## BASIC V1.0 – V1.1

(from DAInamic 11, page 209)

For some time now there have been two versions of DAI-BASIC, V1.0 and the newer V1.1. As far as possible DAI have corrected the bugs, at least the ones they knew about, in the V1.0 ROMs. By dint of experiment with the two ROM sets I have tried to locate the differences - some of this information I obtained directly from DAI. Here are my provisional findings.

1 V1.1 reacts differently on a RUN line-number. In V1.0 all the variables would be cleared but that does not happen with the V1.1 and is a major benefit of the new ROM; now debugging BASIC programs is simpler. For example, while running a program it is now possible to break, to list and even to change directly the value of variables before letting the rest of the program continue.

2 DIM (255,255) is permitted with V1.1

3 Sound channel 2 will be switched off by a hard reset; that was never guaranteed with V1.0 so there was always a possibility that the sound would return when switching on again

4 SAVEA and LOADA now work correctly (see previous DAInamic newsletters)

5 The bug has been removed from the SGN function

6 A minor bug has also been cleared from the EDIT mode. This fault did not have any serious consequences and was reported by only two users. (under some circumstances the cursor disappeared for a while)

7 The basic operator '–' is correctly recognised in V1.1 so that it is no longer necessary to put brackets around the negative expression.

8 There was a bug in the TALK command but I do not know what it was.

9 GETC is now debounced. The keyboard buffer is cleared first.

10 Returning from the EDIT mode it is now unnecessary to key immediately RUN

11 The TAB function now works correctly, ie, independently of the serial channel (printer) if that was not switched on.

12 There was also an incorrect error message given sometimes, such as ERROR IN LINE .... without a line number.

The foregoing differences between the two ROMS are those I know about. If anyone wants a list of those differences in hexadecimal format he can obtain it from Jos Schepens for the cost of copying and postage. Hopefully this short review will enable you to judge if the new ROMs are good value. I do not know what they will cost but I have an idea that the price will be fairly high.

## DAI pc's communicate via public telephone lines.

This article will give you a detailed description how to communicate (TALK), directly via your screen, exchange electronic mail, program-listings or object files, from one DAI to another, by using a low cost (acoustical) modem.
To achieve this, the following steps have to be taken:

a) Load the machine language program, named DAICOM (DAI's COMmunicate) and start it with UT-Z3-0400.
   Not CALLM #400 !!
b) Connect your modem to the RS232 at the rear of the DAI.
c) Make your telephone-call with the other DAI-user,who should also have completed steps a and b.

Assuming you can hear each other loud and clear, as usually on a Dutch line, first make the decision who works in Half/Full-duplex (always in the opposit mode), before switching over to the modem.

The advantage of working in full-duplex is that you can see if something goes wrong during data transmission, because the typed character is first sent to the receiver, who echo's it, whereafter it comes on your own screen.
After this choise is made press both <T> and TALK via the key-board.
Typing cursor-up displays the MENU again and one of the other features can be selected.

### The choise of the modem.

The modem (MOdulate DEmodulate) should comply the CITT V21 standard i.a.w.:

- Full duplex operation.
- 300 baud.
- 2 switch-able channels (each DAI works on a different channel).

The best choise is a modem which is directly connected to the telephone-line, but these types are generally expensive and must have (in Holland) PTT approval.
A good alternative is a cheaper acoustic modem.
We personally have good experience with a modem (kit), offered by the firm NENIJWA in EDE (Holland) telephone number 08380-10856.
With just a little understanding of basic electronics and a good soldering iron (small tip), you should be able to assemble the kit in a couple of hours.
A minor disadvantage of such a modem is that it easily picks-up background noise, but a piece of foam could help reduce this.

### What are the possibilities of working with DAICOM

The most interesting feature is that you can either transmit and receive both ASCII and HEX (object) files.

### Transmit files.

For ASCII, all that is (or moved) in the editbuffer can be transmitted.
You can for example type a message in the edit-mode <E> and afterwards send it away.

### Sending a BASIC program

After DAICOM is loaded and started go to <B> BASIC and simply LOAD a program from tape or DCR (heap pointers are adjusted automatically).
Type : -CLEAR XXXX
-EDIT
-BREAK
-BREAK
The program is in the editbuffer and ready for transmitting.

### Send DNA source listing

DNA source files (bufferdim. 16), can directly be sent by typing <S> in the menu.

### Send HEX files

To transmit a hex-file first the start and end address of the data-block is required, before the transmission starts.

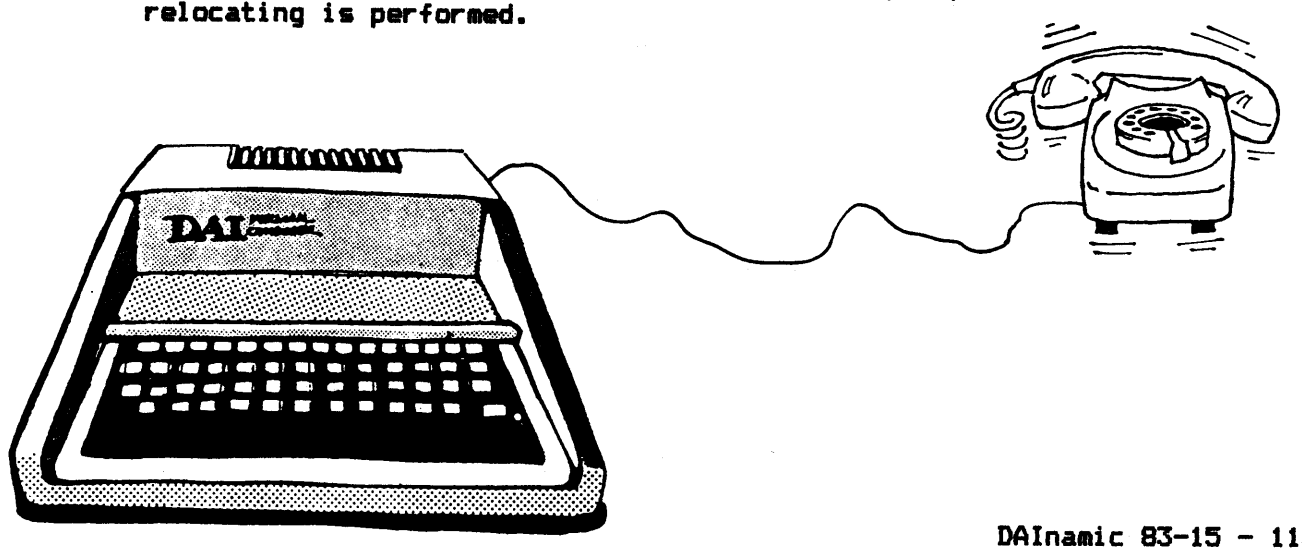
It is important to note, that you can always see on the screen what is coming in or going out.

### Receive files

All incoming data is stored in the (default #3000-#6FFD) buffer.If more space is required ,go to basic and give CLEAR XXXX, EDIT, BREAK, BREAK.

Received basic programmes are ready for LIST or RUN after POKE #135,2 is performed.

If an object file is received, the start and end address of the received file are given and the relocate (start) address is asked.
After converting the ASCII characters in HEX bytes, relocating is performed.



DNA source files can be saved on tape or DCR by first finding the end adddress of the file.
Edit buffer pointer A4-A5 gives the end address (LLHH).
The start address is always #3000.
Go to UT and Display >DA2 A4

Example:the address 3277 is displayed as 77 32

Now typing W3000 HHLL and save the file on tape or DCR.
If DNA is loaded type R. and the file will be read as a normal source file.

## How to work via the menu.

First load the program.

1) UT
2) Z3
3) R
4 G400

Now the menu is displayed and the following options can be selected.

<E> ENTER EDIT MODE:
    This facility can be used to prepare a letter (mini-word processor).
    Also received ASCII files will come automatically in this mode and can easily be edited.
    Escape with BREAK
<H> HALF-DUPLEX (DCE data computer):
    The receiver of files (ASCII or HEX) comes automatically in this mode.
    The program accepts the received character stores it in the (edit)buffer, echo's it back to the transmitter and displays it on the screen.
    During "TALKING" via the key-board, the typed characters are both send to the screen as well to the RS232 (2).
<F> FULL-DUPLEX (DTE data terminal):
    Full-duplex is automaticaly selected during transmitting files.
    Characters are first sent to the receiver (other DAI in receive mode), who echo's the character before you get it back on the screen.
<T> TALK
    In this mode, both can see the typed characters, this means that you can "TALK" via the key-board.
    One should be in full-duplex, whilst the other is in half-duplex.
    With BREAK to MENU.
<S> SEND DNA SOURCE:
    The assembler source listing is transmitted and shown on the screen. At the end or if you hit BREAK, again initialisation is performed.
    This command has a dual function which is usefull in case a basic program has been edited and again the default buffer is required.
<A> SEND ASCII FILE:
    The content of the editbuffer is transmitted and displayed on the screen.
    End of file character followed by the checksum is transmitted.

<#> SEND HEX FILE:
The program asks the (hex) start and end address of the file and start transmitting immediately after the end address is given.
<R> RECEIVE ASCII FILE:
Received characters are displayed on the screen and stored in the editbuffer.The end of text character (3) followed by the checksum are not displayed.
If the checksums are different,in the cursor appears an "F" and the program stays in Receive mode.
Escape with BREAK, and enter the Edit-mode.
<H> RECEIVE HEX FILES:
Similar to <R>, but displays the start and end address of the rceived file and ask the new start address of the file (relocates the file).
<U> BACK TO UTILITY:
<B> BACK TO BASIC:

## Some practical notes.

You have now read the whole article and typed or loaded the programme in the DAI
At this stage you can start playing with it even without modem.
Just take a piece of wire, strip it and connect 2 and 3 of the RS232 (see manual).Be not afraid, all in/outputs on this connector are buffered.
All the "send" commands can be exercised (full-duplex).

Another possibility is if there are 2 DAI's.
In this case,couple them by making the following connections.

| DAI no.1 RS232 pin | | DAI no.2 RS232 pin |
|---|---|---|
| 1o | ---------- | o1 |
| 2o | ---------- | o3 |
| 3o | ---------- | o2 |

1=ground 2=serial out 3=serial in

The same rules apply as if working with a modem.

Finally I would like to thank Noud Rynaerts and Leo v.d.Laak for their valuable comments and suggestions and the extensive "field tests".

If you have problems or questions call me on extension 04780-84180 after 18.00 hrs.

Success,

Ger Gruiters
Laurahof 12
5801 JE Venray

DAICOM will be available on TOOLKIT 5, to be released soon.

```
PAGE 01

001                     *DAICOM V3.3 BY G.GRUITERS 16-1-1983
002                     *
003                     BDRATE  EQU   :84        300 BAUD 1 STOP-BIT
004                     STSREG  EQU   :FFF3      STS-REG IN TICC
005                     INCHAR  EQU   :FFF0      SER INP BUFF
006                     STAEB   EQU   :A2        ADDR START EDIT-BUFFER
007                     EBPTR   EQU   :A4        ADDR END OF TEXT IN EBUFF
008                     ENDEB   EQU   :A6        ADDR END EBUFF
009                     *
010                             ORG   :400       ENTRY PROGRAM
011 0400 CD2304                 CALL  START      INITIALISE
012 0403 C3E604                 JMP   SPECL      DISPLAY MENU
013                     *
014                     *       TALK
015                     *
016 0406 CD6B04         MAINLP  CALL  CHKSTS     CHECK IF CHAR RECEIVED
017 0409 C47104                 CNZ   INTR1      IF YES,HANDLE/DISPLAY
018 040C CDBED6                 CALL  :D6BE      KEY PRESSED?
019 040F DAE604                 JC    SPECL      IF BREAK MENU
020 0412 C4AD04                 CNZ   TRANS      YES,HANDLE/TRANSMIT
021 0415 C30604                 JMP   MAINLP     CONTINUE LOOP
022                     *
023 0418 3E40           TOUT    MVI   A,64       BAUDRATE 9600
024 041A 3205FF                 STA   :FF05
025 041D 3100F9                 LXI   SP,:F900   RESTORE STACKPNTR
026 0420 C309E0                 JMP   :E009      BACK TO UTILITY
027 0423 3A2C08         START   LDA   FLAG1      CHECK IF THIS IS
028 0426 FE01                   CPI   1          THE 1ST PASS,IF NOT
029 0428 CA5404                 JZ    ENDWST     SKIP INIT.
030 042B 219A0A                 LXI   H,ENDPRG   SET START HEAP AFTER
031 042E 229B02                 SHLD  :29B       END PROGRAM
032 0431 210001                 LXI   H,:100     ADJUST HEAP SIZE
033 0434 229D02                 SHLD  :29D
034 0437 CDB8DE                 CALL  :DEB8      RNEW
035 043A 210030                 LXI   H,:3000    SET-UP EBUFF PNTR'S
036 043D 22A200                 SHLD  STAEB
037 0440 21FD6F                 LXI   H,:6FFD
038 0443 22A600                 SHLD  ENDEB
039 0446 CD5F04                 CALL  CLEBUF
040 0449 210000                 LXI   H,0
041 044C 22B400                 SHLD  :B4        NO TABS
042 044F 3E01                   MVI   A,1        SET FLAG INIT DONE
043 0451 322C08                 STA   FLAG1
044 0454 3E84           ENDWST  MVI   A,BDRATE
045 0456 32F5FF                 STA   :FFF5
046 0459 3E01                   MVI   A,1        TO SCREEN ONLY
047 045B 323101                 STA   :131
048 045E C9                     RET
049 045F 2AA200         CLEBUF  LHLD  STAEB      CLEAR EBUFF
050 0462 EB                     XCHG
051 0463 21FF6F                 LXI   H,:6FFF
052 0466 AF                     ZAR
053 0467 CD7CDE                 CALL  :DE7C


PAGE 02

054 046A C9                     RET
055 046B 3AF3FF         CHKSTS  LDA   STSREG
056 046E E608                   ANI   8          CHAR RECEIVED ?
```

```
057 0470 C9                RET
058                 *
059                 * HANDLE RECEIVED CHARACTERS
060                 *
061 0471 3AF0FF    INTR1   LDA   INCHAR     GET RECEIVED CHAR
062 0474 FE7F              CPI   127         NOT VALID IF MSB=1
063 0476 F0                RP
064 0477 FE08      BACKSP  CPI   8           BACKSPACE
065 0479 CA8404            JZ    SCREEN
066 047C FE0D              CPI   13          CAR-RET
067 047E CA8404            JZ    SCREEN
068 0481 FE20              CPI   32          FILTER ALL CHAR'S BELOW 32
069 0483 D8                RC                EXCEPT 8 AND 13
070 0484 CD95D6    SCREEN  CALL  :D695       DISPLAY CHAR
071 0487 CD8F04            CALL  ECHO
072 048A AF                ZAR
073 048B 322F08            STA   FLAG4       CLEAR TALK FLAG
074 048E C9                RET
075 048F 57        ECHO    MOV   D,A
076 0490 3A2E08            LDA   FLAG3
077 0493 FE01              CPI   1
078 0495 C0                RNZ               ECHO ONLY IN HALF-DPL
079 0496 3A2F08            LDA   FLAG4
080 0499 FE01              CPI   1
081 049B C8                RZ                NO ECHO IF TALK FLAG IS SET
082 049C CDA304    WAIT2   CALL  WAIT        READY FOR TRANSMIT?
083 049F 32F6FF            STA   :FFF6       SEND CHAR VIA RS232
084 04A2 C9                RET
085 04A3 3AF3FF    WAIT    LDA   STSREG      MASK FOR "TRANSMIT
086 04A6 E610              ANI   16          BUFF EMPTY, IF NOT
087 04A8 CAA304            JZ    WAIT        TRY AGAIN
088 04AB 7A                MOV   A,D
089 04AC C9                RET
090                 *
091                 * HANDLE CHARACTERS TO TRANSMIT
092                 *
093 04AD F5        TRANS   PUSH  PSW
094 04AE 3E01              MVI   A,1
095 04B0 322F08            STA   FLAG4       SET TALK FLAG
096 04B3 F1                POP   PSW
097 04B4 57        TRANSC  MOV   D,A
098 04B5 CDA304            CALL  WAIT
099 04B8 FE10              CPI   16          IF CURSOR-UP
100 04BA CAE604            JZ    SPECL       DISPLAY MENU
101 04BD FE08              CPI   8           (SEE LABEL BACKSPACE)
102 04BF CACD04            JZ    OUT
103 04C2 FE0D              CPI   13
104 04C4 CACD04            JZ    OUT
105 04C7 FE7F              CPI   127
106 04C9 F0                RP


PAGE 03

107 04CA FE20              CPI   32
108 04CC D8                RC
109 04CD 7A        OUT     MOV   A,D
110 04CE 32F6FF            STA   :FFF6       SEND VIA RS232
111 04D1 3A2E08            LDA   FLAG3
112 04D4 FE01              CPI   1           HALF-DPLX?
113 04D6 CAE204            JZ    HDUPLX
114 04D9 3A3108    SUMCHA  LDA   CHKSUM      LOAD OLD CHECKSUM
115 04DC AA                XRA   D           UPDATE
116 04DD 07                RLC
117 04DE 323108            STA   CHKSUM      STORE NEW CHECKSUM
118 04E1 C9                RET
```

```
119 04E2 7A        HDUPLX  MOV   A,D
120 04E3 C37704            JMP   BACKSP
121                *
122                *       MENU
123                *
124 04E6 21E0DD    SPECL   LXI   H,:DDE0   RESTORE INP SUBR
125 04E9 22D200            SHLD  :D2
126 04EC AF                ZAR              ZERO FLAGS
127 04ED 322F08            STA   FLAG4
128 04F0 323008            STA   FLAG5
129 04F3 3E5F              MVI   A,:5F      NORMAL CURSOR
130 04F5 327500            STA   :75
131 04F8 216308            LXI   H,MSG2     DISPLAY MENU
132 04FB CDD4DA            CALL  :DAD4
133 04FE CDBED6    CHOISE  CALL  :D6BE      WAIT FOR CHOISE
134 0501 FE45              CPI   69         E
135 0503 CA2107            JZ    INITEB
136 0506 FE48              CPI   72         H
137 0508 CA4C05            JZ    HALFD
138 050B FE46              CPI   70         F
139 050D CA5405            JZ    FULLD
140 0510 FE54              CPI   84         T
141 0512 CA3B05            JZ    TALK
142 0515 FE41              CPI   65         A
143 0517 CA6305            JZ    SASCI
144 051A FE23              CPI   35         #
145 051C CA5B05            JZ    SHEX
146 051F FE53              CPI   83         S
147 0521 CAE005            JZ    SSTEB
148 0524 FE52              CPI   82         R
149 0526 CA6D05            JZ    RECASC
150 0529 FE58              CPI   88         X
151 052B CA7705            JZ    RECHEX
152 052E FE55              CPI   85         U
153 0530 CA1804            JZ    TOUT
154 0533 FE42              CPI   66         B
155 0535 CA4405            JZ    TOBAS
156 0538 C3FE04            JMP   CHOISE
157 053B 213C08    TALK    LXI   H,MSG1
158 053E CDD4DA            CALL  :DAD4
159 0541 C30604            JMP   MAINLP
```

PAGE 04

```
160 0544 3E40      TOBAS   MVI   A,64
161 0546 32F5FF            STA   :FFF5
162 0549 C3A0C7            JMP   :C7A0      BACK TO BASIC
163 054C 3E01      HALFD   MVI   A,1
164 054E 322E08            STA   FLAG3      SET HALF-DUPLX
165 0551 C3E604            JMP   SPECL
166 0554 AF        FULLD   ZAR
167 0555 322E08            STA   FLAG3      SET FULL-DUPLX
168 0558 C3E604            JMP   SPECL
169 055B 3E01      SHEX    MVI   A,1
170 055D 322D08            STA   FLAG2      SET HEX
171 0560 C35107            JMP   EDIT2      SEND HEX FILE
172 0563 AF        SASCI   ZAR
173 0564 322D08            STA   FLAG2      SET ASCII
174 0567 CD8205            CALL  SMSG
175 056A C3AF05            JMP   EDIT       SEND ASCII FILE
176 056D AF        RECASC  ZAR
177 056E 322D08            STA   FLAG2
178 0571 CD9805            CALL  RMSG
179 0574 C30106            JMP   READ       RECEIVE ASCII FILE
180 0577 3E01      RECHEX  MVI   A,1
```

```
181 0579 322D08            STA   FLAG2
182 057C CD9805            CALL  RMSG
183 057F C30106            JMP   READ       RECEIVE HEX FILE
184 0582 21CB09     SMSG   LXI   H,MSG3
185 0585 CDD4DA            CALL  :DAD4      MESSAGE"SEND"
186 0588 CDAA05            CALL  CLCHKS
187 058B AF                ZAR
188 058C 322E08            STA   FLAG3
189 058F 3AF0FF            LDA   INCHAR     CLEAR INP BUFF
190 0592 1682              MVI   D,:82
191 0594 CD9C04            CALL  WAIT2      SEND START CHAR
192 0597 C9                RET
193 0598 21EB09     RMSG   LXI   H,MSG4
194 059B CDD4DA            CALL  :DAD4      MESSAGE"RECEIVE"
195 059E CD5F04            CALL  CLEBUF
196 05A1 CDAA05            CALL  CLCHKS
197 05A4 3E01              MVI   A,1
198 05A6 322E08            STA   FLAG3
199 05A9 C9                RET
200 05AA AF         CLCHKS ZAR               ZERO CHECKSUM ADDR.
201 05AB 323108            STA   CHKSUM
202 05AE C9                RET
203                 *
204                 *  SEND EDIT-BUFFER
205                 *
206 05AF 3A2D08     EDIT   LDA   FLAG2       IF HEX FLAG SET,JUMP
207 05B2 FE01              CPI   1           TO HEX SEND ROUTINE
208 05B4 CA5107            JZ    EDIT2
209 05B7 2AA200     SDNAS  LHLD  STAEB       GET 1ST CHAR FROM EBUFF
210 05BA 22A400     NEXTCH SHLD  EBPTR
211 05BD 7E                MOV   A,M
212 05BE FE00              CPI   0           END OF DATA?
```

PAGE 05

```
213 05C0 CAD205            JZ    CHKFL1
214 05C3 E67F              ANI   127         STRIP MSB (DNA SOURCE)
215 05C5 CD0E08            CALL  TRCDEL      TRANSMIT CHAR
216 05C8 23                INX   H           INCR. EBUFF PNTR
217 05C9 CDBED6            CALL  :D6BE       BREAK PRESSED?
218 05CC DAD205            JC    CHKFL1
219 05CF C3BA05            JMP   NEXTCH      NEXT CHAR
220 05D2 3A2C08     CHKFL1 LDA   FLAG1       IF FLAG IS SET THEN
221 05D5 FE00              CPI   0           AGAIN INITIALISATION
222 05D7 C24207            JNZ   STAY1       SEND END OF TEXT & CHECKSUM
223 05DA CD2304            CALL  START
224 05DD C34207            JMP   STAY1
225                 *
226                 *  SEND DNA (16)SOURCE VIA (EDIT)BUFF.
227                 *
228 05E0 3A5912     SSTEB  LDA   :1259
229 05E3 FE00              CPI   0           DNA SOURCE AVAILABLE?
230 05E5 CAEB05            JZ    SSOURC
231 05E8 C3D205            JMP   CHKFL1      IF NOT,TO MENU
232 05EB CD8205     SSOURC CALL  SMSG
233 05EE AF                ZAR
234 05EF 322C08            STA   FLAG1       NEW INIT. REQUIRED
235 05F2 110070            LXI   D,:7000     DNA SOURCE START
236 05F5 2A5512            LHLD  :1255       PNTS TO DNA SYMB-TABLE
237 05F8 010030            LXI   B,:3000
238 05FB CD4FDE            CALL  :DE4F       MOVE SOURCE
239 05FE C3B705            JMP   SDNAS
240                 *
241                 *  TRANSFER TO (EDIT) BUFFER
242                 *
```

```
243 0601 0E00       READ    MVI   C,0       ZERO CNTR
244 0603 2AA200             LHLD  STAEB     START BUFFER
245 0606 22A400             SHLD  EBPTR     POINTS TO START
246 0609 CD4E06             CALL  LOOK3
247 060C DA7C06             JC    STOPTR
248 060F FE82               CPI   :82       START TXT?
249 0611 C20106             JNZ   READ      ELSE WAIT
250 0614 22A400     LOOK    SHLD  EBPTR     UPDATE PNTR
251 0617 CD4E06     LOOK2   CALL  LOOK3
252 061A DA7C06             JC    STOPTR
253 061D FE7F               CPI   127       NOT VALID IF MSB=1
254 061F F21706             JP    LOOK2
255 0622 FE03               CPI   3         END OF TEXT CHAR RECEIVED?
256 0624 CA5D06             JZ    TEST1     YES, GO TO TEST CHECKSUM
257 0627 FE0D               CPI   13
258 0629 CA3106             JZ    DISPL
259 062C FE20               CPI   32        FILTER <32, NOT CAR-RET
260 062E DA1706             JC    LOOK2
261 0631 77         DISPL   MOV   M,A
262 0632 CD60DD             CALL  :DD60     DISPL ON SCRN
263 0635 56                 MOV   D,M
264 0636 CDD904             CALL  SUMCHA    UPDATE CHECKSUM
265 0639 2AA600             LHLD  ENDEB     END OF BUFF REACHED?
```

PAGE 06

```
266 063C EB                 XCHG
267 063D 2AA400             LHLD  EBPTR
268 0640 CD14DE             CALL  :DE14
269 0643 CA7C06             JZ    STOPTR
270 0646 7E                 MOV   A,M
271 0647 23                 INX   H         INCR PNTR
272 0648 CD8F04             CALL  ECHO      RECEIVER (DCE) ECHO'S
273 064B C31406             JMP   LOOK      NEXT CHAR
274 064E B7         LOOK3   ORA   A         CLEAR CARRY
275 064F CDBED6             CALL  :D6BE     IF BREAK STOP
276 0652 D8                 RC              RECEIVING
277 0653 CD6B04             CALL  CHKSTS    CHECK FOR INCOMING
278 0656 CA4E06             JZ    LOOK3     CHAR'S
279 0659 3AF0FF             LDA   INCHAR
280 065C C9                 RET
281 065D CDBED6     TEST1   CALL  :D6BE
282 0660 DA7C06             JC    STOPTR    ABORT IF BREAK
283 0663 CD6B04             CALL  CHKSTS    LOOK IF CHECKSUM ARRIVES
284 0666 CA5D06             JZ    TEST1     WAIT IF NOT
285 0669 3AF0FF             LDA   INCHAR
286 066C 57                 MOV   D,A       STORE DTE CHKSUM IN REG
287 066D 3A3108             LDA   CHKSUM    DCE CHKSUM IN ACCU
288 0670 BA                 CMP   D
289 0671 CA7C06             JZ    STOPTR    JUMP IF CHKSUM OK
290 0674 3E46               MVI   A,70      NOT OK "F" IN CURSOR
291 0676 327500             STA   :75       AND STAY IN RECEIVE MODE
292 0679 C31706             JMP   LOOK2     ESCAPE WITH BREAK
293 067C 3A2D08     STOPTR  LDA   FLAG2
294 067F FE01               CPI   1         IF HEX,CONVERT & RELOCATE
295 0681 C22107             JNZ   INITEB    IF ASCII, TO EDIT-MODE
296 0684 2AA200             LHLD  STAEB
297 0687 223A08             SHLD  BYTE      START ADDR BYTE-PNTR
298 068A 22A400     CVTBUF  SHLD  EBPTR     ADJUST BUFF-PNTR
299 068D 7E                 MOV   A,M
300 068E FE00               CPI   0         LAST CHAR IN BUFF REACHED?
301 0690 CAB706             JZ    PRADDR    TO RELOCATE ROUTINE
302 0693 FE20               CPI   32        SKIP SPACE & CAR-RET
303 0695 CAB306             JZ    INCREB    AND INCR BUFF-PNTR
304 0698 FE0D               CPI   13
```

```
305 069A CAB306            JZ    INCREB
306 069D CDF006            CALL  HEXPAS    ASCII TO HEX CONVERSION
307 06A0 23                INX   H         FIRST PASS DONE
308 06A1 22A400            SHLD  EBPTR     INCR & ADJUST BUFF-PNTR
309 06A4 7E                MOV   A,M
310 06A5 CDF006            CALL  HEXPAS    SECOND PASS
311 06A8 2A3A08            LHLD  BYTE      STORE FINAL BYTE IN ADDR
312 06AB 77                MOV   M,A       POINTED BY BYTE-PNTR
313 06AC 23                INX   H         INCR BYTE-PNTR
314 06AD 223A08            SHLD  BYTE
315 06B0 2AA400            LHLD  EBPTR
316 06B3 23        INCREB  INX   H         INCR BUFF-PNTR
317 06B4 C38A06            JMP   CVTBUF    NEXT CHAR
318 06B7 21500A    PRADDR  LXI   H,MSG7
```

PAGE 07

```
319 06BA CDD4DA            CALL  :DAD4
320 06BD 2AA200            LHLD  STAEB
321 06C0 CD18ED            CALL  :ED18     PRINT START FILE
322 06C3 EB                XCHG
323 06C4 216C0A            LXI   H,MSG8
324 06C7 CDD4DA            CALL  :DAD4
325 06CA 2A3A08            LHLD  BYTE
326 06CD 2B                DCX   H
327 06CE CD18ED            CALL  :ED18     PRINT END FILE
328 06D1 21740A            LXI   H,MSG9
329 06D4 CDD4DA            CALL  :DAD4
330 06D7 CDC607            CALL  HEXINP    MOVE FILE TO ?
331 06DA 3A3008            LDA   FLAG5
332 06DD FE01              CPI   1         RELOCATE ABORTED?
333 06DF C2E506            JNZ   NBRK
334 06E2 2AA200            LHLD  STAEB     NO MOVE
335 06E5 44        NBRK    MOV   B,H       SWAP
336 06E6 4D                MOV   C,L
337 06E7 2A3A08            LHLD  BYTE
338 06EA CD4FDE            CALL  :DE4F     MOVE ROUTINE
339 06ED C3E604            JMP   SPECL
340 06F0 323208    HEXPAS  STA   STORE1    STORE 1ST CHAR BYTE
341 06F3 79                MOV   A,C
342 06F4 FE01              CPI   1         2ND CHAR OF BYTE DONE?
343 06F6 CA0807            JZ    PASS2
344 06F9 CD1307            CALL  CONVRT    GO TO CONVERT
345 06FC 07                RLC
346 06FD 07                RLC
347 06FE 07                RLC
348 06FF 07                RLC
349 0700 E6F0              ANI   :F0       MASK
350 0702 323308            STA   STORE2    STORE RESULT
351 0705 0E01              MVI   C,1       FIRST CHAR DONE
352 0707 C9                RET             FETCH NEXT CHAR
353 0708 CD1307    PASS2   CALL  CONVRT
354 070B 47                MOV   B,A
355 070C 3A3308            LDA   STORE2    GET PREVIOUS RESULT
356 070F B0                ORA   B         AND MAKE FINAL BYTE
357 0710 0E00              MVI   C,0       ZERO CNTR
358 0712 C9                RET             STORE BYTE IN BUFF
359 0713 3A3208    CONVRT  LDA   STORE1    LOOK IF RECEIVED ASCII
360 0716 FE3A              CPI   58        CHAR IS NUMBER OR
361 0718 D21E07            JNC   LETTER    LETTER AND ADJUST
362 071B D630              SUI   48
363 071D C9                RET
364 071E D637      LETTER  SUI   55
365 0720 C9                RET
366 0721 2AA200    INITEB  LHLD  STAEB
```

```
367 0724 7E        LKETXT  MOV   A,M       SEARCH FOR END OF
368 0725 FE00              CPI   0         TEXT IN EBUFF
369 0727 CA2E07            JZ    ADJPTR
370 072A 23                INX   H
371 072B C32407            JMP   LKETXT


PAGE 08

372 072E 23        ADJPTR  INX   H
373 072F 22A400            SHLD  EBPTR
374 0732 EF                RST   5         INIT SCRN EDIT-MODE
375 0733 2A                DATA  :2A
376 0734 CDBED6    GETC    CALL  :D6BE     WAIT FOR INPUT
377 0737 DAE604            JC    SPECL
378 073A CA3407            JZ    GETC
379 073D EF                RST   5         EDIT ON SCRN
380 073E 2D                DATA  :2D
381 073F C33407            JMP   GETC
382 0742 1603      STAY1   MVI   D,3
383 0744 CD9C04            CALL  WAIT2     SEND END  OF TEXT
384 0747 3A3108            LDA   CHKSUM
385 074A 57                MOV   D,A
386 074B CD9C04            CALL  WAIT2     SEND CHECKSUM
387 074E C3E604            JMP   SPECL
388                *
389                * SEND HEX FILE
390                *
391 0751 210E0A    EDIT2   LXI   H,MSG5
392 0754 CDD4DA            CALL  :DAD4
393 0757 CDC607            CALL  HEXINP    INPUT START ADDR
394 075A 223408            SHLD  STAFIL
395 075D 21300A            LXI   H,MSG6
396 0760 CDD4DA            CALL  :DAD4
397 0763 CDC607            CALL  HEXINP    INPUT END ADDR
398 0766 23                INX   H
399 0767 223608            SHLD  ENDFIL    IF WRONG INPUT,TRY AGAIN
400 076A EB                XCHG
401 076B 2A3408            LHLD  STAFIL
402 076E CD14DE            CALL  :DE14
403 0771 D25107            JNC   EDIT2
404 0774 1682              MVI   D,:82
405 0776 CD8205            CALL  SMSG
406 0779 110000            LXI   D,0       ZERO CNTR'S
407 077C 2A3408            LHLD  STAFIL    ADDR FIRST BYTE
408 077F 223808    NXTCH   SHLD  PTRFIL    POINTS TO BYTE
409 0782 7E                MOV   A,M       BYTE IN ACCU
410 0783 0F        SHIFT   RRC
411 0784 0F                RRC
412 0785 0F                RRC
413 0786 0F                RRC
414 0787 E60F      MASK    ANI   :0F       MASK
415 0789 C630              ADI   48        MAKE ASCII 0-9
416 078B FE3A              CPI   58
417 078D DA9207            JC    STEP      IF NOT A NUMBER
418 0790 C607              ADI   7         MAKE ASCII A-F
419 0792 CD2608    STEP    CALL  VERZ      SEND CHAR
420 0795 1C                INR   E
421 0796 3E02              MVI   A,2       2ND CHAR OF BYTE
422 0798 BB                CMP   E         DONE?
423 0799 CAA007            JZ    NXTBYT    NEXT BYTE
424 079C 7E                MOV   A,M
```

```
425 079D C38707              JMP   MASK       GET 2ND CHAR OF BYTE
426 07A0 3E20       NXTBYT   MVI   A,32       SEND SPACE BETWEEN BYTES
427 07A2 CD2608              CALL  VERZ
428 07A5 1E00                MVI   E,0        CLEAR CHAR CNTR
429 07A7 14                  INR   D          INCR BYTE CNTR
430 07A8 3E10                MVI   A,16       LINE FULL?
431 07AA BA                  CMP   D          YES, THEN CAR-RET
432 07AB CC1E08              CZ    CARRET
433 07AE 23                  INX   H          INCR PNTR
434 07AF E5                  PUSH  H
435 07B0 D5                  PUSH  D
436 07B1 EB                  XCHG
437 07B2 2A3608              LHLD  ENDFIL     END BUFF ?
438 07B5 CD14DE              CALL  :DE14
439 07B8 D1                  POP   D
440 07B9 E1                  POP   H
441 07BA CA4207              JZ    STAY1      EXIT AFTER END FILE OR
442 07BD CDBED6              CALL  :D6BE      BREAK
443 07C0 DA4207              JC    STAY1
444 07C3 C37F07              JMP   NXTCH      ELSE NEXT BYTE
445 07C6 21DD07     HEXINP   LXI   H,INSBR    ADDR INP SUBR
446 07C9 22D200              SHLD  :D2
447 07CC 0E00                MVI   C,0        CLEAR CHAR CNTR
448 07CE CD2AC0              CALL  :C02A      HEX INPUT TO MACC
449 07D1 21960A              LXI   H,RESULT
450 07D4 E7                  RST   4
451 07D5 0F                  DATA  :0F
452 07D6 2A980A              LHLD  RESULT+2
453 07D9 7C                  MOV   A,H        SWAP
454 07DA 65                  MOV   H,L
455 07DB 6F                  MOV   L,A
456 07DC C9                  RET
457 07DD 79         INSBR    MOV   A,C        CHAR CNTR
458 07DE FE04                CPI   4          MAX 4 CHAR'S
459 07E0 C8                  RZ
460 07E1 CDBED6     GHEX     CALL  :D6BE      WAIT FOR INP
461 07E4 CAE107              JZ    GHEX
462 07E7 DAF507              JC    SFL5       IF BREAK TO MENU
463 07EA FE0D                CPI   13
464 07EC C2FD07              JNZ   ALFN       IS CHAR CAR-RET?
465 07EF 47                  MOV   B,A
466 07F0 AF                  ZAR
467 07F1 B9                  CMP   C          FIRST INPUT?
468 07F2 C2FB07              JNZ   EXIT
469 07F5 3E01       SFL5     MVI   A,1        YES,SET ABORTFL
470 07F7 323008              STA   FLAG5
471 07FA C9                  RET
472 07FB 78         EXIT     MOV   A,B
473 07FC C9                  RET
474 07FD CD09DE     ALFN     CALL  :DE09      TEST ALFANUM
475 0800 D2E107              JNC   GHEX
476 0803 3F                  CMC
477 0804 FE47                CPI   71         TEST A-F
```



```
478 0806 D2E107              JNC   GHEX
479 0809 3F                  CMC
480 080A CD95D6              CALL  :D695      DISPLAY ON SCRN
481 080D C9                  RET
482 080E CDB404     TRCDEL   CALL  TRANSC     TRANSMIT CHAR
483 0811 06FF                MVI   B,:FF      DELAY CONSTANT
484 0813 CD6B04     TEST     CALL  CHKSTS     LOOK FOR RECEIVED
485 0816 C27104              JNZ   INTR1      CHAR'S DURING DELAY
486 0819 05                  DCR   B
```

```
487 081A C21308            JNZ   TEST
488 081D C9                RET
489 081E 1600      CARRET  MVI   D,0        CLEAR LINE BYTE CNTR.
490 0820 3E0D              MVI   A,13       START NEW LINE
491 0822 CD2608            CALL  VERZ       TRANSMIT
492 0825 C9                RET
493 0826 D5        VERZ    PUSH  D
494 0827 CD0E08            CALL  TRCDEL     TO SEND ROUTINE
495 082A D1                POP   D
496 082B C9                RET
497 082C 00        FLAG1   DATA  0          INIT. FLAG
498 082D 00        FLAG2   DATA  0          ASCII/HEX FLAG
499 082E 00        FLAG3   DATA  0          HALF/FULL DUPLEX FLAG
500 082F 00        FLAG4   DATA  0          TALK FLAG
501 0830 00        FLAG5   DATA  0          ABORT FLAG
502 0831 00        CHKSUM  DATA  0
503 0832 00        STORE1  DATA  0
504 0833 00        STORE2  DATA  0
505 0834 0000      STAFIL  DBL   0
506 0836 0000      ENDFIL  DBL   0
507 0838 0000      PTRFIL  DBL   0
508 083A 0000      BYTE    DBL   0
509 083C 0C0D      MSG1    DATA  12,13
510 083E 202020            ASC   '      TALK !!  (cursor-up for MENU)'
511 0860 0D0D00            DATA  13,13,0
512 0863 0C0D      MSG2    DATA  12,13
513 0865 202045            ASC   '  ENTER EDIT MODE      <E>'
514 087E 0D0D              DATA  13,13
515 0880 202048            ASC   '  HALF DUPLEX (DCE)    <H>'
516 0899 0D                DATA  13
517 089A 202028            ASC   '  (echo"s to DTE)'
518 08AB 0D                DATA  13
519 08AC 202046            ASC   '  FULL DUPLEX (DTE)    <F>'
520 08C5 0D0D              DATA  13,13
521 08C7 202054            ASC   '  TALK                 <T>'
522 08E0 0D0D              DATA  13,13
523 08E2 202053            ASC   '  SEND DNA SOURCE      <S>'
524 08FB 0D                DATA  13
525 08FC 202028            ASC   '  (+warm start init.)'
526 0911 0D                DATA  13
527 0912 202053            ASC   '  SEND ASCII FILE      <A>'
528 092B 0D                DATA  13
529 092C 202053            ASC   '  SEND HEX  FILE       <#>'
530 0945 0D0D              DATA  13,13
```

PAGE 11

```
531 0947 202052            ASC   '  RECEIVE ASCII FILE  <R>'
532 0960 0D                DATA  13
533 0961 202052            ASC   '  RECEIVE HEX  FILE   <X>'
534 097A 0D0D              DATA  13,13
535 097C 202042            ASC   '  BACK TO UTILITY     <U>'
536 0995 0D0D              DATA  13,13
537 0997 202042            ASC   '  BACK TO BASIC       <B>'
538 09B0 0D0D0D            DATA  13,13,13
539 09B3 202054            ASC   '  TYPE INSTRUCTION !   '
540 09CA 00                DATA  0
541 09CB 0C0D      MSG3    DATA  12,13
542 09CD 202020            ASC   '                    SEND FILE'
543 09E8 0D0D00            DATA  13,13,0
544 09EB 0C0D      MSG4    DATA  12,13
545 09ED 202020            ASC   '                    RECEIVE FILE'
546 0A0B 0D0D00            DATA  13,13,0
547 0A0E 0C0D0D    MSG5    DATA  12,13,13
548 0A11 494E50            ASC   'INPUT START ADDRESS OF FILE  #'
```

```
549 0A2F 00                    DATA  0
550 0A30 0D          MSG6      DATA  13
551 0A31 494E50                ASC   'INPUT LAST ADDRESS  OF FILE  #'
552 0A4F 00                    DATA  0
553 0A50 0C0D0D      MSG7      DATA  12,13,13
554 0A53 484558                ASC   'HEX FILE LOCATED FROM  #'
555 0A6B 00                    DATA  0
556 0A6C 202054      MSG8      ASC   '  TO  #'
557 0A73 00                    DATA  0
558 0A74 0D0D        MSG9      DATA  13,13
559 0A76 52454C                ASC   'RELOCATE TO (GIVE STARTADDR.) #'
560 0A95 00                    DATA  0
561 0A96             RESULT    RES   4,0
562 0A9A             ENDPRG    END

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

ADJPTR 072E   ALFN   07FD   BACKSP 0477   BDRATE 0084
BYTE   083A   CARRET 081E   CHKFL1 05D2   CHKSTS 046B
CHKSUM 0831   CHOISE 04FE   CLCHKS 05AA   CLEBUF 045F
CONVRT 0713   CVTBUF 068A   DISPL  0631   EBPTR  00A4
ECHO   048F   EDIT   05AF   EDIT2  0751   ENDEB  00A6
ENDFIL 0836   ENDPRG 0A9A   ENDWST 0454   EXIT   07FB
FLAG1  082C   FLAG2  082D   FLAG3  082E   FLAG4  082F
FLAG5  0830   FULLD  0554   GETC   0734   GHEX   07E1
HALFD  054C   HDUPLX 04E2   HEXINP 07C6   HEXPAS 06F0
INCHAR FFF0   INCREB 06B3   INITEB 0721   INSBR  07DD
INTR1  0471   LETTER 071E   LKETXT 0724   LOOK   0614
LOOK2  0617   LOOK3  064E   MAINLP 0406   MASK   0787
MSG1   083C   MSG2   0863   MSG3   09CB   MSG4   09EB
MSG5   0A0E   MSG6   0A30   MSG7   0A50   MSG8   0A6C
MSG9   0A74   NBRK   06E5   NEXTCH 05BA   NXTBYT 07A0
NXTCH  077F   OUT    04CD   PASS2  0708   PRADDR 06B7
```



```
PTRFIL 0838   READ   0601   RECASC 056D   RECHEX 0577
RESULT 0A96   RMSG   0598   SASCI  0563   SCREEN 0484
SDNAS  05B7   SFL5   07F5   SHEX   055B   SHIFT  0783
SMSG   0582   SPECL  04E6   SSOURC 05EB   SSTEB  05E0
STAEB  00A2   STAFIL 0834   START  0423   STAY1  0742
STEP   0792   STOPTR 067C   STORE1 0832   STORE2 0833
STSREG FFF3   SUMCHA 04D9   TALK   053B   TEST   0813
TEST1  065D   TOBAS  0544   TOUT   0418   TRANS  04AD
TRANSC 04B4   TRCDEL 080E   VERZ   0826   WAIT   04A3
WAIT2  049C
```

* neu * nieuw * new * nouveau *

* Schreibschrift *
* writing-characters *

# DAI # Itho 8510 #

******************************************

Das neue 4k Eprom 'SSV1' ermöglicht die Darstellung von zwei verschiedenen Zeichensätzen im mixed Mode. Das Eprom enthält einen deutschen Schreibschrift- und einen deutschen Standartzeichensatz. Setzt man ein Eprom vom Typ SSV1-DAI in den DAI-Personal Computer und ein Eprom vom Typ SSV1-Itho in den Drucker Itho 8510, so lassen sich beide Zeichensätze sowohl mit dem DAI als auch mit dem Drucker über Steuerzeichen im mixed Mode darstellen.

The new 4k eprom 'SSV1' permits the display of two different character sets in mixed mode. The eprom contains a german writing and a german standart set. If you put for example one eprom type SSV1-DAI into the DAI-Personal Computer and one eprom type SSV1-Itho into the Itho 8510 printer, thus both character sets can be displayed in mixed mode on the DAI-Personal Computer as well as on the Itho printer by useing control signs.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| #A0 | #A1 ! | #A2 " | #A3 # | #A4 $ | #A5 % | #A6 & | #A7 ' | #A8 ( |
| #A9 ) | #AA * | #AB + | #AC , | #AD - | #AE . | #AF / | #B0 0 | #B1 1 |
| #B2 2 | #B3 3 | #B4 4 | #B5 5 | #B6 6 | #B7 7 | #B8 8 | #B9 9 | #BA : |
| #BB ; | #BC < | #BD = | #BE > | #BF ? | #C0 § | #C1 A | #C2 B | #C3 C |
| #C4 D | #C5 E | #C6 F | #C7 G | #C8 H | #C9 I | #CA J | #CB K | #CC L |
| #CD M | #CE N | #CF O | #D0 P | #D1 Q | #D2 R | #D3 S | #D4 T | #D5 U |
| #D6 V | #D7 W | #D8 X | #D9 Y | #DA Z | #DB Ä | #DC Ö | #DD Ü | #DE ^ |
| #DF _ | #E0 ` | #E1 a | #E2 b | #E3 c | #E4 d | #E5 e | #E6 f | #E7 g |
| #E8 h | #E9 i | #EA j | #EB k | #EC l | #ED m | #EE n | #EF o | #F0 p |
| #F1 q | #F2 r | #F3 s | #F4 t | #F5 u | #F6 v | #F7 w | #F8 x | #F9 y |
| #FA z | #FB ä | #FC ö | #FD ü | #FE ß | | | | |

1 Eprom 49,- DM
+ 7,50 DM post & packing.

address:

Frank Caßebaum
Grenzstraße 64
D-2800 Bremen 01
Tel.: 0421/3 96 23 53

eprom type:

SSV1-DAI for DAI-PC
SSV1-Itho for Itho printer 8510

UMLA-Epson for Epson printer LX80 with standart character set including german 'Umlaute' and Graftrax (Plotter)

your order by check or on Bancaccount nr: 550005301309
Verbraucherbank Bremen
Bankleitzahl: 29020300
West - Germany

# * neu * nieuw * new * nouveau *

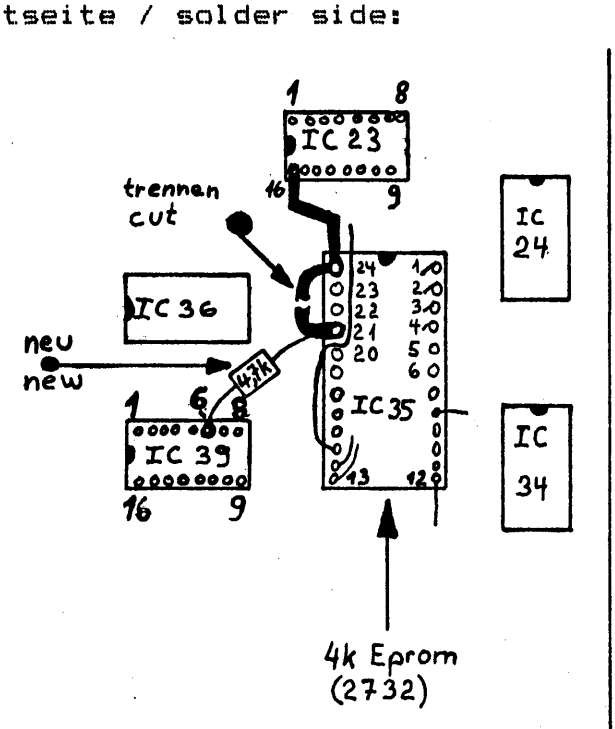
## * 256 *
## * Zeichen *
## * characters *

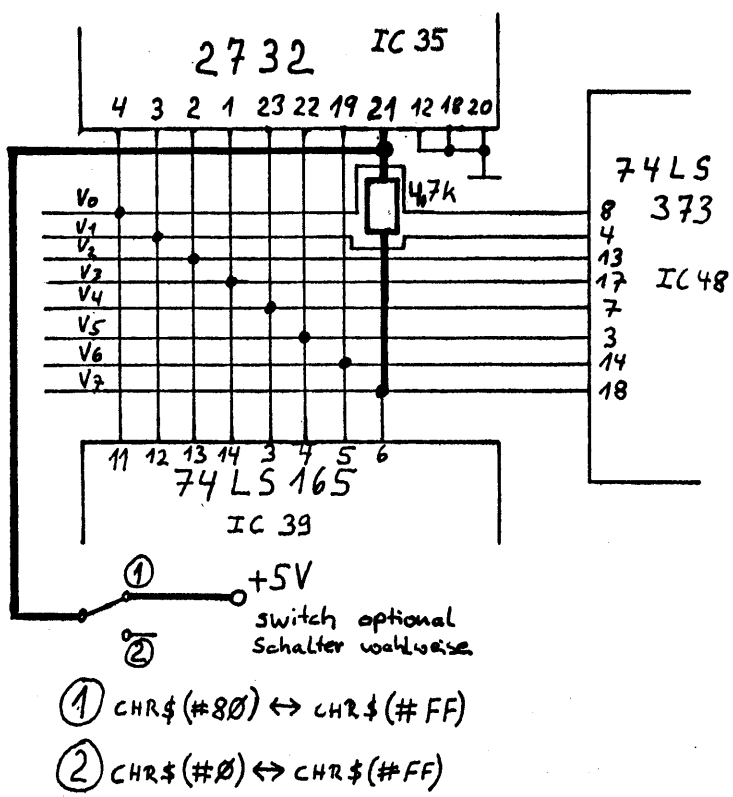
**********

Eine kleine Änderung Ihrem DAI-PC ermöglicht die Darstellung von zusätzlichen 125 Zeichen. Es sind lediglich eine Leiterbahn zu unterbrechen, ein 4,7 kOhm Widerstand und ein 4k Eprom einzubauen. Nach diesem Umbau können Zeichen von #0 bis #FF mit dem DAI dargestellt werden.

Just a little change in your DAI-PC permits the display of additional 125 characters. You have to cut a track and to insert a resistor and a 4k eprom. After doing this, characters from #0 to #FF can be displayed with your DAI-PC.

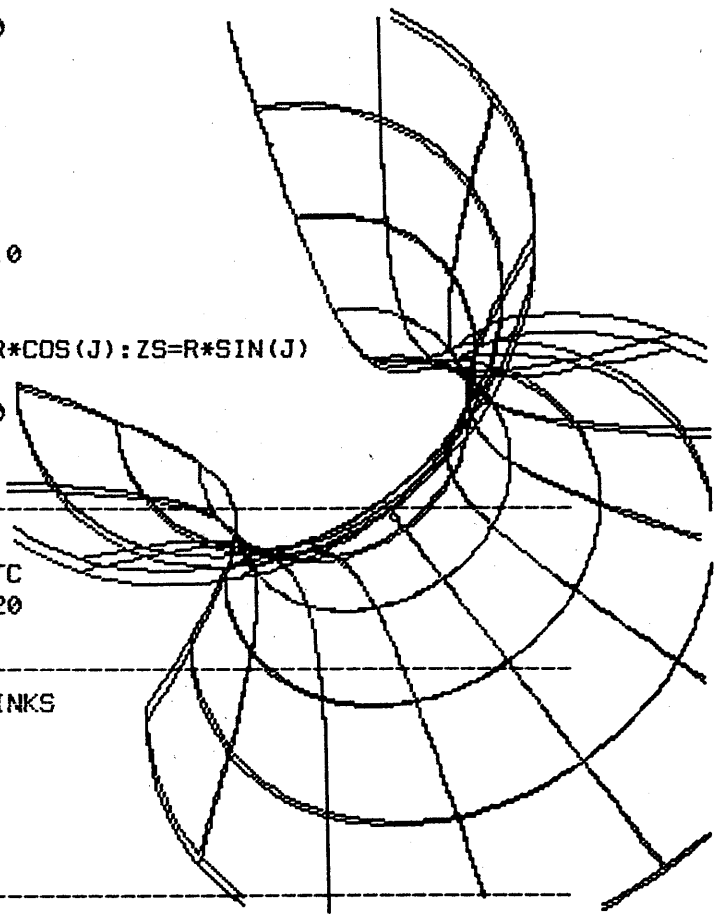
Lötseite / solder side:



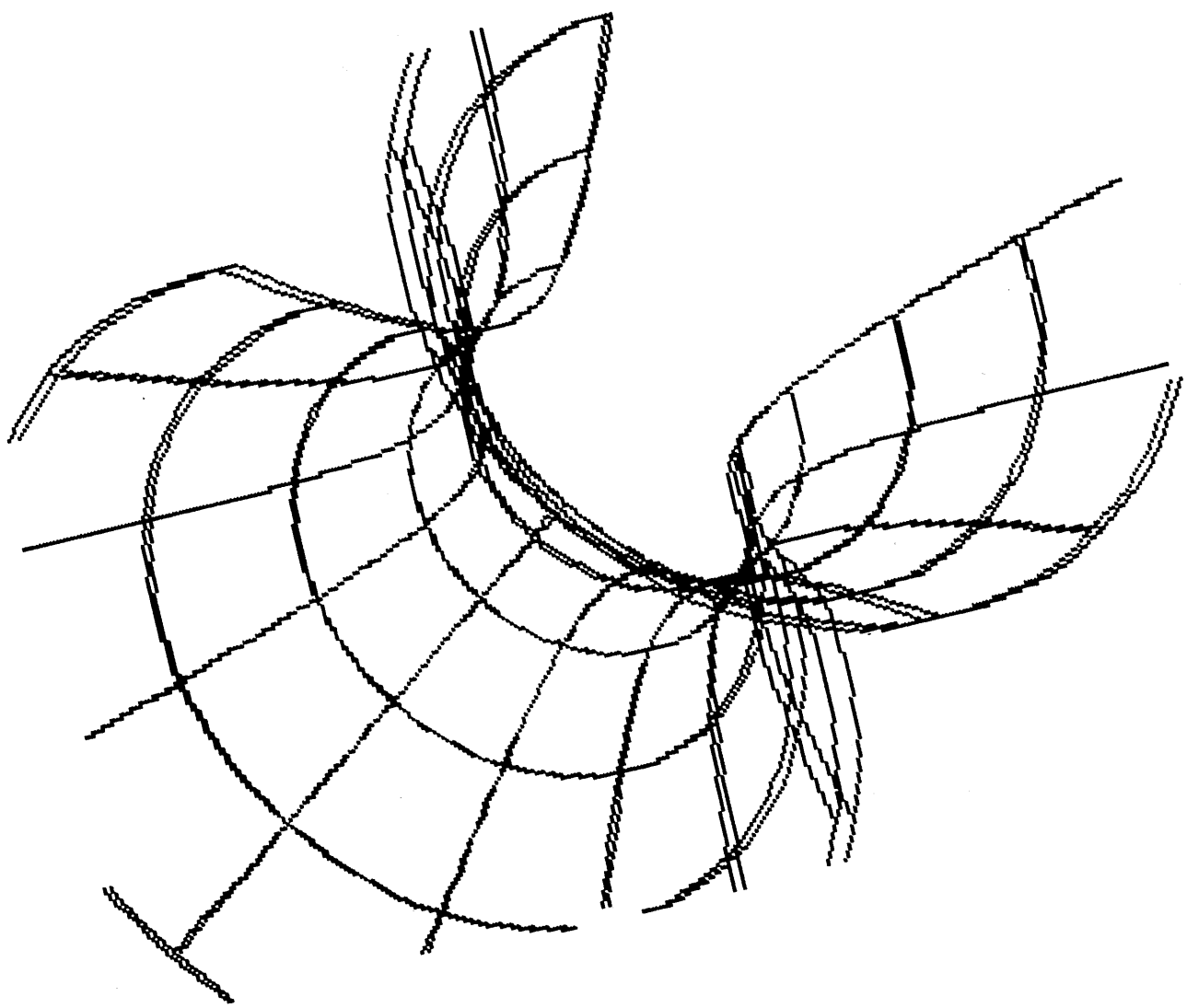




Frank Caßebaum
Grenzstraße 64
D-2800 Bremen 1
Tel.: 0421/3 96 23 53

```
50    MODE 0:PRINT CHR$(12):REM TEKST MODE - VAAG SCHERM
65    CURSOR 0,18
66    PRINT "  ************************************************************"
67    PRINT "  *                                                          *"
70    PRINT "  *  D R I E   D I M E N S I O N E E L   T E K E N E N   *"
72    PRINT "  *                                                          *"
73    PRINT "  *                                                          *"
74    PRINT "  *                    EENBLADIGE                            *"
75    PRINT "  *                                                          *"
76    PRINT "  *             OMWEMTELINGSHYPERBOLOIDE                     *"
77    PRINT "  *                                                          *"
80    PRINT "  *                                              J.ROELANTS  *"
82    PRINT "  ************************************************************"
84    PRINT :PRINT :PRINT
90    REM ------------------------------------------------------------------
100   V=2000.0:L=35.0:REM WAARNEMINGSAFSTAND - AFSTAND TUSSEN W.PUNTEN
600   PRINT " Geef de verdraaiing rond X,Y,Z AS in graden (v.b.70,30,0)"
605   INPUT "----> ";A,B,C:PRINT
610   REM OMZETTING NAAR RADIALEN
620   A=A/57.296:B=B/57.296:C=C/57.296
640   GOSUB 4000
1000  MODE 6:MODE 6:REM GRA FISCHE MODE 336 x 256
1010  REM KLEURKEUZE ZWART ROOD GROEN GRIJS
1020  COLORG 0 3 5 8
1025  REM ------------------------------------------------------------
1030  REM FIGUUR OWEMTELINGSHYPERBOLOIDE
1035  REM ---> WIJZIG DE KEELDOORSNEDE  V.B.  C=2000
1040  A=50.0:C=60.0:LY=140.0:REM RARAMETERS
1060  FOR YS=-LY TO LY STEP 40.0
1070  Q=YS/C:R=A*SQR(1.0+Q*Q)
1080  FOR JJ=0.0 TO 270.0 STEP 3.0
1090  J=JJ*PI/180.0
1100  XS=R*COS(J):ZS=R*SIN(J)
1140  GOSUB 3000:GOSUB 2000
1150  IF JJ<>0.0 THEN GOSUB 2100
1155  XERM=XER:XELM=XEL:YEM=YE
1180  NEXT JJ:NEXT YS
1200  FOR JJ=0.0 TO 270.0 STEP 30.0
1210  J=JJ*PI/180.0
1220  FOR YS=-LY TO LY STEP 5.0
1230  Q=YS/C:R=A*SQR(1.0+Q*Q):XS=R*COS(J):ZS=R*SIN(J)
1240  GOSUB 3000:GOSUB 2000
1245  IF YS<>(-LY) THEN GOSUB 2100
1246  XERM=XER:XELM=XEL:YEM=YE
1250  NEXT YS:NEXT JJ
1260  REM ------------------------------------------------------------
1390  REM WACHTLUS
1400  GET%=GETC:GET%=GETC:GET%=GETC
1420  GET%=GETC:IF GET%=0 THEN 1420
1500  GOTO 50
1990  REM ------------------------------------------------------------
2000  REM BEELDPUNTEN RECHTS EN LINKS
2020  K=V/(ZP+V)
2040  D=(1.0-K)*L:KX=K*XP+168.5
2060  XER=KX+D:YE=K*YP+128.5
2080  XEL=KX-D
2090  RETURN
2095  REM ------------------------------------------------------------
2100  REM TEST OF LIJN NIET BUITEN VALT
2110  IF YE<0.0 OR YE>YMAX OR YEM<0.0 OR YEM>YMAX THEN 2220
2120  IF XER<0.0 OR XER>XMAX OR XERM<0.0 OR XERM>XMAX THEN 2220
2130  IF XEL<0.0 OR XEL>XMAX OR XELM<0.0 OR XELM>XMAX THEN 2220
2140  REM TEKEN RODE EN GROENE LIJN
```



```
2150  DRAW XER,YE XERM,YEM 17:DRAW XEL,YE XELM,YEM 19
2220  RETURN
2230  REM ------------------------------------------------------------
3000  REM ROTATIE
3010  XP=AX*XS+BX*YS+CX*ZS
3020  YP=AY*XS+BY*YS+CY*ZS
3040  ZP=AZ*XS+BZ*YS+CZ*ZS
3200  RETURN
3990  REM ------------------------------------------------------------
4000  AX=COS(B)*COS(C)
4020  BX=SIN(A)*SIN(B)*COS(C)-COS(A)*SIN(C)
4040  CX=COS(A)*SIN(B)*COS(C)+SIN(A)*SIN(C)
4060  AY=COS(B)*SIN(C)
4070  BY=SIN(A)*SIN(B)*SIN(C)+COS(A)*COS(C)
4080  CY=SIN(C)*SIN(B)*COS(A)-COS(C)*SIN(A)
4090  AZ=(-1.0)*SIN(B)
4100  BZ=SIN(A)*COS(B)
4120  CZ=COS(A)*COS(B)
4200  RETURN
4220  REM ************************************************************
```



# AUDIO-CASSETTE INTERFACE

## PART 2: READING FROM TAPE

### 1. INTRODUCTION:

#### 1.1. WRITING TO TAPE:

See a previous Newsletter for part 1 of this article about writing to tape.
Details on the software routines can be found in the DAI pc Firmware Manual.

#### 1.2. SIGNAL FORMATS:

In part 1, the format of the signals on tape, the routines used for writing and the set-up of a tape file are explained.
Part 2 describes the way signals are retrieved from tape by the DAI resident software.

The principle of reading tape files is identical to the way they are written, but of course just the other way around.
The reading routines will be described in the same sequence as done with the writing routines.

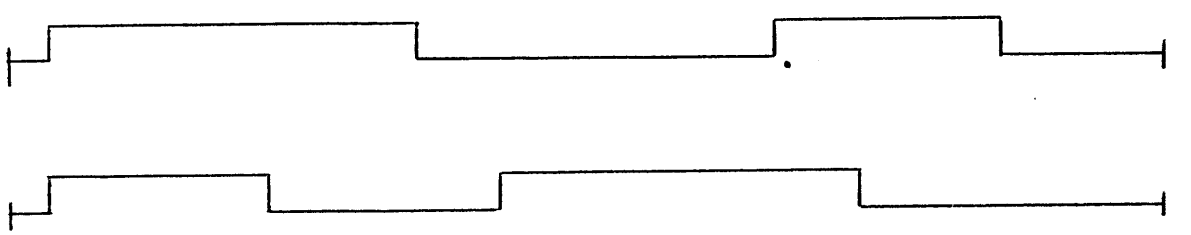
### 2. HARDWARE CONNECTIONS:

The connections between the DAI and the audio cassette recorder are completely described in part 1.

### 3. SUBROUTINES FOR READING FROM TAPE:

#### 3.1. READ A BIT - RBIT/D453:

On entry, the register pair HL contain the address of the recorder input port FD00. The most-significant bit (msb - bit 7) of the data read from this port is the received data. As long as it is 'low', D45B (ORA M) results in '0', causing a loop via JP :D457. If this loop is done too often (checked by a count in register B), an error (carry-flag = 1) is the result via D47E. When a 'high' is received, D45B (ORA M) gives #80 and the routine continues.



data '1' and '0' bits

The loop D45F-D465 waits for the received data to go low again. Here, register C checks if the signal remains too long high. During this loop, a counter (register D) is decremented. When a 'low' is received, counter D stops, and the loop D46B-D470 is entered, waiting for the received signal to go 'high' again (register B checks for too long low).
If this happens, loop D473-D479 waits for a return to 'low' (register C checks too long high). During this loop, counter D is now incremented until the signal goes low.

Depending on which impulse is the longest, the value of counter D at the end will be positive or negative:

| | | |
|---|---|---|
| 1st impulse longer: | D<0 | data = '1' |
| 2nd impulse longer: | D>0 | data = '0' |

The contents of counter D is moved into the accumulator (D47C). If D<0, the msb of A is '1'. If D>0, the msb is '0'. So the received bit is now bit 7 of the accumulator.

If a reading error occurs (the signal is too long high or low), the routine will exit with the carry-flag set (CY=1) via D47E.

Note that reading a bit is time-independent. No relation with a fixed impulse length is used, only a compare between both impulses is made.

### 3.2. READ A BYTE - RBYTE/D4D4:

Because a byte consists of 8 bits, the routine RBIT has to be performed 8 times.
The register E is loaded with #FE (1111 1110). A bit is read via RBIT into the msb of the accumulator. Via the carry-flag, this bit is shifted into the register E (D4E2-D4E5). As long as the bit which is shifted out of E is a '1', the routine is repeated (totally 8 times) via JC :D4DC.
Each time a new bit is read, it is shifted behind the previous one into register E via D4E2-D4E5.
After 8 times, the CY-flag is '0'. Then the received byte is available in the accumulator.

If during RBIT a reading error occurs, RBYTE is aborted via D4DF (JC :D4E9). Then on exit of RBYTE, the carry-flag will be set, indicating a loading error.

### 3.3. READ A LEADER - RLEAD/D480:

Is the absolute timing not important when reading data bits, for the recognition of the leader tone on tape it is very important.
Reading a leader tone consists of reading a continuous tone from tape and checking if its frequency is within a certain margin. If this is true long enough, the tone is recognised as a leader.

- D48A-D48E: As long as the received signal (via port FD00) is high, the loop waits for the signal to go low. Via EI/DI, the cursor remains flashing.
- D491-D49C: Now is waited for the signal to go high. If it lasts too long (register E counts), the leader reading routine is restarted (JZ :D488).
- D49F-D4A6: As soon as the signal goes high, register B is used as a counter. It is incremented as long as the signal remains high. A too long high level restarts the leader check routine.
- D4A9-D4AB: As soon as the signal goes low again, the duration count in register B is compared with an estimate (in register C: #28). The difference is stored in register E (D4B0).
- D4B0-D4B8: The tolerated margin is calculated (04) and compared with the difference in register E. If the difference is too large (that means: no leader impulse), JC :D4C3 occurs.
- D4BB-D4C0: If the received impulse has a length which is within the margin, the next impulse is checked via JNZ :D494. This is repeated 20 times (register D counts) to be sure the received signal is a real leader tone.

If 20 times (#14) a correct impulse is received (that means: leader tone found), the routine waits in a loop for other impulses than those of the leader. Because the leader consists of 2024 impulse pairs of the duration '24', followed by a data '1' bit (duration '3C/24'), the '3C' impulse of the latter will cause the routine to be aborted because it differs too much from the margin. Then D4B8 causes JC :D4C3.

- D4C3-D4C4: If 'out of margin' occurs during the first 20 runs of the routine D494-D4BC, the register D is not yet 0. That means synchronisation is not yet found, and the routine starts anew.
More than 20 runs means synchronisation is detected, and the program continues.
- D4C7-D4CD: After the first impulse of the data '1' bit after the leader tone (duration '3C': out of margin), here the second impulse of this bit is handled.

## 4. FILE HANDLING:

Read again carefully what is written in part 1 about the set-up of a file.
For audio-cassette recorder operation, the resident software has routines to open a file from tape (CROPEN), read data blocks from tape (CRBLK) and to close a file read from tape (CRCLOSE).
The startaddresses of these file handling routines are moved from ROM into RAM during reset. The default addresses for audio-cassette recorder operation are:

| | | |
|---|---|---|
| ROPEN | #02CE | JMP #D325 |
| RBLK | #02D1 | JMP #D340 |
| RCLOSE | #02D4 | JMP #D445 |

### 4.1. OPEN A TAPE FILE - CROPEN/D325:

- Get length of name expected (if given) in register pair DE; HL points to the name (MPT23/D7FF). Switch on cassette motors (CASST/D42E).
- Disable sound interrupts (SNDDI/D98F).
- Read the header (RHDR/D3F4):
  - Read leader, find synchronisation (RLEAD/D480).
  - Read flag byte #55 (RBYTE/D4D4).
  - Read file type byte (RBYTE/D4D4):
    - #30: A Basic program.
    - #31: A Hex file.
    - #32: An array.
- If not load during program run: Display file type byte on the screen (MPT24/D78A); This byte is directly POKE'd into the screen memory.
- Read the name of the program (CMBLK/D337).
  The length of the name (and its checksum) is read, then the name is read and POKE'd into the screen memory, and the checksum on the name is read.
  If a loading error occurs on one of these read routines, a loading error report is prepared via D3ED. If all checks are O.K., the CROPEN routine is ready now.

### 4.2. READ A BLOCK FROM TAPE - CRBLK/D340:

- Calculate the available memory space (MPT25/D790).
- Read the length of the block and the checksum on the length (INLNG/D38D).
  An error is reported if a reading error (via D37E) or checksum error occurs or if the available memory space is too small (via D380).
- Read the contents of the block (D364-D36C). The error exit via D37E is taken if a reading error occurs.
- Read the checksum on the block contents (RBYTE/D4D4). Here again the error exit via D380 if a checksum error has been found.

### 4.3. CLOSE A TAPE FILE - CRCLOSE/D445:

- Switch off the cassette motors (CASSP/D445).

Note: The trailer tone on the tape is not written !

## 5. RESIDENT FILE READING ROUTINES:

Three resident file reading routines are available:

| | | |
|---|---|---|
| 'LOAD' | file type 0 | Read Basic programs. |
| 'UT/Read' | file type 1 | Read Hex files. |
| 'LOADA' | file type 2 | Read Arrays. |

The use of the routines ROPEN, RBLK and RCLOSE is not 100% identical for each of these methods.

5.1. BASIC COMMAND 'LOAD' - RLOAD/D270:

- D270: Clear all variables in the heap and in the symbol table. Evaluate the (eventually) expected file name. Check if load during program run.
- D28A: ROPEN (see 4.1).
- D28D: Get pointers to begin of textbuffer (TXTBGN) and bottom of screen memory (SCRNBOT) for calculation of available memory space.
- D294: RBLK: Read length of textbuffer + checksum.
  Idem for textbuffer contents.
- D297: Get pointer to start of symboltable for calculation of available memory space.
- D29A: RBLK: Read length of symboltable + checksum.
  Idem for symboltable contents.
- D29D: Store endaddress of symboltable.
  RCLOSE (see 4.3).
- D2A2: Eventually, run 'LOADING ERROR 0/1/2/3'.

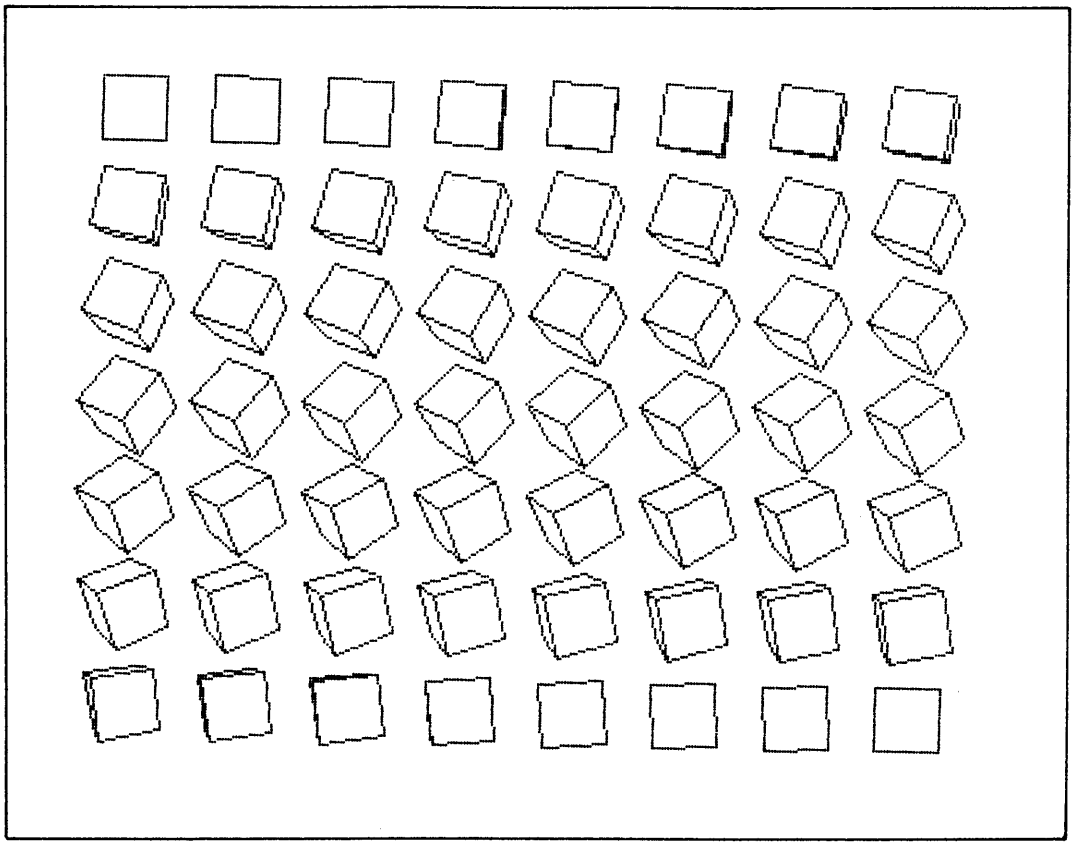
5.2. UTILITY COMMAND 'R(ead)' - RHEXK/3EF0F:

- EF0F: Get an evt. offset for startaddress on stack.
- EF14: Get an evt. name in the encoded input buffer.
- EF1E: ROPEN.
- EF21: Sets the maximum available RAM area to F900 (end of stack RAM). This is incorrect, because C000-EFFF is ROM and F000-F7FF is also not available.
- EF25: RBLK: Read startaddress from tape.
- EF28: Add an evt. offset given.
- EF29: RBLK: read the data block from tape.
  RCLOSE: Stop reading.
- EF2A: Run an evt. loading error (print '?').

5.3. BASIC COMMAND 'LOADA' - RLODA/D85E:

- D85E: Evaluate type of array to be loaded.
- D863: Evaluate an evt. program name; switch to ROM bank 1.
- 1EE0F: ROPEN; RBLK: read array type.
  If INT/FPT arrays: RBLK: read array contents into the array in the heap.
  If string arrays: RBLK: read the array contents into the free RAM space (see Newsletter 1981/p.10). Then move the data into the heap.
  Switch to ROM bank 0.
  RCLOSE.

```
10    CLEAR 256:MODE 0:CLEAR 1500
11    REM ** Cube rotating and translating in space **
20    SV1!=0.0:SV2!=0.0:SV3!=0.0:SQ1!=0.0:SQ2!=0.0:SQ3!=0.0
21    IPTS=0:IPTF=0:K1=0:K2=0:F!=5.9
22    X=-21:XT=42:XS=21:Y=240:YT=37:DF=23
30    NV=8:NE=12:NS=6:CO=3:AN=4
31    DIM M!(CO,CO),NORM!(CO,NS),VERT!(CO,NV)
32    DIM EDGV1(NE),EDGV2(NE),PEDG(AN*NS)
33    DIM EDG(NE),XP!(NE),YP!(NE)
50    GOSUB 11000:MODE 6:COLORG 8 0 10 15
3000  FOR L=1 TO 56
3002  FOR U=1 TO NE:EDG(U)=0:NEXT
3004  IF L=1 OR L=56 THEN RESTORE:GOSUB 10000:GOTO 3020
3010  FOR U=1 TO NV
3011  SV1!=VERT!(1.0,U):SV2!=VERT!(2.0,U):SV3!=VERT!(3.0,U)
3012  VERT!(1.0,U)=SV1!*M!(1.0,1.0)+SV2!*M!(1.0,2.0)+SV3!*M!(1.0,3.0)
3013  VERT!(2.0,U)=SV1!*M!(2.0,1.0)+SV2!*M!(2.0,2.0)+SV3!*M!(2.0,3.0)
3014  VERT!(3.0,U)=SV1!*M!(3.0,1.0)+SV2!*M!(3.0,2.0)+SV3!*M!(3.0,3.0)
3019  NEXT
3020  IPTS=1:FOR U=1 TO NS
3022  SQ1!=NORM!(1.0,U):SQ2!=NORM!(2.0,U):SQ3!=NORM!(3.0,U)
3023  NORM!(1.0,U)=SQ1!*M!(1.0,1.0)+SQ2!*M!(1.0,2.0)+SQ3!*M!(1.0,3.0)
3024  NORM!(2.0,U)=SQ1!*M!(2.0,1.0)+SQ2!*M!(2.0,2.0)+SQ3!*M!(2.0,3.0)
3025  NORM!(3.0,U)=SQ1!*M!(3.0,1.0)+SQ2!*M!(3.0,2.0)+SQ3!*M!(3.0,3.0)
3030  IPTF=IPTS+3:K1=PEDG(IPTS):K2=EDGV1(K1)
3031  XN2!=-VERT!(1.0,K2):YN2!=-VERT!(2.0,K2):ZN2!=-VERT!(3.0,K2)-F!
3032  DT!=NORM!(1.0,U)*XN2!+NORM!(2.0,U)*YN2!+NORM!(3.0,U)*ZN2!:IF DT!<0.0
GOTO 3041
3040  FOR V=IPTS TO IPTF:VV=PEDG(V):EDG(VV)=EDG(VV)+1.0:NEXT
3041  IPTS=IPTS+4
3049  NEXT
3050  FOR U=1 TO NV:ZPF!=VERT!(3.0,U)+F!
3051  XP!(U)=F!*VERT!(1.0,U)/ZPF!:YP!(U)=F!*VERT!(2.0,U)/ZPF!:NEXT
3055  X=X+XT:IF X>XMAX THEN X=XS:Y=Y-YT
3060  FOR U=1 TO NE:IF EDG(U)=0.0 GOTO 3069
3062  X1=XP!(EDGV1(U))*DF:Y1=YP!(EDGV1(U))*DF
3063  X2=XP!(EDGV2(U))*DF:Y2=YP!(EDGV2(U))*DF
3065  DRAW X+X1,Y+Y1 X+X2,Y+Y2 0
3069  NEXT
3099  NEXT
9999  END
10000 FOR I=1 TO CO:FOR J=1 TO NS:READ K:NORM!(I,J)=K:NEXT:NEXT
10002 DATA 0,-1,0,1,0,0,0,0,0,0,-1,1,-1,0,1,0,0,0
10004 FOR I=1 TO CO:FOR J=1 TO NV:READ K:VERT!(I,J)=K-0.5:NEXT:NEXT
10006 DATA 0,0,1,1,0,0,1,1,0,1,1,0,0,1,1,0,0,0,0,0,1,1,1,1
10008 FOR I=1 TO NE:READ EDGV1(I):NEXT
10010 DATA 1,2,3,4,5,6,7,8,1,2,3,4
10012 FOR I=1 TO NE:READ EDGV2(I):NEXT
10014 DATA 2,3,4,1,6,7,8,5,5,6,7,8
10016 FOR I=1 TO NS*AN:READ PEDG(I):NEXT
10018 DATA 1,2,3,4,1,9,5,10,8,7,6,5,3,11,7,12,4,12,8,9,2,10,6,11
10099 RETURN
11000 RAD!=PI/(180.0/1.25):SN!=SIN(RAD!):CS!=COS(RAD!)
11001 M!(1.0,1.0)=SN!*SN!*SN!+CS!*CS!:M!(1.0,2.0)=SN!*CS!:M!(1.0,3.0)=SN!*C
S!-SN!*SN!*CS!
11002 M!(2.0,1.0)=-SN!*CS!+SN!*SN!*CS!:M!(2.0,2.0)=CS!*CS!:M!(2.0,3.0)=-SN!
*SN!-SN!*CS!*CS!
11003 M!(3.0,1.0)=-SN!*CS!:M!(3.0,2.0)=SN!:M!(3.0,3.0)=CS!*CS!
11099 RETURN
```



## LES MESSAGES D'ERREUR

Les 26 messages d'erreur du BASIC sont affiches grace a un programme qui se trouve a l'adresse

# D9F5

L'acces a ce programme se fait de la maniere suivante:

```
3E nn     -   MVI  A,VAL
C3 F5 D9  -   JUMP :D9F5
```

Avec nn = VAL = numero du message. Ci dessous le tableau qui donne la valeur nn (VAL) pour chaque message.

| | |
|---|---|
| 00 : NEXT WITHOUT FOR | 0E : LOADING ERROR 3 |
| 01 : RETURN WITHOUT GOSUB | 0F : UNDEFINED ARRAY |
| 02 : OUT OF DATA | 10 : COLOR NOT AVAILABLE |
| 03 : OVERFLOW | 11 : OFF SCREEN |
| 04 : UNDEFINED LINE NUMBER | 12 : ERROR LINE RUN |
| 05 : SUBSCRIPT ERROR | 13 : OUT OF MEMORY |
| 06 : DIVISION BY ZERO | 14 : TYPE MISMATCH |
| 07 : OUT OF STRING SPACE | 15 : LINE NUMBER OUT OF RANGE |
| 08 : STRING TOO LONG | 16 : STACK OVERFLOW |
| 09 : NUMBER OUT OF RANGE | 17 : SYNTAX ERROR |
| 0A : INVALID NUMBER | 18 : COMMAND INVALID |
| 0B : LOADING ERROR 0 | 19 : CAN'T CONT |
| 0C : LOADING ERROR 1 | 1A : LINE TOO COMPLEX |
| 0D : LOADING ERROR 2 | |

## SENDING-IN PROGRAMS

(from DAInamic 10, page 74)

Many members of DAInamic appear to be unaware that they can submit programs. Programs received here are judged and the sender receives our commentary on it. If the program justifies an exchange and is suitable to be added to DAInamics library the sender will also receive a number of the programs already in our library. The conditions required for a program to be eligible for an exchange are not stringent because even a beginner must have a chance to send in his work. The sender may state a preference for certain collections and we will oblige if possible. People who regularly send in will already know that sometimes they receive nothing but at other times get much more than the value of their current contribution. If your program is subsequently included in one of our "collections" you will receive a bonus, the whole collection, free. (Complain if we should forget).

Now let us talk about submitting programs. We prefer to receive them on DCR cassette so that there will be no loading problems. Failing that, but still appreciated, would be submission on disc or on a normal cassette. The higher cost of minicassette or diskette need not be too burdensome as you get back what you send in. We regularly have loading problems with ordinary cassettes and this in turn delays our reply to you. You can minimise such problems by starting off your tape with a series of 20 tests which will enable us to make our adjustments. Do this in the following way:-

```
NEW return
DIM A(0) return
FOR I=1 TO 20 : SAVEA A "TEST" : NEXT return
and of course do have the recorder on Record !
```

Furthermore you can assist with an accompanying letter, telling us what to expect on the tape. That saves us running the tape right through to its end when there is nothing more on it. An actual written letter is needed because the same information on a tape would be useless if, because of loading problems, it could not be read. Now for the most important part, the program itself.

May I give a spot of advice to all who are about to send in programs, do let someone with no knowledge of computers try out your program first. I have had scores of programs which after a RUN left me waiting only to find that the control was in a closed loop, or sometimes expecting me to give a particular response, although at that time I would have had no knowledge of what was required. At other times, an apparently correct program takes a long time reading in data, and this can be annoying if there has been no prior indication, preferably by an announcement and a running counter which indicates progress. I have also had many programs which after a RUN report OUT OF MEMORY, COLOUR NOT AVAILABLE, NUMBER OUT OF RANGE, OUT OF STRING SPACE, UNDEFINED ARRAY, OUT OF DATA and almost every other message, while further on in the program OFF SCREEN is the winner. Check your programs thoroughly - often another can do it better. A program with a confirmed fault will not be included in a collection. You can avoid many faults by improving the structure of the program. Remember that everybody does not see the same picture equally well, so your chosen colour combination could be difficult for another to read. I find 4/5, 4/8, 4/12 unuseable, but 6/7 and 7/8 are nice and clear. There are obviously still many aspects remaining to be discussed but I would like to end this article with a plea to send in something. Remember that we all started as beginners, and in any case, nobody yet turns out perfect programs. Beginners especially often show originality which many old-timers cannot match.

Frank H. Druijff

```
100   GOTO 500:REM Gemaakt door Hendrik-Jan van Randen.
150   X1=XM-X1(T):Y1=YM-Y1(T):X2=XM-X2(T):Y2=YM-Y2(T)
160   DRAW X1(T),Y1(T) X2(T),Y2(T) Z:DRAW X1,Y1 X2,Y2 Z:DRAW X1,Y1(T) X2,Y2
(T) Z:DRAW X1(T),Y1 X2(T),Y2 Z:RETURN
200   FOR W=0 TO AL STEP A:K=21+RND(3):K(K-21)=1+RND(#F):COLORG 0 K(0) K(1)
 K(2):Z=20:T=W:GOSUB 150
210   X1(W)=RND(XM):Y1(W)=RND(YM):DX=(RND(XM)-X1(W))/A:DY=(RND(YM)-Y1(W))/A
:X2(W)=X1(O):Y2(W)=Y1(O):Z=K:GOSUB 150:V=W
250   FOR T=W+1 TO W+A-1:O=O+1:Z=20:GOSUB 150:X1(T)=X1(V)+DX:Y1(T)=Y1(V)+DY
:X2(T)=X1(O):Y2(T)=Y1(O):Z=K:GOSUB 150:V=T:NEXT
260   O=W:NEXT:W=0:GOTO 200
500   CLEAR 5000:MODE 6:MODE 6:AF=3:A=10:AL=AF*A-1:XM=XMAX:YM=YMAX:DIM X1(A
L),Y1(AL),X2(AL),Y2(AL),K(2):GOTO 200
```

---

```
10    GOTO 500:REM ZANDLOPER / F.H. DRUIJFF 2/82
20    Y=YR-1:DOT X,Y KZ:K=1-K:ON K GOTO 40
30    IF SCRN(X+1,Y-1)<>KA GOTO 50:DOT X,Y KA:X=X+1:Y=Y-1:DOT X,Y KZ:GOTO 3
0
40    IF SCRN(X-1,Y-1)=KA THEN DOT X,Y KA:X=X-1:Y=Y-1:DOT X,Y KZ:GOTO 40
50    IF YR>=44 GOTO 70:DOT X,110-Y KA
60    YR=YR+1-SGN(ABS(35-X)):X=35:GOTO 20
70    DRAW 35,YL 35,YL-2 KA:YL=YL-3:IF YL>42 GOTO 60
80    IF GETC=0 GOTO 80
500   CLEAR 5000:MODE 4:KA=0:KZ=10:KL=3:COLORG KA KR KZ KL
510   X=35:Y=65:FILL 24,79 46,103 KZ:READ A,B
520   FILL 25,104 45,105 KZ:DOT 25,105 KA:DOT 45,105 KA
530   READ C,D:IF C=999 GOTO 560:DRAW X+A,Y+B X+C,Y+D KL
540   DRAW X-A,Y-B X-C,Y-D KL:DRAW X+A,Y-B X+C,Y-D KL
550   DRAW X-A,Y+B X-C,Y+D KL:A=C:B=D:GOTO 530
560   Y=68:FOR I=1 TO 10:DRAW X-I,Y+I X+I,Y+I KZ:NEXT
570   IF SCRN(X,Y)=KA THEN DOT X,Y KZ:WAIT TIME 0:Y=Y-1:GOTO 570
580   YR=Y+2:YL=68:GOTO 20
800   DATA 1,0,1,3,12,14,12,60,11,60,11,61
810   DATA 10,61,10,62,0,62,999,0
```

---

```
5     REM BLOC IN REVERSE F.DRUIJFF
10    MODE 6:COLORG 0 10 10 0:XM=XMAX-1:YM=YMAX-1
15    FOR I=0 TO YMAX STEP 6:DRAW 0,0 XMAX,I 17:DRAW XMAX,0 0,I 17:NEXT
20    X=159:Y=183:S=20:XS=X+S:YS=Y+S:FILL X,Y XS,YS 19
30    H=GETC:IF H=0 GOTO 30
40    IF H<>16 GOTO 50
45    H=GETC:IF H<>0 GOTO 40:IF YS>=YM GOTO 30:DRAW X,YS+1 XS,YS+1 19:DRAW
X,Y XS,Y 18:YS=YS+1:Y=Y+1:GOTO 45
50    IF H<>17 GOTO 60
55    H=GETC:IF H<>0 GOTO 40:IF Y<=1 GOTO 30:DRAW X,Y-1 XS,Y-1 19:DRAW X,YS
 XS,YS 18:YS=YS-1:Y=Y-1:GOTO 55
60    IF H<>18 GOTO 70
65    H=GETC:IF H<>0 GOTO 40:IF X<=1 GOTO 30:DRAW X-1,Y X-1,YS 19:DRAW XS,Y
 XS,YS 18:X=X-1:XS=XS-1:GOTO 65
70    IF H<>19 GOTO 30
75    H=GETC:IF H<>0 GOTO 40:IF XS>=XM GOTO 30:DRAW XS+1,Y XS+1,YS 19:DRAW
X,Y X,YS 18:XS=XS+1:X=X+1:GOTO 75
80    GOTO 30
```

---

## RUN (LINE NUMBER) WITH BASIC V1.0

Nearly all DAI-users who have still the V1.0 BASIC ROM's will have undoubtedly been frequently upset after a non recoverable error occurred during program execution. A re-start will always empty the heap and symbol-table. After study of the differences between V1.0 and V1.1 , I found the following solution.

1) Type the object code of the short MLP, called RUNLIN in RAM, by using the Substitute feature in UTility.
   Note: The object code is relocatable.
2) Adjust the heap pointer 29B-29C, set pointer after last address object code.
3) Type NEW
4) Load your BASIC program and type RUN.
5) To re-start at a certain line number, first define a variable (integer), which is given the value of the required line number, followed by a call to the MLP.

   EXAMPLE: Assume start address MLP is #300.
            Re-start at line 100

Type the following commands in direct mode:

```
LINE%=100:CALLM #300,LINE%
```

The program will now automatically start at line 100, without emptying the heap and symbol-table.

Finally a few suggestions:

-After the object code is saved on tape, it can together with the BASIC program, easily be loaded with the earlier in DAINAMIC published Bootstrap-loader.Step 2 and 3 can be ommitted in this case
-If frequently is re-started at the same line-number,make the variable part of the BASIC program i.e. 10 LINE%=100 and re-start with only CALLM #300,LINE%.

success

G.GRUITERS

```
*RUNLIN BY G.GRUITERS 16-1-83
*
*THIS MLP IS ONLY 31 BYTES LONG
*AND RELOCATABLE!
*USERS WITH BASIC V1.0 CAN RE-START
*THEIR BASIC PROGRAM  RUN (LINE-NR)
*WITHOUT EMPTY-ING HEAP AND SYMBOL
*TABLE.
*
*       ORG   :300      START ADDR MLP
        INX   H         SKIP 1ST 2 ADDRESSES
        INX   H         OF VARIABLE IN SYMB-TABLE.
        MOV   D,M       NEXT 2 ADDRESSES CONTAIN
        INX   H         INTEGER LINE-NUMBER AND ARE
        MOV   E,M       MOVED IN REGISTER DE.
        XCHG            EXCHANGE DE WITH HL
        CALL  :CAF6     SEARCH LINE-NR IN TEXTBUFF
        MOV   B,H       ADDRESS TEXT LINE IN HL IS
        MOV   C,L       MOVED IN REG. B
        CALL  :E401     RESTORE ROUTINE
        LXI   H,0
        SHLD  :115      RESET TRACE/STEP FLAG
        ZAR
        STA   :126      NO SUSPENDED PROGRAM
        LXI   SP,:F900  RESET STACK-PNTR
        ORA   A         CLEAR FLAGS
        JMP   :C88F     RUN BASIC PROGRAM
        END
```

***RUNLIN OBJECT CODE***

```
23 23 56 23 5E EB CD F6 CA 44 4D CD 01 E4 21 00
00 22 15 01 AF 32 26 01 31 00 F9 B7 C3 8F C8
```

# DAI VIDEO-HARDWARE

(DAInamic 10, page 77-81)

1. MODIFYING THE DAI COMPUTER FROM INTERLACING TO NON-INTERLACING. (2)
A TV picture is made up of 625 lines. These lines are not traced on the screen all at once but in two goes. In other words, firstly the odd numbered lines are traced across the screen and then the even numbered ones. Of the 625 lines only about 550 are visible. Your DAI computer works in the same way but here the information on the odd and even lines is the same. Suppose that the next trace is an odd numbered line; the following trace, an even one, carries the same information as the previous one but will display it a little lower on your screen. The distance between the two traced lines is from 0.5 to 1.0 mm. A line trace lasts 20 mS (50 Hz). Summing up, the picture generated by your DAI flickers somewhat in an up-down fashion, with a periodicity of 2 x 20 mS, i.e. 40 mS, and that is the reason for a rather unsteady picture.

PROPOSED MODIFICATIONS. (3)
(In the diagram the Dutch word Aanbrengen means Provide and Verwijderen means Remove.)
Since pin 12 of IC25 is connected to ground (GND) under the chip on the component side of the printed circuit board it is difficult to break the connection. In order to free this pin it is best to snip through it immediately above the print (component side) and bend it up so that a wire can be run from it to pin 14.

(4) The advantage of the foregoing modification will be clear for all to see but there is also a disadvantage. If one is using the 20 mS interrupt (vector 7) in, for example a real time clock, then the clock will run a shade too fast.

f(crystal (ZNA 134)) = 2,562,500 Hz
f(line) = f(crystal) / 164 = 15625 Hz
f(raster) = f(line) / 312.0 (was 312.5) = (50 + 25 / 312) Hz

Thus the 20 mS interrupt now comes every 19.968 mS. Taking it further, the real time clock will gain 138 seconds every 24 hours, assuming that the crystal frequency is as above. (See also section 3 of this letter).

2. MODIFYING DAI's PAL COLOUR BOARD WITH RESPECT TO VIDEO AND COLOUR BANDWIDTH. (5)
I noticed that the test card broadcast by the Dutch TV stations,had a better resolution than the pictures produced by my DAI. The reason for this is that the video bandwidth of the DAI-PAL colour board is only 3MHz (-6dB). Similarly the colour bandwidth is only 0.75 MHz. Why has the DAI company chosen such low bandwidths? Perhaps to overcome interference? Television bandwidths are 5MHz and 1.5MHz respectively. After the modifications given in the following pages the TV interface bandwidths are 7MHz and 2.5MHz respectively. This produces an enormous improvement. Since doing the modification I have had no trouble from interference or similar problems. With some TV sets however there could be. Mine is a Philips with the KT3 chassis, 43cm 90 degree picture tube. The use of a 90 degree tube makes this set ideal for a DAI monitor as the improved focussing possibilities of the tube are better than needed for the original 5MHz bandwidth.

(6) Proposed modification of the DAI-PAL colour board (brightness).
(7) Proposed modification of the DAI-PAL colour board (colour).
The Dutch words in the two diagrams, (6) and (7) can be translated as follows:- wordt = becomes, verwijderen = remove, spoel = coil, en doorverbinden = and connect through, instel = instal (or fit). (trim, adjust)

3. ADJUSTING THE TV LINE FREQUENCY (15625 Hz). (8)
The frequency generated by the DAI was about 5MHz too low. This is not really serious for normal computer use, unless one attaches much importance to the accuracy of the 20 mS interrupt (V7). If the modifications described in section 1 have been done then the error is more serious. As I am a radio and television service amateur I attach great value to the accuracy of the line frequency, for the regulation and checking of the colour demodulator. In order to adjust the line frequency in your DAI a small modification has to be made, namely the replacement of a small capacitor by an adjustable one. (See page 9). Anyone who does not use his DAI as a test pattern generator has no need to bother with the alterations described in this section.

MODIFICATION AND ADJUSTMENT OF TV LINE FREQUENCY. (9)
(The note beneath the diagram reads :-) Replace capacitor C by a 50pF trimmer capacitor. Using the trimmer the frequency can be adjusted to 1,281,250.0 Hz. Connect the frequency counter to pin 10 of IC25 (ZNA 134).

4. ADJUSTMENT AND MODIFICATION OF THE COLOUR CARRIER WAVE. (10)
Using the DAI as a test pattern generator soon taught me that the carrier frequency was so high that some TV sets could not tolerate it, resulting in colour reproduction being over-coupled. To lower the frequency and correct it a 22pF trimmer capacitor is fitted in parallel with the crystal on the modulator board. The frequency must be very precisely adjusted and because my frequency counter was not accurate enough I had to devise another method of measurement - which was to equate the frequency with that of a TV transmitter by means of Lissajous patterns on an oscilloscope. One channel (of the scope) is connected to the input of the DAI's HF modulator (the small tin box on the modulator board, pin 4) while the other channel is connected to the output of the colour reference generator of the TV set. Type on the DAI : 100 COLORG 1 0 0 0:MODE 5:GOTO 100 followed by RUN.

(11). Tune your TV to a transmitter and check that the test card is still displayed in colour. Let the computer and the TV warm up for half an hour. Then adjust the newly fitted trimmer until the Lissajous pattern on the scope is stationary.

5. ADJUSTMENT OF THE SOUND CARRIER. (12)
In the same way, the frequency of the sound carrier is important when the DAI is used as a test pattern generator. A sharpened matchstick is needed to do the adjustment. Connect the frequency counter to pin 1 of the HF modulator. In the lid of the modulator are two small holes. The one you want is that nearest the side holding the connector. Adjust the frequency to 5,500 KHz. During this have 'SOUND OFF'. Adjustment was not necessary on my DAI.

6. PROGRAM 'TEST CARD'. (13)
Because, as you know, I am a service amateur I had a need for my own test pattern generator. This was the chief reason for modifying various parts of my DAI computer (see the previous sections). An example of an inconvenient picture is the test card of the Dutch transmitters. The real problem is that the DAI Basic V1.1 in graphic modes could not cope with the unseen parts, the most important features of a test card. That is why there is so much POKEing to the screen in the program, listed on pages 103 and 104. The circle has turned out a bit smaller because this time I wanted to make use of the basic support. This program is not expected to be suitable for all sorts of tuners, but it is all right in my practice where I want the same picture every time. Here is a short description of the program:-
Lines 100-999 the short main program
Lines 1000-1060 initialise the screen memory so as to push the screen up a little and add a bit below it.
Lines 2000-2804 place the white blocks around the frame of the picture (most are only partially

visible)
Lines 3000-3070 divide the picture into small rectangles.
Lines 4000-4050 draw a circle with short dashes. It is not used in the main program. When needed it is called by a GOSUB 4000 in line 210; change GOSUB 5000 to GOSUB 4000.
Lines 5000-5960 fill the circle in the middle of the test card. The circle is organised from top to bottom as follows:-
-red/yellow bar to check the delay line.
-black reflection bar to check for reflections and side lobes in the intermediate frequency band.
-graded grey bar for linearity check of the final video amplifier.
-colour bars to check the colour reproduction.
-there is a cross in the centre of the circle for convergence.
-frequency bars for checking the frequency characteristics of the IF section and video final amplifier(s).
-graded grey bar.
-yellow/red bar to check the delay line.
At the end of the program is my own version of SGT (slow graphic text). It is for the most part taken from the DAI handbook. Some precautions had to be anticipated and modified so that the text string could run separately from the picture. In addition the program has been extended to include an option for drawing text at various angles, in contrast to other versions of graphic text where this is only possible with angles which are multiples of 90 degrees. SGT can be called with a GOSUB 40000. In the main program one must insert an extra CLEAR 1400 for this subroutine. Variable A$ is the text to be printed. X and Y are the starting point co-ordinates. Variable DEV is the angle at which the string is to be printed, radially anti-clockwise. C is the colour code and F the enlargement factor for drawing the text.

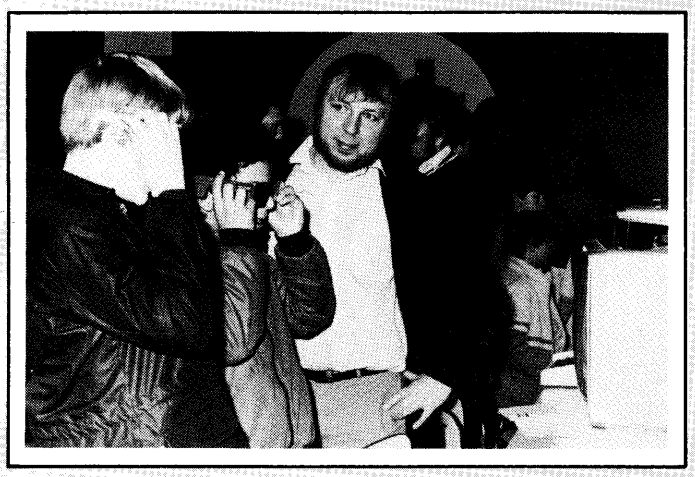
# DAInamic CATALOG 1-4-83

| CODE | TITLE | AUDIO | DCR |
|---|---|---|---|
| G1 | GAMES COLLECTION 1 | 400 | 550 |
| G2 | GAMES COLLECTION 2 | 400 | 550 |
| G3 | GAMES COLLECTION 3 | 400 | 550 |
| G4 | GAMES COLLECTION 4 | 800 | 950 |
| G5 | GAMES COLLECTION 5 | 400 | 550 |
| G6 | GAMES COLLECTION 6 | 750 | 900 |
| G7 | GAMES COLLECTION 7 | 750 | 900 |
| G8 | GAMES COLLECTION 8 | 750 | 900 |
| G9 | GAMES COLLECTION 9 | 750 | 900 |
| G10 | GAMES COLLECTION 10 | 750 | 900 |
| G11 | GAMES COLLECTION 11 | 750 | 900 |
| DNA | DNA ASSEMBLY PACK | 1100 | 1250 |
| FGT | FAST GRAF TEXT | 1000 | 1150 |
| FGTA | FGT-APPLICATIONS | 1000 | 1150 |
| TK1 | TOOLKIT 1 | 1000 | 1150 |
| TK2 | TOOLKIT 2 | 1000 | 1150 |
| TK3 | TOOLKIT 3 | 1000 | 1150 |
| TK4 | TOOLKIT 4 | 1000 | 1150 |
| PE1 | PRIMARY EDUCATION 1 | 1000 | 1150 |
| SE1 | SECUNDARY EDUCATION 1 | 1000 | 1150 |
| SE2 | SECUNDARY EDUCATION 2 | 1000 | 1150 |
| WP | WORD PROCESSOR I | 1000 | 1150 |
| ML | MAILING LIST | 1000 | 1150 |
| GT | GRAPHIC TABLET | 1000 | 1150 |
| M1 | MUSIC COLLECTION 1 | 300 | 450 |
| M2 | MUSIC COLLECTION 2 | 300 | 450 |
| M3 | MUSIC COLLECTION 3 | 300 | 450 |
| DTP | DAI TINY PASCAL | 1000 | 1150 |
| EGT | ENGLISH-GERMAN TRAINER | 1000 | 1150 |
| DD | DAI DEMO + BASIC TUTOR | 500 | 650 |
| CH | SARGON CHESS | 1500 | 1650 |
| SI | SPACE INVADERS I | 800 | 950 |
| T80 | TAPE 80-81 | 850 | 1000 |
| N10 | NEWSLETTER 10 | 500 | 650 |
| N11 | NEWSLETTER 11-12 | 650 | 800 |
| N13 | NEWSLETTER 13-14-15 | 650 | 800 |
| CTP | CENTIPEDE | 600 | 750 |
| DRI | DRIVER | 600 | 750 |
| SUI | SUPER INVADER | 600 | 750 |
| DTX | DAINATEXT | 2000 | 2150 |
| DAPA | DAIPANIC | 800 | 950 |
| JR | MICRO'S-ONDERWIJS | 990 | 1100 |
| SPL | SPL MACRO-ASSEMBLER | 1100 | 1250 |
| W3 | WISKUNDE 3 | 750 | 900 |
| TT1 | TAAL 1 | 750 | 900 |
| F1 | FYSICA 1 | 750 | 900 |
| FB | FAMILIEBUDGET | 500 | 650 |
| GH | GRAFISCHE HULP | 500 | 650 |
| ACR | ACROBATES | 600 | 750 |

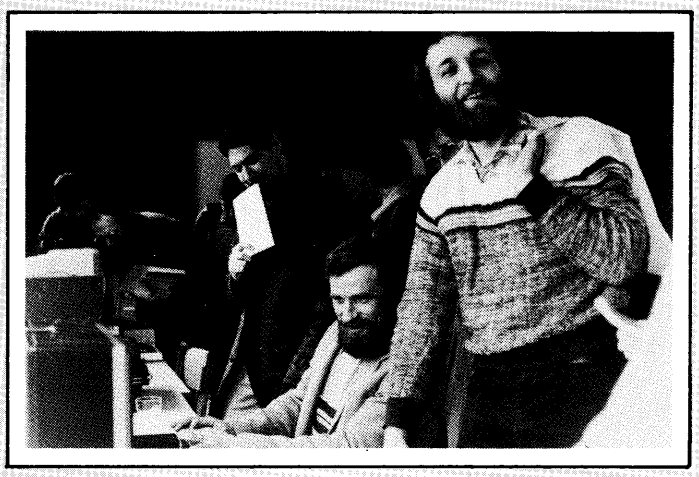
## HARDWARE & PUBLICATIONS

| | | |
|---|---|---|
| PCS | DAIpc SCHEMATICS | 850 |
| BOD | BEST of DAInamic (80-81) | 500 |
| SNG | SUPER NOISE GENERATOR | 1500 |
| NC | NEW CHARACTER GENERATOR | 1000 |
| DCE | DCE-INTERFACE-CARDS | 2500 |
| N8 | NEWSLETTERS 8-13 (1982) | 500 |

# DAInamic meeting 9th april 1983



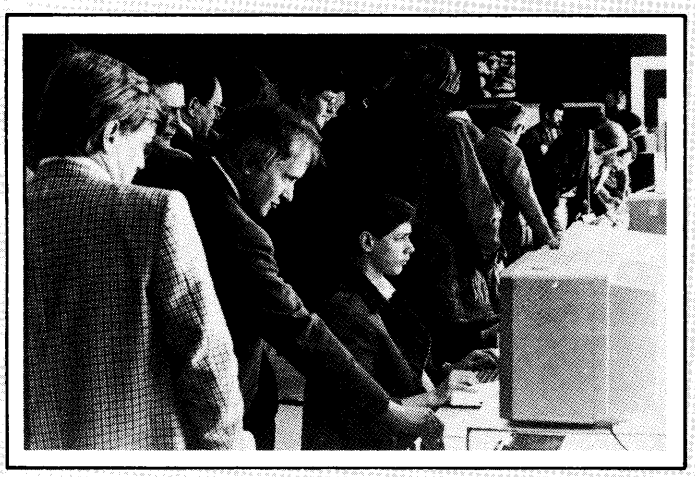
3-D illusion



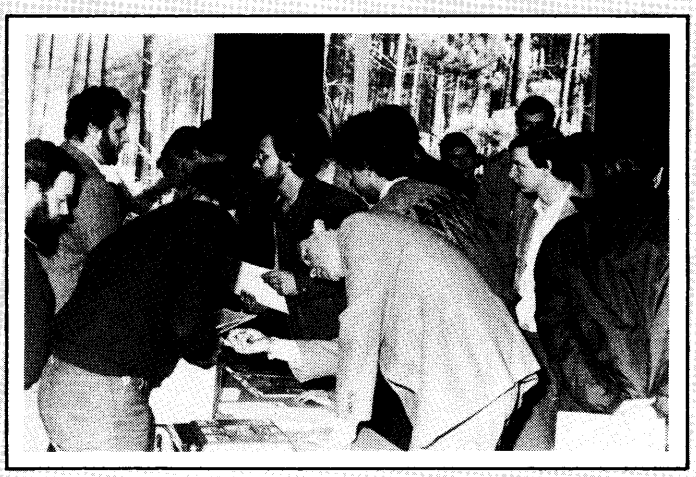
mode 7 & 8, SFGT ...

DAInamic library

diDAIsoft



a lot of software ...



DAInamic software

looking for peripherals

INDATA : present !