--
¥ 6800 Cross-assembler	F 30,--	F 35,--
¥ RS232 communication programma Terminal-emulator	F 30,--	F 35,--
¥ Games tape 1: The vally (uitleg + spel) Advancer 1 en 2	F 30,--	F 35,--
¥ Postzegel-verzameling	F 30,--	F 35,--
¥ Programmotheek tapecontrol/demo's	F 30,--	F 35,--
¥ RAMBLE DAI-DOS KEN-DOS floppy		F 45,--
¥ SHIFT-OTHELLO-DRAWING	F 20,--	F 25,--
¥ DCR-DIRECTORY LISTING Gotische maandKalender FGT	F 30,--	F 35,--
¥ VLAGGEN-GRATEX-ANAGRAM	F 20,--	F 25,--

Voor bestelling en van software Kunt U zich wenden tot :

Theo Verberkt,
Van Buerenstraat 13
5256 KL Oud-Heusden
Tel:04162-2667.

INLEIDING BIJ DE INDEX OP HET DAI-FIRMWARE MANUAL.

Bij eerste inzage in het DAI-Firmware manual viel mij op, dat een index ontbrak. Als ik nog eens wilde naslaan, wat ik al eerder had gelezen, dan was het terugvinden soms erg moeilijk.
En toen ik wat zat te oefenen op een CP/M machine met het programma dBaseII, nam ik vast die Index maar ter hand, want al oefenend had ik dan al meteen wat klaar.
Het is een hele kluif geworden en als ik me alles goed had gerealiseerd, was ik er waarschijnlijk nooit aan begonnen.
De zaak ik zo opgezet, dat voor elke routine of subroutine een trefwoord is opgenomen.
De keuze van trefwoorden is altijd al een discutabele kwestie en door mijn tamelijke onbekendheid met de DAI zal het heel wat op aan te merken zijn.
Bij de trefwoorden heb ik mij soms moeten bedienen van afkortingen. Daarbij zijn in eerste instantie onnodige woorden weggelaten. Vervolgens zijn de klinkers weggelaten, te beginnen achteraan in de woorden. Daarna zijn de medeklinkers weggelaten, zodat lettergrepen ontbreken.(Ik heb overigens al gemerkt, dat ik niet al te consequent ben geweest.)
In de reeks trefwoorden heb ik Basic-functies en -commando's met hoofdletters gebruikt; dat had tot gevolg, dat zij bij het sorteren gegroepeerd vooraan kwamen te staan. (Register-manipulaties kwamen daar onbedoeld ook terecht)
Ik heb in mijn Manual de bladzijden met de hand genummerd; dat gaf wat meer houvast bij het terugzoeken. Ik kan iedereen aanraden dat ook te doen.
Ook de Summary van het Manual heb ik voorzien van de bladzijdenummers; bovendien heb ik hoofdstuknummers aangebracht.
Sommige hex-adressen en dus ook de dec-adressen komen dubbel voor; dat heeft te maken met het bank-switching systeem van de DAI.
Al met al levert het inzien van de Index enig idee van de opzet en geloof ik, dat het systeem bruikbaar is.

.....

Bij dit werk van jan wubben behoort de index van 26 kantjes

Een bijzondere waardering is wel op zijn plaats.

Indien er intresse in is dan is dit werk te bestellen tegen

copy en porto kosten. Vermoedelijke prijs zal +/- fl. 7.50 zijn.

INHOUDSOPGAVE.

SYMBOL TABLES

Er zijn nog meer symbol-tables, terwijl er verspreid door het Manual nog meer jump-tables staan. Hier zijn slechts de tables vermeld, die ook in de inhoudsopgave van het Manual staan.

Blz.	Trefwoord	Omschrijving	Hex. adres	Dec. adres	Hfdst.
48	ASC	part of run ASC (EACF)	CF7E	53118	4
57	CHECK	run basiccmnd CHECK	D2C3	53955	7
46	CLEAR	run CLEAR (part of,E6B5)	CEBB	52923	4
55	CLEAR	part of run CLEAR (0E6B5)	D214	53780	6
90	Common ON..code	common 'ON...' code	DF78	57208	13
97	DATA (not used)	DATA not used	E34A	58186	13
239	DCE init routin	DCE initialisation routine	EF90	61328	20
45	DIM	part of run DIM (E639)	CE5C	52828	4
54	EDIT	part of EDIT (2EEC0)	D1D1	53713	6
55	EDIT	part of EDIT (2EE7B)	D1FD	53757	6
182	HL = HL - DE	HL = HL - DE	E6F2	59122	17
56	LOAD	run basiccmd LOAD	D270	53872	7
66	LOADA	LOADA: evaluate program name	D678	54904	7
67	LOADA	part of LOADA	D6E5	55013	7
245	LOADA	LOADA (updated)	EE6B	61035	21
65	RANDOMISE	run part of RANDOMISE (0E40C)	D658	54872	7
	?un ABS	run basicfunction ABS	[illegible]	59984	13
	[illegible]OS	run basicfunction ACOS			13
		run basicfunction AL[illegible]			
		run basicfunct[illegible]			

RecNo.	blz.	Omschrijving	Hex. adres	Dec. adres	Hfdst.
00001	2	CONTENTS			
00002	3	interrupt vector routines			1
00003	3	bank switching area	0040	64	1
00004	3	utility work area	0047	71	1
00005	3	interrupt vector adresses	0062	98	1
00006	3	screen variables	0072	114	1
00007	4	mathematics working area	00D0	208	1
00008	5	basic variables	0100	256	1
00009	7	heap, program area, screen ram	02EC	748	1
00010	7	rom and cpu area	C000	49152	1
00011	8	i/o device adresses	F900	63744	1
00012	8	mathematics chip amd 9511	FB00	64256	1
00013	8	amd9511 mathematics chip	FB00	64256	1
00014	8	programable interval timer 8253	FC00	64512	1
00015	8	programable interval timer 8253	FC00	64512	1
00016	8	discrete i/o device adresses	FD00	64768	1
00017	9	programable peripheral interface 8255	FE00	65024	1
00018	9	programable peripheral interface 8255	FE00	65024	1
00019	9	ticc timer en interrupt controller 5501	FF00		
[illegible]	9	ticc:timer,interrupt controler 5501			
	11	math.utility entrypoints			
		math.package initialisation			
		[illegible]erlow error routine			
		[illegible]ent error			
		[illegible]error			

A D V E R T E N T I E

VERKRIJGBAAR TEGEN PORTOKOSTEN OF LEUKE PROGRAMMA,S

- DESIGNERS HANDBOEK VOOR DCE - KAARTEN
HANDBOEK ONTWIKKEL SYSTEMEN DCE-DM

BEL OF SCHRIJF NAAR: A. KLAASSEN
P.C. HOOFDLAAN 11
3705 AE ZEIST

==========

TE KOOP WEGENS STOPPEN VAN DE HOBBY:

DAI-48K COLOR	FL.	800,-
RGB - KAART	FL.	125,-
JOY STICKS	FL.	40,-
ALLE NUMMERS VAN DAINAMIC + DAITA	FL.	40,-

TE BEVRAGEN BIJ TH. WANDERS 01714 - 2807

==========

TE KOOP :
DAI 48K COLOR, BARCO KLEUREN MONITOR, MEMOCOM DCR
DAI DISCDRIVE 2x 320K, PHILLIPS AUDIO CASS. RECORDER
EPROM PROGRAMMER, ALL KABELS
30 ST. 5. 1/4 FLOPPY,S MET VEEL SOFTWARE OA. CP/M
80 DCR CASS. MET SOFTWARE
50 AUDIO CASS. MET SOFTWARE
ALLE DENKBARE SOFTWARE
ALLES SLECHT VOOR DE PRIJS VAN FL. 4000,-.
TE BEVRAGEN BIJ:
H.J.A. WANDERS, LIVINGSTONELAAN 808, UTRECHT, 030 - 883167.

Uitschakelen van de RS232-poort.

Rudy Muller

Iedereen die iets aan de RS232-poort heeft hangen zit met het probleem dat DAI-basic de uitvoer zowel naar het scherm als de RS232-poort stuurt. Dit is te sturen door het uitvoer-controle byte (locatie #131) te wijzigen in 1 (alleen output naar scherm), echter zodra er een "soft-reset" plaats vindt (bv na het intoetsen van de BREAK-key) keert het uitvoer-controle byte weer terug naar zijn standaard waarde.
Vooral bij het ontwikkelen van programma's wordt de BREAK-toets nogal eens gebruikt (bv ter onderbreking van een LIST, of ter beeindiging van een EDIT). Ik ondervond het telkenmale terugkeren naar 0 van byte #131 als uitermate hinderlijk, en ben op zoek gegaan naar oplossing voor dit probleem: te weten een kleine assembler-routine die controle uitoefend op het controle-byte.

Het probleem bestond vooral uit het vinden van een plaats om de aanroep van de controle-routine te kunnen inpassen; die plaats moet namelijk aan twee voorwaarden voldoen:
1. Het moet (gezien de tijd) liggen tussen de "soft-reset" en de eerstvolgende uitvoer naar het scherm.
2. Het moet (gezien de plaats) ingrijpen in het systeem op een RAM-locatie.

Punt 2 beperkt het aantal mogelijkheden, en de voor de hand liggende kandidaten zijn de RST-interrupt-verctoren op de adressen #00 .. #38. Het ontwerp van de interrupt-routine afhandeling is er zelfs op gebouwd om omleggingen mogelijk te maken, het adres van de routine in ROM staat in RAM (voor RST-5 op locatie #006C). Een omlegging naar een privé routine kan dan worden gemaakt de oorspronkelijke wijzer te laten verwijzen naar de privé-routine, en door de privé-routine zelf te laten eindigen met een sprong naar de oorspronkelijk ROM-routine.
De RST-5 interrupt voldoet ook aan de eerste eis, voor er scherm-uitvoer wordt gepleegd komt het systeem hier altijd langs. Een nieuw probleem kwam er voor in de plaats, want het wijzigen van de wijzer kan niet mbv POKE's in BASIC worden gedaan, omdat in de tijd tussen het POKEn van de twee bytes een timer-interrupt kan komen. Dat kan weer resulteren in een cursor-flash, die via RST-5 wordt geregeld; er vindt dan een sprong naar een half afgemaakt adres plaats met alle gevolgen vandien. In het programma DISABLE wordt dit mbv een assembler-routine verwezenlijkt.

Het volgende probleem is een veilige plaats voor de controle-routine; ik heb voor een flexibele oplossing gekozen door de gebruiker te laten kiezen:
- op (de bodem van) de HEAP
- op (de zolder van) de STACK
- ergens anders

Als voor de HEAP wordt gekozen, zorgt DISABLE er voor dat de HEAP-pointer wordt aangepast, zodat een volgend BASIC-programma de zaak niet in het honderd kan laten lopen.
Als voor de STACK wordt gekozen, wordt de ruimte in de STACK in feite 18 bytes kleiner; en wat gemener is: een STACK-overflow heeft de zaak al verziekt voor dat er een waarschuwing komt !!

Het laatste probleem is hoe de uitvoer-mogelijkheid via de RS232-poort te herstellen. Dit is gerealiseerd door een nieuw controle-byte in te voeren, maar dan op een plaats die niet wordt verziekt door een BREAK. Ik heb gekozen voor het ongebruikte deel van de RST-5 interrupt-vector, te weten byte #002F (ofwel byte 47). De RS232-uitvoer kan uitgeschakeld worden door een POKE 47,1. De RS232-uitvoer kan weer worden ingeschakeld door een POKE 47,0 gevolgd door ofwel een BREAK, ofwel door een POKE #131,0.

Het programma DISABLE zelf is zo ruim mogelijk opgezet, waaruit naar believen een op maat gesneden programma kan worden afgeleid, zoals is gedaan met het

programma $USER.

Na het geven van een summiere uitleg over de controle (regel 1030..1050) wordt gekeken of er al een omlegging van de RST-5 heeft plaatsgevonden (regel 1110). Als dat zo is wordt er geen verdere actie ondernomen (regel 1120).
Na het laden van de assembly-routines in een INT-array (regel 1130..1140), wordt de gebruiker gevraagd waar de controle-routine moet komen te liggen (regel 1210..1224).
De controle-routine wordt ter bestemde plekke geMOVEd (regel 1240), en het adres in de jump-instructie wordt aangepast (regel 1310..1330). Als gekozen was voor de HEAP wordt de heap-pointer opgehoogd (regel 1410..1420). Tenslotte wordt de omleiding gemaakt door het adres van de oorspronkelijke RST-5 routine in de controle-routine te plaatsen (regel 1510) en het adres van de controle-routine aan de RST-5 te verbinden (regel 1520).

De RS232-controle routine zelf is betrekkelijk eenvoudig (zie 1800-1818):
Na het redden van register A wordt de inhoud van byte #0131 bekeken; is die inhoud gelijk aan 0 (uitvoer naar scherm èn RS232), dan wordt die vervangen door de inhoud van byte #002F. Register A wordt hersteld, en de oorspronkelijke RST-5 ROM-routine wordt geactiveerd.

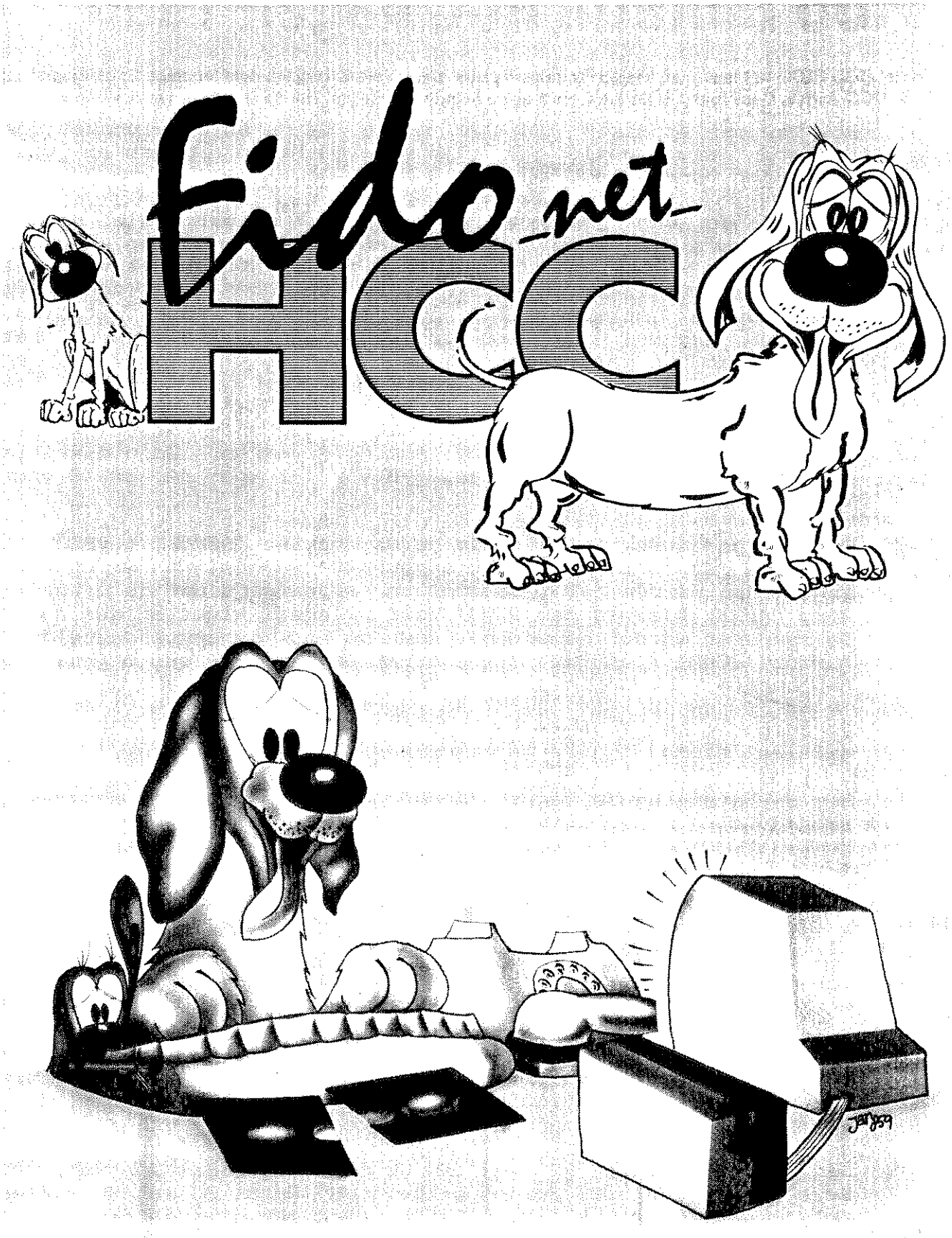
```
!
*LIST
100   REM * * * * * * * * * * * * * * * * * * * *
101   REM *                        *  DAI  V2.1  *
102   REM *   D I S A B L E        * - - - - - - - *
103   REM *                        *  17 dec 85  *
104   REM * * * * * * * * * * * * * * * * * * * *
105   REM *
106   REM *   DISABLE gives full control on the
107   REM *   RS232-output. The control-byte on
108   REM *   #131 is replaced by a byte on 47,
109   REM *   and is not influenced by BREAK !!
110   REM *
111   REM * * * * * * * * * * * * * * * * * * * *
112   REM *
113   REM * (c) Rudy Muller
114   REM *     Jan Steenstr 4
115   REM *     7412 TC   Deventer
116   REM *     tel: 05700 - 18667
117   REM *
1000  CLEAR 1000
1010  COLORT 0 11 0 0
1020  POKE #131,1
1030  PRINT "CONTROL RS232-output:"
1040  PRINT "-  enable by POKE 47,0"
1050  PRINT "-  disable by POKE 47,1"
1100  REM
1101  REM load DISABLE-routine
1102  REM
1110  R=256*PEEK(#6D)+PEEK(#6C)
1120  IF R<>#C6FD THEN END
1130  DIM D(7):D=VARPTR(D(0))
1140  FOR X=0 TO 7:READ D(X):NEXT X
1200  REM
1201  REM locate DISABLE-routine
1202  REM
1210  PRINT "Where must routine be located ?"
1211  PRINT " 1. on the HEAP (#"+HEX$(D)+")"
1212  PRINT " 2. on the STACK (#F800)"
1213  PRINT " 3. on given address (#....)"
1214  C=GETC:IF C=0 GOTO 1214
1220  IF C=ASC("1") OR C=ASC("H") THEN R=D:GOTO 1230
1221  IF C=ASC("2") OR C=ASC("S") THEN R=#F800:GOTO 1230
1222  IF C<>ASC("3") THEN END
1223  INPUT "GIVE ADDRESS :";R
1224  IF R=0 THEN END
1230  PRINT "DISABLE will be located on #"+HEX$(R)
1240  FOR I=0 TO 17:POKE R+I,PEEK(D+I):NEXT
1300  REM
1301  REM relocate JNZ-address
1302  REM
1310  V=R+PEEK(R+6)+256*PEEK(R+7)
1320  V2=VARPTR(V)+2:V3=V2+1
1330  POKE R+6,PEEK(V3):POKE R+7,PEEK(V2)
1400  REM
1401  REM adjust start-of-heap
1402  REM
1410  IF R<>D GOTO 1500
```

```
1420  V=D+18:POKE #29C,PEEK(V2):POKE #29B,PEEK(V3)
1500  REM
1501  REM redirect RST-5 routine
1502  REM
1510  POKE R+16,PEEK(#6C):POKE R+17,PEEK(#6D)
1520  CALLM D+20,D
1530  POKE 47,1
1540  END
1800  REM
1801  REM disable routine
1802  REM
1810  DATA #F53A3101,#B7C20E00,#3A2F0032,#3101F1C3,0
1811  REM F5      PUSH PSW
1812  REM 3A3101  LDA  #0131
1813  REM B7      ORA  A
1814  REM C20C00  JNZ  *+9
1815  REM 3A2F00  LDA  47
1816  REM 323101  STA  #0131
1817  REM F1      POP  PSW
1818  REM C30000  JMP  <adr>
1900  REM
1901  REM redirect routine
1902  REM
1910  DATA #D5232356,#235EEB22,#6C00D1C9
1911  REM D5      PUSH DE
1912  REM 2323    INX  H (2*)
1913  REM 56      MOV  D,M
1914  REM 23      INX  H
1915  REM 5E      MOV  E,M
1916  REM EB      XCHG
1917  REM 226C00  SHLD #006C
1918  REM D1      POP  DE
1919  REM C9      RET


*POKE 47,1
```

```
!
*LIST
100    REM * * * * * * * * * * * * * * * * * * * *
101    REM *                        *  DAI  V2.1  *
102    REM *   $ U S E R            * - - - - - - - *
103    REM *                        *  17 dec 85  *
104    REM * * * * * * * * * * * * * * * * * * * *
105    REM *
106    REM *   Tiny version of DISABLE to control
107    REM *   RS232-output. The control-routine
108    REM *   is located on the heap.
109    REM *   Control-byte #131 replaced by #2F.
110    REM *
111    REM * * * * * * * * * * * * * * * * * * * *
112    REM *
113    REM * (c) Rudy Muller
114    REM *     Jan Steenstr 4
115    REM *     7412 TC   Deventer
116    REM *     tel: 05700 - 18667
117    REM *
1000   CLEAR 1000
1010   IF PEEK(#6D)()#C6 THEN END
1020   DIM D(7):D=VARPTR(D(0)):R=D
1030   FOR X=0 TO 7:READ D(X):NEXT X
1040   V=R+PEEK(R+6)+256*PEEK(R+7)
1050   V2=VARPTR(V)+2:V3=V2+1
1060   POKE R+6,PEEK(V3):POKE R+7,PEEK(V2)
1070   V=D+18:POKE #29C,PEEK(V2):POKE #29B,PEEK(V3)
1080   POKE R+16,PEEK(#6C):POKE R+17,PEEK(#6D)
1090   CALLM D+20,D
1100   POKE 47,1
1110   END
1120   DATA #F53A3101,#B7C20E00,#3A2F0032,#3101F1C3,0
1130   DATA #D5232356,#235EEB22,#6C00D1C9
*POKE 47,1
```

De Hobby Computer Club presenteert:



S O R T E R E N

In een micro is een lijst met de volgende vijf namen opgenonen
Otto, Frits, Paul, Petra en Corrina.
Hoe kan men bereiken dat de micro de namen op alphabetiche volgorde eruit haalt. Er zijn vele methoden waarvan er vier hieronder worden besproken.

Het ordenen van gegevens komt in het commercieele data gebruik erg veel voor. Getallen en woorden willen we graag op een rijtje hebben. Computers zijn in staat dit voor ons op een makkelijke en snelle wijze te doen. De computers maken van de letters uit de woorden interne (kode) getallen. Men kan zeggen: bij het sorteren gaat het altijd om het vergelijken van getallen.

De moderne microcomputers zijn bijna allemaal voorzien van een eigen basic-dialect, allen kunnen logische vergelijkingen tussen woorden (alphabetich (string)) maken. Zo'n operatie levert een duidelijk resultaat. B.v. '-1 voor JA' en '0 voor NEE'.

Geef de computer het volgende bevel: "print 'APPEL' < 'BOOM'". Zodra de return toets wordt ingedrukt verschijnt er op het scherm de waarde '-1'. De computer signaleert dat de uitdrukking "APPEL kleiner dan BOOM" niet helemaal klopt,en dus door de computer wordt bevestigd. Doe nu het tegenovergestelde "print 'BOOM' < 'APPEL' ", nu drukt de computer een 0 (nul) op het scherm. De begin letter "A" van "APPEL" heeft de waarde '65' en de 'B' van "BOOM" de waarde '66'. Op deze wijze beslist de computer dat de bewering "BOOM" kleiner dan "APPEL" niet helemaal klopt.

Met de bovenstaande kennis kunnen we al een sorteer programma schrijven.

```
10 FOR I=1 TO 5
20 L=0
30 FOR K=1 TO 5
40 L=L- (A$(I)) >= A$(K))
50 NEXT K
60 B$(L)=A$(L)
70 NEXT I
```

Ga regel voor regel na, vul de variabelen eventueel in met een waarde en probeer de logika eruit te halen.
Bij regel 10 begint een zogehete programma lus, wat deze programma lus ons laat zien zijn een aantal basic instructies die begint bij regel 10 "FOR" en eindigt bij regel 70 "NEXT I". Deze lus zal 5 maal worden uitgevoerd. Regel 20 zet de waarde "L" op nul. Er zal bij elk vijfde woord een plaats in de lijst worden aangewezen. Op regel 30 begint een nieuwe lus, en deze

wordt ook 5 maal herhaald. De belanrijkste aanwijzing staat in regel 40. Om deze te begrijpen moet men eerst de uitdrukking vertalen. Daarna gaat het erom een woord uit de lijst te vergelijken met dezelfde woorden uit de lijst. Daar in de oorspronkelijke lijst de naam OTTO op de eerste plaats staat is hij de eerste in de rij. Vijf maal probeert de computer of OTTO groter of gelijk is met een van de andere woorden in de lijst. Belangrijk is nu alleen, dat bij iedere vergelijking steeds weer de vergelijking '-1' of de vergelijking '0' te voorschijn kan komen. Deze vergelijking wordt dan van de onder 'L' opgeslagen waarde afgetrokken. Trekt men voor een getal '-1' af dan is dat hetzelfde wanneer bij '+1' een getal opteld. Ondanks dat er verminderd word, neemt toch de waarde van 'L' toe. Bij iedere vergelijking (het eerste woord is groter dan het tweede of gelijk aan het tweede), neemt 'L' steeds met '1' toe. In regel 60 gebruikt de computer de zo ontstane waarde van 'L' om het oude woord met de naam in "A$(I)" de nieuwe naam in "B$(L)" toe te wijzen.
In ons voorbeeld: de naam Otto is groter of gelijk aan Otto, Frits, Corrina. 'L' is daarom aan het einde van de vergelijkingsronde 3, uit "A$(I)" moet "B$(L)" komen. Feitelijk staat Otto in de alfabetisch geordende namenlijst onze 5 voornamen op de derde plaats. De dubbellus is belangrijk en zwaar genoeg om eens stap voor stap door te nemen.

Eerste vergelijking: I=1

```
A$(I)="OTTO"
K=1
A$(K)="OTTO"
"OTTO" is gelijk aan  "OTTO"
L=L-(-1)
0-(-1)=1
```

Tweede vergelijking: I=1

```
A$(I)="OTTO"
K=3
A$(K)="FRITS"
"OTTO" is groter dan "FRITS"
L=L-(-1)
1-(-1)=2
```

Derde vergelijking: I=1

```
A$(I)="OTTO"
K=3
A$(K)="PAUL"
"OTTO" is kleiner dan "PAUL"
L=L-0
2-0=2
```

Vierde vergelijking: I=1

```
A$(I)="OTTO"
```

```
K=4
A$(K)="CORRINA"
"OTTO" is grotert dan "CORRINA"
L=L-(-1)
2-(-1)=3
```

Vijfde vergelijking: I=1

```
A$(I)="OTTO"
K=5
A$(K)="PETRA"
"OTTO" is kleiner dan "PETRA"
L=L-0
3-0=3
```

Daarmee staat vast dat het eerste woord in onze oorspronkelijke namenlijst op de derde plaats in de nieuwe alfabetische geordende lijst wordt geplaats. Toch is het werk nog lang niet gedaan. Vier volgende woorden wachten ons nog, zodat ook hun plaats in de alfabetische lijst wordt gevonden. De computer zet het zoeken voort. Nadat de binnenste lus 5 maal doorlopen is, wordt de buitenste lus met 1 verhoogd en wordt de lus herhaald.

Zesde vergelijking: I=2

```
A$(I)"FRITS"
K=1
A$(K)="OTTO"
"FRITS" is kleiner dan "OTTO"
L=L-0
0-0=0
```

Zevende vergelijking: I=2

```
A$(I)="FRITS"
K=2
A$(K)="FRITS"
"FRITS" is gelijk aan "FRITS"
L=L-(-1)
0-(-1)=1
```

Het is niet nodig, verder vergelijkingen door te nemen.De computer gaat verder tot dat de buitenste lus met de "kontrolevariable" I vijf maal rond is geweest. Nu heeft ieder van de vijf namen in de lijst zijn plaats, en wij kunnen ons resultaat laten afdrukken. Daarvoor zijn in ons voorbeeld programa nog drie regels nodig, namelijk:

```
80 FOR J=1 TO 5
90 PRINT B$(J)
100 NEXT J
```

Voor het geval dat je vingers al jeuken en je wilt zien of dit programma ook daadwerkelijk loopt moet eerst nog de data in

het programma worden gevoegd. Dit geschiedt met de volgende programma regels:

```
1 FOR M=1 TO 5
3 READ A$(M)
5 NEXT M
7 DATA OTTO, FRITS, PAUL,
        PETRA, CORRINA
```

Deze sorteermethoden zijn niet de enige methoden. Men kan ook zeggen dat bovengenoemde methode door zijn eenvoud bestaat. Trouwens er moet bijvermeld worden dat het de huidige micro-computergebruiker bij het programmeren van sorteerhandelingen vergelijkingswijze gemakkelijker heeft.
Vroeger had men bij grote computers vaak zo weinig interne opslagplaats ter beschikking, dat men "extra" sorteren moest. Wat verstond men daar onder?. De gesorteerde data's bevonden zich niet meer in de computer, maar op magneetbanden, en dat sorteren bestond er uit dat die data op die banden met zowel kunstig geprogrammeerde programma's gemixt moesten worden , maar dat het bovendien erg langzaam ging.
Maar de ontwikkeling staat niet stil en de huidige computers kunnen nu meer presteren en hebben die 'extra' sorteermethoden hun betekenis practisch verloren.

We houden ons nu bezig op drie andere belangrijke methoden van "intern" sorteren. We willen bijvoorbeeld de volgende acht getallen sorteren: 9,7,3,5,8,10,6,1.

Eerste methode:

Sorteren door invoegen.

De computer neemt als start de beide eerste getallen voor zich Daar het tweede getal, 7, kleiner is dan, 9, worden de zeven en negen omgedraaid. Dat gaat als volgt: 7,9,3,5,8,10,6,1.
Nu wordt het derde getal, hier een drie, passend in de linker (al gesorteerde) "volgdeel", die uit zeven en negen bestaat, ingevoegd. Dat gaat als volgt: 3,7,9,5,8,10,6,1.
Nu komt het vierde getal uit de rij die in de goede al gesorteerde 3 getalen ingepast moet worden. Bij iedere stap wordt de volgorde, van de al gesorteerde getallen, steeds groter totdat de stappen van onze volgorde van 8 getallen gesorteerd is. Het zoeken met deze methode laat zich nog versnellen wanneer men de "tussen te voegen waarde" niet eerst helemaal vanaf links, maar in het midden van het al gesorteerde deel. Men kan dan de "linker buurman" met het vergelijkende getal vergelijken. Is deze kleiner, dan behoeft alleen nog maar naar rechts gekeken te worden en kunnen de overige getallen, die links staan, voorlopig worden vergeten.

Tweede methoden :

Sorteren door uitkiezen.

Men kijkt als eerste welk getal in de rij de kleinste is. Deze worden op de eerste plaats gezet, en de tot nu kleinste getal neemt de plaats in van dat andere getal dat naar voren is gegaan. Bijvoorbeeld: 1,7,3,5,8,10,6,9. De 1 en de 9 hebben van plaats geruild. De volgende stap: 1,3,5,7,8,10,6,9. De zeven ruilde met de vijf van plaats. En zo vergelijken we steeds weeropnieuw to alle waarden zijn geweest. Het is niet nodig de procedure tot aan het eind te volgen.

Derde methode:

Sorteren door ontleden.

Deze "sortering", ontwikkeld door C.A.R. Hoare, heeft zo'n prestatiesterkte dat het de titel "Quicksort" verdiend. De prestatiesterkte werd natuurlijk sterk bij verkoop aanbevolen, men had nu bij het programmeren een zeer komplete einden verwerking.
Bij deze methoden wordt een "informatie getal" uitgekozen, die in het midden van de rij moet liggen. In onze getallenrij komt de vijf naar voren: 9,7,3,5,8,10,6,1. Nu zoekt men de getallen links van de vijf af. Stoot men op een getal die groter of gelijk is als 5, hier 7, beeindigd hij het zoeken. Nu gaat de computer naar rechts kijken. Dit keer gaat het om het eerste getal, die kleiner of gelijk is aan vijf. De 1 word gevonden. De 1 (rechts van de vijf, kleiner als vijf) wordt met de zeven (links van de vijf, groter als de vijf) omgeruild.

Met deze handeling (naar twee richtingen zoeken) en daarna omruilen, gaat men zolang verder totdat men bij het doorzoeken van links en van rechts elkaar treft.
Voorbeeld: 1,3,5,7,8,10,6,9.
Men ziet: alle getallen links van de vijf zijn nu kleiner dan deze, en alle getallen rechts van de vijf zijn nu groter. Nu wordt er een nieuwe indeling gemaakt. Men zoekt een streefgetal in de rij van een tot vijf en een ander streefgetal in de van zeven tot negen. Het proces wordt zo lang verder gevoerd tot elke van de ontstane "deelkettingen" nu alleen nog een schakel heeft.

Daar de "quiksort" bijzonder moeilijk te programmeren is, vind je aan het eind van dit verhaal een programma, dat je voor je eigen micro zonder al te grote moeite kan aapassen, omdat het BASIC-commando's bevat, die in alle BASIC-dialecten wel voorhanden zijn. De quicksort werd eerst in 1971 openbaar gemaakt, en wel in Computer Journaal 14, nr.4. Sindsdien is er in de computertechniek heel wat veranderd.
De moeilijkste en snelste sorteerhandelingen worden minder belangrijker naar maten de computer sneller kan rekenen. Het mag zijn, dat de langzame maar toch eenvoudiger sorteermethoden betere toekomstkansen hebben.

Typ het onderstaande programma in

Snelste sorteermethoden: QUICKSORT

```
10 REM QUICKSORT
30 DIM A$(100)
40 ?CHR$(12):CLEAR 5000
50 INPUT "Hoeveel woorden worden er gesorteerd?";WT
55  IF  WT>100  THEN  PRINT  "INPUT  GROTER DAN 100":GOTO  40.
60 FOR I=1 TO WT
70 INPUT A$(I)
80 NEXT I
90 M=1:I=1:J=WT
100 IF I>=J THEN 470
110 K=I
120 ZZ=INT ((J+I)/2)
130 T$=A$(ZZ)
140 IF A$(ZZ)<=T$ THEN 180
150 A$(ZZ)=A$(I)
160 A$(I)=T$
170 T$=A$(ZZ)
180 L=J
190 IF A$(J)>=T$ THEN 300
200 A$(ZZ)=A$(J)
210 A$(J)=T$
220 T$=A$(ZZ)
230 IF A$(I)<=T$ THEN 300
240 A$(ZZ)=A$(I)
250 A$(I)=T$
260 T$=A$(ZZ)
270 GOTO 300
280 A$(L)=A$(K)
290 A$(K)=ZZ$
300 L=L-1
310 IF A$(L)>T$ THEN 300
320 ZZ$=A$(L)
330 K=K+1
340 IF A$(K)<T$ THEN 330
350 IF K<=L THEN 280
360 IF (L-I)>=(J-K) THEN 420
370 L1(M)=I
380 U1(M)=L
390 I=K
400 M=M+1
410 GOTO 510
420 L1(M)=K
430 U1(M)=J
440 J=L
450 M=M+1
460 GOTO 510
470 M=M-1
480 IF M<1 THEN 630
490 I=L1(M)                          : 590 IF K=S THEN 610
500 J=U1(M)                          : 600 IF T$<A$(K)  THEN
510 IF (J-I)>10 THEN 110             :         A$(K+1)=A$(K) :
520 IF I=1 THEN 100                  :         K=K-1 :GOTO 590
530 I=I+1                            : 610 A$(K+1)=T$
540 S=1                              : 620 GOTO 550
550 I=I+1                            : 630 FOR I = 1 TO WT
560 IF I=J THEN 470                  : 640 PRINT A$(I)
570 T$=A$(I+1)                       : 650 NEXT I
580 K=I                              : 660 END
```

Bovenstaande verhaal en programma is overgenomen uit COMPUTERHEFT van MAART 1984.

Onderstaande sorteerroutines uit CHIP van JULI 1981

LINEAR - SORT

```
10  REM **********************
20  FOR I = 2 TO N
30  A(0) = A(I)
40  J = I + 1
50  IF A(0) >= A(J) THEN 90
60  A(J+1) = A(J)
70  J = J + 1
80  IF J <> 0 THEN 50
90  A(J+1) = A(0)
100 NEXT I
```

AUSWAHL - SORT

```
10 REM **********************
20 I=0: J=0: M=0
30 FOR I = 1 TO N -1
40 M = I
50 FOR J = I + 1 TO N
60 IF A(J) < A(M) THEN M=J
70 NEXT J
80 A(0) = A(I): A(I) = A(M)
             : A(M) = A(0)
90 NEXT I
```

SHAKE - SORT

```
10  REM **********************
20  L=2: R=N: K=N
30  FOR J = R TO L STEP -1
40  IF A(J-1) <= A(J)
                    THEN 70
50  A(0) = A(J-1):
           A(J-1) = A(J):
              A(J) = A(0)
60  K=J
70  NEXT J
80  L = K + 1
90  FOR J=L TO R
100 IF A(J-1) <= A(J)
                  THEN 130
110 A(0) = A(J-1):
           A(J-1) = A(J):
           A(J) = A(0)
120 K = J
130 NEXT J
140 R = K - 1
150 IF L <= R THEN 30
```

HEAP - SORT

```
10  REM **********************
20  I = N - 1: L = 2: R = N
30  FOR J = R TO L STEP - 1
40  IF A(J-1) <= A(J)
                  THEN 70
50  A(0) = A(J):
           A(J) = A(J-1):
                A(J-1) = A(0)
60  I = J
70  NEXT J
80  L = I + 1
90  FOR J = L TO R
100 IF A(J-1) <= A(J)
                  THEN 130
110 A(0) = A(J):
           A(J) = A(J-1):
           A(J-1) = A(0)
120 I = J
130 NEXT J
140 R = I - 1
150 IF L <= R THEN 30
```

BUBBLE - SORT

```
10 REM **********************
20 FOR I = 2 TO N
30 FOR J = N TO I STEP -1
40 IF A(J-1) > A(J)
                THEN 60
50 GOTO 70
60 A(0) = A(J):
          A(J) = A(J-1):
          A(J-1) = A(J)

70 NEXT J
80 NEXT I
```

RIPPLE - SORT

```
10 REM **********************
20 C=0
30 FOR J = 1 TO N - 1
40 IF A(J) <= A(J+1)
              THEN 60

50 A(0) = A(J):
          A(J)=A(J+1):
          A(J+1) = A(J):
                   C = 1
60 NEXT J
70 IF C = 1 THEN 20
```

SHELL - SORT

```
10  REM **************************************
20  M = N
30  M = INT(M/2)
40  IF M = 0 THEN 170
50  J = 1
60  K = N - M
70  I = J
80  L = I + M
90  IF A(I) < A(L) THEN 140
100 A(0) = A(I): A(I) = A(L): A(L) = A(0)
110 I = I - M
120 IF I < 1 THEN 140
130 GOTO 80
140 J = J + 1
150 IF J <= K THEN 70
160 GOTO 30
170 REM **************************************
```

Schrijf een programma om de volgende getallen te sorteren.

1,5,9,2,4,22,36,55,11,7,21,45,54,12.

Druk het af door invoeging van de print opdracht.

2K-RAM-Erweiterung im Bereich :F000-:F7FF

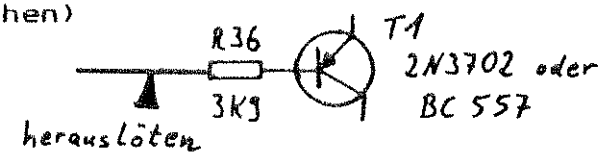
Die RAM-Erweiterung eignet sich dazu um timing-abhängige Programme auszutesten, die später auf 2716 EPROM's "gebrannt" werden sollen.
Die Hardwareänderung wirkt sich so aus, daß der DAI dann über einen 2048+256=2304 Byte Stack verfügt. Allerdings wird kein ordnungsgemäßer Stack Overflow angezeigt, wenn der Stack-Pointer :EFFF erreicht.

Für den Umbau benötigen Sie :

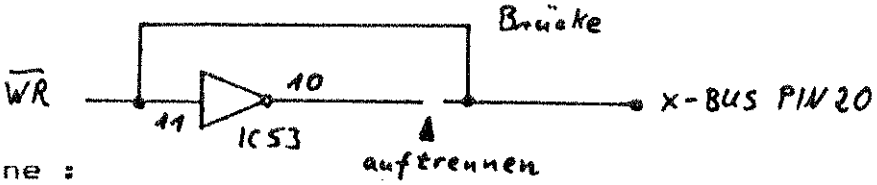
- EPROM-Platine (siehe Layout)
  incl. aller Bauteile ohne EPROM 32 DM+Mwst. --> 36.48 DM
  Erhältlich bei DAMM & JOHANNING
  4800 Bielefeld 1
  Sudbrackstr. 46/48
  Tel.:0521/83036
  Platine wird auf den X-Bus gesteckt
- CMOS-RAM 6116 pinkompatibel zu EPROM 2716 ca. 23.50 DM
- 10 KOhm Widerstand 0.10 DM
- ein Stück isolierten Draht ---
- ein scharfes Messer 60.08 DM
- Lötkolben (30 Watt) ===

Folgende Änderungen sind durchzuführen :

1. Widerstand R36 an einem Ende herauslöten.
   Effekt : Stack Overflow abgeschaltet
   (evtl. schaltbar machen)

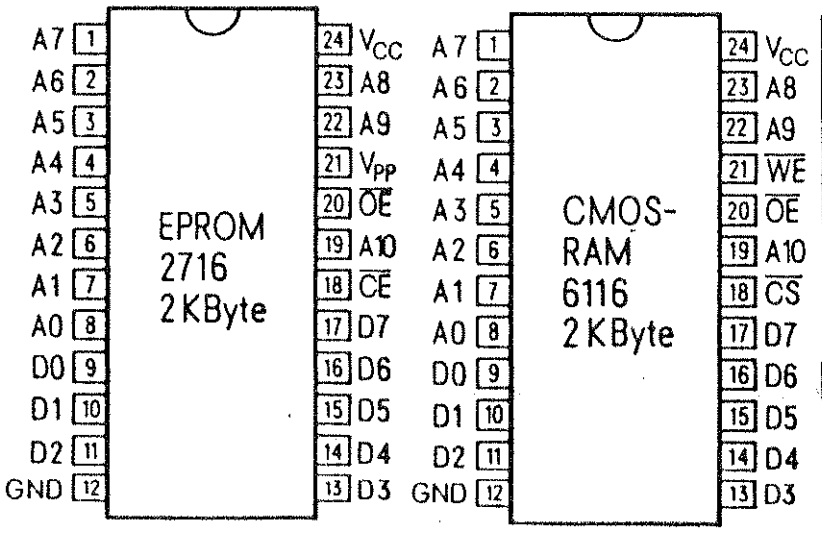


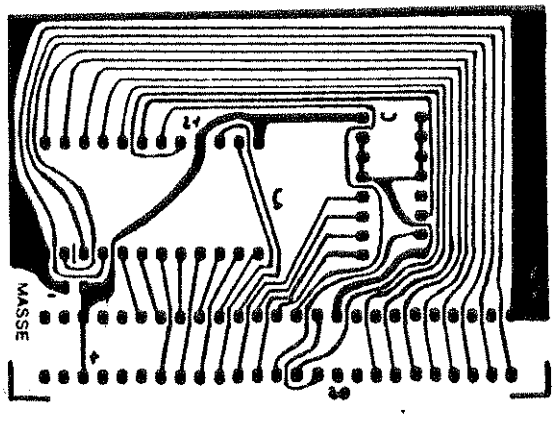
2. Am IC53 :
   Leiterbahn hinter PIN 10 auftrennen (Platinenoberseite).
   Brücke legen zwischen PIN 11 und "hinter" PIN 10
   (Unterseite). Effekt : Aus WR wird $\overline{WR}$



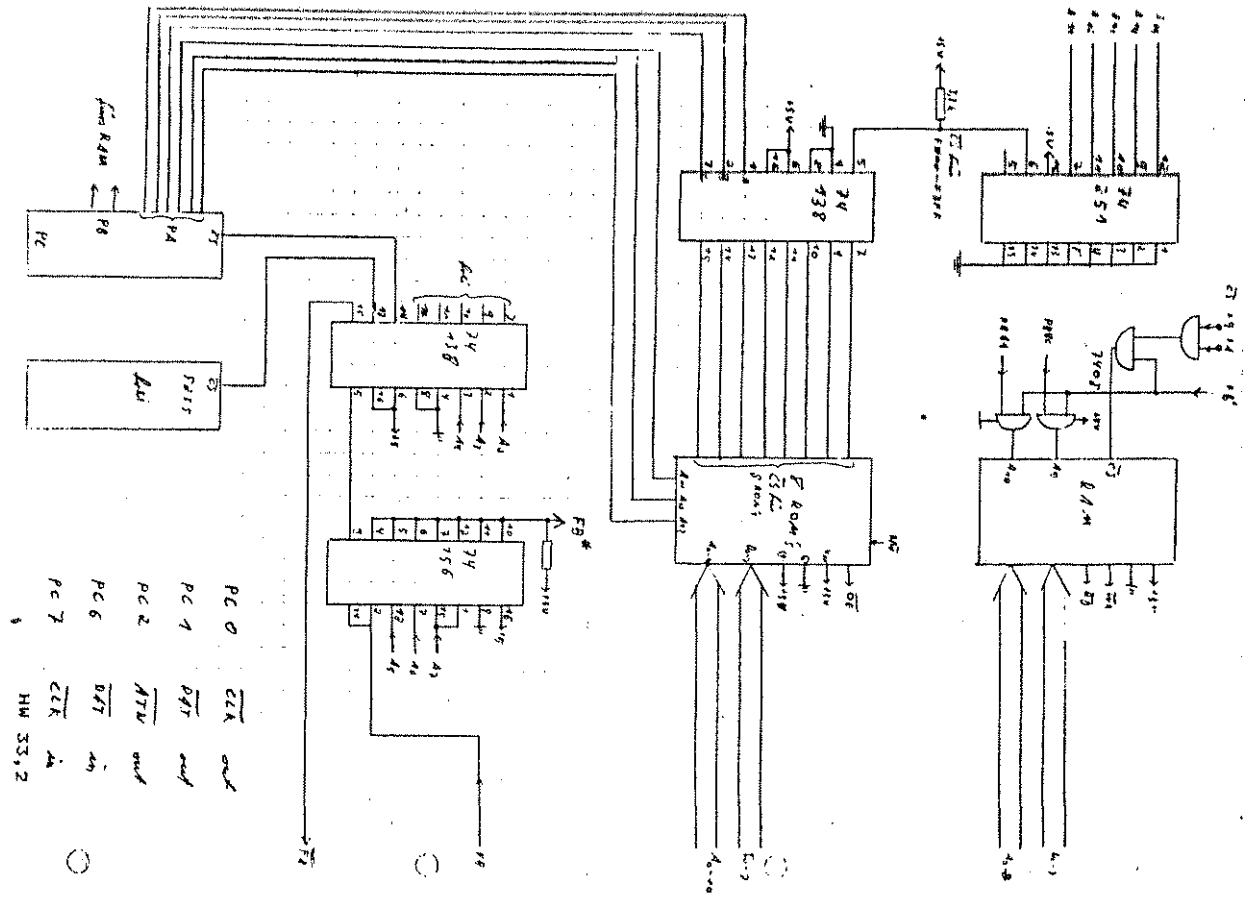
3. Auf der EPROM-Platine :
   PIN 21 vom EPROM freilegen, d.h. Leiterbahnen beidseitig auftrennen und "überbrücken".
   PIN 21/EPROM mit PIN 20/X-Bus über einen 10K Widerstand verbinden.

Um die Bauteile auf der Platine zu lokalisieren verwende man den Bauteilelageplan HW3.

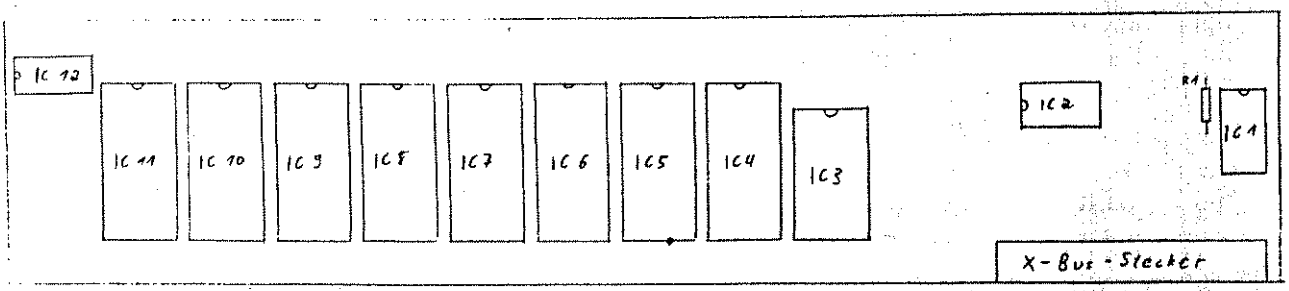


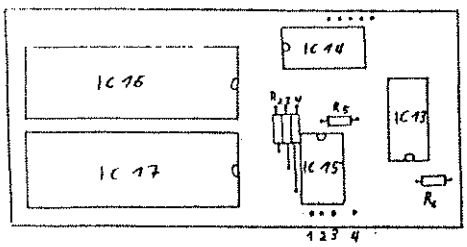


HW 23



BESTÜCKUNGSPLAN





HW 33,3

1 = $\overline{ATN}$
2 = $\overline{DAT}$
3 = $\overline{CLK}$
4 = $\overline{RES}$

IC 1	74LS 151 od. 74LS 251
IC 2	74LS 08
IC 3	6116
IC 4-11	2764, 27128 od. 6264
IC 12/14	74 LS 138
IC 13	74 LS 156
IC 15	74LS(04), 05 od. 06
IC 16/17	8255

Aufgrund der Tatsache, dass im DAI zunaechst nur 2KB fuer Betriebssystemerweiterungen (DAIDOS,TOS,COMDOS,MCU,...) zur Verfuegung stehen , kam die Idee auf, im Bereich von #F000 bis #F7FF mehrere 2KB-Bloecke unterzubringen, und diese mit einer Bankswitchschaltung umzuschalten . Die dabei entstehenden Probleme (z.B. benoetigt man einen nicht umschaltbaren Bereich , von dem aus umgeschaltet wird) wurden folgendermassen geloest:

1. Es existieren 64 Bloecke zu je 2KB im Bereich von #F000 bis #F7FF, die mit Hilfe eines 8255 umgeschaltet werden. Diese Bloecke koennen EPROMs und/oder RAMs sein.
2. Im Bereich von #F900 bis #FAFF sind vier Bloecke zu je 0.5KB RAM untergebracht, die ebenfalls mit dem 8255 umgeschaltet werden.
3. Der Bereich von #FB00 bis #FBFF wurde folgendermassen aufgeteilt:

```
FB00-FB03 : Mathe Prozessor
FB04-FB07 : 8255 zum Umschalten der Banks (Port A fuer die
            2KB Bloecke ,Port B fuer die 0.5KB Bloecke, Port
            C fuer die Commodore Floppy )
FB08-FB0B : 8255 fuer freie Verwendung (z.B. Anschluss von
            MDCR oder DAI-Floppy , damit der DCE-Bus wieder
            frei wird)
FB0C-FB0F : CS (neg.) zur freien Verwendung
FB10-FB13 :            "
FB14-FB17 :            "
FB18-FB1B :            "
FB1C-FB1F :            "
FB20-FBFF : Hier liegt eine Kopie des Bereiches F920-F9FF
            (von RAM-Bank0), damit man einen Bereich hat,der
            nie ausgeblendet ist.
```

Die Arbeitsweise der Hardware ist aus dem Schaltplan ersichtlich . Zur Umschaltung der Banks muessen die Ports A und B des 8255 als Ausgaenge geschaltet werden. Mit einem POKE#FB04,Banknummer*4 wird die EPROM-Bank (2KB) Nummer 'Banknummer' eingeschaltet. Mit POKE#FB05,Banknummer*64 wird die RAM-Bank (0.5KB) Nummer 'Banknummer' eingeschaltet.

In die EPROM/RAM Fassungen koennen IC's der Typen 2764,27128 und 6264 verwendet werden .
Die gesamte Elektronik ist bei mir auf zwei Platinen untergebracht, die aneinandergeschraubt sind. Diese werden dann einfach auf den X-Bus gesteckt. Weiterhin wird ein IC-Sockel auf ein IC gesteckt (auf IC45 ,CS F900-FA00) und ein IC-Sockel in die Fassung des Mathe Prozessors. Hier kann auch eine kleine weitere Platine mit einem Wire-Wrap-Sockel aufgesetzt werden , damit der Mathe Prozessor weiterhin verwendet werden kann.
Fuer Interessenten gibt es die Moeglichkeit die Platinen bei mir zu bestellen. Die Kosten belaufen sich auf ca. DM 110,00 fuer eine getestete Platine (ohne IC's aber mit allen IC Sockeln , gedrehte fuer die EPROMs). Welche IC's benoetigt werden ist auf dem Bestueckungsplan angegeben .

Norbert Rinnen
Moerikestr. 10
D-5060 Bergisch-Gladbach 1
Tel. 02204/81963

BESTELLING DAI-GG-SOFTWARE

Te zenden aan Th. Verberkt, Van Buerenstraat 13, 5256 KL Oud-Heusden.

Ondergetekende : _____

Naam : _____

Adres : _____

Postcode/woonplaats: _____

bestelt hierbij de volgende software:

_____ prijs f _____

_____ ,, ,, _____

_____ ,, ,, _____

_____ ,, ,, _____

totaal: f _____

Het totaal-bedrag heeft hij overgeschreven op postrekening 5314900, t.n.v. Th. Verberkt te Oud-heusden.

Handtekening:

BESTELLING DAI-GG-SOFTWARE

Te zenden aan Th. Verberkt, Van Buerenstraat 13, 5256 KL Oud-Heusden.

Ondergetekende : _____

Naam : _____

Adres : _____

Postcode/woonplaats: _____

bestelt hierbij de volgende software:

_____ prijs f _____

_____ ,, ,, _____

_____ ,, ,, _____

_____ ,, ,, _____

totaal: f _____

Het totaal-bedrag heeft hij overgeschreven op postrekening 5314900, t.n.v. Th. Verberkt te Oud-heusden.

Handtekening: