```
312                     *
313                     *********
314                     * RESET *
315                     *********
316                     *
317                     * BASIC entry point. Entry via hardware reset by
318                     * means of a bootstrap on the addresslines to C000.
319                     *
320                     * This section is responsible for all 'once only'
321                     * initialisation of the hardware and the software
322                     * environment. It initialises pointers to all RAM
323                     * areas required, the interrupt system and the
324                     * software modules.
325                     *
326                     INIT
327 C719 3100F9         RESET   LXI   SP,:F900  Init. stackpointer
328 C71C 3E30                 MVI   A,:30     ) cassette motors off;
329 C71E 324000               STA   :0040     ) paddle enable off;
330 C721 3206FD               STA   :FD06     ) select ROM bank 0
331 C724 CDFBD8               CALL  :D8FB     Init. interrupt system
332 C727 210000               LXI   H,:0000
333 C72A 22A302               SHLD  :02A3     Set for no Basic
334 C72D AF                   XRA   A
335 C72E 329302               STA   :0293     RNDLY=0
336 C731 00                   NOP
337 C732 00                   NOP
338
339                     * Init. math.package:
340
341 C733 11F2C7               LXI   D,:C7F2   Addr table error vectors
342 C736 21E0DD               LXI   H,:DDE0   Addr routine get char/line
343 C739 CD03C0               CALL  :C003     Package initialisation
344
345                     * Init. screen RAM:
346
347 C73C CDFBC7               CALL  :C7FB     Check available RAM space
348 C73F 2B                   DCX   H         Highest RAM address
349 C740 11E0C7               LXI   D,:C7E0   Addr screen default data
350 C743 EF                   RST   5
351 C744 00                   DATA  :00       Init. screen RAM
352
353                     * Init. I/O:
354
355 C745 AF                   XRA   A
356 C746 CD8DEE               CALL  :EE8D     (0) Init. I/O switching
357                                           (input keyb; output screen
358                                           + RS232)
359 C749 323501               STA   :0135     Input from keyboard for
360                                           encoding
361 C74C 3EC0                 MVI   A,:C0
362 C74E 32F5FF               STA   :FFF5     Init. TICC baud rate
363
364                     * Init. screen:
365
366 C751 CDFFDA               CALL  :DAFF     Print 'DAI PERSONAL
367 C754 A8C7                 DBL   :C7A8     COMPUTER'
368 C756 2AA502               LHLD  :02A5     Get bottom screen RAM
369 C759 117B09               LXI   D,:097B
370 C75C 19                   DAD   D         Get line mode byte of line
371                                           with 'DAI pC'
372 C75D 365F                 MVI   M,:5F     Set for medium resolution
373 C75F 11D0FF               LXI   D,:FFD0
```

```
374 C762 19                     DAD   D         Create new line mode byte
375                                               for line with 'COMPUTER'
376 C763 CDF9CE                 CALL  :CEF9     Place 'COMPUTER' on new line
377 C766 160F                   MVI   D,:0F     Nr of blanking lines
378 C768 CDCFCE       HD10      CALL  :CECF     Blank next 15 lines
379 C76B 15                     DCR   D
380 C76C C268C7                 JNZ   :C768     Next line
381
382                   * Prepare BASIC:
383
384 C76F CD2DD7                 CALL  :D72D     Init. Soundgen/DCEbus/
385                                               transfer cassette data/
386                                               set start HEAP/get evt.
387                                               DCE-inputs
388 C772 210001                 LXI   H,:0100
389 C775 229D02                 SHLD  :029D     HEAP size default value
390 C778 CDB5DE                 CALL  :DEB5     Run 'NEW'
391 C77B 3E10                   MVI   A,:10
392 C77D 323D01                 STA   :013D     Select cassette port 1
393 C780 21C5E8                 LXI   H,:E8C5   (3) Ptr. to ASCII table
394 C783 CD60D5                 CALL  :D560     Init keyboard pointers
395 C786 CD01D1                 CALL  :D101     Init string handler
396 C789 3E00                   MVI   A,:00
397 C78B 328F02                 STA   :028F     Default number type FPT
398 C78E 117502                 LXI   D,:0275   Begin IMPTAB
399 C791 218F02                 LXI   H,:028F   End IMPTAB
400 C794 CD7CDE                 CALL  :DE7C     Init. implicit type table
401                                               with 0 (= FPT)
402 C797 FB                     EI
403 C798 CF                     RST   1         Wait for input keyboard
404 C799 15                     DATA  :15       or RS232
405 C79A 00                     NOP
406 C79B 21EEC7                 LXI   H,:C7EE   Ptr. to mode 0 colours
407 C79E EF                     RST   5         Set text colours
408 C79F 06                     DATA  :06
409
410                   * Entry from utility:
411
412 C7A0 CDE4CE       RINIT     CALL  :CEE4     Select ROM bank 0 and print
413 C7A3 D3C7                   DBL   :C7D3     'BASIC V1.0'
414 C7A5 C318C8                 JMP   :C818     Into BASIC monitor
415                   *
416                   * INITIALISATION SCREEN DATA:
417                   *
418 C7A8 0D           MSGHDR    DATA  :0D       Frigged screen header
419 C7A9 0D                     DATA  :0D
420 C7AA 0D                     DATA  :0D
421 C7AB 0D                     DATA  :0D
422 C7AC 0D                     DATA  :0D
423 C7AD 0D                     DATA  :0D
424 C7AE 20                     DATA  :20
425 C7AF 44                     DATA  :44       D
426 C7B0 41                     DATA  :41       A
427 C7B1 49                     DATA  :49       I
428 C7B2 20                     DATA  :20
429 C7B3 50                     DATA  :50       P
430 C7B4 45                     DATA  :45       E
431 C7B5 52                     DATA  :52       R
432 C7B6 53                     DATA  :53       S
433 C7B7 4F                     DATA  :4F       O
434 C7B8 4E                     DATA  :4E       N
435 C7B9 41                     DATA  :41       A
```

```
436 C7BA 4C                     DATA  :4C       L
437 C7BB 20                     DATA  :20
438 C7BC 20                     DATA  :20
439 C7BD 20                     DATA  :20
440 C7BE 20                     DATA  :20
441 C7BF 20                     DATA  :20
442 C7C0 20                     DATA  :20
443 C7C1 20                     DATA  :20
444 C7C2 20                     DATA  :20
445 C7C3 20                     DATA  :20
446 C7C4 20                     DATA  :20
447 C7C5 20                     DATA  :20
448 C7C6 20                     DATA  :20
449 C7C7 20                     DATA  :20
450 C7C8 20                     DATA  :20
451 C7C9 43                     DATA  :43       C
452 C7CA 4F                     DATA  :4F       O
453 C7CB 4D                     DATA  :4D       M
454 C7CC 50                     DATA  :50       P
455 C7CD 55                     DATA  :55       U
456 C7CE 54                     DATA  :54       T
457 C7CF 45                     DATA  :45       E
458 C7D0 52                     DATA  :52       R
459 C7D1 0D                     DATA  :0D
460 C7D2 00                     DATA  :00
461                     *
462 C7D3 0C             MSGIN   DATA  :0C
463 C7D4 42                     DATA  :42       B
464 C7D5 41                     DATA  :41       A
465 C7D6 53                     DATA  :53       S
466 C7D7 49                     DATA  :49       I
467 C7D8 43                     DATA  :43       C
468 C7D9 20                     DATA  :20
469 C7DA 56                     DATA  :56       V
470 C7DB 31                     DATA  :31       1
471 C7DC 2E                     DATA  :2E       .
472 C7DD 30                     DATA  :30       0
473 C7DE 0D                     DATA  :0D
474 C7DF 00                     DATA  :00
475                     *
476                     * SCREEN INITIALISATION PARAMETERS:
477                     *
478 C7E0 01             SIPAR   DATA  :01       Default cursor type
479 C7E1 5F                     DATA  :5F       Default cursor ASCII value
480                     *
481 C7E2 05                     DATA  :05       )
482 C7E3 0F                     DATA  :0F       ) Colours COLORT
483 C7E4 0F                     DATA  :0F       ) during Reset
484 C7E5 05                     DATA  :05       )
485                     *
486 C7E6 00                     DATA  :00       )
487 C7E7 05                     DATA  :05       ) Default colours
488 C7E8 0A                     DATA  :0A       ) COLORG
489 C7E9 0F                     DATA  :0F       )
490                     *
491 C7EA 01CA                   DBL   :CA01     Addr. memory management
492                                             routine
493 C7EC 25CA                   DBL   :CA25     Addr. emergency stop routine
494                     *
495 C7EE 08             STCOL   DATA  :08       )
496 C7EF 00                     DATA  :00       ) Default colours
497 C7F0 00                     DATA  :00       ) COLORT
```

```
498 C7F1 08                     DATA  :08        )
499                     *
500                     * MATH. ERROR ROUTINE VECTORS:
501                     *
502 C7F2 1FDA           MEVEC   DBL   :DA1F      Addr. Overflow error routine
503 C7F4 15DA                   DBL   :DA15      Addr. Number out of range
504                                              error routine
505 C7F6 FAC7                   DBL   :C7FA      Addr. Return
506 C7F8 24DA                   DBL   :DA24      Addr. Error routine Division
507                                              by zero
508                     *
509 C7FA C9             MERET   RET              Return
510                     *
511                     *
512                     *
513 C7FB                        END
```

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

HBC	C653	HCB	C614	HD10	C768	I4	C63D
INIT	C719	IROR	C6B0	LBK30	C6B6	LC120	C61B
LC121	C634	LC122	C641	LC123	C66A	LC124	C66E
LC125	C677	LC126	C680	LC127	C68D	LC128	C691
LC129	C69C	MARST	C6C0	MERET	C7FA	MEVEC	C7F2
MPT26	C6B4	MRDCL	C6F2	MRS10	C6CF	MSGHDR	C7A8
MSGIN	C7D3	PINPLN	C6A0	RESET	C719	RINIT	C7A0
RNDA	C6A8	RNDB	C6AC	SCRST	C6FD	SIPAR	C7E0
SPT02	C6BA	SRS10	C705	STCOL	C7EE	UTRST	C70E

```
002                             ORG   :C7FB
003                     *
004                     *
005                     *
006                     ************************************
007                     * CHECK FOR HIGHEST RAM ADDRESS *
008                     ************************************
009                     *
010                     * Entry: No conditions.
011                     * Exit:  HL points after RAM.
012                     *        BC preserved, ADEF corrupted.
013                     *
014 C7FB 110010         MEMCHK  LXI   D,:1000
015 C7FE 210000                 LXI   H,:0000   Start at 0000
016 C801 19             MCK10   DAD   D         Incr. with #1000
017 C802 7E                     MOV   A,M       Get what is in memory
018 C803 2F                     CMA             Take its complement
019 C804 77                     MOV   M,A       and store it back
020 C805 BE                     CMP   M         Then compare
021 C806 2F                     CMA
022 C807 77                     MOV   M,A       Restore original value
023 C808 CA01C8                 JZ    :C801     Next block if still RAM
024 C80B C9                     RET
025                     *
026                     ************************
027                     * START FROM SCRATCH *
028                     ************************
029                     *
030                     * Entry to Basic monitor.
031                     *
032
033                     * If out of a Hard BREAK:
034
035 C80C 3100F9         RSTART  LXI   SP,:F900  Reset stackpointer
036 C80F CDE4CE                 CALL  :CEE4     Select ROM bank 0,
037 C812 B8DB                   DBL   :DBB8     print '*** BREAK'
038
039                     * Re-enter Basic after run-time error,
040                     * except on input:
041
042 C814 AF             START   XRA   A
043 C815 323501                 STA   :0135     Input from keyb/DINC
044
045                     * Entry on reset, after encoding program line,
046                     * after END:
047
048 C818 2100F9         ST10    LXI   H,:F900
049 C81B 222701                 SHLD  :0127     Reset current base stack
050 C81E F9                     SPHL            Reset stackpointer
051 C81F AF                     XRA   A
052 C820 322601                 STA   :0126     No suspended program
053
054                     * Restart interpreter; entry after end of program,
055                     * after direct command, after soft BREAK, after
056                     * direct command error, after STOP:
057
058 C823 2A2701         ST20    LHLD  :0127     Get saved stackpointer
059 C826 F9                     SPHL            Set it to saved value
060 C827 210000                 LXI   H,:0000
061 C82A 220001                 SHLD  :0100     Reset current line nr.
062 C82D 220401                 SHLD  :0104     No running loops
063 C830 221301                 SHLD  :0113     No active subroutine call
```

```
064 C833 7C                    MOV   A,H
065 C834 322201                STA   :0122     No encoding of stored line
066 C837 00                    NOP
067 C838 00                    NOP
068 C839 00                    NOP
069 C83A 211701                LXI   H,:0117
070 C83D 77                    MOV   M,A       No running of inputs
071 C83E 23                    INX   H
072 C83F 77                    MOV   M,A       No running of program
073 C840 CD88D9                CALL  :D988     Enable keyboard interrupt
074 C843 CDDBD9                CALL  :D9DB     Enable clock interrupt
075 C846 3A3501        ST23    LDA   :0135     Get input direction
076 C849 FE02                  CPI   :02
077 C84B CC79D8                CZ    :D879     EFSW=2: input from editbuf
078 C84E D267C8                JNC   :C867     EFSW>=2: encode ENCODE TEXTLINE IF EDITBUF NOT EMPTY
079 C851 3E2A                  MVI   A,:2A
080 C853 CD1ADD                CALL  :DD1A     Print '*', scan keyboard (ERRATUM DANAMIC 83-17 P231)
081                                            and display characters
082                                            until Break or car.ret
083                                            (If no input is given,
084                                            the DAI remains here in a
085                                            endless loop).
086 C856 DA46C8                JC    :C846     If BREAK: new inputs
087 C859 CDD2DD                CALL  :DDD2     Get char from line, neglect
088                                            TAB and space
089 C85C FE0D                  CPI   :0D
090 C85E CA46C8                JZ    :C846     If car.ret: new inputs
091 C861 CD0DDE                CALL  :DE0D     Check if char is number
092 C864 D26DC8                JNC   :C86D     If no leading nr: encode cmd
093
094                    * Encode program line (if 1st char is number):
095
096 C867 CD18C9         ST24   CALL  :C918     Encode program line, update
097                                            program
098 C86A C318C8                JMP   :C818     Get next input line; kill
099                                            any suspended program
100
101                    * Encode direct command (if 1st char is no number):
102
103 C86D 1680           ST25   MVI   D,:80     Mask for direct command
104 C86F E5                    PUSH  H         Pointer to RUNF
105 C870 213F01                LXI   H,:013F   Addr EBUF
106 C873 E5                    PUSH  H         Save it on stack
107 C874 CF                    RST   1         Encode immediate cmd line
108 C875 00                    DATA  :00
109 C876 3600                  MVI   M,:00     Dummy end of program
110 C878 CD5EDD                CALL  :DD5E     Print car.ret
111 C87B C1                    POP   B         Get EBUF pntr
112 C87C E1                    POP   H         Pntr to RUNF
113 C87D 36FF                  MVI   M,:FF     Set flag running programs
114
115                    * Run a Basic line:
116
117 C87F 0A           ST30     LDAX  B         Get 1st byte from EBUF:
118                                            <  #80: length,
119                                            >= #80: Token.
120 C880 03           ST35     INX   B
121 C881 87                    ADD   A         Calc offset from CF00
122 C882 D2E5C8                JNC   :C8E5     Jump if length byte
123 C885 6F                    MOV   L,A       Get table address in HL
124 C886 26CF                  MVI   H,:CF
125 C888 7E                    MOV   A,M       ) Get addr Basic routine
```

```
126 C889 23                 INX   H          ) from table in HL
127 C88A 66                 MOV   H,M        )
128 C88B 6F                 MOV   L,A        )
129 C88C CDA9C8             CALL  :C8A9      Perform this routine
130
131                 * Commands return here:
132
133 C88F DAAAC8     ENDCOM  JC    :C8AA      Jump if special action
134
135                 * If suspended:
136
137 C892 60                 MOV   H,B
138 C893 69                 MOV   L,C
139 C894 220201             SHLD  :0102      Remember start next cmd
140 C897 3AC402             LDA   :02C4
141 C89A B7                 ORA   A          BREAK flag set?
142 C89B CA7FC8             JZ    :C87F      Run Basic line if not
143 C89E 00                 NOP
144 C89F 00                 NOP
145 C8A0 00                 NOP
146 C8A1 3EFF               MVI   A,:FF
147 C8A3 32C402             STA   :02C4      Set BREAK flag 'serviced'
148 C8A6 C3C0C8             JMP   :C8C0      Handle break
149
150                 * Run a BASIC line:
151
152 C8A9 E9         DCALL   PCHL             Addr Basic routine in PC
153
154                 * If special end of action:
155
156 C8AA FE02         ST40    CPI   :02
157 C8AC CAC0C8             JZ    :C8C0      If soft break (2)
158 C8AF D2B8C8             JNC   :C8B8      If STOP (3)
159 C8B2 EA18C8             JPE   :C818      If can't continu (1)
160 C8B5 C308C9             JMP   :C908      If after LOAD (0)
161
162                 * If 'STOP':
163
164 C8B8 60         ST45    MOV   H,B
165 C8B9 69                 MOV   L,C
166 C8BA 220201             SHLD  :0102      Remember where next cmd
167 C8BD C3C5C8             JMP   :C8C5      Print 'IN LINE ..' and
168                                          handle a break
169
170                 * If suspended (soft Break handling):
171
172 C8C0 CDFFDA     ST50    CALL  :DAFF      Print car.ret; 'BREAK'.
173 C8C3 C5DB               DBL   :DBC5
174 C8C5 CD75DA     ST55    CALL  :DA75      Print 'IN LINE .....'
175                                          or car.ret
176 C8C8 CA23C8             JZ    :C823      Jump if immediate cmd
177
178                 * Only if 'break' in program:
179
180 C8CB 21EBFF             LXI   H,:FFEB    Frame length
181 C8CE 39                 DAD   SP         New stack level
182 C8CF 44                 MOV   B,H
183 C8D0 4D                 MOV   C,L
184 C8D1 222701             SHLD  :0127      Set new base stack
185 C8D4 F9                 SPHL             Set stackpointer
186 C8D5 110001             LXI   D,:0100    ) Boundaries frame
187 C8D8 211501             LXI   H,:0115    )
```

```
188 C8DB CD4FDE             CALL  :DE4F     Save program status
189                                         (FRAME) on stack.
190 C8DE 212601             LXI   H,:0126
191 C8E1 34                 INR   M         Set flag existence saved
192                                         program
193 C8E2 C323C8             JMP   :C823     Run again
194
195                 * Length byte or end flag:
196
197 C8E5 CA23C8      ST60   JZ    :C823     If end immediate cmd
198                                         line or end program
199 C8E8 60                 MOV   H,B
200 C8E9 69                 MOV   L,C
201 C8EA 220001             SHLD  :0100     Store start current line
202 C8ED 2A1501             LHLD  :0115     Get trace + step flag
203 C8F0 7C                 MOV   A,H
204 C8F1 B5                 ORA   L
205 C8F2 CA00C9             JZ    :C900     If no step/trace flag
206
207                 * If step/trace flag set:
208
209 C8F5 E5                 PUSH  H
210 C8F6 C5                 PUSH  B
211 C8F7 CDA4CE             CALL  :CEA4     List current line
212 C8FA C1                 POP   B
213 C8FB F1                 POP   PSW       Get step flag
214 C8FC B7                 ORA   A         If set:
215 C8FD C4DAD6             CNZ   :D6DA     Wait for spacebar pressed
216 C900 03          ST65   INX   B
217 C901 03                 INX   B         Pnts after linenr
218 C902 DACBC8             JC    :C8CB     If Break
219 C905 C37FC8             JMP   :C87F     Run next BASIC line
220
221                 * Special action after LOAD:
222
223 C908 2A0001      ST70   LHLD  :0100     Get start current line
224 C90B 7C                 MOV   A,H
225 C90C B5                 ORA   L         Direct cmd?
226 C90D CA7FC8             JZ    :C87F     Then run Basic line
227 C910 3100F9             LXI   SP,:F900  Else: reset stackpointer
228 C913 3E87               MVI   A,:87     Simulate Token 'RUN'
229 C915 C380C8             JMP   :C880     Pretend RUN cmd
230                 *
231                 * PROGRAM INPUT:
232                 *
233                 * Encodes a program line and updates the stored
234                 * program.
235                 *
236                 * Entry: C: Input count / offset.
237                 * Exit:  C: Offset after line.
238                 *        AFBDEHL preserved.
239                 *
240 C918 213F01      PROGI  LXI   H,:013F   Addr buf for encoded cmds
241 C91B CF                 RST   1         Get linenr
242 C91C 03                 DATA  :03
243 C91D CDD2DD             CALL  :DDD2     Get char from line;
244                                         neglect TAB + space
245 C920 FE0D               CPI   :0D       Car.ret ?
246 C922 CCA2C9             CZ    :C9A2     Delete old version if
247                                         only linenr given
248 C925 CA3BC9             JZ    :C93B     Jump if linenr only
249 C928 D5                 PUSH  D         Remember linenr
```

```
C929 1640          MVI   D,:40     Mask for 'stored cmd'
C92B CD3CC9        CALL  :C93C     Encode a line
C92E 7D            MOV   A,L
C92F D63F          SUI   :3F       Length string in A
C931 323E01        STA   :013E     Length in EBUF
C934 D1            POP   D         Get linenr
C935 CDA2C9        CALL  :C9A2     Delete old line
C938 CDBDC9        CALL  :C9BD     Insert new line
C93B C9      PGI20 RET
                   *
                   * ENCODE A LINE:
                   *
                   * Exit: DE restored.
                   *       HL points to 1st free byte in EBUF.
                   *        C points after car.ret in input.
                   *        A=0, F corrupted.
                   *
C93C D5      ELINA PUSH  D
C93D C5            PUSH  B
C93E E5            PUSH  H
C93F 210000        LXI   H,:0000
C942 39            DAD   SP        Stackpointer in HL
C943 221D01        SHLD  :011D     Save stackpointer
C946 E1            POP   H
C947 E5            PUSH  H
C948 3E01          MVI   A,:01
C94A 322201        STA   :0122     Set encoding a stored line
C94D CF            RST   1         Encode inputs
C94E 00            DATA  :00
C94F D1            POP   D         ) Cancel Push B,H
C950 D1            POP   D         )
C951 AF      ELA10 XRA   A
C952 322201        STA   :0122     No encoding stored line
C955 D1            POP   D
C956 C9            RET
                   *
                   ****************************************
                   * ERROR WHILE ENCODING A STORED LINE *
                   ****************************************
                   *
                   * Restores stackpointer, adds '***' to begin of
                   * line, adds '?' to place of error. Line is entered
                   * into the encoded inputbuffer (EBUF).
                   *
                   * Entry: B: Errorcode.
                   *        C: Place of error.
                   *        On stack: BC points to input.
                   *                  HL points to EBUF.
                   *
C957 2A1D01  ELARS LHLD  :011D     Get ERSSP
C95A F9            SPHL            Restore stackpointer
C95B E1            POP   H         Get buffer pointer
C95C 78            MOV   A,B       Errorcode in A
C95D 51            MOV   D,C       Place of error in D
C95E C1            POP   B         Get input pointer
C95F 47            MOV   B,A
C960 36B1          MVI   M,:B1     Token for '***' in EBUF
C962 23            INX   H
C963 E5            PUSH  H         Save buf pointer
C964 23            INX   H
C965 79      ELA20 MOV   A,C       ) Place of error
C966 BA            CMP   D         ) reached ?
```

```
312 C967 3E3F              MVI   A,:3F
313 C969 CC95C9            CZ    :C995      Then insert '?'
314 C96C CDE0DD            CALL  :DDE0      Get char. from line
315 C96F 0C                INR   C          Update inputpointer
316 C970 FE0D              CPI   :0D        Line done ?
317 C972 C495C9            CNZ   :C995      Insert char in EBUF if not
318 C975 C265C9            JNZ   :C965      Next char if not ready
319 C978 7D                MOV   A,L        Lobyte EBUF pntr in A
320 C979 D1                POP   D          Addr after '***'
321 C97A 93                SUB   E
322 C97B 3D                DCR   A
323 C97C 12                STAX  D          Store length in EBUF
324 C97D 3600              MVI   M,:00      0 after string
325 C97F C5                PUSH  B          Save errormessage pntr
326 C980 42                MOV   B,D        ) EBUF pntr in BC
327 C981 4B                MOV   C,E        )
328 C982 0B                DCX   B
329 C983 0B                DCX   B
330 C984 0B                DCX   B
331 C985 0B                DCX   B          Pnts to begin EBUF
332 C986 CD5EDD            CALL  :DD5E      Print car.ret
333 C989 CDABEC            CALL  :ECAB      (0) List current line
334 C98C C1                POP   B          Get errormessage pntr
335 C98D E5                PUSH  H
336 C98E CD50DA            CALL  :DA50      Print errormessage
337 C991 E1                POP   H
338 C992 C351C9            JMP   :C951      Store 0 in ERSFL, Pop D,
339                                         and ret.
340                  *
341                  * INSERT CHARACTER IN ENCODED INPUT BUFFER:
342                  *
343                  * A character is inserted in the EBUF only if
344                  * there is space available.
345                  *
346                  * Entry: HL: 1st free location in EBUF.
347                  *        A:  Character to be inserted.
348                  * Exit:  HL updated. AFBCDE preserved.
349                  *
350 C995 F5           ELAIN  PUSH  PSW
351 C996 7D                   MOV   A,L        Get lobyte of EBUF pntr
352 C997 FEBC                CPI   :BC        Buffer full ?
353 C999 CAA0C9              JZ    :C9A0      Then abort
354 C99C F1                 POP   PSW
355 C99D 77                 MOV   M,A        Char into EBUF
356 C99E 23                 INX   H          Update pntr
357 C99F C9                 RET
358
359                  * If EBUF full:
360
361 C9A0 F1           EAI10  POP   PSW        No action
362 C9A1 C9                 RET
363                  *
364                  **********************************
365                  * DELETE OLD VERSION OF A LINE *
366                  **********************************
367                  *
368                  * A textline is deleted by moving the rest of the
369                  * textbuffer and the symboltable 'downwards'.
370                  *
371                  * Entry: DE: requested linenumber.
372                  * Exit:  DE points to linenr after deleted line.
373                  *        AFBCHL preserved.
```

```
374                     *
375 C9A2 F5             LDEL    PUSH  PSW
376 C9A3 C5                     PUSH  B
377 C9A4 E5                     PUSH  H
378 C9A5 EB                     XCHG            Linenr in HL
379 C9A6 CDF6CA                 CALL  :CAF6     Addr line in textbuf in HL
380 C9A9 D2B8C9                 JNC   :C9B8     Abort if not found
381 C9AC 7E                     MOV   A,M       Get line length
382 C9AD 2F                     CMA
383 C9AE 5F                     MOV   E,A       Compl. value in E
384 C9AF 16FF                   MVI   D,:FF
385 C9B1 CD39DE                 CALL  :DE39     HL=HL-line length
386 C9B4 EB                     XCHG
387 C9B5 CDD1C9                 CALL  :C9D1     Move program buffers
388 C9B8 EB             LDL10   XCHG
389 C9B9 E1                     POP   H
390 C9BA C1                     POP   B
391 C9BB F1                     POP   PSW
392 C9BC C9                     RET
393                     *
394                     **********************
395                     * INSERT A NEW LINE *
396                     **********************
397                     *
398                     * Inserts a encoded line in the textbuffer.
399                     * Required space for the textline is made by
400                     * shifting the rest of the textbuffer and the
401                     * symboltable 'upwards'.
402                     *
403                     * Entry: DE: Destination address in textbuffer.
404                     *        HL: Points after string in EBUF.
405                     *        A:  Length string in EBUF.
406                     *
407 C9BD C5             LINS    PUSH  B
408 C9BE E5                     PUSH  H
409 C9BF 6F                     MOV   L,A
410 C9C0 2600                   MVI   H,:00     String length in HL
411 C9C2 23                     INX   H         Required space in HEAP
412 C9C3 D5                     PUSH  D
413 C9C4 CDD1C9                 CALL  :C9D1     Move program buffers
414 C9C7 C1                     POP   B
415 C9C8 E1                     POP   H
416 C9C9 113E01                 LXI   D,:013E   Startaddr. EBUF
417 C9CC CD4FDE                 CALL  :DE4F     Transfer data from EBUF
418                                             into textbuffer
419 C9CF C1                     POP   B
420 C9D0 C9                     RET
421                     *
422                     ************************
423                     * MOVE PROGRAM BUFFERS *
424                     ************************
425                     *
426                     * Moves a part (or the whole) textbuffer and the
427                     * whole symboltable up or down.
428                     * The startaddress of the textbuffer and the end
429                     * of the symboltable are set depending on the
430                     * heap size. The heap pointers are updated.
431                     *
432                     * Entry: DE: Address from where to update.
433                     *        HL: Length of area to be inserted/deleted.
434                     * Entry if running 'NEW' only:
435                     *        BC: 0.
```

```
436                     *         DE: startaddress Heap.
437                     *         HL: Size Heap.
438                     *         At entry, the pointers for textbuf and
439                     *         symtab are as if HEAPsize is zero.
440                     * Exit:   AFBCDE preserved.
441                     *         HL: New startaddress.
442                     *
443 C9D1 C5            PROGM   PUSH  B
444 C9D2 D5                    PUSH  D
445 C9D3 42                    MOV   B,D       ) Addr from where to
446 C9D4 4B                    MOV   C,E       ) update in BC
447 C9D5 EB                    XCHG            Length area in DE
448 C9D6 2AA302                LHLD  :02A3     Get end symtab
449 C9D9 E5                    PUSH  H
450 C9DA D5                    PUSH  D
451 C9DB 19                    DAD   D         New end symtab
452 C9DC EB                    XCHG
453 C9DD 2AA502                LHLD  :02A5     Get bottom screen RAM
454 C9E0 CD14DE                CALL  :DE14     Check for overflow
455 C9E3 EB                    XCHG            End symtab in HL
456 C9E4 DA10DA                JC    :DA10     Evt. run 'OUT OF MEMORY'
457 C9E7 22A302                SHLD  :02A3     Store end symbol table
458 C9EA D1                    POP   D
459 C9EB D5                    PUSH  D
460 C9EC 2AA102                LHLD  :02A1     Get begin symtab
461 C9EF 19                    DAD   D         New begin symtab
462 C9F0 22A102                SHLD  :02A1     Store begin symbol table
463 C9F3 E1            PRGM1   POP   H
464 C9F4 50                    MOV   D,B
465 C9F5 59                    MOV   E,C       Startaddr in DE
466 C9F6 19                    DAD   D         New startaddr
467 C9F7 44                    MOV   B,H
468 C9F8 4D                    MOV   C,L       New startaddr in BC
469 C9F9 E3                    XTHL            Get old end symtab
470 C9FA CD4FDE                CALL  :DE4F     Move textbuf + symtab
471                                            from old to new addr.
472 C9FD E1                    POP   H
473 C9FE D1                    POP   D
474 C9FF C1                    POP   B
475 CA00 C9                    RET
476                     *
477                     *
478                     *
479 CA01                       END
```

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

DCALL   C8A9    EAI10   C9A0    ELA10   C951    ELA20   C965
ELAIN   C995    ELARS   C957    ELINA   C93C    ENDCOM  C88F
LDEL    C9A2    LDL10   C9B8    LINS    C9BD    MCK10   C801
MEMCHK  C7FB    PGI20   C93B    PRGM1   C9F3    PROGI   C918
PROGM   C9D1    RSTART  C80C    ST10    C818    ST20    C823
ST23    C846    ST24    C867    ST25    C86D    ST30    C87F
ST35    C880    ST40    C8AA    ST45    C8B8    ST50    C8C0
ST55    C8C5    ST60    C8E5    ST65    C900    ST70    C908
START   C814
```

```
002                             ORG   :CA01
003                     *
004                     *
005                     *
006                     ******************************
007                     * MEMORY MANAGEMENT ROUTINE *
008                     ******************************
009                     *
010                     * This routine is used to obtain and release
011                     * memory for its display, as the mode changes.
012                     *
013                     * Entry: HL: Lowest screen RAM byte required.
014                     *        CY=1: Space to this point at least is
015                     *              now required. Additional space
016                     *              may not be released.
017                     *        CY=0: Space is held to or below this
018                     *              point. Any space held below this
019                     *              point is no longer required.
020                     * Exit:  CY=1: O.K.
021                     *        CY=0: No space available.
022                     *        AFBCDE preserved.
023                     *
024                     ASKRM
025 CA01 00             SMKRM   NOP
026 CA02 F5                     PUSH  PSW
027 CA03 D5                     PUSH  D
028 CA04 D21BCA                 JNC   :CA1B     If space not reqd anymore
029 CA07 EB                     XCHG            Lowest byte in DE
030 CA08 2AA502                 LHLD  :02A5     Get bottom screen RAM
031 CA0B CD14DE                 CALL  :DE14     Check for overflow
032 CA0E DA1ECA                 JC    :CA1E     If OK
033 CA11 2AA302                 LHLD  :02A3     Get begin free RAM
034 CA14 EB                     XCHG            and store it in DE
035 CA15 CD14DE                 CALL  :DE14     Still free RAM available?
036 CA18 DA21CA                 JC    :CA21     If not
037 CA1B 22A502         ASR10   SHLD  :02A5     Update bottom screen RAM
038 CA1E C3ADCE         ASR20   JMP   :CEAD     Return, set CY=1
039
040                     * If no RAM space available:
041
042 CA21 D1             ASR30   POP   D
043 CA22 C3B1CE                 JMP   :CEB1     Return, set CY=0
044                     *
045                     ********************************************
046                     * EMERGENCY STOP ROUTINE (Graphics modes) *
047                     ********************************************
048                     *
049                     * This routine is used if no sufficient space for
050                     * a A-mode is available.
051                     *
052 CA25 CDC6CE         EMSTP   CALL  :CEC6     Set up HEAPsize + buffers
053                                             to default values
054 CA28 3EFF                   MVI   A,:FF
055 CA2A EF                     RST   5         Change to mode 0
056 CA2B 18                     DATA  :18
057 CA2C CDE4CE                 CALL  :CEE4     Select ROM-bank 0; Run error
058 CA2F 89DB                   DBL   :DB89     'OUT OF SPACE FOR MODE'
059 CA31 C318C8                 JMP   :C818     Return to BASIC monitor
060                     *
061                     ******************************************
062                     * FIND STRING BASIC INSTRUCTION IN TABLE *
063                     ******************************************
```

```
064                     *
065                     * Looks for table entry whose name is the initial
066                     * string of input, beginning at C'th position.
067                     * REMARK: Variables, beginning with a reserved
068                     *         string are not allowed.
069                     *
070                     * Entry: HL: Startaddress table.
071                     *        C : Position char on current line.
072                     *        E : Number of info bytes -1.
073                     * Exit:   If found: CY=1:
074                     *             HL: Address in table where string
075                     *                 can be found.
076                     *             C : Position on current line after
077                     *                 last char.
078                     *             A : Last byte of string typed in.
079                     *             BE preserved, D=0.
080                     *         If not found: CY=0:
081                     *             C : Points to next char.
082                     *             HL: Points after end of table.
083                     *             BE preserved. A,D=0.
084                     *
085 CA34 CDD2DD      LOOKC   CALL  :DDD2     Get char from line,
086                                         neglect TAB and space
087 CA37 56          LKC10   MOV   D,M       Get length byte of string
088 CA38 23                  INX   H         Points to 1st stringchar.
089 CA39 7A                  MOV   A,D
090 CA3A B7                  ORA   A         Is length zero?
091 CA3B C8                  RZ              Then abort
092 CA3C C5                  PUSH  B         Save position of 1st char
093 CA3D CDE0DD      LKC20   CALL  :DDE0     Get char from line
094 CA40 0C                  INR   C         Points to next char on line
095 CA41 BE                  CMP   M         Is it identical to the
096                                         one in table?
097 CA42 23                  INX   H         Points to next char in table
098 CA43 C24ECA              JNZ   :CA4E     If not identical
099 CA46 15                  DCR   D         Else: decr string length
100 CA47 C23DCA              JNZ   :CA3D     Get evt. next byte to check
101 CA4A E3                  XTHL            cancell PUSH B
102 CA4B E1                  POP   H
103 CA4C 37                  STC             CY=1
104 CA4D C9                  RET
105
106                     * If strings not identical:
107
108 CA4E 7A          LKC30   MOV   A,D       Get string length
109 CA4F 83                  ADD   E         Add 2
110 CA50 CD30DE              CALL  :DE30     Add A to HL; HL points now
111                                         to next string in table.
112 CA53 C1                  POP   B         Restore C=1
113 CA54 C337CA              JMP   :CA37     Start check on next string
114                     *
115                     ******************
116                     * TABLE LOOK UP *
117                     ******************
118                     *
119                     * Finds an entry in a look-up table.
120                     * LOOK used for symboltable, LOOKX for table of
121                     * Basic functions (CFE6).
122                     *
123                     * Table format:
124                     * [type/length][name][type/length][info]
125                     *
```

(ERRATUM DYNAMIC 83-17 P 231)

```
                        * Entry: BC: Points to start name in input.
                        *        D:  Type/length of name in input
                        *            high nibble: type; low nibble: length.
                        *        HL: Startaddress look-up table.
                        * Exit:   If not found: CY=0:
                        *            HL points to 0 byte at table end.
                        *            ABCD preserved, E corrupted.
                        *        If found: CY=1:
                        *            HL points to T/L of entry found.
                        *            E indicates how manyth entry.
                        *            ABC preserved.
                        *
CA57 2AA102     LOOK    LHLD  :02A1     Get startaddr symtab
                *
CA5A 37         LOOKX   STC              CY=1
CA5B F5                 PUSH  PSW
CA5C C5                 PUSH  B
CA5D 1EFF               MVI   E,:FF
CA5F 1C         LK10    INR   E          Entry count
CA60 7E                 MOV   A,M        Get T/L name in table
CA61 B7                 ORA   A
CA62 CA8BCA             JZ    :CA8B      Abort if end table reached
CA65 BA                 CMP   D          Compare with wanted T/L
CA66 CA6FCA             JZ    :CA6F      Jump if found
CA69 CDAECA     LK15    CALL  :CAAE      Calc addr next entry
CA6C C35FCA             JMP   :CA5F      Check next entry

                * If T/L of name O.K.:

CA6F C1         LK20    POP   B
CA70 C5                 PUSH  B
CA71 48                 MOV   C,B
CA72 D5                 PUSH  D
CA73 7A                 MOV   A,D        Get wanted T/L of name
CA74 E60F               ANI   :0F        Length name only
CA76 57                 MOV   D,A        in D
CA77 E5                 PUSH  H          Save begin table entry
CA78 23         LK30    INX   H
CA79 CDE0DD             CALL  :DDE0      Get char from line
CA7C BE                 CMP   M          Compare char of name
CA7D C28FCA             JNZ   :CA8F      If not correct name

                * If char. identical:

CA80 0C                 INR   C          Points to next char
CA81 15                 DCR   D          Decr length
CA82 C278CA             JNZ   :CA78      Check next char if not
                                         ready
CA85 23                 INX   H          Points after name in table
CA86 D1                 POP   D
CA87 D1                 POP   D
CA88 C1                 POP   B
CA89 F1                 POP   PSW        CY=1: Entry found
CA8A C9                 RET

                * If end of look-up table reached:

CA8B C1         LK40    POP   B
CA8C F1                 POP   PSW
CA8D 3F                 CMC              CY=0: No entry found
CA8E C9                 RET
```

```
188                     * If characters not identical:
189                     *
190 CA8F E1             LK50    POP   H
191 CA90 D1                     POP   D
192 CA91 7A                     MOV   A,D       Length in A
193 CA92 C369CA                 JMP   :CA69     Look further
194                     *
195                     ****************************************
196                     * FIND A VARIABLE IN THE SYMBOLTABLE *
197                     ****************************************
198                     *
199                     * Routine skips through successive symtab entries
200                     * from beginning till past the place pointed by HL.
201                     *
202                     * Entry: HL points to 1st byte required variable.
203                     * Exit:  HL points to (if found) or past (if not
204                     *        found) address required in symbol table.
205                     *        AFBCDE preserved.
206                     *
207 CA95 F5             FNAME   PUSH  PSW
208 CA96 D5                     PUSH  D
209 CA97 EB                     XCHG            Reqd addr in DE
210 CA98 2AA102                 LHLD  :02A1     Get startaddr symtab
211 CA9B E5             FNM10   PUSH  H
212 CA9C CDAECA                 CALL  :CAAE     Calc addr next variable
213 CA9F CD14DE                 CALL  :DE14     Reqd variable reached ?
214 CAA2 D2AACA                 JNC   :CAAA     Quit if true
215 CAA5 33                     INX   SP        ) Cancel Push H
216 CAA6 33                     INX   SP        )
217 CAA7 C39BCA                 JMP   :CA9B     Skip next variable
218 CAAA E1             FNM20   POP   H
219 CAAB D1                     POP   D
220 CAAC F1                     POP   PSW
221 CAAD C9                     RET
222                     *
223                     ******************************************************
224                     * CALCULATE ADDRESS NEXT VARIABLE IN SYMBOLTABLE *
225                     ******************************************************
226                     *
227                     * Adds length of name of variable + length of value
228                     * of variable to beginaddress.
229                     *
230                     * DADD: variable = length/name/length/value.
231                     * DADR: variable = length/name.
232                     *
233                     * Entry: HL points to 1st byte of current variable.
234                     * Exit:  HL points to next variable in symtab.
235                     *
236 CAAE CDB1CA    DADD    CALL  :CAB1     Add length name to HL
237 CAB1 7E        DADR    MOV   A,M       Get info T/L byte
238 CAB2 23                INX   H         Add 1
239 CAB3 E60F              ANI   :0F       Length only
240 CAB5 C330DE            JMP   :DE30     Add length info to HL
241                     *
242                     ************************************************
243                     * INSERT A VARIABLE NAME IN THE SYMBOL TABLE *
244                     ************************************************
245                     *
246                     * Entry: HL points to end symtab.
247                     *        B  points to start of name in input.
248                     *        E  number of bytes of info to reserve.
249                     *        D  T/L byte of name.
```

```
250                     * Exit:  HL points to info T/L byte.
251                     *        AFBCDE preserved.
252                     *
253 CAB8 F5             LOOKI   PUSH  PSW
254 CAB9 C5                     PUSH  B
255 CABA 48                     MOV   C,B       Input pos. in C
256 CABB D5                     PUSH  D
257 CABC D5                     PUSH  D
258 CABD E5                     PUSH  H
259 CABE 7A                     MOV   A,D       T/L name in A
260 CABF E60F                   ANI   :0F       Name length only
261 CAC1 83                     ADD   E         Add length info
262 CAC2 3C                     INR   A
263 CAC3 3C                     INR   A         +2 (length new entry)
264 CAC4 CD30DE                 CALL  :DE30     Calc new end symtab -1
265 CAC7 EB                     XCHG
266 CAC8 2AA502                 LHLD  :02A5     Get bottom screen RAM
267 CACB EB                     XCHG
268 CACC CD14DE                 CALL  :DE14     Compare DE-HL
269 CACF 3E1B                   MVI   A,:1B
270 CAD1 D2F5D9                 JNC   :D9F5     Run 'OUT OF MEMORY' if
271                                             not sufficient free RAM
272 CAD4 3600                   MVI   M,:00     New 'end table' flag
273 CAD6 23                     INX   H         HL is new end symtab
274 CAD7 22A302                 SHLD  :02A3     Store end symtab
275 CADA E1                     POP   H         Get old end symtab
276 CADB D1                     POP   D         Get T/L info
277 CADC 72                     MOV   M,D       Into symtab
278 CADD 23                     INX   H
279 CADE 7A                     MOV   A,D
280 CADF E670                   ANI   :70       Get type only
281 CAE1 B3                     ORA   E         Set low nibble for length
282 CAE2 5F                     MOV   E,A
283 CAE3 7A                     MOV   A,D       Get length name
284 CAE4 E60F                   ANI   :0F       Max. 15 bytes
285 CAE6 57                     MOV   D,A       Length in D for count
286 CAE7 CDE0DD         LKI10   CALL  :DDE0     Get char from line
287 CAEA 77                     MOV   M,A       Char into symtab
288 CAEB 0C                     INR   C         Pnts to next char on line
289 CAEC 23                     INX   H         Next pos in symtab
290 CAED 15                     DCR   D         Decr length name
291 CAEE C2E7CA                 JNZ   :CAE7     Next char if not ready
292 CAF1 73                     MOV   M,E       Info T/L into symtab
293 CAF2 D1                     POP   D
294 CAF3 C1                     POP   B
295 CAF4 F1                     POP   PSW
296 CAF5 C9                     RET
297                     *
298                     ************************************
299                     * FIND LINENUMBER IN TEXTBUFFER *
300                     ************************************
301                     *
302                     * Entry: HL: requested linenumber.
303                     * Exit:  ABCDE preserved. F corrupted.
304                     *        CY=1: Linenr found:
305                     *              HL points to address textline.
306                     *        CY=0: Not found:
307                     *              Z=0: HL points to address textline
308                     *                   with next higher linenumber.
309                     *              Z=1: End of textbuffer reached.
310                     *
311 CAF6 C5             FINDL   PUSH  B
```

```
312 CAF7 F5              PUSH  PSW
313 CAF8 D5              PUSH  D
314 CAF9 0600            MVI   B,:00
315 CAFB EB              XCHG              Req. linenr in DE
316 CAFC 2A9F02          LHLD  :029F       Get startaddr textbuf
317 CAFF 48        FDL05 MOV   C,B         Length prev. instr in C
318 CB00 0600            MVI   B,:00
319 CB02 09              DAD   B           Add this length to beginaddr
320 CB03 46        FDL10 MOV   B,M         Get length current instr
321 CB04 78              MOV   A,B         in A
322 CB05 B7              ORA   A
323 CB06 23              INX   H           Pnts to hibyte linenr
324 CB07 CA1DCB          JZ    :CB1D       Abort if at end textbuf
325 CB0A 7A              MOV   A,D         Get hibyte reqd linenr.
326 CB0B BE              CMP   M           Test high order bits
327 CB0C DA1CCB          JC    :CB1C       Abort if reqd nr lower than
328                                        current one (nr > reqd)
329 CB0F C2FFCA          JNZ   :CAFF       Next textline (nr < reqd)
330 CB12 23              INX   H           Pnts to lobyte linenr
331 CB13 7B              MOV   A,E         Get lobyte reqd linenr
332 CB14 BE              CMP   M
333 CB15 2B              DCX   H
334 CB16 DA1CCB          JC    :CB1C       Abort if reqd nr lower than
335                                        current one (nr > reqd)
336 CB19 C2FFCA          JNZ   :CAFF       Next textline if not found
337                                        (nr < reqd)
338 CB1C 3F        FDL20 CMC               Line found: CY=1
339 CB1D 2B        FDL30 DCX   H
340 CB1E D1              POP   D
341 CB1F C1              POP   B
342 CB20 78              MOV   A,B
343 CB21 C1              POP   B
344 CB22 C9              RET
345                 *
346                 ********************************
347                 * EMPTY SYMBOLTABLE AND HEAP *
348                 ********************************
349                 *
350                 * Zeroes all variables, all pointers in the symtab,
351                 * kill all arrays, strings or stringarrays (zero the
352                 * pointers) referenced by the symtab by setting the
353                 * msb of the sizebit =1. Basic program is moved to a
354                 * location corresponding to the Heapsize.
355                 *
356                 * Exit: All registers preserved.
357                 *
358 CB23 F5         SCRATC PUSH  PSW
359 CB24 C5                PUSH  B
360 CB25 D5                PUSH  D
361 CB26 E5                PUSH  H
362 CB27 2AA102           LHLD  :02A1       Get startaddr symtab
363 CB2A 7E        SCT10  MOV   A,M         ) get name length
364 CB2B E60F             ANI   :0F         )
365 CB2D CA53CB           JZ    :CB53       Abort if at end symtab
366 CB30 CDB1CA           CALL  :CAB1       HL pnts to info length byte
367 CB33 7E               MOV   A,M         ) Get type
368 CB34 E6F0             ANI   :F0         )
369 CB36 FE40             CPI   :40
370 CB38 D24DCB           JNC   :CB4D       If array
371 CB3B 23               INX   H
372 CB3C FE20             CPI   :20
373 CB3E CA47CB           JZ    :CB47       If string
```

```
374
375                     * If numeric variable:
376
377 CB41 CD9ECB               CALL  :CB9E     Set value is 0 (4 bytes)
378 CB44 C32ACB               JMP   :CB2A     Next entry
379
380                     * If string variable:
381
382 CB47 CDA8CB       SCT30   CALL  :CBA8     Erase string reference in
383                                             symbtab and Heap
384 CB4A C32ACB               JMP   :CB2A     Next entry
385
386                     * If array:
387
388 CB4D CD5BCB       SCT40   CALL  :CB5B     Erase array
389 CB50 C32ACB               JMP   :CB2A     Next entry
390
391                     * If ready:
392
393 CB53 CDCADE       SCT90   CALL  :DECA     Organise HEAP + buffers
394 CB56 E1           LC188   POP   H
395 CB57 D1                   POP   D
396 CB58 C1                   POP   B
397 CB59 F1                   POP   PSW
398 CB5A C9                   RET
399                     *
400                     ***************
401                     * ERASE ARRAY *
402                     ***************
403                     *
404                     * The pointer is zeroed, the array size is erased
405                     * (msb=1). For stringarrays: zeroes all pointers
406                     * in the array and erase the string in the Heap.
407                     *
408                     * Entry: HL points to T/L byte of info after a
409                     *           symtab name of an array.
410                     * Exit:  HL points to next symtab entry.
411                     *        AFBCDE preserved.
412                     *
413 CB5B D5           EARRAY  PUSH  D
414 CB5C C5                   PUSH  B
415 CB5D F5                   PUSH  PSW
416 CB5E 7E                   MOV   A,M       Get type info
417 CB5F E630                 ANI   :30
418 CB61 F5                   PUSH  PSW       Save type only
419 CB62 CD51CE               CALL  :CE51     ) Get addr of array in
420 CB65 23                   INX   H         ) Heap in DE,
421 CB66 56                   MOV   D,M       ) Kill pointer in the
422 CB67 3600                 MVI   M,:00     ) symboltable
423 CB69 23                   INX   H         Pnts after symtab entry
424 CB6A 7A                   MOV   A,D
425 CB6B B3                   ORA   E
426 CB6C CA99CB               JZ    :CB99     Abort if entry was already 0
427 CB6F EB                   XCHG            Arrayaddr in HL
428 CB70 2B                   DCX   H
429 CB71 2B                   DCX   H         Pnts to 1st byte Heap entry
430 CB72 46                   MOV   B,M       Get 1st byte
431 CB73 CD36D2               CALL  :D236     Clear heap entry by msb=1
432 CB76 F1                   POP   PSW       Get type info
433 CB77 F5                   PUSH  PSW
434 CB78 FE20                 CPI   :20       String ?
435 CB7A D5                   PUSH  D         Save stringpointer
```

```
436 CB7B C298CB              JNZ   :CB98     Abort if not string
437
438                  * If string array:
439
440 CB7E 23                   INX   H
441 CB7F 4E                   MOV   C,M       Get length Heap entry
442 CB80 23                   INX   H
443 CB81 5E                   MOV   E,M       Get dimension
444 CB82 1C                   INR   E
445 CB83 7B                   MOV   A,E
446 CB84 CD30DE               CALL  :DE30     Calc beginaddr stringpntrs
447 CB87 79                   MOV   A,C
448 CB88 93                   SUB   E         Calc length pntr area
449 CB89 4F                   MOV   C,A       in C
450 CB8A D28ECB               JNC   :CB8E
451 CB8D 05                   DCR   B
452 CB8E CDA8CB      EAR10    CALL  :CBA8     Erase stringreference in
453                                            symtab and Heap
454 CB91 0B                   DCX   B
455 CB92 0B                   DCX   B         Update length pntr area
456 CB93 78                   MOV   A,B
457 CB94 B1                   ORA   C
458 CB95 C28ECB               JNZ   :CB8E     Next string if not ready
459
460                  * If ready:
461
462 CB98 E1           EAR20   POP   H         Get symtab pntr
463 CB99 F1           EAR30   POP   PSW
464 CB9A F1                   POP   PSW
465 CB9B C1                   POP   B
466 CB9C D1                   POP   D
467 CB9D C9                   RET
468                  *
469                  ************************************************
470                  * CLEAR A NUMERIC VARIABLE IN THE SYMBOLTABLE *
471                  ************************************************
472                  *
473                  * Loads '0' into 4 consecutive memory locations.
474                  *
475                  * Entry: Startaddress in HL.
476                  * Exit:  A=0, HL points to next byte.
477                  *        BCDEF preserved.
478                  *
479 CB9E AF           ZFPINT  XRA   A
480 CB9F 77                   MOV   M,A
481 CBA0 23                   INX   H
482 CBA1 77                   MOV   M,A
483 CBA2 23                   INX   H
484 CBA3 77                   MOV   M,A
485 CBA4 23                   INX   H
486 CBA5 77                   MOV   M,A
487 CBA6 23                   INX   H
488 CBA7 C9                   RET
489                  *
490                  ********************************************
491                  * ERASE STRINGREFERENCE IN HEAP AND SYMTAB *
492                  ********************************************
493                  *
494                  * The pointer in the symtab is set to '0', the
495                  * msb of the sizebyte of the Heap entry is set
496                  * to 1.
497                  *
```

```
498                     * Entry: HL points to stringpointer in symtab.
499                     * Exit:  HL points after this pointer.
500                     *        DE stringpointer.
501                     *        BC preserved, AF corrupted
502                     *
503 CBA8 5E             RSVHL   MOV   E,M       ) Stringpointer
504 CBA9 3600                   MVI   M,:00     ) in DE and then
505 CBAB 23                     INX   H         ) erased.
506 CBAC 56                     MOV   D,M       )
507 CBAD 3600                   MVI   M,:00     )
508 CBAF 23                     INX   H
509 CBB0 E5                     PUSH  H
510 CBB1 2A9F02                 LHLD  :029F     Get startaddr textbuf
511 CBB4 EB                     XCHG            in DE; stringpntr in HL
512 CBB5 7C                     MOV   A,H
513 CBB6 B5                     ORA   L         Stringpntr is already 0 ?
514 CBB7 C414DE                 CNZ   :DE14     If not: test if end of
515                                             heap reached
516 CBBA DC87D1                 CC    :D187     If not: clear heap entry
517 CBBD E1                     POP   H
518 CBBE C9                     RET
519                     *
520                     *
521                     *
522 CBBF                        END
```

****
* SYMBOL TABLE *
****

ASKRM	CA01	ASR10	CA1B	ASR20	CA1E	ASR30	CA21
DADD	CAAE	DADR	CAB1	EAR10	CB8E	EAR20	CB98
EAR30	CB99	EARRAY	CB5B	EMSTP	CA25	FDL05	CAFF
FDL10	CB03	FDL20	CB1C	FDL30	CB1D	FINDL	CAF6
FNAME	CA95	FNM10	CA9B	FNM20	CAAA	LC188	CB56
LK10	CA5F	LK15	CA69	LK20	CA6F	LK30	CA78
LK40	CA8B	LK50	CA8F	LKC10	CA37	LKC20	CA3D
LKC30	CA4E	LKI10	CAE7	LOOK	CA57	LOOKC	CA34
LOOKI	CAB8	LOOKX	CA5A	RSVHL	CBA8	SCRATC	CB23
SCT10	CB2A	SCT30	CB47	SCT40	CB4D	SCT90	CB53
SMKRM	CA01	ZFPINT	CB9E				

```
002                             ORG   :CBBF
003                     *
004                     *
005                     *
006                     ****************************
007                     * STRINGS BASIC COMMANDS *
008                     ****************************
009                     *
010                     * The first byte of each string is a length byte.
011                     *
012                     * The first byte after the string is the 'type'
013                     * byte. It is used to compose the TOKEN of the
014                     * particular Basic command:
015                     *    type byte, ANI :3F, ORI :80, gives TOKEN.
016                     *
017                     * Commands with type bytes bit 7=1 can be executed
018                     * during a program run. If bit 6=1, commands are
019                     * valid as direct command.
020                     *
021                     * The address given is the location of the encoding
022                     * routine for this particular command. These
023                     * routines can be found in ROM bank 3.
024                     *
025                     CMDTB
026 CBBF 03             SNEW    DATA  :03
027 CBC0 4E                     DATA  :4E       N
028 CBC1 45                     DATA  :45       E
029 CBC2 57                     DATA  :57       W
030 CBC3 81                     DATA  :81
031 CBC4 69E3                   DBL   :E369
032                     *
033 CBC6 04             SCONT   DATA  :04
034 CBC7 43                     DATA  :43       C
035 CBC8 4F                     DATA  :4F       O
036 CBC9 4E                     DATA  :4E       N
037 CBCA 54                     DATA  :54       T
038 CBCB 82                     DATA  :82
039 CBCC 69E3                   DBL   :E369
040                     *
041 CBCE 04             SSTOP   DATA  :04
042 CBCF 53                     DATA  :53       S
043 CBD0 54                     DATA  :54       T
044 CBD1 4F                     DATA  :4F       O
045 CBD2 50                     DATA  :50       P
046 CBD3 43                     DATA  :43
047 CBD4 69E3                   DBL   :E369
048                     *
049 CBD6 03             SEND    DATA  :03
050 CBD7 45                     DATA  :45       E
051 CBD8 4E                     DATA  :4E       N
052 CBD9 44                     DATA  :44       D
053 CBDA 44                     DATA  :44
054 CBDB 69E3                   DBL   :E369
055                     *
056 CBDD 07             SREST   DATA  :07
057 CBDE 52                     DATA  :52       R
058 CBDF 45                     DATA  :45       E
059 CBE0 53                     DATA  :53       S
060 CBE1 54                     DATA  :54       T
061 CBE2 4F                     DATA  :4F       O
062 CBE3 52                     DATA  :52       R
063 CBE4 45                     DATA  :45       E
```

```
064 CBE5 C5                     DATA  :C5
065 CBE6 69E3                   DBL   :E369
066                     *
067 CBE8 06             SRET    DATA  :06
068 CBE9 52                     DATA  :52       R
069 CBEA 45                     DATA  :45       E
070 CBEB 54                     DATA  :54       T
071 CBEC 55                     DATA  :55       U
072 CBED 52                     DATA  :52       R
073 CBEE 4E                     DATA  :4E       N
074 CBEF 46                     DATA  :46
075 CBF0 69E3                   DBL   :E369
076                     *
077 CBF2 03             SRUN    DATA  :03
078 CBF3 52                     DATA  :52       R
079 CBF4 55                     DATA  :55       U
080 CBF5 4E                     DATA  :4E       N
081 CBF6 87                     DATA  :87
082 CBF7 95E2                   DBL   :E295
083                     *
084 CBF9 04             SGOTO   DATA  :04
085 CBFA 47                     DATA  :47       G
086 CBFB 4F                     DATA  :4F       O
087 CBFC 54                     DATA  :54       T
088 CBFD 4F                     DATA  :4F       O
089 CBFE 49                     DATA  :49
090 CBFF 6AE3                   DBL   :E36A
091                     *
092 CC01 05             SGOSUB  DATA  :05
093 CC02 47                     DATA  :47       G
094 CC03 4F                     DATA  :4F       O
095 CC04 53                     DATA  :53       S
096 CC05 55                     DATA  :55       U
097 CC06 42                     DATA  :42       B
098 CC07 4A                     DATA  :4A
099 CC08 6AE3                   DBL   :E36A
100                     *
101 CC0A 03             SIMP    DATA  :03
102 CC0B 49                     DATA  :49       I
103 CC0C 4D                     DATA  :4D       M
104 CC0D 50                     DATA  :50       P
105 CC0E B5                     DATA  :B5
106 CC0F 9FE2                   DBL   :E29F
107                     *
108 CC11 05             SSAVA   DATA  :05
109 CC12 53                     DATA  :53       S
110 CC13 41                     DATA  :41       A
111 CC14 56                     DATA  :56       V
112 CC15 45                     DATA  :45       E
113 CC16 41                     DATA  :41       A
114 CC17 F9                     DATA  :F9
115 CC18 A9E4                   DBL   :E4A9
116                     *
117 CC1A 05             SLODA   DATA  :05
118 CC1B 4C                     DATA  :4C       L
119 CC1C 4F                     DATA  :4F       O
120 CC1D 41                     DATA  :41       A
121 CC1E 44                     DATA  :44       D
122 CC1F 41                     DATA  :41       A
123 CC20 FA                     DATA  :FA
124 CC21 A9E4                   DBL   :E4A9
125                     *
```

```
126 CC23 03          SOUT    DATA  :03
127 CC24 4F                  DATA  :4F       O
128 CC25 55                  DATA  :55       U
129 CC26 54                  DATA  :54       T
130 CC27 CE                  DATA  :CE
131 CC28 02E3                DBL   :E302
132                  *
133 CC2A 04          SPOKE   DATA  :04
134 CC2B 50                  DATA  :50       P
135 CC2C 4F                  DATA  :4F       O
136 CC2D 4B                  DATA  :4B       K
137 CC2E 45                  DATA  :45       E
138 CC2F CF                  DATA  :CF
139 CC30 02E3                DBL   :E302
140                  *
141 CC32 04          SWAIT   DATA  :04
142 CC33 57                  DATA  :57       W
143 CC34 41                  DATA  :41       A
144 CC35 49                  DATA  :49       I
145 CC36 54                  DATA  :54       T
146 CC37 D0                  DATA  :D0
147 CC38 59E2                DBL   :E259
148                  *
149 CC3A 04          SLIST   DATA  :04
150 CC3B 4C                  DATA  :4C       L
151 CC3C 49                  DATA  :49       I
152 CC3D 53                  DATA  :53       S
153 CC3E 54                  DATA  :54       T
154 CC3F D3                  DATA  :D3
155 CC40 28E2                DBL   :E228
156                  *
157 CC42 04          SEDIT   DATA  :04
158 CC43 45                  DATA  :45       E
159 CC44 44                  DATA  :44       D
160 CC45 49                  DATA  :49       I
161 CC46 54                  DATA  :54       T
162 CC47 B6                  DATA  :B6
163 CC48 28E2                DBL   :E228
164                  *
165 CC4A 05          SSOUND  DATA  :05
166 CC4B 53                  DATA  :53       S
167 CC4C 4F                  DATA  :4F       O
168 CC4D 55                  DATA  :55       U
169 CC4E 4E                  DATA  :4E       N
170 CC4F 44                  DATA  :44       D
171 CC50 D6                  DATA  :D6
172 CC51 17E3                DBL   :E317
173                  *
174 CC53 05          SNOISE  DATA  :05
175 CC54 4E                  DATA  :4E       N
176 CC55 4F                  DATA  :4F       O
177 CC56 49                  DATA  :49       I
178 CC57 53                  DATA  :53       S
179 CC58 45                  DATA  :45       E
180 CC59 D7                  DATA  :D7
181 CC5A 25E3                DBL   :E325
182                  *
183 CC5C 08          SENV    DATA  :08
184 CC5D 45                  DATA  :45       E
185 CC5E 4E                  DATA  :4E       N
186 CC5F 56                  DATA  :56       V
187 CC60 45                  DATA  :45       E
```

```
188 CC61 4C                     DATA  :4C       L
189 CC62 4F                     DATA  :4F       O
190 CC63 50                     DATA  :50       P
191 CC64 45                     DATA  :45       E
192 CC65 D8                     DATA  :D8
193 CC66 F6E1                   DBL   :E1F6
194                     *
195 CC68 06             SCURS   DATA  :06
196 CC69 43                     DATA  :43       C
197 CC6A 55                     DATA  :55       U
198 CC6B 52                     DATA  :52       R
199 CC6C 53                     DATA  :53       S
200 CC6D 4F                     DATA  :4F       O
201 CC6E 52                     DATA  :52       R
202 CC6F D9                     DATA  :D9
203 CC70 02E3                   DBL   :E302
204                     *
205 CC72 04             SMODE   DATA  :04
206 CC73 4D                     DATA  :4D       M
207 CC74 4F                     DATA  :4F       O
208 CC75 44                     DATA  :44       D
209 CC76 45                     DATA  :45       E
210 CC77 DA                     DATA  :DA
211 CC78 D1E1                   DBL   :E1D1
212                     *
213 CC7A 03             SDOT    DATA  :03
214 CC7B 44                     DATA  :44       D
215 CC7C 4F                     DATA  :4F       O
216 CC7D 54                     DATA  :54       T
217 CC7E DB                     DATA  :DB
218 CC7F 8FE2                   DBL   :E28F
219                     *
220 CC81 04             SDRAW   DATA  :04
221 CC82 44                     DATA  :44       D
222 CC83 52                     DATA  :52       R
223 CC84 41                     DATA  :41       A
224 CC85 57                     DATA  :57       W
225 CC86 DC                     DATA  :DC
226 CC87 8CE2                   DBL   :E28C
227                     *
228 CC89 04             SFILL   DATA  :04
229 CC8A 46                     DATA  :46       F
230 CC8B 49                     DATA  :49       I
231 CC8C 4C                     DATA  :4C       L
232 CC8D 4C                     DATA  :4C       L
233 CC8E DD                     DATA  :DD
234 CC8F 8CE2                   DBL   :E28C
235                     *
236 CC91 06             SCOLT   DATA  :06
237 CC92 43                     DATA  :43       C
238 CC93 4F                     DATA  :4F       O
239 CC94 4C                     DATA  :4C       L
240 CC95 4F                     DATA  :4F       O
241 CC96 52                     DATA  :52       R
242 CC97 54                     DATA  :54       T
243 CC98 DE                     DATA  :DE
244 CC99 0BE3                   DBL   :E30B
245                     *
246 CC9B 06             SCOLG   DATA  :06
247 CC9C 43                     DATA  :43       C
248 CC9D 4F                     DATA  :4F       O
249 CC9E 4C                     DATA  :4C       L
```

```
250 CC9F 4F                 DATA  :4F       O
251 CCA0 52                 DATA  :52       R
252 CCA1 47                 DATA  :47       G
253 CCA2 DF                 DATA  :DF
254 CCA3 0BE3               DBL   :E30B
255                 *
256 CCA5 05         SINPUT  DATA  :05
257 CCA6 49                 DATA  :49       I
258 CCA7 4E                 DATA  :4E       N
259 CCA8 50                 DATA  :50       P
260 CCA9 55                 DATA  :55       U
261 CCAA 54                 DATA  :54       T
262 CCAB 60                 DATA  :60
263 CCAC 15E1               DBL   :E115
264                 *
265 CCAE 04         SDATA   DATA  :04
266 CCAF 44                 DATA  :44       D
267 CCB0 41                 DATA  :41       A
268 CCB1 54                 DATA  :54       T
269 CCB2 41                 DATA  :41       A
270 CCB3 62                 DATA  :62
271 CCB4 A2E8               DBL   :E8A2
272                 *
273 CCB6 04         SREAD   DATA  :04
274 CCB7 52                 DATA  :52       R
275 CCB8 45                 DATA  :45       E
276 CCB9 41                 DATA  :41       A
277 CCBA 44                 DATA  :44       D
278 CCBB 63                 DATA  :63
279 CCBC 27E1               DBL   :E127
280                 *
281 CCBE 03         SLET    DATA  :03
282 CCBF 4C                 DATA  :4C       L
283 CCC0 45                 DATA  :45       E
284 CCC1 54                 DATA  :54       T
285 CCC2 E4                 DATA  :E4
286 CCC3 FEE0               DBL   :E0FE
287                 *
288 CCC5 02         SIF     DATA  :02
289 CCC6 49                 DATA  :49       I
290 CCC7 46                 DATA  :46       F
291 CCC8 66                 DATA  :66
292 CCC9 BCE0               DBL   :E0BC
293                 *
294 CCCB 03         SREM    DATA  :03
295 CCCC 52                 DATA  :52       R
296 CCCD 45                 DATA  :45       E
297 CCCE 4D                 DATA  :4D       M
298 CCCF 69                 DATA  :69
299 CCD0 66E3               DBL   :E366
300                 *
301 CCD2 03         SFOR    DATA  :03
302 CCD3 46                 DATA  :46       F
303 CCD4 4F                 DATA  :4F       O
304 CCD5 52                 DATA  :52       R
305 CCD6 EA                 DATA  :EA
306 CCD7 5FE0               DBL   :E05F
307                 *
308 CCD9 04         SNEXT   DATA  :04
309 CCDA 4E                 DATA  :4E       N
310 CCDB 45                 DATA  :45       E
311 CCDC 58                 DATA  :58       X
```

```
312 CCDD 54                     DATA  :54       T
313 CCDE EB                     DATA  :EB
314 CCDF A9E0                   DBL   :E0A9
315                     *
316 CCE1 05             SPRINT  DATA  :05
317 CCE2 50                     DATA  :50       P
318 CCE3 52                     DATA  :52       R
319 CCE4 49                     DATA  :49       I
320 CCE5 4E                     DATA  :4E       N
321 CCE6 54                     DATA  :54       T
322 CCE7 ED                     DATA  :ED
323 CCE8 9FE1                   DBL   :E19F
324                     *
325 CCEA 01             S?      DATA  :01
326 CCEB 3F                     DATA  :3F       ?  (abbr. for PRINT)
327 CCEC ED                     DATA  :ED
328 CCED 9FE1                   DBL   :E19F
329                     *
330 CCEF 02             SON     DATA  :02
331 CCF0 4F                     DATA  :4F       O
332 CCF1 4E                     DATA  :4E       N
333 CCF2 6E                     DATA  :6E
334 CCF3 76E1                   DBL   :E176
335                     *
336 CCF5 03             SDIM    DATA  :03
337 CCF6 44                     DATA  :44       D
338 CCF7 49                     DATA  :49       I
339 CCF8 4D                     DATA  :4D       M
340 CCF9 F0                     DATA  :F0
341 CCFA 66E1                   DBL   :E166
342                     *
343 CCFC 03                     DATA  :03
344 CCFD 2A                     DATA  :2A       *
345 CCFE 2A                     DATA  :2A       *
346 CCFF 2A                     DATA  :2A       *
347 CD00 71                     DATA  :71
348 CD01 66E3                   DBL   :E366
349                     *
350 CD03 02             SUT     DATA  :02
351 CD04 55                     DATA  :55       U
352 CD05 54                     DATA  :54       T
353 CD06 B2                     DATA  :B2
354 CD07 69E3                   DBL   :E369
355                     *
356 CD09 05             SCALM   DATA  :05
357 CD0A 43                     DATA  :43       C
358 CD0B 41                     DATA  :41       A
359 CD0C 4C                     DATA  :4C       L
360 CD0D 4C                     DATA  :4C       L
361 CD0E 4D                     DATA  :4D       M
362 CD0F F3                     DATA  :F3
363 CD10 44E3                   DBL   :E344
364                     *
365 CD12 05             SCLEAR  DATA  :05
366 CD13 43                     DATA  :43       C
367 CD14 4C                     DATA  :4C       L
368 CD15 45                     DATA  :45       E
369 CD16 41                     DATA  :41       A
370 CD17 52                     DATA  :52       R
371 CD18 F4                     DATA  :F4
372 CD19 14E3                   DBL   :E314
373                     *
```

```
374 CD1B 04          SLOAD   DATA  :04
375 CD1C 4C                  DATA  :4C       L
376 CD1D 4F                  DATA  :4F       O
377 CD1E 41                  DATA  :41       A
378 CD1F 44                  DATA  :44       D
379 CD20 CB                  DATA  :CB
380 CD21 55E3                DBL   :E355
381                  *
382 CD23 04          SSAVE   DATA  :04
383 CD24 53                  DATA  :53       S
384 CD25 41                  DATA  :41       A
385 CD26 56                  DATA  :56       V
386 CD27 45                  DATA  :45       E
387 CD28 8C                  DATA  :8C
388 CD29 55E3                DBL   :E355
389                  *
390 CD2B 05          SCHECK  DATA  :05
391 CD2C 43                  DATA  :43       C
392 CD2D 48                  DATA  :48       H
393 CD2E 45                  DATA  :45       E
394 CD2F 43                  DATA  :43       C
395 CD30 4B                  DATA  :4B       K
396 CD31 8D                  DATA  :8D
397 CD32 69E3                DBL   :E369
398                  *
399 CD34 05                  DATA  :05       (Cancelled instr.)
400 CD35 00                  DATA  :00
401 CD36 52                  DATA  :52       R
402 CD37 41                  DATA  :41       A
403 CD38 53                  DATA  :53       S
404 CD39 45                  DATA  :45       E
405 CD3A FB                  DATA  :FB
406 CD3B 73E8                DBL   :E873
407                  *
408 CD3D 04          SSTEP   DATA  :04
409 CD3E 53                  DATA  :53       S
410 CD3F 54                  DATA  :54       T
411 CD40 45                  DATA  :45       E
412 CD41 50                  DATA  :50       P
413 CD42 BC                  DATA  :BC
414 CD43 69E3                DBL   :E369
415                  *
416 CD45 04          STRON   DATA  :04
417 CD46 54                  DATA  :54       T
418 CD47 52                  DATA  :52       R
419 CD48 4F                  DATA  :4F       O
420 CD49 4E                  DATA  :4E       N
421 CD4A FD                  DATA  :FD
422 CD4B 69E3                DBL   :E369
423                  *
424 CD4D 05          STROF   DATA  :05
425 CD4E 54                  DATA  :54       T
426 CD4F 52                  DATA  :52       R
427 CD50 4F                  DATA  :4F       O
428 CD51 46                  DATA  :46       F
429 CD52 46                  DATA  :46       F
430 CD53 FE                  DATA  :FE
431 CD54 69E3                DBL   :E369
432                  *
433 CD56 04          STALK   DATA  :04
434 CD57 54                  DATA  :54       T
435 CD58 41                  DATA  :41       A
```

```
436 CD59 4C                    DATA  :4C       L
437 CD5A 4B                    DATA  :4B       K
438 CD5B FB                    DATA  :FB
439 CD5C 14E3                  DBL   :E314
440                    *
441 CD5E 00                    DATA  :00       End of table
442 CD5F E5                    DATA  :E5       Assignment (= LET)
443 CD60 FEE0                  DBL   :E0FE
444                    *
445 CD62 FF                    DATA  :FF
446 CD63 FF                    DATA  :FF
447                    *
448                    **********************
449                    * RUN basiccmd TALK *
450                    **********************
451                    *
452 CD64 CDF8E6        RTALK   CALL  :E6F8     (0) Get addr parameter
453                                            block in HL
454 CD67 7E            RTK10   MOV   A,M       Get code
455 CD68 23                    INX   H
456 CD69 B7                    ORA   A
457 CD6A F8                    RM              Ready if code = FF (end)
458 CD6B FE05                  CPI   :05
459 CD6D DA45CE                JC    :CE45     Jump if freq. code
460 CD70 FE0C                  CPI   :0C
461 CD72 DA94EE                JC    :EE94     (0) Jump if volume code
462 CD75 C36DEC                JMP   :EC6D     (0) Continu
463                    *
464                    ****************
465                    * STRING DATA *
466                    ****************
467                    *
468 CD78 08            LC236   DATA  :08
469 CD79 57                    DATA  :57       W
470 CD7A 41                    DATA  :41       A
471 CD7B 49                    DATA  :49       I
472 CD7C 54                    DATA  :54       T
473 CD7D 20                    DATA  :20
474 CD7E 4D                    DATA  :4D       M
475 CD7F 45                    DATA  :45       E
476 CD80 4D                    DATA  :4D       M
477                    *
478 CD81 09            LC237   DATA  :09
479 CD82 57                    DATA  :57       W
480 CD83 41                    DATA  :41       A
481 CD84 49                    DATA  :49       I
482 CD85 54                    DATA  :54       T
483 CD86 20                    DATA  :20
484 CD87 54                    DATA  :54       T
485 CD88 49                    DATA  :49       I
486 CD89 4D                    DATA  :4D       M
487 CD8A 45                    DATA  :45       E
488                    *
489                    *
490                    *
491 CD8B                       END

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

CMDTB  CBBF    LC236  CD78    LC237  CD81    RTALK  CD64
```

RTK10	CD67	S?	CCEA	SCALM	CD09	SCHECK	CD2B
SCLEAR	CD12	SCOLG	CC9B	SCOLT	CC91	SCONT	CBC6
SCURS	CC68	SDATA	CCAE	SDIM	CCF5	SDOT	CC7A
SDRAW	CC81	SEDIT	CC42	SEND	CBD6	SENV	CC5C
SFILL	CC89	SFOR	CCD2	SGOSUB	CC01	SGOTO	CBF9
SIF	CCC5	SIMP	CC0A	SINPUT	CCA5	SLET	CCBE
SLIST	CC3A	SLOAD	CD1B	SLODA	CC1A	SMODE	CC72
SNEW	CBBF	SNEXT	CCD9	SNOISE	CC53	SON	CCEF
SOUT	CC23	SPOKE	CC2A	SPRINT	CCE1	SREAD	CCB6
SREM	CCCB	SREST	CBDD	SRET	CBE8	SRUN	CBF2
SSAVA	CC11	SSAVE	CD23	SSOUND	CC4A	SSTEP	CD3D
SSTOP	CBCE	STALK	CD56	STROF	CD4D	STRON	CD45
SUT	CD03	SWAIT	CC32				

```
002                             ORG   :CD8B
003                     *
004                     *
005                     *
006                     ****************************************
007                     * POINTERS TO STRINGS OF BASIC COMMANDS *
008                     ****************************************
009                     *
010                     * This table, with base at CC08, is used for
011                     * printing the Basic instructions during a
012                     * listing.
013                     *
014                     * From the TOKEN, the address in this table can
015                     * be found by:
016                     *       CC08 + 3x TOKEN = address in table
017                     * This is done by a routine on OECCC.
018                     *
019                     * The address given points to the memory location
020                     * on which the particular string can be found.
021                     *
022                     * The data byte after the address is an offset
023                     * with base at OECF8. Therewith the instructions
024                     * can be found about what to print after the
025                     * Basic statement (when performing LIST).
026                     *
027 CD8B BFCB        CDTAB   DBL   :CBBF     NEW
028 CD8D 00                  DATA  :00
029                     *
030 CD8E C6CB                DBL   :CBC6     CONT
031 CD90 00                  DATA  :00
032                     *
033 CD91 CECB                DBL   :CBCE     STOP
034 CD93 00                  DATA  :00
035                     *
036 CD94 D6CB                DBL   :CBD6     END
037 CD96 00                  DATA  :00
038                     *
039 CD97 DDCB                DBL   :CBDD     RESTORE
040 CD99 00                  DATA  :00
041                     *
042 CD9A E8CB                DBL   :CBE8     RETURN
043 CD9C 00                  DATA  :00
044                     *
045 CD9D F2CB                DBL   :CBF2     RUN
046 CD9F 00                  DATA  :00
047                     *
048 CDA0 F2CB                DBL   :CBF2     RUN
049 CDA2 01                  DATA  :01
050                     *
051 CDA3 F9CB                DBL   :CBF9     GOTO
052 CDA5 01                  DATA  :01
053                     *
054 CDA6 01CC                DBL   :CC01     GOSUB
055 CDA8 01                  DATA  :01
056                     *
057 CDA9 1BCD                DBL   :CD1B     LOAD
058 CDAB 04                  DATA  :04
059                     *
060 CDAC 23CD                DBL   :CD23     SAVE
061 CDAE 04                  DATA  :04
062                     *
063 CDAF 2BCD                DBL   :CD2B     CHECK
```

```
064 CDB1 00                    DATA  :00
065                  *
066 CDB2 23CC                  DBL   :CC23     OUT
067 CDB4 05                    DATA  :05
068                  *
069 CDB5 2ACC                  DBL   :CC2A     POKE
070 CDB7 05                    DATA  :05
071                  *
072 CDB8 32CC                  DBL   :CC32     WAIT
073 CDBA 0A                    DATA  :0A
074                  *
075 CDBB 78CD                  DBL   :CD78     WAIT MEM
076 CDBD 0A                    DATA  :0A
077                  *
078 CDBE 81CD                  DBL   :CD81     WAIT TIME
079 CDC0 04                    DATA  :04
080                  *
081 CDC1 3ACC                  DBL   :CC3A     LIST
082 CDC3 00                    DATA  :00
083                  *
084 CDC4 3ACC                  DBL   :CC3A     LIST
085 CDC6 01                    DATA  :01
086                  *
087 CDC7 3ACC                  DBL   :CC3A     LIST
088 CDC9 0B                    DATA  :0B
089                  *
090 CDCA 4ACC                  DBL   :CC4A     SOUND
091 CDCC 0C                    DATA  :0C
092                  *
093 CDCD 53CC                  DBL   :CC53     NOISE
094 CDCF 0D                    DATA  :0D
095                  *
096 CDD0 5CCC                  DBL   :CC5C     ENVELOPE
097 CDD2 0E                    DATA  :0E
098                  *
099 CDD3 68CC                  DBL   :CC68     CURSOR
100 CDD5 05                    DATA  :05
101                  *
102 CDD6 75CC                  DBL   :CC75     MODE
103 CDD8 0F                    DATA  :0F
104                  *
105 CDD9 7ACC                  DBL   :CC7A     DOT
106 CDDB 07                    DATA  :07
107                  *
108 CDDC 81CC                  DBL   :CC81     DRAW
109 CDDE 08                    DATA  :08
110                  *
111 CDDF 89CC                  DBL   :CC89     FILL
112 CDE1 08                    DATA  :08
113                  *
114 CDE2 91CC                  DBL   :CC91     COLORT
115 CDE4 09                    DATA  :09
116                  *
117 CDE5 9BCC                  DBL   :CC9B     COLORG
118 CDE7 09                    DATA  :09
119                  *
120 CDE8 A5CC                  DBL   :CCA5     INPUT
121 CDEA 11                    DATA  :11
122                  *
123 CDEB A5CC                  DBL   :CCA5     INPUT
124 CDED 10                    DATA  :10
125                  *
```

```
126 CDEE AECC                DBL   :CCAE     DATA
127 CDF0 03                  DATA  :03
128                  *
129 CDF1 B6CC                DBL   :CCB6     READ
130 CDF3 11                  DATA  :11
131                  *
132 CDF4 BECC                DBL   :CCBE     LET
133 CDF6 13                  DATA  :13
134                  *
135 CDF7 0000                DBL   :0000     Assignment (= LET)
136 CDF9 13                  DATA  :13
137                  *
138 CDFA C5CC                DBL   :CCC5     IF
139 CDFC 14                  DATA  :14
140                  *
141 CDFD C5CC                DBL   :CCC5     IF
142 CDFF 15                  DATA  :15
143                  *
144 CE00 C5CC                DBL   :CCC5     IF
145 CE02 16                  DATA  :16
146                  *
147 CE03 CBCC                DBL   :CCCB     REM
148 CE05 03                  DATA  :03
149                  *
150 CE06 D2CC                DBL   :CCD2     FOR
151 CE08 17                  DATA  :17
152                  *
153 CE09 D9CC                DBL   :CCD9     NEXT
154 CE0B 00                  DATA  :00
155                  *
156 CE0C D9CC                DBL   :CCD9     NEXT
157 CE0E 18                  DATA  :18
158                  *
159 CE0F E1CC                DBL   :CCE1     PRINT
160 CE11 19                  DATA  :19
161                  *
162 CE12 EFCC                DBL   :CCEF     ON
163 CE14 1A                  DATA  :1A
164                  *
165 CE15 EFCC                DBL   :CCEF     ON
166 CE17 1B                  DATA  :1B
167                  *
168 CE18 F5CC                DBL   :CCF5     DIM
169 CE1A 11                  DATA  :11
170                  *
171 CE1B FCCC                DBL   :CCFC     '***'
172 CE1D 03                  DATA  :03
173                  *
174 CE1E 03CD                DBL   :CD03     UT
175 CE20 00                  DATA  :00
176                  *
177 CE21 09CD                DBL   :CD09     CALLM
178 CE23 1C                  DATA  :1C
179                  *
180 CE24 12CD                DBL   :CD12     CLEAR
181 CE26 04                  DATA  :04
182                  *
183 CE27 0000                DBL   :0000     IMP
184 CE29 00                  DATA  :00
185                  *
186 CE2A 3ACC                DBL   :CC3A     LIST
187 CE2C 00                  DATA  :00
```

```
188                     *
189 CE2D 3ACC                  DBL   :CC3A     LIST
190 CE2F 01                    DATA  :01
191                     *
192 CE30 3ACC                  DBL   :CC3A     LIST
193 CE32 0B                    DATA  :0B
194                     *
195 CE33 11CC                  DBL   :CC11     SAVEA
196 CE35 1E                    DATA  :1E
197                     *
198 CE36 1ACC                  DBL   :CC1A     LOADA
199 CE38 1E                    DATA  :1E
200                     *
201 CE39 56CD                  DBL   :CD56     TALK
202 CE3B 04                    DATA  :04
203                     *
204 CE3C 3DCD                  DBL   :CD3D     STEP
205 CE3E 00                    DATA  :00
206                     *
207 CE3F 45CD                  DBL   :CD45     TRON
208 CE41 00                    DATA  :00
209                     *
210 CE42 4DCD                  DBL   :CD4D     TROFF
211 CE44 00                    DATA  :00
212                     *
213                     ******************************
214                     * part of Run 'TALK' (CD6D) *
215                     ******************************
216                     *
217                     * Set frequencies of channels 0,1 or 2.
218                     *
219 CE45 5F             RTK60   MOV   E,A       Code in E (=lobyte
220                                             osc. addr)
221 CE46 16FC                  MVI   D,:FC
222 CE48 7E                    MOV   A,M       Get 1st byte freq.code
223 CE49 12                    STAX  D         into osc.
224 CE4A 23                    INX   H
225 CE4B 7E                    MOV   A,M       Get 2nd byte freq.code
226 CE4C 12                    STAX  D         into osc.
227 CE4D C347EA                JMP   :EA47     (0) Handle next code
228                     *
229 CE50 FF                    DATA  :FF
230                     *
231                     ******************************
232                     * GET (M+1) IN E, ZERO M+1 *
233                     ******************************
234                     *
235                     * Part of EARRAY (CB5B).
236                     *
237                     * Entry: HL points to M.
238                     * Exit:  HL points to M+1. (M+1) in E.
239                     *        AFBCD preserved.
240                     *
241 CE51 23             MPT09   INX   H
242 CE52 5E                    MOV   E,M
243 CE53 3600                  MVI   M,:00
244 CE55 C9                    RET
245                     *
246                     ***************
247                     * STRING DATA *
248                     ***************
249                     *
```

```
CE56 05          MSPACE  DATA  :05
CE57 53                  DATA  :53        S
CE58 50                  DATA  :50        P
CE59 41                  DATA  :41        A
CE5A 43                  DATA  :43        C
CE5B 45                  DATA  :45        E
                 *
                 ****************************
                 * part of RUN DIM (OE639) *
                 ****************************
                 *
CE5C 2B          MPT41   DCX   H
CE5D C35BCB              JMP   :CB5B      Erase array if exists
                 *
                 ************************************
                 * GET TABNUMBER IN L, DOUTC IN A *
                 ************************************
                 *
                 * Part of Run 'TAB'.
                 *
CE60 CD1DE7      MPT50   CALL  :E71D      (0) Get nr of tabs in A
CE63 6F                  MOV   L,A       save it in L
CE64 3A3101              LDA   :0131     Get output direction
CE67 C9                  RET
                 *
                 ******************************************
                 * PRINT EXPRESSION FOLLOWED BY A SPACE *
                 ******************************************
                 *
                 * Entry SCHSP frequently used to print a space.
                 *
CE68 CDA2EE      LC230   CALL  :EEA2      (0) Print expression
CE6B 3E20        SCHSP   MVI   A,:20
CE6D C360DD              JMP   :DD60     Print space
                 *
                 *************
                 * PRINT ',' *
                 *************
                 *
                 * Entry: None.
                 * Exit:  FBCDEHL preserved.
                 *
CE70 3E2C        SCHCO   MVI   A,:2C
CE72 C360DD              JMP   :DD60     Print ','
                 *
                 ***********************************
                 * PRINT A STRING BETWEEN SPACES *
                 ***********************************
                 *
                 * Entry: Pointer to stringpointer on stack.
                 * Exit:  BC preserved. AFDEHL corrupted.
                 *
CE75 CD6BCE      STXSS   CALL  :CE6B      Print space
CE78 E3          STXTS   XTHL             Get stringpntr from stack
CE79 5E                  MOV   E,M       ) Store addr string in DE
CE7A 23                  INX   H         )
CE7B 56                  MOV   D,M       )
CE7C 23                  INX   H         )
CE7D E3                  XTHL            Addr after pntr on stack
CE7E EB                  XCHG            Addr string in HL
CE7F CD32DB              CALL  :DB32     Print string pointed by HL
CE82 C36BCE              JMP   :CE6B     Print space
```

```
312                     *
313                     ******************************
314                     * EDIT: PRINT TEXT COMPLETE *
315                     ******************************
316                     *
317 CE85 CD17EF         LC216   CALL  :EF17     (2) Print text complete
318 CE88 C338E1                 JMP   :E138     (2) Popall, ret
319                     *
320                     ***********************************
321                     * LIST ARRAY NAME - (not used) *
322                     ***********************************
323                     *
324 CE8B D5             LC217   PUSH  D
325 CE8C CDF7EE                 CALL  :EEF7     (0) List array name
326 CE8F D1                     POP   D
327 CE90 C9                     RET
328                     *
329                     ****************
330                     * part of C6BA *
331                     ****************
332                     *
333 CE91 E5             LC196   PUSH  H
334 CE92 C32DE9                 JMP   :E92D     (2) Now set up screen bits
335                                             for mode 1
336                     *
337                     **************
338                     * (not used) *
339                     **************
340                     *
341 CE95 3AA200         LC218   LDA   :00A2     Get startaddr edit buffer
342 CE98 C37CE9                 JMP   :E97C     (0)
343                     *
344                     ****************************
345                     * CONVERT MACC FOR OUTPUT *
346                     ****************************
347                     *
348                     * The MACC contents is converted from FPT to ASCII.
349                     *
350                     * Exit: AF corrupted, BCDEHL preserved.
351                     *
352 CE9B CD21C0         FBCP    CALL  :C021     Convert FPT nr for output
353 CE9E C5                     PUSH  B
354 CE9F 0601                   MVI   B,:01     Cannot trim last dec.place
355 CEA1 C365DB                 JMP   :DB65     Tidy up into external form
356                     *
357                     **********************
358                     * LIST CURRENT LINE *
359                     **********************
360                     *
361                     * Lists a program line if trace flag set.
362                     * Part of C8F5.
363                     *
364 CEA4 0B             MPT06   DCX   B
365 CEA5 CD55DD                 CALL  :DD55     Cursor to begin next line
366 CEA8 C3ABEC                 JMP   :ECAB     (0) List current line
367                     *
368 CEAB D1             LC219   POP   D         (Not used)
369 CEAC C9                     RET
370                     *
371                     ************************
372                     * part of SMKRM (CA01) *
373                     ************************
```

```
374                     *
375 CEAD D1             MPT07   POP   D          Return, CY=1
376 CEAE F1                     POP   PSW
377 CEAF 37                     STC
378 CEB0 C9                     RET
379                     *
380 CEB1 F1             MPT08   POP   PSW        Return, CY=0
381 CEB2 37                     STC
382 CEB3 3F                     CMC
383 CEB4 C9                     RET
384                     *
385                     ************************
386                     * CHANGE SCREEN MODE *
387                     ************************
388                     *
389                     * Part of Run 'MODE' (0E5BB).
390                     *
391                     * Entry: New mode in A.
392                     *
393 CEB5 EF             MPT40   RST   5          Change mode
394 CEB6 18                     DATA  :18
395 CEB7 DA10DA                 JC    :DA10      If insufficient memory:
396                                              error 'OUT OF MEMORY'.
397 CEBA C9                     RET
398                     *
399                     ********************************
400                     * part of RUN CLEAR (0E6B5) *
401                     ********************************
402                     *
403                     * Checks if more than 4 bytes are cleared.
404                     *
405                     * Entry: HL: Number of bytes to be cleared.
406                     *        F : flags on hibyte HL.
407                     *
408 CEBB 110400           MPT43   LXI   D,:0004    Must be at least 4 bytes
409 CEBE F414DE                   CP    :DE14      Compare HL-DE if not >32k
410 CEC1 DA15DA                   JC    :DA15      Run error 'NUMBER OUT
411                                              OF RANGE'if < 4.
412 CEC4 C9                     RET
413                     *
414 CEC5 FF                     DATA  :FF
415                     *
416                     ************************************
417                     * SET HEAP SIZE TO DEFAULT VALUE *
418                     ************************************
419                     *
420                     * Part of emergency stop routine (CA25).
421                     * Also runs a NEW command.
422                     *
423 CEC6 210001           HRNEW   LXI   H,:0100
424 CEC9 229D02                   SHLD  :029D      Store HEAP default value
425 CECC C3B5DE                   JMP   :DEB5      Run 'NEW'
426                     *
427                     ****************************************
428                     * RE-ORGANISE SCREEN AFTER 'DAI-PC' *
429                     ****************************************
430                     *
431                     * Lines after 'DAI PERSONAL COMPUTER' are set in
432                     * unit colour mode.
433                     * Set line mode byte to 7F and first char.
434                     * byte to 20, colour 00 during screen init.
435                     *
```

```
436                     * Entry: HL line mode byte of part. line.
437                     *
438 CECF 367F           MPT04   MVI   M,:7F     Wide char line contr byte
439 CED1 2B                     DCX   H
440 CED2 2B                     DCX   H
441 CED3 3620                   MVI   M,:20     Load space
442 CED5 2B                     DCX   H
443 CED6 3600                   MVI   M,:00     Colour no change
444 CED8 2B                     DCX   H
445 CED9 C9                     RET
446                     *
447                     ******************************************
448                     * part of RUN 'WAIT(TIME)' (DFD5/DFF7) *
449                     ******************************************
450                     *
451 CEDA 0B             REX1D   DCX   B
452 CEDB C31DE7                 JMP   :E71D     (0) Get value of argument
453                                             in A (max. FF)
454                     *
455                     ************************************************
456                     * EVALUATE ARGUMENTS IN NUMERIC EXPRESSION *
457                     ************************************************
458                     *
459                     * Not used.
460                     *
461 CEDE F5             LC231   PUSH  PSW
462 CEDF CD19E8                 CALL  :E819     (0) Evaluate arguments
463 CEE2 F1                     POP   PSW
464 CEE3 C9                     RET
465                     *
466                     ***************************************
467                     * SELECT ROM BANK 0; PRINT MESSAGE *
468                     ***************************************
469                     *
470                     * When CEE4 is called, the 2 bytes following
471                     * the CALL-instruction indicate the message to
472                     * be printed by the routine DAFF.
473                     *
474 CEE4 3A4000           SELB0   LDA   :0040     Get POROM
475 CEE7 E63F                     ANI   :3F       Select ROM bank 0
476 CEE9 324000                   STA   :0040     Set POROM
477 CEEC 3206FD                   STA   :FD06     and PORO
478 CEEF C3FFDA                   JMP   :DAFF     Print message
479                     *
480                     ********************************************
481                     * EDIT: RETURN FROM 'DELETE CHARACTER' *
482                     ********************************************
483                     *
484                     * Part of 2EFCC.
485                     *
486 CEF2 E1             LC232   POP   H
487 CEF3 CD30E3                 CALL  :E330     (2) Put cursor on screen
488 CEF6 C395EF                 JMP   :EF95     (2) Popall, ret, CY=1
489                     *
490                     ******************************************
491                     * PRINT 'COMPUTER' UNDER 'DAI PERSONAL' *
492                     ******************************************
493                     *
494                     * Part of RESET (C751).
495                     * During screen initialisation used to set a
496                     * new line mode byte between both parts of the
497                     * message. Line colour bytes are set for medium
```

```
498                     * resolution.
499                     *
500                     * Entry: HL: line mode byte to be changed.
501                     *         DE: offset for calculation next
502                     *             line mode byte.
503                     * Exit:   HL: next line mode byte.
504                     *
505 CEF9 365F            MPT03   MVI   M,:5F     Set line mode byte
506 CEFB 2B                      DCX   H
507 CEFC 3640                    MVI   M,:40     Set line colour byte
508 CEFE 23                      INX   H
509 CEFF 19                      DAD   D         Addr. next line mode byte
510 CF00 C9                      RET
511                     *
512 CF01 FF                      DATA  :FF
513                     *
514                     *
515                     *
516 CF02                         END
```

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

CDTAB	CD8B	FBCP	CE9B	HRNEW	CEC6	LC196	CE91
LC216	CE85	LC217	CE8B	LC218	CE95	LC219	CEAB
LC230	CE68	LC231	CEDE	LC232	CEF2	MPT03	CEF9
MPT04	CECF	MPT06	CEA4	MPT07	CEAD	MPT08	CEB1
MPT09	CE51	MPT40	CEB5	MPT41	CE5C	MPT43	CEBB
MPT50	CE60	MSPACE	CE56	REX1D	CEDA	RTK60	CE45
SCHCO	CE70	SCHSP	CE6B	SELBO	CEE4	STXSS	CE75
STXTS	CE78						

```
002                             ORG   :CF02
003                     *
004                     *
005                     *
006                     ****************************************
007                     * POINTERS TO ROUTINES BASICCOMMANDS *
008                     ****************************************
009                     *
010                     * This table, with base at CF00, gives the
011                     * addresses of the routines for execution of the
012                     * Basic statements.
013                     * The offset from baseaddress CF00 can be found
014                     * by adding 2x TOKEN to the base address.
015                     * Address indicates begin subroutine (ROM bank 0).
016                     *
017                     * The number given between brackets is the TOKEN.
018                     *
019 CF02 B5DE           CITAB   DBL   :DEB5     (81) NEW
020 CF04 D5DE                   DBL   :DED5     (82) CONT
021 CF06 03DF                   DBL   :DF03     (83) STOP
022 CF08 0CDF                   DBL   :DF0C     (84) END
023 CF0A 01E4                   DBL   :E401     (85) RESTORE
024 CF0C 4CDF                   DBL   :DF4C     (86) RETURN
025 CF0E 9EDF                   DBL   :DF9E     (87) RUN
026 CF10 BADF                   DBL   :DFBA     (88) RUN <linenumber>
027 CF12 63DF                   DBL   :DF63     (89) GOTO
028 CF14 2ADF                   DBL   :DF2A     (8A) GOSUB
029 CF16 70D2                   DBL   :D270     (8B) LOAD
030 CF18 3DD2                   DBL   :D23D     (8C) SAVE
031 CF1A C3D2                   DBL   :D2C3     (8D) CHECK
032 CF1C C9DF                   DBL   :DFC9     (8E) OUT
033 CF1E C0DF                   DBL   :DFC0     (8F) POKE
034 CF20 D5DF                   DBL   :DFD5     (90) WAIT
035 CF22 F7DF                   DBL   :DFF7     (91) WAIT MEM
036 CF24 16E0                   DBL   :E016     (92) WAIT TIME
037 CF26 97E1                   DBL   :E197     (93) LIST <whole program>
038 CF28 AAE1                   DBL   :E1AA     (94) LIST <linenumber>
039 CF2A B6E1                   DBL   :E1B6     (95) LIST <part of progr>
040 CF2C BCE4                   DBL   :E4BC     (96) SOUND
041 CF2E 0CE5                   DBL   :E50C     (97) NOISE
042 CF30 70E5                   DBL   :E570     (98) ENVELOPE
043 CF32 B2E5                   DBL   :E5B2     (99) CURSOR
044 CF34 BBE5                   DBL   :E5BB     (9A) MODE
045 CF36 C1E5                   DBL   :E5C1     (9B) DOT
046 CF38 CEE5                   DBL   :E5CE     (9C) DRAW
047 CF3A D7E5                   DBL   :E5D7     (9D) FILL
048 CF3C 0EE6                   DBL   :E60E     (9E) COLORT
049 CF3E 15E6                   DBL   :E615     (9F) COLORG
050 CF40 02E3                   DBL   :E302     (A0) INPUT
051 CF42 FCE2                   DBL   :E2FC     (A1) INPUT <with prompt>
052 CF44 8FE1                   DBL   :E18F     (A2) DATA
053 CF46 23E3                   DBL   :E323     (A3) READ
054 CF48 5AE4                   DBL   :E45A     (A4) LET
055 CF4A 5AE4                   DBL   :E45A     (A5) assignment
056 CF4C 20DF                   DBL   :DF20     (A6) IF THEN <statement>
057 CF4E 15DF                   DBL   :DF15     (A7) IF GOTO
058 CF50 15DF                   DBL   :DF15     (A8) IF THEN <linenumber>
059 CF52 8FE1                   DBL   :E18F     (A9) REM
060 CF54 2BE0                   DBL   :E02B     (AA) FOR .. TO
061 CF56 E5E0                   DBL   :E0E5     (AB) NEXT
062 CF58 C5E0                   DBL   :E0C5     (AC) NEXT <variable>
063 CF5A B3E2                   DBL   :E2B3     (AD) PRINT
```

```
064 CF5C 6ADF                DBL   :DF6A      (AE) ON GOTO
065 CF5E 71DF                DBL   :DF71      (AF) ON GOSUB
066 CF60 2FE6                DBL   :E62F      (B0) DIM
067 CF62 33DA                DBL   :DA33      (B1) *** (error line run)
068 CF64 9EE6                DBL   :E69E      (B2) UT
069 CF66 A4E6                DBL   :E6A4      (B3) CALLM
070 CF68 B5E6                DBL   :E6B5      (B4) CLEAR
071 CF6A 95E1                DBL   :E195      (B5) IMP
072 CF6C F5E1                DBL   :E1F5      (B6) EDIT <total>
073 CF6E 53E2                DBL   :E253      (B7) EDIT <linenumber>
074 CF70 5CE2                DBL   :E25C      (B8) EDIT <part>
075 CF72 1DD8                DBL   :D81D      (B9) SAVEA
076 CF74 5ED8                DBL   :D85E      (BA) LOADA
077 CF76 64CD                DBL   :CD64      (BB) TALK
078 CF78 FEDE                DBL   :DEFE      (BC) STEP
079 CF7A CEE6                DBL   :E6CE      (BD) TRON
080 CF7C D5E6                DBL   :E6D5      (BE) TROFF
081                     *
082                     ******************************
083                     * part of RUN 'ASC' (0EACF) *
084                     ******************************
085                     *
086 CF7E B7             MPT51  ORA   A         Nulstring ?
087 CF7F CA7CEB                 JZ    :EB7C    (0) Then 0 into MACC
088 CF82 23                     INX   H        Else:
089 CF83 C3C7EA                 JMP   :EAC7    (0) Next byte from MEM
090                                            into MACC
091                     *
092                     ********************************************
093                     * TABLE PREFIXES FOR UNITARY OPERATIONS *
094                     ********************************************
095                     *
096                     * Only used by LIST for decoding.
097                     * The prefix gives a code for unitary operators,
098                     * which on encoding are directly determined, not
099                     * looked up in a table.
100                     *
101                     * Format: length of name / name / 5-bit opcode
102                     *
103                     OPTBB
104 CF86 01             SBRA   DATA  :01
105 CF87 28                    DATA  :28       (
106 CF88 1A                    DATA  :1A
107                     *
108 CF89 01                    DATA  :01
109 CF8A 2D                    DATA  :2D       -
110 CF8B 1D                    DATA  :1D
111                     *
112 CF8C 01                    DATA  :01
113 CF8D 2B                    DATA  :2B       +
114 CF8E 1C                    DATA  :1C
115                     *
116 CF8F 00                    DATA  :00       Fixed FPT conversion
117 CF90 1F                    DATA  :1F
118                     *
119                     ***************************
120                     * TABLE BINARY OPERATORS *
121                     ***************************
122                     *
123                     * Format: 1 byte:  length of name.
124                     *         n bytes: name.
                                  1 byte:  code byte:
```

```
                        *                       highest 3 bits: priority.
                        *                       lowest  5 bits: opcode.
                        *
                        OPTAB
CF91 01                 SDIV    DATA  :01
CF92 2F                         DATA  :2F       /
CF93 C2                         DATA  :C2
                        *
CF94 01                         DATA  :01
CF95 2A                         DATA  :2A       *
CF96 C3                         DATA  :C3
                        *
CF97 03                         DATA  :03
CF98 4D                         DATA  :4D       M
CF99 4F                         DATA  :4F       O
CF9A 44                         DATA  :44       D
CF9B CF                         DATA  :CF
                        *
CF9C 01                         DATA  :01
CF9D 5E                         DATA  :5E       ^
CF9E E4                         DATA  :E4
                        *
CF9F 04                         DATA  :04
CFA0 49                         DATA  :49       I
CFA1 41                         DATA  :41       A
CFA2 4E                         DATA  :4E       N
CFA3 44                         DATA  :44       D
CFA4 69                         DATA  :69
                        *
CFA5 03                         DATA  :03
CFA6 49                         DATA  :49       I
CFA7 4F                         DATA  :4F       O
CFA8 52                         DATA  :52       R
CFA9 6A                         DATA  :6A
                        *
CFAA 04                         DATA  :04
CFAB 49                         DATA  :49       I
CFAC 58                         DATA  :58       X
CFAD 4F                         DATA  :4F       O
CFAE 52                         DATA  :52       R
CFAF 6C                         DATA  :6C
                        *
CFB0 03                         DATA  :03
CFB1 53                         DATA  :53       S
CFB2 48                         DATA  :48       H
CFB3 4C                         DATA  :4C       L
CFB4 8D                         DATA  :8D
                        *
CFB5 03                         DATA  :03
CFB6 53                         DATA  :53       S
CFB7 48                         DATA  :48       H
CFB8 52                         DATA  :52       R
CFB9 8E                         DATA  :8E
                        *
CFBA 02                         DATA  :02
CFBB 3E                         DATA  :3E       >
CFBC 3D                         DATA  :3D       =
CFBD 50                         DATA  :50
                        *
CFBE 01                         DATA  :01
CFBF 3E                         DATA  :3E       >
CFC0 51                         DATA  :51
```

```
188                     *
189 CFC1 02             *       DATA    :02
190 CFC2 3C             *       DATA    :3C     <
191 CFC3 3E             *       DATA    :3E     >
192 CFC4 52             *       DATA    :52
193                     *
194 CFC5 02             *       DATA    :02
195 CFC6 3C             *       DATA    :3C     <
196 CFC7 3D             *       DATA    :3D     =
197 CFC8 53             *       DATA    :53
198                     *
199 CFC9 01             *       DATA    :01
200 CFCA 3C             *       DATA    :3C     <
201 CFCB 54             *       DATA    :54
202                     *
203 CFCC 01             *       DATA    :01
204 CFCD 3D             *       DATA    :3D     >
205 CFCE 55             *       DATA    :55
206                     *
207 CFCF 03             *SAND   DATA    :03
208 CFD0 41             *       DATA    :41     A
209 CFD1 4E             *       DATA    :4E     N
210 CFD2 44             *       DATA    :44     D
211 CFD3 38             *       DATA    :38
212                     *
213 CFD4 02             *       DATA    :02
214 CFD5 4F             *       DATA    :4F     O
215 CFD6 52             *       DATA    :52     R
216 CFD7 39             *       DATA    :39
217                     *
218                     ******************************
219                     * TABLE UNITARY OPERATORS *
220                     ******************************
221                     *
222                     * Format: 1 byte:  Length of name.
223                     *         n bytes: Name.
224                     *         1 byte:  Code byte:
225                     *                  3 highest bits: priority.
226                     *                  5 lowest bits:  opcode.
227                     *
228 CFD8 04             *OPTBM  DATA    :04
229 CFD9 49             *       DATA    :49     I
230 CFDA 4E             *       DATA    :4E     N
231 CFDB 4F             *       DATA    :4F     O
232 CFDC 54             *       DATA    :54     T
233 CFDD 1E             *       DATA    :1E
234                     *
235 CFDE 01             *       DATA    :01
236 CFDF 2B             *       DATA    :2B     +
237 CFE0 A0             *       DATA    :A0
238                     *
239 CFE1 01             *       DATA    :01
240 CFE2 2D             *       DATA    :2D     -
241 CFE3 A1             *       DATA    :A1
242                     *
243 CFE4 00             *       DATA    :00     End of table
244 CFE5 00             *       DATA    :00
245                     *
246                     ***********************************
247                     * TABLE STRINGS BASIC FUNCTIONS *
248                     ***********************************
249                     *
```

```
                        * Format: 1 byte:  Length of name.
                        *         n bytes: Name.
                        *         1 byte:  High nibble: type of info.
                        *                  Low nibble: nr of arguments.
                        *         1 byte:  High nibble: required type
                        *                  of variable expected:
                        *                  (0=FPT, 1=INT, 2=STR).
                        *
                        FUNTB
CFE6 03                 SABS    DATA  :03
CFE7 41                         DATA  :41       A
CFE8 42                         DATA  :42       B
CFE9 53                         DATA  :53       S
CFEA 01                         DATA  :01
CFEB 00                         DATA  :00
                        *
CFEC 04                 SALOG   DATA  :04
CFED 41                         DATA  :41       A
CFEE 4C                         DATA  :4C       L
CFEF 4F                         DATA  :4F       O
CFF0 47                         DATA  :47       G
CFF1 01                         DATA  :01
CFF2 00                         DATA  :00
                        *
CFF3 03                 SASC    DATA  :03
CFF4 41                         DATA  :41       A
CFF5 53                         DATA  :53       S
CFF6 43                         DATA  :43       C
CFF7 11                         DATA  :11
CFF8 20                         DATA  :20
                        *
CFF9 04                 SCHR    DATA  :04
CFFA 43                         DATA  :43       C
CFFB 48                         DATA  :48       H
CFFC 52                         DATA  :52       R
CFFD 24                         DATA  :24       $
CFFE 21                         DATA  :21
CFFF 10                         DATA  :10
                        *
D000 04                 SCURX   DATA  :04
D001 43                         DATA  :43       C
D002 55                         DATA  :55       U
D003 52                         DATA  :52       R
D004 58                         DATA  :58       X
D005 10                         DATA  :10
                        *
D006 04                 SCURY   DATA  :04
D007 43                         DATA  :43       C
D008 55                         DATA  :55       U
D009 52                         DATA  :52       R
D00A 59                         DATA  :59       Y
D00B 10                         DATA  :10
                        *
D00C 03                 SEXP    DATA  :03
D00D 45                         DATA  :45       E
D00E 58                         DATA  :58       X
D00F 50                         DATA  :50       P
D010 01                         DATA  :01
D011 00                         DATA  :00
                        *
D012 04                 SFRAC   DATA  :04
D013 46                         DATA  :46       F
```

```
312 D014 52                     DATA  :52       R
313 D015 41                     DATA  :41       A
314 D016 43                     DATA  :43       C
315 D017 01                     DATA  :01
316 D018 00                     DATA  :00
317                     *
318 D019 03             SFRE    DATA  :03
319 D01A 46                     DATA  :46       F
320 D01B 52                     DATA  :52       R
321 D01C 45                     DATA  :45       E
322 D01D 10                     DATA  :10
323                     *
324 D01E 04             SFREQ   DATA  :04
325 D01F 46                     DATA  :46       F
326 D020 52                     DATA  :52       R
327 D021 45                     DATA  :45       E
328 D022 51                     DATA  :51       Q
329 D023 11                     DATA  :11
330 D024 00                     DATA  :00
331                     *
332 D025 04             SGETC   DATA  :04
333 D026 47                     DATA  :47       G
334 D027 45                     DATA  :45       E
335 D028 54                     DATA  :54       T
336 D029 43                     DATA  :43       C
337 D02A 10                     DATA  :10
338                     *
339 D02B 04             SHEX    DATA  :04
340 D02C 48                     DATA  :48       H
341 D02D 45                     DATA  :45       E
342 D02E 58                     DATA  :58       X
343 D02F 24                     DATA  :24       $
344 D030 21                     DATA  :21
345 D031 10                     DATA  :10
346                     *
347 D032 03             SINP    DATA  :03
348 D033 49                     DATA  :49       I
349 D034 4E                     DATA  :4E       N
350 D035 50                     DATA  :50       P
351 D036 11                     DATA  :11
352 D037 10                     DATA  :10
353                     *
354 D038 03             SINT    DATA  :03
355 D039 49                     DATA  :49       I
356 D03A 4E                     DATA  :4E       N
357 D03B 54                     DATA  :54       T
358 D03C 01                     DATA  :01
359 D03D 00                     DATA  :00
360                     *
361 D03E 05             SLEFT   DATA  :05
362 D03F 4C                     DATA  :4C       L
363 D040 45                     DATA  :45       E
364 D041 46                     DATA  :46       F
365 D042 54                     DATA  :54       T
366 D043 24                     DATA  :24       $
367 D044 22                     DATA  :22
368 D045 20                     DATA  :20
369 D046 10                     DATA  :10
370                     *
371 D047 03             SLEN    DATA  :03
372 D048 4C                     DATA  :4C       L
373 D049 45                     DATA  :45       E
```

```
374 D04A 4E                     DATA  :4E       N
375 D04B 11                     DATA  :11
376 D04C 20                     DATA  :20
377                     *
378 D04D 06             SVPT    DATA  :06
379 D04E 56                     DATA  :56       V
380 D04F 41                     DATA  :41       A
381 D050 52                     DATA  :52       R
382 D051 50                     DATA  :50       P
383 D052 54                     DATA  :54       T
384 D053 52                     DATA  :52       R
385 D054 11                     DATA  :11
386 D055 30                     DATA  :30
387                     *
388 D056 03             SLOG    DATA  :03
389 D057 4C                     DATA  :4C       L
390 D058 4F                     DATA  :4F       O
391 D059 47                     DATA  :47       G
392 D05A 01                     DATA  :01
393 D05B 00                     DATA  :00
394                     *
395 D05C 04             SLOGT   DATA  :04
396 D05D 4C                     DATA  :4C       L
397 D05E 4F                     DATA  :4F       O
398 D05F 47                     DATA  :47       G
399 D060 54                     DATA  :54       T
400 D061 01                     DATA  :01
401 D062 00                     DATA  :00
402                     *
403 D063 04             SXMAX   DATA  :04
404 D064 58                     DATA  :58       X
405 D065 4D                     DATA  :4D       M
406 D066 41                     DATA  :41       A
407 D067 58                     DATA  :58       X
408 D068 10                     DATA  :10
409                     *
410 D069 04             SYMAX   DATA  :04
411 D06A 59                     DATA  :59       Y
412 D06B 4D                     DATA  :4D       M
413 D06C 41                     DATA  :41       A
414 D06D 58                     DATA  :58       X
415 D06E 10                     DATA  :10
416                     *
417 D06F 04             SMID    DATA  :04
418 D070 4D                     DATA  :4D       M
419 D071 49                     DATA  :49       I
420 D072 44                     DATA  :44       D
421 D073 24                     DATA  :24       $
422 D074 23                     DATA  :23
423 D075 20                     DATA  :20
424 D076 10                     DATA  :10
425 D077 10                     DATA  :10
426                     *
427 D078 03             SPDL    DATA  :03
428 D079 50                     DATA  :50       P
429 D07A 44                     DATA  :44       D
430 D07B 4C                     DATA  :4C       L
431 D07C 11                     DATA  :11
432 D07D 10                     DATA  :10
433                     *
434 D07E 04             SPEEK   DATA  :04
435 D07F 50                     DATA  :50       P
```

```
436 D080 45                     DATA  :45       E
437 D081 45                     DATA  :45       E
438 D082 4B                     DATA  :4B       K
439 D083 11                     DATA  :11
440 D084 10                     DATA  :10
441                     *
442 D085 02             SPI     DATA  :02
443 D086 50                     DATA  :50       P
444 D087 49                     DATA  :49       I
445 D088 00                     DATA  :00
446                     *
447 D089 06             SRIGHT  DATA  :06
448 D08A 52                     DATA  :52       R
449 D08B 49                     DATA  :49       I
450 D08C 47                     DATA  :47       G
451 D08D 48                     DATA  :48       H
452 D08E 54                     DATA  :54       T
453 D08F 24                     DATA  :24       $
454 D090 22                     DATA  :22
455 D091 20                     DATA  :20
456 D092 10                     DATA  :10
457                     *
458 D093 03             SRND    DATA  :03
459 D094 52                     DATA  :52       R
460 D095 4E                     DATA  :4E       N
461 D096 44                     DATA  :44       D
462 D097 01                     DATA  :01
463 D098 00                     DATA  :00
464                     *
465 D099 04             SSCRN   DATA  :04
466 D09A 53                     DATA  :53       S
467 D09B 43                     DATA  :43       C
468 D09C 52                     DATA  :52       R
469 D09D 4E                     DATA  :4E       N
470 D09E 12                     DATA  :12
471 D09F 10                     DATA  :10
472 D0A0 10                     DATA  :10
473                     *
474 D0A1 03             SSGN    DATA  :03
475 D0A2 53                     DATA  :53       S
476 D0A3 47                     DATA  :47       G
477 D0A4 4E                     DATA  :4E       N
478 D0A5 01                     DATA  :01
479 D0A6 00                     DATA  :00
480                     *
481 D0A7 03             SSPC    DATA  :03
482 D0A8 53                     DATA  :53       S
483 D0A9 50                     DATA  :50       P
484 D0AA 43                     DATA  :43       C
485 D0AB 21                     DATA  :21
486 D0AC 10                     DATA  :10
487                     *
488 D0AD 03             SSQR    DATA  :03
489 D0AE 53                     DATA  :53       S
490 D0AF 51                     DATA  :51       Q
491 D0B0 52                     DATA  :52       R
492 D0B1 01                     DATA  :01
493 D0B2 00                     DATA  :00
494                     *
495 D0B3 04             SSTR    DATA  :04
496 D0B4 53                     DATA  :53       S
497 D0B5 54                     DATA  :54       T
```

```
498 D0B6 52                DATA  :52       R
499 D0B7 24                DATA  :24       $
500 D0B8 21                DATA  :21
501 D0B9 00                DATA  :00
502                *
503 D0BA 03        STAB    DATA  :03
504 D0BB 54                DATA  :54       T
505 D0BC 41                DATA  :41       A
506 D0BD 42                DATA  :42       B
507 D0BE 21                DATA  :21
508 D0BF 10                DATA  :10
509                *
510 D0C0 03        SVAL    DATA  :03
511 D0C1 56                DATA  :56       V
512 D0C2 41                DATA  :41       A
513 D0C3 4C                DATA  :4C       L
514 D0C4 01                DATA  :01
515 D0C5 20                DATA  :20
516                *
517 D0C6 03        SSIN    DATA  :03
518 D0C7 53                DATA  :53       S
519 D0C8 49                DATA  :49       I
520 D0C9 4E                DATA  :4E       N
521 D0CA 01                DATA  :01
522 D0CB 00                DATA  :00
523                *
524 D0CC 03        SCOS    DATA  :03
525 D0CD 43                DATA  :43       C
526 D0CE 4F                DATA  :4F       O
527 D0CF 53                DATA  :53       S
528 D0D0 01                DATA  :01
529 D0D1 00                DATA  :00
530                *
531 D0D2 03        STAN    DATA  :03
532 D0D3 54                DATA  :54       T
533 D0D4 41                DATA  :41       A
534 D0D5 4E                DATA  :4E       N
535 D0D6 01                DATA  :01
536 D0D7 00                DATA  :00
537                *
538 D0D8 04        SASIN   DATA  :04
539 D0D9 41                DATA  :41       A
540 D0DA 53                DATA  :53       S
541 D0DB 49                DATA  :49       I
542 D0DC 4E                DATA  :4E       N
543 D0DD 01                DATA  :01
544 D0DE 00                DATA  :00
545                *
546 D0DF 04        SACOS   DATA  :04
547 D0E0 41                DATA  :41       A
548 D0E1 43                DATA  :43       C
549 D0E2 4F                DATA  :4F       O
550 D0E3 53                DATA  :53       S
551 D0E4 01                DATA  :01
552 D0E5 00                DATA  :00
553                *
554 D0E6 03        SATN    DATA  :03
555 D0E7 41                DATA  :41       A
556 D0E8 54                DATA  :54       T
557 D0E9 4E                DATA  :4E       N
558 D0EA 01                DATA  :01
559 D0EB 00                DATA  :00
```

```
                        *
D0EC 00                 DATA  :00       End of table
                        *
                        ********
                        * DATA *
                        ********
                        *
                        ENDFT
D0ED 15                 FPOSC DATA  :15       Sound constant
D0EE F4                       DATA  :F4
D0EF 24                       DATA  :24
D0F0 00                       DATA  :00
                        *
D0F1 81                 FPM1  DATA  :81       FPT (-1)
D0F2 80                       DATA  :80
D0F3 00                       DATA  :00
D0F4 00                       DATA  :00
                        *
D0F5 02                 FPPI  DATA  :02       FPT (PI)
D0F6 C9                       DATA  :C9
D0F7 0F                       DATA  :0F
D0F8 DB                       DATA  :DB
                        *
D0F9 00                 I4    DATA  :00       INT (4)  (not used)
D0FA 00                       DATA  :00
D0FB 00                       DATA  :00
D0FC 04                       DATA  :04
                        *
D0FD 00                 IRAND DATA  :00       AND mask
D0FE FF                       DATA  :FF
D0FF FF                       DATA  :FF
D100 FF                       DATA  :FF
                        *
                        *
                        *
D101                          END
```

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

CITAB	CF02	ENDFT	D0ED	FPM1	D0F1	FPOSC	D0ED
FPPI	D0F5	FUNTB	CFE6	I4	D0F9	IRAND	D0FD
MPT51	CF7E	OPTAB	CF91	OPTBB	CF86	OPTBM	CFD8
SABS	CFE6	SACOS	D0DF	SALOG	CFEC	SAND	CFCF
SASC	CFF3	SASIN	D0D8	SATN	D0E6	SBRA	CF86
SCHR	CFF9	SCOS	D0CC	SCURX	D000	SCURY	D006
SDIV	CF91	SEXP	D00C	SFRAC	D012	SFRE	D019
SFREQ	D01E	SGETC	D025	SHEX	D02B	SINP	D032
SINT	D038	SLEFT	D03E	SLEN	D047	SLOG	D056
SLOGT	D05C	SMID	D06F	SPDL	D078	SPEEK	D07E
SPI	D085	SRIGHT	D089	SRND	D093	SSCRN	D099
SSGN	D0A1	SSIN	D0C6	SSPC	D0A7	SSQR	D0AD
SSTR	D0B3	STAB	D0BA	STAN	D0D2	SVAL	D0C0
SVPT	D04D	SXMAX	D063	SYMAX	D069		