CompUser

INTERNATIONAL COMPUTING

ISSN: 0922-4203

Volume 2, Number 1, January 1989.

COMPUSER
International Computing

COMPUSER
Exchanging Computer Knowledge

CONTENTS VOLUME 2, NUMBER 1, JANUARY 1989.

COMPUSER is a magazine issued by:
COMPUSER International Computing;
Exchanging computer knowledge.
ISSN: 0922-4203.
Founded at Krimpen a.d. IJssel,
The Netherlands, by:
Willem L. van Pelt
Ir. Coen J. Boltjes.

Address all editorial correspondence to the editor COMPUSER:
Willem L. van Pelt
Jacob Jordaensstraat 15
NL 2923 CK Krimpen a.d. IJssel
The Netherlands.
Phone: (31) 01807 - 19881.

Regular participants:
Fred A. Behringer (W-Germany)
Hans Ebert (W-Germany)
Andrew Gregory (England)
Marc Lachaert (Belgium)
Iddy Oort (Holland)
Leif Rasmussen (Denmark)
Ronald van Vugt (Holland)

Drawings:
Herman Zondag (Holland)
Leo de Kok (Holland)

Translations:
Cor Bergshoeff (Denmark)
Piet K. de Vries (Holland)

(c) 1989 by:
COMPUSER International Computing.
Copying done for other than personal or internal reference use without the permission of the publisher is prohibited.
Practicing of the published programs and hardware etc. without responsibility of the publisher and for personal purpose only.
In no event will COMPUSER or the author be liable to you for any damages, including any lost profits, lost savings or other incidental or consequential damages arising out of the use of any of the published programs, schemes or articles.
The articles to be published have to be written by the sender. All in english language, unless one needs help by one of COMPUSER's translators.

To participate in COMPUSER International Computing the yearly contribution is :
Inside Europe :
ƒ 59,50 (bankcheque)
ƒ 50,00 (eurocheque or postgiro 841433 to W.L. van Pelt, Krimpen a.d.IJssel.
Outside Europe: ask for prices.

TRANSFER OCTOPUS INTO A DEVELOPMENT-SYSTEM !

By : Peter F. de Fauw, Wietzendorf, West-Germany.

From the very first beginning when I met OHIO's 6502 DOS (I'll call it OS-DOS) there were a lot of things that seemed very strange, unconfortable, even clumsy to me. E.g. the available DOS-commands/to have to work via BEXEC* and so on. In the past a lot of changes and trials for changes have been made on the disk SYSTEM-LOYS, of which the most succesfull ones were composed by prominents, who (unfortunately) are still gathered in COMPUSER. However, the moment I got the WORDPROCESSOR 3.1 at hand, I noticed this was not just another disk, and not only for the amazing WP3.1! The DOS can now finally also be called a DOS at the point of commands, it is now equiped with a flexible set of commands. Also the BASIC has had attention; e.g. it has now auto-line numbering and a printer-init. menu, another 'little' enhancement (but oh so handy) is the DEL-line command (how I missed it). Then, last but not least, the amazing WP3.1.....
I would say, say no more.. thanks LEIF!

As I was trained on, and used to work with 'development-systems', I found it very annoying that on boot-up the BASIC was loaded and BEXEC* started (development-systems are characterized by the phylosophy : have only loaded what you need now and so have the utmost of free RAM for development).
Put the case; I want to develop a program in machinecode, so I need the assembler and the editor (not a BASIC and a BEXEC* or do I know what!). So what do I need? I need a system that, when I boot it, loads the DOS and waits for a command.

Well with SYSTEM LOYS XI (the WORDPROCESSOR 3.1 disk) it is attractive and quite easy to build such a system, as the DOS command-set is so flexible now, that it is no longer necessary to revert to BEXEC* every time (and so BASIC has not to be loaded eather). Further it appeared that almost every initial POKE in the header of BEXEC* is now needles (build in the system now). So nothing stands in the way ...

First thing to do is to eliminate, in the Systemloader, the jump to the 'A, B or W'-menu by changing it into a jump into the DOS-command loop. Also the input-distributer has to be set to the 'keyboard' and at last the drive-head has to be unloaded.

The second step, is to look at the 'Transient Processor-Handler'. For our purpose we need an universal 'Handler', that when entrance is requested to a certain trapro, first look if it was already loaded, if so, just restart and if not, load it. Only for BASIC a few inits have to be made, it must also take care of that (the old 'Handler' is based on the fact that BASIC was loaded at boot-up).

The third and last step, is to take care of the 'trapro-prefix' that is visable just before the drive-indicator in the DOS-prompt. It indicates in which trapro you were before you entered the DOS. Right after boot-up you weren't in any transient-processor, so in that case there mustn't be a prefix. To solve that, 2 things have to be done; first let the Systemloader init the 'trapro-flag' to $03 (illegal), second change the prefix-handler in the DOS-commandloop, so that >02 now will cause no prefix to be printed.

According to the above three points, the System-loader (no not the bootstrap-loader, he resides in the Monitor-Eprom), the Transientprocessor-handler and the Trapro-prefix-handler are rewritten (sie listings).

Notice that a few patches have to be made on the new disk!

- the address for the wordprocessor in the DOS-command-dispatchtable is now: $E736 (was $E733)!
  [TT 01,1 : $2E46 = $33, change into : $36!]
- in the 'RE-enter' extensions the same has to be done, the 'enter'-address is now : $E737 (was E734)!
  [TT 14,1 : $D8C8 = $34, change into : $37!]
- the returnaddress of the Track 0 R/W utilities - Exit function has to be changed into a jump into the DOS-commandloop (was Basic).
  [TT 39,1 : $65E5 = $D1, change into : $2A!
  $65E6 = $E7, change into : $51!]

The COPIER that resides on the WORDPROCESSOR3.1 disk, is a modified version of the one modified by Dr. Tietsch. When your drive configuration differs from the one proposed by Dr. Tietsch, you have a problem. It also appears that the track00 reading still causes troubles. The first problem is solved by rewriting the 'check if on same disk'-routine, which, in the present form, checks if the source- and target drive are on the same drive, and at last also asks if they are on the same drive. So it is possible that the routine found out you're working with one drive, but you answer the question with 'no'. How confused he will be (you too). To solve this, the routine has been rewritten (see listing), and two bytes have to be chang-

ed: $CD51 & $CDB8 both contain $30, change into 10 ! (BMI becomes BPL).
The second problem is solved by making sure that $CFFF (last byte) is $00!

Working with the WORDPROCESSOR3.1 disk, a few little buggies seem to have remained, in BEXEC*.

When you make a hardcopy of the directory, you'll see the first filenames printed right after the =========-line, and the '..entries free' message also. The remedy is to transfer every 'PRINT' statement in that part into 'PRINT#DV,' now the statement is active on every (!) device selected. But also on the screen there is an empty line too much. At last, on screen the directory is centered between the two =========-lines, not so on the printer.

REMEDY : - line 175 has to start with 'DV=1' (default device)
- line 250 ends with twice the PRINT statement, remove one and change the other in 'PRINT#DV,'.
- line 265 starts with 3 x PRINT, remove two and change the remaining one in 'PRINT#DV,'.
- change in line 845 M=0 into M=1.
- change in line 925 M=0 into M=1.
- change in line 975 'PRINT" "' into 'PRINT#DV," ";'.

Because we have no longer a BASIC-system, we have to change entry-9 in the BEXEC*-menu "enter OS65D system" into "Basic" (it is just another trapro!). While entry-23 now becomes the system entrance, with :"Return to DOS".
So : - type in line 70 behind 23› :'Return to DOS"',
- therefore change line 1155 into: 'EXIT:END'.
- change in line 100 'enter OS65D system' into 'Basic'.

The only remaining problem is ASS-114, if you don't own it, you can put your assembler in the tracks 22-24. Notice that the cold start is now $0204, and the warmstart is now $228B!

A TIP: When you have a double-sided drive, and want to be able to run the word-processor-installation program quickly, or be able to read the 'manual' and 'readme' without exchanging disks; delete lines 1165 and 1170, and change in line 1180 'SE A' into 'SE C' (of course the message must also be changed).
In the header of 'Instal', the command 'DISK!"SE A"' should be placed!!

```
OS6502   ASSEMBLER            PAGE 01

0010:
0020:                  This program loads the Operating System, checks the
0030:                  RAM-memory part and initiates the System-hardware,
0040:                  then prints a message and jumps into the DOS.
0050:
0060:
0070:
0080: 2200                     ORG   $2200
0090:
0100:
0110:                  *** SYSTEM LOADER FOR SYSTEM DISK ***
0120:
0130:
0140:                  › Filename               : SYSLDR
0150:                  › Physical track (TT,S)  : 00,1
0160:                  › Physical memory        : $2200 - $22C0
0170:                  › Date                   : 01-03-'88
0180:
0190:
0200:
0210:
0220:                  *
0230:                  ***************************************************************
0240:                  *                        EXTERNALS                            *
0250:                  ***************************************************************
0260:                  *
0270:
0280: 2200             SECTNM *      $265E  sector number
0290: 2200             SETTK  *      $26BC  set track
0300: 2200             READDK *      $2967  read from disk
0310: 2200             LDHEAD *      $2754  load head
0320: 2200             MEMHI  *      $00FF  highest free page/load address
0330: 2200             MEMLO  *      $00FE
0340: 2200             UNLDHD *      $2761  unload head
```

```
0350: 2200            MEMCHK *      $22EC  memory checker
0360: 2200            MEMTOP *      $2300  high-byte of pointer to top of RAM
0370: 2200            INIKBD *      $F707  init par. keyboard I/O
0380: 2200            CTLCMD *      $F72D  get acia-setting and set it!
0390: 2200            GETFOT *      $F759  get screenformat setting
0400: 2200            MOVCRT *      $F3E1  move CRTC-inittable to RAM
0410: 2200            RESET  *      $F32F  reset screen
0420: 2200            INICEN *      $F6F1  init centronics I/O
0430: 2200            SERFLG *      $E7C7  serial I/O flag (MONI)
0440: 2200            PARFLG *      $E7C8  parallel I/O flag (MONI)
0450: 2200            INPFLG *      $2321  system input-flag
0460: 2200            OUTFLG *      $2322  system output-flag
0470: 2200            TRAPRO *      $E7F5  trapro-flag
0480: 2200            TOPTOP *      $D7FF  last free RAM byte
0490: 2200            STROUT *      $2D73  output text-string
0500: 2200            DOSLP  *      $2A51  DOS command-loop
0510:
0520:
0530:                 *
0540:                 ***************************************************************
0550:                 *                      MAIN ENTRY                             *
0560:                 ***************************************************************
0570:                 *
0580:
0590: 2200 4C 0C 22   ENTRY  JMP    START  go load the System
0600:
0610: 2203 8E 5E 26   READTT STX    SECTNM
0620: 2206 20 BC 26          JSR    SETTK
0630: 2209 4C 67 29          JMP    READDK *** READ TT,S AND RETURN ***
0640:
0650:
0660:                 *
0670:                 ***************************************************************
0680:                 *                    SYSTEM LOADER                            *
0690:                 ***************************************************************
0700:                 *
0710:
0720:
0730: 220C 20 54 27   START  JSR    LDHEAD load the drive-head
0740:
0750:
0760:                 *READ TT,S=01,1 ---> 2A00...31FF
0770:
0780: 220F A9 00             LDAIM $00
0790: 2211 A2 2A             LDXIM $2A      >>>2A00
0800: 2213 85 FE             STA    MEMLO
0810: 2215 86 FF             STX    MEMHI
0820: 2217 A9 01             LDAIM $01      TT--->A
0830: 2219 AA                TAX            S--->X
0840: 221A 20 03 22          JSR    READTT *** LOAD DOS/2 ***
0850:
0860:
0870:                 *READ TT,S=06,2 ---> 3200...32FF
0880:
0890: 221D A9 00             LDAIM $00
0900: 221F A2 32             LDXIM $32
0910: 2221 85 FE             STA    MEMLO
0920: 2223 86 FF             STX    MEMHI
0930: 2225 A9 06             LDAIM $06      TT--->A
0940: 2227 A2 02             LDXIM $02      S--->X
0950: 2229 20 03 22          JSR    READTT *** LOAD DOS/3 ***
0960:
0970:
0980:                 *READ TT,S=06,3 ---> 0000...00FF
0990:
1000: 222C A9 00             LDAIM $00
1010: 222E 85 FE             STA    MEMLO
1020: 2230 85 FF             STA    MEMHI
1030: 2232 A9 06             LDAIM $06      TT--->A
1040: 2234 A2 03             LDXIM $03      S--->X
1050: 2236 20 03 22          JSR    READTT *** LOAD DOS/0 ***
1060:
1070:
1080:                 *READ TT,S=06,4 ---> E400...E7FF
1090:
1100: 2239 A9 00             LDAIM $00
```

```
1110: 223B A2 E4            LDXIM $E4
1120: 223D 85 FE            STA   MEMLO
1130: 223F 86 FF            STX   MEMHI
1140: 2241 A9 06            LDAIM $06     TT--->A
1150: 2243 A2 04            LDXIM $04     S--->X
1160: 2245 20 03 22         JSR   READTT *** LOAD DOS/5 ***
1170:
1180:
1190:                 *READ TT,S=13,1 ---> 3274...3A73
1200:
1210: 2248 A9 74            LDAIM $74
1220: 224A A2 32            LDXIM $32
1230: 224C 85 FE            STA   MEMLO
1240: 224E 86 FF            STX   MEMHI
1250: 2250 A9 13            LDAIM $13     TT--->A
1260: 2252 A2 01            LDXIM $01     S--->X
1270: 2254 20 03 22         JSR   READTT *** LOAD DOS/4 ***
1280:
1290:
1300:                 *READ TT,S=14,1 ---> D800...DFFF
1310:
1320: 2257 A9 00            LDAIM $00
1330: 2259 A2 D8            LDXIM $D8
1340: 225B 85 FE            STA   MEMLO
1350: 225D 86 FF            STX   MEMHI
1360: 225F A9 14            LDAIM $14     TT--->A
1370: 2261 A2 01            LDXIM $01     S--->X
1380: 2263 20 03 22         JSR   READTT *** LOAD LR-XTRAS ***
1390:
1400: 2266 20 61 27  UNLOAD JSR   UNLDHD unload the drive-head
1410:
1420:
1430:
1440:                 *
1450:                 ****************************************************************
1460:                 *                      MEMORY CHECKER                          *
1470:                 ****************************************************************
1480:                 *
1490:
1500: 2269 A0 D7            LDYIM $D7     }
1510: 226B A9 00            LDAIM $00     } set highest
1520: 226D 85 FE            STA   MEMLO   } free page
1530: 226F 84 FF            STY   MEMHI   }
1540:
1550: 2271 20 EC 22  CHECK  JSR   MEMCHK ]
1560: 2274 F0 03            BEQ   CHKEND ] check
1570: 2276 88               DEY          ] memory
1580: 2277 D0 F8            BNE   CHECK  ]
1590:
1600: 2279 8C 00 23  CHKEND STY   MEMTOP reset 'top of RAM'-pointer
1610:
1620:
1630:
1640:                 *
1650:                 ****************************************************************
1660:                 *                      INIT THE SYSTEM                         *
1670:                 ****************************************************************
1680:                 *
1690:
1700: 227C 20 07 F7         JSR   INIKBD init keyboard I/O
1710: 227F 20 2D F7         JSR   CTLCMD get acia setting and set it !
1720: 2282 20 59 F7         JSR   GETFOT get screenformat setting
1730: 2285 20 E1 F3         JSR   MOVCRT move CRTC-inittable to RAM
1740: 2288 20 2F F3         JSR   RESET  reset the VDU and clear screen
1750: 228B 20 F1 F6         JSR   INICEN init the centronics I/O
1760:
1770: 228E A2 00            LDXIM $00
1780: 2290 8E C7 E7         STX   SERFLG reset serialflag
1790: 2293 8E C8 E7         STX   PARFLG reset parallelflag
1800:
1810: 2296 E8               INX
1820: 2297 8E 21 23         STX   INPFLG set input = keyboard
1830: 229A 8E 22 23         STX   OUTFLG set output = video
1840:
1850: 229D E8               INX
1860: 229E E8               INX
```

```
1870: 229F 8E F5 E7          STX   TRAPRO reset traproflag
1880:
1890: 22A2 A9 0D             LDAIM $0D
1900: 22A4 8D FF D7          STA   TOPTOP set last RAM-byte to 'CR'
1910:
1920:
1930:                  *
1940:                  ****************************************************************
1950:                  *                    PRINT START-UP MESSAGE                    *
1960:                  ****************************************************************
1970:                  *
1980:
1990: 22A7 20 73 2D  PRMESS JSR   STROUT print the message
2000:
2010: 22AA 0D                =     $0D     'CR'
2020: 22AB 0A                =     $0A     'LF'
2030: 22AC 0A                =     $0A     'LF'
2040: 22AD 2A                =     '*
2050: 22AE 4F                =     'O
2060: 22AF 53                =     'S
2070: 22B0 2D                =     '-
2080: 22B1 44                =     'D
2090: 22B2 4F                =     'O
2100: 22B3 53                =     'S
2110: 22B4 20                =     '
2120: 22B5 56                =     'V
2130: 22B6 33                =     '3
2140: 22B7 2E                =     '.
2150: 22B8 33                =     '3
2160: 22B9 2A                =     '*
2170: 22BA 0D                =     $0D     'CR'
2180: 22BB 0A                =     $0A     'LF'
2190: 22BC 0A                =     $0A     'LF'
2200: 22BD 00                =     $00     'end of message'-indicator
2210:
2220:
2230:                  *
2240:                  ****************************************************************
2250:                  *                    JUMP IN THE COMMAND-LOOP                  *
2260:                  ****************************************************************
2270:                  *
2280:
2290: 22BE 4C 51 2A  GODOS  JMP   DOSLP
2300:
2310:

OS6502   ASSEMBLER           PAGE 01


0010:                  This little program insures the right traproindicator
0020:                  printed before the drive indicator, when stepping into
0030:                  the DOS.
0040:
0050:
0060: 2BFD                   ORG   $2BFD
0070:
0080:                  *** TRANSIENT-PROCESSOR INDICATOR PRINTER ***
0090:
0100:                  > Filename               : TPINDI
0110:                  > Physical track (TT,S)  : 01,1
0120:                  > Physical memory        : $2BFD - $2C19
0130:                  > Date                   : 01-03-'88
0140:
0150:                  Returning from the BASIC         ; a    'b' is printed,
0160:                                the ASSEMBLER      ; an   'a' is printed,
0170:                                the WORDPROCESSOR; a    'w' is printed,
0180:
0190:                  if no transient-processor was entered before (boot-up)
0200:                  no prefix will be printed.
0210:
0220:
0230:                  *
0240:                  ****************************************************************
0250:                  *                       EXTERNALS                              *
0260:                  ****************************************************************
0270:                  *
```

```
0280:
0290: 2BFD                CRLF    *       $2D6A   print a 'CR' & 'LF'
0300: 2BFD                TRAPRO  *       $E7F5   Transient-Processor flag
0310: 2BFD                PRINT   *       $2343   print character in the accu
0320:
0330:
0340:                     *
0350:                     ***************************************************************
0360:                     *                       TRAPRO-INDICATOR PRINTER                *
0370:                     ***************************************************************
0380:                     *
0390:
0400: 2BFD 20 6A 2D       ENTRY   JSR     CRLF    print a 'CR' and a 'LF'
0410:
0420:
0430: 2C00 AD F5 E7               LDA     TRAPRO  get the flag
0440: 2C03 F0 0C                  BEQ     B       go print 'b' if BASIC
0450:
0460: 2C05 C9 01                  CMPIM   $01     or was it the ASSEMBLER ?
0470: 2C07 F0 05                  BEQ     A       if so, go print 'a' !
0480:
0490: 2C09 C9 02                  CMPIM   $02     or was it the WORDPROCESSOR ?
0500: 2C0B F0 07                  BEQ     W       if so, go print 'w' !
0510:
0520:
0530: 2C0D 2C             =       $2C     'BIT' (skip the next 2 instr.)
0540: 2C0E A9 61          A       LDAIM   $61     'a'
0550:
0560: 2C10 2C             =       $2C     'BIT'
0570: 2C11 A9 62          B       LDAIM   $62     'b'
0580:
0590: 2C13 2C             =       $2C     'BIT'
0600: 2C14 A9 77          W       LDAIM   $77     'w'
0610:
0620:
0630: 2C16 20 43 23               JSR     PRINT    go print the prefix!
0640:
0650: 2C19 60                     RTS               back to the command-loop

OS6502    ASSEMBLER               PAGE 01


0010:                     This program is a real universal Transient-Processor
0020:                     handler; now only three 'Trapro's' are supported, but
0030:                     it's easy to adapt others. But then there must be
0040:                     enough space on disk and the command-interpreter must
0050:                     be modified.
0060:
0070:
0080: E71D                        ORG     $E71D
0090:
0100:
0110:                     *** TRANSIENT-PROCESSOR HANDLER ***
0120:
0130:                     > Physical track (TT,S) : 06,4
0140:                     > Physical memory       : $E71D - $E7B2
0150:                     > Filename              : TPHAND
0160:                     > Date                  : 01-03-'88
0170:
0180:                     > Transient Processor Flag = 0 if BASIC
0190:                                                = 1 if ASSEMBLER
0200:                                                = 2 if WORDPROCESSOR
0210:
0220:
0230:                     *
0240:                     ***************************************************************
0250:                     *                     EXTERNAL DEFINITIONS                      *
0260:                     ***************************************************************
0270:                     *
0280:
0290: E71D                TRAPRO  *       $E7F5   Transient-Processor flag
0300: E71D                CLRSCN  *       $F32F   clear screen (EPROM)
0310: E71D                WBASIC  *       $20C4   BASIC warm start
0320: E71D                WAS/WP  *       $0204   ASSEMBLER / WORDPROCESSOR warm start
0330: E71D                DOSLP   *       $2A51   DOS command loop
0340: E71D                BASIC   *       $2AE6   BASIC cold start
```

```
0350: E71D            ASSMBL *      $2200  ASSEMBLER cold start
0360: E71D            WRDPRC *      $228B  WORDPROCESSOR cold start
0370: E71D            LDCMN  *      $2AEE  load 4 tracks to $0200...
0380: E71D            SELDRV *      $D906  select drive A & load head
0390: E71D            RAMTOP *      $2300  high-byte of pointer to RAM-top
0400:
0410:
0420:                 *
0430:                 ***************************************************************
0440:                 *                        TRAPRO ENTRIES                       *
0450:                 ***************************************************************
0460:                 *
0470:
0480: E71D A2 00      BA      LDXIM $00
0490: E71F EC F5 E7           CPX   TRAPRO BASIC active ?
0500: E722 D0 20              BNE   COMMON if not, force it !
0510: E724 20 2F F3           JSR   CLRSCN if so, clear screen
0520: E727 4C C4 20           JMP   WBASIC and goto BASIC-warm start
0530:
0540: E72A A2 01      AS      LDXIM $01
0550: E72C EC F5 E7           CPX   TRAPRO ASSEMBLER active ?
0560: E72F D0 13              BNE   COMMON if not, force it !
0570: E731 20 2F F3           JSR   CLRSCN if so, clear screen
0580: E734 4C 04 02           JMP   WAS/WP and goto ASSEMBLER
0590:
0600: E737 A2 02      WP      LDXIM $02
0610: E739 EC F5 E7           CPX   TRAPRO WORDPROCESSOR active ?
0620: E73C D0 06              BNE   COMMON if not, force it !
0630: E73E 20 2F F3           JSR   CLRSCN if so, clear screen
0640: E741 4C 04 02           JMP   WAS/WP and goto WORDPROCESSOR
0650:
0660:
0670:                 *
0680:                 ***************************************************************
0690:                 *                        HANGUP KILLER                        *
0700:                 ***************************************************************
0710:                 *
0720:
0730: E744 68         COMMON  PLA           remove
0740: E745 68                 PLA           return address of any 'JSR' entry
0750: E746 8A                 TXA           save
0760: E747 48                 PHA           current trapro-flag
0770: E748 20 06 D9           JSR   SELDRV  always select drive A & load head
0780: E74B 68                 PLA
0790: E74C AA                 TAX           restore trapro-flag
0800:
0810: E74D E0 03      FLGTST  CPXIM $03     check if flag illegal
0820: E74F B0 0E              BCS   ILLEGL  if so, handle it
0830: E751 E0 00              CPXIM $00     did you want BASIC ?
0840: E753 F0 10              BEQ   CBASIC  then go do cold start
0850: E755 E0 01              CPXIM $01     or ASSEMBLER ?
0860: E757 F0 18              BEQ   CASSEM  then go do ASSEMBLER
0870: E759 E0 02              CPXIM $02     or did you type 'WP' ?
0880: E75B F0 1F              BEQ   CWP     then go cold start WORDPROCESSOR
0890: E75D D0 EE              BNE   FLGTST  non of these 3? try again !
0900:
0910: E75F 20 2F F3   ILLEGL  JSR   CLRSCN when illegal, clear screen,
0920: E762 4C 51 2A           JMP   DOSLP   and return control to commandloop
0930:
0940:
0950:                 *
0960:                 ***************************************************************
0970:                 *                    TRAPRO COLD START HANDLER                *
0980:                 ***************************************************************
0990:                 *
1000:
1010: E765 8E F5 E7   CBASIC  STX   TRAPRO *** cold start the BASIC ***
1020: E768 20 2F F3           JSR   CLRSCN
1030: E76B 20 87 E7           JSR   INIBAS
1040: E76E 4C E6 2A           JMP   BASIC
1050:
1060: E771 8E F5 E7   CASSEM  STX   TRAPRO *** cold start the ASSEMBLER ***
1070: E774 A9 22              LDAIM $22
1080: E776 20 EE 2A           JSR   LDCMN
1090: E779 4C 00 22           JMP   ASSMBL
1100:
```

```
1110: E77C 8E F5 E7  CWP     STX   TRAPRO *** cold start the WORDPROCESSOR ***
1120: E77F A9 27             LDAIM $27
1130: E781 20 EE 2A          JSR   LDCMN
1140: E784 4C 8B 22          JMP   WRDPRC
1150:
1160:
1170:
1180:                  *
1190:                  ************************************************************
1200:                  *                    INITIALS FOR BASIC MODE                 *
1210:                  ************************************************************
1220:                  *
1230:
1240: E787 A9 51       INIBAS LDAIM $51    }
1250: E789 8D 55 2A          STA   $2A55   } preserve loopaddress
1260: E78C A9 2A             LDAIM $2A     } for 'EXCOM' in 'DOSLP'
1270: E78E 8D 57 2A          STA   $2A57   }
1280:
1290:
1300:
1310: E791 A9 9C             LDAIM $9C     }
1320: E793 A0 00             LDYIM $00     } init
1330: E795 8D FB 2B          STA   $2BFB   } for 'MEM'
1340: E798 8C FC 2B          STY   $2BFC   }
1350:
1360: E79B AD 00 23          LDA   RAMTOP
1370: E79E C9 D7             CMPIM $D7     if RAM-top is $D7
1380: E7A0 D0 0A             BNE SKIP
1390: E7A2 A9 00             LDAIM $00     ]
1400: E7A4 A0 DD             LDYIM $DD     ] init pointer
1410: E7A6 8D 62 E6          STA   $E662   ] to DD00
1420: E7A9 8C 63 E6          STY   $E663   ]
1430:
1440: E7AC 60          SKIP  RTS

OS6502   ASSEMBLER           PAGE 01


0010:                  This routine replaces the 'check if on same disk'-
0020:                  routine, of the COPIER on the System-disk; the
0030:                  routine has been changed for the configuration
0040:                  and the foolish question at the end.
0050:
0060:
0070: CAD0                   ORG   $CAD0
0080:
0090:
0100:                  *** DRIVE CONFIGURATION WATCHER ***
0110:
0120:                  > Filename              : DRWTCH
0130:                  > Physical track (TT,S) : 37,1
0140:                  > Physical memory       : $CAD0 - $CAF4
0150:                  > Date                  : 01-03-'88
0160:
0170:                  valid configuration : DRIVE 1 = side A & side C
0180:                                        DRIVE 2 = side B & side D
0190:
0200:
0210:
0220:                  *
0230:                  ************************************************************
0240:                  *                    EXTERNALS                               *
0250:                  ************************************************************
0260:                  *
0270:
0280: CAD0             FRMDRV *      $CFF0  drive to copy FROM - flag
0290: CAD0             TODRV  *      $CFF1  drive to copy TO - FLAG
0300: CAD0             DRVFLG *      $CFF8  indicates source & target on same disk
0310:
0320:                  *
0330:                  ************************************************************
0340:                  *                    WATCHER                                 *
0350:                  ************************************************************
0360:                  *
0370:
0380: CAD0 A9 FF             LDAIM $FF     set drive-flag to
```

```
0390: CAD2 8D F8 CF          STA   DRVFLG 'same-drive'
0400:
0410: CAD5 AD F1 CF          LDA   TODRV
0420: CAD8 CD F0 CF          CMP   FRMDRV
0430: CADB F0 0F             BEQ   RETURN if TO=FROM return
0440: CADD 10 0E             BPL   PRPCHK if TO<FROM prepare check
0450:
0460: CADF AC F0 CF          LDY   FRMDRV if TO>FROM
0470: CAE2 C8                INY
0480: CAE3 C8                INY
0490:
0500: CAE4 CC F1 CF  COMPAR  CPY   TODRV  check if TO=FROM
0510: CAE7 F0 03             BEQ   RETURN if so, return
0520: CAE9 EE F8 CF          INC   DRVFLG else set drive-flag to 'not-same'drive'
0530:
0540: CAEC 60        RETURN  RTS
0550:
0560: CAED AC F0 CF  PRPCHK  LDY   FRMDRV if TO<FROM
0570: CAF0 88                DEY
0580: CAF1 88                DEY           then FROM=FROM-2
0590:
0600: CAF2 18                CLC
0610: CAF3 90 EF             BCC   COMPAR force branch-always
0620:
-
```

A SMALL BASIC ROUTINE TO PRINT OUT THE "COMPUSER - HEADER"

by: Hans Ebert, Ringseisstr. 1, 8000 München 2

If you are writing for the CompUser magazine it is helpful to have a routine to print the CompUser header for your paper. This small BASIC routine will do this.

```
100 ' ------------------------------------------------
110 ' Printer routine to print out the CompUser - Header
115 ' Printers: Epson / IBM XL / Star NL 10 / Star NG 10
120 ' ------------------------------------------------
130 '
140 ' Start:
145 '
150     CLS
160     LOCATE 10,10
170     PRINT "Please turn your printer on"
180     C$=INPUT$(1)
190 '
200 ' PrintOutHeader:
210 '
220      LPRINT CHR$(27);"W1";          'expanded on
230      LPRINT CHR$(27);"G";           'boldface
240      LPRINT CHR$(18);               'Elite
250      LPRINT CHR$(27);"M";
260      LPRINT tab(1);"COMPUSER                    COMPUSER";
270      LPRINT CHR$(13);
280      LPRINT CHR$(27);"W0";          'expanded off
290      LPRINT TAB(1);"International computering";
300      LPRINT TAB(51);"Exchanging computer knowledge"
310 '
320 ' EndProgram:
325 '
330      CLS
340      END
```

READ TIME AND DATE

EC-65 15:16:12 - 03/12/88 Page: 0001

```
0005:                       TTL   READ TIME AND DATE
0010:
0015: 3590                  ORG   $3590
0020:
0025:                 ****************************************
0030:                 *     O C T O P U S  /  E C 6 5.       *
0035:                 * Interrupt program to present time and *
0040:                 * date continuous in the upper right -  *
0045:                 * corner of the screen.                 *
0050:                 *                                       *
0055:                 *     written 21.07.1988  by:           *
0060:                 *     Peter Lindstroem                  *
0065:                 *     Solhaven 8,  DK 2990 Nivaa,       *
0070:                 *     D E N M A R K.                    *
0075:                 ****************************************
0080:
0085:                 Rev. 03.12.1988.  (Stop-routine)
0090:
0095:                 This program is desisned to be used together
0100:                 with ELEKTOR's 'real time clock' april -
0105:                 1985.
0110:
0115:                 Remember to make wire-link 'M' on the
0120:                 PC-board, then just run the INIT routine,
0125:                 and the time and date will appear on the
0130:                 upper right corner of the screen.
0135:
0140:                 Some programs can't run when the IRQ-
0145:                 clock is running, fx. COPIER and EDMO,
0150:                 so use the STOP-routine to stop the clock
0155:                 before using this other programs.
0160:
0165:                 There is room for the program in the
0170:                 DOS system from $3590. (trk 13,1)
0175:                 Move the object-code to $3590 and:
0180:                 SA 13,1=3274/8.
0185:
0190:
0195: FA6B            NIBASC  EQU   $FA6B   convert a hex nibble to an ASCII chr
0200: E120            CLKADR  EQU   $E120   clock-address
0205: E121            CLKDAT  EQU   $E121   clock-data
0210: E83C            BASE    EQU   $E83C   Video memory
0215: 00F0            VDU     EQU   $00F0   VDU pointer
0220: E7CD            FLN     EQU   $E7CD   first line pointer
0225: E7CB            IRQJMP  EQU   $E7CB   interrupt vector
0230:
0235:
0240:                 *** INIT ROUTINE ***
0245:
0250: 3590 A9 CE      INIT    LDAIM BEGIN   set IRQ vector
0255: 3592 8D CB E7           STA   IRQJMP
0260: 3595 A9 35              LDAIM BEGIN   /
0265: 3597 8D CC E7           STA   IRQJMP  +01
0270:
0275: 359A A9 0F              LDAIM $0F     set periodic IRQ rate to 500 ms
0280: 359C A2 0A              LDXIM $0A
0285: 359E 8E 20 E1           STX   CLKADR
0290: 35A1 8D 21 E1           STA   CLKDAT
0295: 35A4 E8                 INX
0300:
0305: 35A5 A9 4A              LDAIM $4A     set mode
0310: 35A7 8E 20 E1           STX   CLKADR
0315: 35AA 8D 21 E1           STA   CLKDAT
0320: 35AD 58                 CLI           clear IRQ flag
0325: 35AE 60                 RTS
0330:
0335:
0340:                 *** STOP ROUTINE ***
0345:
0350: 35AF 78         STOP    SEI           interrupt disable
0355: 35B0 A9 FB              LDAIM $FB     reset IRQ vector
0360: 35B2 A2 5C              LDXIM $5C
0365: 35B4 8E CB E7           STX   IRQJMP
```

READ TIME AND DATE

EC-65 15:16:13 - 03/12/88 Page: 0002

```
0370: 35B7 8D CC E7        STA   IRQJMP  +01
0375: 35BA A9 00           LDAIM $00
0380: 35BC A2 0A           LDXIM $0A
0385: 35BE 8E 20 E1        STX   CLKADR  reg. A
0390: 35C1 8D 21 E1        STA   CLKDAT  reset rate selector
0395: 35C4 E8              INX
0400: 35C5 A9 0A           LDAIM $0A
0405: 35C7 8E 20 E1        STX   CLKADR  reg. B
0410: 35CA 8D 21 E1        STA   CLKDAT  set mode
0415: 35CD 60              RTS
0420:
0425:              *** INTERRUPT ROUTINE ***
0430:
0435: 35CE 08      BEGIN   PHP           save all registers
0440: 35CF 48              PHA
0445: 35D0 98              TYA
0450: 35D1 48              PHA
0455: 35D2 8A              TXA
0460: 35D3 48              PHA
0465:
0470: 35D4 A2 0C           LDXIM $0C     reset the IRQ flag
0475: 35D6 8E 20 E1        STX   CLKADR
0480: 35D9 AD 21 E1        LDA   CLKDAT
0485:
0490: 35DC A2 FF           LDXIM $FF
0495: 35DE 8E 9F 36        STX   INDEX
0500:
0505: 35E1 18              CLC           calculate VDU address
0510: 35E2 A9 3C           LDAIM BASE
0515: 35E4 6D CD E7        ADC   FLN
0520: 35E7 85 F0           STA   VDU
0525: 35E9 A9 E8           LDAIM BASE    /
0530: 35EB 6D CE E7        ADC   FLN     +01
0535: 35EE 85 F1           STA   VDU     +01
0540:
0545: 35F0 20 5F 36        JSR   GETDATA get hour
0550: 35F3 AC 9F 36        LDY   INDEX
0555: 35F6 20 74 36        JSR   WRITE   write hour
0560:
0565: 35F9 A9 3A           LDAIM ':
0570: 35FB C8              INY
0575: 35FC 91 F0           STAIY VDU
0580: 35FE 8C 9F 36        STY   INDEX
0585: 3601 20 5F 36        JSR   GETDATA get minute
0590: 3604 AC 9F 36        LDY   INDEX
0595: 3607 20 74 36        JSR   WRITE   write minute
0600:
0605: 360A A9 3A           LDAIM ':
0610: 360C C8              INY
0615: 360D 91 F0           STAIY VDU
0620: 360F 8C 9F 36        STY   INDEX
0625: 3612 20 5F 36        JSR   GETDATA get seconds
0630: 3615 AC 9F 36        LDY   INDEX
0635: 3618 20 74 36        JSR   WRITE   write seconds
0640:
0645: 361B C8              INY
0650: 361C A9 20           LDAIM $20
0655: 361E 91 F0           STAIY VDU     write space
0660: 3620 C8              INY
0665: 3621 A9 2D           LDAIM '-
0670: 3623 91 F0           STAIY VDU     write '-'
0675: 3625 A9 20           LDAIM $20
0680: 3627 C8              INY
0685: 3628 91 F0           STAIY VDU
0690: 362A 8C 9F 36        STY   INDEX
0695: 362D 20 5F 36        JSR   GETDATA get date
0700: 3630 AC 9F 36        LDY   INDEX
0705: 3633 20 74 36        JSR   WRITE   write date
0710:
0715: 3636 A9 2F           LDAIM '/
0720: 3638 C8              INY
0725: 3639 91 F0           STAIY VDU
0730: 363B 8C 9F 36        STY   INDEX
```

```
READ TIME AND DATE

EC-65   15:16:15 - 03/12/88                Page: 0003

0735: 363E 20 5F 36          JSR   GETDATA get month
0740: 3641 AC 8F 36          LDY   INDEX
0745: 3644 20 74 36          JSR   WRITE   write month
0750:
0755: 3647 A9 2F             LDAIM '/
0760: 3649 C8                INY
0765: 364A 91 F0             STAIY VDU
0770: 364C 8C 8F 36          STY   INDEX
0775: 364F 20 5F 36          JSR   GETDATA get year
0780: 3652 AC 8F 36          LDY   INDEX
0785: 3655 20 74 36          JSR   WRITE   write year
0790:
0795: 3658 68                PLA           restore all registers
0800: 3659 AA                TAX
0805: 365A 68                PLA
0810: 365B A8                TAY
0815: 365C 68                PLA
0820: 365D 28                PLP
0825: 365E 40                RTI
0830:
0835:
0840:                *** SUBROUTINES ***
0845:
0850: 365F E8        GETDATA INX
0855: 3660 A0 0A             LDYIM $0A
0860: 3662 8C 20 E1  UPDATE  STY   CLKADR
0865: 3665 AD 21 E1          LDA   CLKDAT  update in progress?
0870: 3668 30 FB             BMI   UPDATE  yes, wait
0875: 366A BC 89 36          LDYX  REG_NO
0880: 366D 8C 20 E1          STY   CLKADR
0885: 3670 AD 21 E1          LDA   CLKDAT
0890: 3673 60                RTS
0895:
0900: 3674 C8        WRITE   INY
0905: 3675 48                PHA
0910: 3676 4A                LSRA
0915: 3677 4A                LSRA
0920: 3678 4A                LSRA
0925: 3679 4A                LSRA
0930: 367A 20 EB FA          JSR   NIBASC
0935: 367D 91 F0             STAIY VDU
0940: 367F C8                INY
0945: 3680 68                PLA
0950: 3681 29 0F             ANDIM $0F
0955: 3683 20 EB FA          JSR   NIBASC
0960: 3686 91 F0             STAIY VDU
0965: 3688 60                RTS
0970:
0975:
0980:
0985: 3689 04        REG_NO  DFB   $04     hours
0990: 368A 02                DFB   $02     minutes
0995: 368B 00                DFB   $00     seconds
1000: 368C 07                DFB   $07     date
1005: 368D 08                DFB   $08     month
1010: 368E 09                DFB   $09     year
1015:
1020: 368F 01        INDEX   DFB   $01

))) Error in 0000 statement(s)

))) Op-Code: $8000 - $80FF / $3590 - $368F / 0256 Bytes / 01 Page(s)

))) Assembled by ASS114 / 3.4
```

SECOND HAND MARKET

Universal Memory Board 8K RAM + Eprom Elektor Print Service EPS 80120 Sept. 1980, including 16 * 2114 RAMs and all other components, exluding ROMs. Two boards available. Price per board: Hfl. 109,50 on bankcheque or Hfl. 100,00 on eurocheque to be sent to W.L. van Pelt, Jacob Jordaensstr. 15, NL 2923 CK Krimpen a.d. IJssel.

Delta Elektronika C 5-5 Power Supply. Input 2 * 8.5V AC Output 5V DC 5A, with 2 * 2N3055 on separated cooler. Price : Hfl. 84,50 on bankcheque or Hfl. 75,00 on eurocheque to be sent to W.L. van Pelt, Jacob Jordaensstr. 15, NL 2923 CK Krimpen a.d. IJssel.

Elektor's EPS 82159 Floppy Disc Interface for EC65 (Octopus) Computer, with components. Price : Hfl. 69,50 on bankcheque or Hfl. 60,00 on eurocheque to be sent to W.L. van Pelt, Jacob Jordaensstr. 15, NL 2923 CK Krimpen a.d. IJssel.

Two home made disk interface (euro)-cards with INS1771-1 controller. Call for the price: W.L. van Pelt, editorial office, Holland, Tel.: 01807 - 19881.

JUNIOR - DOS65

By : Erik van den Broek, Holland.

It is realy rather easy to implement the DOS65 (as far as I know todays best keyboard-driven-65XX(X)-operating system), once you have build a JUNIOR equiped with a VDU-card. You wouldn't say so if you glance at the drawing below, but if you look better you will see that it is all rather straightforward in fact.

Features:
- switch-selection between: 1) bit-by-bit, pin-by-pin compatible (VDU)Junior
  2) simultanious Junior/DOS65 operation
  3) bit-by-bit, pin-by-pin compatible DOS65
- no change in PCB's
- all JUNIOR software remains the same (even cassette)
- easy debugging

Requirements for the Junior-with-Interface-and-VDU-card-owner:

- FDC-card (incl. parts) *	± ƒ 120,=
- DOS65 V2.01 hardware-manual / operating-manual / floppy *	± ƒ 65,=
- 2764 I/O65 EPROM *	± ƒ 25,=
- switch, 2 ways, 3 positions	± ƒ 8,=
- 6522, for timing and keyboard	± ƒ 15,=
- 6116, for storage of DOS65 variables (=garbage-ram)	± ƒ 8,=
- 56 K RAM (= 7 IC's 6264)	± ƒ 60,=
- 74LS00; 74LS04 (2); 74LS20; 74LS133; 74LS138; 74LS157	± ƒ 8,=
- Extra (do it yourself)print, connector, sockets, etc.	± ƒ 75,=
- Floppy-drive (SHUGART-compatible (is not IBM-comp.))	± ƒ 450,= ?
pessimistic estimation:	± ƒ 834,=

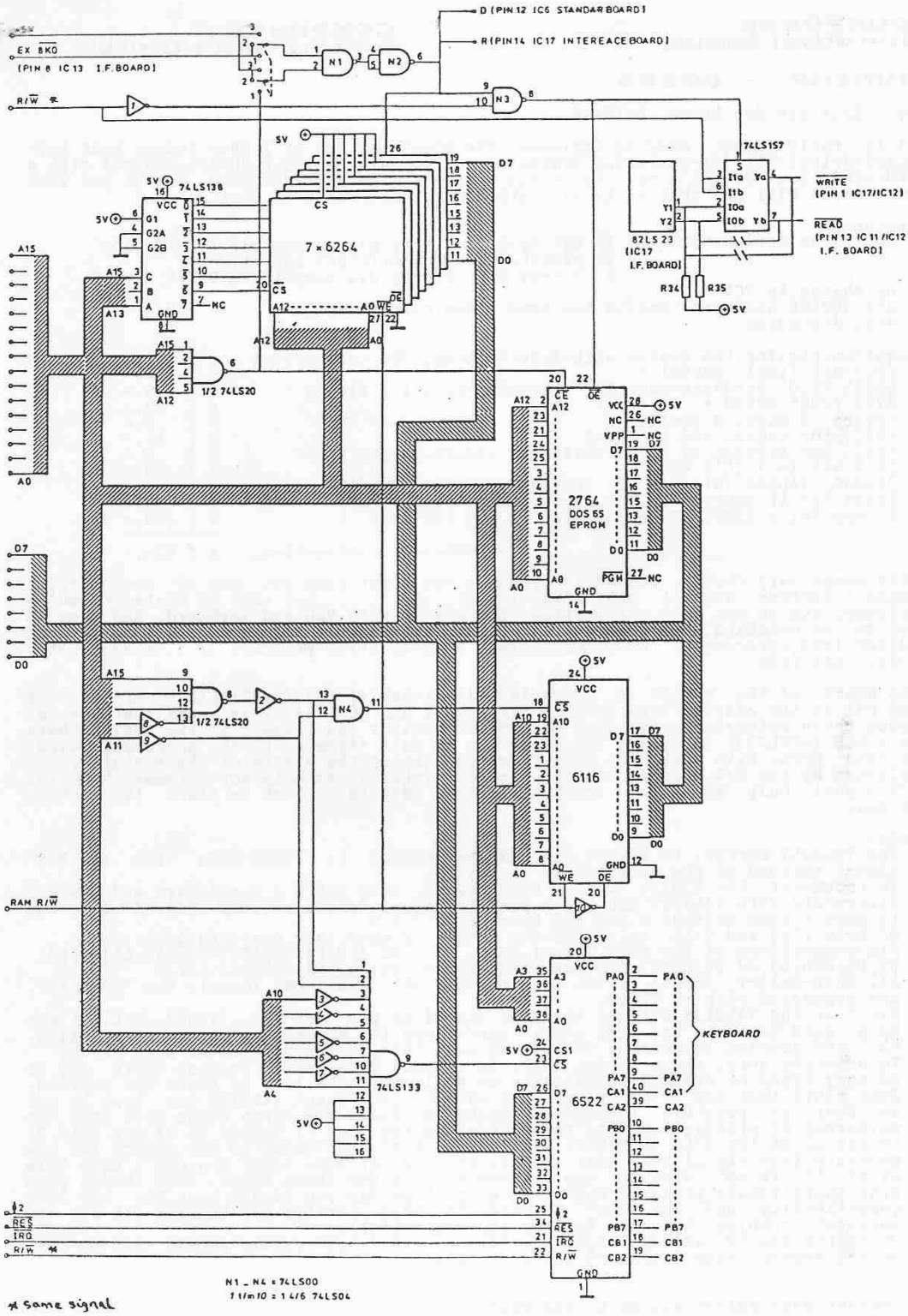
This seems very costly, but it is probably the first time you see an amount which doesn't express what it only costs if, if, if ....., but what it costs if you're unlucky, and do not have yet anything but JUNIOR with VDU and keyboard. And you do get an astonishing good computer, because of all the software like full-screen-editor (=wordprocessor), macro-assembler, communication programs (f.i. modem-protocols), all free.

The heart of the matter is to replace (switchable) the (small)RAM's, EPROMs, VIA and PIA in the address-area $0000 - $1FFF by 1 RAM IC 6264. This means that you can debug these extensions with your old JUNIOR-monitor and tracer. In the JUNIOR there is a ROM (=82LS23) to select the direction of data (from or to the processor-board) in this area. With only RAM, this direction (being the status of the busbuffer) is selected by the R/W signal. A digital switch (=LS157) selects now between ROM- and R/W-signal. Only while you are putting this (digital)switch in place, the machine is dead.

Notes:
- The 74LS157 serves, to direct the databusbuffers in DOS65-mode with the R/W-signal instead of the promsignals Y1 and Y2.
- To reconnect the 82S23: remove from socket, bend pin 1 & 2 sideways and connect discretely with 74LS157 pin 2 & 5 respectively. Remove pull-up R's 34 & 35 and reconnect them between mains and those pins 2 & 5.
- Of both 2764 and 6116, only half the memory is used (you don't notice).
- Pin-connections of the 6264 (pin 1 until 28): NC;A12;A7;A6;A5;A4;A3;A2;A1;A0;D0; D1;D2;GND;D3;D4;D5;D6;D7;CS1(low);A10;OE(low);A11;A9;A8;CS2;WE(low);+
- All chip-select inputs of the 6264's (=pin 26), EXEPT ONE (namely the 'lowest'), are connected with +5 Volts.
- Pin 7 of the 74LS138 carries the same signal as pin 6 of the 74LS20, but it may be a good idea to use two extra cards: one for memory and one for EPROM 6116, 6522 and address-selection. The signal must then be represented on both cards.
- To describe here, where to put what, is a waste of effort, because there may be as many types of JUNIORS (physically as well as 'mentally') as there are JUNIORs. Some hints that may help however: the switch, 74LS00 and 74LS157 can best be put as near as possible to the motherboard. There are three wires then from the motherboard/interface-unit to the card(s) on the bus: 1) to pin 26 of one 6264 2) to pin 22 of the 2764 (=OE not) 3) from pin 6 of the 74LS20 to the switch (if you generate this signal very near the motherboard yet once more, you don't need this wire). It is not wise to use bus-connections for these wires, even though some pins there remain probably unused forever. Some day you forget your bus has lost compatibility and implement a card intended for the new ELEKTOR-65K-bus (no incompat. problem there but for two pins (16A/16C)).
- Of course the 'D' and 'R' signal-inputs on both (old) cards, should not be connected anymore with either +5 Volts or mass.

* Contact your editor Willem L. van Pelt.

JUNIOR _ DOS 65



Herman Zondag

```
        ttl "tp DOS65 Directory sort utility V0.10          %s              page %d"
;
        pag     66
;
;file            dirsort.MAC
;
;last modified   8th April 1988
;
;program         DIRSORT
;
;function        put a directory in alpabetical order
;
;usage           DIRSORT dir:
;
;by              Andrew Gregory
;                35 Stafford Road
;                Sidcup
;                Kent
;                DA14 6PU
;                England
;
;
;DOS65 routines and variables
        lib     dvar.mac
        opt     lis
;
;main workspace
;
secmem  equ     $20
;
;zero page workspace
;
        org     $80
ndirs   res     1               number of directories loaded
subd    res     1               sub-dir
ndrive  res     1
tmp     res     1               workspace
a0      res     2               workspace
a1      res     2               address pointers to filenames
a2      res     2
;
s2      res     2
s3      res     2
s4      res     2
s5      res     2
cbp     res     2
order   res     1
curdir  res     1
;
        org     $1000
        jmp     begin
;
;absolute workspace
fdirsec res     1               first sector
fdirtk  res     1
;
;give help reply
        fcc     $C8,$C5,$CC,$D0
help    jsr     print
        fcc     'Directory sort utility V0.10\r'
        fcc     'Syntax  : DIRSORT dir:\r'
        fcc     'Options : -Y Do not ask for permission\r'
        fcc     'Example : DIRSORT 0:A/\r'
        fcc     'Sorts subdirectory A on drive 0',0
;
        fcc     'By A.P.Gregory',0
;
4       jsr     print
        fcc     'Illegal -options\r',0
        jmp     13.f
;
begin   php                     save flags
        jsr     sopt
        fcc     'Y',0
```

```
            bcs     4.b             if error
            sty     cbp
            sta     cbp+1           command string address
            jsr     sync            printer spooler would
                                    -interfere by changing rwpoin
;
            lda     udrive          default is U:
            jsr     separ           set fdrive and subd
;
            ldy     #-1             skip spaces
3           iny
            lda     [cbp],y         get char from string
            beq     begin2          default if none
            cmp     #13             CR?
            beq     begin2
            cmp     #32             Space
            beq     3.b
;
;First character is a '0', '1', '2', 'U', 'S', or 'W'
            cmp     #'3
            bcc     3.f             number
;
;'S', 'U' or 'W'
            jsr     loupch          upper case
            ldx     #2              search table for it
1           cmp     drtab,x
            beq     2.f
            dex
            bpl     1.b
            bmi     12.f            illegal
2           lda     sdrive,x
            jsr     separ           separate drive and dir
            jsr     ckcolon
            bne     12.f
            jsr     cklast
            bne     12.f            if error
begin2      jmp     begin1          proceed
;
drtab       fcc     'SUW'
;
separ       pha
            and     #%11
            sta     fdrive
            sta     ndrive
            pla
            lsra                    get sub-dir
            lsra
            sta     subd
            rts
;
ckcolon     iny
            lda     [cbp],y
            cmp     #':             nothing else will do
            rts
;
cklast      iny
            lda     [cbp],y
            beq     1.f
            cmp     #13
            beq     1.f
            cmp     #32
1           rts
;
;illegal directory
12          jsr     print
            fcc     'Illegal drive/ directory specification\r',0
13          jsr     seter
exit        plp
            rts                 return to DOS65
;
;
;'0', '1' or '2'
3           sec
            sbc     #'0
            bmi     12.b            error
```

```
                sta     ndrive          drive number
                sta     fdrive          needed by dirinit
                jsr     ckcolon         check next is colon
                bne     12.b
                iny                     next char @, A..G
                ldx     #0
                stx     subd            @ is default sub-dir
                lda     [cbp],y
                beq     begin1          default is @
                cmp     #13             CR?
                beq     begin1
                cmp     #32             Space
                beq     begin1
                cmp     #'@
                beq     begin1
                jsr     loupch          Upper case Y saved
                cmp     #'H
                bcs     121.f           error
                sbc     #'A-2
                bmi     121.f           error
                sta     subd
                iny
                lda     [cbp],y
                cmp     #'/
                beq     1.f
                dey                     / not compulsory
        1       jsr     cklast          check no more
                beq     begin1
        121     jmp     12.b
        ;
        begin1  lda     opt             -Y option?
                bmi     1.f
                jsr     print
                fcc     'Sort directory ',0
                lda     fdrive          print dr:dir/
                clc
                adc     #'0
                jsr     out
                lda     #':
                jsr     out
                lda     subd
                bne     2.f
                lda     #'@
                bne     3.f
        2       clc
                adc     #'A-1
        3       jsr     out
                jsr     print
                fcc     '/   (Y/N*)? ',0
                jsr     bufin
                jsr     loupch
                cmp     #'Y
                bne     exit1
        ;
        1       jsr     dirinit         open directory
                bcs     doserr
                lda     #0
                sta     ndirs
                lda     #secmem
                sta     curdir          next dir sec
                lda     subd            @ directory?
                beq     20.f            if yes
        ;
                asla                    multiply by 2
                clc
                adc     #s.dir-2        get directory pos
                tay
                lda     [rwpoin],y
                tax                     track
                iny
                lda     [rwpoin],y      sector
                beq     exit1           if non-existent
                tay
                jmp     21.f
```

```
;
exit1   jmp     exit
;
doserr  jsr     ermes           error
        jmp     exit
;
20      ldx     #dir0tk
        ldy     #dir0sc
;
21      stx     fdirtk          first directory trk
        sty     fdirsec         first directory sector
;
;directory loading loop
lloop   lda     curdir
        cmp     #$A1            max 32 sectors
        bcs     3.f
        jsr     ldsec           loads sec, sets rwpoin
        bcs     doserr
        inc     ndirs
        jsr     ndts            get X and Y of next sector
        beq     2.f             last if sector 0
        inc     curdir
        jmp     lloop
;
3       jsr     print
        fcc     '*** Directory too big\r',0
        jmp     exit
;
2       lda     ndirs
        cmp     #3
        bcc     1.f
        jsr     print
        fcc     'Please wait...\r',0
1       jsr     zero            Obliterate all invalid names
        jsr     sort
        lda     #$40
        jsr     clstcon         disable ^C
;
;save sorted  directory
;Note - surplus directory sectors are not freed.
;DOS65 does not provide a mechanism for doing this.
sloop   lda     #secmem
        sta     curdir
        ldx     fdirtk
        ldy     fdirsec
sloop1  lda     curdir
        jsr     svsec           save sector
        bcs     doserr1
        inc     curdir          next memory
        jsr     ndts            set track-sector
        bne     sloop1          unless end
        jmp     exit
;
doserr1 jmp     doserr
;
;simple shell sort routine.
;Compare each pair and interchange if wrong way round.
;repeat until in order.
sort    lda     ndirs
        jsr     mul15           s2 is number of filenames
        lda     s2
        ora     s2
        bne     sort1
        rts                     return if empty dir
;
sort1   lda     #0
        sta     order           in order flag
        jsr     sort5
        lda     order           if further sorting needed
        bne     sort1
        rts
;
sort5   ldx     s2              set up s4 to last number+1
        stx     s4
```

```
            ldx     s2+1
            stx     s4+1
            jmp     sort2
;
sort3       lda     s4+1            address of s4 into a1
            ldx     s4
            ldy     #0
            jsr     entad1
            lda     s5+1            address of s5 into a2
            ldx     s5
            ldy     #2
            jsr     entad1
;
            jsr     coma2a1         compare them
            bcc     sort2           branch if in order
            lda     #$ff            set flag
            sta     order
            jsr     swpa2a1         swap them
;
sort2       dec     s4              higher in list
            bne     2.f
            lda     s4+1
            beq     1.f             if reached top
            dec     s4+1
2           lda     s4
            sec
            sbc     #1              s5 one above s4
            sta     s5
            lda     s4+1
            sbc     #0
            sta     s5+1
            jmp     sort3
1           rts
;
;routine to compare names at [a2] and
;[a1] and set carry flag if interchange
;is needed.
coma2a1     ldy     #0
            lda     [a1],y          lower entry zero?
            beq     3.f             do not swap if so
            lda     [a2],y          one above [a1]
            beq     4.f             swap if zero
5           pha
            lda     [a1],y          ignore bit 7
            and     #$7f
            sta     tmp
            pla
            cmp     tmp
            bcc     2.f             if lower entry higher
            bne     4.f             if not same
            iny
            lda     [a2],y
            and     #$7f
            bne     5.b             no swap if zero
3           clc
2           rts
;
4           sec
            rts
;
;routine to interchange names [a2] and [a1]
swpa2a1     ldy     #15
1           lda     [a2],y
            pha
            lda     [a1],y
            sta     [a2],y
            pla
            sta     [a1],y
            dey
            bpl     1.b
            rts
;
;fill all invalid filenames with zeros
zero        jsr     s2s3set
```

```
            jmp     zloop1
;
zloop       jsr     entaddr
            ldy     #0
            lda     [a1],y
            bmi     1.f             if deleted
            bne     zloop2          if valid
1           tya
            ldy     #15             fill with 0's
2           sta     [a1],y
            dey
            bpl     2.b
zloop2      inc     s3
            bne     zloop1
            inc     s3+1
zloop1      dec     s2
            bne     zloop
            dec     s2+1
            bne     zloop
            rts
;
;prepare for scan through directory memory.
s2s3set lda     #0
            sta     s3              zero s3
            sta     s3+1
s2set       lda     ndirs
            jsr     mul15           max entries in s2
            inc     s2
            inc     s2+1
            rts
;
;routine to multiply A by 15 (A<50)
;answer in s2
mul15       sta     s2
            ldx     #0
            stx     s2+1
            asla                    by 2
            asla                    by 4
            clc
            adc     s2              by 5
            sta     s2
            asl     s2              by 10
            rol     s2+1
            clc
            adc     s2              by 15
            sta     s2
            lda     s2+1
            adc     #0
            sta     s2+1
            rts
;
;routine to convert entry number s3 to
;an address and store in a1
;divide by 15. Answer is page, remainder*16 pos.
entaddr lda     s3+1
            ldx     s3
            ldy     #0
;
;entry point to put answer in a1+y (y even)
entad1  stx     a0                  store low part
            sta     a0+1            and high part
            tya
            pha
            ldx     #16             16 bits
            lda     #0
            tay
1           asl     a0
            rol     a0+1
            rola
            cmp     #15
            bcc     2.f
            sbc     #15
2           pha
            tya
```

```
            rola
            tay
            pla
            dex
            bne     1.b
            asla
            asla
            asla
            asla
            sta     a0              offset in page
            tya                     page number
            clc
            adc     #secmem         add start address
            sta     a0+1
            pla
            tax
            lda     a0
            sta     a1,x
            lda     a0+1
            sta     a1+1,x
            rts
;
;Make X=track and Y=sector of next directory.
;set Z if none
;Note rwpoin must be set up.
ndts        ldy     #240
            lda     [rwpoin],y      track of next dir
            tax
            iny
            lda     [rwpoin],y
            tay                     sector of next dir
            rts
;
;subroutines to load and save logical sector Y track X at page A
;system sector must be loaded.
;save
ldsec       jsr     rwset
            jsr     readsect        reads logical sector
            rts                     C=1 if error
;save
svsec       jsr     rwset
            jsr     writsect
            rts
;;
rwset       sta     rwpoin+1        set page address and
            lda     #0              prepare for read/writesect
            sta     rwpoin
            lda     ndrive          drive number
            rts
```