Elsewhere in this issue we described the theory behind Basicode-2 so it is only natural that we should show how the Junior Computer can use it. Here we give the Basicode software and everything else that is necessary to allow the Junior Computer to use Basicode-2. This means that the Junior Computer can now easily exchange BASIC programs stored on cassette tape with other computers. Moreover, 'received' programs can run directly on the JC, so that BASIC in combination with Basicode is a universal, completely exchangeable computer language.

basicode-2
interface for the Junior Computer

Junior communicates with other computers

As we have already described all the various facets of Basicode, we will simply begin here by talking about the Junior with Basicode. The Basicode translation programs for the expanded Junior and the DOS Junior are not the same as they use different BASICs and handle their memory in different ways. To use Basicode, either an expanded Junior with KB-9-BASIC and Elekterminal or a DOS Junior and Elekterminal are needed.

The translation programs

The translation programs for both Junior versions are written in machine code. The complete source listing is given in table 1, complete with explanatory text. This is for the expanded Junior with KB-9-BASIC. The source listing for the DOS Junior is not given as it is almost the same as this listing, only a few of the addresses are different. The hexdumps are shown in table 2 (Junior with KB-9-BASIC) and table 3 (DOS Junior).

In the expanded Junior with KB-9-BASIC the translation program is at addresses $Ø2ØØ . . . $Ø59B, and in the DOS Junior it is at $EØØØ . . . $E39B. These ranges are selected because there is generally RAM there, and the programs really have to be in RAM to work properly (so they cannot be placed in an EPROM). Once the program is typed in, it can simply be written to a cassette or floppy disk, so that the next time it is to be used it can easily be read in. The program consists of a write and a read section. We will concentrate on the expanded Junior in order to describe how the program is used, but at any point where the DOS Junior differs, this is indicated by the comments in brackets.

Reading

First the Basicode translation program is typed in (or read in, if it is already stored on cassette). Both read and write programs can be stored in one file on cassette: SA = Ø2ØØ, EA = Ø59C (DOS Junior: SA = EØØØ, EA = E39C). Next the KB-9-BASIC is read in from cassette (see Elektor April 1982), or from a floppy in the case of the DOS Junior. Then the BASIC can be started in the usual way. At this stage a Basicode program can be loaded. This requires the small interface described at the end of this article. A program is loaded as follows:

First type NEW to erase any old programs.
Then type:
POKE 8256,Ø : POKE 8257,4 : X = USR(X)
(POKE 574,Ø : POKE 575,226 : X = USR(X))
followed by a (carriage) return.

The sign ≡ now appears on the hex display of the Junior and indicates that there is *no* synchronization. The cassette recorder can then be started. If the program receives any signals the ≡ sign jumps back and forth on the two right-hand displays. If the 2400 Hz header is now received, a slowly jumping sign appears on the right-hand displays. This shows that the program is working on synchronizing. This jumping display lasts about 2 seconds, then the sign is stationary on both displays for the rest of the leader. At the end of the leader when the actual program begins, both displays show //, and as long as the data is properly received this sign lights evenly on both displays. When the complete program is received, the computer automatically gives a listing of it on the screen or printer. After doing this, the computer gives an 'OK'.

If an error has appeared while reading in the program, the message 'CHECKSUM ERROR' is given after the listing. In this case the program must be checked or it could be read in again in the hope of a better result. On no account must the listing be interrupted by pressing a key. If this is done, there is a chance that both BASIC and Basicode programs will have to be read again (or retyped!). Even if faults are seen in the listing, such as lines being written over one another (that can happen if there is sudden interference on the tape), you must still wait until the

Table 1

```
0010:
0020:                    BASICODE WRITE PROGRAM FOR JUNIOR COMPUTER
0030:                    WITH KB-9 BASIC
0040:
0050:                    DATE:14-6-'83
0060:
0070: 0200                      ORG   $0200
0080:
0090:                    POINTERS IN PAGE ZERO (BASIC)
0100:
0110: 0200               INTER  *     $0004
0120: 0200               BFRSPL *     $007E  BEGINNING OF FREE SPACE POINTER
0130: 0200               BFRSPH *     $007F
0140: 0200               HIMEML *     $0084  END OF RAM POINTER
0150: 0200               HIMEMH *     $0085
0160:
0170:                    6522-IC REGISTERS
0180:
0190: 0200               TOCL   *     $1804  TIMER ONE COUNTER
0200: 0200               TOCH   *     $1805
0210: 0200               TOLL   *     $1806  TIMER ONE LATCH
0220: 0200               TOLH   *     $1807
0230: 0200               ACR    *     $180B  AUXILIARY CONTROL REGISTER
0240: 0200               IFR    *     $180D  INTERRUPT FLAG REGISTER
0250: 0200               IER    *     $180E  INTERRUPT ENABLE REGISTER
0260:
0270:                    OUTPUT VECTOR (BASIC)
0280:
0290: 0200               OUTVL  *     $2A52  OUTPUT VECTOR
0300: 0200               OUTVH  *     $2A53
0310:
0320:                    TEMPORARY DATA BUFFERS
0330:
0340: 0200               SOVL   *     $03F0
0350: 0200               SOVH   *     $03F1
0360: 0200               SAVINL *     $03F2
0370: 0200               SAVINH *     $03F3
0380: 0200               CHSUM  *     $03F4
0390: 0200               DESTFL *     $03FB
0400:
0410:                    EXTERNAL SUBROUTINE
0420:
0430: 0200               MESSY  *     $0559
0440:
0450:                    :
0460:                    : INITIALISE POINTERS
0470:                    : TABLE ROUTINE STARTADDRESS TO OUTPUT VECTOR
0480:                    :
0490:
0500: 0200 AD 52 2A      SAVE   LDA   OUTVL
0510: 0203 AC 53 2A             LDY   OUTVH
0520: 0206 8D F0 03             STA   SOVL
0530: 0209 8C F1 03             STY   SOVH   SAVE OUTPUT VECTOR
0540: 020C A9 42                LDAIM TABLE
0550: 020E A0 02                LDYIM TABLE  /256
0560: 0210 8D 52 2A             STA   OUTVL
0570: 0213 8C 53 2A             STY   OUTVH  NEW OUTPUT VECTOR
0580: 0216 A5 04                LDA   INTER
0590: 0218 A4 05                LDY   INTER  +01
0600: 021A 8D F2 03             STA   SAVINL
0610: 021D 8C F3 03             STY   SAVINH SAVE INTER VECTOR
0620: 0220 A9 77                LDAIM SVECAS
0630: 0222 A0 02                LDYIM SVECAS /256
0640: 0224 85 04                STA   INTER
0650: 0226 84 05                STY   INTER  +01 NEW INTER VECTOR
0660: 0228 A5 7E                LDA   BFRSPL
0670: 022A A4 7F                LDY   BFRSPH
0680: 022C 8D 49 02             STA   STTABL +01 BEGINNING FREE SPACE POINTER
0690: 022F 8C 4A 02             STY   STTABL +02 TO INDIRECT ADDRESSES STTABL AND
0700: 0232 8D A0 02             STA   LDTABL +01 LDTABL ( TABLE POINTERS )
0710: 0235 8C A1 02             STY   LDTABL +02
0720: 0238 A9 00                LDAIM $00
0730: 023A 8D F4 03             STA   CHSUM  CLEAR CHECKSUM
0740: 023D 8D FB 03             STA   DESTFL CLEAR DESTROY FLAG
0750: 0240 A9 02                LDAIM $02    START OF TEXT CHARACTER
0760:
0770:                    :
0780:                    : THIS ROUTINE IS CALLED BY BASIC AND PUTS THE
0790:                    : LISTING OF THE PROGRAM IN A TABLE
0800:                    :
0810:
0820: 0242 C9 0A         TABLE  CMPIM $0A
0830: 0244 F0 30                BEQ   OK     SKIP LF'S
0840: 0246 09 80                ORAIM $80
0850: 0248 8D FF FF      STTABL STA   $FFFF  CHARACTER TO TABLE
0860: 024B 4D F4 03             EOR   CHSUM
0870: 024E 8D F4 03             STA   CHSUM  UPDATE CHECKSUM
0880: 0251 EE 49 02             INC   STTABL +01 INCREMENT TABLE POINTER
0890: 0254 D0 03                BNE   OUTOFM
0900: 0256 EE 4A 02             INC   STTABL +02
0910: 0259 A5 85         OUTOFM LDA   HIMEMH
0920: 025B CD 4A 02             CMP   STTABL +02
0930: 025E D0 16                BNE   OK     NOT END OF RAM? THEN BRANCH
0940: 0260 A5 84                LDA   HIMEML
0950: 0262 CD 49 02             CMP   STTABL +01
0960: 0265 D0 0F                BNE   OK
0970: 0267 A9 FF                LDAIM $FF
0980: 0269 8D FB 03             STA   DESTFL SET DESTROY FLAG
0990: 026C A5 78                LDA   $0078       GO ON AT BEGINNING OF BASIC WORKSPACE
1000: 026E 8D 49 02             STA   STTABL +01  AND REWRITE BASIC PROGRAM BY TABLE
1010: 0271 A5 79                LDA   $0079
1020: 0273 8D 4A 02             STA   STTABL +02
1030: 0276 60            OK     RTS
1040:
1050:                    :
1060:                    : PROGRAM TO TAPE
1070:                    :
1080:
1090: 0277 48            SVECAS PHA
1100: 0278 98                   TYA
1110: 0279 48                   PHA
1120: 027A A9 0D                LDAIM $0D
1130: 027C 20 42 02             JSR   TABLE  CARRIAGE RETURN TO TABLE
1140: 027F A9 03                LDAIM $03
1150: 0281 20 42 02             JSR   TABLE  END OF TEXT TO TABLE
1160: 0284 AD F4 03             LDA   CHSUM
1170: 0287 20 48 02             JSR   STTABL CHECKSUM TO TABLE (WITHOUT SETTING BIT7)
1180: 028A A9 7F                LDAIM $7F
1190: 028C 8D 0E 18             STA   IER    SET INTERRUPT DISABLE MODE
1200: 028F A9 C0                LDAIM $C0
1210: 0291 8D 0B 18             STA   ACR    SET SQUARE WAVE OUTPUT (PB7)
1220: 0294 A9 01                LDAIM $01
1230: 0296 8D 05 18             STA   TOCH   START TIMER ONE
1240: 0299 20 2B 03             JSR   HEADER
1250: 029C 20 39 03      LOADTA JSR   ZERO   STARTBIT=0
1260: 029F AD FF FF      LDTABL LDA   $FFFF
1270: 02A2 A0 08                LDYIM $08    BITCOUNTER=8
1280: 02A4 4A            OUTCHA LSRA         ONE BIT TO CARRY
1290: 02A5 48                   PHA
1300: 02A6 B0 05                BCS   HIGH
1310: 02A8 20 39 03             JSR   ZERO   ONE PERIOD OF 1200HZ (=0)
1320: 02AB 70 03                BVS   NEXT   BRANCH ALWAYS
1330: 02AD 20 46 03      HIGH   JSR   ONE    TWO PERIODS OF 2400HZ (=1)
1340: 02B0 68            NEXT   PLA
1350: 02B1 88                   DEY
1360: 02B2 D0 F0                BNE   OUTCHA BITCOUNTER NOT ZERO? THEN BRANCH
1370: 02B4 20 46 03             JSR   ONE    STOPBIT=1
1380: 02B7 20 46 03             JSR   ONE    STOPBIT=1
1390: 02BA EE A0 02             INC   LDTABL +01 INCREMENT TABLE POINTER
1400: 02BD D0 03                BNE   LDT
1410: 02BF EE A1 02             INC   LDTABL +02
1420: 02C2 AC A1 02      LDT    LDY   LDTABL +02
1430: 02C5 AE A0 02             LDX   LDTABL +01
1440: 02C8 C4 85                CPY   HIMEMH
1450: 02CA D0 0E                BNE   CMPARE NOT END OF RAM? THEN BRANCH
1460: 02CC E4 84                CPX   HIMEML
1470: 02CE D0 0A                BNE   CMPARE
1480: 02D0 A5 78                LDA   $0078
1490: 02D2 8D A0 02             STA   LDTABL +01  GO ON AT BEGINNING OF BASIC WORKSPACE
1500: 02D5 A5 79                LDA   $0079
1510: 02D7 8D A1 02             STA   LDTABL +02
1520: 02DA CC 4A 02      CMPARE CPY   STTABL +02
1530: 02DD D0 BD                BNE   LOADTA NOT END OF TABLE? THEN BRANCH
1540: 02DF EC 49 02             CPX   STTABL +01
1550: 02E2 D0 B8                BNE   LOADTA
1560: 02E4 20 2B 03             JSR   HEADER
1570: 02E7 8C 0B 18             STY   ACR    DISABLE PB7
1580: 02EA AD F2 03             LDA   SAVINL
1590: 02ED AC F3 03             LDY   SAVINH
1600: 02F0 85 04                STA   INTER  RESET INTER VECTOR
1610: 02F2 84 05                STY   INTER  +01
1620: 02F4 AD F0 03             LDA   SOVL
1630: 02F7 AC F1 03             LDY   SOVH
1640: 02FA 8D 52 2A             STA   OUTVL  RESET OUTPUT VECTOR
1650: 02FD 8C 53 2A             STY   OUTVH
1660: 0300 AD FB 03             LDA   DESTFL
1670: 0303 F0 20                BEQ   RTRN   DESTROY FLAG NOT SET? THEN BRANCH
1680: 0305 A5 79                LDA   $0079  DESTROY BASIC PROGRAM IF DESTFL IS SET
1690: 0307 85 7B                STA   $007B
1700: 0309 85 7D                STA   $007D
1710: 030B 85 7F                STA   $007F
1720: 030D A4 78                LDY   $0078
1730: 030F C8                   INY
1740: 0310 C8                   INY
1750: 0311 C8                   INY
1760: 0312 84 7A                STY   $007A
1770: 0314 84 7C                STY   $007C
1780: 0316 84 7E                STY   $007E
1790: 0318 A9 00                LDAIM $00
1800: 031A A8                   TAY
1810: 031B 91 78                STAIY $0078
1820: 031D C8                   INY
1830: 031E 91 78                STAIY $0078
1840: 0320 A0 2C                LDYIM $2C
1850: 0322 20 59 05             JSR   MESSY  "NEW" MESSAGE
1860: 0325 68            RTRN   PLA
1870: 0326 A8                   TAY
1880: 0327 68                   PLA
1890: 0328 4C 03 00             JMP   INTER  -01 RETURN TO BASIC
1900:
1910:                    :
1920:                    : 5 SEC 2400HZ
1930:                    :
1940:
1950: 032B A2 70         HEADER LDXIM $70    SET X- AND Y-REGISTER FOR
1960: 032D A0 17                LDYIM $17    12000 PERIODS OF 2400HZ (=5 SEC 2400HZ)
1970: 032F 20 46 03      HDR    JSR   ONE    TWO PERIODS OF 2400HZ
1980: 0332 CA                   DEX
1990: 0333 D0 FA                BNE   HDR
2000: 0335 88                   DEY
2010: 0336 D0 F7                BNE   HDR
2020: 0338 60                   RTS
2030:
2040:                    :
2050:                    : ZERO=ONE PERIOD OF 1200HZ
2060:                    : ONE =TWO PERIODS OF 2400HZ
2070:                    :
2080:
2090: 0339 A9 9F         ZERO   LDAIM $9F
2100: 033B 8D 06 18             STA   TOLL
2110: 033E A9 01                LDAIM $01
2120: 0340 8D 07 18             STA   TOLH   SET TIMER ONE FOR 417 MICRO SEC.
2130: 0343 4C 53 03             JMP   PERIOD
2140: 0346 20 49 03      ONE    JSR   SUBONE
2150: 0349 A9 CE         SUBONE LDAIM $CE
2160: 034B 8D 06 18             STA   TOLL
2170: 034E A9 00                LDAIM $00
2180: 0350 8D 07 18             STA   TOLH   SET TIMER ONE FOR 208 MICRO SEC.
2190: 0353 AD 04 18      PERIOD LDA   TOCL   CLEAR TIMER ONE INTERRUPT FLAG
2200: 0356 2C 0D 18      PER    BIT   IFR
2210: 0359 50 FB                BVC   PER    NOT TIME-OFF?
2220: 035B AD 04 18             LDA   TOCL
2230: 035E 2C 0D 18      PRD    BIT   IFR
2240: 0361 50 FB                BVC   PRD
2250: 0363 60                   RTS
```

computer is finished with the listing and gives the 'OK' or 'CHECKSUM ERROR' message. Then by simply working in BASIC, you can check the program and correct it. There is also a possibility that the computer may not recognize the end of the program and carries on as if it were reading the program, and the hex display remains lighting. In this case the RST key could be pressed, but then the BASIC would have to be read in again. That is not the ideal situation. A better solution is to look for the end of another Basicode program on the tape and play this out. The computer will then recognize this end and will report back on the screen. Of course, the last part of the program read in will no longer be correct but at least you can examine the part of the program that is correct and in this case the BASIC does not have to be read in again.

If the program that must be read in is too large for the available memory space, the computer returns with 'OUT OF MEMORY', and no listing appears. If a listing of the part that is written in is required it can be obtained as follows:

POKE 8256,156 : POKE 8257,4 : X = USR(X)
(POKE 574,156 : POKE 575,226 : X = USR (X))
followed by a (carriage) return.

Writing

A BASIC program is written out in the following manner using Basicode:

```
0020:                 BASICODE READ PROGRAM FOR JUNIOR COMPUTER
0030:                 WITH KB-9 BASIC
0040:
0050:                 DATE:14-6-'83
0060:
0070: 0400                  ORG   $0400
0080:
0090:                 POINTERS IN PAGE ZERO (BASIC)
0100:
0110: 0400            BFRSPL *     $007E  BEGINNING OF FREE SPACE POINTER
0120: 0400            BFRSPH *     $007F
0130: 0400            HIMEML *     $0084  END OF RAM POINTER
0140: 0400            HIMEMH *     $0085
0150:
0160:                 6522-IC REGISTERS
0170:
0180: 0400            PCR    *     $180C  PERIPHERAL CONTROL REGISTER
0190: 0400            IFR    *     $180D  INTERRUPT FLAG REGISTER
0200: 0400            IER    *     $180E  INTERRUPT ENABLE REGISTER
0210:
0220:                 6532-IC REGISTERS
0230:
0240: 0400            PAD    *     $1A80  DATA REGISTER OF PORT A
0250: 0400            PBD    *     $1A82  DATA REGISTER OF PORT B
0260: 0400            CNTB   *     $1AF5  CLK8T (INTERRUPT DISABLE)
0270:
0280:                 INPUT VECTOR (BASIC)
0290:
0300: 0400            INVECL *     $2457  INPUT VECTOR
0310: 0400            INVECH *     $2458
0320:
0330:                 TEMPORARY DATA BUFFERS
0340:
0350: 0400            CHSUM  *     $03F4  CHECKSUM
0360: 0400            PRCNTL *     $03F5  PERIOD COUNTER
0370: 0400            PRCNTH *     $03F6
0380: 0400            ZERO   *     $03F7  PERIODTIME
0390: 0400            HLFPTM *     $03F8  HALF PERIODTIME
0400: 0400            SIVL   *     $03F9
0410: 0400            SIVH   *     $03FA
0420:
0430:                 EXTERNAL SUBROUTINE
0440:
0450: 0400            PRCHA  *     $1334  PRINT CHARACTER ROUTINE
0460:
0470:                 :
0480:                 : INITIALISE AND RECEIVE
0490:                 :
0500:
0510: 0400 A9 7F      READ   LDAIM $7F
0520: 0402 8D 0E 18          STA   IER    SET INTERRUPT DISABLE MODE
0530: 0405 A9 00             LDAIM $00
0540: 0407 8D 0C 18          STA   PCR    SET CA1 NEGATIVE EDGE DETECT
0550: 040A 8D F4 03          STA   CHSUM  CLEAR CHECKSUM
0560: 040D A9 73             LDAIM $73
0570: 040F 8D 82 1A          STA   PBD
0580: 0412 A5 7E             LDA   BFRSPL     BEGINNING OF FREE SPACE POINTER
0590: 0414 8D 65 04          STA   STAIND +01 TO INDIRECT ADDRESS STAIND
0600: 0417 A6 7F             LDX   BFRSPH
0610: 0419 E8                INX              256 BYTE FOR EXPANSION
0620: 041A E4 85             CPX   HIMEMH
0630: 041C B0 63             BCS   STOP
0640: 041E 8E 66 04          STX   STAIND +02
0650:
0660: 0421 A9 10      HEADER LDAIM $10
0670: 0423 8D F6 03          STA   PRCNTH SET PERIOD COUNTER
0680: 0426 A9 36             LDAIM $36
0690: 0428 20 4D 05          JSR   NOSYNC DISPLAY NOSYNC CHARACTER
0700: 042B 20 FD 04   HDR    JSR   PERIOD
0710: 042E C9 22             CMPIM $22
0720: 0430 90 EF             BCC   HEADER PERIODTIME << 2400HZ PERIODTIME
0730: 0432 C9 4E             CMPIM $4E
0740: 0434 B0 EB             BCS   HEADER PERIODTIME >> 2400HZ PERIODTIME
0750: 0436 EE F5 03          INC   PRCNTL
0760: 0439 D0 F0             BNE   HDR    NOT 256 PERIODS? THEN BRANCH
0770: 043B A8                TAY
0780: 043C 20 4B 05          JSR   SYNC   DISPLAY SYNC CHARACTER
0790: 043F 98                TYA
0800: 0440 CE F6 03          DEC   PRCNTH
0810: 0443 D0 E6             BNE   HDR    NOT 16*256 PERIODS OF 2400HZ? THEN BRANCH
0820: 0445 0A                ASLA         PERIODTIME X 2 (=1200HZ PERIODTIME )
0830: 0446 38                SEC
0840: 0447 E9 0C             SBCIM $0C    1200HZ PERIODTIME - 100US = ZERO
0850: 0449 8D F7 03          STA   ZERO
0860: 044C 20 4B 05   STABIT JSR   SYNC
0870: 044F 20 00 05          JSR   HLFPER FIND STARTBIT
0880: 0452 CD F7 03          CMP   ZERO
0890: 0455 90 F5             BCC   STABIT NO 1200HZ PERIOD? THEN BRANCH
0900: 0457 B0 08             BCS   READBY BRANCH ALWAYS
0910: 0459 20 00 05   STRTBT JSR   HLFPER FIND STARTBIT
0920: 045C CD F7 03          CMP   ZERO
0930: 045F 90 F8             BCC   STRTBT
0940: 0461 20 24 05   READBY JSR   RBYT   READ ONE BYTE
0950:
0960: 0464 8D FF FF   STAIND STA   $FFFF  CHARACTER TO TABLE
0970: 0467 C9 03             CMPIM $03
0980: 0469 F0 21             BEQ   EOT    END OF TEXT CHARACTER? THEN BRANCH
0990: 046B EE 65 04          INC   STAIND +01 INCREMENT POINTER
1000: 046E D0 03             BNE   EORAM
1010: 0470 EE 66 04          INC   STAIND +02
1020: 0473 A5 85      EORAM  LDA   HIMEMH
1030: 0475 CD 66 04          CMP   STAIND +02
1040: 0478 D0 DF             BNE   STRTBT NOT END OF RAM? THEN BRANCH
1050: 047A A5 84             LDA   HIMEML
1060: 047C CD 65 04          CMP   STAIND +01
1070: 047F D0 D8             BNE   STRTBT
1080: 0481 A0 00      STOP   LDYIM $00
1090: 0483 20 59 05          JSR   MESSY  "OUT OF MEMORY" MESSAGE
1100: 0486 A9 67             LDAIM $67
1110: 0488 8D 82 1A          STA   PBD
1120: 048B 60                RTS          RETURN TO BASIC
1130:
1140: 048C 20 00 05   EOT    JSR   HLFPER
1150: 048F CD F7 03          CMP   ZERO
1160: 0492 90 F8             BCC   EOT
1170: 0494 20 24 05          JSR   RBYT   READ CHECKSUM
1180: 0497 A9 67             LDAIM $67
1190: 0499 8D 82 1A          STA   PBD
1200: 049C A5 7E             LDA   BFRSPL TABLE STARTADDRESS TO LDIND
1210: 049E 8D BF 04          STA   LDIND  +01
1220: 04A1 A6 7F             LDX   BFRSPH
1230: 04A3 E8                INX
1240: 04A4 8E C0 04          STX   LDIND  +02
1250: 04A7 AD 57 24          LDA   INVECL
1260: 04AA AC 58 24          LDY   INVECH
1270: 04AD 8D F9 03          STA   SIVL   SAVE INPUT VECTOR
1280: 04B0 8C FA 03          STY   SIVH
1290: 04B3 A9 BE             LDAIM LDIND
1300: 04B5 A0 04             LDYIM LDIND  /256
1310: 04B7 8D 57 24          STA   INVECL NEW INPUT VECTOR
1320: 04BA 8C 58 24          STY   INVECH
1330: 04BD 60                RTS
1340:
1350:                 :
1360:                 : THIS ROUTINE IS CALLED BY BASIC AND TRANSFERS
1370:                 : EVERY CHARACTER FROM THE TABLE TO BASIC
1380:                 : IT ALSO GIVES A LISTING OF THE PROGRAM
1390:                 :
1400:
1410: 04BE AD FF FF   LDIND  LDA   $FFFF  CHARACTER FROM TABLE
1420: 04C1 A8                TAY
1430: 04C2 20 34 13          JSR   PRCHA
1440: 04C5 EE BF 04          INC   LDIND  +01 INCREMENT POINTER
1450: 04C8 D0 03             BNE   INCPNT
1460: 04CA EE C0 04          INC   LDIND  +02
```

```
1470: 04CD AD 66 04   INCPNT LDA   STAIND +02
1480: 04D0 CD C0 04          CMP   LDIND  +02
1490: 04D3 D0 26             BNE   RETBAS NOT END OF TABLE? THEN BRANCH
1500: 04D5 AD 65 04          LDA   STAIND +01
1510: 04D8 CD BF 04          CMP   LDIND  +01
1520: 04DB D0 1E             BNE   RETBAS
1530: 04DD AD F9 03          LDA   SIVL
1540: 04E0 8D 57 24          STA   INVECL RESET INPUT VECTOR
1550: 04E3 AD FA 03          LDA   SIVH
1560: 04E6 8D 58 24          STA   INVECH
1570: 04E9 AD F4 03          LDA   CHSUM
1580: 04EC F0 05             BEQ   RETURN NO CHECKSUM ERROR? THEN BRANCH
1590: 04EE A0 12             LDYIM $12
1600: 04F0 20 59 05          JSR   MESSY  "CHECKSUM ERROR" MESSAGE
1610: 04F3 A0 25      RETURN LDYIM $25
1620: 04F5 20 59 05          JSR   MESSY  "OK" MESSAGE
1630: 04F8 A9 0D             LDAIM $0D    CARRIAGE RETURN
1640: 04FA 60                RTS
1650: 04FB 98         RETBAS TYA
1660: 04FC 60                RTS
1670:
1680:                 :
1690:                 : MEASURE ONE PERIODTIME
1700:                 :
1710:
1720: 04FD 20 00 05   PERIOD JSR   HLFPER
1730: 0500 A9 02      HLFPER LDAIM $02
1740: 0502 2C 0D 18   HLF    BIT   IFR
1750: 0505 F0 FB             BEQ   HLF    NO ACTIVE EDGE ON CA1-INPUT? THEN BRANCH
1760: 0507 8D 0D 18          STA   IFR    CLEAR CA1 FLAG
1770: 050A A9 01             LDAIM $01
1780: 050C 4D 0C 18          EOR   PCR
1790: 050F 8D 0C 18          STA   PCR    OPPOSITE ACTIVE CA1 EDGE DETECT
1800: 0512 A9 FF             LDAIM $FF
1810: 0514 AA                TAX
1820: 0515 4D F6 1A          EOR   $1AF6  GET ELAPSED TIME IN ACCU
1830: 0518 8E F5 1A          STX   CNTB   RESET TIMER
1840: 051B AA                TAX
1850: 051C 18                CLC
1860: 051D 6D F8 03          ADC   HLFPTM FULL PERIODTIME IN ACCU
1870: 0520 8E F8 03          STX   HLFPTM SAVE LAST HALF PERIODTIME
1880: 0523 60                RTS
1890:
1900:                 :
1910:                 : READ ONE BYTE
1920:                 :
1930:
1940: 0524 A9 55      RBYT   LDAIM $55
1950: 0526 20 4D 05          JSR   NOSYNC
1960: 0529 A0 08             LDYIM $08    SET BITCOUNTER
1970: 052B 48         RB     PHA          SAVE ACCU
1980: 052C 20 FD 04          JSR   PERIOD
1990: 052F CD F7 03          CMP   ZERO
2000: 0532 B0 06             BCS   FNDZRO 1200HZ PERIOD? THEN BRANCH
2010: 0534 20 FD 04          JSR   PERIOD SECOND 2400HZ PERIOD
2020: 0537 38                SEC
2030: 0538 B0 01             BCS   SHIFT  BRANCH ALWAYS
2040: 053A 18         FNDZRO CLC
2050: 053B 68         SHIFT  PLA
2060: 053C 6A                RORA         NEXT BIT TO ACCU
2070: 053D 88                DEY
2080: 053E D0 EB             BNE   RB     NOT 8 BITS? THEN BRANCH
2090: 0540 48                PHA
2100: 0541 4D F4 03          EOR   CHSUM
2110: 0544 8D F4 03          STA   CHSUM  UPDATE CHECKSUM
2120: 0547 68                PLA
2130: 0548 29 7F             ANDIM $7F    CLEAR BIT 7
2140: 054A 60                RTS
2150:
2160:                 :
2170:                 : DISPLAY SYNC-CHARACTER ON 7-SEGM.DISPLAY
2180:                 :
2190:
2200: 054B A9 69      SYNC   LDAIM $69
2210: 054D 8D 80 1A   NOSYNC STA   PAD
2220: 0550 AD 82 1A          LDA   PBD
2230: 0553 49 02             EORIM $02
2240: 0555 8D 82 1A          STA   PBD
2250: 0558 60                RTS
2260:
2270:                 :
2280:                 : OUTPUT MESSAGE (MESS+Y)
2290:                 :
2300:
2310: 0559 B9 68 05   MESSY  LDAY  MESS   LOAD CHARACTER
2320: 055C C9 03             CMPIM $03
2330: 055E F0 07             BEQ   MESEND END OF TEXT CHARACTER?
2340: 0560 20 34 13          JSR   PRCHA
2350: 0563 C8                INY
2360: 0564 4C 59 05          JMP   MESSY
2370: 0567 60         MESEND RTS
2380:
2390: 0568 0D         MESS   =     $0D
2400: 0569 0A                =     $0A
2410: 056A 4F                =     'O
2420: 056B 55                =     'U
2430: 056C 54                =     'T
2440: 056D 20                =     '
2450: 056E 4F                =     'O
2460: 056F 46                =     'F
2470: 0570 20                =     '
2480: 0571 4D                =     'M
2490: 0572 45                =     'E
2500: 0573 4D                =     'M
2510: 0574 4F                =     'O
2520: 0575 52                =     'R
2530: 0576 59                =     'Y
2540: 0577 0A                =     $0A
2550: 0578 0D                =     $0D
2560: 0579 03                =     $03
2570: 057A 0D                =     $0D
2580: 057B 0A                =     $0A
2590: 057C 43                =     'C
2600: 057D 48                =     'H
2610: 057E 45                =     'E
2620: 057F 43                =     'C
2630: 0580 4B                =     'K
2640: 0581 53                =     'S
2650: 0582 55                =     'U
2660: 0583 4D                =     'M
2670: 0584 20                =     '
2680: 0585 45                =     'E
2690: 0586 52                =     'R
2700: 0587 52                =     'R
2710: 0588 4F                =     'O
2720: 0589 52                =     'R
2730: 058A 0A                =     $0A
2740: 058B 0D                =     $0D
2750: 058C 03                =     $03
2760: 058D 0D                =     $0D
2770: 058E 0A                =     $0A
2780: 058F 4F                =     'O
2790: 0590 4B                =     'K
2800: 0591 0A                =     $0A
2810: 0592 0D                =     $0D
2820: 0593 03                =     $03
2830: 0594 0D                =     $0D
2840: 0595 0A                =     $0A
2850: 0596 4E                =     'N
2860: 0597 45                =     'E
2870: 0598 57                =     'W
2880: 0599 0A                =     $0A
2890: 059A 0D                =     $0D
2900: 059B 03                =     $03
```

Table 1. This is the source listing for the complete translation program. This particular example is for the expanded Junior with KB-9-BASIC. The listing for the DOS Junior is almost identical to the one shown here, with the exception of a few address locations.

Basicode-2 interface for the Junior Computer elektor october 1983

Table 2

```
M
HEXDUMP: 0200,0363
       0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
0200: AD 52 2A AC 53 2A 8D F0 03 8C F1 03 A9 42 A0 02
0210: 8D 52 2A 8C 53 2A A5 04 A4 05 8D F2 03 8C F3 03
0220: A9 77 A0 02 85 04 84 05 A5 7E A4 7F 8D 49 02 8C
0230: 4A 02 8D A0 02 8C A1 02 A9 00 8D F4 03 8D FB 03
0240: A9 02 C9 0A F0 30 09 80 8D FF FF 4D F4 03 8D F4
0250: 03 EE 49 02 D0 03 EE 4A 02 A5 85 CD 4A 02 D0 16
0260: A5 84 CD 49 02 D0 0F A9 FF 8D FB 03 A5 78 8D 49
0270: 02 A5 79 8D 4A 02 60 48 98 48 A9 0D 20 42 02 A9
0280: 03 20 42 02 AD F4 03 20 48 02 A9 7F 8D 0E 18 A9
0290: C0 8D 0B 18 A9 01 8D 05 18 20 2B 03 20 39 03 AD
02A0: FF FF A0 08 4A 48 B0 05 20 39 03 70 03 20 46 03
02B0: 68 88 D0 F0 20 46 03 20 46 03 EE A0 02 D0 03 EE
02C0: A1 02 AC A1 02 AE A0 02 C4 85 D0 0E E4 84 D0 0A
02D0: A5 78 8D A0 02 A5 79 8D A1 02 CC 4A 02 D0 BD EC
02E0: 49 02 D0 B8 20 2B 03 8C 0B 18 AD F2 03 AC F3 03
02F0: 85 04 84 05 AD F0 03 AC F1 03 8D 52 2A 8C 53 2A
0300: AD FB 03 F0 20 A5 79 85 7B 85 7D 85 7F A4 78 C8
0310: C8 C8 84 7A 84 7C 84 7E A9 00 A8 91 78 C8 91 78
0320: A0 2C 20 59 05 68 A8 68 4C 03 00 A2 70 A0 17 20
0330: 46 03 CA D0 FA 88 D0 F7 60 A9 9F 8D 06 18 A9 01
0340: 8D 07 18 4C 53 03 20 49 03 A9 CE 8D 06 18 A9 00
0350: 8D 07 18 AD 04 18 2C 0D 18 50 FB AD 04 18 2C 0D
0360: 18 50 FB 60

JUNIOR
```

```
M
HEXDUMP: 0400,059B
       0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
0400: A9 7F 8D 0E 18 A9 00 8D 0C 18 8D F4 03 A9 73 8D
0410: 82 1A A5 7E 8D 65 04 A6 7F E8 E4 85 B0 63 8E 66
0420: 04 A9 10 8D F6 03 A9 36 20 4D 05 20 FD 04 C9 22
0430: 90 EF C9 4E B0 EB EE F5 03 D0 F0 A8 20 4B 05 98
0440: CE F6 03 D0 E6 0A 38 E9 0C 8D F7 03 20 4B 05 20
0450: 00 05 CD F7 03 90 F5 B0 08 20 00 05 CD F7 03 90
0460: F8 20 24 05 8D FF FF C9 03 F0 21 EE 65 04 D0 03
0470: EE 66 04 A5 85 CD 66 04 D0 DF A5 84 CD 65 04 D0
0480: D8 A0 00 20 59 05 A9 67 8D 82 1A 60 20 00 05 CD
0490: F7 03 90 F8 20 24 05 A9 67 8D 82 1A A5 7E 8D BF
04A0: 04 A6 7F E8 8E C0 04 AD 57 24 AC 58 24 8D F9 03
04B0: 8C FA 03 A9 BE A0 04 8D 57 24 8C 58 24 60 AD FF
04C0: FF A8 20 34 13 EE BF 04 D0 03 EE C0 04 AD 66 04
04D0: CD C0 04 D0 26 AD 65 04 CD BF 04 D0 1E AD F9 03
04E0: 8D 57 24 AD FA 03 8D 58 24 AD F4 03 F0 05 A0 12
04F0: 20 59 05 A0 25 20 59 05 A9 0D 60 98 60 20 00 05
0500: A9 02 2C 0D 18 F0 FB 8D 0D 18 A9 01 4D 0C 18 8D
0510: 0C 18 A9 FF AA 4D F6 1A 8E F5 1A AA 18 6D F8 03
0520: 8E F8 03 60 A9 55 20 4D 05 A0 08 48 20 FD 04 CD
0530: F7 03 B0 06 20 FD 04 38 B0 01 18 68 6A 88 D0 EB
0540: 48 4D F4 03 8D F4 03 68 29 7F 60 A9 69 8D 80 1A
0550: AD 82 1A 49 02 8D 82 1A 60 B9 68 05 C9 03 F0 07
0560: 20 34 13 C8 4C 59 05 60 0D 0A 4F 55 54 20 4F 46
0570: 20 4D 45 4D 4F 52 59 0A 0D 03 0D 0A 43 48 45 43
0580: 4B 53 55 4D 20 45 52 52 4F 52 0A 0D 03 0D 0A 4F
0590: 4B 0A 0D 03 0D 0A 4E 45 57 0A 0D 03

JUNIOR
```

Table 2. The hexdump for the translation program for the Junior with KB-9-BASIC.

Table 3

```
HEXDUMP: E000,E163
       0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
E000: AD 11 23 AC 12 23 8D F0 E1 8C F1 E1 A9 41 A0 E0
E010: 8D 11 23 8C 12 23 A5 04 A4 05 8D F2 E1 8C F3 E1
E020: A9 77 A0 E0 85 04 84 05 A5 7E A4 7F 8D 49 E0 8C
E030: 4A E0 8D A0 E0 8C A1 E0 A9 00 8D F4 E1 8D FB E1
E040: A9 02 C9 0A F0 30 09 80 8D FF FF 4D F4 E1 8D F4
E050: E1 EE 49 E0 D0 03 EE 4A E0 A5 85 CD 4A E0 D0 16
E060: A5 84 CD 49 E0 D0 0F A9 FF 8D FB E1 A5 78 8D 49
E070: E0 A5 79 8D 4A E0 60 48 98 48 A9 0D 20 42 E0 A9
E080: 03 20 42 E0 AD F4 E1 20 48 E0 A9 7F 8D 0E F8 A9
E090: C0 8D 0B F8 A9 01 8D 05 F8 20 2B E1 20 39 E1 AD
E0A0: FF FF A0 08 4A 48 B0 05 20 39 E1 70 03 20 46 E1
E0B0: 68 88 D0 F0 20 46 E1 20 46 E1 EE A0 E0 D0 03 EE
E0C0: A1 E0 AC A1 E0 AE A0 E0 C4 85 D0 0E E4 84 D0 0A
E0D0: A5 78 8D A0 E0 A5 79 8D A1 E0 CC 4A E0 D0 BD EC
E0E0: 49 E0 D0 B8 20 2B E1 8C 0B F8 AD F2 E1 AC F3 E1
E0F0: 85 04 84 05 AD F0 E1 AC F1 E1 8D 11 23 8C 12 23
E100: AD FB E1 F0 20 A5 79 85 7B 85 7D 85 7F A4 78 C8
E110: C8 C8 84 7A 84 7C 84 7E A9 00 A8 91 78 C8 91 78
E120: A0 2C 20 59 E3 68 A8 68 4C 03 00 A2 70 A0 17 20
E130: 46 E1 CA D0 FA 88 D0 F7 60 A9 9F 8D 06 F8 A9 01
E140: 8D 07 F8 4C 53 E1 20 49 E1 A9 CE 8D 06 F8 A9 00
E150: 8D 07 F8 AD 04 F8 2C 0D F8 50 FB AD 04 F8 2C 0D
E160: F8 50 FB 60

JUNIOR
```

```
M
HEXDUMP: E200,E39B
       0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
E200: A9 7F 8D 0E F8 A9 00 8D 0C F8 8D F4 E1 A9 73 8D
E210: 82 FA A5 7E 8D 65 E2 A6 7F E8 E4 85 B0 63 8E 66
E220: E2 A9 10 8D F6 E1 A9 36 20 4D E3 20 FD E2 C9 22
E230: 90 EF C9 4E B0 EB EE F5 E1 D0 F0 A8 20 4B E3 98
E240: CE F6 E1 D0 E6 0A 38 E9 0C 8D F7 E1 20 4B E3 20
E250: 00 E3 CD F7 E1 90 F5 B0 08 20 00 E3 CD F7 E1 90
E260: F8 20 24 E3 8D FF FF C9 03 F0 21 EE 65 E2 D0 03
E270: EE 66 E2 A5 85 CD 66 E2 D0 DF A5 84 CD 65 E2 D0
E280: D8 A0 00 20 59 E3 A9 67 8D 82 FA 60 20 00 E3 CD
E290: F7 E1 90 F8 20 24 E3 A9 67 8D 82 FA A5 7E 8D BF
E2A0: E2 A6 7F E8 8E C0 E2 AD 01 23 AC 02 23 8D F9 E1
E2B0: 8C FA E1 A9 BD A0 E2 8D 01 23 8C 02 23 60 AD FF
E2C0: FF A8 8D 63 23 EE BF E2 D0 03 EE C0 E2 AD 66 E2
E2D0: CD C0 E2 D0 26 AD 65 E2 CD BF E2 D0 1E AD F9 E1
E2E0: 8D 01 23 AD FA E1 8D 02 23 AD F4 E1 F0 05 A0 12
E2F0: 20 59 E3 A0 25 20 59 E3 A9 0D 60 98 60 20 00 E3
E300: A9 02 2C 0D F8 F0 FB 8D 0D F8 A9 01 4D 0C F8 8D
E310: 0C F8 A9 FF AA 4D F6 FA 8E F5 FA AA 18 6D F8 E1
E320: 8E F8 E1 60 A9 55 20 4D E3 A0 08 48 20 FD E2 CD
E330: F7 E1 B0 06 20 FD E2 38 B0 01 18 68 6A 88 D0 EB
E340: 48 4D F4 E1 8D F4 E1 68 29 7F 60 A9 69 8D 80 FA
E350: AD 82 FA 49 02 8D 82 FA 60 B9 68 E3 C9 03 F0 07
E360: 20 43 23 C8 4C 59 E3 60 0D 0A 4F 55 54 20 4F 46
E370: 20 4D 45 4D 4F 52 59 0A 0D 03 0D 0A 43 48 45 43
E380: 4B 53 55 4D 20 45 52 52 4F 52 0A 0D 03 0D 0A 4F
E390: 4B 0A 0D 03 0D 0A 4E 45 57 0A 0D 03

JUNIOR
```

Table 3. Hexdump for the translation program for the DOS Junior.

POKE 8256,Ø : POKE 8257,2 : X = USR(X) : LIST
(POKE 574,Ø : POKE 575,224 : X = USR(X) : LIST)
The recorder is then set to record and started. Only then is the (carriage) return given. The whole program is then saved on the tape in Basicode form. After the computer gives the 'OK' signal the recorder can be stopped. It is also possible to save only a part of the program on tape (for example, lines 1000-1090):
POKE 8256,Ø : POKE 8257,2 : X = USR(X) : LIST 1000-1090
(POKE 574,Ø : POKE 575,224 : X = USR(X) : LIST 1000-1090).
Before the BASIC program is stored on tape, the computer 'translates' the program first into 'LIST' format and places that in a table which appears above the BASIC program in the RAM range. With large programs, the RAM range may not be big enough to store both of these so after the program is stored on tape the computer returns the 'NEW' message. This means that the original BASIC program is erased from the memory. As it is in Basicode form on the tape anyway, it can also be read in again.

Details of the translation program

This next section is a description of the write and read routines (more details are given in the listing of table 1).

The write program

When this routine is called by means of X = USR(X), the OUTPUT vector (of the BASIC Junior) is changed for the start address of a machine code routine (TABLE in the write program). This routine stores an ASCII character from ACCU into RAM. After giving a LIST command (with POKE . . . : POKE . . . : X = USR(X) : LIST), the computer will list the program on the screen (or on the printer). Because the OUTPUT vector is changed (it normally points to the 'print character' routine), the TABLE routine is used to store the listing in RAM *above* the original BASIC program. The program is then stored in this table in LIST format.
After the BASIC Junior notes the end of the program and is therefore finished listing, it jumps via the JMP command at addresses ØØØ3 . . . ØØØ5 to SVECAS. This routine sets the whole table onto cassette with 1200 and 2400 Hz tones. When that is done the OUTPUT vector and the JMP at address ØØØ3 are reset and the computer returns to BASIC.

The read program

After this program is called by X = USR(X), the Basicode program is read from cassette and stored in the form of a table in RAM. Again the program is in LIST format. When the 'end of text' character and the checksum are read in, the whole program is located in this table, the INPUT vector (in the BASIC Junior) is changed for the start address of the LDIND routine, and the computer returns to normal BASIC.

The computer should now really wait for an input from the terminal (the INPUT vector normally points to the receive character routine), but because the INPUT vector points to the LDIND routine the characters are called one by one from the table by the BASIC Junior (and printed at the same time). This makes it seem as if a program is being typed in at high speed. The program thus read out of the table is then processed and stored in the normal way.

Finally, the INPUT vector is reset and the computer returns with 'OK'. The user can then work with the program as usual.

BASIC subroutines

Apart from the translation program there is also a need for some subroutines, written in Basicode-2 protocol. These are dealt with in depth in the descriptive article, 'Basicode-2', in this issue.

Three of these subroutines are not usable with the Junior/Elekterminal combination. These are routines 120, 200 and 250. Subroutine 120 relates to the position of the cursor on the screen and subroutine 200 checks whether at a specific moment a key is pressed. Neither is possible because of the arrangement of the Elekterminal. Subroutine 250 just gives a bleep, but the Elekterminal is mute.

If the main BASIC program calls subroutine 120 or 250 nothing happens because in the Junior these subroutines consist of the 'RETURN' command. For subroutine 200 IN$ is an empty string so that it seems as if no key is pressed at that moment.

The standard subroutines for the expanded Junior and the DOS Junior, both with the Elekterminal, are given in tables 4 and 5 respectively. Subroutines 350 and 360 should really refer to a printer but in our case they refer to the terminal.

The subroutines can be read in either before or after the Basicode program. That makes no difference as long as they are present when the program is RUN. If, for example, the Basicode program has already been read in, the subroutines can simply be added by reading them in using POKE . . . : POKE . . . : X = USR(X).

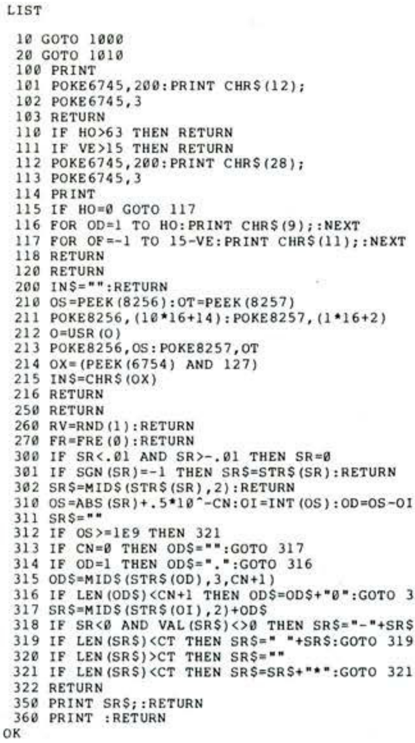
Two program sections can be added to form one program by reading them both in separately. The only prerequisite is that the two parts have no identical line numbers.

Practical points

After reading in a Basicode programme it is only common sense to check it through carefully. Often there are some details that have a different meaning on your computer to what they meant to the computer on which the program was developed. This is a common reason for programs not to work.

Consider this case, for example: we have a Basicode program that draws a maze, and it contains the necessary PRINT statements. If part of the maze is now drawn on the screen and the program wants to PRINT something in the middle of the maze, a carriage return and line feed are automatically generated after the print statement. With the Elekterminal a carriage return

Table 4

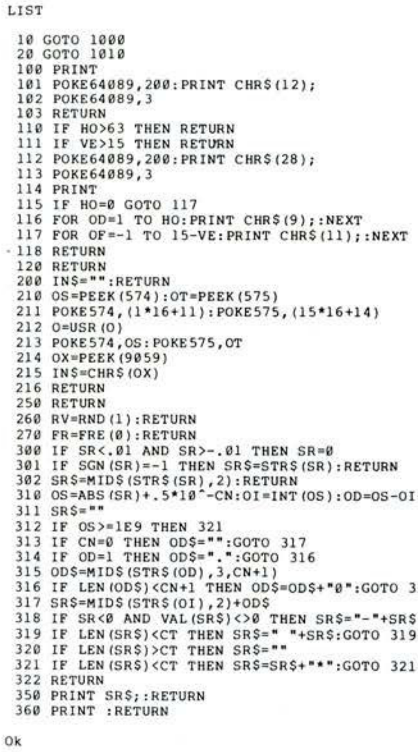


```
LIST

 10 GOTO 1000
 20 GOTO 1010
 100 PRINT
 101 POKE6745,200:PRINT CHR$(12);
 102 POKE6745,3
 103 RETURN
 110 IF HO>63 THEN RETURN
 111 IF VE>15 THEN RETURN
 112 POKE6745,200:PRINT CHR$(28);
 113 POKE6745,3
 114 PRINT
 115 IF HO=0 GOTO 117
 116 FOR OD=1 TO HO:PRINT CHR$(9);:NEXT
 117 FOR OF=-1 TO 15-VE:PRINT CHR$(11);:NEXT
 118 RETURN
 120 RETURN
 200 IN$="":RETURN
 210 OS=PEEK(8256):OT=PEEK(8257)
 211 POKE8256,(10*16+14):POKE8257,(1*16+2)
 212 O=USR(O)
 213 POKE8256,OS:POKE8257,OT
 214 OX=(PEEK(6754) AND 127)
 215 IN$=CHR$(OX)
 216 RETURN
 250 RETURN
 260 RV=RND(1):RETURN
 270 FR=FRE(0):RETURN
 300 IF SR<.01 AND SR>-.01 THEN SR=0
 301 IF SGN(SR)=-1 THEN SR$=STR$(SR):RETURN
 302 SR$=MID$(STR$(SR),2):RETURN
 310 OS=ABS(SR)+.5*10^-CN:OI=INT(OS):OD=OS-OI+1
 311 SR$=""
 312 IF OS>=1E9 THEN 321
 313 IF CN=0 THEN OD$="":GOTO 317
 314 IF OD=1 THEN OD$=".":GOTO 316
 315 OD$=MID$(STR$(OD),3,CN+1)
 316 IF LEN(OD$)<CN+1 THEN OD$=OD$+"0":GOTO 316
 317 SR$=MID$(STR$(OI),2)+OD$
 318 IF SR<0 AND VAL(SR$)<>0 THEN SR$="-"+SR$
 319 IF LEN(SR$)<CT THEN SR$=" "+SR$:GOTO 319
 320 IF LEN(SR$)>CT THEN SR$=""
 321 IF LEN(SR$)<CT THEN SR$=SR$+"*":GOTO 321
 322 RETURN
 350 PRINT SR$;:RETURN
 360 PRINT :RETURN
OK
```

Table 4. The standard subroutines for the expanded Junior with KB-9-BASIC.

Table 5



```
LIST

 10 GOTO 1000
 20 GOTO 1010
 100 PRINT
 101 POKE64089,200:PRINT CHR$(12);
 102 POKE64089,3
 103 RETURN
 110 IF HO>63 THEN RETURN
 111 IF VE>15 THEN RETURN
 112 POKE64089,200:PRINT CHR$(28);
 113 POKE64089,3
 114 PRINT
 115 IF HO=0 GOTO 117
 116 FOR OD=1 TO HO:PRINT CHR$(9);:NEXT
 117 FOR OF=-1 TO 15-VE:PRINT CHR$(11);:NEXT
 118 RETURN
 120 RETURN
 200 IN$="":RETURN
 210 OS=PEEK(574):OT=PEEK(575)
 211 POKE574,(1*16+11):POKE575,(15*16+14)
 212 O=USR(O)
 213 POKE574,OS:POKE575,OT
 214 OX=PEEK(9059)
 215 IN$=CHR$(OX)
 216 RETURN
 250 RETURN
 260 RV=RND(1):RETURN
 270 FR=FRE(0):RETURN
 300 IF SR<.01 AND SR>-.01 THEN SR=0
 301 IF SGN(SR)=-1 THEN SR$=STR$(SR):RETURN
 302 SR$=MID$(STR$(SR),2):RETURN
 310 OS=ABS(SR)+.5*10^-CN:OI=INT(OS):OD=OS-OI+1
 311 SR$=""
 312 IF OS>=1E9 THEN 321
 313 IF CN=0 THEN OD$="":GOTO 317
 314 IF OD=1 THEN OD$=".":GOTO 316
 315 OD$=MID$(STR$(OD),3,CN+1)
 316 IF LEN(OD$)<CN+1 THEN OD$=OD$+"0":GOTO 316
 317 SR$=MID$(STR$(OI),2)+OD$
 318 IF SR<0 AND VAL(SR$)<>0 THEN SR$="-"+SR$
 319 IF LEN(SR$)<CT THEN SR$=" "+SR$:GOTO 319
 320 IF LEN(SR$)>CT THEN SR$=""
 321 IF LEN(SR$)<CT THEN SR$=SR$+"*":GOTO 321
 322 RETURN
 350 PRINT SR$;:RETURN
 360 PRINT :RETURN

Ok
```

Table 5. The standard subroutines for the DOS Junior.

1

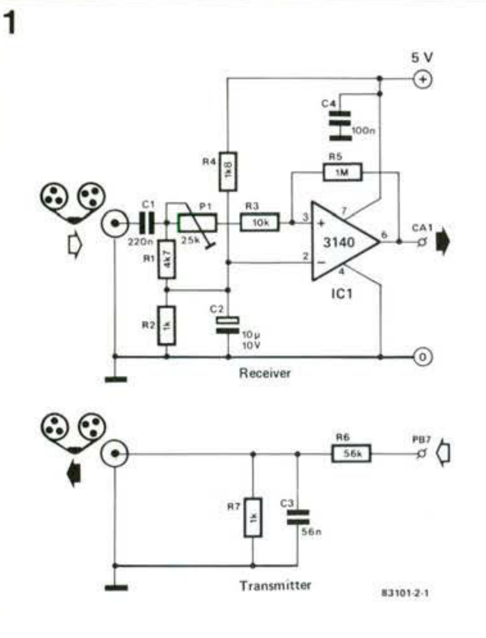


Figure 1. The circuit diagram for the interface circuit that must be connected between a cassette recorder and the Junior Computer.

Parts list

Resistors:
R1 = 4k7
R2,R4,R7 = 1 k
R3 = 10 k
R5 = 1 M
R6 = 56 k
P1 = 25 k preset

Capacitors:
C1 = 220 n
C2 = 10 µ/10 V
C3 = 56 n
C4 = 100 n

Semiconductors:
IC1 = 3140

2

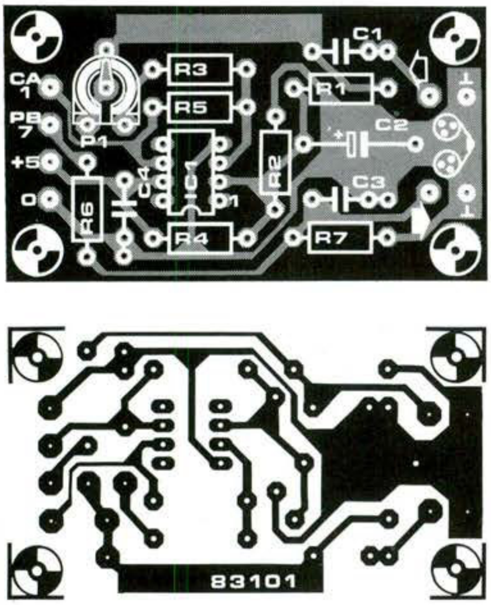
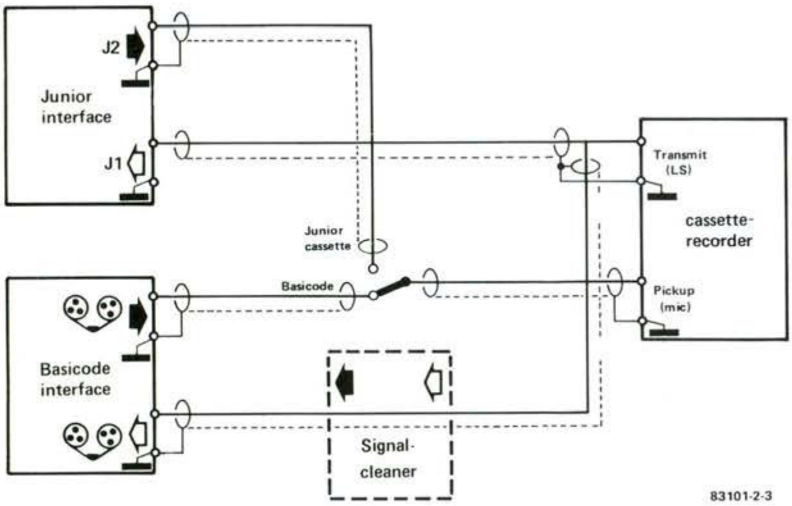


Figure 2. The printed circuit board layout for the interface circuit.

Figure 3. This shows the wiring layout that must be used if both the normal Junior cassette interface and the Basicode interface are to be used together.

3



means that everything after the print statement on this line is erased. In this example the program can easily be adapted by following the PRINTs in question with a ;. The CR and LF are not produced then and the program runs properly.

A program could, of course, also call a subroutine that the Junior/Elekterminal does not recognize (120, 200 and 250). Subroutines 200 and 250 are no real problem and can easily be avoided, but is is sometimes more difficult to do without routine 120. If sub 120 is used, for example, in a game to define the position of the cursor on the screen, it can be very difficult to adapt the program. Subroutine 120 is also quite often used to define the screen size. This can also be done by leaving out the appropriate lines and stating on the free lines how large the screen is (16 lines of 64 characters on the Elekterminal). In the case where, for example, the screen format is defined for a section of a program, and after leaving this section, variables VV and HH must contain the height and width of the screen. In our case this program section is simply changed by VV = 15 : HH = 63 (remember that the first position has always number zero).

A final note about the @ sign in KB-9-BASIC. If the computer sees this sign the whole line is erased and CR and LF are given.

The hardware

The hardware for the Basicode interface consists of a small adapter circuit which is connected between the cassette recorder and the Junior Computer. The circuit diagram is shown in figure 1. It consists of a transmitter and a receiver section. The receiver contains only one IC (3140) which is connected as a schmitt trigger/level adapter. Using P1, the trigger level can be set between certain limits, but normally the circuit works correctly if the pot is roughly in mid position. The transmitter section simply reduces the output signal from the Junior and filters out the higher harmonics from the signal.

The printed circuit board for the interface (figure 2) is designed so that two phono plugs (for input and output) can be soldered directly onto the board using some wire links. Points CA1 and PB7 are connected to the corresponding points on the VIA connector on the interface board.

If the normal Junior cassette interface and the Basicode interface are to be connected at the same time (the former is needed to read in machine code programs), care must be taken when wiring the interfaces. The wiring diagram for connecting both interfaces is given in figure 3. Any deviation from this layout is likely to result in earth losses occurring and the possibility of oscillation is greater. This same diagram also shows a block called signal cleaner! This circuit, which is also described in this issue, is only needed if the signal from the recorder (or radio) is of very poor quality. It is easy to try without this interface first and if this does not work, the circuit could always be added. ◀