```
EC83 0C                   DATA   :0C
EC84 FA89EC               JM     :EC89      Ready if already negative
EC87 E7                   RST    4          Else change sign MACC
EC88 1B                   DATA   :1B        (make MACC  +1)
EC89 C9           L0E257  RET
                  *
                  **********************
                  * TEST A FPT VARIABLE *
                  **********************
                  *
                  * Entry: Variable in MACC.
                  * Exit:  Z=1: Variable is zero.
                  *        Z=0: Other flags set on exponent byte
                  *              of variable.
                  *        ABCDEHL preserved.
                  *
EC8A C5           FTEST   PUSH   B
EC8B D5                   PUSH   D
EC8C F5                   PUSH   PSW
EC8D E7                   RST    4          Copy MACC to reg A,B,C,D
EC8E 15                   DATA   :15
EC8F 5F                   MOV    E,A        Exp byte in E
EC90 B0                   ORA    B          )
EC91 B1                   ORA    C          ) Check if nr is zero
EC92 B2                   ORA    D          )
EC93 CA98EC               JZ     :EC98      Then quit
EC96 7B                   MOV    A,E        Get exp byte
EC97 B7                   ORA    A          Set flags on it
EC98 D1           FTS10   POP    D
EC99 7A                   MOV    A,D
EC9A D1                   POP    D
EC9B C1                   POP    B
EC9C C9                   RET
                  *
                  **************************
                  * RUN basicfunction SCRN *
                  **************************
                  *
EC9D CDF3E5       RSCRN   CALL   :E5F3      Eval given coord
ECA0 C5                   PUSH   B
ECA1 4F                   MOV    C,A        Y-coord in C
ECA2 EF                   RST    5          Ask colour of dot on screen
ECA3 27                   DATA   :27        + size graphics screen
ECA4 C1                   POP    B
ECA5 DA02E6               JC     :E602      Evt run screen error
ECA8 C37CEB               JMP    :EB7C      Contents screen loc in MACC
                  *
                  *
                  *  ==============
                  *** LIST HANDLER ***
                  *  ==============
                  *
                  * This module lists a program from the textbuffer
                  * onto the screen (or into other required direction)
                  *
                  **********************
                  * LIST A PROGRAM LINE *
                  **********************
                  *
                  * Entry: BC: Points to start of textline.
                  * Exit:  BC: Points to start of next line.
                  *        DEHL preserved, AF corrupted.
```

```
                   *
ECAB D5            SLINE   PUSH  D
ECAC E5                    PUSH  H
ECAD 03                    INX   B           Pnts to line nr
ECAE CDAEEF                CALL  :EFAE       List line nr
ECB1 3E08                  MVI   A,:08
ECB3 CD2ADB                CALL  :DB2A       Cursor to tab 8
ECB6 CDCCEC        L0E262  CALL  :ECCC       List statement
ECB9 0A                    LDAX  B           Get next byte
ECBA B7                    ORA   A
ECBB F2C5EC                JP    :ECC5       If no more statements
ECBE CDF5EF                CALL  :EFF5       Else: print ':'
ECC1 3A                    DATA  :3A
ECC2 C3B6EC                JMP   :ECB6       List next statement
ECC5 CDF5EF        L0E263  CALL  :EFF5       print car.ret
ECC8 0D                    DATA  :0D
ECC9 E1                    POP   H
ECCA D1                    POP   D
ECCB C9                    RET
                   *
                   *******************
                   * LIST A STATEMENT *
                   *******************
                   *
                   * Based on the token in the textbuffer, a particular
                   * statement will be printed.
                   * At first, the Basiccommand will be printed. The
                   * pointers to the particular strings are in a table
                   * starting at CD8B. The base for the table is CC08;
                   * the offset is calculated by TOKEN *3.
                   *
                   * The databyte after the stringaddress pointer
                   * indicates which list-routine has to be used for
                   * the rest of the statement. This byte is a offset
                   * for table ECF8.
                   *
                   * Entry: BC: Points to token.
                   * Exit:  BC: Points to next statement.
                   *        AFDEHL corrupted.
                   *        On stack: Returnaddress from this sub-
                   *        routine.
                   *
ECCC 0A            SCOM    LDAX  B           Get token
ECCD 03                    INX   B           Update pointer
ECCE 5F                    MOV   E,A         token in E
ECCF 1600                  MVI   D,:00
ECD1 2108CC                LXI   H,:CC08     Startaddr stringtable
ECD4 19                    DAD   D           ) Add 3* token
ECD5 19                    DAD   D           )
ECD6 19                    DAD   D           )
ECD7 5E                    MOV   E,M         Get lobyte stringaddr
ECD8 23                    INX   H
ECD9 56                    MOV   D,M         Get hibyte stringaddr
ECDA 23                    INX   H           Point to data after addr
ECDB EB                    XCHG              Stringaddr in HL
ECDC 7E                    MOV   A,M         Get length byte of string
ECDD B7                    ORA   A           Length=0?
ECDE CD32DB                CALL  :DB32       List Basiccmd string
ECE1 1A                    LDAX  D           Get data byte after string-
                                             address
ECE2 CAEAEC                JZ    :ECEA       If length string =0
ECE5 FE00                  CPI   :00
```

```
ECE7 C46BCE                CNZ    :CE6B      Print space if byte after
                                             stringaddr <>0
ECEA 1A           SCM10    LDAX   D          Get data byte (= offset)
ECEB 87                    ADD    A          Offset *2
ECEC 5F                    MOV    E,A        in E
ECED 1600                  MVI    D,:00
ECEF 21F8EC                LXI    H,:ECF8    Startaddr table Listroutines
ECF2 19                    DAD    D          Add offset
ECF3 5E                    MOV    E,M        Get addr in DE
ECF4 23                    INX    H
ECF5 56                    MOV    D,M
ECF6 EB                    XCHG              Addr routine in HL
ECF7 E9                    PCHL              Go to this adress
                  *
                  **********************************
                  * POINTERS LIST HANDLING ROUTINES *
                  **********************************
                  *
                  * Table with addresses of listroutines for the
                  * part of a statement after a token.
                  *
                  * Startaddress table is ECF8. The offset (given
                  * between brackets) is identical to the data
                  * byte after the addresses in the table on CD8B.
                  *
ECF8 3AED         L0E355   DBL    :ED3A      (00)   nothing more
ECFA 41ED                  DBL    :ED41      (01)   linenr
ECFC 3BED                  DBL    :ED3B      (02)   linenr linenr
                                                    (not used)
ECFE 44ED                  DBL    :ED44      (03)   unquoted string
ED00 4DED                  DBL    :ED4D      (04)   E (E=expr)
ED02 47ED                  DBL    :ED47      (05)   E,E
ED04 56ED                  DBL    :ED56      (06)   E E
ED06 62ED                  DBL    :ED62      (07)   E,E E
ED08 5CED                  DBL    :ED5C      (08)   E,E E,E E
ED0A 50ED                  DBL    :ED50      (09)   E E E E
ED0C 68ED                  DBL    :ED68      (0A)   E
ED0E 7AED                  DBL    :ED7A      (0B)   liner-linenr
ED10 84ED                  DBL    :ED84      (0C)   sound
ED12 9BED                  DBL    :ED9B      (0D)   noise
ED14 A4ED                  DBL    :EDA4      (0E)   envelope
ED16 C1ED                  DBL    :EDC1      (0F)   mode
ED18 D9ED                  DBL    :EDD9      (10)   input <string>
ED1A E0ED                  DBL    :EDE0      (11)   input/read/dim
ED1C F0ED                  DBL    :EDF0      (12)   (not used [*])
ED1E FFED                  DBL    :EDFF      (13)   let
ED20 09EE                  DBL    :EE09      (14)   if then <E>
ED22 1AEE                  DBL    :EE1A      (15)   if goto <linenr>
ED24 25EE                  DBL    :EE25      (16)   if then <linenr>
ED26 30EE                  DBL    :EE30      (17)   for to step
ED28 4FEE                  DBL    :EE4F      (18)   next
ED2A 52EE                  DBL    :EE52      (19)   print
ED2C 66EE                  DBL    :EE66      (1A)   on goto
ED2E 71EE                  DBL    :EE71      (1B)   on gosub
ED30 87EE                  DBL    :EE87      (1C)   callm
ED32 94EE                  DBL    :EE94      (1D)   (not used [*])
ED34 9ED8                  DBL    :D89E      (1E)   savea/loada
                  *
                  * The vectors marked with [*] are no pointers to
                  * LIST routines.
                  *
                           DATA   :FF
```

```
ED37 FF                DATA   :FF
ED38 FF                DATA   :FF
ED39 FF                DATA   :FF
              *
              ********************************
              * LIST NO FURTHER EXPRESSIONS *
              ********************************
              *
ED3A C9       SCN1     RET
              *
              ***************************
              * LIST 1 OR 2 LINENUMBERS *
              ***************************
              *
              * SCN2: List 1 line number.
              * SCN3: List 2 line numbers, separated by space.
              *       (This last entry is not used).
              *
              * Exit: BC updated, DE preserved, AFHL corrupted.
              *
ED3B CDAEEF   SCN3     CALL   :EFAE      List linenr
ED3E CD6BCE            CALL   :CE6B      Print space
ED41 C3AEEF   SCN2     JMP    :EFAE      List linenr
              *
              *************************
              * LIST UNQUOTED STRING *
              *************************
              *
ED44 C3EDEF   SCN5     JMP    :EFED      List unquoted string
              *
              ***********************
              * LIST <EXPR>,<EXPR> *
              ***********************
              *
              * Exit: BC updated, DE preserved, AFHL corrupted.
              *
ED47 CDA2EE   SCN7     CALL   :EEA2      List <expr>
ED4A CD70CE   SCOEX    CALL   :CE70      Print ','
ED4D C3A2EE   SCN6     JMP    :EEA2      List <expr>
              *
              ************************************
              * LIST <EXPR> <EXPR> <EXPR> <EXPR> *
              ************************************
              *
              * Exit: BC updated, DE preserved, AFHL corrupted.
              *
ED50 CDFCEF   SCN11    CALL   :EFFC      List <expr>; print space
ED53 CDFCEF   S3EXP    CALL   :EFFC      Idem
ED56 CDFCEF   SCN8     CALL   :EFFC      Idem
ED59 C3A2EE            JMP    :EEA2      List <expr>
              *
              ******************************************
              * LIST <EXPR>,<EXPR> <EXPR>,<EXPR> <EXPR> *
              ******************************************
              *
              * Exit: BC updated, DE preserved, AFHL corrupted.
              *
ED5C CD47ED   SCN10    CALL   :ED47      List <expr>,<expr>
ED5F CD6BCE            CALL   :CE6B      Print space
ED62 CD47ED   SCN9     CALL   :ED47      List <expr>,<expr>
ED65 CD6BCE   SCSEX    CALL   :CE6B      Print space
ED68 C3A2EE   LOE343   JMP    :EEA2      List <expr>
```

```
                     *
                     ********************************
                     * LIST <EXPR>,<EXPR>(,<EXPR>) *
                     ********************************
                     *
 ED6B CD47ED         SCN12   CALL   :ED47       List <expr>,<expr>
 ED6E 0A             S1210   LDAX   B           Get next byte
 ED6F FEFF                   CPI    :FF         Terminator?
 ED71 03                     INX    B
 ED72 C8                     RZ                 Then abort
 ED73 0B                     DCX    B
 ED74 CD70CE                 CALL   :CE70       Print ','
 ED77 C3A2EE                 JMP    :EEA2       List <expr>
                     *
                     ***************************
                     * LIST <LINENR>-<LINENR> *
                     ***************************
                     *
                     * Exit: BC updated, DE preserved, AFHL corrupted.
                     *
 ED7A CDAEEF         SCN13   CALL   :EFAE       List linenr
 ED7D CDF5EF                 CALL   :EFF5       Print '-'
 ED80 2D                     DATA   :2D
 ED81 C3AEEF                 JMP    :EFAE       List linenr
                     *
                     *********************************
                     * LIST EXPRESSION AFTER 'SOUND' *
                     *********************************
                     *
                     * Exit: BC updated, AFHL corrupted.
                     *       DE: preserved if ON, corrupted if OFF.
                     *
 ED84 0A             SCN14   LDAX   B           Get byte
 ED85 FEFF                   CPI    :FF         OFF sign?
 ED87 C4FCEF                 CNZ    :EFFC       If not: List <expr>, print
                                                space
 ED8A 0A                     LDAX   B           Get next byte
 ED8B FEFF                   CPI    :FF         OFF sign?
 ED8D C250ED                 JNZ    :ED50       If not: List <expr> <expr>
                                                <expr> <expr>; abort
 ED90 CD78CE         S1410   CALL   :CE78       Else: print 'OFF'
 ED93 97ED                   DBL    :ED97
 ED95 03                     INX    B
 ED96 C9                     RET
                     *
                     * DATA:
                     *
 ED97 03             LOE354  DATA   :03
 ED98 4F                     DATA   :4F         O
 ED99 46                     DATA   :46         F
 ED9A 46                     DATA   :46         F
                     *
                     *********************************
                     * LIST EXPRESSION AFTER 'NOISE' *
                     *********************************
                     *
                     * Exit: BC updated, E preserved, AFDHL corrupted.
                     *
 ED9B 0A             SC14A   LDAX   B           Get 1st byte NCB
 ED9C FEFF                   CPI    :FF         OFF sign?
 ED9E CA90ED                 JZ     :ED90       Then print 'OFF', abort
 EDA1 C356ED                 JMP    :ED56       Else: List <expr> <expr>
```

```
                      *
                      ************************************
                      * LIST EXPRESSION AFTER 'ENVELOPE' *
                      ************************************
                      *
                      * Exit: BC updated, E preserved, AFDHL corrupted.
                      *
EDA4 CDFCEF           SCN15   CALL   :EFFC        List <expr>, print
                                                  space
EDA7 0A                       LDAX   B            Get length of expr
EDA8 03                       INX    B
EDA9 57                       MOV    D,A          into D
EDAA 15               S1510   DCR    D
EDAB FAB8ED                   JM     :EDB8        If ready
EDAE CD47ED                   CALL   :ED47        List <V>,<T>
EDB1 CDF5EF                   CALL   :EFF5        Print ';'
EDB4 3B                       DATA   :3B
EDB5 C3AAED                   JMP    :EDAA        Next <V>,<T>
EDB8 0A               S1520   LDAX   B            Get last byte of expr
EDB9 03                       INX    B
EDBA FEFF                     CPI    :FF          Terminator?
EDBC C8                       RZ                  Then abort
EDBD 0B                       DCX    B
EDBE C3A2EE                   JMP    :EEA2        List <expr>
                      *
                      ********************************
                      * LIST EXPRESSION AFTER 'MODE' *
                      ********************************
                      *
EDC1 0A               SCN16   LDAX   B            Get mode byte
EDC2 03                       INX    B
EDC3 1630                     MVI    D,:30
EDC5 B7                       ORA    A            Mode 0 (FF) ?
EDC6 FAD4ED                   JM     :EDD4        Then print '0'
EDC9 1F                       RAR                 CY=1 if A-mode
EDCA 3C               RSA10   INR    A
EDCB F5                       PUSH   PSW
EDCC 82                       ADD    D            Convert to ASCII
EDCD CD60DD                   CALL   :DD60        Print modenr
EDD0 F1                       POP    PSW
EDD1 3F                       CMC
EDD2 1641                     MVI    D,:41        Prepare print 'A'
EDD4 7A               S1610   MOV    A,D
EDD5 D460DD                   CNC    :DD60        Print 'A' if A-mode
EDD8 C9                       RET
                      *
                      *******************************************
                      * LIST EXPRESSION AFTER 'INPUT'-'READ'-'DIM' *
                      *******************************************
                      *
                      * Input with string:

EDD9 CDA2EE           SCN17   CALL   :EEA2        List string
EDDC CDF5EF                   CALL   :EFF5        Print ';'
EDDF 3B                       DATA   :3B

                      * Rest:

EDE0 0A               SCN18   LDAX   B            Get nr of variables
EDE1 03                       INX    B
EDE2 57                       MOV    D,A          into D
EDE3 D5               S1810   PUSH   D
```

```
EDE4 CDFCEE                   CALL   :EEFC     List variable reference
EDE7 D1                       POP    D
EDE8 15                       DCR    D         Decr nr of variables
EDE9 C8                       RZ               Abort if ready
EDEA CD70CE                   CALL   :CE70     Print ','
EDED C3E3ED                   JMP    :EDE3     List next variable
                   *
                   ****************************************
                   * part of RUN 'DIM': CALC. REQD. SPACE *
                   ****************************************
                   *
EDF0 C28FDE        RDM40      JNZ    :DE8F     If length element <= 254:
                                               then HL=HL*A
EDF3 7C                       MOV    A,H       Else:
EDF4 B7                       ORA    A         Set flags on hibyte
EDF5 65                       MOV    H,L
EDF6 2E00                     MVI    L,:00
EDF8 C8                       RZ               Abort if H=0
EDF9 37                       STC              Else: CY=1, L into H
EDFA C9                       RET
                   *
                   ************************
                   * RUBBISH - (not used) *
                   ************************
                   *
EDFB CE            LOE352     DATA   :CE
EDFC C3                       DATA   :C3
EDFD F4                       DATA   :F4
EDFE ED                       DATA   :ED
                   *
                   ********************************
                   * LIST EXPRESSION AFTER 'LET' *
                   ********************************
                   *
                   * Entry: BC: Points to assign statement.
                   * Exit:  BC updated, AFDEHL corrupted.
                   *
EDFF CDFCEE        SCN20      CALL   :EEFC     List lefthand variable
                                               reference
EE02 CDF5EF                   CALL   :EFF5     Print '='
EE05 3D                       DATA   :3D
EE06 C3A2EE                   JMP    :EEA2     List righthand expr
                   *
EE09                          END

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

FTEST   EC8A    FTS10   EC98    LOE252 EC23    LOE257 EC89
LOE262 ECB6     LOE263 ECC5     LOE343 ED68    LOE352 EDFB
LOE354 ED97     LOE355 ECF8     MPT46   EC6D   RDM40   EDF0
RPEEK   EC16    RPI     EC1D    RRD10   EC44   RRND    EC27
RSA10   EDCA    RSCRN   EC9D    RSGN    EC7B   S1210   ED6E
S1410   ED90    S1510   EDAA    S1520   EDB8   S1610   EDD4
S1810   EDE3    S3EXP   ED53    SC14A   ED9B   SCM10   ECEA
SCN1    ED3A    SCN10   ED5C    SCN11   ED50   SCN12   ED6B
SCN13   ED7A    SCN14   ED84    SCN15   EDA4   SCN16   EDC1
SCN17   EDD9    SCN18   EDE0    SCN2    ED41   SCN20   EDFF
SCN3    ED3B    SCN5    ED44    SCN6    ED4D   SCN7    ED47
SCN8    ED56    SCN9    ED62    SCOEX   ED4A   SCOM    ECCC
```

```
                          ORG    :EE09
                   *
                   *
                   *
                   **********************************
                   * LIST EXPRESSION AFTER 'IF' (1) *
                   **********************************
                   *
                   * Lists '<expr> THEN <expr>'.
                   *
EE09 CDFCEF        SCN21  CALL   :EFFC      List <expr>; print space
EE0C CD78CE               CALL   :CE78      Print 'THEN'
EE0F 15EE          MPT17  DBL    :EE15
EE11 03                   INX    B
EE12 C3CCEC               JMP    :ECCC      List statement
                   *
                   * DATA :
                   *
EE15 04            LOE350 DATA   :04
EE16 54                   DATA   :54        T
EE17 48                   DATA   :48        H
EE18 45                   DATA   :45        E
EE19 4E                   DATA   :4E        N
                   *
                   **********************************
                   * LIST EXPRESSION AFTER 'IF' (2) *
                   **********************************
                   *
                   * Lists '<expr> GOTO <linenr>'.
                   *
EE1A CDFCEF        SCN22  CALL   :EFFC      List <expr>; print space
EE1D CD78CE               CALL   :CE78      Print 'GOTO'
EE20 F9CB                 DBL    :CBF9
EE22 C3AEEF               JMP    :EFAE      List linenr
                   *
                   **********************************
                   * LIST EXPRESSION AFTER 'IF' (3) *
                   **********************************
                   *
                   * Lists '<expr> THEN <linenr>'.
                   *
EE25 CDFCEF        SC22A  CALL   :EFFC      List <expr>; print space
EE28 CD78CE               CALL   :CE78      Print 'THEN'
EE2B 15EE                 DBL    :EE15
EE2D C3AEEF               JMP    :EFAE      List linenr
                   *
                   ******************************
                   * LIST EXPRESSION AFTER 'FOR' *
                   ******************************
                   *
                   * Lists '<LET statement> TO <expr> (STEP <expr>)'.
                   *
EE30 CDFFED        SCN23  CALL   :EDFF      List expr of LET statement
EE33 CD6BCE               CALL   :CE6B      Print space
EE36 CD78CE               CALL   :CE78      Print 'TO'
EE39 4CEE                 DBL    :EE4C
EE3B CDA2EE               CALL   :EEA2      List <expr>
EE3E 0A                   LDAX   B          Get next byte
EE3F 03                   INX    B
EE40 FEFF                 CPI    :FF        Terminator?
EE42 C8                   RZ                Then abort
EE43 0B                   DCX    B          Else:
```

```
EE44 CD75CE                   CALL   :CE75       Print 'STEP'
EE47 3DCD                     DBL    :CD3D
EE49 C3A2EE                   JMP    :EEA2       List <expr>
                     *
                     * DATA:
                     *
EE4C 02              RLA20    DATA   :02
EE4D 54                       DATA   :54         T
EE4E 4F                       DATA   :4F         O
                     *
                     ********************************
                     * LIST EXPRESSION AFTER 'NEXT' *
                     ********************************
                     *
EE4F C3FCEE          SCN24    JMP    :EEFC       List var.ref.
                     *
                     **********************************
                     * LIST EXPRESSIONS AFTER 'PRINT' *
                     **********************************
                     *
EE52 0A              SCN25    LDAX   B           Get nr of expr
EE53 03                       INX    B
EE54 57                       MOV    D,A         into D
EE55 15              S2510    DCR    D
EE56 F8                       RM                 Abort if ready
EE57 03                       INX    B
EE58 CDA2EE                   CALL   :EEA2       List <expr>
EE5B 0A                       LDAX   B           Get next byte
EE5C 03                       INX    B
EE5D FEFF                     CPI    :FF         Terminator?
EE5F C8                       RZ                 Then abort
EE60 CD60DD                   CALL   :DD60       Else: print this byte
EE63 C355EE                   JMP    :EE55       List next expr
                     *
                     **********************************
                     * LIST EXPRESSION AFTER 'ON' (1) *
                     **********************************
                     *
                     * Lists '<expr> GOTO <linenrs>'.
                     *
EE66 CDFCEF          SCN26    CALL   :EFFC       List <expr>; print space
EE69 CD78CE                   CALL   :CE78       Print 'GOTO'
EE6C F9CB                     DBL    :CBF9
EE6E C379EE                   JMP    :EE79       List linenrs
                     *
                     **********************************
                     * LIST EXPRESSION AFTER 'ON' (2) *
                     **********************************
                     *
                     * Lists '<expr> GOSUB <linenrs>'.
                     *
EE71 CDFCEF          SCN27    CALL   :EFFC       List <expr>; print space
EE74 CD78CE                   CALL   :CE78       Print 'GOSUB'
EE77 01CC                     DBL    :CC01
EE79 0A              S2710    LDAX   B           Get nr of linenrs
EE7A 03                       INX    B
EE7B 57                       MOV    D,A         into D
EE7C CDAEEF          S2720    CALL   :EFAE       List linenr
EE7F 15                       DCR    D
EE80 C8                       RZ                 Abort if ready
EE81 CD70CE                   CALL   :CE70       Print ','
EE84 C37CEE                   JMP    :EE7C       List next linenr
```

```
                  *
                  ***********************************
                  * LIST EXPRESSION AFTER 'CALLM' *
                  ***********************************
                  *
EE87 CDA2EE       SCN28   CALL   :EEA2      List <expr>
EE8A C36EED               JMP    :ED6E      Print ','; List next expr
                  *
                  **********************
                  * SET I/O DIRECTION *
                  **********************
                  *
                  * Part of RESET (C719). Only used for A = 0.
                  * Depending on A, the input switch #0296 and
                  * the output switch #0131 are set.
                  * Default DINC is RS232.
                  *
                  *              INSW:        OTSW:
                  *       A=0:  keyboard      screen/RS232
                  *       A=1:  DINC          screen
                  *       A=2:  DINC          editbuffer
                  *       A=3:  DINC          DOUTC
                  *
EE8D 329602       MPT02   STA    :0296      Select keyb or DINC
EE90 323101               STA    :0131      Select screen/RS232/edit/
                                            DOUTC
EE93 C9                   RET
                  *
                  ***************
                  * SET VOLUMES *
                  ***************
                  *
                  * Part of RUN TALK (CD64).
                  *
                  * Entry: A:   Parameter code.
                  *        HL:  Pointer to volume byte.
                  *
EE94 E5           RTK40   PUSH   H          Save pntr to volume
EE95 6E                   MOV    L,M        Volume in L
EE96 260F                 MVI    H,:0F
EE98 1F                   RAR               Check parameter code
EE99 D27CE6               JNC    :E67C      Jump if channel 0/2

                  * If channel 1/N:

EE9C 29                   DAD    H
EE9D 29                   DAD    H
EE9E 29                   DAD    H
EE9F C37BE6               JMP    :E67B      Continu
                  *
                  **********************
                  * LIST AN EXPRESSION *
                  **********************
                  *
                  * Entry: BC:   Points to expression in program.
                  *        (BC): 1.... Expr starts with operator.
                  *              01... Variable reference.
                  *              001.. Function call.
                  *              Else  Constant.
                  * Exit:  BC points after expression.
                  *        DE preserved, AFHL corrupted.
                  *
```

```
EEA2 D5          SCEXP   PUSH  D
EEA3 0A                  LDAX  B         Get opcode
EEA4 B7                  ORA   A
EEA5 F2DFEE              JP    :EEDF     If no starting operator

                 * If starting with operator:

EEA8 03                  INX   B
EEA9 E61F                ANI   :1F       Only 5 bits of opcode
EEAB FE1A                CPI   :1A       '(' ?
EEAD F5                  PUSH  PSW       Save opcode
EEAE DCA2EE              CC    :EEA2     List expr if binary
                                         operation
EEB1 2186CF              LXI   H,:CF86   Addr table opcode strings
EEB4 54          LOE300  MOV   D,H       ) in DE
EEB5 5D                  MOV   E,L       )
EEB6 CD39DE              CALL  :DE39     HL points after table
EEB9 F1                  POP   PSW       Get opcode
EEBA F5                  PUSH  PSW
EEBB AE                  XRA   M         Comp it with table
EEBC 23                  INX   H
EEBD E61F                ANI   :1F
EEBF C2B4EE              JNZ   :EEB4     Check next opcode if not
                                         found
EEC2 EB                  XCHG            If found: addr string in HL
EEC3 23                  INX   H
EEC4 7E                  MOV   A,M       Get 1st char
EEC5 2B                  DCX   H         Pnts to length
EEC6 CD02DE              CALL  :DE02     Check if upper case char
EEC9 F5                  PUSH  PSW
EECA DC6BCE              CC    :CE6B     Print space if 1st char is
                                         a letter
EECD CD32DB              CALL  :DB32     Print string from table
EED0 F1                  POP   PSW
EED1 DC6BCE              CC    :CE6B     Print space if 1st char was
                                         a letter
EED4 CDA2EE              CALL  :EEA2     List remaining operand
EED7 F1                  POP   PSW       Get orig. opcode
EED8 FE1A                CPI   :1A       Was it '(' ?
EEDA CC55EF              CZ    :EF55     Then print ')'
EEDD D1                  POP   D
EEDE C9                  RET

                 * Not starting with operator:

EEDF 07          LOE301  RLC
EEE0 07                  RLC
EEE1 DAF2EE              JC    :EEF2     Jump if var.ref
EEE4 07                  RLC
EEE5 DAEDEE              JC    :EEED     Jump if function call

                 * If constant:

EEE8 CD84EF              CALL  :EF84     List constant
EEEB D1                  POP   D
EEEC C9                  RET

                 * If function call:

EEED CD5AEF      LOE302  CALL  :EF5A     List function reference
EEF0 D1                  POP   D
EEF1 C9                  RET
```

```

                    * If variable reference:

EEF2 CDFCEE         LOE303  CALL  :EEFC     List a var.reference
                                            (array with arguments)
EEF5 D1                     POP   D
EEF6 C9                     RET
                    *
                    ****************************
                    * LIST A VARIABLE REFERENCE *
                    ****************************
                    *
                    * SCARN:  Entry for arrays without arguments (name
                    *         only).
                    * LOE305: Entry for arrays with arguments.
                    *
                    * Entry: BC points to variable reference in program.
                    *
EEF7 16BF           SCARN   MVI   D,:BF     Set mask 'no arg'
EEF9 C3FEEE                 JMP   :EEFE
                    *
EEFC 16FF           LOE305  MVI   D,:FF     Set mask 'with arg'
EEFE D5             LOE306  PUSH  D         Save mask
EEFF 0A                     LDAX  B         Get byte
EF00 03                     INX   B
EF01 E63F                   ANI   :3F       Skip bit 6,7
EF03 57                     MOV   D,A       Rest in D
EF04 0A                     LDAX  B         Get next byte
EF05 03                     INX   B
EF06 5F                     MOV   E,A       in E (Now DE is offset
                                            of start of symtab)
EF07 2AA102                 LHLD  :02A1     Get start symtab
EF0A 19                     DAD   D         Add offset from start
EF0B D1                     POP   D         Get mask
EF0C E5                     PUSH  H         Save var.addr in symtab
EF0D CD95CA                 CALL  :CA95     Find name in symtab
EF10 7E                     MOV   A,M       Get T/L byte
EF11 A2                     ANA   D         AND with mask
EF12 F5                     PUSH  PSW
EF13 23                     INX   H
EF14 E60F                   ANI   :0F       Get length
EF16 5E                     MOV   E,M       Get 1st byte of name in E
EF17 CD44DB                 CALL  :DB44     List name; addr in HL,
                                            length in A
EF1A 1600                   MVI   D,:00
EF1C 213402                 LXI   H,:0234   Startaddr for IMPTAB
EF1F 19                     DAD   D         Addr var.type in IMPTAB
EF20 F1                     POP   PSW       Get mask
EF21 F5                     PUSH  PSW
EF22 E630                   ANI   :30       Bits 4,5 only
EF24 BE                     CMP   M         Comp with IMPTAB
EF25 CA3CEF                 JZ    :EF3C     Jump if identical
EF28 FE00                   CPI   :00       FPT ?
EF2A 1621                   MVI   D,:21     Then D: '!'
EF2C CA38EF                 JZ    :EF38
EF2F FE10                   CPI   :10       INT ?
EF31 1625                   MVI   D,:25     Then D: '%'
EF33 CA38EF                 JZ    :EF38
EF36 1624                   MVI   D,:24     Else: D: '$'
EF38 7A             LOE307  MOV   A,D
EF39 CD60DD                 CALL  :DD60     Print type sign
EF3C F1             LOE308  POP   PSW       Get mask
```

```
EF3D E640                 ANI    :40          Bit 6 only
EF3F E1                   POP    H            Get addr of string
EF40 C8                   RZ

                  * Bit 6=1 (array with arguments):

EF41 0A                   LDAX   B            Get nr of expressions
EF42 03                   INX    B
EF43 57                   MOV    D,A          in D
EF44 CDF5EF               CALL   :EFF5        Print '('
EF47 28                   DATA   :28
EF48 03           L0E309  INX    B
EF49 D5                   PUSH   D
EF4A CDA2EE               CALL   :EEA2        List expression
EF4D D1                   POP    D
EF4E 15                   DCR    D            Ready ?
EF4F C470CE               CNZ    :CE70        If not: print ','
EF52 C248EF               JNZ    :EF48        and list next expr
EF55 CDF5EF       L0E310  CALL   :EFF5        If ready: Print ')'
EF58 29                   DATA   :29
EF59 C9                   RET
                  *
                  *****************************
                  * LIST A FUNCTION REFERENCE *
                  *****************************
                  *
                  * Finds functionname in table with startaddress
                  * #CFE6 and prints it. Eventual arguments are
                  * printed between brackets.
                  *
                  * Entry: BC points to function code (#20).
                  * Exit:  BC updated, AFEHL preserved, D=0.
                  *
EF5A 03           SFUN    INX    B
EF5B 0A                   LDAX   B
EF5C 03                   INX    B
EF5D 57                   MOV    D,A
EF5E 21E6CF               LXI    H,:CFE6      Startaddr function table
EF61 15           L0E312  DCR    D
EF62 FA6BEF               JM     :EF6B        If found
EF65 CDAECA               CALL   :CAAE        Calc addr next string in tab
EF68 C361EF               JMP    :EF61        Test next function name
EF6B CD32DB       L0E313  CALL   :DB32        List function name
EF6E 7E                   MOV    A,M          Get byte after string
EF6F E60F                 ANI    :0F          Only nr of following args
EF71 57                   MOV    D,A          in D
EF72 C8                   RZ                  Abort if no arguments
EF73 CDF5EF               CALL   :EFF5        Print '('
EF76 28                   DATA   :28
EF77 CDA2EE       L0E314  CALL   :EEA2        List expression
EF7A 15                   DCR    D            Decr nr of arg
EF7B C470CE               CNZ    :CE70        If <>0, print ','
EF7E C277EF               JNZ    :EF77        and list next expr
EF81 C355EF               JMP    :EF55        If ready: print ')'
                  *
                  ********************
                  * LIST A CONSTANT *
                  ********************
                  *
                  * The constant is decoded to ASCII, prettied and
                  * printed.
                  *
```

```
                         * Codes:   #10: FPT       #18: Quoted string
                         *          #14: INT       #19: Unquoted string
                         *          #15: HEX
                         *
                         * Entry: BC: Points to constant in program.
                         * Exit:  BC updated, DE preserved, AF corrupted.
                         *         HL: points after end of printed string.
                         *
 EF84 0A                 SCON      LDAX  B          Get type of constant
 EF85 03                           INX   B
 EF86 60                           MOV   H,B        ) Addr constant in HL
 EF87 69                           MOV   L,C        )

                         * If string:

 EF88 FE18                         CPI   :18        Quoted string ?
 EF8A CAE1EF                       JZ    :EFE1      Then list it
 EF8D FE19                         CPI   :19        Unquoted string ?
 EF8F CAEDEF                       JZ    :EFED      Then list it

                         * If number:

 EF92 E7                           RST   4          Copy constant value to MACC
 EF93 0C                           DATA  :0C
 EF94 FE10                         CPI   :10        FPT ?
 EF96 CAA8EF                       JZ    :EFA8      Then list FPT value
 EF99 FE14                         CPI   :14        INT ?
 EF9B F5                           PUSH  PSW
 EF9C CCBDEF                       CZ    :EFBD      List INT value
 EF9F F1                           POP   PSW
 EFA0 C4DAEF                       CNZ   :EFDA      Else: list HEX value
 EFA3 03                 SC010     INX   B
 EFA4 03                           INX   B
 EFA5 03                           INX   B
 EFA6 03                           INX   B          BC points after constant
 EFA7 C9                           RET
 EFA8 CDD4EF             SC020     CALL  :EFD4      List FPT value
 EFAB C3A3EF                       JMP   :EFA3
                         *
                         *********************
                         * LIST A LINENUMBER *
                         *********************
                         *
                         * Entry: BC: Points to linenumber.
                         * Exit:  BC updated, DE preserved, AFHL corrupted.
                         *
 EFAE 0A                 LOE318    LDAX  B          Get hibyte linenr
 EFAF 03                           INX   B
 EFB0 67                           MOV   H,A        in H
 EFB1 0A                           LDAX  B          Get lobyte linenr
 EFB2 03                           INX   B
 EFB3 6F                           MOV   L,A        in L
 EFB4 CD46EB             SLN10     CALL  :EB46      Linenr into MACC
 EFB7 7C                           MOV   A,H
 EFB8 B5                           ORA   L          Linenr <>0 ?
 EFB9 C4BDEF                       CNZ   :EFBD      Then list linenr
 EFBC C9                           RET
                         *
                         ************************************
                         * LIST A INT VALUE OF MACC CONTENTS *
                         ************************************
                         *
```

```
                  * The value is the contents of the MACC, prepared
                  * for output, and moved into the outputbuffer
                  * DECBUF (#00E3). A Leading space is omitted.
                  *
                  * Exit: BCDE preserved. A corrupted.
                  *       HL: Points after string in DECBUF.
                  *
EFBD CD5FDB       SCINT   CALL  :DB5F     Convert MACC for INT
                                          output into DECBUF
EFC0 2A33C0       SSSPC   LHLD  :C033     Get addr DECBUF
EFC3 D5                   PUSH  D
EFC4 56                   MOV   D,M       String length in D
EFC5 23                   INX   H
EFC6 7E                   MOV   A,M       1st char in A
EFC7 FE20                 CPI   :20       Space ?
EFC9 C2CEEF               JNZ   :EFCE     Jump if no leading space
EFCC 23                   INX   H         ) Omit leading space
EFCD 15                   DCR   D         )
EFCE 7A           SSS10   MOV   A,D       Get nr of char in A
EFCF D1                   POP   D
EFD0 CD44DB               CALL  :DB44     Print contents DECBUF
EFD3 C9                   RET
                  *
                  ************************************
                  * LIST FPT VALUE OF CONTENTS MACC *
                  ************************************
                  *
                  * Exit: BCDE preserved. AF corrupted.
                  *       HL points after string in DECBUF.
                  *
EFD4 CD9BCE       SCFPT   CALL  :CE9B     Convert MACC for FPT output
EFD7 C3C0EF               JMP   :EFC0     List its contents
                  *
                  ************************************
                  * LIST HEX VALUE OF CONTENTS MACC *
                  ************************************
                  *
                  * Exit: BCDE preserved. AF corrupted.
                  *       HL points after string in DECBUF.
                  *
EFDA CDF5EF       SCHEX   CALL  :EFF5     Print '#'
EFDD 23                   DATA  :23
EFDE C34ADB               JMP   :DB4A     List in hex
                  *
                  *************************
                  * LIST A QUOTED STRING *
                  *************************
                  *
                  * Entry: BC: Points to string.
                  * Exit:  BC and HL point after string.
                  *        AF corrupted. DE preserved.
                  *
EFE1 CDF5EF       SQTS    CALL  :EFF5     Print "
EFE4 22                   DATA  :22
EFE5 CDEDEF               CALL  :EFED     List string
EFE8 CDF5EF               CALL  :EFF5     Print "
EFEB 22                   DATA  :22
EFEC C9                   RET
                  *
                  *************************
                  * LIST UNQUOTED STRING *
                  *************************
```

```
                    *
                    * Entry: BC points to string.
                    * Exit:  BC annd HL point after string.
                    *        AFDE preserved.
                    *
EFED 60             SUQTS   MOV    H,B        ) Stringaddr in HL
EFEE 69                     MOV    L,C        )
EFEF CD32DB                 CALL   :DB32      List string
EFF2 44                     MOV    B,H        )
EFF3 4D                     MOV    C,L        ) Addr after string in BC
EFF4 C9                     RET
                    *
                    ******************
                    * LIST CHARACTER *
                    ******************
                    *
                    * Entry: On stack: The address of the character to
                    *                  be printed.
                    * Exit:  BCDEHL preserved. F corrupted.
                    *        A: Character printed.
                    *
EFF5 E3             SCHRI   XTHL              Get addr from stack
EFF6 7E                     MOV    A,M        Get char
EFF7 23                     INX    H          HL pnts after char
EFF8 E3                     XTHL              Returnaddr on stack
EFF9 C360DD                 JMP    :DD60      List char.
                    *
                    ********************************
                    * LIST EXPRESSION; PRINT SPACE *
                    ********************************
                    *
EFFC C368CE         SEXPS   JMP    :CE68      List expr, print space
                    *
EFFF 3E                     DATA   :3E
                    *
                    *
                    *
F000                        END

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

LOE300 EEB4   LOE301 EEDF   LOE302 EEED   LOE303 EEF2
LOE305 EEFC   LOE306 EEFE   LOE307 EF3B   LOE308 EF3C
LOE309 EF48   LOE310 EF55   LOE312 EF61   LOE313 EF6B
LOE314 EF77   LOE318 EFAE   LOE350 EE15   MPT02  EE8D
MPT17  EE0F   RLA20  EE4C   RTK40  EE94   S2510  EE55
S2710  EE79   S2720  EE7C   SC010  EFA3   SC020  EFA8
SC22A  EE25   SCARN  EEF7   SCEXP  EEA2   SCFPT  EFD4
SCHEX  EFDA   SCHRI  EFF5   SCINT  EFBD   SCN21  EE09
SCN22  EE1A   SCN23  EE30   SCN24  EE4F   SCN25  EE52
SCN26  EE66   SCN27  EE71   SCN28  EE87   SCON   EFB4
SEXPS  EFFC   SFUN   EF5A   SLN10  EFB4   SQTS   EFE1
SSS10  EFCE   SSSPC  EFC0   SUQTS  EFED
```