

!A

LLOAD ROM2E.L,D1,A\$4000

*** End of Pass 1

LLOAD C0.L,A\$4000

LLOAD C4.L,A\$4000

LLOAD SW.L,A\$4000

LLOAD C8.L,A\$4000

LLOAD CC.L,A\$4000

LLOAD D0.L,D2,A\$4000

LLOAD D4.L,A\$4000

LLOAD D8.L,A\$4000

LLOAD DC.L,A\$4000

LLOAD E0.L,D2,A\$4000

LLOAD E4.L,A\$4000

LLOAD E8.L,A\$4000

LLOAD EA.L,A\$4000

LLOAD EC.L,A\$4000

LLOAD F0.L,D2,A\$4000

LLOAD F4.L,A\$4000

LLOAD F8.L,A\$4000

LLOAD FC.L,A\$4000

LLOAD ROM2E.L,D1,A\$4000

*** End of Pass 2

```
0800      1          ttl "ROM Source Code, ROM2E.L"
0800      2          src "ROM2E.L,D1"
0800      3      ;
0800      4      ;
0800      5      ; ROM2E.L
0800      6      ;
0800      7      ;
0800      8      ; ROM2E Source Code, READ Tape only, with Sweet 16, and
0800      9      ; new Garbage Collection (Cornelis Bongers concept)
0800     10      ;
0800     11      ; Build Version #12
0800     12      ;
0800     13      ; 2024 November 24
0800     14      ;
0800     15      ;
0800     16      ; DOS 4.5, Build 08
0800     17      ;
0800     18      ; 2024 November 24
0800     19      ;
0800     20      ;
0800     21      ; Start of Source Code: 0x4000
0800     22      ; Start of Symbol List: 0x7000
0800     23      ;
0800     24      ;
0800     25      ; Copyright (c) 2024 November 24 by
0800     26      ; Walland Philip Vrbancic Jr
0800     27      ;
0800     28      ; 6223 East Peabody Street
0800     29      ; Long Beach, California 90808
0800     30      ; Unitied States of America
0800     31      ;
0800     32      ; All Rights Reserved
0800     33      ;
0800     34      ; This software is the confidential and
0800     35      ; proprietary intellectual property of
0800     36      ; Walland Philip Vrbancic Jr
0800     37      ;
0800     38      ;
0000     39      LOC0      epz $00
0001     40      LOC1      epz $01
0002     41      LOC2      epz $02
0800     42      ;
0000     43      R0L      epz $00
0001     44      R0H      epz $01
0018     45      R12L     epz $18
0019     46      R12H     epz $19
001C     47      R14L     epz $1C
001D     48      R14H     epz $1D
001E     49      R15L     epz $1E
001F     50      R15H     epz $1F
0800     51      ;
0000     52      GOWARM    epz $00          ; 00:02, JMP RESTART
0003     53      GOSTROUT epz $03          ; 03:05, JMP STROUT
000A     54      GOUSR     epz $0A          ; 0A:0C, JMP <USER ADDR>
0090     55      JMPADRS   epz $90          ; 90:92, GETS JMP ...
0800     56      ;
000D     57      CHARAC    epz $0D
000E     58      ENDCHR    epz $0E
000F     59      EOLPTR     epz $0F
000F     60      NUMDIM     epz $0F
```

000F	61	TOKNCNTR	epz	\$0F	
0800	62	;			
0010	63	DIMFLG	epz	\$10	
0011	64	VALTYP	epz	\$11	; 11:12
0013	65	DATAFLG	epz	\$13	
0013	66	GARFLG	epz	\$13	
0014	67	SUBFLG	epz	\$14	
0015	68	INPUTFLG	epz	\$15	
0016	69	CPRMASK	epz	\$16	
0016	70	SIGNFLG	epz	\$16	
001A	71	SHAPE	epz	\$1A	; 1A:1B
001C	72	COLBITS	epz	\$1C	
001D	73	COLCOUNT	epz	\$1D	
0800	74	;			
0020	75	WNDLFT	epz	\$20	
0021	76	WNDWDTH	epz	\$21	
0022	77	WNDTOP	epz	\$22	
0023	78	WNDBTM	epz	\$23	
0024	79	CH	epz	\$24	
0025	80	CV	epz	\$25	
0026	81	GBASL	epz	\$26	
0027	82	GBASH	epz	\$27	
0028	83	BASL	epz	\$28	
0029	84	BASH	epz	\$29	
002A	85	BAS2L	epz	\$2A	
002B	86	BAS2H	epz	\$2B	
002C	87	H2	epz	\$2C	
002C	88	LMNEM	epz	\$2C	
002D	89	V2	epz	\$2D	
002D	90	RMNEM	epz	\$2D	
002E	91	CHKSUM	epz	\$2E	
002E	92	FORMAT	epz	\$2E	
002E	93	MASK	epz	\$2E	
002F	94	LENGTH	epz	\$2F	
002F	95	LASTIN	epz	\$2F	
002F	96	SIGN	epz	\$2F	
0800	97	;			
0030	98	COLOR	epz	\$30	
0030	99	HMASK	epz	\$30	
0031	100	MODE	epz	\$31	
0032	101	INVFLG	epz	\$32	
0033	102	PROMPT	epz	\$33	
0034	103	YSAV	epz	\$34	
0035	104	YSAV1	epz	\$35	
0036	105	CSWL	epz	\$36	
0037	106	CSWH	epz	\$37	
0038	107	KSWL	epz	\$38	
0039	108	KSWH	epz	\$39	
003A	109	PCL	epz	\$3A	
003B	110	PCH	epz	\$3B	
003C	111	A1L	epz	\$3C	
003D	112	A1H	epz	\$3D	
003E	113	A2L	epz	\$3E	
003F	114	A2H	epz	\$3F	
0800	115	;			
0040	116	A3L	epz	\$40	
0041	117	A3H	epz	\$41	
0042	118	A4L	epz	\$42	
0043	119	A4H	epz	\$43	
0044	120	A5L	epz	\$44	
0044	121	MACSTAT	epz	\$44	

0044	122	OPRND	epz \$44	
0045	123	AREG	epz \$45	
0046	124	XREG	epz \$46	
0047	125	YREG	epz \$47	
0048	126	PREG	epz \$48	
0049	127	SPNT	epz \$49	
004E	128	RNDL	epz \$4E	
004F	129	RNDH	epz \$4F	
0800	130	;		
0050	131	LINNUM	epz \$50	; 50:51
0050	132	ACL	epz \$50	
0051	133	ACH	epz \$51	
0052	134	TEMPPT	epz \$52	
0053	135	LASTPT	epz \$53	; 53:54
0055	136	TEMPST	epz \$55	; 55:5D, str scratch name/len
005E	137	INDEX	epz \$5E	; 5E:5F
0800	138	;		
0060	139	DEST	epz \$60	; 60:61
0061	140	OFFSET	epz \$61	; trick assembler in SHFTFMUL
0062	141	MULMANT	epz \$62	; 62:65
0066	142	MULGUARD	epz \$66	
0067	143	PRGTAB	epz \$67	; 67:68
0069	144	VARTAB	epz \$69	; 69:6A
006B	145	ARYTAB	epz \$6B	; 6B:6C
006D	146	STREND	epz \$6D	; 6D:6E
006F	147	FRETOP	epz \$6F	; 6F:70
0800	148	;		
0071	149	FRESPEC	epz \$71	; 71:72
0073	150	MEMSIZE	epz \$73	; 73:74
0075	151	CURLIN	epz \$75	; 75:76
0077	152	OLDLIN	epz \$77	; 77:78
0079	153	TEXTPTR	epz \$79	; 79:7A
007B	154	DATLIN	epz \$7B	; 7B:7C
007D	155	DATPTR	epz \$7D	; 7D:7E
007F	156	SRCPTR	epz \$7F	; 7F:80
0800	157	;		
0081	158	VARNAM	epz \$81	; 81:82
0083	159	VARPNT	epz \$83	; 83:84
0085	160	FORPNT	epz \$85	; 85:86
0087	161	LASTOP	epz \$87	
0087	162	TXPTRSAV	epz \$87	; 87:88
0089	163	CPRTYPE	epz \$89	
008A	164	FUNCNAM	epz \$8A	; 8A:8B
008A	165	TEMP3	epz \$8A	; 8A:8E
008C	166	DSCPTR	epz \$8C	; 8C:8D
008F	167	DSCLLEN	epz \$8F	
008F	168	SPCLFLAG	epz \$8F	
0800	169	;		
0091	170	RTNADLEN	epz \$91	
0092	171	ARGGUARD	epz \$92	
0093	172	TEMP1	epz \$93	; 93:97
0094	173	HIGHDS	epz \$94	; 94:95
0094	174	LEN	epz \$94	
0095	175	PROCESS	epz \$95	
0096	176	HIGHTR	epz \$96	; 96:97
0098	177	TEMP2	epz \$98	; 98:9C
0099	178	COUNTER	epz \$99	; FPOUT
009A	179	EXPCOUNT	epz \$9A	; FPOUT
009B	180	LOWTR	epz \$9B	; 9B:9C
009D	181	DSCTMP	epz \$9D	; 9D:9F
009D	182	FACEXP	epz \$9D	

```

009E      183  FACMANT  epz  $9E      ; 9E:A1
0800      184  ;
00A0      185  VARPTR   epz  $A0      ; A0:A1
00A2      186  FACSIGN  epz  $A2
00A3      187  SERLEN   epz  $A3
00A4      188  SAVARGEX epz  $A4
00A5      189  ARGEXP   epz  $A5
00A6      190  ARGMANT  epz  $A6      ; A6:A9
00AA      191  ARGSIGN  epz  $AA
00AB      192  XORSIGN  epz  $AB
00AC      193  FACGUARD epz  $AC
00AB      194  STRING1  epz  $AB      ; AB:AC
00AD      195  STRING2  epz  $AD      ; AD:AE
00AD      196  SAVY     epz  $AD      ; FPOUT
00AF      197  PRGEND   epz  $AF      ; AF:B0
0800      198  ;
00B1      199  CHRGTADR epz  $B1
00B8      200  TXTPTR   epz  $B8
0800      201  ;
00C9      202  IRAND     epz  $C9      ; C9:CC 4-byte integer
00CD      203  TOKEN     epz  $CD
0800      204  ;
00D0      205  HRXSTRT  epz  $D0      ; D0:D1
00D2      206  HRYSTRT  epz  $D2
00D3      207  HRXEND   epz  $D3      ; D3:D4
00D5      208  HRYEND   epz  $D5
0800      209  ;
00D0      210  SHPVAL    epz  $D0
00D1      211  ROTQVAL   epz  $D1
00D2      212  ROTHVAL   epz  $D2
00D3      213  ROTVVAL   epz  $D3
00D4      214  ROTHSUM   epz  $D4
00D5      215  ROTVSUM   epz  $D5
00D7      216  SHPOLD    epz  $D7
0800      217  ;
00D6      218  RUNFLAG   epz  $D6
00D8      219  ERRFLG    epz  $D8
00DA      220  ERRLIN    epz  $DA      ; DA:DB
00DC      221  ERRPOS    epz  $DC      ; DC:DD
00DE      222  ERRNUM    epz  $DE
00DF      223  ERRSTK    epz  $DF
0800      224  ;
00E0      225  HRXCOOR   epz  $E0      ; E0:E1
00E2      226  HRYCOOR   epz  $E2
00E4      227  HRCOLOR   epz  $E4
00E5      228  HRHORZ    epz  $E5
00E6      229  HRPAG     epz  $E6
00E7      230  HRSCALE   epz  $E7
00E8      231  HRSHTBL   epz  $E8      ; E8:E9
00EA      232  HRCOLCNT  epz  $EA
0800      233  ;
00F0      234  FIRST     epz  $F0
00F1      235  SPEEDBYT  epz  $F1
00F2      236  TRACEFLG  epz  $F2
00F3      237  FLASHBYT  epz  $F3
00F4      238  TXTPTRSV  epz  $F4      ; F4:F5
00F6      239  CURLINSV  epz  $F6      ; F6:F7
00F8      240  REMSTK    epz  $F8
00F9      241  HRROT     epz  $F9
0800      242  ;
0800      243  enz

```

```
0800      244 ;
0001      245 HLINMOD equ 1 ; HLIN routine modification
0001      246 FPCDMOD equ 1 ; floating point routines mod
0800      247 ;
0000      248 ZERO equ $00
0002      249 IVARLEN equ $02
0003      250 AVARLEN equ $03
0004      251 MANTLEN equ $04
0005      252 FACLEN equ $05
0005      253 AHDRLN equ $05
0007      254 SVARLEN equ $07
0007      255 PXLSBYTE equ $07
0008      256 BYTEBITS equ $08
000B      257 DFLTDIM equ $0B
0020      258 MANTBITS equ MANTLEN*BYTEBITS
0800      259 ;
0003      260 MOVEMASK equ $03
0004      261 PLOTMASK equ $04
0007      262 SCMDMASK equ $07
000F      263 SROTMASK equ $0F
0800      264 ;
003F      265 INVERSE equ $3F
007F      266 FLASH equ $7F
007F      267 INVRSE80 equ $7F
007F      268 MSBCLR equ $7F
0080      269 MSBSET equ $80
0080      270 EXPBIAS equ $80
00DF      271 LWRMASK equ $DF
00EF      272 MAXINPUT equ $EF ; max APPLESOFT input
00FE      273 NEG TWO equ $FE
00FF      274 NEG ONE equ $FF
0800      275 ;
0082      276 CTRLB equ $82
0083      277 CTRLC equ $83
0085      278 CTRL E equ $85
0087      279 ASCIBELL equ $87
0088      280 CTRLH equ $88
0088      281 LARROW equ $88
008A      282 DARROW equ $8A
008A      283 CTRLJ equ $8A
008B      284 CTRLK equ $8B
008B      285 UARROW equ $8B
008D      286 RETURN equ $8D
0090      287 CTRLP equ $90
0093      288 CTRLS equ $93
0095      289 CTRLU equ $95
0095      290 RARROW equ $95
0098      291 CTRLX equ $98
0099      292 CTRL Y equ $99
009B      293 ESCAPE equ $9B
00A0      294 SPACE equ $A0
0800      295 ;
0006      296 GOODF8 equ $06 ; 0xF8 ROM version
0800      297 ;
00A5      298 PWRUPBYT equ $A5
0800      299 ;
00FF      300 ERROR.1 equ $FF ; ctrl-C input error
00FE      301 ERROR.2 equ $FE ; INPUT error
0800      302 ;
0800      303 ; Tokens values used directly.
0800      304 ;
```

```

00C0      305  TKTAB      equ  $C0
00C1      306  TKTO      equ  $C1
00C2      307  TKFN      equ  $C2
00C3      308  TKSPC     equ  $C3
00C4      309  TKTHEN    equ  $C4
00C5      310  TKAT      equ  $C5
00C6      311  TKNOT     equ  $C6
00C7      312  TKSTEP    equ  $C7
00C8      313  TKPLUS    equ  $C8
00C9      314  TKMINUS   equ  $C9
00CF      315  TKGRTR    equ  $CF
00D0      316  TKEQUAL   equ  $D0
00D2      317  TKSGN     equ  $D2
00D7      318  TKSCRN    equ  $D7
0800      319  ;
0800      320  ; Operator tag values used directly.
0800      321  ;
0079      322  OTVPLUS   equ  $79
0079      323  OTVMINUS  equ  $79
007B      324  OTVMULT   equ  $7B
007B      325  OTVDIVID  equ  $7B
007D      326  OTVPOWER  equ  $7D
0050      327  OTVAND     equ  $50
0046      328  OTVOR      equ  $46
007F      329  OTVGREAT  equ  $7F
007F      330  OTVEQUAL  equ  $7F
0064      331  OTVLESS   equ  $64
0800      332  ;
0100      333  PAGESIZE  equ  $100
0100      334  STACK     equ  $100
0800      335  ;
0200      336  INPUT     equ  $200
0800      337  ;
03ED      338  XFERADR   equ  $3ED          ; XFER destination address
0800      339  ;
03EF      340  AUTOBRK   equ  $3EF          ; 3EF:3F1
03F2      341  AUTORSET  equ  $3F2          ; 3F2:3F3
03F4      342  PWRSTATE  equ  $3F4
03F5      343  USRAHAND  equ  $3F5          ; 3F5:3F7
03F8      344  USRYHAND  equ  $3F8          ; 3F8:3FA
03FB      345  NMASKIRQ  equ  $3FB          ; 3FB:3FD
03FE      346  MASKIRQ   equ  $3FE          ; 3FE:3FF
0800      347  ;
0400      348  TEXTPG1   equ  $400
05B0      349  PG1TXLOC  equ  $5B0
0800      350  ;
047B      351  OLDCH     equ  $47B          ; last CH used by video F/W
04FB      352  XMODE     equ  $4FB          ; video firmware mode
0800      353  ;
057B      354  OURCH     equ  $57B          ; 80 column CH
05FB      355  OURCV     equ  $5FB          ; 80 column CV
0800      356  ;
067B      357  XCHAR     equ  $67B          ; character to be printed/read
06FB      358  XCOORD    equ  $6FB          ; GOTOXY X-coord (Pascal only)
0800      359  ;
077B      360  XTEMP1    equ  $77B          ; temp
077B      361  OLDBASL   equ  $77B          ; last BASL (Pascal only)
07FB      362  XTEMP2    equ  $7FB          ; temp2
07FB      363  OLDBASH   equ  $7FB          ; last BASH (Pascal only)
07F8      364  MSLOT     equ  $7F8
0800      365  ;

```

```
0800      366  PAGE08      equ  $0800
0C00      367  PAGE0C      equ  $0C00
1000      368  PAGE10      equ  $1000
2000      369  PAGE20      equ  $2000
4000      370  PAGE40      equ  $4000
BD00      371  PAGEBD      equ  $BD00
BF00      372  PAGEBF      equ  $BF00
C000      373  PAGEC0      equ  $C000
C100      374  PAGEC1      equ  $C100
C800      375  PAGEC8      equ  $C800
F000      376  PAGEF0      equ  $F000
FE00      377  PAGEFE      equ  $FE00
0800      378  ;
C000      379  KEY          equ  $C000
C000      380  STR80OFF      equ  $C000
C001      381  STR80ON      equ  $C001
C002      382  RAMRDOFF      equ  $C002
C003      383  RAMRDON      equ  $C003
C004      384  RAMWROFF      equ  $C004
C005      385  RAMWRON      equ  $C005
C006      386  CXROMOFF      equ  $C006
C007      387  CXROMON      equ  $C007
C008      388  AUXZPOFF      equ  $C008
C009      389  AUXZPON      equ  $C009
C00A      390  C3ROMOFF      equ  $C00A
C00B      391  C3ROMON      equ  $C00B
C00C      392  VID80OFF      equ  $C00C
C00D      393  VID80ON      equ  $C00D
C00E      394  ALTCHOFF      equ  $C00E
C00F      395  ALTCHON      equ  $C00F
0800      396  ;
C010      397  CLRKEY        equ  $C010
C011      398  RDBANK2      equ  $C011
C012      399  RDLGRAM      equ  $C012
C013      400  RDRAMRD      equ  $C013
C014      401  RDRAMWR      equ  $C014
C015      402  RDCXROM      equ  $C015
C016      403  RDAUXZP      equ  $C016
C017      404  RDC3ROM      equ  $C017
C018      405  RDSTR80      equ  $C018
C019      406  RDVRTBLK     equ  $C019
C01A      407  RDTEXT        equ  $C01A
C01B      408  RDMIXED      equ  $C01B
C01C      409  RDPAGE2      equ  $C01C
C01D      410  RDHIRES      equ  $C01D
C01E      411  RDALTCH      equ  $C01E
C01F      412  RDVID80      equ  $C01F
0800      413  ;
C020      414  TAPEOUT      equ  $C020
C030      415  SPKRTOGL     equ  $C030
0800      416  ;
C050      417  TEXTOFF      equ  $C050
C051      418  TEXTON       equ  $C051
C052      419  MIXEDOFF     equ  $C052
C053      420  MIXEDON      equ  $C053
C054      421  PAGE1ON      equ  $C054
C055      422  PAGE2ON      equ  $C055
C056      423  HIRESOFF     equ  $C056
C057      424  HIRESON      equ  $C057
C058      425  ANN1OFF      equ  $C058
C05A      426  ANN2OFF      equ  $C05A
```



```

C05D      427  ANN3ON      equ  $C05D
C05F      428  ANN4ON      equ  $C05F
0800      429  ;
C060      430  TAPEIN      equ  $C060
C061      431  PB1IN       equ  $C061
C062      432  PB2IN       equ  $C062
C064      433  GC1IN       equ  $C064
0800      434  ;
C070      435  GCTOGL      equ  $C070
0800      436  ;
C080      437  RAM2WP       equ  $C080
C081      438  ROM2WE       equ  $C081
C082      439  ROM2WP       equ  $C082
C083      440  RAM2WE       equ  $C083
C088      441  RAM1WP       equ  $C088
C08B      442  RAM1WE       equ  $C08B
0800      443  ;
D000      444  PAGED0      equ  $D000
0800      445  ;
0800      446  ;
0800      447  ; BASIC mode bits
0800      448  ;
0800      449  ; 0..... BASIC active
0800      450  ; 1..... Pascal active
0800      451  ; .0.....
0800      452  ; .1.....
0800      453  ; ..0..... print control characters
0800      454  ; ..1..... do not print ctrl chars
0800      455  ; ...0.....
0800      456  ; ...1.....
0800      457  ; ....0... print next ctrl char
0800      458  ; ....1... do not print next ctrl char
0800      459  ; .....0..
0800      460  ; .....1..
0800      461  ; .....0.
0800      462  ; .....1.
0800      463  ; .....0 mouse text inactive
0800      464  ; .....1 mouse text active
0800      465  ;
0040      466  M.6          equ  %01000000
0020      467  M.CTL2      equ  %00100000      ; do not print controls
0010      468  M.4          equ  %00010000
0008      469  M.CTL       equ  %00001000      ; temp ctrl disable
0004      470  M.2          equ  %00000100
0002      471  M.1          equ  %00000010
0001      472  M.MOUSE      equ  %00000001
0800      473  ;
0800      474  ;
0800      475  ; Pascal mode bits
0800      476  ;
0800      477  ; 0..... BASIC active
0800      478  ; 1..... Pascal active
0800      479  ; .0.....
0800      480  ; .1.....
0800      481  ; ..0.....
0800      482  ; ..1.....
0800      483  ; ...0..... cursor always on
0800      484  ; ...1..... cursor always off
0800      485  ; ....0... GOTOXY n/a
0800      486  ; ....1... GOTOXY in progress
0800      487  ; .....0.. normal video

```

```
0800      488 ; .....1.. inverse video
0800      489 ; .....0. Pascal 1.1 firmware active
0800      490 ; .....1. Pascal 1.0 interface
0800      491 ; .....0 mouse text inactive
0800      492 ; .....1 mouse text active
0800      493 ;
0080      494 M.PASCAL equ %10000000 ; Pascal active
0010      495 M.CURSOR equ %00010000 ; do not print cursor
0008      496 M.GOXY equ %00001000 ; GOTOXY in progress
0004      497 M.VMODE equ %00000100 ; Pascal video mode
0002      498 M.PAS1.0 equ %00000010 ; Pascal 1.0 mode
0800      499 ;
0800      500 ;
0800      501 ; Definitions
0800      502 ;
0800      503 ; ACA - At Correct Address. Same address as unaltered ROM.
0800      504 ;
0800      505 ;
0800      506          icl "C0.L"
```

```
LLOAD C0.L,A$4000
```

```

0800      1          ttl "ROM Source Code, C0.L"
0800      2      ;
0800      3      ;
0800      4      ; C0.L
0800      5      ;
0800      6      ;
C000      7          org PAGEC0
C000      8          obj PAGE10
C000      9          usr
C000     10      ;
C000     11      ;
C000     12      IOSPACE:
C000     13      ;
C000     14      ;
C000     15          dfs PAGESIZE,ZERO
C100     16      ;
C100     17      ;
C100     18      ; DOCXCMD is called by the patched 0xF8 ROM. It provides
C100     19      ; an extension to the 0xF8 routines that do not work in 80
C100     20      ; columns.
C100     21      ;
C100     22      ; Before jumping here the 0xF8 ROM disables slot I/O and
C100     23      ; enables ROM I/O. This makes the entire space from 0xC100
C100     24      ; to 0xCFFF available except for the 0xC300 page.
C100     25      ;
C100     26      ; On exit, slot I/O is restored if necessary.
C100     27      ;
C100     28      ; The stack has the PHP for status of the internal 0xCN00
C100     29      ; ROM.
C100     30      ;
C100     31      ; If VID80 is on and the XMODE byte is valid, this call
C100     32      ; will be handled by the 80 column routine. Otherwise, it
C100     33      ; will be handled by the 40 column routine. Return to the
C100     34      ; Autostart ROM is done by CXEXIT.
C100     35      ;
C100     36      ;
C100 4C 11 C2     37      DOCXCMD    jmp DOCMD
C103     38      ;
C103     39      ;
C103     40      ; CLREOP support.
C103     41      ;
C103 A4 24     42      XCLREOP    ldy CH
C105 A5 25     43                  lda CV
C107     44      ;
C107 48     45      ^1        pha
C108     46      ;
C108 20 00 CE   47                  jsr XVTAB3
C10B 20 F3 C1   48                  jsr XCLREOL1
C10E     49      ;
C10E A0 00     50                  ldy #ZERO
C110     51      ;
C110 68     52                  pla
C111 69 00     53                  adc #ZERO
C113     54      ;
C113 C5 23     55                  cmp WNDBTM
C115 90 F0     56                  bcc <1
C117     57      ;
C117 B0 34     58                  bcs >5          ; always taken
C119     59      ;
C119     60      ;

```

```

C119      61 ; HOME support.
C119      62 ;
C119 A5 22 63 XHOME      lda WNDTOP
C11B 85 25 64           sta CV
C11D      65 ;
C11D A0 00 66           ldy #ZERO
C11F 84 24 67           sty CH
C121      68 ;
C121 F0 E4 69           beq <1                ; always taken
C123      70 ;
C123      71 ;
C123      72 ; SCROLL support.
C123      73 ;
C123 A5 22 74 XSCROLL    lda WNDTOP
C125 48    75           pha
C126      76 ;
C126 20 00 CE 77           jsr XVTAB3
C129      78 ;
C129 A5 28 79 ^2        lda BASL
C12B 85 2A 80           sta BAS2L
C12D      81 ;
C12D A5 29 82           lda BASH
C12F 85 2B 83           sta BAS2H
C131      84 ;
C131 A4 21 85           ldy WNDWDTH
C133 88    86           dey
C134      87 ;
C134 68    88           pla
C135 69 01 89           adc #1
C137      90 ;
C137 C5 23 91           cmp WNDBTM
C139 B0 0D 92           bcs >4
C13B      93 ;
C13B 48    94           pha
C13C      95 ;
C13C 20 00 CE 96           jsr XVTAB3
C13F      97 ;
C13F B1 28 98 ^3        lda (BASL),Y
C141 91 2A 99           sta (BAS2L),Y
C143     100 ;
C143 88    101           dey
C144 10 F9 102           bpl <3
C146     103 ;
C146 30 E1 104           bmi <2                ; always taken
C148     105 ;
C148 A0 00 106 ^4        ldy #ZERO
C14A     107 ;
C14A 20 F3 C1 108           jsr XCLREOL1
C14D     109 ;
C14D A5 25 110 ^5        lda CV
C14F     111 ;
C14F 4C 00 CE 112 ^6        jmp XVTAB3
C152     113 ;
C152     114 ;
C152     115 ; SETWND support.
C152     116 ;
C152 A9 28 117 XSETWND    lda #40
C154 85 21 118           sta WNDWDTH
C156     119 ;
C156 A9 18 120           lda #24
C158 85 23 121           sta WNDBTM

```

```

C15A          122 ;
C15A A9 17    123         lda #23
C15C 85 25    124         sta CV
C15E D0 EF    125         bne <6                ; always taken
C160          126 ;
C160          127 ;
C160          128 ; CLEOLZ support.
C160          129 ;
C160 A4 2A    130 XCLREOL2 ldy BAS2L
C162          131 ;
C162 4C F3 C1 132         jmp XCLREOL1
C165          133 ;
C165          134 ;
C165          135 ; 80 column routines begin here.
C165          136 ;
C165          137 ; 80 column SCROLL support.
C165          138 ;
C165 4C EB CB 139 YSCROLL jmp SCROLLUP
C168          140 ;
C168          141 ;
C168          142 ; 80 column CLREOL support.
C168          143 ;
C168 4C 97 CC 144 YCLREOL jmp XGS
C16B          145 ;
C16B          146 ;
C16B          147 ; 80 column CLEOLZ support.
C16B          148 ;
C16B A4 2A    149 YCLREOL2 ldy BAS2L
C16D          150 ;
C16D 4C 9A CC 151         jmp XGSEOLZ
C170          152 ;
C170          153 ;
C170          154 ; 80 column CLREOP support.
C170          155 ;
C170 4C 77 CC 156 YCLREOP jmp XVT
C173          157 ;
C173          158 ;
C173          159 ; 80 column SETWND support.
C173          160 ;
C173 4C A5 C2 161 YSETWND jmp XSETWNDX
C176          162 ;
C176          163 ;
C176          164 ; RESET support.
C176          165 ;
C176          166 XRESET:
C176 4C B5 C2 167 YRESET jmp XRESETX
C179          168 ;
C179          169 ;
C179          170 ; 80 column RDKEY support.
C179          171 ;
C179 4C 11 CE 172 YRDKEY jmp XRDKEY2
C17C          173 ;
C17C          174 ;
C17C          175 ; 80 column HOME support.
C17C          176 ;
C17C 20 74 CC 177 YHOME jsr XFF
C17F          178 ;
C17F AD 7B 05 179         lda OURCH
C182 85 24    180         sta CH
C184 8D 7B 04 181         sta OLDCH
C187          182 ;

```

```

C187 4C FB CD      183          jmp XVTAB2
C18A              184          ;
C18A              185          ;
C18A              186          ; 80 column IOPORT support.
C18A              187          ;
C18A B4 00        188 YIOPORT  ldy LOC0,X
C18C F0 0F        189          beq >2
C18E              190          ;
C18E C0 1B        191          cpy #KEYIN
C190 F0 0E        192          beq ISO
C192              193          ;
C192 20 7D CD     194          jsr QUIT
C195              195          ;
C195              196          ;
C195              197          ; IOPORT support.
C195              198          ;
C195 B4 00        199 XIOPORT  ldy LOC0,X
C197 F0 04        200          beq >2
C199              201          ;
C199 A9 FD        202 ^1      lda /KEYIN
C19B 95 01        203          sta LOC1,X
C19D              204          ;
C19D B5 01        205 ^2      lda LOC1,X
C19F              206          ;
C19F 60           207          rts
C1A0              208          ;
C1A0              209          ;
C1A0 A5 37        210 ISO      lda CSWH
C1A2 C9 C3        211          cmp /C3SPACE
C1A4 D0 F3        212          bne <1
C1A6              213          ;
C1A6 4C 32 C8     214          jmp C3IN
C1A9              215          ;
C1A9              216          ;
C1A9              217          ; RDKEY support.
C1A9              218          ;
C1A9 A4 24        219 XRDKEY  ldy CH
C1AB              220          ;
C1AB B1 28        221          lda (BASL),Y
C1AD 48           222          pha
C1AE              223          ;
C1AE 29 3F        224          and #INVERSE          ; set screen to inverse
C1B0 09 40        225          ora #$40
C1B2 91 28        226          sta (BASL),Y
C1B4              227          ;
C1B4 68           228          pla
C1B5              229          ;
C1B5 60           230          rts
C1B6              231          ;
C1B6              232          ;
C1B6              233          ; Implemented BASCALC support that restores Y-reg.
C1B6              234          ;
C1B6 A4 28        235 XBASCLC  ldy BASL
C1B8              236          ;
C1B8 20 BA CA     237          jsr XBASCALC
C1BB 90 4B        238          bcc CXEXIT          ; always taken
C1BD              239          ;
C1BD              240          ;
C1BD              241          ; RDESC support.
C1BD              242          ;
C1BD              243          ; Map ^H to J, ^U to K, ^J to M, and ^K to I.

```

```

C1BD      244 ;
C1BD 20 01 FD 245 XRDESC      jsr UPRCASE
C1C0      246 ;
C1C0 A0 03   247             ldy #KBDOUT-KBDTBL+1
C1C2      248 ;
C1C2 D9 EB C2 249 ^1        cmp KBDTBL,Y
C1C5 D0 03   250             bne >2
C1C7      251 ;
C1C7 B9 EF C2 252             lda KBDOUT,Y
C1CA      253 ;
C1CA 88      254 ^2        dey
C1CB 10 F5   255             bpl <1
C1CD      256 ;
C1CD 30 39   257             bmi CXEXIT          ; always taken
C1CF      258 ;
C1CF      259 ;
C1CF      260 ; NEWVW VIDWAIT support.
C1CF      261 ;
C1CF 20 70 C8 262 XNEWVW      jsr CXNEWVW
C1D2      263 ;
C1D2 4C 08 C2 264             jmp CXEXIT
C1D5      265 ;
C1D5      266 ;
C1D5      267 ; GETFMT support.
C1D5      268 ;
C1D5      269 XGETFMT:
C1D5 8A      270 YGETFMT      txa
C1D6 85 2E   271             sta FORMAT
C1D8      272 ;
C1D8 29 03   273             and #3
C1DA 85 2F   274             sta LENGTH
C1DC      275 ;
C1DC A5 2A   276             lda BAS2L          ; recall opcode index
C1DE      277 ;
C1DE 4C D5 C5 278             jmp XGETFMT2
C1E1      279 ;
C1E1      280 ;
C1E1      281 ; MINIASM support.
C1E1      282 ;
C1E1      283 XGOMINI:
C1E1 20 F0 FC 284 YGOMINI      jsr NXTLINE
C1E4      285 ;
C1E4 8A      286             txa
C1E5 85 34   287             sta YSAV
C1E7      288 ;
C1E7 60      289             rts
C1E8      290 ;
C1E8      291 ;
C1E8      292 ; PICKFIX support.
C1E8      293 ;
C1E8      294 XPICKFIX:
C1E8 AC 7B 05 295 YPICKFIX      ldy OURCH
C1EB 20 44 CE 296             jsr PICK
C1EE      297 ;
C1EE 09 80   298             ora #MSBSET
C1F0      299 ;
C1F0 60      300             rts
C1F1      301 ;
C1F1      302 ;
C1F1      303 ; CLREOL support.
C1F1      304 ;

```

```

C1F1 A4 24      305 XCLREOL ldy CH
C1F3           306 ;
C1F3 A9 A0      307 XCLREOL1 lda #SPACE
C1F5           308 ;
C1F5 2C 1E C0   309          bit RDALTCH
C1F8 10 06      310          bpl >1
C1FA           311 ;
C1FA 24 32      312          bit INVFLG
C1FC 30 02      313          bmi >1
C1FE           314 ;
C1FE A9 20      315          lda #' '          ; use inverse blank
C200           316 ;
C200 4C A5 CC   317 ^1          jmp CLR40
C203           318 ;
C203           319 ;
C203           320 ; VTAB support. Modified to save Y-reg in BASL. Fall
C203           321 ; into CXEXIT.
C203           322 ;
C203 A4 28      323 XVTAB      ldy BASL
C205 20 00 CE   324          jsr XVTAB3
C208           325 ;
C208           326 ;
C208           327 ; If PLP is +, turn CX ROM off, otherwise leave CX ROM on.
C208           328 ;
C208 28         329 CXEXIT    plp
C209 30 03      330          bmi >1
C20B           331 ;
C20B 4C 3C FA   332          jmp CXOFF
C20E           333 ;
C20E 4C 3F FA   334 ^1          jmp CXRTN
C211           335 ;
C211           336 ;
C211 88         337 DOCMD     dey
C212 30 BB      338          bmi XNEWVW          ; Y-reg = 0
C214           339 ;
C214 88         340          dey
C215 30 A6      341          bmi XRDESC          ; Y-reg = 1
C217           342 ;
C217 88         343          dey
C218 30 9C      344          bmi XBASCLC          ; Y-reg = 2
C21A           345 ;
C21A 88         346          dey
C21B 30 3E      347          bmi XKEYIN          ; Y-reg = 3
C21D           348 ;
C21D 88         349          dey
C21E 30 E3      350          bmi XVTAB          ; Y-reg = 4
C220           351 ;
C220 A9 C2      352          lda /CXEXIT-1
C222 48         353          pha
C223           354 ;
C223 A9 07      355          lda #CXEXIT-1
C225 48         356          pha
C226           357 ;
C226 AD FB 04   358          lda XMODE
C229 29 D6      359          and #M.PASCAL|M.6|M.4|M.2|M.1
C22B D0 0E      360          bne >1
C22D           361 ;
C22D 98         362          tya
C22E           363 ;
C22E D8         364          cld          ; added to clear DECIMAL
C22F 18         365          clc

```



```

C230          366 ;
C230 69 0C    367      adc #YREGTBLY-YREGTBLX
C232 48       368      pha
C233          369 ;
C233 20 50 C8 370      jsr CSETUP
C236 20 FB CD 371      jsr XVTAB2
C239          372 ;
C239 68       373      pla
C23A A8       374      tay
C23B          375 ;
C23B A9 C1    376 ^1    lda /PAGEC1
C23D 48       377      pha
C23E          378 ;
C23E B9 43 C2 379      lda YREGTBLX,Y
C241 48       380      pha
C242          381 ;
C242 60       382      rts
C243          383 ;
C243          384 ;
C243          385 ; PAGEC1 routines.
C243          386 ;
C243 18       387 YREGTBLX byt XHOME-1      ; Y-reg = 5
C244 22       388      byt XSCROLL-1      ; Y-reg = 6
C245 F0       389      byt XCLREOL-1      ; Y-reg = 7
C246 5F       390      byt XCLREOL2-1     ; Y-reg = 8
C247 75       391      byt XRESET-1      ; Y-reg = 9
C248 02       392      byt XCLREOP-1     ; Y-reg = 10
C249 A8       393      byt XRDKEY-1      ; Y-reg = 11
C24A 51       394      byt XSETWND-1     ; Y-reg = 12
C24B E0       395      byt XGOMINI-1     ; Y-reg = 13
C24C 94       396      byt XIOPORT-1    ; Y-reg = 14
C24D E7       397      byt XPICKFIX-1   ; Y-reg = 15
C24E D4       398      byt XGETFMT-1    ; Y-reg = 16
C24F          399 ;
C24F 7B       400 YREGTBLY byt YHOME-1    ; Y-reg = 5
C250 64       401      byt YSCROLL-1     ; Y-reg = 6
C251 67       402      byt YCLREOL-1     ; Y-reg = 7
C252 6A       403      byt YCLREOL2-1   ; Y-reg = 8
C253 75       404      byt YRESET-1     ; Y-reg = 9
C254 6F       405      byt YCLREOP-1    ; Y-reg = 10
C255 78       406      byt YRDKEY-1     ; Y-reg = 11
C256 72       407      byt YSETWND-1    ; Y-reg = 12
C257 E0       408      byt YGOMINI-1     ; Y-reg = 13
C258 89       409      byt YIOPORT-1    ; Y-reg = 14
C259 E7       410      byt YPICKFIX-1   ; Y-reg = 15
C25A D4       411      byt YGETFMT-1    ; Y-reg = 16
C25B          412 ;
C25B          413 ;
C25B          414 ; KEYIN support.
C25B          415 ;
C25B 2C 1F C0 416 XKEYIN  bit RDVID80
C25E 10 06    417      bpl >2
C260          418 ;
C260 20 74 C8 419      jsr CXNEWVW2
C263          420 ;
C263 4C 08 C2 421 ^1    jmp CXEXIT
C266          422 ;
C266 A8       423 ^2    tay
C267          424 ;
C267 8A       425      txa
C268 48       426      pha

```

```

C269          427 ;
C269 98        428      tya
C26A 48        429      pha
C26B          430 ;
C26B 48        431      pha
C26C          432 ;
C26C 68        433 ^3    pla
C26D C9 FF     434      cmp #NEGONE
C26F F0 04     435      beq >4
C271          436 ;
C271 A9 FF     437      lda #NEGONE
C273 D0 02     438      bne >5          ; always taken
C275          439 ;
C275 68        440 ^4    pla
C276 48        441      pha
C277          442 ;
C277 48        443 ^5    pha
C278          444 ;
C278 A4 24     445      ldy CH
C27A          446 ;
C27A 91 28     447      sta (BASL),Y
C27C          448 ;
C27C E6 4E     449 ^6    inc RNDL
C27E D0 0A     450      bne >7
C280          451 ;
C280 A5 4F     452      lda RNDH          ; time to blink?
C282          453 ;
C282 E6 4F     454      inc RNDH
C284          455 ;
C284 45 4F     456      eor RNDH
C286 29 40     457      and #$40
C288 D0 E2     458      bne <3          ; yes, blink it
C28A          459 ;
C28A AD 00 C0  460 ^7    lda KEY
C28D 10 ED     461      bpl <6
C28F          462 ;
C28F 68        463      pla
C290          464 ;
C290 A4 24     465      ldy CH
C292          466 ;
C292 68        467      pla
C293 91 28     468      sta (BASL),Y
C295          469 ;
C295 68        470      pla
C296 AA        471      tax
C297          472 ;
C297 AD 00 C0  473      lda KEY
C29A          474 ;
C29A 2C 10 C0  475      bit CLRKEY
C29D          476 ;
C29D C9 FF     477      cmp #NEGONE
C29F D0 C2     478      bne <1
C2A1          479 ;
C2A1 A9 88     480      lda #LARROW
C2A3 D0 BE     481      bne <1          ; always taken
C2A5          482 ;
C2A5          483 ;
C2A5 20 52 C1  484 XSETWNDX jsr XSETWND
C2A8          485 ;
C2A8 2C 1F C0  486      bit RDVID80
C2AB 10 02     487      bpl >1

```

```

C2AD          488 ;
C2AD 06 21    489      asl WNDWDTH
C2AF          490 ;
C2AF A5 25    491 ^1      lda CV
C2B1 8D FB 05 492      sta OURCV
C2B4          493 ;
C2B4 60       494      rts
C2B5          495 ;
C2B5          496 ;
C2B5          497 ; RESET support.
C2B5          498 ;
C2B5 D8       499 XRESETX cld
C2B6          500 ;
C2B6 20 08 FB 501      jsr RSETINIT
C2B9          502 ;
C2B9 A9 FF    503      lda #NEGONE
C2BB 8D FB 04 504      sta XMODE
C2BE          505 ;
C2BE          506 ;
C2BE          507 ; If the solid Apple key is pressed go to DIAGS.
C2BE          508 ;
C2BE          509 ;      lda PB2IN
C2BE          510 ;      bpl >1
C2BE          511 ;
C2BE          512 ;      jmp DIAGS
C2BE          513 ;
C2BE          514 ;
C2BE          515 ; If the open Apple key is pressed destroy memory and
C2BE          516 ; coldstart the system.
C2BE          517 ;
C2BE AD 61 C0 518 ^1      lda PB1IN
C2C1 10 16    519      bpl CXRESET
C2C3          520 ;
C2C3          521 ;
C2C3          522 ; Write 2 bytes of SPACE on each page including the RESET
C2C3          523 ; vector (rewritten using X-reg). Avoid the DOS 4.5.06
C2C3          524 ; interface page and begin on PAGEBD rather than on PAGEBF.
C2C3          525 ;
C2C3 A9 00    526      lda #PAGEBD
C2C5 85 3C    527      sta A1L
C2C7          528 ;
C2C7 A2 BD    529      ldx /PAGEBD
C2C9          530 ;
C2C9 A0 AC    531      ldy #$B0-4          ; starting page offset
C2CB          532 ;
C2CB A9 A0    533      lda #SPACE
C2CD          534 ;
C2CD 86 3D    535 ^2      stx A1H
C2CF          536 ;
C2CF 91 3C    537      sta (A1L),Y
C2D1          538 ;
C2D1 88       539      dey
C2D2          540 ;
C2D2 91 3C    541      sta (A1L),Y
C2D4          542 ;
C2D4 CA       543      dex
C2D5 E0 01    544      cpx #1          ; avoid the stack
C2D7 D0 F4    545      bne <2
C2D9          546 ;
C2D9          547 ;
C2D9          548 ; If there is a ROM slot card in slot 3, do not enable the

```

```

C2D9      549 ; internal C3 ROM space.  If no card, switch in the C3 ROM
C2D9      550 ; space only if there is a RAM slot card in the video slot.
C2D9      551 ;
C2D9      552 ; The //e powers up with the internal C3 ROM switched in.
C2D9      553 ; TSTROMCD switches it out.  CXRESET may or may not switch
C2D9      554 ; it back in.
C2D9      555 ;
C2D9 8D 0B C0 556 CXRESET sta C3ROMON
C2DC      557 ;
C2DC 20 89 CA 558          jsr TSTROMCD
C2DF D0 03    559          bne >1
C2E1      560 ;
C2E1 8D 0A C0 561          sta C3ROMOFF
C2E4      562 ;
C2E4 AD FF CF 563 ^1      lda CLRROM          ; disable extension ROM
C2E7      564 ;
C2E7 2C 10 C0 565          bit CLRKEY
C2EA      566 ;
C2EA 60      567          rts
C2EB      568 ;
C2EB      569 ;
C2EB 88 95 8A 570 KBDTBL   byt CTRLH,CTRLU,CTRLJ,CTRLK
C2EE 8B
C2EF      571 ;
C2EF CA CB CD 572 KBDOUT   asc "JKMI"
C2F2 C9
C2F3      573 ;
C2F3      574 ;
C2F3      575          dfs PAGESIZE-*)&NEGONE,ZERO ; 13 bytes
C300      576 ;
C300      577 ;
C300      578 C3SPACE:
C300      579 ;
C300      580 ; This page must not be used by any F8 ROM routines.  When
C300      581 ; enabled it claims the C800 space.  Thus peripheral cards
C300      582 ; cannot use AUXMOVE or XFER from their C800 space.
C300      583 ;
C300 2C 2E CA 584 BASICINT bit BITBYT60
C303 70 12    585          bvs BASICENT          ; always taken
C305      586 ;
C305 38      587 BASICIN   sec
C306      588 ;
C306 90 00    589          bcc *+2
C308      590          dfs !-1
C307      591 ;
C307 18      592 BASICOUT clc
C308      593 ;
C308 B8      594          clv
C309 50 0C    595          bvc BASICENT          ; always taken
C30B      596 ;
C30B      597 ;
C30B 01      598          hex 01          ; generic signature byte
C30C 88      599          hex 88          ; device signature byte
C30D      600 ;
C30D 4A      601          byt JPINIT          ; Pascal INIT
C30E 50      602          byt JPREAD          ; Pascal READ
C30F 56      603          byt JPWRITE        ; Pascal WRITE
C310 5C      604          byt JPSTAT          ; Pascal STATUS
C311      605 ;
C311      606 ;
C311      607 ; 128K support routines.

```

```

C311      608 ;
C311 4C 76 C3 609 AUXMOVE jmp DOMOVE ; memory move across banks
C314      610 ;
C314 4C C3 C3 611 XFER jmp DOXFER ; transfer across banks
C317      612 ;
C317      613 ;
C317 8D 7B 06 614 BASICENT sta XCHAR
C31A      615 ;
C31A 98      616 tya
C31B 48      617 pha
C31C      618 ;
C31C 8A      619 txa
C31D 48      620 pha
C31E      621 ;
C31E 08      622 php
C31F      623 ;
C31F AD FB 04 624 lda XMODE
C322      625 ;
C322 2C F8 07 626 bit MSLOT
C325 30 05    627 bmi >2
C327      628 ;
C327 09 08    629 ora #M.CTL
C329 8D FB 04 630 sta XMODE
C32C      631 ;
C32C 20 6D C3 632 ^2 jsr SETC8
C32F      633 ;
C32F 28      634 plp
C330 70 15    635 bvs JBASINIT
C332      636 ;
C332 90 10    637 bcc JC8
C334      638 ;
C334 AA      639 tax
C335 10 0D    640 bpl JC8
C337      641 ;
C337 20 58 CD 642 jsr SETKEYIN
C33A      643 ;
C33A 68      644 pla
C33B AA      645 tax
C33C      646 ;
C33C 68      647 pla
C33D A8      648 tay
C33E      649 ;
C33E AD 7B 06 650 lda XCHAR
C341      651 ;
C341 6C 38 00 652 jmp (KSWL)
C344      653 ;
C344      654 ;
C344 4C 7C C8 655 JC8 jmp CXVIDCK3
C347      656 ;
C347      657 ;
C347 4C 03 C8 658 JBASINIT jmp BASCINIT
C34A      659 ;
C34A      660 ;
C34A 20 6D C3 661 JPINIT jsr SETC8
C34D      662 ;
C34D 4C B4 C9 663 jmp PPINIT
C350      664 ;
C350      665 ;
C350 20 6D C3 666 JPREAD jsr SETC8
C353      667 ;
C353 4C D6 C9 668 jmp PPREAD

```

```

C356          669 ;
C356          670 ;
C356 20 6D C3 671 JPWRITE jsr SETC8
C359          672 ;
C359 4C F0 C9 673          jmp PPWRITE
C35C          674 ;
C35C          675 ;
C35C AA       676 JPSTAT tax
C35D F0 08    677          beq >1
C35F          678 ;
C35F CA       679          dex
C360 D0 07    680          bne >2
C362          681 ;
C362 2C 00 C0 682          bit KEY
C365 10 04    683          bpl >3
C367          684 ;
C367 38       685 ^1      sec
C368          686 ;
C368 60       687          rts
C369          688 ;
C369 A2 03    689 ^2      ldx #3          ; error flag
C36B          690 ;
C36B 18       691 ^3      clc
C36C          692 ;
C36C 60       693          rts
C36D          694 ;
C36D          695 ;
C36D A2 C3    696 SETC8    ldx /C3SPACE
C36F 8E F8 07 697          stx MSLOT
C372          698 ;
C372 AE FF CF 699          ldx CLRROM
C375          700 ;
C375 60       701          rts
C376          702 ;
C376          703 ;
C376          704 ; AUXMOVE routine; crossbank memory move.
C376          705 ;
C376          706 ; A1L/H = source start address
C376          707 ; A2L/H = source end address
C376          708 ; A4L/H = destination start address
C376          709 ;
C376          710 ; carry set = main -> aux
C376          711 ; carry clear = aux -> main
C376          712 ;
C376 48       713 DOMOVE pha
C377          714 ;
C377 98       715          tya
C378 48       716          pha
C379          717 ;
C379 AD 13 C0 718          lda RDRAMRD
C37C 48       719          pha
C37D          720 ;
C37D AD 14 C0 721          lda RDRAMWR
C380 48       722          pha
C381          723 ;
C381 90 08    724          bcc >1
C383          725 ;
C383 8D 02 C0 726          sta RAMRDOFF
C386 8D 05 C0 727          sta RAMWRON
C389          728 ;
C389 B0 06    729          bcs >2          ; always taken

```

```

C38B      730 ;
C38B 8D 04 C0 731 ^1      sta RAMWROFF
C38E 8D 03 C0 732      sta RAMRDON
C391      733 ;
C391 A0 00    734 ^2      ldy #ZERO
C393      735 ;
C393 B1 3C    736 ^3      lda (A1L),Y
C395 91 42    737      sta (A4L),Y
C397      738 ;
C397 E6 42    739      inc A4L
C399 D0 02    740      bne >4
C39B      741 ;
C39B E6 43    742      inc A4H
C39D      743 ;
C39D A5 3C    744 ^4      lda A1L
C39F C5 3E    745      cmp A2L
C3A1      746 ;
C3A1 A5 3D    747      lda A1H
C3A3 E5 3F    748      sbc A2H
C3A5      749 ;
C3A5 E6 3C    750      inc A1L
C3A7 D0 02    751      bne >5
C3A9      752 ;
C3A9 E6 3D    753      inc A1H
C3AB      754 ;
C3AB 90 E6    755 ^5      bcc <3
C3AD      756 ;
C3AD 8D 04 C0 757      sta RAMWROFF
C3B0      758 ;
C3B0 68      759      pla
C3B1 10 03    760      bpl >6
C3B3      761 ;
C3B3 8D 05 C0 762      sta RAMWRON
C3B6      763 ;
C3B6 8D 02 C0 764 ^6      sta RAMRDOFF
C3B9      765 ;
C3B9 68      766      pla
C3BA 10 03    767      bpl >7
C3BC      768 ;
C3BC 8D 03 C0 769      sta RAMRDON
C3BF      770 ;
C3BF 68      771 ^7      pla
C3C0 A8      772      tay
C3C1      773 ;
C3C1 68      774      pla
C3C2      775 ;
C3C2 60      776      rts
C3C3      777 ;
C3C3      778 ;
C3C3      779 ; XFER routine.
C3C3      780 ;
C3C3      781 ; 0x3ED/0x3EE = destination address
C3C3      782 ;
C3C3      783 ; carry set = transfer to aux
C3C3      784 ; carry clear = transfer to main
C3C3      785 ;
C3C3      786 ; vflag set = use aux ZP/STACK
C3C3      787 ; vflag clear = use main ZP/STACK
C3C3      788 ;
C3C3 48      789 DOXFER pha
C3C4      790 ;

```

```

C3C4 AD ED 03      791      lda XFERADR
C3C7 48            792      pha
C3C8              793      ;
C3C8 AD EE 03      794      lda XFERADR+1
C3CB 48            795      pha
C3CC              796      ;
C3CC 90 08         797      bcc >1
C3CE              798      ;
C3CE 8D 03 C0      799      sta RAMRDON
C3D1 8D 05 C0      800      sta RAMWRON
C3D4              801      ;
C3D4 B0 06         802      bcs >2                ; always taken
C3D6              803      ;
C3D6 8D 02 C0      804      ^1      sta RAMRDOFF
C3D9 8D 04 C0      805      sta RAMWROFF
C3DC              806      ;
C3DC 68           807      ^2      pla
C3DD 8D EE 03      808      sta XFERADR+1
C3E0              809      ;
C3E0 68           810      pla
C3E1 8D ED 03      811      sta XFERADR
C3E4              812      ;
C3E4 68           813      pla
C3E5              814      ;
C3E5 70 05         815      bvs >3
C3E7              816      ;
C3E7 8D 08 C0      817      sta AUXZPOFF
C3EA              818      ;
C3EA 50 03         819      bvc >4                ; always taken
C3EC              820      ;
C3EC 8D 09 C0      821      ^3      sta AUXZPON
C3EF              822      ;
C3EF 6C ED 03      823      ^4      jmp (XFERADR)
C3F2              824      ;
C3F2              825      ;
C3F2              826      dfs 2,ZERO                ; 2 bytes
C3F4              827      ;
C3F4              828      ;
C3F4              829      ; This is where the interrupt routine returns to. The ROM
C3F4              830      ; is not necessarily enabled.
C3F4              831      ;
C3F4 8D 81 C0      832      IRQDONE sta ROM2WE
C3F7              833      ;
C3F7 4C 7A FC      834      jmp IRQDONE2
C3FA              835      ;
C3FA              836      ;
C3FA              837      ; This is the main entry point for the interrupt handler.
C3FA              838      ; It enables the internal ROM and simply falls into the
C3FA              839      ; main part of the interrupt handler at 0xC400.
C3FA              840      ;
C3FA 2C 15 C0      841      IRQRTN  bit RDCXROM
C3FD              842      ;
C3FD 8D 07 C0      843      sta CXROMON
C400              844      ;
C400              845      ;
C400              846      icl "C4.L"

```

LLOAD C4.L,A\$4000


```

C400      1          ttl "ROM Source Code, C4.L"
C400      2      ;
C400      3      ;
C400      4      ; C4.L
C400      5      ;
C400      6      ;
C400      7      ; Interrupt Handler
C400      8      ;
C400      9      ; The bits of a system status byte is created and saved
C400     10      ; to 0x44. If a bit is on, that function is turned off.
C400     11      ; LC RAM is turned off.
C400     12      ;
C400     13      ; Bit 7      Bit 6      Bit 5      Bit 4
C400     14      ; -----
C400     15      ; RDAUXZP RDPAGE2 RDRAMRD RDRAMWR
C400     16      ; 0=main  0=off   0=off   0=off
C400     17      ; 1=AUX   1=on    1=on    1=on
C400     18      ;
C400     19      ; Bit 3      Bit 2      Bit 1      Bit 0
C400     20      ; -----
C400     21      ; LCRAM   BANK2    BANK1    RDCXROM
C400     22      ; 0=ROM   0=off   0=off   0=off
C400     23      ; 1=RAM   1=on    1=on    1=on
C400     24      ;
C400     25      ; If Bit 3 off, Bits 1 and 2 are off.
C400     26      ; If Bit 3 on, Bit 1 or Bit 2 is on.
C400     27      ;
C400     28      ;
C400 D8     29          cld
C401      30      ;
C401 38     31          sec                      ; C=1 for internal slot space
C402      32      ;
C402 30 01  33          bmi >1
C404      34      ;
C404 18     35          clc
C405      36      ;
C405 48     37      ^1      pha                      ; save on stack and not 0x45
C406 48     38          pha
C407 48     39          pha
C408      40      ;
C408 8A     41          txa
C409      42      ;
C409 BA     43          tsx
C40A      44      ;
C40A E8     45          inx                      ; ridiculous
C40B E8     46          inx
C40C E8     47          inx
C40D E8     48          inx
C40E      49      ;
C40E 48     50          pha
C40F      51      ;
C40F 98     52          tya
C410 48     53          pha
C411      54      ;
C411 BD 00 01 55          lda STACK,X          ; get BREAK status
C414 29 10     56          and #$10            ; mask for Z flag
C416 A8       57          tay                  ; save for later
C417      58      ;
C417      59      ;
C417      60      ; Determine current machine state.

```

```

C417      61 ;
C417 AD 18 C0      62      lda RDSTR80
C41A 2D 1C C0      63      and RDPAGE2
C41D      64 ;
C41D 29 80      65      and #MSBSET
C41F F0 05      66      beq >2
C421      67 ;
C421 A9 20      68      lda #$20
C423 8D 54 C0      69      sta PAGE1ON
C426      70 ;
C426 2A      71 ^2      rol
C427      72 ;
C427 2C 13 C0      73      bit RDRAMRD
C42A 10 05      74      bpl >3
C42C      75 ;
C42C 8D 02 C0      76      sta RAMRDOFF
C42F      77 ;
C42F 09 20      78      ora #$20
C431      79 ;
C431 2C 14 C0      80 ^3      bit RDRAMWR
C434 10 05      81      bpl >4
C436      82 ;
C436 8D 04 C0      83      sta RAMWROFF
C439      84 ;
C439 09 10      85      ora #$10
C43B      86 ;
C43B 2C 12 C0      87 ^4      bit RDLCRAM
C43E 10 0C      88      bpl >6
C440      89 ;
C440 09 0C      90      ora #$0C
C442      91 ;
C442 2C 11 C0      92      bit RDBANK2
C445 10 02      93      bpl >5
C447      94 ;
C447 49 06      95      eor #6
C449      96 ;
C449 8D 81 C0      97 ^5      sta ROM2WE
C44C      98 ;
C44C 2C 16 C0      99 ^6      bit RDAUXZP
C44F 10 0D      100     bpl >7
C451      101 ;
C451 BA      102     tsx
C452 8E 01 01      103     stx STACK+1      ; save aux stack pointer
C455      104 ;
C455 AE 00 01      105     ldx STACK      ; restore main stack pointer
C458 9A      106     txs
C459      107 ;
C459 8D 08 C0      108     sta AUXZPOFF
C45C      109 ;
C45C 09 80      110     ora #MSBSET
C45E      111 ;
C45E 88      112 ^7     dey      ; check for BREAK
C45F 30 0C      113     bmi >8
C461      114 ;
C461 85 44      115     sta MACSTAT      ; save machine state
C463      116 ;
C463 68      117     pla
C464 A8      118     tay
C465      119 ;
C465 68      120     pla
C466 AA      121     tax

```

```

C467          122 ;
C467 68        123      pla
C468 68        124      pla
C469 68        125      pla
C46A          126 ;
C46A 4C 47 FA  127      jmp NEWBREAK
C46D          128 ;
C46D 48        129 ^8    pha                ; save machine state
C46E          130 ;
C46E AD F8 07  131      lda MSLOT
C471 48        132      pha
C472          133 ;
C472 A9 C3     134      lda /IRQDONE          ; save return RTI address
C474 48        135      pha
C475          136 ;
C475 A9 F4     137      lda #IRQDONE
C477 48        138      pha
C478          139 ;
C478 08        140      php
C479          141 ;
C479 4C 74 FC  142      jmp GOTOIRQ          ; enter user's IRQ handler
C47C          143 ;
C47C          144 ;
C47C          145 ; The ROM must be reenabled with a LDA ROM2WE if the
C47C          146 ; language card is write protected, write enabled, or
C47C          147 ; being write enabled. An INC ABS,X is used because
C47C          148 ; some of the switches are Read and some are Write since
C47C          149 ; both the 6502 and 65C02 do 2 reads before the write.
C47C          150 ;
C47C AD 81 C0  151 NEWIRQ  lda ROM2WE
C47F          152 ;
C47F 68        153      pla                ; recall machine state
C480 10 07     154      bpl >1
C482          155 ;
C482 8D 09 C0  156      sta AUXZPON
C485          157 ;
C485 AE 01 01  158      ldx STACK+1          ; recall aux stack pointer
C488 9A        159      txs
C489          160 ;
C489 A0 06     161 ^1    ldy #SWTBLEN
C48B          162 ;
C48B 10 06     163 ^2    bpl >3
C48D          164 ;
C48D BE C1 C4  165      ldx RAMSWTBL,Y
C490          166 ;
C490 FE 00 C0  167      inc IOSPACE,X
C493          168 ;
C493 88        169 ^3    dey
C494 30 03     170      bmi >4
C496          171 ;
C496 0A        172      asl
C497 D0 F2     173      bne <2
C499          174 ;
C499 0A        175 ^4    asl                ; if internal slot space
C49A 0A        176      asl                ; then C=1
C49B          177 ;
C49B 68        178      pla
C49C A8        179      tay
C49D          180 ;
C49D BA        181      tsx
C49E          182 ;

```

```

C49E A9 00      183      lda #*-*          ; get RTI opcode
C4A0           184      dfs !-1
C49F 40        185      rti
C4A0 48        186      pha
C4A1           187      ;
C4A1 A9 C0     188      lda /CXROMOFF
C4A3 48        189      pha
C4A4           190      ;
C4A4 A9 06     191      lda #CXROMOFF
C4A6 69 00     192      adc #ZERO          ; add 1 if internal slot space
C4A8 48        193      pha
C4A9           194      ;
C4A9 A9 00     195      lda #*-*          ; get STA ABS opcode
C4AB           196      dfs !-1
C4AA 8D 00 00  197      sta *-*
C4AD           198      dfs !-2
C4AB 48        199      pha
C4AC           200      ;
C4AC           201      ;
C4AC           202      ; Make return address point to code just pushed onto the
C4AC           203      ; stack.
C4AC           204      ;
C4AC 9A        205      txs
C4AD 8A        206      txa
C4AE           207      ;
C4AE 69 03     208      adc #3
C4B0 AA        209      tax
C4B1           210      ;
C4B1 38        211      sec
C4B2           212      ;
C4B2 E9 07     213      sbc #7
C4B4 9D 00 01  214      sta STACK,X
C4B7           215      ;
C4B7 E8        216      inx
C4B8           217      ;
C4B8 A9 01     218      lda /STACK
C4BA 9D 00 01  219      sta STACK,X
C4BD           220      ;
C4BD 68        221      pla
C4BE AA        222      tax
C4BF           223      ;
C4BF 68        224      pla
C4C0           225      ;
C4C0 60        226      rts          ; go to code on stack
C4C1           227      ;
C4C1           228      ;
C4C1 83        229      RAMSWTBL byt RAM2WE
C4C2 8B        230      byt RAM1WE
C4C3 8B        231      byt RAM1WE
C4C4 05        232      byt RAMWRON
C4C5 03        233      byt RAMRDON
C4C6 55        234      byt PAGE2ON
C4C7           235      ;
0006           236      SWTBLLEN equ *-RAMSWTBL
C4C7           237      ;
C4C7           238      ;
C4C7           239      dfs 1,ZERO          ; 1 byte
C4C8           240      ;
C4C8           241      ;
C4C8           242      ; Continuation of NXTLINE2.
C4C8           243      ;

```

```

C4C8 20 00 C5      244 FORM      jsr GETNSP
C4CB              245 ;
C4CB 84 34        246      sty YSAV
C4CD              247 ;
C4CD DD 30 FA     248      cmp CHAR1,X
C4D0 D0 13        249      bne >1
C4D2              250 ;
C4D2 20 00 C5     251      jsr GETNSP
C4D5              252 ;
C4D5 DD 36 FA     253      cmp CHAR2,X
C4D8 F0 0D        254      beq >3
C4DA              255 ;
C4DA BD 36 FA     256      lda CHAR2,X
C4DD F0 07        257      beq >2
C4DF              258 ;
C4DF C9 A4        259      cmp #"$"
C4E1 F0 03        260      beq >2
C4E3              261 ;
C4E3 A4 34        262      ldy YSAV
C4E5              263 ;
C4E5 18           264 ^1      clc
C4E6              265 ;
C4E6 88           266 ^2      dey
C4E7              267 ;
C4E7 26 44        268 ^3      rol OPRND
C4E9              269 ;
C4E9 E0 03        270      cpx #3
C4EB D0 0D        271      bne >5
C4ED              272 ;
C4ED 20 A7 FF     273      jsr GETNUM
C4F0              274 ;
C4F0 A5 3F        275      lda A2H
C4F2 F0 01        276      beq >4
C4F4              277 ;
C4F4 E8           278      inx
C4F5              279 ;
C4F5 86 35        280 ^4      stx YSAV1
C4F7              281 ;
C4F7 A2 03        282      ldx #3
C4F9              283 ;
C4F9 88           284      dey
C4FA              285 ;
C4FA 86 3D        286 ^5      stx A1H
C4FC              287 ;
C4FC CA          288      dex
C4FD 10 C9        289      bpl FORM
C4FF              290 ;
C4FF 60           291      rts
C500              292 ;
C500              293 ;
C500              294 ; GETNSP gets the next non-blank character for the
C500              295 ; Mini-Assembler.
C500              296 ;
C500 20 FD FC     297 GETNSP  jsr UPRMON
C503              298 ;
C503 C9 A0        299      cmp #SPACE
C505 F0 F9        300      beq GETNSP
C507              301 ;
C507 60           302      rts
C508              303 ;
C508              304 ;

```

```

C508 20 75 FE    305 CXSTEP    jsr A1PC
C50B 20 D0 F8    306          jsr INSTDSP
C50E             307 ;
C50E 68          308          pla
C50F 68          309          pla
C510             310 ;
C510 A2 08       311          ldx #INITBLEN
C512             312 ;
C512 BD CC C5    313 ^1      lda INITBL-1,X
C515 95 3C       314          sta A1L,X
C517             315 ;
C517 CA          316          dex
C518 D0 F8       317          bne <1
C51A             318 ;
C51A AD 00 C0    319          lda KEY
C51D 10 19       320          bpl >4
C51F             321 ;
C51F 2C 10 C0    322          bit CLRKEY
C522             323 ;
C522 C9 A0       324          cmp #" "
C524 D0 0E       325          bne >3
C526             326 ;
C526 AD 00 C0    327 ^2      lda KEY
C529 10 FB       328          bpl <2
C52B             329 ;
C52B 2C 10 C0    330          bit CLRKEY
C52E             331 ;
C52E C9 9B       332          cmp #ESCAPE          ; ESC check
C530 D0 02       333          bne >3
C532             334 ;
C532 E6 34       335          inc YSAV
C534             336 ;
C534 C9 83       337 ^3      cmp #$83          ; ^C break
C536 F0 04       338          beq >5
C538             339 ;
C538 A1 3A       340 ^4      lda (PCL,X)
C53A D0 03       341          bne >6
C53C             342 ;
C53C 38          343 ^5      sec          ; break return
C53D B0 77       344          bcs >4          ; always taken
C53F             345 ;
C53F A4 2F       346 ^6      ldy LENGTH
C541             347 ;
C541 C9 00       348          cmp #*- *
C543             349          dfs !-1
C542 60          350          rts
C543 F0 43       351          beq XRTS
C545             352 ;
C545 C9 00       353          cmp #*- *
C547             354          dfs !-1
C546 20 00 00    355          jsr *- *
C549             356          dfs !-2
C547 F0 4F       357          beq XJSR
C549             358 ;
C549 C9 00       359          cmp #*- *
C54B             360          dfs !-1
C54A 4C 00 00    361          jmp *- *
C54D             362          dfs !-2
C54B F0 56       363          beq XJMP
C54D             364 ;
C54D C9 00       365          cmp #*- *

```

```

C54F      366      dfs !-1
C54E 6C 00 00 367      jmp (*-*)
C551      368      dfs !-2
C54F F0 53    369      beq XJMPAT
C551      370      ;
C551 C9 00    371      cmp #*-*
C553      372      dfs !-1
C552 7C 00 00 373      jmp (*-*,X)
C555      374      dfs !-2
C553 F0 20    375      beq XJMPX
C555      376      ;
C555 C9 00    377      cmp #*-*
C557      378      dfs !-1
C556 40       379      rti
C557 F0 2B    380      beq XRTI
C559      381      ;
C559 C9 00    382      cmp #*-*
C55B      383      dfs !-1
C55A 80 00    384      bra *+2
C55C      385      dfs !-1
C55B D0 02    386      bne >7
C55D      387      ;
C55D A9 00    388      lda #*-*
C55F      389      dfs !-1
C55E 10 00    390      bpl *+2
C560      391      dfs !-1
C55F      392      ;
C55F 29 1F    393      ^7 and #$1F
C561 49 14    394      eor #$14
C563      395      ;
C563 C9 04    396      cmp #4
C565 F0 02    397      beq >9
C567      398      ;
C567 B1 3A    399      ^8 lda (PCL),Y
C569      400      ;
C569 99 3C 00 401      ^9 sta A1L,Y
C56C      402      ;
C56C 88       403      dey
C56D 10 F8    404      bpl <8
C56F      405      ;
C56F 20 3F FF 406      jsr RESTORE
C572      407      ;
C572 4C 3C 00 408      jmp A1L
C575      409      ;
C575      410      ;
C575 B1 3A    411      XJMPX lda (PCL),Y
C577 AA       412      tax
C578      413      ;
C578 88       414      dey
C579      415      ;
C579 18       416      clc
C57A      417      ;
C57A B1 3A    418      lda (PCL),Y
C57C 65 46    419      adc XREG
C57E 90 01    420      bcc >1
C580      421      ;
C580 E8       422      inx
C581      423      ;
C581 38       424      ^1 sec
C582 B0 26    425      bcs >2
C584      426      ;

```

; always taken

```

C584 18          427  XRTI      clc
C585          428  ;
C585 68          429          pla
C586 85 48       430          sta PREG
C588          431  ;
C588 68          432  XRTS      pla
C589 85 3A       433          sta PCL
C58B          434  ;
C58B 68          435          pla
C58C          436  ;
C58C 85 3B       437  PCINC2    sta PCH
C58E          438  ;
C58E A5 2F       439  PCINC3    lda LENGTH
C590 20 56 F9    440          jsr PCADJ3
C593          441  ;
C593 84 3B       442          sty PCH
C595          443  ;
C595 18          444          clc
C596 90 14       445          bcc >3          ; always taken
C598          446  ;
C598 18          447  XJSR      clc
C599          448  ;
C599 20 54 F9    449          jsr PCADJ2
C59C          450  ;
C59C AA          451          tax
C59D          452  ;
C59D 98          453          tya
C59E 48          454          pha
C59F          455  ;
C59F 8A          456          txa
C5A0 48          457          pha
C5A1          458  ;
C5A1 A0 02       459          ldy #2
C5A3          460  ;
C5A3 18          461  XJMP      clc
C5A4          462  ;
C5A4 B1 3A       463  XJMPAT    lda (PCL),Y
C5A6 AA          464          tax
C5A7          465  ;
C5A7 88          466          dey
C5A8          467  ;
C5A8 B1 3A       468          lda (PCL),Y
C5AA          469  ;
C5AA 86 3B       470  ^2      stx PCH
C5AC          471  ;
C5AC 85 3A       472  ^3      sta PCL
C5AE          473  ;
C5AE B0 F3       474          bcs XJMP
C5B0          475  ;
C5B0 20 D7 FA    476          jsr REGDSP
C5B3          477  ;
C5B3 A4 34       478          ldy YSAV
C5B5          479  ;
C5B5 18          480          clc          ; normal return
C5B6          481  ;
C5B6 4C CA FC    482  ^4      jmp STEPRTN
C5B9          483  ;
C5B9          484  ;
C5B9 18          485  BRANCH    clc
C5BA          486  ;
C5BA A0 01       487          ldy #1

```



```

C5BC          488 ;
C5BC B1 3A    489     lda (PCL),Y
C5BE 20 56 F9 490     jsr PCADJ3
C5C1          491 ;
C5C1 85 3A    492     sta PCL
C5C3          493 ;
C5C3 98       494     tya
C5C4          495 ;
C5C4 38       496     sec
C5C5 B0 C5    497     bcs PCINC2           ; always taken
C5C7          498 ;
C5C7          499 ;
C5C7 20 4A FF 500 BRANCH2 jsr SAVE
C5CA          501 ;
C5CA 38       502     sec
C5CB B0 C1    503     bcs PCINC3           ; always taken
C5CD          504 ;
C5CD          505 ;
C5CD EA       506 INITBL  nop
C5CE EA       507     nop
C5CF          508 ;
C5CF 4C C7 C5 509     jmp BRANCH2
C5D2          510 ;
C5D2 4C B9 C5 511     jmp BRANCH
C5D5          512 ;
0008          513 INITBLEN equ *-INITBL
C5D5          514 ;
C5D5          515 ;
C5D5          516 ; Form index into mnemonic table.
C5D5          517 ;
C5D5          518 ; 1) 1XXX1010 => 00101XXX
C5D5          519 ; 2) XXXYYY01 => 00111XXX
C5D5          520 ; 3) XXXYYY10 => 00110XXX
C5D5          521 ; 4) XXXYY100 => 00100XXX
C5D5          522 ; 5) XXXXX000 => 000XXXXX
C5D5          523 ;
C5D5 A2 0B    524 XGETFMT2 ldx #OPTBLL-OPTBLC+1
C5D7          525 ;
C5D7 DD 71 CA 526 ^1     cmp OPTBLC,X
C5DA D0 06    527     bne >2
C5DC          528 ;
C5DC BD 7D CA 529     lda OPTBLL,X
C5DF          530 ;
C5DF A0 00    531     ldy #ZERO
C5E1          532 ;
C5E1 60       533     rts
C5E2          534 ;
C5E2 CA       535 ^2     dex
C5E3 10 F2    536     bpl <1
C5E5          537 ;
C5E5 29 8F    538     and #$8F
C5E7          539 ;
C5E7 AA       540     tax
C5E8          541 ;
C5E8 A5 2A    542     lda BAS2L           ; get OP CODE
C5EA          543 ;
C5EA A0 03    544     ldy #3
C5EC          545 ;
C5EC E0 8A    546     cpx #$8A
C5EE F0 0B    547     beq >5
C5F0          548 ;

```

```
C5F0 4A          549 ^3      lsr
C5F1 90 08       550      bcc >5
C5F3             551 ;
C5F3 4A          552      lsr
C5F4             553 ;
C5F4 4A          554 ^4      lsr
C5F5 09 20       555      ora #$20
C5F7             556 ;
C5F7 88          557      dey
C5F8 D0 FA       558      bne <4
C5FA             559 ;
C5FA C8          560      iny
C5FB             561 ;
C5FB 88          562 ^5      dey
C5FC D0 F2       563      bne <3
C5FE             564 ;
C5FE 60          565      rts
C5FF             566 ;
C5FF             567 ;
C5FF             568      dfs 1,ZERO      ; 1 byte
C600             569 ;
C600             570 ;
C600             571      icl "SW.L"
```

```
LLOAD SW.L,A$4000
```

```

C600          1          ttl "ROM Source Code, SW.L"
C600          2          ;
C600          3          ;
C600          4          ; SW.L
C600          5          ;
C600          6          ;
C600          7          ; Implementation of Cornelis Bongers Garbage Collection
C600          8          ; algorithm.
C600          9          ;
C600 24 95      10 PROCVAR bit PROCESS
C602 30 4A      11          bmi PROCSPCL
C604          12          ;
C604 B1 9B      13          lda (LOWTR),Y
C606 85 5E      14          sta INDEX
C608          15          ;
C608 C5 6F      16          cmp FRETOP
C60A          17          ;
C60A C8         18          iny
C60B          19          ;
C60B B1 9B      20          lda (LOWTR),Y
C60D 85 5F      21          sta INDEX+1
C60F          22          ;
C60F E5 70      23          sbc FRETOP+1
C611 90 3A      24          bcc >4
C613          25          ;
C613 A0 00      26          ldy #ZERO
C615          27          ;
C615 B1 5E      28          lda (INDEX),Y
C617 48         29          pha
C618          30          ;
C618 B1 9B      31          lda (LOWTR),Y
C61A F0 31      32          beq >4
C61C          33          ;
C61C C8         34          iny
C61D          35          ;
C61D C9 02      36          cmp #2
C61F F0 05      37          beq >1
C621          38          ;
C621 B0 12      39          bcs >2
C623          40          ;
C623 A9 FF      41          lda #NEGONE
C625          42          ;
C625 2C 00 00   43          bit *-*
C628          44          dfs !-2
C626          45          ;
C626 B1 5E      46          ^1 lda (INDEX),Y
C628          47          ;
C628 91 9B      48          sta (LOWTR),Y
C62A          49          ;
C62A C8         50          iny
C62B          51          ;
C62B A9 FF      52          lda #NEGONE
C62D 85 8F      53          sta SPCLFLAG
C62F 91 9B      54          sta (LOWTR),Y
C631          55          ;
C631 A0 00      56          ldy #ZERO
C633 F0 15      57          beq >3          ; always taken
C635          58          ;
C635 B1 5E      59          ^2 lda (INDEX),Y
C637          60          ;

```

```

C637 C8          61      iny
C638          62      ;
C638 91 9B      63      sta (LOWTR),Y
C63A          64      ;
C63A B1 5E      65      lda (INDEX),Y
C63C 09 80      66      ora #MSBSET
C63E 91 5E      67      sta (INDEX),Y
C640          68      ;
C640 A0 00      69      ldy #ZERO
C642          70      ;
C642 A5 9B      71      lda LOWTR
C644 91 5E      72      sta (INDEX),Y
C646          73      ;
C646 C8          74      iny
C647          75      ;
C647 8A          76      txa
C648 91 5E      77      sta (INDEX),Y
C64A          78      ;
C64A 68          79      ^3    pla
C64B 91 9B      80      sta (LOWTR),Y
C64D          81      ;
C64D 60          82      ^4    rts
C64E          83      ;
C64E          84      ;
C64E C8          85      PROCSPCL iny
C64F 84 94      86      sty LEN
C651          87      ;
C651 A9 FF      88      lda #NEGONE
C653 D1 9B      89      cmp (LOWTR),Y
C655 D0 F6      90      bne <4
C657          91      ;
C657 88          92      dey
C658          93      ;
C658 D1 9B      94      cmp (LOWTR),Y
C65A D0 02      95      bne >1
C65C          96      ;
C65C C6 94      97      dec LEN
C65E          98      ;
C65E 20 73 E5   99      ^1    jsr COPYVAR
C661          100     ;
C661 A5 94      101     lda LEN
C663 91 9B      102     sta (LOWTR),Y
C665          103     ;
C665 C8          104     iny
C666          105     ;
C666 A5 8A      106     lda FUNCNAM
C668 91 9B      107     sta (LOWTR),Y
C66A          108     ;
C66A C8          109     iny
C66B          110     ;
C66B A5 8B      111     lda FUNCNAM+1
C66D 91 9B      112     sta (LOWTR),Y
C66F          113     ;
C66F 60          114     rts
C670          115     ;
C670          116     ;
C670          117     ; This version of the Sweet 16 Metaprocessor has been
C670          118     ; revised from the original; it has 5 added instructions.
C670          119     ;
C670          120     ; Sweet 16 Registers
C670          121     ;

```

```

C670      122 ; Register Description
C670      123 ; -----
C670      124 ; R0 Accumulator (ACC)
C670      125 ; R1-R11 User registers
C670      126 ; R12 Subroutine return Stack Pointer
C670      127 ; R13 Compare instruction results
C670      128 ; R14 Status Register (PR & carry flag)
C670      129 ; R15 Program Counter (PC)
C670      130 ;
C670      131 ;
C670      132 ; Sweet 16 Non-Register Opcodes
C670      133 ;
C670      134 ; Opcode Mnemonic Description
C670      135 ; -----
C670      136 ; 00 RTN Return to 6502 mode
C670      137 ; 01 BR adr Branch always, PC+ = adr+2
C670      138 ; 02 BNC adr Branch if no carry, PC+ = adr+2
C670      139 ; 03 BC adr Branch if carry, PC+ = adr+2
C670      140 ; 04 BP adr Branch if plus, PC+ = adr+2
C670      141 ; 05 BM adr Branch if minus, PC+ = adr+2
C670      142 ; 06 BZ adr Branch if zero, PC+ = adr+2
C670      143 ; 07 BNZ adr Branch if not zero, PC+ = adr+2
C670      144 ; 08 BM1 adr Branch if minus 1, PC+ = adr+2
C670      145 ; 09 BNM1 adr Branch if not minus 1, PC+ = adr+2
C670      146 ; 0A SOUT chr Send "chr" to COUT
C670      147 ; 0B RS Return from subroutine, PC = (R12--)
C670      148 ; 0C BS adr Branch to subroutine, (R12++) = PC
C670      149 ; 0D RSNS Return from subroutine, PC = R12
C670      150 ; 0E BSNS adr Branch to subroutine, R12 = PC
C670      151 ; 0F SJMP adr Jump to address, PC = adr
C670      152 ;
C670      153 ;
C670      154 ; Sweet 16 Register Opcodes
C670      155 ;
C670      156 ; Opcode Mnemonic Description
C670      157 ; -----
C670      158 ; 1n SET Rn,C Load Rn immediate (with constant)
C670      159 ; 2n LD Rn Load LO ACC from Rn
C670      160 ; 3n ST Rn Move LO ACC into Rn
C670      161 ; 4n LD @Rn Load LO ACC indirectly using Rn, Rn+1
C670      162 ; 5n ST @Rn Move LO ACC indirectly using Rn, Rn+1
C670      163 ; 6n LDD @Rn Load ACC indirectly using Rn, Rn+2
C670      164 ; 7n STD @Rn Move ACC indirectly using Rn, Rn+2
C670      165 ; 8n POP @Rn Rn-1, load LO ACC indirectly using Rn
C670      166 ; 9n STP @Rn Rn-1, move LO ACC indirectly using Rn
C670      167 ; An ADD Rn Add Rn to ACC with carry out
C670      168 ; Bn SUB Rn Subtract Rn from ACC, with carry out
C670      169 ; Cn POPD @Rn Rn-2, HO ACC, Rn-1, LO ACC indirectly
C670      170 ; Dn CPR Rn ACC-Rn->R13, with carry out
C670      171 ; En INR Rn Rn+1->Rn
C670      172 ; Fn DCR Rn Rn-1->Rn
C670      173 ;
C670      174 ;
C670      175 ; The Status Register contains the previous register*2 in
C670      176 ; the LO byte and the carry flag in bit 0 of the HO byte.
C670      177 ;
C670      178 ; Branch opcodes depend on the results of the prior opcode.
C670      179 ; The register*2 of the prior opcode is saved in LO R14.
C670      180 ; It is the value currently in the prior register that is
C670      181 ; tested for the selected branch opcode.
C670      182 ;

```

```

C670      183 ; Before the BS/RS opcodes can be utilized, R12 must be
C670      184 ; initialized with the address of the stack that contains
C670      185 ; the return from subroutine addresses.
C670      186 ;
C670      187 ;
C670      188 ; Preserve 6502 registers and init SW16 program counter.
C670      189 ;
C670 20 4A FF 190 SW16      jsr SAVE
C673      191 ;
C673 68      192          pla
C674 85 1E    193          sta R15L
C676      194 ;
C676 68      195          pla
C677 85 1F    196          sta R15H
C679      197 ;
C679      198 ;
C679      199 ; Interpret a single SW16 opcode, then fetch the next SW16
C679      200 ; opcode.
C679      201 ;
C679 20 7F C6 202 SW16B      jsr SW16C
C67C      203 ;
C67C 4C 79 C6 204          jmp SW16B
C67F      205 ;
C67F      206 ;
C67F      207 ; Increment the SW16 program counter.
C67F      208 ;
C67F E6 1E    209 SW16C      inc R15L
C681 D0 02    210          bne >1
C683      211 ;
C683 E6 1F    212          inc R15H
C685      213 ;
C685      214 ;
C685      215 ; Push the common HO routine address byte onto the stack
C685      216 ; and process the SW16 opcode.
C685      217 ;
C685 A9 C7    218 ^1      lda /RTNCMD
C687 48      219          pha
C688      220 ;
C688 A0 00    221          ldy #ZERO
C68A      222 ;
C68A      223 ;
C68A      224 ; Mask the specified SW16 register and double it for
C68A      225 ; non-register address table indexing or prior register.
C68A      226 ;
C68A B1 1E    227          lda (R15L),Y
C68C 29 0F    228          and #$0F
C68E      229 ;
C68E 0A      230          asl
C68F AA      231          tax
C690      232 ;
C690      233 ;
C690      234 ; Extract opcode. If the opcode is zero, process a non-
C690      235 ; register opcode routine.
C690      236 ;
C690 4A      237          lsr
C691      238 ;
C691 51 1E    239          eor (R15L),Y
C693 F0 0D    240          beq TOBR
C695      241 ;
C695      242 ;
C695      243 ; Save register*2 in LO status byte and clear HO status

```

```

C695      244 ; byte. Form opcode*2 for register address table indexing.
C695      245 ;
C695 86 1C  246      stx R14L
C697 84 1D  247      sty R14H
C699      248 ;
C699 4A     249      lsr
C69A 4A     250      lsr
C69B 4A     251      lsr
C69C      252 ;
C69C A8     253      tay
C69D      254 ;
C69D      255 ;
C69D      256 ; Save the LO address onto the stack to process this SW16
C69D      257 ; register opcode. Also C-flag = 0; N-flag = 0 for SETCMD.
C69D      258 ;
C69D B9 E1 C6 259      lda OPTBL-2,Y
C6A0 48     260      pha
C6A1      261 ;
C6A1 60     262      rts
C6A2      263 ;
C6A2      264 ;
C6A2      265 ; Increment the SW16 program counter for the expected
C6A2      266 ; branch address. Save the LO address onto the stack to
C6A2      267 ; process this SW16 non-register opcode.
C6A2      268 ;
C6A2 E6 1E  269 TOBR    inc R15L
C6A4 D0 02  270      bne >2
C6A6      271 ;
C6A6 E6 1F  272      inc R15H
C6A8      273 ;
C6A8 BD E2 C6 274 ^2     lda BRTBL,X
C6AB 48     275      pha
C6AC      276 ;
C6AC      277 ;
C6AC      278 ; Recall prior register*2 in LO status byte and set up
C6AC      279 ; carry flag from HO status byte for BC and BNC opcodes.
C6AC      280 ; Also N-flag = 0 and Y-reg = 0.
C6AC      281 ;
C6AC A6 1C  282      ldx R14L
C6AE      283 ;
C6AE A5 1D  284      lda R14H
C6B0 4A     285      lsr
C6B1      286 ;
C6B1 60     287      rts
C6B2      288 ;
C6B2      289 ;
C6B2      290 ; Y-reg is set to 2 by SET opcode index*2. First get HO
C6B2      291 ; byte, then LO byte for designated register.
C6B2      292 ;
C6B2 B1 1E  293 SETCMD2  lda (R15L),Y
C6B4 95 01  294      sta R0H,X
C6B6      295 ;
C6B6 88     296      dey
C6B7      297 ;
C6B7 B1 1E  298      lda (R15L),Y
C6B9 95 00  299      sta R0L,X
C6BB      300 ;
C6BB      301 ;
C6BB      302 ; Increment the SW16 program counter by 2. C-flag clear.
C6BB      303 ;
C6BB A5 1E  304      lda R15L

```

```

C6BD 69 02      305      adc #2
C6BF 85 1E      306      sta R15L
C6C1 90 02      307      bcc >3
C6C3           308      ;
C6C3 E6 1F      309      inc R15H
C6C5           310      ;
C6C5 60         311      ^3      rts
C6C6           312      ;
C6C6           313      ;
C6C6           314      ; R12 must contain the stack address where to pop the
C6C6           315      ; return SW16 program counter. Pop the HO byte, then
C6C6           316      ; the LO byte. Then R12 - 2 -> R12.
C6C6           317      ;
C6C6           318      ;
C6C6 A2 18      319      RSCMD2    ldx #R12L
C6C8           320      ;
C6C8 20 F7 C7   321      jsr DCRCMD
C6CB           322      ;
C6CB A1 00      323      lda (R0L,X)
C6CD 85 1F      324      sta R15H
C6CF           325      ;
C6CF 20 F7 C7   326      jsr DCRCMD
C6D2           327      ;
C6D2 A1 00      328      lda (R0L,X)
C6D4 85 1E      329      sta R15L
C6D6           330      ;
C6D6 60         331      rts
C6D7           332      ;
C6D7           333      ;
C6D7           334      ; Copy the current SW16 program counter in R15 to R12.
C6D7           335      ;
C6D7 A5 1E      336      BSNSCMD2  lda R15L
C6D9 85 18      337      sta R12L
C6DB           338      ;
C6DB A5 1F      339      lda R15H
C6DD 85 19      340      sta R12H
C6DF           341      ;
C6DF 4C 9C C7   342      jmp BRCMD
C6E2           343      ;
C6E2           344      ;
C6E2           345      ; The byte-pairs of this table have been swapped to remove
C6E2           346      ; the unused Fn+1 entry.
C6E2           347      ;
C6E2 00         348      BRTBL      byt RTNCMD-1      ; 0
C6E3 08         349      OPTBL      byt SETCMD-1      ; 1n
C6E4 9B         350      byt BRCMD-1      ; 1
C6E5 0E         351      byt LDCMD-1      ; 2n
C6E6 9C         352      byt BNCCMD-1     ; 2
C6E7 17         353      byt STCMD-1      ; 3n
C6E8 AD         354      byt BCCMD-1      ; 3
C6E9 20         355      byt LD@CMD-1     ; 4n
C6EA B0         356      byt BPCMD-1      ; 4
C6EB 2A         357      byt ST@CMD-1     ; 5n
C6EC B5         358      byt BMCMD-1      ; 5
C6ED 39         359      byt LDD@CMD-1    ; 6n
C6EE BA         360      byt BZCMD-1      ; 6
C6EF 42         361      byt STD@CMD-1    ; 7n
C6F0 C1         362      byt BNZCMD-1     ; 7
C6F1 52         363      byt POP@CMD-1    ; 8n
C6F2 C8         364      byt BM1CMD-1     ; 8
C6F3 62         365      byt STP@CMD-1    ; 9n

```



```

C6F4 D1      366      byt BNM1CMD-1      ; 9
C6F5 6B      367      byt ADDCMD-1      ; An
C6F6 DA      368      byt SOUTCMD-1     ; A
C6F7 79      369      byt SUBCMD-1      ; Bn
C6F8 0A      370      byt RSCMD-1       ; B
C6F9 4B      371      byt POPD@CMD-1    ; Cn
C6FA 8F      372      byt BSCMD-1       ; C
C6FB 7B      373      byt CPRCMD-1      ; Dn
C6FC DF      374      byt RSNSCMD-1     ; D
C6FD 32      375      byt INRCMD-1      ; En
C6FE 0C      376      byt BSNSCMD-1     ; E
C6FF F6      377      byt DCRCMD-1      ; Fn
C700 E8      378      byt SJMPCMD-1     ; F
C701          379      ;
C701          380      ;
C701          381      ; The following routines must reside on the same page.
C701          382      ;
C701          383      ; RTN opcode returns to 6502 processing mode. Pop the
C701          384      ; SW16C return address, restore the 6502 registers, and
C701          385      ; return to 6502 mode using the SW16 program counter.
C701          386      ;
C701 68      387      RTNCMD    pla
C702 68      388              pla
C703          389      ;
C703 20 3F FF 390              jsr RESTORE
C706          391      ;
C706 4C 78 FA 392              jmp SW16RTN
C709          393      ;
C709          394      ;
C709          395      ; SET opcode gets a 2-byte immediate constant.
C709          396      ;
C709 10 A7    397      SETCMD    bpl SETCMD2      ; always taken
C70B          398      ;
C70B          399      ;
C70B          400      ; RS opcode is return from a BS subroutine call.
C70B          401      ;
C70B 10 B9    402      RSCMD     bpl RSCMD2      ; always taken
C70D          403      ;
C70D          404      ;
C70D          405      ; BSNS opcode is branch to subroutine that does not use a
C70D          406      ; return address stack.
C70D          407      ;
C70D 10 C8    408      BSNSCMD   bpl BSNSCMD2     ; always taken
C70F          409      ;
C70F          410      ;
C70F          411      ; LD opcode moves the contents of Rn to ACC.
C70F          412      ;
C70F B5 00    413      LDCMD     lda R0L,X
C711 85 00    414              sta R0L
C713          415      ;
C713 B5 01    416              lda R0H,X
C715 85 01    417              sta R0H
C717          418      ;
C717 60      419              rts
C718          420      ;
C718          421      ;
C718          422      ; ST opcode moves the contents of ACC to Rn.
C718          423      ;
C718 A5 00    424      STCMD     lda R0L
C71A 95 00    425              sta R0L,X
C71C          426      ;

```

```

C71C A5 01      427          lda R0H
C71E 95 01      428          sta R0H,X
C720            429      ;
C720 60         430          rts
C721            431      ;
C721            432      ;
C721            433      ; LD@ opcode loads the LO ACC indirectly from memory
C721            434      ; using the address in Rn.  HO ACC is cleared.  Make R0
C721            435      ; the prior register.
C721            436      ;
C721 A1 00      437 LD@CMD   lda (R0L,X)
C723 85 00      438          sta R0L
C725            439      ;
C725 A0 00      440          ldy #ZERO
C727 84 01      441          sty R0H
C729 F0 06      442          beq >4                ; always taken
C72B            443      ;
C72B            444      ;
C72B            445      ; ST@ opcode stores the contents of LO ACC indirectly to
C72B            446      ; memory using the address in Rn.  Make R0 the prior
C72B            447      ; register.  Fall into INRCMD.
C72B            448      ;
C72B A5 00      449 ST@CMD   lda R0L
C72D            450      ;
C72D 81 00      451 ST@CMD2  sta (R0L,X)
C72F            452      ;
C72F A0 00      453          ldy #ZERO
C731            454      ;
C731 84 1C      455 ^4      sty R14L
C733            456      ;
C733            457      ;
C733            458      ; INR opcode increments the register Rn.
C733            459      ;
C733 F6 00      460 INRCMD   inc R0L,X
C735 D0 02      461          bne >5
C737            462      ;
C737 F6 01      463          inc R0H,X
C739            464      ;
C739 60         465 ^5      rts
C73A            466      ;
C73A            467      ;
C73A            468      ; LDD@ opcode loads the ACC indirectly from memory using
C73A            469      ; the address in Rn.  The LO byte is loaded first.  Make
C73A            470      ; R0 the prior register.
C73A            471      ;
C73A 20 21 C7   472 LDD@CMD  jsr LD@CMD
C73D            473      ;
C73D A1 00      474          lda (R0L,X)
C73F 85 01      475          sta R0H
C741            476      ;
C741 90 F0      477          bcc INRCMD                ; always taken
C743            478      ;
C743            479      ;
C743            480      ; STD@ opcode stores the contents of the ACC indirectly to
C743            481      ; memory using the address in Rn.  LO ACC is stored first.
C743            482      ; Make R0 the prior register.
C743            483      ;
C743 20 2B C7   484 STD@CMD  jsr ST@CMD
C746            485      ;
C746 A5 01      486          lda R0H
C748 81 00      487          sta (R0L,X)

```

```

C74A          488 ;
C74A 90 E7    489         bcc INRCMD             ; always taken
C74C          490 ;
C74C          491 ;
C74C          492 ; POPD@ opcode decrements Rn and gets the HO byte
C74C          493 ; indirectly from memory using the address in Rn.
C74C          494 ; Fall into POP@CMD.
C74C          495 ;
C74C 20 F7 C7 496 POPD@CMD jsr DCRCMD
C74F          497 ;
C74F A1 00    498         lda (R0L,X)
C751 A8       499         tay
C752          500 ;
C752 2C 00 00 501         bit *-*
C755          502         dfs !-2
C753          503 ;
C753          504 ;
C753          505 ; POP@ opcode sets the HO byte to zero, then decrements Rn
C753          506 ; and gets the LO byte indirectly from memory using the
C753          507 ; address in Rn. Move the HO and LO bytes into the ACC.
C753          508 ; Make R0 the prior register.
C753          509 ;
C753 A0 00    510 POP@CMD ldy #ZERO
C755          511 ;
C755 20 F7 C7 512         jsr DCRCMD
C758          513 ;
C758 A1 00    514         lda (R0L,X)
C75A 85 00    515         sta R0L
C75C          516 ;
C75C 84 01    517         sty R0H
C75E          518 ;
C75E A0 00    519 ^6      ldy #ZERO
C760 84 1C    520         sty R14L
C762          521 ;
C762 60       522         rts
C763          523 ;
C763          524 ;
C763          525 ; STP@ opcode decrements Rn and stores the LO ACC
C763          526 ; indirectly to memory using the address in Rn. Make R0
C763          527 ; the prior register.
C763          528 ;
C763 20 F7 C7 529 STP@CMD jsr DCRCMD
C766          530 ;
C766 A5 00    531         lda R0L
C768 81 00    532         sta (R0L,X)
C76A          533 ;
C76A 90 F2    534         bcc <6             ; always taken
C76C          535 ;
C76C          536 ;
C76C          537 ; ADD opcode sets Y-reg to 0 so ACC + Rn -> ACC, carry
C76C          538 ; saved to HO status byte. Make R0 the prior register.
C76C          539 ;
C76C A5 00    540 ADDCMD  lda R0L
C76E 75 00    541         adc R0L,X
C770 85 00    542         sta R0L
C772          543 ;
C772 A5 01    544         lda R0H
C774 75 01    545         adc R0H,X
C776          546 ;
C776 A0 00    547         ldy #ZERO
C778 F0 0E    548         beq >7             ; always taken

```

```

C77A          549 ;
C77A          550 ;
C77A          551 ; SUB opcode sets Y-reg to 0 so ACC - Rn -> ACC, carry
C77A          552 ; saved to HO status byte. Make R0 the prior register.
C77A          553 ;
C77A A0 00    554 SUBCMD    ldy #ZERO
C77C          555 ;
C77C          556 ;
C77C          557 ; CPR opcode leaves Y-reg set to 13*2 (SET opcode index*2)
C77C          558 ; so ACC - Rn -> R13, carry saved to HO status byte. Make
C77C          559 ; R13 the prior register.
C77C          560 ;
C77C 38       561 CPRCMD    sec
C77D          562 ;
C77D A5 00    563             lda R0L
C77F F5 00    564             sbc R0L,X
C781 99 00 00 565             sta R0L,Y
C784          566 ;
C784 A5 01    567             lda R0H
C786 F5 01    568             sbc R0H,X
C788          569 ;
C788 99 01 00 570 ^7       sta R0H,Y
C78B          571 ;
C78B 84 1C    572             sty R14L
C78D 26 1D    573             rol R14H                ; save carry out bit
C78F          574 ;
C78F 60       575             rts
C790          576 ;
C790          577 ;
C790          578 ; BS opcode is branch to subroutine. Save the current
C790          579 ; SW16 program counter indirectly to memory using the
C790          580 ; address in R12. R15L is saved first, then R15H; finally
C790          581 ; R12 + 2 -> R12. Fall into BRCMD.
C790          582 ;
C790 A2 18    583 BSCMD    ldx #R12L
C792          584 ;
C792 A5 1E    585             lda R15L
C794 20 2D C7 586             jsr ST@CMD2
C797          587 ;
C797 A5 1F    588             lda R15H
C799 20 2D C7 589             jsr ST@CMD2
C79C          590 ;
C79C          591 ;
C79C          592 ; BR opcode is branch always. Fall into BNCCMD.
C79C          593 ;
C79C 18       594 BRCMD    clc
C79D          595 ;
C79D          596 ;
C79D          597 ; BNC opcode is branch if carry is clear from prior opcode.
C79D          598 ;
C79D B0 0E    599 BNCCMD    bcs >9
C79F          600 ;
C79F          601 ;
C79F          602 ; Get displacement byte. If it is negative set Y-reg to
C79F          603 ; minus one for 2's compliment addition. Y-reg = 0.
C79F          604 ;
C79F B1 1E    605             lda (R15L),Y
C7A1 10 01    606             bpl >8
C7A3          607 ;
C7A3 88       608             dey
C7A4          609 ;

```

```

C7A4      610 ;
C7A4      611 ; Add displacement to program counter.
C7A4      612 ;
C7A4 65 1E 613 ^8      adc R15L
C7A6 85 1E 614          sta R15L
C7A8      615 ;
C7A8 98    616          tya
C7A9      617 ;
C7A9 65 1F 618          adc R15H
C7AB 85 1F 619          sta R15H
C7AD      620 ;
C7AD 60    621 ^9      rts
C7AE      622 ;
C7AE      623 ;
C7AE      624 ; BC opcode is branch if carry is set from prior opcode.
C7AE      625 ;
C7AE B0 EC 626 BCCMD    bcs BRCMD
C7B0      627 ;
C7B0 60    628          rts
C7B1      629 ;
C7B1      630 ;
C7B1      631 ; BP opcode is branch if the value of the prior register is
C7B1      632 ; positive.
C7B1      633 ;
C7B1 B5 01 634 BPCMD    lda R0H,X
C7B3 10 E7 635          bpl BRCMD
C7B5      636 ;
C7B5 60    637          rts
C7B6      638 ;
C7B6      639 ;
C7B6      640 ; BM opcode is branch if the value of the prior register is
C7B6      641 ; negative.
C7B6      642 ;
C7B6 B5 01 643 BMCMD    lda R0H,X
C7B8 30 E2 644          bmi BRCMD
C7BA      645 ;
C7BA 60    646          rts
C7BB      647 ;
C7BB      648 ;
C7BB      649 ; BZ opcode is branch if the value of the prior register is
C7BB      650 ; zero.
C7BB      651 ;
C7BB B5 00 652 BZCMD    lda R0L,X
C7BD 15 01 653          ora R0H,X
C7BF F0 DB 654          beq BRCMD
C7C1      655 ;
C7C1 60    656          rts
C7C2      657 ;
C7C2      658 ;
C7C2      659 ; BNZ opcode is branch if the value of the prior register
C7C2      660 ; is not zero.
C7C2      661 ;
C7C2 B5 00 662 BNZCMD    lda R0L,X
C7C4 15 01 663          ora R0H,X
C7C6 D0 D4 664          bne BRCMD
C7C8      665 ;
C7C8 60    666          rts
C7C9      667 ;
C7C9      668 ;
C7C9      669 ; BM1 opcode is branch if the value of the prior register
C7C9      670 ; is negative one.

```

```

C7C9          671 ;
C7C9 B5 00    672 BM1CMD   lda R0L,X
C7CB 35 01    673          and R0H,X
C7CD          674 ;
C7CD 49 FF    675          eor #NEGONE
C7CF F0 CB    676          beq BRCMD
C7D1          677 ;
C7D1 60       678          rts
C7D2          679 ;
C7D2          680 ;
C7D2          681 ; BNM1 opcode is branch if the value of the prior register
C7D2          682 ; is not negative one.
C7D2          683 ;
C7D2 B5 00    684 BNM1CMD   lda R0L,X
C7D4 35 01    685          and R0H,X
C7D6          686 ;
C7D6 49 FF    687          eor #NEGONE
C7D8 D0 C2    688          bne BRCMD
C7DA          689 ;
C7DA 60       690          rts
C7DB          691 ;
C7DB          692 ;
C7DB          693 ; SOUT opcode sends the "chr" value to COUT.
C7DB          694 ;
C7DB B1 1E    695 SOUTCMD   lda (R15L),Y
C7DD          696 ;
C7DD 4C ED FD 697          jmp COUT
C7E0          698 ;
C7E0          699 ;
C7E0          700 ; RSNS opcode is return from a BSNS subroutine call that
C7E0          701 ; does not use a return address stack. Copy the saved SW16
C7E0          702 ; program counter in R12 to R15.
C7E0          703 ;
C7E0 A5 18    704 RSNSCMD   lda R12L
C7E2 85 1E    705          sta R15L
C7E4          706 ;
C7E4 A5 19    707          lda R12H
C7E6 85 1F    708          sta R15H
C7E8          709 ;
C7E8 60       710          rts
C7E9          711 ;
C7E9          712 ;
C7E9          713 ; SJMP opcode gets a 2-byte immediate address for the SW16
C7E9          714 ; program counter. Get the LO byte, then the HO byte, and
C7E9          715 ; save the address to R15. Make R15 the prior register.
C7E9          716 ; Fall into DCRCMD.
C7E9          717 ;
C7E9 B1 1E    718 SJMPCMD   lda (R15L),Y
C7EB AA       719          tax
C7EC          720 ;
C7EC C8       721          iny
C7ED          722 ;
C7ED B1 1E    723          lda (R15L),Y
C7EF          724 ;
C7EF 86 1E    725          stx R15L
C7F1 85 1F    726          sta R15H
C7F3          727 ;
C7F3 A2 1E    728          ldx #R15L
C7F5 86 1C    729          stx R14L
C7F7          730 ;
C7F7          731 ;

```

```
C7F7          732 ; DCR opcode decrements the register Rn.
C7F7          733 ;
C7F7 B5 00    734 DCRCMD   lda R0L,X
C7F9 D0 02    735          bne >1
C7FB          736 ;
C7FB D6 01    737          dec R0H,X
C7FD          738 ;
C7FD D6 00    739 ^1      dec R0L,X
C7FF          740 ;
C7FF 60       741          rts
C800          742 ;
C800          743 ;
C800          744          icl "C8.L"
```

```
LLOAD C8.L,A$4000
```

```

C800          1          ttl "ROM Source Code, C8.L"
C800          2          ;
C800          3          ;
C800          4          ; C8.L
C800          5          ;
C800          6          ;
C800          7          ; This entry point is only used by Pascal 1.0.
C800          8          ;
C800 4C B0 C9   9  PXINIT    jmp PXINIT2
C803         10          ;
C803         11          ;
C803         12          ; BASIC initialization. This is called by the 0xC3 space
C803         13          ; only after a PR#3 or a JSR $C300.
C803         14          ;
C803         15          ; If the language card is enabled and the ID byte does not
C803         16          ; match, the 0xF8 ROM is copied to the language card. If
C803         17          ; an 80 column card is detected it is enabled, otherwise
C803         18          ; 40 column is enabled. The screen is cleared and the
C803         19          ; character in A-reg is printed.
C803         20          ;
C803 20 F4 CE   21  BASCINIT  jsr COPYROM
C806 20 2A C8   22          jsr C3HOOKS
C809 20 2B CD   23          jsr DO40
C80C         24          ;
C80C A9 01      25          lda #M.MOUSE
C80E 8D FB 04   26          sta XMODE
C811         27          ;
C811 20 90 CA   28          jsr TESTCARD
C814 D0 08      29          bne >1
C816         30          ;
C816 06 21      31          asl WNDWDTH
C818         32          ;
C818 8D 01 C0   33          sta STR80ON
C81B 8D 0D C0   34          sta VID80ON
C81E         35          ;
C81E 8D 0F C0   36          ^1 sta ALTCHON
C821         37          ;
C821 20 74 CC   38          jsr XFF
C824         39          ;
C824 AC 7B 05   40          ldy OURCH
C827         41          ;
C827 4C 7E C8   42          jmp CXVIDCK4
C82A         43          ;
C82A         44          ;
C82A A9 07      45  C3HOOKS  lda #BASICOUT
C82C 85 36      46          sta CSWL
C82E         47          ;
C82E A9 C3      48          lda /BASICOUT
C830 85 37      49          sta CSWH
C832         50          ;
C832         51          ;
C832 A9 05      52  C3IN     lda #BASICIN
C834 85 38      53          sta KSWL
C836         54          ;
C836 A9 C3      55          lda /BASICIN
C838 85 39      56          sta KSWH
C83A         57          ;
C83A 60         58          rts
C83B         59          ;
C83B         60          ;

```



```

C83B      61 ; Moved GETKEY as CXKEYIN after ESCTABL where STAUx was.
C83B      62 ;
C83B      63 ;           dfs $C84D-*,ZERO      ; 18 bytes
C84D      64 ;
C84D      65 ;
C84D      66 ; Pascal 1.0 input entry point. Must be at 0xC84D.
C84D      67 ;
C84D 4C 50 C3 68 PXREAD    jmp JPREAD
C850      69 ;
C850      70 ;
C850      71 ; CSETUP compensates for changes to CV, CH, OURCH, and
C850      72 ; WNDWDTH. It updates the video firmware's versions of
C850      73 ; these values.
C850      74 ;
C850 A5 25 75 CSETUP    lda CV
C852 8D FB 05 76         sta OURCV
C855      77 ;
C855 A4 24 78         ldy CH
C857      79 ;
C857 CC 7B 04 80         cpy OLDCH
C85A F0 03 81         beq >1
C85C      82 ;
C85C 8C 7B 05 83         sty OURCH
C85F      84 ;
C85F 18 85 ^1        clc
C860      86 ;
C860 A5 21 87         lda WNDWDTH
C862 ED 7B 05 88         sbc OURCH
C865 B0 05 89         bcs >2
C867      90 ;
C867 A0 00 91         ldy #ZERO
C869 8C 7B 05 92         sty OURCH
C86C      93 ;
C86C AC 7B 05 94 ^2    ldy OURCH
C86F      95 ;
C86F 60 96         rts
C870      97 ;
C870      98 ;
C870      99 ; NEWVW support. CXNEWVW and CXNEWVW2 are used by the
C870     100 ; 0xF8 ROM to input and output characters while VID80 is
C870     101 ; ON. These routines are only called by the 0xC100 to
C870     102 ; 0xC2FF space so as not to cause possible conflict with
C870     103 ; other 0xC800 users.
C870     104 ;
C870 A4 35 105 CXNEWVW  ldy YSAV1
C872     106 ;
C872 18 107         clc
C873     108 ;
C873 B0 00 109         bcs *+2
C875     110         dfs !-1
C874     111 ;
C874 38 112 CXNEWVW2  sec
C875     113 ;
C875 8D 7B 06 114         sta XCHAR
C878     115 ;
C878 98 116         tya
C879 48 117         pha
C87A     118 ;
C87A 8A 119         txa
C87B 48 120         pha
C87C     121 ;

```

```

C87C B0 5E      122 CXVIDCK3 bcs >5
C87E           123 ;
C87E 20 50 C8   124 CXVIDCK4 jsr CSETUP
C881           125 ;
C881 AD 7B 06   126         lda XCHAR
C884           127 ;
C884 C9 8D      128         cmp #RETURN
C886 D0 18      129         bne >2
C888           130 ;
C888 AE 00 C0   131         ldx KEY
C88B 10 13      132         bpl >2
C88D           133 ;
C88D E0 93      134         cpx #CTRLS           ; CTRL-S check
C88F D0 0F      135         bne >2
C891           136 ;
C891 2C 10 C0   137         bit CLRKEY
C894           138 ;
C894 AE 00 C0   139 ^1         ldx KEY
C897 10 FB      140         bpl <1
C899           141 ;
C899 E0 83      142         cpx #CTRLC           ; CTRL-C check
C89B F0 03      143         beq >2
C89D           144 ;
C89D 2C 10 C0   145         bit CLRKEY
C8A0           146 ;
C8A0 29 7F      147 ^2         and #MSBCLR           ; clear MSB
C8A2           148 ;
C8A2 C9 20      149         cmp #$20           ; control character check
C8A4 B0 06      150         bcs >3
C8A6           151 ;
C8A6 20 D2 CA   152         jsr CTRLCHR2
C8A9           153 ;
C8A9 4C BD C8   154         jmp CTRLON
C8AC           155 ;
C8AC AD 7B 06   156 ^3         lda XCHAR
C8AF 20 38 CE   157         jsr STORCHAR
C8B2           158 ;
C8B2 C8         159         iny
C8B3           160 ;
C8B3 8C 7B 05   161         sty OURCH
C8B6           162 ;
C8B6 C4 21      163         cpy WNDWDTH
C8B8 90 03      164         bcc CTRLON
C8BA           165 ;
C8BA 20 51 CB   166         jsr XCR
C8BD           167 ;
C8BD AD FB 04   168 CTRLON    lda XMODE
C8C0 29 F7      169         and #NEGONE-M.CTL
C8C2 8D FB 04   170         sta XMODE
C8C5           171 ;
C8C5 AD 7B 05   172 BIORET    lda OURCH
C8C8           173 ;
C8C8 2C 1F C0   174         bit RDVID80
C8CB 10 02      175         bpl >4
C8CD           176 ;
C8CD A9 00      177         lda #ZERO
C8CF           178 ;
C8CF 85 24      179 ^4         sta CH
C8D1 8D 7B 04   180         sta OLDCH
C8D4           181 ;
C8D4 68         182         pla

```

```

C8D5 AA          183          tax
C8D6          184          ;
C8D6 68          185          pla
C8D7 A8          186          tay
C8D8          187          ;
C8D8 AD 7B 06    188          lda XCHAR
C8DB          189          ;
C8DB 60          190          rts
C8DC          191          ;
C8DC A4 24       192          ^5    ldy CH
C8DE          193          ;
C8DE AD 7B 06    194          lda XCHAR
C8E1 91 28       195          sta (BASL),Y
C8E3          196          ;
C8E3 20 50 C8    197          jsr CSETUP
C8E6          198          ;
C8E6 20 26 CE    199          XINPUT jsr INVERT
C8E9 20 8D C9    200          jsr CXKEYIN
C8EC          201          ;
C8EC 8D 7B 06    202          sta XCHAR
C8EF          203          ;
C8EF 20 26 CE    204          jsr INVERT
C8F2          205          ;
C8F2 A8          206          tay
C8F3          207          ;
C8F3 AD FB 04    208          lda XMODE
C8F6 29 08       209          and #M.CTL
C8F8 F0 CB       210          beq BIORET
C8FA          211          ;
C8FA C0 8D       212          cpy #RETURN
C8FC D0 08       213          bne >6
C8FE          214          ;
C8FE AD FB 04    215          lda XMODE
C901 29 F7       216          and #NEGONE-M.CTL
C903 8D FB 04    217          sta XMODE
C906          218          ;
C906 C0 9B       219          ^6    cpy #ESCAPE          ; ESC check
C908 F0 11       220          beq >7
C90A          221          ;
C90A C0 95       222          cpy #RARROW
C90C D0 B7       223          bne BIORET
C90E          224          ;
C90E AC 7B 05    225          ldy OURCH
C911 20 44 CE    226          jsr PICK
C914          227          ;
C914 09 80       228          ora #MSBSET          ; set MSB
C916 8D 7B 06    229          sta XCHAR
C919 D0 AA       230          bne BIORET          ; always taken
C91B          231          ;
C91B          232          ;
C91B          233          ; Escape sequence start. Only if one of the following
C91B          234          ; characters is pressed is it executed, otherwise it is
C91B          235          ; ignored.
C91B          236          ;
C91B          237          ; @ - home and clear
C91B          238          ; E - clear to end of line
C91B          239          ; F - clear to end of screen
C91B          240          ; I - move cursor up
C91B          241          ; J - move cursor left
C91B          242          ; K - move cursor right
C91B          243          ; M - move cursor down

```

```

C91B      244 ; 4 - enter 40 column mode
C91B      245 ; 8 - enter 80 column mode
C91B      246 ;
C91B      247 ; ^D - disable printing of control characters
C91B      248 ; ^E - enable printing of control characters
C91B      249 ; ^Q - quit, like PR#0 or IN#0
C91B      250 ;
C91B 20 B1 CE 251 ^7 jsr ESCON
C91E 20 8D C9 252 jsr CXKEYIN
C921 20 C4 CE 253 jsr ESCOFF
C924 20 01 FD 254 jsr UPRCASE
C927      255 ;
C927 29 7F      256 and #MSBCLR ; clear MSB
C929      257 ;
C929 A0 10      258 ldy #ESCTABL-ESCCHAR+1
C92B      259 ;
C92B D9 7C C9 260 ^8 cmp ESCTABL,Y
C92E F0 05      261 beq >9
C930      262 ;
C930 88      263 dey
C931 10 F8      264 bpl <8
C933      265 ;
C933 30 0F      266 bmi >1 ; always taken
C935      267 ;
C935 B9 6B C9 268 ^9 lda ESCCHAR,Y
C938 29 7F      269 and #MSBCLR ; clear MSB
C93A      270 ;
C93A 20 D6 CA 271 jsr CTRLCHR
C93D      272 ;
C93D B9 6B C9 273 lda ESCCHAR,Y
C940 30 D9      274 bmi <7
C942      275 ;
C942 10 A2      276 bpl XINPUT
C944      277 ;
C944 A8      278 ^1 tay
C945      279 ;
C945 AD FB 04 280 lda XMODE
C948      281 ;
C948 C0 11      282 cpy #17 ; was it Quit (DC1)?
C94A D0 0B      283 bne >2
C94C      284 ;
C94C 20 4A CD 285 jsr XNAK
C94F      286 ;
C94F A9 98      287 lda #CTRLX ; fake a ^X
C951 8D 7B 06 288 sta XCHAR
C954      289 ;
C954 4C C5 C8 290 jmp BIORET
C957      291 ;
C957 C0 05      292 ^2 cpy #5 ; was it a ^E?
C959 D0 08      293 bne >5
C95B      294 ;
C95B 29 DF      295 and #NEGONE-M.CTL2 ; enable control characters
C95D      296 ;
C95D 8D FB 04 297 ^3 sta XMODE
C960      298 ;
C960 4C E6 C8 299 ^4 jmp XINPUT
C963      300 ;
C963 C0 04      301 ^5 cpy #4 ; was it a ^D?
C965 D0 F9      302 bne <4
C967      303 ;
C967 09 20      304 ora #M.CTL2 ; disable control characters

```

```

C969 D0 F2      305          bne <3          ; always taken
C96B            306          ;
C96B            307          ;
C96B            308          ; These control characters perform the escape functions
C96B            309          ; when they execute. If the MSB is set, escape mode is
C96B            310          ; not exited after they execute their function.
C96B            311          ;
C96B 0C         312  ESCCHAR  hex 0C          ; @, formfeed
C96C 1C         313          hex 1C          ; A, FS
C96D 08         314          hex 08          ; B, BS
C96E 0A         315          hex 0A          ; C, LF
C96F 1F         316          hex 1F          ; D, US
C970 1D         317          hex 1D          ; E, GS
C971 0B         318          hex 0B          ; F, VT
C972 9F         319          hex 9F          ; I, US (stay ESC)
C973 88         320          hex 88          ; J, BS (stay ESC)
C974 9C         321          hex 9C          ; K, FS (stay ESC)
C975 8A         322          hex 8A          ; M, LF (stay ESC)
C976 11         323          hex 11          ; 4, DC1
C977 12         324          hex 12          ; 8, DC2
C978 88         325          hex 88          ; <-, BS (stay ESC)
C979 8A         326          hex 8A          ; DN, LF (stay ESC)
C97A 9F         327          hex 9F          ; UP, US (stay ESC)
C97B 9C         328          hex 9C          ; ->, FS (stay ESC)
C97C            329          ;
C97C 40         330  ESCTABL  asc '@'          ; handle old escapes
C97D 41         331          asc 'A'
C97E 42         332          asc 'B'
C97F 43         333          asc 'C'
C980 44         334          asc 'D'
C981 45         335          asc 'E'
C982 46         336          asc 'F'
C983 49         337          asc 'I'
C984 4A         338          asc 'J'
C985 4B         339          asc 'K'
C986 4D         340          asc 'M'
C987 34         341          asc '4'
C988 38         342          asc '8'
C989 08         343          byt LARROW^MSBSET ; left arrow
C98A 0A         344          byt DARROW^MSBSET ; down arrow
C98B 0B         345          byt UARROW^MSBSET ; up arrow
C98C 15         346          byt RARROW^MSBSET ; right arrow
C98D            347          ;
C98D            348          ;
C98D E6 4E      349  CXKEYIN  inc RNDL
C98F D0 02      350          bne >1
C991            351          ;
C991 E6 4F      352          inc RNDH
C993            353          ;
C993 AD 00 C0   354          ^1          lda KEY
C996 10 F5      355          bpl CXKEYIN
C998            356          ;
C998 2C 10 C0   357          bit CLRKEY
C99B            358          ;
C99B C9 FF      359          cmp #NEGONE
C99D D0 02      360          bne >2
C99F            361          ;
C99F A9 88      362          lda #LARROW
C9A1            363          ;
C9A1 60         364          ^2          rts
C9A2            365          ;

```

```

C9A2      366 ;
C9A2      367 ;           dfs $C9AA-*,ZERO           ; 8 bytes
C9AA      368 ;
C9AA      369 ;
C9AA      370 ; Pascal 1.0 output entry point.  Must be at 0xC9AA.
C9AA      371 ;
C9AA AD 7B 06 372 PXWRITE   lda XCHAR
C9AD 4C 56 C3 373           jmp JPWRITE
C9B0      374 ;
C9B0      375 ;
C9B0 A9 83 376 PXINIT2   lda #M.PASCAL+M.PAS1.0+M.MOUSE
C9B2 D0 02 377           bne >1                   ; always taken
C9B4      378 ;
C9B4      379 ;
C9B4 A9 81 380 PPINIT    lda #M.PASCAL+M.MOUSE
C9B6      381 ;
C9B6 48 382 ^1          pha
C9B7      383 ;
C9B7 20 90 CA 384           jsr TESTCARD
C9BA F0 04 385           beq >2
C9BC      386 ;
C9BC 68 387           pla
C9BD      388 ;
C9BD A2 09 389           ldx #9                   ; 'NO DEVICE' error
C9BF      390 ;
C9BF 60 391           rts
C9C0      392 ;
C9C0 68 393 ^2          pla
C9C1 8D FB 04 394          sta XMODE
C9C4      395 ;
C9C4 8D 01 C0 396          sta STR80ON
C9C7 8D 0D C0 397          sta VID80ON
C9CA 8D 0F C0 398          sta ALTCHON
C9CD      399 ;
C9CD 20 D4 CE 400          jsr PSETUP
C9D0 20 74 CC 401          jsr XFF
C9D3      402 ;
C9D3 4C 1F CA 403          jmp DOBASL
C9D6      404 ;
C9D6      405 ;
C9D6      406 ; Character is always returned with its MSB off.
C9D6      407 ;
C9D6 20 D4 CE 408 PPREAD   jsr PSETUP
C9D9 20 8D C9 409          jsr CXKEYIN
C9DC      410 ;
C9DC 29 7F 411          and #MSBCLR             ; clear MSB
C9DE 8D 7B 06 412          sta XCHAR
C9E1      413 ;
C9E1 A2 00 414          ldx #ZERO
C9E3      415 ;
C9E3 AD FB 04 416          lda XMODE
C9E6 29 02 417          and #M.PAS1.0
C9E8 F0 02 418          beq >1
C9EA      419 ;
C9EA A2 C3 420          ldx /C3SPACE
C9EC      421 ;
C9EC AD 7B 06 422 ^1          lda XCHAR
C9EF      423 ;
C9EF 60 424          rts
C9F0      425 ;
C9F0      426 ;

```

```

C9F0 29 7F      427 PPWRITE and #MSBCLR      ; clear MSB
C9F2 AA        428 tax
C9F3          429 ;
C9F3 20 D4 CE   430 jsr PSETUP
C9F6          431 ;
C9F6 A9 08      432 lda #M.GOXY
C9F8          433 ;
C9F8 2C FB 04   434 bit XMODE
C9FB D0 32      435 bne >2
C9FD          436 ;
C9FD 8A        437 txa
C9FE          438 ;
C9FE 2C 2E CA   439 bit BITBYT60
CA01 F0 50      440 beq >3
CA03          441 ;
CA03 AC 7B 05   442 ldy OURCH
CA06          443 ;
CA06 24 32      444 bit INVFLG
CA08 10 02      445 bpl >1
CA0A          446 ;
CA0A 09 80      447 ora #MSBSET      ; set MSB
CA0C          448 ;
CA0C 20 70 CE   449 ^1 jsr STORIT
CA0F          450 ;
CA0F C8        451 iny
CA10 8C 7B 05   452 sty OURCH
CA13          453 ;
CA13 C4 21      454 cpy WNDWDTH
CA15 90 08      455 bcc DOBASL
CA17          456 ;
CA17 A9 00      457 lda #ZERO      ; carriage return
CA19 8D 7B 05   458 sta OURCH
CA1C          459 ;
CA1C 20 D8 CB   460 jsr XLF
CA1F          461 ;
CA1F          462 ;
CA1F A5 28      463 DOBASL lda BASL
CA21 8D 7B 07   464 sta OLDBASL
CA24          465 ;
CA24 A5 29      466 lda BASH
CA26 8D FB 07   467 sta OLDBASH
CA29          468 ;
CA29          469 ;
CA29 20 1F CE   470 PWRITER jsr PASINV
CA2C          471 ;
CA2C A2 00      472 ldx #ZERO      ; return, no error
CA2E          473 ;
CA2E 60        474 BITBYT60 rts
CA2F          475 ;
CA2F          476 ;
CA2F          477 ; Handle GOTOXY stuff.
CA2F          478 ;
CA2F 20 1F CE   479 ^2 jsr PASINV
CA32          480 ;
CA32 8A        481 txa
CA33          482 ;
CA33 38        483 sec
CA34          484 ;
CA34 E9 20      485 sbc #$20      ; make binary
CA36          486 ;
CA36 2C FB 06   487 bit XCOORD

```

```

CA39 30 30      488      bmi >5
CA3B           489      ;
CA3B 8D FB 05   490      sta OURCV
CA3E 85 25      491      sta CV
CA40           492      ;
CA40 20 BA CA   493      jsr XBASCALC
CA43           494      ;
CA43 AD FB 06   495      lda XCOORD
CA46 8D 7B 05   496      sta OURCH
CA49           497      ;
CA49 A9 F7      498      lda #NEGONE-M.GOXY
CA4B 2D FB 04   499      and XMODE
CA4E 8D FB 04   500      sta XMODE
CA51 D0 CC      501      bne DOBASL          ; always taken
CA53           502      ;
CA53 20 1F CE   503      ^3      jsr PASINV
CA56           504      ;
CA56 8A         505      txa
CA57           506      ;
CA57 C9 1E      507      cmp #$1E          ; GOTOXY command?
CA59 F0 06      508      beq >4
CA5B           509      ;
CA5B 20 D6 CA   510      jsr CTRLCHR
CA5E           511      ;
CA5E 4C 1F CA   512      jmp DOBASL
CA61           513      ;
CA61           514      ;
CA61           515      ; Start the GOTOXY sequence.
CA61           516      ;
CA61 A9 08      517      ^4      lda #M.GOXY
CA63 0D FB 04   518      ora XMODE
CA66 8D FB 04   519      sta XMODE
CA69           520      ;
CA69 A9 FF      521      lda #NEGONE
CA6B           522      ;
CA6B 8D FB 06   523      ^5      sta XCOORD
CA6E           524      ;
CA6E 4C 29 CA   525      jmp PWRITER
CA71           526      ;
CA71           527      ;
CA71           528      ; 65C02 opcode remapping to MNEML/MNEMR index.
CA71           529      ;
CA71           530      ; If the opcode is found in OPTBLC, then OPTBLL contains
CA71           531      ; the correct MNEML/MNEMR index.
CA71           532      ;
CA71 1A 3A 89   533      OPTBLC   hex 1A3A8903
CA74 03
CA75 5A 7A 14   534      hex 5A7A141C
CA78 1C
CA79 64 74 9C   535      hex 64749C9E
CA7C 9E
CA7D           536      ;
CA7D 37 36 21   537      OPTBLL   hex 37362140
CA80 40
CA81 41 42 43   538      hex 41424343
CA84 43
CA85 44 44 44   539      hex 44444444
CA88 44
CA89           540      ;
CA89           541      ;
CA89           542      ; Enable slot 3 and test for a ROM card.  If none, test

```



```

CA89          543 ; for an 80 column card.  If none, return with a BNE.
CA89          544 ;
CA89 20 B6 F8 545 TSTROMCD jsr TESTROM
CA8C D0 02    546 bne TESTCARD
CA8E          547 ;
CA8E C8       548 iny
CA8F          549 ;
CA8F 60       550 rts
CA90          551 ;
CA90          552 ;
CA90 AD 1C C0 553 TESTCARD lda RDPAGE2
CA93 0A       554 asl
CA94          555 ;
CA94 A9 88    556 lda #$88 ; test character
CA96          557 ;
CA96 2C 18 C0 558 bit RDSTR80
CA99          559 ;
CA99 8D 01 C0 560 sta STR80ON
CA9C          561 ;
CA9C 08       562 php ; save N and C flags
CA9D          563 ;
CA9D 8D 55 C0 564 sta PAGE2ON
CAA0          565 ;
CAA0 AC 00 04 566 ldy TEXTPG1
CAA3          567 ;
CAA3 8D 00 04 568 sta TEXTPG1
CAA6          569 ;
CAA6 AD 00 04 570 lda TEXTPG1
CAA9          571 ;
CAA9 8C 00 04 572 sty TEXTPG1
CAAC          573 ;
CAAC 28       574 plp ; recall status
CAAD B0 03    575 bcs >1
CAAF          576 ;
CAAF 8D 54 C0 577 sta PAGE1ON
CAB2          578 ;
CAB2 30 03    579 ^1 bmi >2
CAB4          580 ;
CAB4 8D 00 C0 581 sta STR80OFF
CAB7          582 ;
CAB7 C9 88    583 ^2 cmp #$88 ; same character?
CAB9          584 ;
CAB9 60       585 rts
CABA          586 ;
CABA          587 ;
CABA          588 ; XBASCALC is the same as the original BASCALC at 0xFBC1.
CABA          589 ;
CABA          590 ; 0 <= line number <= 23. A-reg = 000abcde. Generate
CABA          591 ; BASL = eabab000, BASH = 000001cd.
CABA          592 ;
CABA 48       593 XBASCALC pha
CABB          594 ;
CABB 4A       595 lsr
CABC 29 03    596 and #3
CABE          597 ;
CABE 09 04    598 ora #4
CAC0 85 29    599 sta BASH
CAC2          600 ;
CAC2 68       601 pla
CAC3 29 18    602 and #$18
CAC5          603 ;

```

```

CAC5 90 02      604      bcc >1
CAC7            605      ;
CAC7 69 7F      606      adc #MSBSET-1      ; carry adds 1
CAC9            607      ;
CAC9 85 28      608      ^1      sta BASL
CACB            609      ;
CACB 0A         610      asl
CACC 0A         611      asl
CACD            612      ;
CACD 05 28      613      ora BASL
CACF 85 28      614      sta BASL
CAD1            615      ;
CAD1 60         616      rts      ; carry must be clear
CAD2            617      ;
CAD2            618      ;
CAD2 2C 2E CA   619      CTRLCHR2 bit BITBYT60      ; used to set V-flag
CAD5            620      ;
CAD5 50 00      621      bvc *+2
CAD7            622      dfs !-1
CAD6            623      ;
CAD6 B8         624      CTRLCHR clv      ; clear V-flag, ignore M.CTL
CAD7            625      ;
CAD7 8D 7B 07   626      sta XTEMP1
CADA 48         627      pha
CADB            628      ;
CADB 98         629      tya
CADC 48         630      pha
CADD            631      ;
CADD AC 7B 07   632      ldy XTEMP1
CAE0            633      ;
CAE0 C0 05      634      cpy #5      ; is it NUL..EOT?
CAE2 90 13      635      bcc >1
CAE4            636      ;
CAE4 B9 B4 CB   637      lda CTRLADRH-5,Y
CAE7 F0 0E      638      beq >1      ; CTRL not implemented
CAE9            639      ;
CAE9 50 12      640      bvc >3      ; CTRLCHR always executes
CAEB 30 10      641      bmi >3      ; CR, BEL, LF, BS always done
CAED            642      ;
CAED 8D 7B 07   643      sta XTEMP1
CAF0            644      ;
CAF0 AD FB 04   645      lda XMODE
CAF3 29 28      646      and #M.CTL+M.CTL2
CAF5 F0 03      647      beq >2
CAF7            648      ;
CAF7 38         649      ^1      sec
CAF8 B0 09      650      bcs >4      ; always taken
CAFA            651      ;
CAFA AD 7B 07   652      ^2      lda XTEMP1
CAFD            653      ;
CAFD 09 80      654      ^3      ora #MSBSET      ; set MSB
CAFF            655      ;
CAFF 20 07 CB   656      jsr CTRLXFER
CB02            657      ;
CB02 18         658      clc
CB03            659      ;
CB03 68         660      ^4      pla
CB04 A8         661      tay
CB05            662      ;
CB05 68         663      pla
CB06            664      ;

```

```

CB06 60          665          rts
CB07          666          ;
CB07          667          ;
CB07          668          ; Now execute the subroutine.
CB07          669          ;
CB07 48          670 CTRLXFER pha
CB08          671          ;
CB08 B9 99 CB    672          lda CTRLADRL-5,Y
CB0B 48          673          pha
CB0C          674          ;
CB0C 60          675          rts
CB0D          676          ;
CB0D          677          ;
CB0D          678          ; Turn cursor on for Pascal only.
CB0D          679          ;
CB0D AD FB 04    680 PCURON   lda XMODE
CB10 10 05       681          bpl >2
CB12          682          ;
CB12 29 EF       683          and #$EF
CB14          684          ;
CB14 8D FB 04    685 ^1      sta XMODE
CB17          686          ;
CB17 60          687 ^2      rts
CB18          688          ;
CB18 AD FB 04    689 PCUROFF  lda XMODE
CB1B 10 FA       690          bpl <2
CB1D          691          ;
CB1D 09 10       692          ora #$10
CB1F D0 F3       693          bne <1          ; always taken
CB21          694          ;
CB21          695          ;
CB21          696          ; Removed XBELL and CXWAIT. The same code is at 0xFBDD.
CB21          697          ;
CB21          698          dfs $CB40-*,ZERO    ; 31 bytes
CB40          699          ;
CB40          700          ;
CB40          701          ; Execute backspace.
CB40          702          ;
CB40 CE 7B 05    703 XBS     dec OURCH
CB43 10 0B       704          bpl >1
CB45          705          ;
CB45 A5 21       706          lda WNDWDTH
CB47 8D 7B 05    707          sta OURCH
CB4A          708          ;
CB4A CE 7B 05    709          dec OURCH
CB4D          710          ;
CB4D 20 79 CB    711          jsr XUS
CB50          712          ;
CB50 60          713 ^1      rts
CB51          714          ;
CB51          715          ;
CB51          716          ; Execute carriage return.
CB51          717          ;
CB51 A9 00       718 XCR     lda #ZERO
CB53 8D 7B 05    719          sta OURCH
CB56          720          ;
CB56 AD FB 04    721          lda XMODE
CB59 30 03       722          bmi >2          ; Pascal, avoid auto LF
CB5B          723          ;
CB5B 20 D8 CB    724          jsr XLF
CB5E          725          ;

```

```

CB5E 60          726 ^2      rts
CB5F            727 ;
CB5F            728 ;
CB5F            729 ; Execute HOME.
CB5F            730 ;
CB5F A5 22      731 XEM      lda WNDTOP
CB61 85 25      732          sta CV
CB63            733 ;
CB63 A9 00      734          lda #ZERO
CB65 8D 7B 05   735          sta OURCH
CB68            736 ;
CB68 4C FB CD   737          jmp XVTAB2
CB6B            738 ;
CB6B            739 ;
CB6B            740 ; Execute forward space.
CB6B            741 ;
CB6B EE 7B 05   742 XFS      inc OURCH
CB6E            743 ;
CB6E AD 7B 05   744          lda OURCH
CB71 C5 21      745          cmp WNDWDTH
CB73 90 03      746          bcc >3
CB75            747 ;
CB75 20 51 CB   748          jsr XCR
CB78            749 ;
CB78 60          750 ^3      rts
CB79            751 ;
CB79            752 ;
CB79            753 ; Execute reverse linefeed.
CB79            754 ;
CB79 A5 22      755 XUS      lda WNDTOP
CB7B C5 25      756          cmp CV
CB7D B0 1E      757          bcs >7
CB7F            758 ;
CB7F C6 25      759          dec CV
CB81            760 ;
CB81 4C FB CD   761          jmp XVTAB2
CB84            762 ;
CB84            763 ;
CB84            764 ; Execute NORMAL video.
CB84            765 ;
CB84 AD FB 04   766 XSO      lda XMODE
CB87 10 02      767          bpl >4
CB89            768 ;
CB89 29 FB      769          and #NEGONE-M.VMODE ; set NORMAL
CB8B            770 ;
CB8B A0 FF      771 ^4      ldy #NEGONE
CB8D D0 09      772          bne >6                ; always taken
CB8F            773 ;
CB8F            774 ;
CB8F            775 ; Execute INVERSE video.
CB8F            776 ;
CB8F AD FB 04   777 XSI      lda XMODE
CB92 10 02      778          bpl >5
CB94            779 ;
CB94 09 04      780          ora #M.VMODE                ; set INVERSE
CB96            781 ;
CB96 A0 7F      782 ^5      ldy #INVRSE80
CB98            783 ;
CB98 8D FB 04   784 ^6      sta XMODE
CB9B            785 ;
CB9B 84 32      786          sty INVFLG

```

```

CB9D          787 ;
CB9D 60       788 ^7      rts
CB9E          789 ;
CB9E          790 ;
CB9E 0C       791 CTRLADRL byt PCURON-1      ; ENQ
CB9F 17       792          byt PCUROFF-1     ; ACK
CBA0 DC       793          byt RINGBELL-1 ; BEL
CBA1 3F       794          byt XBS-1      ; BS
CBA2 00       795          byt ZERO      ; HT, not imlemented
CBA3 D7       796          byt XLF-1     ; LF
CBA4 76       797          byt XVT-1     ; VT
CBA5 73       798          byt XFF-1     ; FF
CBA6 50       799          byt XCR-1     ; CR
CBA7 83       800          byt XSO-1     ; SO
CBA8 8E       801          byt XSI-1     ; SI
CBA9 00       802          byt ZERO      ; DLE, not implemented
CBAA E6       803          byt XDC1-1    ; DC1
CBAB F8       804          byt XDC2-1    ; DC2
CBAC 00       805          byt ZERO      ; DC3, not implemented
CBAD 00       806          byt ZERO      ; DC4, not implemented
CBAE 49       807          byt XNAK-1    ; NAK
CBAF D3       808          byt SCROLLDN-1 ; SYN
CBB0 EA       809          byt SCROLLUP-1 ; ETB
CBB1 39       810          byt MOUSEOFF-1
CBB2 5E       811          byt XEM-1     ; EM
CBB3 92       812          byt XSUB-1    ; SUB
CBB4 40       813          byt MOUSEON-1
CBB5 6A       814          byt XFS-1     ; FS
CBB6 96       815          byt XGS-1     ; GS
CBB7 00       816          byt ZERO      ; RS, not implemented
CBB8 78       817          byt XUS-1     ; US
CBB9          818 ;
CBB9 4B       819 CTRLADRH hby PCURON-$8001 ; ENQ
CBBA 4B       820          hby PCUROFF-$8001 ; ACK
CBBB FB       821          hby RINGBELL-1 ; BEL
CBBC CB       822          hby XBS-1      ; BS
CBBD 00       823          byt ZERO      ; HT, not imlemented
CBBE CB       824          hby XLF-1     ; LF
CBBF 4C       825          hby XVT-$8001 ; VT
CBC0 4C       826          hby XFF-$8001 ; FF
CBC1 CB       827          hby XCR-1     ; CR
CBC2 4B       828          hby XSO-$8001 ; SO
CBC3 4B       829          hby XSI-$8001 ; SI
CBC4 00       830          byt ZERO      ; DLE, not implemented
CBC5 4C       831          hby XDC1-$8001 ; DC1
CBC6 4C       832          hby XDC2-$8001 ; DC2
CBC7 00       833          byt ZERO      ; DC3, not implemented
CBC8 00       834          byt ZERO      ; DC4, not implemented
CBC9 4D       835          hby XNAK-$8001 ; NAK
CBCA 4B       836          hby SCROLLDN-$8001 ; SYN
CBCB 4B       837          hby SCROLLUP-$8001 ; ETB
CBCC 4D       838          hby MOUSEOFF-$8001
CBCD 4B       839          hby XEM-$8001 ; EM
CBCE 4C       840          hby XSUB-$8001 ; SUB
CBCF 4D       841          hby MOUSEON-$8001
CBD0 4B       842          hby XFS-$8001 ; FS
CBD1 4C       843          hby XGS-$8001 ; GS
CBD2 00       844          byt ZERO      ; RS, not implemented
CBD3 4B       845          hby XUS-$8001 ; US
CBD4          846 ;
CBD4          847 ;

```

CBD4 848 icl "CC.L"

LLOAD CC.L,A\$4000

```

CBD4      1          ttl "ROM Source Code, CC.L"
CBD4      2      ;
CBD4      3      ;
CBD4      4      ; CC.L
CBD4      5      ;
CBD4      6      ;
CBD4      7      ; SCROLLDN and SCROLLUP scroll the screen down and up,
CBD4      8      ; respectively, depending on the value of Y-reg. It
CBD4      9      ; scrolls within windows having even or odd edges for both
CBD4     10      ; 40 and 80 columns down to 1 characters wide.
CBD4     11      ;
CBD4 A0 00     12 SCROLLDN ldy #ZERO          ; scroll down
CBD6 F0 15     13          beq >2          ; always taken
CBD8     14      ;
CBD8     15      ;
CBD8     16      ; Execute linefeed.
CBD8     17      ;
CBD8 E6 25     18 XLF          inc CV
CBDA     19      ;
CBDA A5 25     20          lda CV
CBDC 8D FB 05  21          sta OURCV
CBDF     22      ;
CBDF C5 23     23          cmp WNDBTM
CBE1 B0 03     24          bcs >1
CBE3     25      ;
CBE3 4C 00 CE  26          jmp XVTAB3
CBE6     27      ;
CBE6 CE FB 05  28 ^1          dec OURCV
CBE9     29      ;
CBE9 C6 25     30          dec CV
CBEB     31      ;
CBEB     32      ;
CBEB A0 01     33 SCROLLUP ldy #1          ; scroll up
CBED     34      ;
CBED 8A     35 ^2          txa
CBEE 48     36          pha
CBEF     37      ;
CBEF 8C 7B 07  38          sty XTEMP1
CBF2     39      ;
CBF2 A5 21     40          lda WNDWDTH
CBF4 48     41          pha
CBF5     42      ;
CBF5 2C 1F C0  43          bit RDVID80
CBF8 10 1C     44          bpl >5
CBFA     45      ;
CBFA 8D 01 C0  46          sta STR80ON
CBFD 4A     47          lsr
CBFE AA     48          tax
CBFF     49      ;
CBFF A5 20     50          lda WNDLFT
CC01 4A     51          lsr
CC02     52      ;
CC02 B8     53          clv          ; left edge even
CC03     54      ;
CC03 90 03     55          bcc >3
CC05     56      ;
CC05 2C 2E CA  57          bit BITBYT60      ; left edge odd
CC08     58      ;
CC08 2A     59 ^3          rol
CC09 45 21     60          eor WNDWDTH

```

```

CC0B 4A          61      lsr
CC0C 70 03       62      bvs >4
CC0E             63      ;
CC0E B0 01       64      bcs >4
CC10             65      ;
CC10 CA          66      dex
CC11             67      ;
CC11 86 21       68      ^4      stx WNDWDTH
CC13             69      ;
CC13 AD 1F C0    70      lda RDVID80
CC16             71      ;
CC16 08          72      ^5      php                ; save N, Z, V flags
CC17             73      ;
CC17 A6 22       74      ldx WNDTOP
CC19             75      ;
CC19 98          76      tya
CC1A D0 03       77      bne >6                ; direction
CC1C             78      ;
CC1C A6 23       79      ldx WNDBTM
CC1E CA          80      dex
CC1F             81      ;
CC1F 8A          82      ^6      txa
CC20             83      ;
CC20 20 00 CE    84      jsr XVTAB3
CC23             85      ;
CC23 A5 28       86      ^7      lda BASL
CC25 85 2A       87      sta BAS2L
CC27             88      ;
CC27 A5 29       89      lda BASH
CC29 85 2B       90      sta BAS2H
CC2B             91      ;
CC2B AD 7B 07    92      lda XTEMP1
CC2E F0 32       93      beq >5
CC30             94      ;
CC30 E8          95      inx
CC31             96      ;
CC31 E4 23       97      cpx WNDBTM
CC33 B0 32       98      bcs >6
CC35             99      ;
CC35 8A          100     ^8      txa
CC36             101     ;
CC36 20 00 CE    102     jsr XVTAB3
CC39             103     ;
CC39 A4 21       104     ldy WNDWDTH
CC3B             105     ;
CC3B 28          106     plp                ; recall status
CC3C 08          107     php
CC3D 10 1E       108     bpl >4                ; only do 40 columns
CC3F             109     ;
CC3F AD 55 C0    110     lda PAGE2ON
CC42             111     ;
CC42 98          112     tya
CC43 F0 07       113     beq >1
CC45             114     ;
CC45 B1 28       115     ^9      lda (BASL),Y        ; scroll even bytes
CC47 91 2A       116     sta (BAS2L),Y
CC49             117     ;
CC49 88          118     dey
CC4A D0 F9       119     bne <9
CC4C             120     ;
CC4C 70 04       121     ^1      bvs >2                ; if odd, skip

```



```

CC4E      122 ;
CC4E B1 28 123      lda (BASL),Y
CC50 91 2A 124      sta (BAS2L),Y
CC52      125 ;
CC52 AD 54 C0 126 ^2      lda PAGE1ON
CC55      127 ;
CC55 A4 21 128      ldy WNDWDTH
CC57      129 ;
CC57 B0 04 130      bcs >4
CC59      131 ;
CC59 B1 28 132 ^3      lda (BASL),Y      ; scroll odd bytes
CC5B 91 2A 133      sta (BAS2L),Y
CC5D      134 ;
CC5D 88 135 ^4      dey
CC5E 10 F9 136      bpl <3
CC60      137 ;
CC60 30 C1 138      bmi <7      ; always taken
CC62      139 ;
CC62 CA 140 ^5      dex
CC63      141 ;
CC63 E4 22 142      cpx WNDTOP
CC65 10 CE 143      bpl <8
CC67      144 ;
CC67 28 145 ^6      plp      ; recall status
CC68      146 ;
CC68 68 147      pla
CC69 85 21 148      sta WNDWDTH
CC6B      149 ;
CC6B 20 93 CC 150      jsr XSUB
CC6E 20 FB CD 151      jsr XVTAB2
CC71      152 ;
CC71 68 153      pla
CC72 AA 154      tax
CC73      155 ;
CC73 60 156      rts
CC74      157 ;
CC74      158 ;
CC74      159 ; Execute clear (moved here).
CC74      160 ;
CC74 20 5F CB 161 XFF      jsr XEM
CC77      162 ;
CC77      163 ;      jmp XVT
CC77      164 ;
CC77      165 ;
CC77      166 ; Execute CLR to EOS.
CC77      167 ;
CC77 20 97 CC 168 XVT      jsr XGS
CC7A      169 ;
CC7A A5 25 170      lda CV
CC7C 48 171      pha
CC7D 10 06 172      bpl >2
CC7F      173 ;
CC7F 20 00 CE 174 ^1      jsr XVTAB3
CC82 20 93 CC 175      jsr XSUB
CC85      176 ;
CC85 E6 25 177 ^2      inc CV
CC87      178 ;
CC87 A5 25 179      lda CV
CC89 C5 23 180      cmp WNDBTM
CC8B 90 F2 181      bcc <1
CC8D      182 ;

```

```

CC8D 68          183          pla
CC8E 85 25       184          sta CV
CC90             185          ;
CC90 4C FB CD    186          jmp XVTAB2
CC93             187          ;
CC93             188          ;
CC93             189          ; Execute clear line.
CC93             190          ;
CC93 A0 00       191 XSUB      ldy #ZERO
CC95 F0 03       192          beq XGSEOLZ          ; always taken
CC97             193          ;
CC97             194          ;
CC97             195          ; Execute clear to EOL.
CC97             196          ;
CC97 AC 7B 05    197 XGS       ldy OURCH
CC9A             198          ;
CC9A             199          ;
CC9A A5 32       200 XGSEOLZ   lda INVFLG
CC9C 29 80       201          and #MSBSET
CC9E 09 20       202          ora #' '          ; make it a blank
CCA0             203          ;
CCA0 2C 1F C0    204          bit RDVID80
CCA3 30 15       205          bmi CLR80
CCA5             206          ;
CCA5             207          ;
CCA5             208          ; Clear to end of line for 40 colums.
CCA5             209          ;
CCA5 91 28       210 CLR40     sta (BASL),Y
CCA7             211          ;
CCA7 C8          212          iny
CCA8             213          ;
CCA8 C4 21       214          cpy WNDWDTH
CCAA 90 F9       215          bcc CLR40
CCAC             216          ;
CCAC 60          217          rts
CCAD             218          ;
CCAD             219          ;
CCAD 86 2A       220 CLRHALF   stx BAS2L
CCAF             221          ;
CCAF A2 D8       222          ldx #!-40
CCB1 A0 14       223          ldy #20
CCB3             224          ;
CCB3 A5 32       225          lda INVFLG
CCB5 29 A0       226          and #" "
CCB7             227          ;
CCB7 4C D2 CC    228          jmp CLR2
CCBA             229          ;
CCBA             230          ;
CCBA             231          ; Clear to end of line for 80 columns.
CCBA             232          ;
CCBA 86 2A       233 CLR80     stx BAS2L
CCBC             234          ;
CCBC 48          235          pha
CCBD             236          ;
CCBD 98          237          tya
CCBE 48          238          pha
CCBF             239          ;
CCBF 38          240          sec
CCC0             241          ;
CCC0 E5 21       242          sbc WNDWDTH          ; count = WNDWDTH - Y-reg - 1
CCC2 AA          243          tax

```

```

CCC3      244 ;
CCC3 98    245 tya
CCC4 4A    246 lsr
CCC5 A8    247 tay
CCC6      248 ;
CCC6 68    249 pla
CCC7 45 20 250 eor WNDLFT ; get starting page
CCC9 6A    251 ror
CCCA B0 03 252 bcs >4
CCCC      253 ;
CCCC 10 01 254 bpl >4
CCCE      255 ;
CCCE C8    256 iny
CCCF      257 ;
CCCF 68    258 ^4 pla
CCD0      259 ;
CCD0 B0 0B 260 bcs >5
CCD2      261 ;
CCD2      262 ;
CCD2 2C 55 C0 263 CLR2 bit PAGE2ON
CCD5      264 ;
CCD5 91 28 265 sta (BASL),Y
CCD7      266 ;
CCD7 2C 54 C0 267 bit PAGE1ON
CCDA      268 ;
CCDA E8    269 inx
CCDB F0 06 270 beq >6
CCDD      271 ;
CCDD 91 28 272 ^5 sta (BASL),Y
CCDF      273 ;
CCDF C8    274 iny
CCE0      275 ;
CCE0 E8    276 inx
CCE1 D0 EF 277 bne CLR2
CCE3      278 ;
CCE3 A6 2A 279 ^6 ldx BAS2L
CCE5      280 ;
CCE5 38    281 sec
CCE6      282 ;
CCE6 60    283 rts
CCE7      284 ;
CCE7      285 ;
CCE7      286 ; Execute 40 column mode.
CCE7      287 ;
CCE7 AD FB 04 288 XDC1 lda XMODE
CCEA 30 4D 289 bmi >4
CCEC      290 ;
CCEC 20 2E CD 291 XDC1.2 jsr SETTOP
CCEF      292 ;
CCEF 2C 1F C0 293 bit RDVID80
CCF2 10 12 294 bpl >1
CCF4      295 ;
CCF4 20 8E CD 296 jsr SCR84
CCF7 90 0D 297 bcc >1
CCF9      298 ;
CCF9      299 ;
CCF9      300 ; Execute 80 column mode.
CCF9      301 ;
CCF9 20 90 CA 302 XDC2 jsr TESTCARD
CCFC D0 3B 303 bne >4
CCFE      304 ;

```

```

CCFE 2C 1F C0 305 bit RDVID80
CD01 30 03 306 bmi >1
CD03 307 ;
CD03 20 C1 CD 308 jsr SCR48 ; make it 80 column mode
CD06 309 ;
CD06 18 310 ^1 clc
CD07 311 ;
CD07 AD 7B 05 312 lda OURCH
CD0A 65 20 313 adc WNDLFT
CD0C 314 ;
CD0C 2C 1F C0 315 bit RDVID80
CD0F 30 06 316 bmi >2
CD11 317 ;
CD11 C9 28 318 cmp #40 ; in 40 column mode
CD13 90 02 319 bcc >2
CD15 320 ;
CD15 A9 27 321 lda #39
CD17 322 ;
CD17 8D 7B 05 323 ^2 sta OURCH
CD1A 85 24 324 sta CH
CD1C 325 ;
CD1C A5 25 326 lda CV
CD1E 20 BA CA 327 jsr XBASCALC
CD21 328 ;
CD21 2C 1F C0 329 bit RDVID80
CD24 10 05 330 bpl DO40
CD26 331 ;
CD26 20 6E CD 332 jsr FULL80
CD29 F0 03 333 beq SETTOP ; always taken
CD2B 334 ;
CD2B 335 ;
CD2B 20 6A CD 336 DO40 jsr FULL40
CD2E 337 ;
CD2E 338 ;
CD2E A9 00 339 SETTOP lda #ZERO
CD30 340 ;
CD30 2C 1A C0 341 bit RDTEXT ; mixed?
CD33 30 02 342 bmi >3
CD35 343 ;
CD35 A9 14 344 lda #20
CD37 345 ;
CD37 85 22 346 ^3 sta WNDTOP
CD39 347 ;
CD39 60 348 ^4 rts
CD3A 349 ;
CD3A 350 ;
CD3A 351 ; Execute mouse text OFF and ON.
CD3A 352 ;
CD3A AD FB 04 353 MOUSEOFF lda XMODE
CD3D 09 01 354 ora #M.MOUSE ; set mouse bit
CD3F D0 05 355 bne >5 ; always taken
CD41 356 ;
CD41 357 ;
CD41 AD FB 04 358 MOUSEON lda XMODE
CD44 29 FE 359 and #NEGONE-M.MOUSE ; clear mouse bit
CD46 360 ;
CD46 8D FB 04 361 ^5 sta XMODE
CD49 362 ;
CD49 60 363 rts
CD4A 364 ;
CD4A 365 ;

```

```

CD4A      366 ; Execute Quit.
CD4A      367 ;
CD4A AD FB 04 368 XNAK      lda XMODE                ; only valid in BASIC
CD4D 30 1A      369                bmi >6                ; ignore if Pascal
CD4F      370 ;
CD4F 20 2B CD 371                jsr DO40
CD52 20 7D CD 372                jsr QUIT
CD55 20 61 CD 373                jsr SETCOUT2
CD58      374 ;
CD58      375 ;
CD58 A9 FD      376 SETKEYIN  lda /KEYIN
CD5A 85 39      377                sta KSWH
CD5C      378 ;
CD5C A9 1B      379                lda #KEYIN
CD5E 85 38      380                sta KSWL
CD60      381 ;
CD60 60      382                rts
CD61      383 ;
CD61      384 ;
CD61 A9 FD      385 SETCOUT2  lda /COUT2
CD63 85 37      386                sta CSWH
CD65      387 ;
CD65 A9 F0      388                lda #COUT2
CD67 85 36      389                sta CSWL
CD69      390 ;
CD69 60      391 ^6          rts
CD6A      392 ;
CD6A      393 ;
CD6A      394 ; Set full 40 column window.
CD6A      395 ;
CD6A A9 28      396 FULL40    lda #40
CD6C D0 02      397                bne >7                ; always taken
CD6E      398 ;
CD6E      399 ;
CD6E      400 ; Set full 80 column window.
CD6E      401 ;
CD6E A9 50      402 FULL80    lda #80
CD70      403 ;
CD70 85 21      404 ^7          sta WNDWDTH
CD72      405 ;
CD72 A9 18      406                lda #24
CD74 85 23      407                sta WNDBTM
CD76      408 ;
CD76 A9 00      409                lda #ZERO
CD78 85 22      410                sta WNDDTOP
CD7A 85 20      411                sta WNDLFT
CD7C      412 ;
CD7C 60      413                rts
CD7D      414 ;
CD7D      415 ;
CD7D      416 ; QUIT as used by PR#0 to turn off everything.
CD7D      417 ;
CD7D 2C 1F C0 418 QUIT      bit RDVID80
CD80 10 03      419                bpl >8
CD82      420 ;
CD82 20 EC CC 421                jsr XDC1.2
CD85      422 ;
CD85 8D 0E C0 423 ^8          sta ALTCHOFF
CD88      424 ;
CD88 A9 FF      425                lda #NEGONE                ; destroy mode
CD8A 8D FB 04 426                sta XMODE

```

```

CD8D          427 ;
CD8D 60        428           rts
CD8E          429 ;
CD8E          430 ;
CD8E          431 ; SCRN84 and SCRN48 convert screens between 40 and 80
CD8E          432 ; columns. WNDTOP must be set up to indicate the last
CD8E          433 ; line to process.
CD8E          434 ;
CD8E 8A        435 SCRN84   txa
CD8F 48        436           pha
CD90          437 ;
CD90 A2 17     438           ldx #23           ; start at bottom
CD92 8D 01 C0  439           sta STR80ON
CD95          440 ;
CD95 8A        441 ^1       txa
CD96          442 ;
CD96 20 BA CA  443           jsr XBASCALC
CD99          444 ;
CD99 A0 27     445           ldy #39           ; start at far right
CD9B          446 ;
CD9B 84 2A     447 ^2       sty BAS2L
CD9D          448 ;
CD9D 98        449           tya
CD9E          450 ;
CD9E 4A        451           lsr
CD9F B0 03     452           bcs >3
CDA1          453 ;
CDA1 2C 55 C0  454           bit PAGE2ON
CDA4          455 ;
CDA4 A8        456 ^3       tay           ; 80 column index
CDA5          457 ;
CDA5 B1 28     458           lda (BASL),Y
CDA7          459 ;
CDA7 2C 54 C0  460           bit PAGE1ON
CDAA          461 ;
CDAA A4 2A     462           ldy BAS2L           ; 40 column index
CDAC          463 ;
CDAC 91 28     464           sta (BASL),Y
CDAE          465 ;
CDAE 88        466           dey
CDAF 10 EA     467           bpl <2
CDB1          468 ;
CDB1 CA        469           dex
CDB2 30 04     470           bmi >4
CDB4          471 ;
CDB4 E4 22     472           cpx WNDTOP
CDB6 B0 DD     473           bcs <1
CDB8          474 ;
CDB8 8D 00 C0  475 ^4       sta STR80OFF       ; for 40 columns
CDBB 8D 0C C0  476           sta VID80OFF      ; for 40 columns
CDBE          477 ;
CDBE 4C F5 CD  478           jmp SCRNRET
CDC1          479 ;
CDC1          480 ;
CDC1 8A        481 SCRN48   txa
CDC2 48        482           pha
CDC3          483 ;
CDC3 A2 17     484           ldx #23           ; start at bottom
CDC5          485 ;
CDC5 8A        486 ^1       txa
CDC6          487 ;

```

```

CDC6 20 BA CA 488      jsr XBASCALC
CDC9          489      ;
CDC9 A0 00      490      ldy #ZERO          ; start at far left
CDCB          491      ;
CDCB 8D 01 C0 492      sta STR800N
CDCE          493      ;
CDCE B1 28      494      ^2      lda (BASL),Y
CDD0          495      ;
CDD0 84 2A      496      sty BAS2L          ; save 40 column index
CDD2          497      ;
CDD2 48          498      pha
CDD3          499      ;
CDD3 98          500      tya
CDD4          501      ;
CDD4 4A          502      lsr
CDD5 B0 03      503      bcs >3
CDD7          504      ;
CDD7 8D 55 C0 505      sta PAGE2ON
CDDA          506      ;
CDDA A8          507      ^3      tay          ; get 80 column index
Cddb          508      ;
Cddb 68          509      pla
CDDC 91 28      510      sta (BASL),Y
CDDE          511      ;
CDDE 8D 54 C0 512      sta PAGE1ON
CDE1          513      ;
CDE1 A4 2A      514      ldy BAS2L
CDE3          515      ;
CDE3 C8          516      iny
CDE4          517      ;
CDE4 C0 28      518      cpy #40
CDE6 90 E6      519      bcc <2
CDE8          520      ;
CDE8 20 AD CC 521      jsr CLRHALF
CDEB          522      ;
CDEB CA          523      dex
CDEC 30 04      524      bmi >4
CDEE          525      ;
CDEE E4 22      526      cpx WNDTOP
CDF0 B0 D3      527      bcs <1
CDF2          528      ;
CDF2 8D 0D C0 529      ^4      sta VID800N
CDF5          530      ;
CDF5 20 FB CD 531      SCRNET  jsr XVTAB2
CDF8          532      ;
CDF8 68          533      pla
CDF9 AA          534      tax
CDFA          535      ;
CDFA 60          536      rts
CDFB          537      ;
CDFB          538      ;
CDFB A5 25      539      XVTAB2  lda CV
CDFD 8D FB 05 540      sta OURCV
CE00          541      ;
CE00 20 BA CA 542      XVTAB3  jsr XBASCALC
CE03          543      ;
CE03 A5 20      544      lda WNDLFT
CE05          545      ;
CE05 2C 1F C0 546      bit RDVID80
CE08 10 01      547      bpl >1
CE0A          548      ;

```

```

CE0A 4A          549          lsr
CE0B             550          ;
CE0B 18          551      ^1      clc
CE0C             552          ;
CE0C 65 28        553          adc BASL
CE0E 85 28        554          sta BASL
CE10             555          ;
CE10 60           556          rts
CE11             557          ;
CE11             558          ;
CE11 A4 24        559      XRDKEY2  ldy CH
CE13             560          ;
CE13 B1 28        561          lda (BASL),Y
CE15             562          ;
CE15 2C 1F C0     563          bit RDVID80
CE18 10 0C        564          bpl INVERT
CE1A             565          ;
CE1A 60           566          rts
CE1B             567          ;
CE1B             568          ;
CE1B             569          dfs $CE1F-*,ZERO      ; 4 bytes
CE1F             570          ;
CE1F             571          ;
CE1F             572          ; Must be at 0xCE1F.
CE1F             573          ;
CE1F AD FB 04     574      PASINV  lda XMODE
CE22 29 10        575          and #M.CURSOR
CE24 D0 11        576          bne >1
CE26             577          ;
CE26 48           578      INVERT  pha
CE27             579          ;
CE27 98           580          tya
CE28 48           581          pha
CE29             582          ;
CE29 AC 7B 05     583          ldy OURCH
CE2C             584          ;
CE2C 20 44 CE     585          jsr PICK
CE2F             586          ;
CE2F 49 80        587          eor #MSBSET          ; flip INVERSE/NORMAL
CE31             588          ;
CE31 20 70 CE     589          jsr STORIT
CE34             590          ;
CE34 68           591          pla
CE35 A8           592          tay
CE36             593          ;
CE36 68           594          pla
CE37             595          ;
CE37 60           596      ^1      rts
CE38             597          ;
CE38             598          ;
CE38             599          ; Store a character onto the screen.
CE38             600          ;
CE38 48           601      STORCHAR pha
CE39             602          ;
CE39 24 32        603          bit INVFLG
CE3B 30 02        604          bmi >1
CE3D             605          ;
CE3D 29 7F        606          and #MSBCLR          ; clear MSB
CE3F             607          ;
CE3F 20 70 CE     608      ^1      jsr STORIT
CE42             609          ;

```



```

CE42 68          610          pla
CE43          611          ;
CE43 60          612          rts
CE44          613          ;
CE44          614          ;
CE44          615          ; Get a character from the screen.
CE44          616          ;
CE44 B1 28       617 PICK      lda (BASL),Y
CE46          618          ;
CE46 2C 1F C0    619          bit RDVID80
CE49 10 19       620          bpl >2
CE4B          621          ;
CE4B 8D 01 C0    622          sta STR80ON
CE4E          623          ;
CE4E 84 2A       624          sty BAS2L
CE50 98          625          tya
CE51 45 20       626          eor WNDLFT          ; get starting page
CE53          627          ;
CE53 6A          628          ror
CE54 B0 04       629          bcs >1
CE56          630          ;
CE56 AD 55 C0    631          lda PAGE2ON
CE59          632          ;
CE59 C8          633          iny
CE5A          634          ;
CE5A 98          635 ^1      tya
CE5B 4A          636          lsr
CE5C A8          637          tay
CE5D          638          ;
CE5D B1 28       639          lda (BASL),Y
CE5F          640          ;
CE5F 2C 54 C0    641          bit PAGE1ON
CE62          642          ;
CE62 A4 2A       643          ldy BAS2L
CE64          644          ;
CE64          645          ;
CE64          646          ; Only allow mouse text if alternate character set enabled.
CE64          647          ;
CE64 2C 1E C0    648 ^2      bit RDALTCH
CE67 10 06       649          bpl >3
CE69          650          ;
CE69 C9 20       651          cmp #$20
CE6B B0 02       652          bcs >3
CE6D          653          ;
CE6D 09 40       654          ora #$40
CE6F          655          ;
CE6F 60          656 ^3      rts
CE70          657          ;
CE70          658          ;
CE70          659          ; Store character.
CE70          660          ;
CE70 48          661 STORIT    pha
CE71 29 FF       662          and #NEGONE
CE73 30 16       663          bmi >1
CE75          664          ;
CE75 AD FB 04    665          lda XMODE
CE78 6A          666          ror
CE79          667          ;
CE79 68          668          pla
CE7A 48          669          pha
CE7B          670          ;

```

```

CE7B 90 0E      671      bcc >1
CE7D            672      ;
CE7D            673      ;
CE7D            674      ; Only process mouse text if alternate character set is
CE7D            675      ; enabled.
CE7D            676      ;
CE7D 2C 1E C0   677      bit RDALTCH
CE80 10 09      678      bpl >1
CE82            679      ;
CE82 49 40      680      eor #$40
CE84            681      ;
CE84 2C 2E CA   682      bit BITBYT60
CE87 F0 02      683      beq >1
CE89            684      ;
CE89 49 40      685      eor #$40
CE8B            686      ;
CE8B 2C 1F C0   687      ^1 bit RDVID80
CE8E 10 1D      688      bpl >3
CE90            689      ;
CE90 8D 01 C0   690      sta STR80ON
CE93 48          691      pha
CE94            692      ;
CE94 84 2A      693      sty BAS2L          ; temp storage
CE96 98          694      tya
CE97 45 20      695      eor WNDLFT        ; get starting page
CE99            696      ;
CE99 4A          697      lsr
CE9A B0 04      698      bcs >2
CE9C            699      ;
CE9C AD 55 C0   700      lda PAGE2ON
CE9F            701      ;
CE9F C8          702      iny
CEA0            703      ;
CEA0 98          704      ^2 tya
CEA1 4A          705      lsr
CEA2 A8          706      tay
CEA3            707      ;
CEA3 68          708      pla
CEA4 91 28      709      sta (BASL),Y
CEA6            710      ;
CEA6 AD 54 C0   711      lda PAGE1ON
CEA9            712      ;
CEA9 A4 2A      713      ldY BAS2L
CEAB            714      ;
CEAB 68          715      pla
CEAC            716      ;
CEAC 60          717      rts
CEAD            718      ;
CEAD            719      ;
CEAD            720      ; Quick 40 column store.
CEAD            721      ;
CEAD 91 28      722      ^3 sta (BASL),Y
CEAF            723      ;
CEAF 68          724      pla
CEB0            725      ;
CEB0 60          726      rts
CEB1            727      ;
CEB1            728      ;
CEB1            729      ; Turn ON Escape cursor.
CEB1            730      ;
CEB1 48          731      ESCON pha

```

```

CEB2          732 ;
CEB2 98       733      tya
CEB3 48       734      pha
CEB4          735 ;
CEB4 AC 7B 05 736      ldy OURCH
CEB7 20 44 CE 737      jsr PICK
CEBA          738 ;
CEBA 8D 7B 06 739      sta XCHAR
CEBD          740 ;
CEBD 29 80    741      and #MSBSET      ; save NORMAL/INVERSE bit
CEBF 49 AB    742      eor #"+"      ; make it inverse
CEC1          743 ;
CEC1 4C CD CE 744      jmp ESCRTN
CEC4          745 ;
CEC4          746 ;
CEC4          747 ; Turn off Escape cursor.
CEC4          748 ;
CEC4 48       749 ESCOFF pha
CEC5          750 ;
CEC5 98       751      tya
CEC6 48       752      pha
CEC7          753 ;
CEC7 AC 7B 05 754      ldy OURCH
CECA AD 7B 06 755      lda XCHAR
CECD          756 ;
CECD          757 ;
CECD          758 ; Return for ESCON and ESCOFF.
CECD          759 ;
CECD 20 70 CE 760 ESCRTN jsr STORIT
CED0          761 ;
CED0 68       762      pla
CED1 A8       763      tay
CED2          764 ;
CED2 68       765      pla
CED3          766 ;
CED3 60       767      rts
CED4          768 ;
CED4          769 ;
CED4          770 ; Set up page zero for Pascal.
CED4          771 ;
CED4 20 6E CD 772 PSETUP jsr FULL80
CED7          773 ;
CED7 A9 FF    774      lda #NEGONE
CED9 85 32    775      sta INVFLG
CEDB          776 ;
CEDB AD FB 04 777      lda XMODE
CEDE 29 04    778      and #M.VMODE
CEE0 F0 02    779      beq >1
CEE2          780 ;
CEE2 46 32    781      lsr INVFLG      ; make it INVERSE
CEE4          782 ;
CEE4 AD 7B 07 783 ^1      lda OLDBASL
CEE7 85 28    784      sta BASL
CEE9          785 ;
CEE9 AD FB 07 786      lda OLDBASH
CEEC 85 29    787      sta BASH
EEEE          788 ;
EEEE AD FB 05 789      lda OURCV
CEF1 85 25    790      sta CV
CEF3          791 ;
CEF3 60       792      rts

```

```

CEF4      793 ;
CEF4      794 ;
CEF4      795 ; COPYROM is called when the video firmware is initialized.
CEF4      796 ; Only if the 0xF8 ROM's signature byte does not match, the
CEF4      797 ; language card is enabled for reading. COPYROM copies the
CEF4      798 ; 0xF8 ROM to the language card and restores the state of
CEF4      799 ; the language card.
CEF4      800 ;
CEF4 2C 12 C0 801 COPYROM bit RDLGRAM
CEF7 10 3F    802 bpl >4
CEF9      803 ;
CEF9 A9 06    804 lda #GOODF8
CEFB CD B3 FB 805 cmp SIGROM
CEFE F0 38    806 beq >4
CF00      807 ;
CF00 A2 03    808 ldx #RAM2WE&$F ; bank 2 RAM W/E
CF02      809 ;
CF02 2C 11 C0 810 bit RDBANK2
CF05 30 02    811 bmi >1
CF07      812 ;
CF07 A2 0B    813 ldx #RAM1WE&$F ; bank 1 RAM W/E
CF09      814 ;
CF09 8D B3 FB 815 ^1 sta SIGROM
CF0C      816 ;
CF0C 2C 80 C0 817 bit RAM2WP ; now read RAM
CF0F      818 ;
CF0F AD B3 FB 819 lda SIGROM
CF12 C9 06    820 cmp #GOODF8
CF14 F0 03    821 beq >2
CF16      822 ;
CF16      823 ;
CF16      824 ; This makes X-reg = 0x04 or 0x0C. Better to decrement
CF16      825 ; X-reg 3 times.
CF16      826 ;
CF16      827 ; inx ; make it write protect
CF16      828 ;
CF16 CA      829 dex
CF17 CA      830 dex
CF18 CA      831 dex
CF19      832 ;
CF19 2C 81 C0 833 ^2 bit ROM2WE ; enable ROM
CF1C 2C 81 C0 834 bit ROM2WE ; write enable RAM
CF1F      835 ;
CF1F A0 00    836 ldy #F8SPACE
CF21      837 ;
CF21 A9 F8    838 lda /F8SPACE
CF23 85 37    839 sta CSWH
CF25      840 ;
CF25 84 36    841 sty CSWL
CF27      842 ;
CF27 B1 36    843 ^3 lda (CSWL),Y
CF29 91 36    844 sta (CSWL),Y
CF2B      845 ;
CF2B C8      846 iny
CF2C D0 F9    847 bne <3
CF2E      848 ;
CF2E E6 37    849 inc CSWH
CF30 D0 F5    850 bne <3
CF32      851 ;
CF32 BD 80 C0 852 lda RAM2WP,X ; write protect RAM
CF35 BD 80 C0 853 lda RAM2WP,X ; unnecessary

```

```

CF38      854 ;
CF38 60    855 ^4      rts
CF39      856 ;
CF39      857 ;
CF39      858 ;      dfs 3,ZERO      ; 3 bytes
CF39      859 ;      dfs 1,ZERO      ; 1 byte
CF3A      860 ;
CF3A      861 ;
CF3A      862 ; Mini-Assembler support routines.  Calculate offset byte
CF3A      863 ; for relative addresses.
CF3A      864 ;
CF3A E9 81  865 REL      sbc #$81
CF3C      866 ;
CF3C 4A     867          lsr
CF3D D0 14  868          bne >3
CF3F      869 ;
CF3F A4 3F  870          ldy A2H
CF41      871 ;
CF41 A6 3E  872          ldx A2L
CF43 D0 01  873          bne >1
CF45      874 ;
CF45 88     875          dey
CF46      876 ;
CF46 CA     877 ^1      dex
CF47 8A     878          txa
CF48      879 ;
CF48 18     880          clc
CF49      881 ;
CF49 E5 3A  882          sbc PCL
CF4B 85 3E  883          sta A2L
CF4D 10 01  884          bpl >2
CF4F      885 ;
CF4F C8     886          iny
CF50      887 ;
CF50 98     888 ^2      tya
CF51      889 ;
CF51 E5 3B  890          sbc PCH
CF53      891 ;
CF53 D0 40  892 ^3      bne >7
CF55      893 ;
CF55      894 ;
CF55      895 ; Move instruction to memory.
CF55      896 ;
CF55 A4 2F  897 ^4      ldy LENGTH
CF57      898 ;
CF57 B9 3D 00 899 ^5      lda A1H,Y
CF5A 91 3A  900          sta (PCL),Y
CF5C      901 ;
CF5C 88     902          dey
CF5D 10 F8  903          bpl <5
CF5F      904 ;
CF5F      905 ;
CF5F      906 ; Display instruction.
CF5F      907 ;
CF5F 20 48 F9 908          jsr PRBLNK
CF62      909 ;
CF62 20 1A FC 910          jsr UP
CF65 20 1A FC 911          jsr UP
CF68      912 ;
CF68 4C E3 FC 913          jmp FINDOP
CF6B      914 ;

```

```

CF6B      915 ;
CF6B      916 ; Compare disassembly of all known opcodes with the one
CF6B      917 ; typed in until a match is found.
CF6B      918 ;
CF6B A5 3D  919 TRYNEXT lda A1H
CF6D      920 ;
CF6D 20 8E F8 921 jsr INSDS2
CF70      922 ;
CF70 AA     923 tax
CF71      924 ;
CF71 BD EB F9 925 lda MNEMR,X
CF74 C5 42   926 cmp A4L
CF76 D0 13   927 bne >6
CF78      928 ;
CF78 BD A6 F9 929 lda MNEML,X
CF7B C5 43   930 cmp A4H
CF7D D0 0C   931 bne >6
CF7F      932 ;
CF7F A5 44   933 lda OPRND
CF81      934 ;
CF81 A4 2E   935 ldy FORMAT
CF83 C0 9D   936 cpy #$9D
CF85 F0 B3   937 beq REL
CF87      938 ;
CF87 C5 2E   939 cmp FORMAT
CF89 F0 CA   940 beq <4
CF8B      941 ;
CF8B C6 3D   942 ^6 dec A1H
CF8D D0 DC   943 bne TRYNEXT
CF8F      944 ;
CF8F E6 44   945 inc OPRND
CF91      946 ;
CF91 C6 35   947 dec YSAV1
CF93 F0 D6   948 beq TRYNEXT
CF95      949 ;
CF95      950 ;
CF95      951 ; Point to the error with a caret, beep, and fall into
CF95      952 ; the Mini-Assembler.
CF95      953 ;
CF95 A4 34   954 ^7 ldy YSAV
CF97      955 ;
CF97 98     956 ^8 tya
CF98 AA     957 tax
CF99      958 ;
CF99 4C D2 FC 959 jmp XERR
CF9C      960 ;
CF9C      961 ;
CF9C      962 ; Read a line of input. If prefaced with " ", decode
CF9C      963 ; mnemonic. If "$" do monitor command. Otherwise, parse
CF9C      964 ; HEX address before decoding mnemonic.
CF9C      965 ;
CF9C 20 C7 FF 966 NXTLINE2 jsr ZMODE
CF9F      967 ;
CF9F AD 00 02 968 lda INPUT
CFA2 C9 A0   969 cmp #SPACE
CFA4 F0 12   970 beq >2
CFA6      971 ;
CFA6 C9 8D   972 cmp #RETURN
CFA8 D0 01   973 bne >9
CFAA      974 ;
CFAA 60     975 rts

```

```

CFAB          976 ;
CFAB 20 A7 FF 977 ^9      jsr GETNUM
CFAE          978 ;
CFAE C9 93    979      cmp #$89+$B0^": "
CFB0          980 ;
CFB0 D0 E5    981 ^1      bne <8
CFB2          982 ;
CFB2 8A       983      txa
CFB3 F0 E2    984      beq <8
CFB5          985 ;
CFB5 20 78 FE 986      jsr A1PCLP
CFB8          987 ;
CFB8 A9 03    988 ^2      lda #3                ; starting opcode
CFBA 85 3D    989      sta A1H
CFBC          990 ;
CFBC 20 00 C5 991 ^3      jsr GETNSP
CFBF          992 ;
CFBF 0A       993      asl
CFC0          994 ;
CFC0 E9 BE    995      sbc #$BE
CFC2          996 ;
CFC2 C9 C2    997      cmp #$C2
CFC4 90 D1    998      bcc <8
CFC6          999 ;
CFC6          1000 ;
CFC6          1001 ; Form mnemonic for later comparison.
CFC6          1002 ;
CFC6 0A       1003      asl
CFC7 0A       1004      asl
CFC8          1005 ;
CFC8 A2 04    1006      ldx #4
CFCA          1007 ;
CFCA 0A       1008 ^4      asl
CFCB          1009 ;
CFCB 26 42    1010      rol A4L
CFCD 26 43    1011      rol A4H
CFCF          1012 ;
CFCF CA       1013      dex
CFD0 10 F8    1014      bpl <4
CFD2          1015 ;
CFD2 C6 3D    1016      dec A1H
CFD4 F0 F4    1017      beq <4
CFD6          1018 ;
CFD6 10 E4    1019      bpl <3
CFD8          1020 ;
CFD8 A2 05    1021      ldx #5
CFDA 20 C8 C4 1022      jsr FORM
CFDD          1023 ;
CFDD A5 44    1024      lda OPRND
CFDF          1025 ;
CFDF 0A       1026      asl
CFE0 0A       1027      asl
CFE1          1028 ;
CFE1 05 35    1029      ora YSAV1
CFE3          1030 ;
CFE3 C9 20    1031      cmp #$20
CFE5 B0 06    1032      bcs >5
CFE7          1033 ;
CFE7 A6 35    1034      ldx YSAV1
CFE9 F0 02    1035      beq >5
CFEB          1036 ;

```

```
CFEB 09 80      1037      ora #MSBSET
CFED           1038      ;
CFED 85 44      1039      ^5      sta OPRND
CFEF 84 34      1040      sty YSAV
CFF1           1041      ;
CFF1 B9 00 02   1042      lda INPUT,Y
CFF4 C9 BB      1043      cmp #";"
CFF6 F0 04      1044      beq >6
CFF8           1045      ;
CFF8 C9 8D      1046      cmp #RETURN
CFFA D0 B4      1047      bne <1
CFFC           1048      ;
CFFC 4C 6B CF   1049      ^6      jmp TRYNEXT
CFFF           1050      ;
CFFF           1051      ;
CFFF           1052      CLRROM      dfs 1,ZERO      ; 1 byte
D000           1053      ;
D000           1054      ;
```

BSAVE C0ROM,D1,A\$1000,B,L\$1000

```
D000           1055      usr C0ROM,D1
D000           1056      ;
D000           1057      ;
D000           1058      icl "D0.L,D2"
```

LLOAD D0.L,D2,A\$4000


```

D000      1          ttl "ROM Source Code, D0.L"
D000      2      ;
D000      3      ;
D000      4      ; D0.L
D000      5      ;
D000      6      ;
D000      7          obj PAGE10
D000      8          usr
D000      9      ;
D000     10      ;
D000     11      ; Basic statement address.  BASIC# = #ADDR/2 + $80.
D000     12      ;
D000     13      BASADDR:
D000 6F D8     14          adr BEND-1          ; END
D002 65 D7     15      BSFOR      adr BFOR-1          ; FOR
D004 F8 DC     16          adr BNEXT-1          ; NEXT
D006 94 D9     17      BSDATA    adr BDATA-1          ; DATA
D008 B1 DB     18          adr BINPUT-1          ; INPUT
D00A 30 F3     19          adr BDEL-1           ; DEL
D00C D8 DF     20          adr BDIM-1           ; DIM
D00E E1 DB     21          adr BREAD-1          ; READ
D010 8F F3     22          adr BGR-1            ; GR
D012 98 F3     23          adr BTEXT-1          ; TEXT
D014 E4 F1     24          adr BPR-1            ; PR#
D016 DD F1     25          adr BIN-1            ; IN#
D018 D4 F1     26          adr BCALL-1           ; CALL
D01A 24 F2     27          adr BPLOT-1           ; PLOT
D01C 31 F2     28          adr BHLIN-1          ; HLIN
D01E 40 F2     29          adr BVLIN-1          ; VLIN
D020 D7 F3     30          adr BHGR2-1          ; HGR2
D022 E1 F3     31          adr BHGR-1           ; HGR
D024 E8 F6     32          adr BHCOLOR-1        ; HCOLOR=
D026 FD F6     33          adr BHPlot-1         ; HPLOT
D028 C3 F5     34          adr BDRAW-1           ; DRAW
D02A C5 F5     35          adr BXDRAW-1          ; XDRAW
D02C E6 F7     36          adr BHTAB-1          ; HTAB
D02E 57 FC     37          adr HOME-1           ; HOME
D030 20 F7     38          adr BROT-1           ; ROT=
D032 26 F7     39          adr BSCALE-1         ; SCALE=
D034         40      ;
D034         41      ;          adr HF775-1          ; SHLOAD
D034 57 FF     42          adr IORTS-1
D036         43      ;
D036 6C F2     44          adr BTRACE-1          ; TRACE
D038 6E F2     45          adr BNOTRACE-1        ; NOTRACE
D03A 72 F2     46          adr BNORMAL-1         ; NORMAL
D03C 75 F2     47          adr BINVERSE-1        ; INVERSE
D03E 7F F2     48          adr BFLASH-1         ; FLASH
D040 4E F2     49          adr BCOLOR-1         ; COLOR=
D042 6A D9     50      BSPOP      adr BPOP-1          ; POP
D044 55 F2     51          adr BVTAB-1          ; VTAB
D046 85 F2     52          adr BHIMEM-1         ; HIMEM:
D048 A5 F2     53          adr BLOMEM-1         ; LOMEM:
D04A CA F2     54          adr BONERR-1         ; ONERR
D04C 17 F3     55          adr BRESUME-1        ; RESUME
D04E         56      ;
D04E         57      ;          adr HF3BC-1          ; RECALL
D04E 57 FF     58          adr IORTS-1
D050         59      ;          adr HF39F-1          ; STORE
D050 57 FF     60          adr IORTS-1

```

```

D052      61 ;
D052 61 F2      62      adr BSPEED-1      ; SPEED=
D054 45 DA      63      adr BLET-1      ; LET
D056 3D D9      64 BSGOTO      adr BGOTO-1      ; GOTO
D058 11 D9      65      adr BRUN-1      ; RUN
D05A C8 D9      66      adr BIF-1      ; IF
D05C 48 D8      67      adr BRESTORE-1      ; RESTORE
D05E F4 03      68      adr USRAHAND-1      ; &
D060 20 D9      69 BSGOSUB      adr BGOSUB-1      ; GOSUB
D062 6A D9      70      adr BRETURN-1      ; RETURN
D064 DB D9      71 BSREM      adr BREM-1      ; REM
D066 6D D8      72      adr BSTOP-1      ; STOP
D068 EB D9      73      adr BON-1      ; ON
D06A 83 E7      74      adr BWAIT-1      ; WAIT
D06C DB D8      75      adr BLOAD-1      ; LOAD
D06E      76 ;
D06E      77 ;      adr HD8B0-1      ; SAVE
D06E 57 FF      78      adr IORTS-1
D070      79 ;
D070 12 E3      80      adr BDEF-1      ; DEF
D072 7A E7      81      adr BPOKE-1      ; POKE
D074 D4 DA      82 BSPRINT      adr BPRINT-1      ; PRINT
D076 95 D8      83      adr BCONT-1      ; CONT
D078 A4 D6      84      adr BLIST-1      ; LIST
D07A 69 D6      85      adr BCLEAR-1      ; CLEAR
D07C 9F DB      86      adr BGET-1      ; GET
D07E 48 D6      87      adr BNEW-1      ; NEW
D080      88 ;
D080      89 ;
D080      90 ; Function statement address. BASIC# = #ADDR/2 + $92.
D080      91 ;
D080      92 FN1ADDR:
D080 90 EB      93 FS1SGN      adr FSGN      ; SGN
D082 23 EC      94      adr FINT      ; INT
D084 AF EB      95      adr FABS      ; ABS
D086 0A 00      96      adr GOUSR      ; USR
D088 DE E2      97      adr FFRE      ; FRE
D08A 12 D4      98      adr FSCRN      ; SCRN(
D08C CD DF      99      adr FPDL      ; PDL
D08E FF E2     100      adr FPOS      ; POS
D090 81 EE     101      adr FSQR      ; SQR
D092 AC EF     102      adr FRND      ; RND
D094 CB F1     103      adr FLOG      ; LOG
D096 09 EF     104      adr FEXP      ; EXP
D098 EA EF     105      adr FCOS      ; COS
D09A F1 EF     106      adr FSIN      ; SIN
D09C 3A F0     107      adr FTAN      ; TAN
D09E 9E F0     108      adr FATAN      ; ATN
D0A0 64 E7     109      adr FPEEK      ; PEEK
D0A2 D6 E6     110      adr FLEN      ; LEN
D0A4 C5 E3     111      adr FSTR      ; STR$
D0A6 07 E7     112      adr FVAL      ; VAL
D0A8 E5 E6     113      adr FASC      ; ASC
D0AA 46 E6     114      adr FCHR      ; CHR$
D0AC      115 ;
D0AC 5A E6     116 FS2LEFT      adr FLEFT      ; LEFT$
D0AE 86 E6     117      adr FRIGHT      ; RIGHT$
D0B0 91 E6     118      adr FMID      ; MID$
D0B2      119 ;
D0B2 41 E9     120 FS3LN      adr FLN      ; LN
D0B4      121 ;

```

```

D0B4      122 ;
D0B4      123 ; Operator tag address.
D0B4      124 ;
D0B4      125 TAGADDR:
D0B4 79    126      byt OTVPLUS          ; +
D0B5 C0 E7 127      adr OPLUS-1
D0B7 79    128      byt OTVMINUS        ; -
D0B8 A9 E7 129      adr OMINUS-1
D0BA 7B    130      byt OTVMULT         ; *
D0BB 81 E9 131      adr OMULT-1
D0BD 7B    132      byt OTVDIVID        ; /
D0BE 68 EA 133      adr ODIVIDE-1
D0C0 7D    134      byt OTVPOWER        ; ^
D0C1 96 EE 135      adr OPOWER-1
D0C3 50    136      byt OTVAND          ; AND
D0C4 54 DF 137      adr OAND-1
D0C6 46    138      byt OTVOR           ; OR
D0C7 4E DF 139      adr OOR-1
D0C9 7F    140      byt OTVGREAT        ; >
D0CA CC EE 141      adr OGT-1
D0CC 7F    142      byt OTVEQUAL        ; =
D0CD 9A DE 143      adr OEQUAL-1
D0CF 64    144 OTLESS byt OTVLESS       ; <
D0D0 64 DF 145      adr OLT-1
D0D2      146 ;
D0D2      147 ;
D0D2      148 ; BASIC statement name.
D0D2      149 ;
D0D2      150 BASNAME:
D0D2 45 4E C4 151      dci 'END'          ; 0x80
D0D5 46 4F D2 152      dci 'FOR'
D0D8 4E 45 58 153      dci 'NEXT'
D0DB D4
D0DC 44 41 54 154      dci 'DATA'
D0DF C1
D0E0 49 4E 50 155      dci 'INPUT'
D0E3 55 D4
D0E5 44 45 CC 156      dci 'DEL'
D0E8 44 49 CD 157      dci 'DIM'
D0EB 52 45 41 158      dci 'READ'
D0EE C4
D0EF 47 D2    159      dci 'GR'
D0F1 54 45 58 160      dci 'TEXT'
D0F4 D4
D0F5 50 52 A3 161      dci 'PR#'
D0F8 49 4E A3 162      dci 'IN#'
D0FB 43 41 4C 163      dci 'CALL'
D0FE CC
D0FF 50 4C 4F 164      dci 'PLOT'
D102 D4
D103 48 4C 49 165      dci 'HLIN'
D106 CE
D107 56 4C 49 166      dci 'VLIN'
D10A CE
D10B 48 47 52 167      dci 'HGR2'          ; 0x90
D10E B2
D10F 48 47 D2 168      dci 'HGR'
D112 48 43 4F 169      dci 'HCOLOR='
D115 4C 4F 52
D118 BD
D119 48 50 4C 170      dci 'HPLOT'

```

D11C	4F	D4			
D11E	44	52	41	171	dci 'DRAW'
D121	D7				
D122	58	44	52	172	dci 'XDRAW'
D125	41	D7			
D127	48	54	41	173	dci 'HTAB'
D12A	C2				
D12B	48	4F	4D	174	dci 'HOME'
D12E	C5				
D12F	52	4F	54	175	dci 'ROT='
D132	BD				
D133	53	43	41	176	dci 'SCALE='
D136	4C	45	BD		
D139				177	;
D139				178	;
D139	47	D2		179	dci 'SHLOAD'
D13B				180	dci 'GR'
D13B	54	52	41	181	;
D13E	43	C5			dci 'TRACE'
D140	4E	4F	54	182	dci 'NOTRACE'
D143	52	41	43		
D146	C5				
D147	4E	4F	52	183	dci 'NORMAL'
D14A	4D	41	CC		
D14D	49	4E	56	184	dci 'INVERSE'
D150	45	52	53		
D153	C5				
D154	46	4C	41	185	dci 'FLASH'
D157	53	C8			
D159	43	4F	4C	186	dci 'COLOR=' ; 0xA0
D15C	4F	52	BD		
D15F	50	4F	D0	187	dci 'POP'
D162	56	54	41	188	dci 'VTAB'
D165	C2				
D166	48	49	4D	189	dci 'HIMEM:'
D169	45	4D	BA		
D16C	4C	4F	4D	190	dci 'LOMEM:'
D16F	45	4D	BA		
D172	4F	4E	45	191	dci 'ONERR'
D175	52	D2			
D177	52	45	53	192	dci 'RESUME'
D17A	55	4D	C5		
D17D				193	;
D17D				194	;
D17D	47	D2		195	dci 'RECALL'
D17F				196	dci 'GR'
D17F	47	D2		197	;
D181				198	dci 'STORE'
D181	53	50	45	199	dci 'GR'
D184	45	44	BD		dci 'SPEED='
D187	4C	45	D4	200	dci 'LET'
D18A	47	4F	54	201	dci 'GOTO'
D18D	CF				
D18E	52	55	CE	202	dci 'RUN'
D191	49	C6		203	dci 'IF'
D193	52	45	53	204	dci 'RESTORE'
D196	54	4F	52		
D199	C5				
D19A	A6			205	asc "&"
D19B	47	4F	53	206	dci 'GOSUB' ; 0xB0
D19E	55	C2			

D1A0	52	45	54	207	dc	i	'RETURN'	
D1A3	55	52	CE					
D1A6	52	45	CD	208	dc	i	'REM'	
D1A9	53	54	4F	209	dc	i	'STOP'	
D1AC	D0							
D1AD	4F	CE		210	dc	i	'ON'	
D1AF	57	41	49	211	dc	i	'WAIT'	
D1B2	D4							
D1B3	4C	4F	41	212	dc	i	'LOAD'	
D1B6	C4							
D1B7				213				;
D1B7				214				;
D1B7	47	D2		215	dc	i	'SAVE'	
D1B9				216	dc	i	'GR'	
D1B9	44	45	C6	217				;
D1BC	50	4F	4B	218	dc	i	'DEF'	
D1BF	C5				dc	i	'POKE'	
D1C0	50	52	49	219	dc	i	'PRINT'	
D1C3	4E	D4						
D1C5	43	4F	4E	220	dc	i	'CONT'	
D1C8	D4							
D1C9	4C	49	53	221	dc	i	'LIST'	
D1CC	D4							
D1CD	43	4C	45	222	dc	i	'CLEAR'	
D1D0	41	D2						
D1D2	47	45	D4	223	dc	i	'GET'	
D1D5	4E	45	D7	224	dc	i	'NEW'	; 0xBF
D1D8				225				;
D1D8	54	41	42	226	dc	i	'TAB('	; 0xC0
D1DB	A8							
D1DC	54	CF		227	dc	i	'TO'	
D1DE	46	CE		228	dc	i	'FN'	
D1E0	53	50	43	229	dc	i	'SPC('	
D1E3	A8							
D1E4	54	48	45	230	dc	i	'THEN'	
D1E7	CE							
D1E8	41	D4		231	dc	i	'AT'	
D1EA	4E	4F	D4	232	dc	i	'NOT'	
D1ED	53	54	45	233	dc	i	'STEP'	; 0xC7
D1F0	D0							
D1F1				234				;
D1F1	AB			235	asc		"+"	; 0xC8
D1F2	AD			236	asc		" - "	
D1F3	AA			237	asc		" * "	
D1F4	AF			238	asc		" / "	
D1F5	DE			239	asc		" ^ "	
D1F6	41	4E	C4	240	dc	i	'AND'	
D1F9	4F	D2		241	dc	i	'OR'	
D1FB	BE			242	asc		" > "	
D1FC	BD			243	asc		" = "	
D1FD	BC			244	asc		" < "	; 0xD1
D1FE				245				;
D1FE	53	47	CE	246	dc	i	'SGN'	; 0xD2
D201	49	4E	D4	247	dc	i	'INT'	
D204	41	42	D3	248	dc	i	'ABS'	
D207	55	53	D2	249	dc	i	'USR'	
D20A	46	52	C5	250	dc	i	'FRE'	
D20D	53	43	52	251	dc	i	'SCRN('	
D210	4E	A8						
D212	50	44	CC	252	dc	i	'PDL'	
D215	50	4F	D3	253	dc	i	'POS'	

D218	53	51	D2	254		dci	´SQR´	
D21B	52	4E	C4	255		dci	´RND´	
D21E	4C	4F	C7	256		dci	´LOG´	
D221	45	58	D0	257		dci	´EXP´	
D224	43	4F	D3	258		dci	´COS´	
D227	53	49	CE	259		dci	´SIN´	
D22A	54	41	CE	260		dci	´TAN´	
D22D	41	54	CE	261		dci	´ATN´	
D230	50	45	45	262		dci	´PEEK´	
D233	CB							
D234	4C	45	CE	263		dci	´LEN´	
D237	53	54	52	264		dci	´STR\$´	
D23A	A4							
D23B	56	41	CC	265		dci	´VAL´	
D23E	41	53	C3	266		dci	´ASC´	
D241	43	48	52	267		dci	´CHR\$´	; 0xE7
D244	A4							
D245				268	;			
D245	4C	45	46	269		dci	´LEFT\$´	; 0xE8
D248	54	A4						
D24A	52	49	47	270		dci	´RIGHT\$´	
D24D	48	54	A4					
D250	4D	49	44	271		dci	´MID\$´	; 0xEA
D253	A4							
D254				272	;			
D254	4C	CE		273		dci	´LN´	; 0xEB
D256				274	;			
D256	00			275		hex	00	
D257				276	;			
D257				277	;			
D257				278	MESGS:			
D257	4E	45	58	279	MSG01	dci	´NEXT without FOR´	
D25A	54	20	77					
D25D	69	74	68					
D260	6F	75	74					
D263	20	46	4F					
D266	D2							
D267	53	79	6E	280	MSG02	dci	´Syntax´	
D26A	74	61	F8					
D26D	52	45	54	281	MSG03	dci	´RETURN without GOSUB´	
D270	55	52	4E					
D273	20	77	69					
D276	74	68	6F					
D279	75	74	20					
D27C	47	4F	53					
D27F	55	C2						
D281	4F	75	74	282	MSG04	dci	´Out of Data´	
D284	20	6F	66					
D287	20	44	61					
D28A	74	E1						
D28C	49	6C	6C	283	MSG05	dci	´Illegal Quantity´	
D28F	65	67	61					
D292	6C	20	51					
D295	75	61	6E					
D298	74	69	74					
D29B	F9							
D29C	4F	76	65	284	MSG06	dci	´Overflow´	
D29F	72	66	6C					
D2A2	6F	F7						
D2A4	4F	75	74	285	MSG07	dci	´Out of Memory´	
D2A7	20	6F	66					

```

D2AA 20 4D 65
D2AD 6D 6F 72
D2B0 F9
D2B1 55 6E 64    286  MSG08    dci 'Undefined Statement'
D2B4 65 66 69
D2B7 6E 65 64
D2BA 20 53 74
D2BD 61 74 65
D2C0 6D 65 6E
D2C3 F4
D2C4 42 61 64    287  MSG09    dci 'Bad Subscript'
D2C7 20 53 75
D2CA 62 73 63
D2CD 72 69 70
D2D0 F4
D2D1 52 65 64    288  MSG10    dci 'Redefined Array'
D2D4 65 66 69
D2D7 6E 65 64
D2DA 20 41 72
D2DD 72 61 F9
D2E0 44 69 76    289  MSG11    dci 'Divide by Zero'
D2E3 69 64 65
D2E6 20 62 79
D2E9 20 5A 65
D2EC 72 EF
D2EE 49 6C 6C    290  MSG12    dci 'Illegal Direct'
D2F1 65 67 61
D2F4 6C 20 44
D2F7 69 72 65
D2FA 63 F4
D2FC 54 79 70    291  MSG13    dci 'Type Mismatch'
D2FF 65 20 4D
D302 69 73 6D
D305 61 74 63
D308 E8
D309 53 74 72    292  MSG14    dci 'String too Long'
D30C 69 6E 67
D30F 20 74 6F
D312 6F 20 4C
D315 6F 6E E7
D318 46 6F 72    293  MSG15    dci 'Formula too Complex'
D31B 6D 75 6C
D31E 61 20 74
D321 6F 6F 20
D324 43 6F 6D
D327 70 6C 65
D32A F8
D32B 43 61 6E    294  MSG16    dci 'Cannot Continue'
D32E 6E 6F 74
D331 20 43 6F
D334 6E 74 69
D337 6E 75 E5
D33A 55 6E 64    295  MSG17    dci 'Undefined Function'
D33D 65 66 69
D340 6E 65 64
D343 20 46 75
D346 6E 63 74
D349 69 6F EE
D34C                                296  ;
D34C 20 45 72    297  MSG18    asc 'Error'
D34F 72 6F 72

```

```

D352 07 00      298      byt ASCIBELL&MSBCLR,ZERO
D354           299      ;
D354 20 69 6E   300      MSG19  asc ' in '
D357 20
D358 00         301      byt ZERO
D359           302      ;
D359 0D         303      MSG20  byt RETURN&MSBCLR
D35A 42 72 65   304      asc 'Break'
D35D 61 6B
D35F 07 00      305      byt ASCIBELL&MSBCLR,ZERO
D361           306      ;
D361           307      ;
D361           308      dfs 1,ZERO          ; 1 byte
D362           309      ;
D362           310      ;
D362 BA        311      GTFORPNT tsx
D363           312      ;
D363 E8         313      inx
D364 E8         314      inx
D365 E8         315      inx
D366 E8         316      inx
D367           317      ;
D367 BD 01 01   318      ^1      lda STACK+1,X
D36A C9 81      319      cmp #$80+BSFOR/2
D36C D0 24      320      bne >4
D36E           321      ;
D36E A5 86      322      lda FORPNT+1
D370 D0 0D      323      bne >2
D372           324      ;
D372 BD 02 01   325      lda STACK+2,X
D375 85 85      326      sta FORPNT
D377           327      ;
D377 BD 03 01   328      lda STACK+3,X
D37A 85 86      329      sta FORPNT+1
D37C           330      ;
D37C C5 86      331      cmp FORPNT+1          ; accelerate code; force Z=1
D37E           332      ;
D37E 60         333      rts
D37F           334      ;
D37F DD 03 01   335      ^2      cmp STACK+3,X
D382 D0 07      336      bne >3
D384           337      ;
D384 A5 85      338      lda FORPNT
D386 DD 02 01   339      cmp STACK+2,X
D389 F0 07      340      beq >4
D38B           341      ;
D38B 18         342      ^3      clc
D38C           343      ;
D38C 8A         344      txa
D38D 69 12      345      adc #18
D38F           346      ;
D38F AA         347      tax
D390 D0 D5      348      bne <1
D392           349      ;
D392 60         350      ^4      rts
D393           351      ;
D393           352      ;
D393           353      ; Block transfer utility (ACA).
D393           354      ;
D393 20 E3 D3   355      BLTU      jsr CKSTRSIZ
D396           356      ;

```



```

D396 85 6D      357      sta STREND
D398 84 6E      358      sty STREND+1
D39A           359      ;
D39A 38         360      sec
D39B           361      ;
D39B A5 96      362      lda HIGHTR
D39D E5 9B      363      sbc LOWTR
D39F 85 5E      364      sta INDEX
D3A1           365      ;
D3A1 A8         366      tay
D3A2           367      ;
D3A2 A5 97      368      lda HIGHTR+1
D3A4 E5 9C      369      sbc LOWTR+1
D3A6 AA         370      tax
D3A7           371      ;
D3A7 E8         372      inx
D3A8           373      ;
D3A8 98         374      tya
D3A9 F0 23      375      beq >4
D3AB           376      ;
D3AB 38         377      sec
D3AC           378      ;
D3AC A5 96      379      lda HIGHTR
D3AE E5 5E      380      sbc INDEX
D3B0 85 96      381      sta HIGHTR
D3B2 B0 03      382      bcs >1
D3B4           383      ;
D3B4 C6 97      384      dec HIGHTR+1
D3B6           385      ;
D3B6 38         386      sec
D3B7           387      ;
D3B7 A5 94      388      ^1    lda HIGHDS
D3B9 E5 5E      389      sbc INDEX
D3BB 85 94      390      sta HIGHDS
D3BD B0 08      391      bcs >3
D3BF           392      ;
D3BF C6 95      393      dec HIGHDS+1
D3C1 90 04      394      bcc >3      ; always taken
D3C3           395      ;
D3C3 B1 96      396      ^2    lda (HIGHTR),Y
D3C5 91 94      397      sta (HIGHDS),Y
D3C7           398      ;
D3C7 88         399      ^3    dey
D3C8 D0 F9      400      bne <2
D3CA           401      ;
D3CA B1 96      402      lda (HIGHTR),Y
D3CC 91 94      403      sta (HIGHDS),Y
D3CE           404      ;
D3CE C6 97      405      ^4    dec HIGHTR+1
D3D0 C6 95      406      dec HIGHDS+1
D3D2           407      ;
D3D2 CA         408      dex
D3D3 D0 F2      409      bne <3
D3D5           410      ;
D3D5 60         411      rts
D3D6           412      ;
D3D6           413      ;
D3D6           414      ; Check stack size.
D3D6           415      ;
D3D6 0A         416      CKSTKSIZ asl
D3D7           417      ;

```

```

D3D7 69 36      418      adc #$36
D3D9 B0 35      419      bcs PRMSG07
D3DB           420      ;
D3DB 85 5E      421      sta INDEX
D3DD           422      ;
D3DD BA        423      tsx
D3DE           424      ;
D3DE E4 5E      425      cpx INDEX
D3E0 90 2E      426      bcc PRMSG07
D3E2           427      ;
D3E2 60         428      rts
D3E3           429      ;
D3E3           430      ;
D3E3           431      ; Check space between arrays and strings.
D3E3           432      ;
D3E3 C4 70      433      CKSTRSIZ cpy FRETOP+1
D3E5 90 28      434      bcc >4
D3E7           435      ;
D3E7 D0 04      436      bne >1
D3E9           437      ;
D3E9 C5 6F      438      cmp FRETOP
D3EB 90 22      439      bcc >4
D3ED           440      ;
D3ED 48         441      ^1 pha
D3EE           442      ;
D3EE A2 09      443      ldx #9
D3F0           444      ;
D3F0 98         445      tya
D3F1           446      ;
D3F1 48         447      ^2 pha
D3F2           448      ;
D3F2 B5 93      449      lda TEMP1,X
D3F4           450      ;
D3F4 CA        451      dex
D3F5 10 FA      452      bpl <2
D3F7           453      ;
D3F7 20 84 E4   454      jsr GARBAG
D3FA           455      ;
D3FA A2 F7      456      ldx #!-9
D3FC           457      ;
D3FC 68         458      ^3 pla
D3FD 95 9D      459      sta DSCTMP,X
D3FF           460      ;
D3FF E8         461      inx
D400 30 FA      462      bmi <3
D402           463      ;
D402 68         464      pla
D403 A8         465      tay
D404           466      ;
D404 68         467      pla
D405           468      ;
D405 C4 70      469      cpy FRETOP+1
D407 90 06      470      bcc >4
D409           471      ;
D409 D0 05      472      bne PRMSG07
D40B           473      ;
D40B C5 6F      474      cmp FRETOP
D40D B0 01      475      bcs PRMSG07
D40F           476      ;
D40F 60         477      ^4 rts
D410           478      ;

```

```
D410          479  ;  
D410          480      icl "D4.L"  
  
LLOAD D4.L,A$4000
```

```

D410          1          ttl "ROM Source Code, D4.L"
D410          2          ;
D410          3          ;
D410          4          ; D4.L
D410          5          ;
D410          6          ;
D410 A2 4D      7 PRMSG07 ldx #MSG07-MESGS      ; Out of Memory error
D412          8          ;
D412 24 D8      9 FSCRN   bit ERRFLG
D414 10 03     10         bpl >1
D416          11         ;
D416 4C E9 F2   12         jmp HANDLERR
D419          13         ;
D419 20 FB DA   14 ^1      jsr PRTCR
D41C 20 5A DB   15         jsr OUTQSTN
D41F          16         ;
D41F BD 57 D2   17 ^2      lda MSGS,X
D422 48         18         pha
D423          19         ;
D423 20 5C DB   20         jsr OUTCHR
D426          21         ;
D426 E8         22         inx
D427          23         ;
D427 68         24         pla
D428 10 F5      25         bpl <2
D42A          26         ;
D42A 20 83 D6   27         jsr STKINIT
D42D          28         ;
D42D A9 4C      29         lda #MSG18
D42F A0 D3      30         ldy /MSG18
D431          31         ;
D431          32         ;
D431 20 3A DB   33 PR.LINUM jsr STROUT
D434          34         ;
D434 A4 76      35         ldy CURLIN+1
D436          36         ;
D436 C8         37         iny
D437 F0 03      38         beq RESTART
D439          39         ;
D439 20 19 ED   40         jsr PRTMSG19
D43C          41         ;
D43C          42         ;
D43C          43         ; This is the DOS default WARMADR.
D43C          44         ;
D43C 20 FB DA   45 RESTART jsr PRTCR
D43F          46         ;
D43F A2 DD      47         ldx #"]"
D441 20 2E D5   48         jsr INLIN2
D444          49         ;
D444 86 B8      50         stx TXTPTR
D446 84 B9      51         sty TXTPTR+1
D448          52         ;
D448 46 D8      53         lsr ERRFLG
D44A          54         ;
D44A 20 B1 00   55         jsr CHRGET
D44D          56         ;
D44D AA         57         tax
D44E F0 EC      58         beq RESTART
D450          59         ;
D450 A2 FF      60         ldx #NEGONE

```

```

D452 86 76      61      stx CURLIN+1
D454           62      ;
D454 90 06      63      bcc >1                      ; a number from CHRGET
D456           64      ;
D456 20 59 D5   65      jsr PARSINPT
D459           66      ;
D459 4C 05 D8   67      jmp DOTRACE
D45C           68      ;
D45C A6 AF      69      ^1      ldx PRGEND
D45E 86 69      70      stx VARTAB
D460           71      ;
D460 A6 B0      72      ldx PRGEND+1
D462 86 6A      73      stx VARTAB+1
D464           74      ;
D464 20 0C DA   75      jsr LINGET
D467 20 59 D5   76      jsr PARSINPT
D46A           77      ;
D46A 84 0F      78      sty EOLPTR
D46C           79      ;
D46C 20 1A D6   80      jsr FNDLIN
D46F 90 44      81      bcc >4
D471           82      ;
D471 A0 01      83      ldy #1
D473           84      ;
D473 B1 9B      85      lda (LOWTR),Y
D475 85 5F      86      sta INDEX+1
D477           87      ;
D477 A5 69      88      lda VARTAB
D479 85 5E      89      sta INDEX
D47B           90      ;
D47B A5 9C      91      lda LOWTR+1
D47D 85 61      92      sta DEST+1
D47F           93      ;
D47F A5 9B      94      lda LOWTR
D481           95      ;
D481 88         96      dey
D482           97      ;
D482 F1 9B      98      sbc (LOWTR),Y
D484           99      ;
D484 18        100     clc
D485          101     ;
D485 65 69     102     adc VARTAB
D487 85 69     103     sta VARTAB
D489 85 60     104     sta DEST
D48B          105     ;
D48B A5 6A     106     lda VARTAB+1
D48D 69 FF     107     adc #NEGONE
D48F 85 6A     108     sta VARTAB+1
D491          109     ;
D491 E5 9C     110     sbc LOWTR+1
D493 AA       111     tax
D494          112     ;
D494 38       113     sec
D495          114     ;
D495 A5 9B     115     lda LOWTR
D497 E5 69     116     sbc VARTAB
D499 A8       117     tay
D49A B0 03     118     bcs >2
D49C          119     ;
D49C E8       120     inx
D49D          121     ;

```

```

D49D C6 61      122      dec DEST+1
D49F           123      ;
D49F 18         124      ^2      clc
D4A0           125      ;
D4A0 65 5E      126      adc INDEX
D4A2 90 03      127      bcc >3
D4A4           128      ;
D4A4 C6 5F      129      dec INDEX+1
D4A6           130      ;
D4A6 18         131      clc
D4A7           132      ;
D4A7 B1 5E      133      ^3      lda (INDEX),Y
D4A9 91 60      134      sta (DEST),Y
D4AB           135      ;
D4AB C8         136      iny
D4AC D0 F9      137      bne <3
D4AE           138      ;
D4AE E6 5F      139      inc INDEX+1
D4B0 E6 61      140      inc DEST+1
D4B2           141      ;
D4B2 CA        142      dex
D4B3 D0 F2      143      bne <3
D4B5           144      ;
D4B5 AD 00 02   145      ^4      lda INPUT
D4B8 F0 38      146      beq ASENTER
D4BA           147      ;
D4BA A5 73      148      lda MEMSIZE
D4BC A4 74      149      ldy MEMSIZE+1
D4BE           150      ;
D4BE 85 6F      151      sta FRETOP
D4C0 84 70      152      sty FRETOP+1
D4C2           153      ;
D4C2 A5 69      154      lda VARTAB
D4C4 85 96      155      sta HIGHTR
D4C6           156      ;
D4C6 65 0F      157      adc EOLPTR
D4C8 85 94      158      sta HIGHDS
D4CA           159      ;
D4CA A4 6A      160      ldy VARTAB+1
D4CC 84 97      161      sty HIGHTR+1
D4CE           162      ;
D4CE 90 01      163      bcc >5
D4D0           164      ;
D4D0 C8         165      iny
D4D1           166      ;
D4D1 84 95      167      ^5      sty HIGHDS+1
D4D3           168      ;
D4D3 20 93 D3   169      jsr BLTU
D4D6           170      ;
D4D6 A5 50      171      lda ACL
D4D8 A4 51      172      ldy ACH
D4DA           173      ;
D4DA 8D FE 01   174      sta INPUT-2
D4DD 8C FF 01   175      sty INPUT-1
D4E0           176      ;
D4E0 A5 6D      177      lda STREND
D4E2 A4 6E      178      ldy STREND+1
D4E4           179      ;
D4E4 85 69      180      sta VARTAB
D4E6 84 6A      181      sty VARTAB+1
D4E8           182      ;

```

```

D4E8 A4 0F      183      ldy EOLPTR
D4EA           184      ;
D4EA B9 FB 01   185      ^6      lda INPUT-5,Y
D4ED           186      ;
D4ED 88         187      dey
D4EE           188      ;
D4EE 91 9B      189      sta (LOWTR),Y
D4F0           190      ;
D4F0 D0 F8      191      bne <6
D4F2           192      ;
D4F2           193      ;
D4F2           194      ; Applesoft interpreter. Used by DOS for the LOAD command
D4F2           195      ; as ASROMRST and is the DOS default RESETADR.
D4F2           196      ;
D4F2 20 65 D6   197      ASENTER jsr SETPTRS
D4F5           198      ;
D4F5 A5 67      199      lda PRGTAB
D4F7 A4 68      200      ldy PRGTAB+1
D4F9           201      ;
D4F9 85 5E      202      sta INDEX
D4FB 84 5F      203      sty INDEX+1
D4FD           204      ;
D4FD 18         205      clc
D4FE           206      ;
D4FE A0 01      207      ^1      ldy #1
D500           208      ;
D500 B1 5E      209      lda (INDEX),Y
D502 D0 0B      210      bne >2
D504           211      ;
D504 A5 69      212      lda VARTAB
D506 85 AF      213      sta PRGEND
D508           214      ;
D508 A5 6A      215      lda VARTAB+1
D50A 85 B0      216      sta PRGEND+1
D50C           217      ;
D50C 4C 3C D4   218      jmp RESTART
D50F           219      ;
D50F A0 04      220      ^2      ldy #4
D511           221      ;
D511 C8         222      ^3      iny
D512           223      ;
D512 B1 5E      224      lda (INDEX),Y
D514 D0 FB      225      bne <3
D516           226      ;
D516 C8         227      iny
D517           228      ;
D517 98         229      tya
D518 65 5E      230      adc INDEX
D51A AA         231      tax
D51B           232      ;
D51B A0 00      233      ldy #ZERO
D51D           234      ;
D51D 91 5E      235      sta (INDEX),Y
D51F           236      ;
D51F A5 5F      237      lda INDEX+1
D521 69 00      238      adc #ZERO
D523           239      ;
D523 C8         240      iny
D524           241      ;
D524 91 5E      242      sta (INDEX),Y
D526           243      ;

```

```

D526 86 5E      244      stx INDEX
D528 85 5F      245      sta INDEX+1
D52A           246      ;
D52A 90 D2      247      bcc <1
D52C           248      ;
D52C           249      ;
D52C A2 80      250      INLIN    ldx #$80
D52E           251      ;
D52E 86 33      252      INLIN2   stx PROMPT
D530           253      ;
D530 20 6A FD    254      jsr GETLINE
D533           255      ;
D533 E0 EF      256      cpx #MAXINPUT
D535 90 02      257      bcc >1
D537           258      ;
D537 A2 EF      259      ldx #MAXINPUT
D539           260      ;
D539 A9 00      261      ^1      lda #ZERO
D53B 9D 00 02    262      sta INPUT,X
D53E           263      ;
D53E 8A         264      txa
D53F F0 0B      265      beq >3
D541           266      ;
D541 BD FF 01    267      ^2      lda INPUT-1,X
D544 29 7F      268      and #MSBCLR
D546 9D FF 01    269      sta INPUT-1,X
D549           270      ;
D549 CA         271      dex
D54A D0 F5      272      bne <2
D54C           273      ;
D54C A9 00      274      ^3      lda #ZERO
D54E A2 FF      275      ldx #NEGONE
D550 A0 01      276      ldy #1
D552           277      ;
D552 60         278      rts
D553           279      ;
D553           280      ;
D553           281      ; Removed INCHR routine.
D553           282      ;
D553           283      dfs 6,ZERO          ; 6 bytes
D559           284      ;
D559           285      ;
D559           286      ; Parse and tokenize the input line.
D559           287      ;
D559 A6 B8      288      PARSINPT ldx TXTPTR
D55B           289      ;
D55B CA         290      dex
D55C           291      ;
D55C A0 04      292      ldy #4
D55E 84 13      293      sty DATAFLG
D560           294      ;
D560 24 D6      295      bit RUNFLAG
D562 10 08      296      bpl >1
D564           297      ;
D564 68         298      pla
D565 68         299      pla
D566           300      ;
D566 20 65 D6    301      jsr SETPTRS
D569           302      ;
D569 4C D2 D7    303      jmp NEWSTT
D56C           304      ;

```



```

D56C E8          305 ^1      inx
D56D             306 ;
D56D 20 9E F7    307 ^2      jsr PARSIEX2
D570             308 ;
D570 24 13       309      bit DATAFLG
D572 70 04       310      bvs >3
D574             311 ;
D574 C9 20       312      cmp #' '
D576 F0 F4       313      beq <1
D578             314 ;
D578 85 0E       315 ^3      sta ENDCHR
D57A             316 ;
D57A C9 22       317      cmp #'" '
D57C F0 74       318      beq >2
D57E             319 ;
D57E 70 4D       320      bvs >1
D580             321 ;
D580 C9 3F       322      cmp #'? '
D582 D0 04       323      bne >4
D584             324 ;
D584 A9 BA       325      lda #$80+BSPRINT/2
D586 D0 45       326      bne >1
D588             327 ;
D588 C9 30       328 ^4      cmp #'0 '
D58A 90 04       329      bcc >5
D58C             330 ;
D58C C9 3C       331      cmp #'< '
D58E 90 3D       332      bcc >1
D590             333 ;
D590 84 AD       334 ^5      sty STRING2
D592             335 ;
D592 A9 D2       336      lda #BASNAME-PAGESIZE
D594 85 9D       337      sta DSCTMP
D596             338 ;
D596 A9 CF       339      lda /BASNAME-PAGESIZE
D598 85 9E       340      sta DSCTMP+1
D59A             341 ;
D59A A0 00       342      ldy #ZERO
D59C 84 0F       343      sty TOKNCNTR
D59E             344 ;
D59E 88          345      dey
D59F             346 ;
D59F 86 B8       347      stx TXTPTR
D5A1             348 ;
D5A1 CA          349      dex
D5A2             350 ;
D5A2 C8          351 ^6      iny
D5A3 D0 02       352      bne >7
D5A5             353 ;
D5A5 E6 9E       354      inc DSCTMP+1
D5A7             355 ;
D5A7 E8          356 ^7      inx
D5A8             357 ;
D5A8 20 9E F7    358 ^8      jsr PARSIEX2
D5AB             359 ;
D5AB C9 20       360      cmp #' '
D5AD F0 F8       361      beq <7
D5AF             362 ;
D5AF 38          363      sec
D5B0             364 ;
D5B0 F1 9D       365      sbc (DSCTMP),Y

```

```

D5B2 F0 EE      366      beq <6
D5B4            367      ;
D5B4 C9 80      368      cmp #$80
D5B6 D0 41      369      bne >6
D5B8            370      ;
D5B8 05 0F      371      ora TOKNCNTR
D5BA C9 C5      372      cmp #TKAT
D5BC D0 0D      373      bne >9
D5BE            374      ;
D5BE 20 99 F7    375      jsr PARSIEX1
D5C1            376      ;
D5C1 C9 4E      377      cmp #'N'          ; ATN over AT
D5C3 F0 34      378      beq >6
D5C5            379      ;
D5C5 C9 4F      380      cmp #'O'          ; TO over AT
D5C7 F0 30      381      beq >6
D5C9            382      ;
D5C9 A9 C5      383      lda #TKAT
D5CB            384      ;
D5CB A4 AD      385      ^9      ldy STRING2
D5CD            386      ;
D5CD E8          387      ^1      inx
D5CE C8          388      iny
D5CF            389      ;
D5CF 99 FB 01    390      sta INPUT-5,Y
D5D2            391      ;
D5D2 B9 FB 01    392      lda INPUT-5,Y
D5D5 F0 39      393      beq >8
D5D7            394      ;
D5D7 38          395      sec
D5D8            396      ;
D5D8 E9 3A      397      sbc #'9'+1
D5DA F0 04      398      beq >3
D5DC            399      ;
D5DC C9 49      400      cmp #$46+BSDATA/2      ; 0x80-': '
D5DE D0 02      401      bne >4
D5E0            402      ;
D5E0 85 13      403      ^3      sta DATAFLG
D5E2            404      ;
D5E2 38          405      ^4      sec
D5E3            406      ;
D5E3 E9 78      407      sbc #$46+BSREM/2      ; 0x80-': '
D5E5 D0 86      408      bne <2
D5E7            409      ;
D5E7 85 0E      410      sta ENDCHR
D5E9            411      ;
D5E9 20 9E F7    412      ^5      jsr PARSIEX2
D5EC F0 DF      413      beq <1
D5EE            414      ;
D5EE C5 0E      415      cmp ENDCHR
D5F0 F0 DB      416      beq <1
D5F2            417      ;
D5F2 C8          418      ^2      iny
D5F3            419      ;
D5F3 99 FB 01    420      sta INPUT-5,Y
D5F6            421      ;
D5F6 E8          422      inx
D5F7 D0 F0      423      bne <5
D5F9            424      ;
D5F9 A6 B8      425      ^6      ldx TXTPTR
D5FB            426      ;

```

```

D5FB E6 0F      427      inc TOKNCNTR
D5FD            428      ;
D5FD B1 9D      429      ^7      lda (DSCTMP),Y
D5FF            430      ;
D5FF C8         431      iny
D600 D0 02      432      bne >1
D602            433      ;
D602 E6 9E      434      inc DSCTMP+1
D604            435      ;
D604 0A         436      ^1      asl
D605 90 F6      437      bcc <7
D607            438      ;
D607 B1 9D      439      lda (DSCTMP),Y
D609 D0 9D      440      bne <8
D60B            441      ;
D60B 20 AC F7   442      jsr PARSIEX3
D60E 10 BB      443      bpl <9
D610            444      ;
D610 99 FD 01   445      ^8      sta INPUT-3,Y
D613            446      ;
D613 C6 B9      447      dec TXTPTR+1
D615            448      ;
D615 A9 FF      449      lda #NEGONE
D617 85 B8      450      sta TXTPTR
D619            451      ;
D619 60         452      rts
D61A            453      ;
D61A            454      ;
D61A A5 67      455      FNDLIN      lda PRGTAB
D61C A6 68      456      ldx PRGTAB+1
D61E            457      ;
D61E A0 01      458      FNDLIN2     ldy #1
D620            459      ;
D620 85 9B      460      sta LOWTR
D622 86 9C      461      stx LOWTR+1
D624            462      ;
D624 B1 9B      463      lda (LOWTR),Y
D626 F0 1F      464      beq >3
D628            465      ;
D628 C8         466      iny
D629 C8         467      iny
D62A            468      ;
D62A A5 51      469      lda ACH          ; LINNUM+1
D62C D1 9B      470      cmp (LOWTR),Y
D62E 90 18      471      bcc >4
D630            472      ;
D630 F0 03      473      beq >1
D632            474      ;
D632 88         475      dey
D633 D0 09      476      bne >2
D635            477      ;
D635 A5 50      478      ^1      lda ACL          ; LINNUM
D637            479      ;
D637 88         480      dey
D638            481      ;
D638 D1 9B      482      cmp (LOWTR),Y
D63A 90 0C      483      bcc >4
D63C            484      ;
D63C F0 0A      485      beq >4
D63E            486      ;
D63E 88         487      ^2      dey

```

```

D63F          488 ;
D63F B1 9B    489     lda (LOWTR),Y
D641 AA       490     tax
D642          491 ;
D642 88       492     dey
D643          493 ;
D643 B1 9B    494     lda (LOWTR),Y
D645          495 ;
D645 B0 D7    496     bcs FNDLIN2
D647          497 ;
D647 18       498     ^3   clc
D648          499 ;
D648 60       500     ^4   rts
D649          501 ;
D649          502 ;
D649 D0 FD    503     BNEW   bne <4
D64B          504 ;
D64B          505 ;
D64B A9 00    506     SCRTCH  lda #ZERO
D64D 85 D6    507     sta RUNFLAG
D64F A8       508     tay
D650          509 ;
D650 91 67    510     sta (PRGTAB),Y
D652          511 ;
D652 C8       512     iny
D653          513 ;
D653 91 67    514     sta (PRGTAB),Y
D655          515 ;
D655 A5 67    516     lda PRGTAB
D657 69 02    517     adc #2
D659 85 69    518     sta VARTAB
D65B 85 AF    519     sta PRGEND
D65D          520 ;
D65D A5 68    521     lda PRGTAB+1
D65F 69 00    522     adc #ZERO
D661 85 6A    523     sta VARTAB+1
D663 85 B0    524     sta PRGEND+1
D665          525 ;
D665          526 ;
D665          527 ; Used by DOS for the RUN and CHAIN commands as ASROMCLR.
D665          528 ;
D665 20 97 D6 529     SETPTRS  jsr STXTPTR
D668          530 ;
D668 A9 00    531     lda #ZERO
D66A          532 ;
D66A          533 ;
D66A D0 2A    534     BCLEAR   bne >0
D66C          535 ;
D66C          536 ;
D66C A5 73    537     CLEARC   lda MEMSIZE
D66E A4 74    538     ldy MEMSIZE+1
D670          539 ;
D670 85 6F    540     sta FRETOP
D672 84 70    541     sty FRETOP+1
D674          542 ;
D674 A5 69    543     lda VARTAB
D676 A4 6A    544     ldy VARTAB+1
D678          545 ;
D678 85 6B    546     sta ARYTAB
D67A 84 6C    547     sty ARYTAB+1
D67C          548 ;

```

```

D67C 85 6D      549      sta STREND
D67E 84 6E      550      sty STREND+1
D680           551      ;
D680 20 49 D8   552      jsr BRESTORE
D683           553      ;
D683           554      ;
D683 A2 55      555 STKINIT ldx #TEMPST
D685 86 52      556      stx TEMPPT
D687           557      ;
D687 68         558      pla
D688 A8         559      tay
D689           560      ;
D689 68         561      pla
D68A           562      ;
D68A A2 F8      563      ldx #$F8
D68C 9A         564      txs
D68D           565      ;
D68D 48         566      pha
D68E           567      ;
D68E 98         568      tya
D68F 48         569      pha
D690           570      ;
D690 A9 00      571      lda #ZERO
D692 85 7A      572      sta TEXTPTR+1
D694 85 14      573      sta SUBFLG
D696           574      ;
D696 60         575      ^0    rts
D697           576      ;
D697           577      ;
D697 18         578 STXTPTR clc
D698           579      ;
D698 A5 67      580      lda PRGTAB
D69A 69 FF      581      adc #NEGONE
D69C 85 B8      582      sta TXTPTR
D69E           583      ;
D69E A5 68      584      lda PRGTAB+1
D6A0 69 FF      585      adc #NEGONE
D6A2 85 B9      586      sta TXTPTR+1
D6A4           587      ;
D6A4 60         588      ^0    rts
D6A5           589      ;
D6A5           590      ;
D6A5 90 0A      591 BLIST   bcc >1
D6A7 F0 08      592      beq >1
D6A9           593      ;
D6A9 C9 C9      594      cmp #TKMINUS
D6AB F0 04      595      beq >1
D6AD           596      ;
D6AD C9 2C      597      cmp #', '
D6AF D0 F3      598      bne <0
D6B1           599      ;
D6B1 20 0C DA   600      ^1    jsr LINGET
D6B4 20 1A D6   601      jsr FNDLIN
D6B7           602      ;
D6B7 20 B7 00   603      jsr CHRGOT
D6BA F0 10      604      beq >3
D6BC           605      ;
D6BC C9 C9      606      cmp #TKMINUS
D6BE F0 04      607      beq >2
D6C0           608      ;
D6C0 C9 2C      609      cmp #', '

```

```

D6C2 D0 E0      610      bne <0
D6C4           611      ;
D6C4 20 B1 00   612      ^2      jsr CHRGET
D6C7           613      ;
D6C7 20 0C DA   614      jsr LINGET
D6CA D0 D8      615      bne <0
D6CC           616      ;
D6CC 68         617      ^3      pla
D6CD 68         618      pla
D6CE           619      ;
D6CE A5 50      620      lda ACL
D6D0 05 51      621      ora ACH
D6D2 D0 06      622      bne >4
D6D4           623      ;
D6D4 A9 FF      624      lda #NEGONE
D6D6 85 50      625      sta ACL
D6D8 85 51      626      sta ACH
D6DA           627      ;
D6DA A0 01      628      ^4      ldy #1
D6DC           629      ;
D6DC B1 9B      630      lda (LOWTR),Y
D6DE F0 44      631      beq >1
D6E0           632      ;
D6E0 20 58 D8   633      jsr ISCNTLC
D6E3 20 FB DA   634      jsr PRTCR
D6E6           635      ;
D6E6 C8         636      iny
D6E7           637      ;
D6E7 B1 9B      638      lda (LOWTR),Y
D6E9 AA         639      tax
D6EA           640      ;
D6EA C8         641      iny
D6EB           642      ;
D6EB B1 9B      643      lda (LOWTR),Y
D6ED C5 51      644      cmp ACH
D6EF D0 04      645      bne >5
D6F1           646      ;
D6F1 E4 50      647      cpx ACL
D6F3 F0 02      648      beq >6
D6F5           649      ;
D6F5 B0 2D      650      ^5      bcs >1
D6F7           651      ;
D6F7 84 85      652      ^6      sty FORPNT
D6F9           653      ;
D6F9 20 BC F7   654      jsr BLISTEX1
D6FC           655      ;
D6FC A9 20      656      lda #` `
D6FE           657      ;
D6FE A4 85      658      ^7      ldy FORPNT
D700           659      ;
D700 29 7F      660      and #MSBCLR
D702           661      ;
D702 20 5C DB   662      ^8      jsr OUTCHR
D705           663      ;
D705 20 C6 F7   664      jsr BLISTEX2
D708 EA         665      nop
D709 90 07      666      bcc >9
D70B           667      ;
D70B 20 FB DA   668      jsr PRTCR
D70E           669      ;
D70E A9 05      670      lda #5

```

```

D710 85 24      671      sta CH
D712           672      ;
D712 C8         673      ^9      iny
D713           674      ;
D713 B1 9B      675      lda (LOWTR),Y
D715 D0 1D      676      bne >2
D717           677      ;
D717 A8         678      tay
D718           679      ;
D718 B1 9B      680      lda (LOWTR),Y
D71A AA         681      tax
D71B           682      ;
D71B C8         683      iny
D71C           684      ;
D71C B1 9B      685      lda (LOWTR),Y
D71E           686      ;
D71E 86 9B      687      stx LOWTR
D720 85 9C      688      sta LOWTR+1
D722           689      ;
D722 D0 B6      690      bne <4
D724           691      ;
D724 A9 0D      692      ^1      lda #RETURN&MSBCLR
D726 20 5C DB   693      jsr OUTCHR
D729           694      ;
D729 4C D2 D7   695      jmp NEWSTT
D72C           696      ;
D72C           697      ;
D72C C8         698      GETCHR    iny
D72D D0 02      699      bne >1
D72F           700      ;
D72F E6 9E      701      inc DSCTMP+1
D731           702      ;
D731 B1 9D      703      ^1      lda (DSCTMP),Y
D733           704      ;
D733 60         705      rts
D734           706      ;
D734           707      ;
D734 10 CC      708      ^2      bpl <8
D736           709      ;
D736 38         710      sec
D737           711      ;
D737 E9 7F      712      sbc #$7F
D739 AA         713      tax
D73A           714      ;
D73A 84 85      715      sty FORPNT
D73C           716      ;
D73C A0 D2      717      ldy #BASNAME-PAGESIZE
D73E 84 9D      718      sty DSCTMP
D740           719      ;
D740 A0 CF      720      ldy /BASNAME-PAGESIZE
D742 84 9E      721      sty DSCTMP+1
D744           722      ;
D744 A0 FF      723      ldy #NEGONE
D746           724      ;
D746 CA         725      ^3      dex
D747 F0 07      726      beq >5
D749           727      ;
D749 20 2C D7   728      ^4      jsr GETCHR
D74C 10 FB      729      bpl <4
D74E           730      ;
D74E 30 F6      731      bmi <3

```

```

D750          732 ;
D750 A9 20    733 ^5      lda #$20
D752 20 5C DB 734      jsr OUTCHR
D755          735 ;
D755 20 2C D7 736 ^6      jsr GETCHR
D758 30 05    737      bmi >8
D75A          738 ;
D75A 20 5C DB 739      jsr OUTCHR
D75D D0 F6    740      bne <6
D75F          741 ;
D75F 20 5C DB 742 ^8      jsr OUTCHR
D762          743 ;
D762 A9 20    744      lda #$20
D764 D0 98    745      bne <7          ; always taken
D766          746 ;
D766          747 ;
D766 A9 80    748 BFOR    lda #$80
D768 85 14    749      sta SUBFLG
D76A          750 ;
D76A 20 46 DA 751      jsr BLET
D76D          752 ;
D76D 20 62 D3 753      jsr GTFORPNT
D770 D0 05    754      bne >1
D772          755 ;
D772 8A       756      txa
D773 69 0F    757      adc #15
D775 AA       758      tax
D776          759 ;
D776 9A       760      txs
D777          761 ;
D777 68       762 ^1      pla
D778 68       763      pla
D779          764 ;
D779 A9 09    765      lda #9
D77B 20 D6 D3 766      jsr CKSTKSIZ
D77E          767 ;
D77E 20 A3 D9 768      jsr DATSCAN
D781          769 ;
D781 18       770      clc
D782          771 ;
D782 98       772      tya
D783 65 B8    773      adc TXTPTR
D785 48       774      pha
D786          775 ;
D786 A5 B9    776      lda TXTPTR+1
D788 69 00    777      adc #ZERO
D78A 48       778      pha
D78B          779 ;
D78B A5 76    780      lda CURLIN+1
D78D 48       781      pha
D78E          782 ;
D78E A5 75    783      lda CURLIN
D790 48       784      pha
D791          785 ;
D791 A9 C1    786      lda #TKTO
D793 20 C0 DE 787      jsr SYNTAXCHK
D796          788 ;
D796 20 6A DD 789      jsr CHKNUM
D799 20 67 DD 790      jsr FRMNUM
D79C          791 ;
D79C A5 A2    792      lda FACSIGN

```



```

D79E 09 7F      793      ora #$7F
D7A0 25 9E      794      and FACMANT
D7A2 85 9E      795      sta FACMANT
D7A4           796      ;
D7A4 A9 AF      797      lda #FP1.0ADR
D7A6 A0 D7      798      ldy /FP1.0ADR
D7A8           799      ;
D7A8 85 5E      800      sta INDEX
D7AA 84 5F      801      sty INDEX+1
D7AC           802      ;
D7AC 4C 23 DE   803      jmp FRMSTAK3
D7AF           804      ;
D7AF           805      ;
D7AF A9 04      806      FP1.0ADR lda #FP1.0
D7B1 A0 EF      807      ldy /FP1.0
D7B3           808      ;
D7B3 20 F9 EA   809      jsr LOADFAC
D7B6 20 B7 00   810      jsr CHRGOT
D7B9           811      ;
D7B9 C9 C7      812      cmp #TKSTEP
D7BB D0 06      813      bne >1
D7BD           814      ;
D7BD 20 B1 00   815      jsr CHRGET
D7C0 20 67 DD   816      jsr FRMNUM
D7C3           817      ;
D7C3 20 82 EB   818      ^1 jsr CKFACSGN
D7C6 20 15 DE   819      jsr FRMSTAK2
D7C9           820      ;
D7C9 A5 86      821      lda FORPNT+1
D7CB 48          822      pha
D7CC           823      ;
D7CC A5 85      824      lda FORPNT
D7CE 48          825      pha
D7CF           826      ;
D7CF A9 81      827      lda #$80+BSFOR/2
D7D1 48          828      pha
D7D2           829      ;
D7D2           830      ;
D7D2           831      ; Used by DOS for the RUN and CHAIN commands as ASROMNEW.
D7D2           832      ;
D7D2 BA         833      NEWSTT tsx
D7D3 86 F8      834      stx REMSTK
D7D5           835      ;
D7D5 20 58 D8   836      jsr ISCNTLC
D7D8           837      ;
D7D8 A5 B8      838      lda TXTPTR
D7DA A4 B9      839      ldy TXTPTR+1
D7DC           840      ;
D7DC A6 76      841      ldx CURLIN+1      ; MODE?
D7DE E8         842      inx
D7DF F0 04      843      beq >1      ; DIRECT
D7E1           844      ;
D7E1 85 79      845      sta TEXTPTR      ; RUNNING
D7E3 84 7A      846      sty TEXTPTR+1
D7E5           847      ;
D7E5 A0 00      848      ^1 ldy #ZERO
D7E7           849      ;
D7E7 B1 B8      850      lda (TXTPTR),Y
D7E9 D0 57      851      bne >4
D7EB           852      ;
D7EB 18         853      clc

```

```
D7EC          854 ;
D7EC A0 02    855     ldy #2
D7EE          856 ;
D7EE B1 B8    857     lda (TXTPTR),Y
D7F0 F0 34    858     beq >2
D7F2          859 ;
D7F2 C8       860     iny
D7F3          861 ;
D7F3 B1 B8    862     lda (TXTPTR),Y
D7F5 85 75    863     sta CURLIN
D7F7          864 ;
D7F7 C8       865     iny
D7F8          866 ;
D7F8 B1 B8    867     lda (TXTPTR),Y
D7FA 85 76    868     sta CURLIN+1
D7FC          869 ;
D7FC 98       870     tya
D7FD          871 ;
D7FD 65 B8    872     adc TXTPTR
D7FF 85 B8    873     sta TXTPTR
D801 90 02    874     bcc DOTRACE
D803          875 ;
D803 E6 B9    876     inc TXTPTR+1
D805          877 ;
D805          878 ;
D805          879     icl "D8.L"
```

LLOAD D8.L,A\$4000

```

D805          1          ttl "ROM Source Code, D8.L"
D805          2          ;
D805          3          ;
D805          4          ; D8.L
D805          5          ;
D805          6          ;
D805 24 F2      7  DOTRACE  bit  TRACEFLG
D807 10 14      8          bpl >1
D809          9          ;
D809 A6 76     10         ldx CURLIN+1
D80B E8        11         inx
D80C F0 0F     12         beq >1
D80E          13         ;
D80E A9 23     14         lda #$23
D810 20 5C DB  15         jsr OUTCHR
D813          16         ;
D813 A6 75     17         ldx CURLIN
D815 A5 76     18         lda CURLIN+1
D817          19         ;
D817 20 24 ED  20         jsr LINPRT
D81A 20 57 DB  21         jsr OUTSPC
D81D          22         ;
D81D 20 B1 00  23         ^1      jsr CHRGET
D820 20 28 D8  24         jsr DOSTAMT
D823          25         ;
D823 4C D2 D7  26         jmp  NEWSTT
D826          27         ;
D826          28         ;
D826 F0 62     29         ^2      beq >6                ; extended branch
D828          30         ;
D828          31         ;
D828 F0 2D     32         DOSTAMT  beq RTN.D8.5          ; end of line?
D82A          33         ;
D82A          34         ;
D82A E9 80     35         DOSTAMT2  sbc #$80            ; remove bias
D82C 90 11     36         bcc >3
D82E          37         ;
D82E          38         ;
D82E          39         ; Separate BASADDR AND FN1ADDR statements.
D82E          40         ;
D82E C9 40     41         cmp #FN1ADDR/2
D830 B0 14     42         bcs >5
D832          43         ;
D832 0A        44         asl
D833 A8        45         tay
D834          46         ;
D834 B9 01 D0  47         lda BASADDR+1,Y
D837 48        48         pha
D838          49         ;
D838 B9 00 D0  50         lda BASADDR,Y
D83B 48        51         pha
D83C          52         ;
D83C 4C B1 00  53         jmp  CHRGET
D83F          54         ;
D83F 4C 46 DA  55         ^3      jmp  BLET
D842          56         ;
D842          57         ;
D842 C9 3A     58         ^4      cmp #' ':'          ; separator
D844 F0 BF     59         beq  DOTRACE
D846          60         ;

```

```

D846 4C C9 DE    61 ^5      jmp DOMESG02
D849             62 ;
D849             63 ;
D849 38          64 BRESTORE sec
D84A             65 ;
D84A A5 67       66      lda PRGTAB
D84C E9 01       67      sbc #1
D84E             68 ;
D84E A4 68       69      ldy PRGTAB+1
D850             70 ;
D850 B0 01       71      bcs SETDAPTR
D852             72 ;
D852 88          73      dey
D853             74 ;
D853 85 7D       75 SETDAPTR sta DATPTR
D855 84 7E       76      sty DATPTR+1
D857             77 ;
D857 60          78 RTN.D8.5 rts
D858             79 ;
D858             80 ;
D858             81 ; Modified the ISCNTLC routine by moving the contents of
D858             82 ; INCHR to 0xD85F and moving the branch to HANDLERR.
D858             83 ;
D858 AD 00 C0    84 ISCNTLC  lda KEY
D85B C9 83       85      cmp #CTRLC
D85D F0 01       86      beq >1
D85F             87 ;
D85F 60          88      rts
D860             89 ;
D860 2C 10 C0    90 ^1      bit CLRKEY
D863             91 ;
D863 A2 FF       92 DOCTRL.C ldx #ERROR.1      ; ctrl-C input error
D865             93 ;
D865             94 ;
D865             95 ; This is the DOS default ERRORADR.
D865             96 ;
D865 24 D8       97      bit ERRFLG
D867 30 47       98      bmi DOHANDER
D869             99 ;
D869 EA          100     nop
D86A             101 ;
D86A 29 7F       102     and #MSBCLR
D86C C9 03       103     cmp #CTRLC&MSBCLR    ; set C=1, Z=1
D86E             104 ;
D86E             105 ;
D86E B0 01       106 BSTOP   bcs >3
D870             107 ;
D870             108 ;
D870 18          109 BEND    clc
D871             110 ;
D871 D0 3C       111 ^3      bne >9
D873             112 ;
D873 A5 B8       113      lda TXTPTR
D875 A4 B9       114      ldy TXTPTR+1
D877             115 ;
D877 A6 76       116      ldx CURLIN+1
D879 E8          117      inx
D87A F0 0C       118      beq >4
D87C             119 ;
D87C 85 79       120      sta TEXTPTR
D87E 84 7A       121      sty TEXTPTR+1

```

```

D880          122 ;
D880 A5 75    123     lda CURLIN
D882 A4 76    124     ldy CURLIN+1
D884          125 ;
D884 85 77    126     sta OLDLIN
D886 84 78    127     sty OLDLIN+1
D888          128 ;
D888 68       129 ^4    pla
D889 68       130     pla
D88A          131 ;
D88A 90 07    132 ^6    bcc >7
D88C          133 ;
D88C A9 59    134     lda #MESG20
D88E A0 D3    135     ldy /MESG20
D890          136 ;
D890 4C 31 D4 137     jmp PR.LINUM
D893          138 ;
D893 4C 3C D4 139 ^7    jmp RESTART
D896          140 ;
D896          141 ;
D896 D0 17    142 BCONT bne >9
D898          143 ;
D898 A2 D4    144     ldx #MESG16-MESGS ; Cannot Continue error
D89A          145 ;
D89A A4 7A    146     ldy TEXTPTR+1
D89C D0 03    147     bne >8
D89E          148 ;
D89E 4C 12 D4 149     jmp FSCRN
D8A1          150 ;
D8A1 A5 79    151 ^8    lda TEXTPTR
D8A3          152 ;
D8A3 85 B8    153     sta TXTPTR
D8A5 84 B9    154     sty TXTPTR+1
D8A7          155 ;
D8A7 A5 77    156     lda OLDLIN
D8A9 A4 78    157     ldy OLDLIN+1
D8AB          158 ;
D8AB 85 75    159     sta CURLIN
D8AD 84 76    160     sty CURLIN+1
D8AF          161 ;
D8AF 60       162 ^9    rts
D8B0          163 ;
D8B0 4C E9 F2 164 DOHANDER jmp HANDLERR
D8B3          165 ;
D8B3          166 ;
D8B3          167     dfs 9,ZERO ; 9 bytes
D8BC          168 ;
D8BC          169 ;
D8BC A2 08    170 RDBYTE ldx #8 ; bit counter
D8BE          171 ;
D8BE 48       172 ^6    pha ; push current data
D8BF          173 ;
D8BF 20 CA D8 174     jsr RD2BIT ; get data bit in C-flag
D8C2          175 ;
D8C2 68       176     pla ; recall data
D8C3 2A       177     rol ; roll in bit, MSB first
D8C4          178 ;
D8C4 A0 3A    179     ldy #$3A ; waveform change, 696 cycles
D8C6          180 ;
D8C6 CA       181     dex ; next bit
D8C7 D0 F5    182     bne <6

```

```

D8C9          183 ;
D8C9 60       184 ;           rts
D8CA          185 ;
D8CA          186 ;
D8CA          187 ; Read audio waveform for two transitions and return bit
D8CA          188 ; value in the C-flag.
D8CA          189 ;
D8CA 20 CD D8 190 RD2BIT    jsr RDBIT
D8CD          191 ;
D8CD 88       192 RDBIT     dey
D8CE          193 ;
D8CE AD 60 C0 194          lda TAPEIN          ; read waveform voltage level
D8D1 45 2F    195          eor LASTIN         ; compare MSB to last read
D8D3 10 F8    196          bpl RDBIT          ; branch if no change
D8D5          197 ;
D8D5 45 2F    198          eor LASTIN         ; compare MSB to last saved
D8D7 85 2F    199          sta LASTIN         ; save the new MSB
D8D9          200 ;
D8D9 C0 80    201          cpy #$80           ; get bit value into C-flag
D8DB          202 ;
D8DB 60       203          rts
D8DC          204 ;
D8DC          205 ;
D8DC          206 ; Applesoft LOAD command for c2t processing. Copy address
D8DC          207 ; of LINNUM to A1 for start address. Copy address of
D8DC          208 ; LINNUM+2 to A2 for end address. Thus, read two data
D8DC          209 ; bytes, the RUNFLAG, and a checksum.
D8DC          210 ;
D8DC A9 50    211 BLOAD     lda #LINNUM
D8DE A0 00    212          ldy /LINNUM
D8E0          213 ;
D8E0 85 3C    214          sta A1L
D8E2 84 3D    215          sty A1H
D8E4          216 ;
D8E4 A9 52    217          lda #LINNUM+2
D8E6          218 ;
D8E6 85 3E    219          sta A2L
D8E8 84 3F    220          sty A2H
D8EA          221 ;
D8EA 84 D6    222          sty RUNFLAG
D8EC          223 ;
D8EC 20 9F F3 224          jsr CXREAD
D8EF          225 ;
D8EF          226 ;
D8EF          227 ; Copy address in PRGTAB to A1 for start address. Add
D8EF          228 ; address in PRGTAB to LINNUM for VARTAB and A2 for end
D8EF          229 ; address. Thus, read LINNUM bytes and a checksum.
D8EF          230 ;
D8EF 18       231          clc
D8F0          232 ;
D8F0 A5 67    233          lda PRGTAB
D8F2 85 3C    234          sta A1L
D8F4          235 ;
D8F4 65 50    236          adc LINNUM
D8F6 85 69    237          sta VARTAB
D8F8 85 3E    238          sta A2L
D8FA          239 ;
D8FA A5 68    240          lda PRGTAB+1
D8FC 85 3D    241          sta A1H
D8FE          242 ;
D8FE 65 51    243          adc LINNUM+1

```

```

D900 85 6A      244      sta VARTAB+1
D902 85 3F      245      sta A2H
D904           246      ;
D904 A5 52      247      lda TEMPPT
D906 85 D6      248      sta RUNFLAG
D908           249      ;
D908 20 9F F3   250      jsr CXREAD          ; read Applesoft program
D90B           251      ;
D90B 24 D6      252      bit RUNFLAG
D90D 30 09      253      bmi >1
D90F           254      ;
D90F 4C F2 D4   255      jmp ASENTER          ; enter Applesoft interpreter
D912           256      ;
D912           257      ;
D912 08         258      BRUN      php
D913           259      ;
D913 C6 76      260      dec CURLIN+1
D915           261      ;
D915 28         262      plp
D916 D0 03      263      bne >2
D918           264      ;
D918 4C 65 D6   265      ^1      jmp SETPTRS          ; run the Applesoft program
D91B           266      ;
D91B 20 6C D6   267      ^2      jsr CLEARC
D91E           268      ;
D91E 4C 35 D9   269      jmp BGOSUB2
D921           270      ;
D921           271      ;
D921 A9 03      272      BGOSUB      lda #3
D923 20 D6 D3   273      jsr CKSTKSIZ
D926           274      ;
D926 A5 B9      275      lda TXTPTR+1
D928 48         276      pha
D929           277      ;
D929 A5 B8      278      lda TXTPTR
D92B 48         279      pha
D92C           280      ;
D92C A5 76      281      lda CURLIN+1
D92E 48         282      pha
D92F           283      ;
D92F A5 75      284      lda CURLIN
D931 48         285      pha
D932           286      ;
D932 A9 B0      287      lda #$80+BSGOSUB/2
D934 48         288      pha
D935           289      ;
D935 20 B7 00   290      BGOSUB2      jsr CHRGOT
D938 20 3E D9   291      jsr BGOTO
D93B           292      ;
D93B 4C D2 D7   293      jmp NEWSTT
D93E           294      ;
D93E           295      ;
D93E 20 0C DA   296      BGOTO      jsr LINGET
D941 20 A6 D9   297      jsr DATSCAN2
D944           298      ;
D944 A5 76      299      lda CURLIN+1
D946 C5 51      300      cmp ACH
D948 B0 0B      301      bcs >1
D94A           302      ;
D94A 98         303      tya
D94B           304      ;

```

```

D94B 38          305          sec
D94C          306          ;
D94C 65 B8      307          adc TXTPTR
D94E          308          ;
D94E A6 B9      309          ldx TXTPTR+1
D950          310          ;
D950 90 07      311          bcc >2
D952          312          ;
D952 E8         313          inx
D953          314          ;
D953 B0 04      315          bcs >2
D955          316          ;
D955          317          ;
D955          318          ; Used by DOS for the RUN and CHAIN commands as ASROMSET
D955          319          ; to initialize the start of Applesoft when LINNUM is
D955          320          ; specified in these commands.
D955          321          ;
D955 A5 67      322          ^1      lda PRGTAB
D957 A6 68      323          ldx PRGTAB+1
D959          324          ;
D959 20 1E D6   325          ^2      jsr FNDLIN2
D95C 90 1E      326          bcc DOMESG08
D95E          327          ;
D95E A5 9B      328          lda LOWTR
D960 E9 01      329          sbc #1
D962 85 B8      330          sta TXTPTR
D964          331          ;
D964 A5 9C      332          lda LOWTR+1
D966 E9 00      333          sbc #ZERO
D968 85 B9      334          sta TXTPTR+1
D96A          335          ;
D96A 60         336          RTN.D9.6 rts
D96B          337          ;
D96B          338          ;
D96B          339          ; Entry for POP and RETURN statements.  Fixed FORPNT bug.
D96B          340          ;
D96B          341          BRETURN:
D96B D0 FD      342          BPOP      bne RTN.D9.6
D96D          343          ;
D96D A9 FF      344          lda #NEGONE
D96F          345          ;
D96F          346          ;      sta FORPNT
D96F 85 86      347          sta FORPNT+1
D971          348          ;
D971 20 62 D3   349          jsr GTFORPNT
D974          350          ;
D974 9A         351          txs
D975          352          ;
D975 C9 B0      353          cmp #$80+BSGOSUB/2
D977 F0 0B      354          beq >1
D979          355          ;
D979 A2 16      356          ldx #MESG03-MESGS      ; RETURN without GOSUB error
D97B          357          ;
D97B 2C 00 00   358          bit *-*
D97E          359          dfs !-2
D97C          360          ;
D97C A2 5A      361          DOMESG08 ldx #MESG08-MESGS      ; Undefined Statement error
D97E          362          ;
D97E 4C 12 D4   363          jmp FSCRN
D981          364          ;
D981          365          ;

```



```

D981 4C C9 DE 366 PRMSG02 jmp DOMESG02
D984 367 ;
D984 368 ;
D984 68 369 ^1 pla
D985 68 370 pla
D986 371 ;
D986 C0 42 372 cpy #2*$80+BSPOP/2
D988 F0 3B 373 beq >5
D98A 374 ;
D98A 85 75 375 sta CURLIN
D98C 376 ;
D98C 68 377 pla
D98D 85 76 378 sta CURLIN+1
D98F 379 ;
D98F 68 380 pla
D990 85 B8 381 sta TXTPTR
D992 382 ;
D992 68 383 pla
D993 85 B9 384 sta TXTPTR+1
D995 385 ;
D995 386 ;
D995 20 A3 D9 387 BDATA jsr DATSCAN
D998 388 ;
D998 98 389 BDATA2 tya
D999 390 ;
D999 18 391 clc
D99A 392 ;
D99A 65 B8 393 adc TXTPTR
D99C 85 B8 394 sta TXTPTR
D99E 90 02 395 bcc >2
D9A0 396 ;
D9A0 E6 B9 397 inc TXTPTR+1
D9A2 398 ;
D9A2 60 399 ^2 rts
D9A3 400 ;
D9A3 401 ;
D9A3 A2 3A 402 DATSCAN ldx #$3A
D9A5 403 ;
D9A5 2C 00 00 404 bit *-*
D9A8 405 dfs !-2
D9A6 406 ;
D9A6 A2 00 407 DATSCAN2 ldx #ZERO
D9A8 408 ;
D9A8 86 0D 409 stx CHARAC
D9AA 410 ;
D9AA A0 00 411 ldy #ZERO
D9AC 84 0E 412 sty ENDCHR
D9AE 413 ;
D9AE A5 0E 414 ^3 lda ENDCHR
D9B0 A6 0D 415 ldx CHARAC
D9B2 416 ;
D9B2 85 0D 417 sta CHARAC
D9B4 86 0E 418 stx ENDCHR
D9B6 419 ;
D9B6 B1 B8 420 ^4 lda (TXTPTR),Y
D9B8 F0 E8 421 beq <2
D9BA 422 ;
D9BA C5 0E 423 cmp ENDCHR
D9BC F0 E4 424 beq <2
D9BE 425 ;
D9BE C8 426 iny

```

```

D9BF          427 ;
D9BF C9 22    428      cmp #' " '
D9C1 D0 F3    429      bne <4
D9C3          430 ;
D9C3 F0 E9    431      beq <3                ; always taken
D9C5          432 ;
D9C5 68       433 ^5      pla
D9C6 68       434      pla
D9C7 68       435      pla
D9C8          436 ;
D9C8 60       437      rts
D9C9          438 ;
D9C9          439 ;
D9C9 20 7B DD 440 BIF     jsr FRMEVAL
D9CC 20 B7 00 441      jsr CHRGOT
D9CF          442 ;
D9CF C9 AB    443      cmp # $80+BSGOTO/2
D9D1 F0 05    444      beq >1
D9D3          445 ;
D9D3 A9 C4    446      lda #TKTHEN
D9D5 20 C0 DE 447      jsr SYNTAXCHK
D9D8          448 ;
D9D8 A5 9D    449 ^1     lda DSCTMP
D9DA D0 05    450      bne >2
D9DC          451 ;
D9DC          452 ;
D9DC 20 A6 D9 453 BREM    jsr DATSCAN2
D9DF F0 B7    454      beq BDATA2
D9E1          455 ;
D9E1 20 B7 00 456 ^2     jsr CHRGOT
D9E4 B0 03    457      bcs >3
D9E6          458 ;
D9E6 4C 3E D9 459      jmp BGOTO
D9E9          460 ;
D9E9 4C 28 D8 461 ^3     jmp DOSTAMT
D9EC          462 ;
D9EC          463 ;
D9EC 20 F8 E6 464 BON     jsr GETBYT
D9EF          465 ;
D9EF 48       466      pha
D9F0          467 ;
D9F0 C9 B0    468      cmp # $80+BSGOSUB/2
D9F2 F0 04    469      beq >5
D9F4          470 ;
D9F4 C9 AB    471 ^4     cmp # $80+BSGOTO/2
D9F6 D0 89    472      bne PRMMSG02
D9F8          473 ;
D9F8 C6 A1    474 ^5     dec VARPTR+1
D9FA D0 04    475      bne >6
D9FC          476 ;
D9FC 68       477      pla
D9FD          478 ;
D9FD 4C 2A D8 479      jmp DOSTAMT2
DA00          480 ;
DA00 20 B1 00 481 ^6     jsr CHRGET
DA03 20 0C DA 482      jsr LINGET
DA06          483 ;
DA06 C9 2C    484      cmp #' , '
DA08 F0 EE    485      beq <5
DA0A          486 ;
DA0A 68       487      pla

```

```

DA0B          488 ;
DA0B 60       489 RTN.DA.0 rts
DA0C          490 ;
DA0C          491 ;
DA0C A2 00    492 LINGET     ldx #ZERO
DA0E 86 50    493             stx ACL
DA10 86 51    494             stx ACH
DA12          495 ;
DA12 B0 F7    496 LINGET2    bcs RTN.DA.0
DA14          497 ;
DA14 E9 2F    498             sbc #'0'-1
DA16 85 0D    499             sta CHARAC
DA18          500 ;
DA18 A5 51    501             lda ACH
DA1A 85 5E    502             sta INDEX
DA1C          503 ;
DA1C C9 19    504             cmp /6400
DA1E B0 D4    505             bcs <4
DA20          506 ;
DA20 A5 50    507             lda ACL
DA22          508 ;
DA22 0A       509             asl
DA23 26 5E    510             rol INDEX
DA25          511 ;
DA25 0A       512             asl
DA26 26 5E    513             rol INDEX
DA28          514 ;
DA28 65 50    515             adc ACL
DA2A 85 50    516             sta ACL
DA2C          517 ;
DA2C A5 5E    518             lda INDEX
DA2E 65 51    519             adc ACH
DA30 85 51    520             sta ACH
DA32          521 ;
DA32 06 50    522             asl ACL
DA34 26 51    523             rol ACH
DA36          524 ;
DA36 A5 50    525             lda ACL
DA38 65 0D    526             adc CHARAC
DA3A 85 50    527             sta ACL
DA3C 90 02    528             bcc >1
DA3E          529 ;
DA3E E6 51    530             inc ACH
DA40          531 ;
DA40 20 B1 00 532 ^1         jsr CHRGET
DA43          533 ;
DA43 4C 12 DA 534             jmp LINGET2
DA46          535 ;
DA46          536 ;
DA46 20 E3 DF 537 BLET       jsr PTRGET
DA49          538 ;
DA49 85 85    539             sta FORPNT
DA4B 84 86    540             sty FORPNT+1
DA4D          541 ;
DA4D A9 D0    542             lda #TKEQUAL
DA4F 20 C0 DE 543             jsr SYNTAXCHK
DA52          544 ;
DA52 A5 12    545             lda VALTYP+1
DA54 48       546             pha
DA55          547 ;
DA55 A5 11    548             lda VALTYP

```

```

DA57 48          549      pha
DA58          550      ;
DA58 20 7B DD    551      jsr FRMEVAL
DA5B          552      ;
DA5B 68          553      pla
DA5C 2A          554      rol
DA5D          555      ;
DA5D 20 6D DD    556      jsr CHKVAL
DA60 D0 18       557      bne >3
DA62          558      ;
DA62 68          559      pla
DA63          560      ;
DA63          561      ;
DA63 10 12       562      BLET2    bpl >2
DA65          563      ;
DA65 20 72 EB    564      jsr RNDUP
DA68 20 0C E1    565      jsr AYINT
DA6B          566      ;
DA6B A0 00       567      ldy #ZERO
DA6D          568      ;
DA6D A5 A0       569      lda VARPTR
DA6F 91 85       570      sta (FORPNT),Y
DA71          571      ;
DA71 C8          572      iny
DA72          573      ;
DA72 A5 A1       574      lda VARPTR+1
DA74 91 85       575      sta (FORPNT),Y
DA76          576      ;
DA76 60          577      rts
DA77          578      ;
DA77 4C 27 EB    579      ^2      jmp COPYF2FR
DA7A          580      ;
DA7A 68          581      ^3      pla
DA7B          582      ;
DA7B          583      ;
DA7B A0 02       584      PUTSTR    ldy #2
DA7D          585      ;
DA7D B1 A0       586      lda (VARPTR),Y
DA7F C5 70       587      cmp FRETOP+1
DA81 90 17       588      bcc >5
DA83          589      ;
DA83 D0 07       590      bne >4
DA85          591      ;
DA85 88          592      dey
DA86          593      ;
DA86 B1 A0       594      lda (VARPTR),Y
DA88 C5 6F       595      cmp FRETOP
DA8A 90 0E       596      bcc >5
DA8C          597      ;
DA8C A4 A1       598      ^4      ldy VARPTR+1
DA8E C4 6A       599      cpy VARTAB+1
DA90 90 08       600      bcc >5
DA92          601      ;
DA92 D0 0D       602      bne >6
DA94          603      ;
DA94 A5 A0       604      lda VARPTR
DA96 C5 69       605      cmp VARTAB
DA98 B0 07       606      bcs >6
DA9A          607      ;
DA9A A5 A0       608      ^5      lda VARPTR
DA9C A4 A1       609      ldy VARPTR+1

```

```

DA9E          610 ;
DA9E          611 ;      jmp COPYSTR
DA9E          612 ;
DA9E 90 17    613      bcc COPYSTR      ; always taken
DAA0          614 ;
DAA0          615      dfs 1,ZERO      ; 1 byte
DAA1          616 ;
DAA1 A0 00    617 ^6      ldy #ZERO
DAA3          618 ;
DAA3 B1 A0    619      lda (VARPTR),Y
DAA5 20 D5 E3 620      jsr STRINI
DAA8          621 ;
DAA8 A5 8C    622      lda DSCPTR
DAAA A4 8D    623      ldy DSCPTR+1
DAAC          624 ;
DAAC 85 AB    625      sta STRING1
DAAE 84 AC    626      sty STRING1+1
DAB0          627 ;
DAB0 20 D4 E5 628      jsr MOVINS
DAB3          629 ;
DAB3 A9 9D    630      lda #FACEXP
DAB5 A0 00    631      ldy /FACEXP
DAB7          632 ;
DAB7          633 ;
DAB7 85 8C    634 COPYSTR sta DSCPTR
DAB9 84 8D    635      sty DSCPTR+1
DABB          636 ;
DABB 20 35 E6 637      jsr FRETMS
DABE          638 ;
DABE A0 00    639      ldy #ZERO
DAC0          640 ;
DAC0 B1 8C    641      lda (DSCPTR),Y
DAC2 91 85    642      sta (FORPNT),Y
DAC4          643 ;
DAC4 C8       644      iny
DAC5          645 ;
DAC5 B1 8C    646      lda (DSCPTR),Y
DAC7 91 85    647      sta (FORPNT),Y
DAC9          648 ;
DAC9 C8       649      iny
DACA          650 ;
DACA B1 8C    651      lda (DSCPTR),Y
DACC 91 85    652      sta (FORPNT),Y
DACE          653 ;
DACE 60       654      rts
DACF          655 ;
DACF          656 ;
DACF 20 3D DB 657 PRSTRING jsr STRPRT
DAD2 20 B7 00 658      jsr CHRGOT
DAD5          659 ;
DAD5          660 ;
DAD5 F0 24    661 BPRINT  beq PRTCR
DAD7          662 ;
DAD7 F0 29    663 BPRINT2  beq RTN.DB.0
DAD9          664 ;
DAD9 C9 C0    665      cmp #TKTAB
DADB F0 3C    666      beq >4
DADD          667 ;
DADD C9 C3    668      cmp #TKSPC
DADF          669 ;
DADF 18       670      clc

```

```

DAE0          671 ;
DAE0 F0 37    672      beq >4
DAE2          673 ;
DAE2 C9 2C    674      cmp #' , '
DAE4          675 ;
DAE4 18       676      clc
DAE5          677 ;
DAE5 F0 1C    678      beq >2
DAE7          679 ;
DAE7 C9 3B    680      cmp #' ; '
DAE9 F0 44    681      beq >8
DAEB          682 ;
DAEB 20 7B DD 683      jsr FRMEVAL
DAEE          684 ;
DAEE 24 11    685      bit VALTYP
DAF0 30 DD    686      bmi PRSTRING
DAF2          687 ;
DAF2 20 34 ED 688      jsr FPOUT
DAF5 20 E7 E3 689      jsr STRLIT
DAF8          690 ;
DAF8 4C CF DA 691      jmp PRSTRING
DAFB          692 ;
DAFB          693 ;
DAFB A9 0D    694 PRTCR   lda #RETURN&MSBCLR
DAFD 20 5C DB 695      jsr OUTCHR
DB00          696 ;
DB00 49 FF    697      eor #NEGONE
DB02          698 ;
DB02 60       699 RTN.DB.0 rts
DB03          700 ;
DB03 20 C6 F7 701      ^2      jsr PRTCRESX1
DB06 30 09    702      bmi >3
DB08          703 ;
DB08 C9 18    704      cmp #$18
DB0A 90 05    705      bcc >3
DB0C          706 ;
DB0C 20 FB DA 707      jsr PRTCR
DB0F D0 1E    708      bne >8
DB11          709 ;
DB11 69 10    710      ^3      adc #$10
DB13 29 F0    711      and #$F0
DB15 AA       712      tax
DB16          713 ;
DB16 38       714      sec
DB17 B0 0C    715      bcs >5          ; always taken
DB19          716 ;
DB19 08       717      ^4      php
DB1A          718 ;
DB1A 20 F5 E6 719      jsr GETBYTC
DB1D          720 ;
DB1D C9 29    721      cmp #' ) '
DB1F D0 62    722      bne >4
DB21          723 ;
DB21 28       724      plp
DB22 90 07    725      bcc >6
DB24          726 ;
DB24 CA       727      dex
DB25          728 ;
DB25 20 D5 F7 729      ^5      jsr PRTCRESX2
DB28 90 05    730      bcc >8
DB2A          731 ;

```

```

DB2A AA          732          tax
DB2B             733          ;
DB2B E8          734          ^6      inx
DB2C             735          ;
DB2C CA          736          ^7      dex
DB2D D0 06       737          bne >9
DB2F             738          ;
DB2F 20 B1 00    739          ^8      jsr CHRGET
DB32             740          ;
DB32 4C D7 DA    741          jmp BPRINT2
DB35             742          ;
DB35 20 57 DB    743          ^9      jsr OUTSPC
DB38 D0 F2       744          bne <7
DB3A             745          ;
DB3A             746          ;
DB3A 20 E7 E3    747      STROUT      jsr STRLIT
DB3D             748          ;
DB3D             749          ;
DB3D             750          ; Remove unnecessary logic.
DB3D             751          ;
DB3D 20 00 E6    752      STRPRT      jsr FRESTR2
DB40             753          ;
DB40 AA          754          tax
DB41 E8          755          inx
DB42             756          ;
DB42 A0 00       757          ldy #ZERO
DB44             758          ;
DB44 CA          759          ^2      dex
DB45 F0 BB       760          beq RTN.DB.0
DB47             761          ;
DB47 B1 5E       762          lda (INDEX),Y
DB49 20 5C DB    763          jsr OUTCHR
DB4C             764          ;
DB4C C8          765          iny
DB4D             766          ;
DB4D             767          ; cmp #RETURN&MSBCLR
DB4D             768          ; bne <2
DB4D             769          ;
DB4D             770          ; eor #NEGONE
DB4D             771          ;
DB4D             772          ; jmp <2
DB4D             773          ;
DB4D D0 F5       774          bne <2          ; always taken
DB4F             775          ;
DB4F             776          ;
DB4F             777          ; dfs 8,ZERO          ; 8 bytes
DB57             778          ;
DB57             779          ;
DB57 A9 20       780      OUTSPC      lda #' '
DB59             781          ;
DB59 2C 00 00    782          bit *-*
DB5C             783          ; dfs !-2
DB5A             784          ;
DB5A             785          ;
DB5A             786      OUTQSTN:
DB5A             787          ; lda #'?'
DB5A A9 3E       788          ; lda #'>'
DB5C             789          ;
DB5C             790          ;
DB5C 09 80       791      OUTCHR      ora #MSBSET
DB5E             792          ;

```

```

DB5E C9 A0      793      cmp #SPACE
DB60 90 02      794      bcc >1
DB62           795      ;
DB62 05 F3      796      ora FLASHBYT
DB64           797      ;
DB64 20 ED FD   798      ^1      jsr COUT
DB67           799      ;
DB67 29 7F      800      and #MSBCLR
DB69 48         801      pha
DB6A           802      ;
DB6A A5 F1      803      lda SPEEDBYT
DB6C 20 A8 FC   804      jsr WAIT
DB6F           805      ;
DB6F 68         806      pla
DB70           807      ;
DB70 60         808      rts
DB71           809      ;
DB71           810      ;
DB71 A5 15      811      INPUTERR lda INPUTFLG
DB73 F0 12      812      beq >5
DB75           813      ;
DB75 30 04      814      bmi >2
DB77           815      ;
DB77 A0 FF      816      ldy #NEGONE
DB79 D0 04      817      bne >3
DB7B           818      ;
DB7B A5 7B      819      ^2      lda DATLIN
DB7D A4 7C      820      ldy DATLIN+1
DB7F           821      ;
DB7F 85 75      822      ^3      sta CURLIN
DB81 84 76      823      sty CURLIN+1
DB83           824      ;
DB83 4C C9 DE   825      ^4      jmp DOMESG02
DB86           826      ;
DB86           827      ;
DB86 68         828      INPERR  pla
DB87           829      ;
DB87 24 D8      830      ^5      bit ERRFLG
DB89 10 05      831      bpl >6
DB8B           832      ;
DB8B A2 FE      833      ldx #ERROR.2      ; bad response to INPUT
DB8D           834      ;
DB8D 4C E9 F2   835      jmp HANDLERR
DB90           836      ;
DB90 A9 EF      837      ^6      lda #MESG22      ; reenter
DB92 A0 DC      838      ldy /MESG22
DB94           839      ;
DB94 20 3A DB   840      jsr STROUT
DB97           841      ;
DB97 A5 79      842      lda TEXTPTR
DB99 A4 7A      843      ldy TEXTPTR+1
DB9B           844      ;
DB9B 85 B8      845      sta TXTPTR
DB9D 84 B9      846      sty TXTPTR+1
DB9F           847      ;
DB9F 60         848      rts
DBA0           849      ;
DBA0           850      ;
DBA0 20 06 E3   851      BGET      jsr ERRDIR
DBA3           852      ;
DBA3 A2 01      853      ldx #1

```



```

DBA5 A0 02      854      ldy #2
DBA7           855      ;
DBA7 A9 00      856      lda #ZERO
DBA9 8D 01 02   857      sta INPUT+1
DBAC           858      ;
DBAC A9 40      859      lda #%01000000      ; set bit 6
DBAE           860      ;
DBAE           861      ;      jsr DO.LIST
DBAE           862      ;
DBAE           863      ;      rts
DBAE           864      ;
DBAE 4C EB DB   865      jmp DO.LIST
DBB1           866      ;
DBB1           867      dfs 1,ZERO      ; 1 byte
DBB2           868      ;
DBB2           869      ;
DBB2 C9 22      870      BINPUT      cmp #'"'
DBB4 D0 0E      871      bne >7
DBB6           872      ;
DBB6 20 84 DE   873      jsr STRTXT
DBB9           874      ;
DBB9 A9 3B      875      lda #$3B
DBBB 20 C0 DE   876      jsr SYNTAXCHK
DBBE           877      ;
DBBE 20 3D DB   878      jsr STRPRT
DBC1           879      ;
DBC1 4C C7 DB   880      jmp >8
DBC4           881      ;
DBC4           882      ;
DBC4           883      icl "DC.L"

```

```

LLOAD DC.L,A$4000

```

```

DBC4      1          ttl "ROM Source Code, DC.L"
DBC4      2      ;
DBC4      3      ;
DBC4      4      ; DC.L
DBC4      5      ;
DBC4      6      ;
DBC4 20 5A DB      7      ^7      jsr OUTQSTN
DBC7      8      ;
DBC7 20 06 E3      9      ^8      jsr ERRDIR
DBCA     10      ;
DBCA A9 2C      11          lda #', '
DBCC 8D FF 01      12          sta INPUT-1
DBCF     13      ;
DBCF 20 2C D5      14          jsr INLIN
DBD2     15      ;
DBD2 AD 00 02      16          lda INPUT
DBD5 C9 03      17          cmp #CTRLC&MSBCLR
DBD7 D0 10      18          bne >9
DBD9     19      ;
DBD9 4C 63 D8      20          jmp DOCTRL.C
DBDC     21      ;
DBDC     22      ;
DBDC 20 5A DB      23      HEXTIN      jsr OUTQSTN
DBDF     24      ;
DBDF 4C 2C D5      25          jmp INLIN
DBE2     26      ;
DBE2     27      ;
DBE2 A6 7D      28      BREAD      ldx DATPTR
DBE4 A4 7E      29          ldy DATPTR+1
DBE6     30      ;
DBE6 A9 98      31          lda #%10011000      ; set bit 7
DBE8     32      ;
DBE8 2C 00 00      33          bit *-*
DBEB     34          dfs !-2
DBE9     35      ;
DBE9 A9 00      36      ^9      lda #%00000000
DBEB     37      ;
DBEB     38      ;
DBEB     39      ; INPUTFLG - 0x00 for INPUT
DBEB     40      ;          0x40 for GET
DBEB     41      ;          0x98 for READ
DBEB     42      ;
DBEB 85 15      43      DO.LIST      sta INPUTFLG
DBED     44      ;
DBED 86 7F      45          stx SRCPTR
DBEF 84 80      46          sty SRCPTR+1
DBF1     47      ;
DBF1 20 E3 DF      48      DO.ITEM      jsr PTRGET
DBF4     49      ;
DBF4 85 85      50          sta FORPNT
DBF6 84 86      51          sty FORPNT+1
DBF8     52      ;
DBF8 A5 B8      53          lda TXTPTR
DBFA A4 B9      54          ldy TXTPTR+1
DBFC     55      ;
DBFC 85 87      56          sta TXPTRSAV
DBFE 84 88      57          sty TXPTRSAV+1
DC00     58      ;
DC00 A6 7F      59          ldx SRCPTR
DC02 A4 80      60          ldy SRCPTR+1

```

```

DC04          61 ;
DC04 86 B8    62      stx TXTPTR
DC06 84 B9    63      sty TXTPTR+1
DC08          64 ;
DC08 20 B7 00 65      jsr CHRGOT
DC0B D0 1E    66      bne INSTART
DC0D          67 ;
DC0D 24 15    68      bit INPUTFLG
DC0F 50 0E    69      bvc >1
DC11          70 ;
DC11 20 13 FD 71      jsr RDKEY2
DC14          72 ;
DC14 29 7F    73      and #MSBCLR
DC16 8D 00 02 74      sta INPUT
DC19          75 ;
DC19 A2 FF    76      ldx #INPUT-1
DC1B A0 01    77      ldy /INPUT-1
DC1D D0 08    78      bne >2                ; always taken
DC1F          79 ;
DC1F 30 7F    80      ^1      bmi >0
DC21          81 ;
DC21 20 5A DB 82      jsr OUTQSTN
DC24 20 DC DB 83      jsr HEXTIN
DC27          84 ;
DC27 86 B8    85      ^2      stx TXTPTR
DC29 84 B9    86      sty TXTPTR+1
DC2B          87 ;
DC2B 20 B1 00 88      INSTART jsr CHRGET
DC2E          89 ;
DC2E 24 11    90      bit VALTYP
DC30 10 31    91      bpl >7
DC32          92 ;
DC32 24 15    93      bit INPUTFLG
DC34 50 09    94      bvc >3
DC36          95 ;
DC36 E8       96      inx
DC37 86 B8    97      stx TXTPTR
DC39          98 ;
DC39 A9 00    99      lda #ZERO
DC3B 85 0D   100      sta CHARAC
DC3D F0 0C   101      beq >4                ; always taken
DC3F          102 ;
DC3F 85 0D   103      ^3      sta CHARAC
DC41 C9 22   104      cmp #'"'
DC43 F0 07   105      beq >5
DC45          106 ;
DC45 A9 3A   107      lda #': '
DC47 85 0D   108      sta CHARAC
DC49          109 ;
DC49 A9 2C   110      lda #', '
DC4B          111 ;
DC4B 18       112      ^4      clc
DC4C          113 ;
DC4C 85 0E   114      ^5      sta ENDCHR
DC4E          115 ;
DC4E A5 B8   116      lda TXTPTR
DC50 A4 B9   117      ldy TXTPTR+1
DC52          118 ;
DC52 69 00   119      adc #ZERO
DC54 90 01   120      bcc >6
DC56          121 ;

```

```

DC56 C8          122      iny
DC57             123      ;
DC57 20 ED E3    124      ^6      jsr STRLIT2
DC5A 20 3D E7    125      jsr STRCOPY
DC5D 20 7B DA    126      jsr PUTSTR
DC60             127      ;
DC60 4C 72 DC    128      jmp >9
DC63             129      ;
DC63 48          130      ^7      pha
DC64             131      ;
DC64 AD 00 02    132      lda INPUT
DC67 F0 30       133      beq >2
DC69             134      ;
DC69 68          135      ^8      pla
DC6A 20 4A EC    136      jsr GETINT
DC6D             137      ;
DC6D A5 12       138      lda VALTYP+1
DC6F 20 63 DA    139      jsr BLET2
DC72             140      ;
DC72 20 B7 00    141      ^9      jsr CHRGOT
DC75 F0 07       142      beq >1
DC77             143      ;
DC77 C9 2C       144      cmp #' , '
DC79 F0 03       145      beq >1
DC7B             146      ;
DC7B 4C 71 DB    147      jmp INPUTERR
DC7E             148      ;
DC7E A5 B8       149      ^1      lda TXTPTR
DC80 A4 B9       150      ldy TXTPTR+1
DC82             151      ;
DC82 85 7F       152      sta SRCPTR
DC84 84 80       153      sty SRCPTR+1
DC86             154      ;
DC86 A5 87       155      lda TXPTRSAV
DC88 A4 88       156      ldy TXPTRSAV+1
DC8A             157      ;
DC8A 85 B8       158      sta TXTPTR
DC8C 84 B9       159      sty TXTPTR+1
DC8E             160      ;
DC8E 20 B7 00    161      jsr CHRGOT
DC91 F0 33       162      beq >4
DC93             163      ;
DC93 20 BE DE    164      jsr CHKCOM
DC96             165      ;
DC96 4C F1 DB    166      jmp DO.ITEM
DC99             167      ;
DC99 A5 15       168      ^2      lda INPUTFLG
DC9B D0 CC       169      bne <8
DC9D             170      ;
DC9D 4C 86 DB    171      jmp INPERR
DCA0             172      ;
DCA0 20 A3 D9    173      ^0      jsr DATSCAN
DCA3             174      ;
DCA3 C8          175      iny
DCA4             176      ;
DCA4 AA          177      tax
DCA5 D0 12       178      bne >3
DCA7             179      ;
DCA7 A2 2A       180      ldx #MESG04-MESGS ; Out of Data error
DCA9             181      ;
DCA9 C8          182      iny

```

```

DCAA          183 ;
DCAA B1 B8    184     lda (TXTPTR),Y
DCAC F0 5F    185     beq >8
DCAE          186 ;
DCAE C8       187     iny
DCAF          188 ;
DCAF B1 B8    189     lda (TXTPTR),Y
DCB1 85 7B    190     sta DATLIN
DCB3          191 ;
DCB3 C8       192     iny
DCB4          193 ;
DCB4 B1 B8    194     lda (TXTPTR),Y
DCB6 85 7C    195     sta DATLIN+1
DCB8          196 ;
DCB8 C8       197     iny
DCB9          198 ;
DCB9 B1 B8    199     ^3   lda (TXTPTR),Y
DCBB AA       200     tax
DCBC          201 ;
DCBC 20 98 D9 202     jsr BDATA2
DCBF          203 ;
DCBF E0 83    204     cpx #$80+BSDATA/2
DCC1 D0 DD    205     bne <0
DCC3          206 ;
DCC3 4C 2B DC 207     jmp INSTART
DCC6          208 ;
DCC6 A5 7F    209     ^4   lda SRCPTR
DCC8 A4 80    210     ldy SRCPTR+1
DCCA          211 ;
DCCA A6 15    212     ldx INPUTFLG
DCCC 10 03    213     bpl >5
DCCE          214 ;
DCCE 4C 53 D8 215     jmp SETDAPTR
DCD1          216 ;
DCD1 A0 00    217     ^5   ldy #ZERO
DCD3          218 ;
DCD3 B1 7F    219     lda (SRCPTR),Y
DCD5 D0 01    220     bne >6
DCD7          221 ;
DCD7 60       222     rts
DCD8          223 ;
DCD8 A9 DF    224     ^6   lda #MSG21
DCDA A0 DC    225     ldy /MSG21
DCDC          226 ;
DCDC 4C 3A DB 227     jmp STROUT
DCDF          228 ;
DCDF          229 ;
DCDF          230 MSG21:
DCDF          231 ;     asc `?`
DCDF 3E       232     asc `>`
DCE0 45 78 74 233     asc `Extra Ignored`
DCE3 72 61 20
DCE6 49 67 6E
DCE9 6F 72 65
DCEC 64
DCED 0D 00    234     byt RETURN&MSBCLR,ZERO
DCEF          235 ;
DCEF          236 MSG22:
DCEF          237 ;     asc `?`
DCEF 3E       238     asc `>`
DCF0 52 65 65 239     asc `Reenter`

```

```

DCF3 6E 74 65
DCF6 72
DCF7 0D 00      240      byt RETURN&MSBCLR,ZERO
DCF9      241      ;
DCF9      242      ;
DCF9 D0 04      243 BNEXT      bne BNEXT2
DCFB      244      ;
DCFB A0 00      245      ldy #ZERO
DCFD F0 03      246      beq >7      ; always taken
DCFF      247      ;
DCFF      248      ;
DCFF 20 E3 DF    249 BNEXT2      jsr PTRGET
DD02      250      ;
DD02 85 85      251 ^7      sta FORPNT
DD04 84 86      252      sty FORPNT+1
DD06      253      ;
DD06 20 62 D3    254      jsr GTFORPNT
DD09 F0 04      255      beq >9
DD0B      256      ;
DD0B A2 00      257      ldx #MSG01-MESGS      ; NEXT without FOR error
DD0D      258      ;
DD0D F0 69      259 ^8      beq >6      ; always taken, long branch
DD0F      260      ;
DD0F 9A      261 ^9      txs
DD10      262      ;
DD10 E8      263      inx
DD11 E8      264      inx
DD12 E8      265      inx
DD13 E8      266      inx
DD14      267      ;
DD14 8A      268      txa      ; location of FAC
DD15      269      ;
DD15 E8      270      inx
DD16 E8      271      inx
DD17 E8      272      inx
DD18 E8      273      inx
DD19 E8      274      inx
DD1A E8      275      inx
DD1B      276      ;
DD1B 86 60      277      stx DEST
DD1D      278      ;
DD1D A0 01      279      ldy /STACK
DD1F 20 F9 EA    280      jsr LOADFAC
DD22      281      ;
DD22 BA      282      tsx
DD23      283      ;
DD23 BD 09 01    284      lda STACK+9,X
DD26 85 A2      285      sta FACSIGN
DD28      286      ;
DD28 A5 85      287      lda FORPNT
DD2A A4 86      288      ldy FORPNT+1
DD2C      289      ;
DD2C 20 BE E7    290      jsr FADD
DD2F 20 27 EB    291      jsr COPYF2FR
DD32      292      ;
DD32 A0 01      293      ldy /STACK
DD34 20 B4 EB    294      jsr FPCOMP2
DD37      295      ;
DD37 BA      296      tsx
DD38      297      ;
DD38 38      298      sec

```

```

DD39          299 ;
DD39 FD 09 01 300      sbc STACK+9,X
DD3C F0 17    301      beq >2
DD3E          302 ;
DD3E BD 0F 01 303      lda STACK+15,X
DD41 85 75    304      sta CURLIN
DD43          305 ;
DD43 BD 10 01 306      lda STACK+16,X
DD46 85 76    307      sta CURLIN+1
DD48          308 ;
DD48 BD 12 01 309      lda STACK+18,X
DD4B 85 B8    310      sta TXTPTR
DD4D          311 ;
DD4D BD 11 01 312      lda STACK+17,X
DD50 85 B9    313      sta TXTPTR+1
DD52          314 ;
DD52 4C D2 D7 315      ^1      jmp NEWSTT
DD55          316 ;
DD55 8A       317      ^2      txa
DD56 69 11    318      adc #17
DD58 AA       319      tax
DD59          320 ;
DD59 9A       321      txs
DD5A          322 ;
DD5A 20 B7 00 323      jsr CHRGOT
DD5D          324 ;
DD5D C9 2C    325      cmp #', '
DD5F D0 F1    326      bne <1
DD61          327 ;
DD61 20 B1 00 328      jsr CHRGET
DD64 20 FF DC 329      jsr BNEXT2
DD67          330 ;
DD67          331 ;
DD67 20 7B DD 332      FRMNUM      jsr FRMEVAL
DD6A          333 ;
DD6A 18       334      CHKNUM      clc
DD6B          335 ;
DD6B 24 00    336      bit LOC0
DD6D          337      dfs !-1
DD6C          338 ;
DD6C 38       339      CHKSTR      sec
DD6D          340 ;
DD6D 24 11    341      CHKVAL      bit VALTYP
DD6F 30 03    342      bmi >4
DD71          343 ;
DD71 B0 03    344      bcs >5
DD73          345 ;
DD73 60       346      ^3      rts
DD74          347 ;
DD74 B0 FD    348      ^4      bcs <3
DD76          349 ;
DD76 A2 A5    350      ^5      ldx #MSG13-MESGS ; Type Mismatch error
DD78          351 ;
DD78 4C 12 D4 352      ^6      jmp FSCRN
DD7B          353 ;
DD7B          354 ;
DD7B A6 B8    355      FRMEVAL      ldx TXTPTR
DD7D D0 02    356      bne >7
DD7F          357 ;
DD7F C6 B9    358      dec TXTPTR+1
DD81          359 ;

```

```

DD81 C6 B8      360 ^7      dec TXTPTR
DD83           361 ;
DD83 A2 00      362      ldx #ZERO
DD85           363 ;
DD85 24 00      364      bit LOC0
DD87           365      dfs !-1
DD86           366 ;
DD86 48         367 FRMEVAL2 pha
DD87           368 ;
DD87 8A         369      txa
DD88 48         370      pha
DD89           371 ;
DD89 A9 01      372      lda #1
DD8B 20 D6 D3   373      jsr CKSTKSIZ
DD8E           374 ;
DD8E 20 63 DE   375      jsr FRMELMNT
DD91           376 ;
DD91 A9 00      377      lda #ZERO
DD93 85 89      378      sta CPRTYPE
DD95           379 ;
DD95 20 B7 00   380 FRMEVAL3 jsr CHRGOT
DD98           381 ;
DD98 38         382 ^8      sec
DD99           383 ;
DD99 E9 CF      384      sbc #TKGRTR
DD9B 90 17      385      bcc >9
DD9D           386 ;
DD9D C9 03      387      cmp #3
DD9F B0 13      388      bcs >9
DDA1           389 ;
DDA1 C9 01      390      cmp #1
DDA3           391 ;
DDA3 2A         392      rol
DDA4 49 01      393      eor #1
DDA6           394 ;
DDA6 45 89      395      eor CPRTYPE
DDA8 C5 89      396      cmp CPRTYPE
DDAA 90 61      397      bcc >7
DDAC           398 ;
DDAC 85 89      399      sta CPRTYPE
DDAE           400 ;
DDAE 20 B1 00   401      jsr CHRGET
DDB1           402 ;
DDB1 4C 98 DD   403      jmp <8
DDB4           404 ;
DDB4 A6 89      405 ^9      ldx CPRTYPE
DDB6 D0 2C      406      bne >4
DDB8           407 ;
DDB8 B0 7E      408      bcs NOTMATH
DDBA           409 ;
DDBA 69 07      410      adc #TKGRTR-TKPLUS
DDBC 90 7A      411      bcc NOTMATH
DDBE           412 ;
DDBE 65 11      413      adc VALTYP
DDC0 D0 03      414      bne >1
DDC2           415 ;
DDC2 4C 97 E5   416      jmp CAT2STR
DDC5           417 ;
DDC5 69 FF      418 ^1      adc #NEGONE
DDC7 85 5E      419      sta INDEX
DDC9           420 ;

```



```

DDC9 0A          421      asl
DDCA             422      ;
DDCA 65 5E       423      adc INDEX
DDCC A8          424      tay
DDCD             425      ;
DDCD 68          426      ^2      pla
DDCE D9 B4 D0    427      cmp TAGADDR,Y
DDD1 B0 6A       428      bcs >1
DDD3             429      ;
DDD3 20 6A DD    430      jsr CHKNUM
DDD6             431      ;
DDD6 48          432      ^3      pha
DDD7             433      ;
DDD7 20 FD DD    434      SAVOP   jsr FRMRECUR
DDDA             435      ;
DDDA 68          436      pla
DDDB             437      ;
DDDB A4 87       438      ldy LASTOP
DDDD 10 17       439      bpl >6
DDDF             440      ;
DDDF AA          441      tax
DDE0 F0 59       442      beq >9
DDE2             443      ;
DDE2 D0 62       444      bne >3          ; always taken
DDE4             445      ;
DDE4 46 11       446      ^4      lsr VALTYP
DDE6             447      ;
DDE6 8A          448      txa
DDE7 2A          449      rol
DDE8             450      ;
DDE8 A6 B8       451      ldx TXTPTR
DDEA D0 02       452      bne >5
DDEC             453      ;
DDEC C6 B9       454      dec TXTPTR+1
DDEE             455      ;
DDEE C6 B8       456      ^5      dec TXTPTR
DDF0             457      ;
DDF0 A0 1B       458      ldy #OTLESS-TAGADDR
DDF2             459      ;
DDF2 85 89       460      sta CPRTYPE
DDF4             461      ;
DDF4 D0 D7       462      bne <2
DDF6             463      ;
DDF6 D9 B4 D0    464      ^6      cmp TAGADDR,Y
DDF9 90 DB       465      bcc <3
DDFB             466      ;
DDFB B0 49       467      bcs >3          ; always taken
DDFD             468      ;
DDFD             469      ;
DDFD B9 B6 D0    470      FRMRECUR lda TAGADDR+2,Y
DE00 48          471      pha
DE01             472      ;
DE01 B9 B5 D0    473      lda TAGADDR+1,Y
DE04 48          474      pha
DE05             475      ;
DE05 20 10 DE    476      jsr FRMSTAK
DE08             477      ;
DE08 A5 89       478      lda CPRTYPE
DE0A             479      ;
DE0A 4C 86 DD    480      jmp FRMEVAL2
DE0D             481      ;

```

```

DE0D          482 ;
DE0D 4C C9 DE 483 ^7      jmp DOMESG02
DE10          484 ;
DE10          485 ;
DE10 A5 A2    486 FRMSTAK lda FACSIGN
DE12          487 ;
DE12 BE B4 D0 488      ldx TAGADDR,Y
DE15          489 ;
DE15          490 ;
DE15          491 ; Pull return address and increment. Should not assume not
DE15          492 ; an end of page address. Extra bytes come from OEQUAL.
DE15          493 ;
DE15 A8       494 FRMSTAK2 tay          ; FACSIGN
DE16          495 ;
DE16 18       496      clc
DE17          497 ;
DE17 68       498      pla
DE18 69 01    499      adc #1
DE1A 85 5E    500      sta INDEX
DE1C          501 ;
DE1C 68       502      pla
DE1D 69 00    503      adc #ZERO
DE1F 85 5F    504      sta INDEX+1
DE21          505 ;
DE21 98       506      tya
DE22 48       507      pha
DE23          508 ;
DE23          509 ;
DE23 20 72 EB 510 FRMSTAK3 jsr RNDUP
DE26          511 ;
DE26 A5 A1    512      lda FACMANT+3
DE28 48       513      pha
DE29          514 ;
DE29 A5 A0    515      lda FACMANT+2
DE2B 48       516      pha
DE2C          517 ;
DE2C A5 9F    518      lda FACMANT+1
DE2E 48       519      pha
DE2F          520 ;
DE2F A5 9E    521      lda FACMANT
DE31 48       522      pha
DE32          523 ;
DE32 A5 9D    524      lda FACEXP
DE34 48       525      pha
DE35          526 ;
DE35 6C 5E 00 527      jmp (INDEX)
DE38          528 ;
DE38          529 ;
DE38 A0 FF    530 NOTMATH ldy #NEGONE
DE3A          531 ;
DE3A 68       532      pla
DE3B          533 ;
DE3B F0 23    534 ^9      beq >4
DE3D          535 ;
DE3D C9 64    536 ^1      cmp #OTVLESS
DE3F F0 03    537      beq >2
DE41          538 ;
DE41 20 6A DD 539      jsr CHKNUM
DE44          540 ;
DE44 84 87    541 ^2      sty LASTOP
DE46          542 ;

```

```

DE46 68          543  ^3      pla
DE47 4A          544          lsr
DE48 85 16       545          sta CPRMASK
DE4A             546  ;
DE4A 68          547          pla
DE4B 85 A5       548          sta ARGEXP
DE4D             549  ;
DE4D 68          550          pla
DE4E 85 A6       551          sta ARGMANT
DE50             552  ;
DE50 68          553          pla
DE51 85 A7       554          sta ARGMANT+1
DE53             555  ;
DE53 68          556          pla
DE54 85 A8       557          sta ARGMANT+2
DE56             558  ;
DE56 68          559          pla
DE57 85 A9       560          sta ARGMANT+3
DE59             561  ;
DE59 68          562          pla
DE5A 85 AA       563          sta ARGSIGN
DE5C             564  ;
DE5C 45 A2       565          eor FACSIGN
DE5E 85 AB       566          sta XORSIGN
DE60             567  ;
DE60 A5 9D       568  ^4      lda FACEXP
DE62             569  ;
DE62 60          570          rts
DE63             571  ;
DE63             572  ;
DE63 A9 00       573  FRMELMNT lda #ZERO
DE65 85 11       574          sta VALTYP
DE67             575  ;
DE67 20 B1 00    576  ^5      jsr CHRGET
DE6A B0 03       577          bcs >7
DE6C             578  ;
DE6C 4C 4A EC    579  ^6      jmp GETINT
DE6F             580  ;
DE6F 20 7D E0    581  ^7      jsr CHKASCII
DE72 B0 61       582          bcs GETIVAL
DE74             583  ;
DE74 C9 2E       584          cmp #'.'
DE76 F0 F4       585          beq <6
DE78             586  ;
DE78 C9 C9       587          cmp #TKMINUS
DE7A F0 52       588          beq >4
DE7C             589  ;
DE7C C9 C8       590          cmp #TKPLUS
DE7E F0 E7       591          beq <5
DE80             592  ;
DE80 C9 22       593          cmp #'"'
DE82 D0 0F       594          bne >9
DE84             595  ;
DE84             596  ;
DE84 A5 B8       597  STRTXT   lda TXTPTR
DE86 A4 B9       598          ldy TXTPTR+1
DE88             599  ;
DE88 69 00       600          adc #ZERO
DE8A 90 01       601          bcc >8
DE8C             602  ;
DE8C C8          603          iny

```

```

DE8D      604 ;
DE8D 20 E7 E3 605 ^8      jsr STRLIT
DE90      606 ;
DE90 4C 3D E7 607      jmp STRCOPY
DE93      608 ;
DE93 C9 C6     609 ^9      cmp #TKNOT
DE95 D0 10     610      bne >2
DE97      611 ;
DE97 A0 18     612      ldy #24
DE99 D0 35     613      bne >5
DE9B      614 ;
DE9B      615 ;
DE9B A5 9D     616 OEQUAL    lda DSCTMP
DE9D D0 03     617      bne >1
DE9F      618 ;
DE9F A0 01     619      ldy #1
DEA1      620 ;
DEA1 2C 00 00  621      bit *-*
DEA4      622      dfs !-2
DEA2      623 ;
DEA2 A0 00     624 ^1      ldy #ZERO
DEA4      625 ;
DEA4 4C 01 E3  626      jmp SNGFLT
DEA7      627 ;
DEA7 C9 C2     628 ^2      cmp #TKFN
DEA9 D0 03     629      bne >3
DEAB      630 ;
DEAB 4C 54 E3  631      jmp CALLFNC
DEAE      632 ;
DEAE C9 D2     633 ^3      cmp #TKSGN
DEB0 B0 5A     634      bcs >9          ; accelerate code
DEB2      635 ;
DEB2      636 ;
DEB2      637 ; Evaluate "(expression)".
DEB2      638 ;
DEB2 20 BB DE  639 PARENCHK  jsr CHKOPNP
DEB5 20 7B DD  640      jsr FRMEVAL
DEB8      641 ;
DEB8 A9 29     642 CHKCLSP   lda #' ) '
DEBA      643 ;
DEBA 2C 00 00  644      bit *-*
DEBD      645      dfs !-2
DEBB      646 ;
DEBB A9 28     647 CHKOPNP   lda #' ( '
DEBD      648 ;
DEBD 2C 00 00  649      bit *-*
DEC0      650      dfs !-2
DEBE      651 ;
DEBE A9 2C     652 CHKCOM    lda #' , '
DEC0      653 ;
DEC0      654 ;
DEC0 A0 00     655 SYNTAXCHK ldy #ZERO
DEC2      656 ;
DEC2 D1 B8     657      cmp (TXTPTR),Y
DEC4 D0 03     658      bne DOMESG02
DEC6      659 ;
DEC6 4C B1 00  660      jmp CHRGET
DEC9      661 ;
DEC9 A2 10     662 DOMESG02 ldx #MSG02-MESGS ; Syntax error
DECB      663 ;
DECB 4C 12 D4  664      jmp FSCRN

```

```

DECE          665 ;
DECE A0 15    666 ^4      ldy #$15
DED0          667 ;
DED0 68      668 ^5      pla
DED1 68      669      pla
DED2          670 ;
DED2 4C D7 DD 671      jmp SAVOP
DED5          672 ;
DED5          673 ;
DED5 20 E3 DF 674 GETIVAL jsr PTRGET
DED8          675 ;
DED8 85 A0    676      sta VARPTR
DEDA 84 A1    677      sty VARPTR+1
DEDC          678 ;
DEDC A6 11    679      ldx VALTYP
DEDE F0 05    680      beq >6
DEE0          681 ;
DEE0 A2 00    682      ldx #ZERO
DEE2 86 AC    683      stx FACGUARD
DEE4          684 ;
DEE4 60      685      rts
DEE5          686 ;
DEE5          687 ;
DEE5          688 ; Get value from VARPTR into A/Y-reg to put into FACMANT.
DEE5          689 ;
DEE5 A6 12    690 ^6      ldx VALTYP+1
DEE7 10 0D    691      bpl >7
DEE9          692 ;
DEE9 A0 00    693      ldy #ZERO
DEEB          694 ;
DEEB B1 A0    695      lda (VARPTR),Y
DEED AA      696      tax
DEEE          697 ;
DEEE C8      698      iny
DEEF          699 ;
DEEF B1 A0    700      lda (VARPTR),Y
DEF1 A8      701      tay
DEF2          702 ;
DEF2 8A      703      txa
DEF3          704 ;
DEF3 4C F2 E2 705      jmp GIVAYFP
DEF6          706 ;
DEF6 4C F9 EA 707 ^7      jmp LOADFAC
DEF9          708 ;
DEF9 20 B1 00 709 ^8      jsr CHRGET
DEFC 20 EC F1 710      jsr PLOTFNS
DEFF          711 ;
DEFF 8A      712      txa
DF00          713 ;
DF00 A4 F0    714      ldy FIRST
DF02          715 ;
DF02 20 71 F8 716      jsr SCRN
DF05          717 ;
DF05 A8      718      tay
DF06          719 ;
DF06 20 01 E3 720      jsr SNGFLT
DF09          721 ;
DF09 4C B8 DE 722      jmp CHKCLSP
DF0C          723 ;
DF0C          724 ;
DF0C          725 ; This code has been rewritten in order to add the LN

```

```

DF0C          726 ; statement. This requires the new TOKEN variable.
DF0C          727 ;
DF0C C9 D7    728 ^9      cmp #TKSCRN
DF0E F0 E9    729          beq <8
DF10          730 ;
DF10 AA       731          tax
DF11          732 ;
DF11 0A       733          asl
DF12 85 CD    734          sta TOKEN
DF14          735 ;
DF14 20 B1 00 736          jsr CHRGET
DF17          737 ;
DF17 E0 E8    738          cpx #$92+FS2LEFT/2 ; is < LEFT$?
DF19 90 1E    739          bcc >1
DF1B          740 ;
DF1B E0 EB    741          cpx #$92+FS3LN/2 ; is >= LN?
DF1D B0 1A    742          bcs >1
DF1F          743 ;
DF1F 20 BB DE 744          jsr CHKOPNP
DF22 20 7B DD 745          jsr FRMEVAL
DF25 20 BE DE 746          jsr CHKCOM
DF28 20 6C DD 747          jsr CHKSTR
DF2B          748 ;
DF2B A5 A1    749          lda VARPTR+1
DF2D 48       750          pha
DF2E          751 ;
DF2E A5 A0    752          lda VARPTR
DF30 48       753          pha
DF31          754 ;
DF31 20 F8 E6 755          jsr GETBYT
DF34          756 ;
DF34 8A       757          txa
DF35 48       758          pha
DF36          759 ;
DF36 4C 3C DF 760          jmp >2
DF39          761 ;
DF39 20 B2 DE 762 ^1      jsr PARENCHK
DF3C          763 ;
DF3C          764 ;
DF3C          765 ; Index into ( FN1ADDR - TOKEN.SGN - TOKEN.SGN + PAGESIZE )
DF3C          766 ;
DF3C A4 CD    767 ^2      ldy TOKEN
DF3E          768 ;
DF3E B9 DC CF 769          lda PAGESIZE+FN1ADDR-2*NEGONE&$92+FS1SGN/2,Y
DF41 85 91    770          sta JMPADRS+1
DF43          771 ;
DF43 B9 DD CF 772          lda PAGESIZE+1+FN1ADDR-2*NEGONE&$92+FS1SGN/2,Y
DF46 85 92    773          sta JMPADRS+2
DF48          774 ;
DF48 20 90 00 775          jsr JMPADRS
DF4B          776 ;
DF4B 4C 6A DD 777          jmp CHKNUM
DF4E          778 ;
DF4E          779 ;
DF4E          780          dfs 1,ZERO ; 1 byte
DF4F          781 ;
DF4F          782 ;
DF4F A5 A5    783 OOR      lda ARGEXP
DF51 05 9D    784          ora FACEXP
DF53 D0 0B    785          bne >2
DF55          786 ;

```

```

DF55 A5 A5      787 OAND      lda ARGEXP
DF57 F0 04      788          beq >1
DF59            789 ;
DF59 A5 9D      790          lda FACEXP
DF5B D0 03      791          bne >2
DF5D            792 ;
DF5D A0 00      793 ^1      ldy #ZERO
DF5F            794 ;
DF5F 2C 00 00   795          bit *-*
DF62            796          dfs !-2
DF60            797 ;
DF60 A0 01      798 ^2      ldy #1
DF62            799 ;
DF62 4C 01 E3   800          jmp SNGFLT
DF65            801 ;
DF65            802 ;
DF65 20 6D DD   803 OLT      jsr CHKVAL
DF68 B0 13      804          bcs >1
DF6A            805 ;
DF6A A5 AA      806          lda ARGSIGN
DF6C 09 7F      807          ora #MSBCLR
DF6E            808 ;
DF6E 25 A6      809          and ARGMANT
DF70 85 A6      810          sta ARGMANT
DF72            811 ;
DF72 A9 A5      812          lda #ARGEXP
DF74 A0 00      813          ldy /ARGEXP
DF76            814 ;
DF76 20 B2 EB   815          jsr FPCOMP
DF79            816 ;
DF79 AA         817          tax
DF7A            818 ;
DF7A 4C B0 DF   819          jmp >0
DF7D            820 ;
DF7D A9 00      821 ^1      lda #ZERO
DF7F 85 11      822          sta VALTYP
DF81            823 ;
DF81 C6 89      824          dec CPRTYPE
DF83            825 ;
DF83 20 00 E6   826          jsr FRESTR2
DF86            827 ;
DF86 85 9D      828          sta FACEXP
DF88 86 9E      829          stx FACMANT
DF8A 84 9F      830          sty FACMANT+1
DF8C            831 ;
DF8C A5 A8      832          lda ARGMANT+2
DF8E A4 A9      833          ldy ARGMANT+3
DF90            834 ;
DF90 20 04 E6   835          jsr FRETMP
DF93            836 ;
DF93 86 A8      837          stx ARGMANT+2
DF95 84 A9      838          sty ARGMANT+3
DF97            839 ;
DF97 AA         840          tax
DF98            841 ;
DF98 38         842          sec
DF99            843 ;
DF99 E5 9D      844          sbc FACEXP
DF9B F0 08      845          beq >2
DF9D            846 ;
DF9D A9 01      847          lda #1

```

```

DF9F          848 ;
DF9F 90 04    849      bcc >2
DFA1          850 ;
DFA1 A6 9D    851      ldx FACEXP
DFA3          852 ;
DFA3 A9 FF    853      lda #NEGONE
DFA5          854 ;
DFA5 85 A2    855 ^2    sta FACSIGN
DFA7          856 ;
DFA7 A0 FF    857      ldy #NEGONE
DFA9          858 ;
DFA9 E8       859      inx
DFAA          860 ;
DFAA C8       861 ^3    iny
DFAB          862 ;
DFAB CA       863      dex
DFAC D0 07    864      bne >4
DFAE          865 ;
DFAE A6 A2    866      ldx FACSIGN
DFB0          867 ;
DFB0 30 0F    868 ^0    bmi >5
DFB2          869 ;
DFB2 18       870      clc
DFB3 90 0C    871      bcc >5          ; always taken
DFB5          872 ;
DFB5 B1 A8    873 ^4    lda (ARGMANT+2),Y
DFB7 D1 9E    874      cmp (FACMANT),Y
DFB9 F0 EF    875      beq <3
DFBB          876 ;
DFBB A2 FF    877      ldx #NEGONE
DFBD          878 ;
DFBD B0 02    879      bcs >5
DFBF          880 ;
DFBF A2 01    881      ldx #1
DFC1          882 ;
DFC1 E8       883 ^5    inx
DFC2          884 ;
DFC2 8A       885      txa
DFC3 2A       886      rol
DFC4          887 ;
DFC4 25 16    888      and CPRMASK
DFC6 F0 02    889      beq >6
DFC8          890 ;
DFC8 A9 01    891      lda #1
DFCA          892 ;
DFCA 4C 93 EB 893 ^6    jmp FLOAT
DFCD          894 ;
DFCD          895 ;
DFCD 20 FB E6 896 FPD L  jsr CONVINT
DFD0 20 1E FB 897      jsr PREAD
DFD3          898 ;
DFD3 4C 01 E3 899      jmp SNGFLT
DFD6          900 ;
DFD6          901 ;
DFD6 20 BE DE 902 ^7    jsr CHKCOM
DFD9          903 ;
DFD9 AA       904 BDIM  tax
DFDA          905 ;
DFDA 20 E8 DF 906      jsr PTRGET2
DFDD          907 ;
DFDD 20 B7 00 908      jsr CHRGOT

```



```

DFE0 D0 F4      909      bne <7
DFE2            910      ;
DFE2 60         911      rts
DFE3            912      ;
DFE3            913      ;
DFE3            914      ; PTRGET is controlled by DIMFLG and SUBFLG.
DFE3            915      ; DIMFLG - nonzero is call from "DIM", else zero.
DFE3            916      ; SUBFLG - 0x00 initial value
DFE3            917      ; 0x40 if called from GETARYPT (removed)
DFE3            918      ; 0x80 if called from DEF FN
DFE3            919      ; 0xC1:DA if called from FN
DFE3            920      ;
DFE3 A2 00      921 PTRGET   ldx #ZERO
DFE5            922      ;
DFE5 20 B7 00   923      jsr CHRGOT
DFE8            924      ;
DFE8 86 10      925 PTRGET2  stx DIMFLG
DFEA            926      ;
DFEA 85 81      927 PTRGET3  sta VARNAM
DFEC            928      ;
DFEC 20 B7 00   929      jsr CHRGOT
DFEF            930      ;
DFEF 20 7D E0   931      jsr CHKASCI
DFF2 B0 03      932      bcs >8          ; not digit
DFF4            933      ;
DFF4 4C C9 DE   934 GOMESG02 jmp DOMESG02
DFF7            935      ;
DFF7 A2 00      936 ^8      ldx #ZERO
DFF9 86 11      937      stx VALTYP
DFFB 86 12      938      stx VALTYP+1
DFFD            939      ;
DFFD 4C 07 E0   940      jmp PTRCONT
E000            941      ;
E000            942      ;

```

BSAVE D0ROM,D1,A\$1000,B,L\$1000

```

E000            943      usr D0ROM,D1
E000            944      ;
E000            945      ;
E000            946      icl "E0.L,D2"

```

LLOAD E0.L,D2,A\$4000

```

E000          1          ttl "ROM Source Code, E0.L"
E000          2          ;
E000          3          ;
E000          4          ; E0.L
E000          5          ;
E000          6          ;
E000          7          obj PAGE10
E000          8          usr
E000          9          ;
E000         10          ;
E000         11          ; This is the DOS default COLDADR.
E000         12          ;
E000 4C 28 F1  13 BASIC      jmp COLDSTRT
E003         14          ;
E003         15          ;
E003 4C 3C D4  16 BASIC2     jmp RESTART
E006         17          ;
E006         18          ;
E006         19          ; Not sure what purpose this servers.
E006         20          ;
E006 C4        21          hex C4
E007         22          ;
E007         23          ;
E007 20 B1 00  24 PTRCONT    jsr CHRGET
E00A 90 05     25            bcc >1
E00C         26          ;
E00C 20 7D E0  27            jsr CHKASCI
E00F 90 0B     28            bcc >3
E011         29          ;
E011 AA       30 ^1         tax
E012         31          ;
E012 20 B1 00  32 ^2         jsr CHRGET
E015 90 FB     33            bcc <2
E017         34          ;
E017 20 7D E0  35            jsr CHKASCI
E01A B0 F6     36            bcs <2
E01C         37          ;
E01C C9 24     38 ^3         cmp #'$`
E01E D0 06     39            bne >4
E020         40          ;
E020 A9 FF     41            lda #NEGONE
E022 85 11     42            sta VALTYP
E024 D0 10     43            bne >5                      ; always taken
E026         44          ;
E026 C9 25     45 ^4         cmp #$25
E028 D0 13     46            bne >6
E02A         47          ;
E02A A5 14     48            lda SUBFLG
E02C 30 C6     49            bmi GOMESG02
E02E         50          ;
E02E A9 80     51            lda #$80
E030 85 12     52            sta VALTYP+1
E032         53          ;
E032 05 81     54            ora VARNAM
E034 85 81     55            sta VARNAM
E036         56          ;
E036 8A       57 ^5         txa
E037 09 80     58            ora #$80
E039 AA       59            tax
E03A         60          ;

```

```

E03A 20 B1 00      61      jsr CHRGET
E03D              62      ;
E03D 86 82        63      ^6      stx VARNAM+1
E03F              64      ;
E03F 38           65      sec
E040              66      ;
E040 05 14        67      ora SUBFLG
E042              68      ;
E042 E9 28        69      sbc #$28
E044 D0 03        70      bne >8
E046              71      ;
E046 4C 26 E1     72      ^7      jmp ARRAY
E049              73      ;
E049 24 14        74      ^8      bit SUBFLG
E04B 30 02        75      bmi >9
E04D              76      ;
E04D 70 F7        77      bvs <7
E04F              78      ;
E04F A9 00        79      ^9      lda #ZERO
E051 85 14        80      sta SUBFLG
E053              81      ;
E053 A5 69        82      lda VARTAB
E055 A6 6A        83      ldx VARTAB+1
E057              84      ;
E057 A0 00        85      ldy #ZERO
E059              86      ;
E059 86 9C        87      ^1      stx LOWTR+1
E05B              88      ;
E05B 85 9B        89      ^2      sta LOWTR
E05D              90      ;
E05D E4 6C        91      cpx ARYTAB+1
E05F D0 04        92      bne >3
E061              93      ;
E061 C5 6B        94      cmp ARYTAB
E063 F0 21        95      beq >2
E065              96      ;
E065 A5 81        97      ^3      lda VARNAM
E067 D1 9B        98      cmp (LOWTR),Y
E069 D0 08        99      bne >4
E06B             100      ;
E06B C8           101      iny
E06C             102      ;
E06C A5 82        103      lda VARNAM+1
E06E D1 9B        104      cmp (LOWTR),Y
E070 F0 6B        105      beq >5
E072             106      ;
E072 88           107      dey
E073             108      ;
E073 18           109      ^4      clc
E074             110      ;
E074 A5 9B        111      lda LOWTR
E076 69 07        112      adc #SVARLEN      ; next descriptor
E078 90 E1        113      bcc <2
E07A             114      ;
E07A E8           115      inx
E07B D0 DC        116      bne <1      ; always taken
E07D             117      ;
E07D             118      ;
E07D             119      ; If A-reg contains an ASCII letter A-Z, set C-flag,
E07D             120      ; otherwise clear C-flag. This code is faster and makes
E07D             121      ; room for five bytes for FP8000.

```

```

E07D          122 ;
E07D C9 5B    123 CHKASCII  cmp #'Z'+1
E07F B0 03    124          bcs >1
E081          125 ;
E081 C9 41    126          cmp #'A'
E083          127 ;
E083 60       128          rts
E084          129 ;
E084 18       130 ^1      clc
E085          131 ;
E085 60       132          rts
E086          133 ;
E086          134 ;
E086          135 ; Variable not found.
E086          136 ;
E086 68       137 ^2      pla
E087 48       138          pha
E088          139 ;
E088 C9 D7    140          cmp #GETIVAL+2
E08A D0 0F    141          bne >3
E08C          142 ;
E08C BA       143          tsx
E08D          144 ;
E08D BD 02 01 145          lda STACK+2,X
E090 C9 DE    146          cmp /GETIVAL+2
E092 D0 07    147          bne >3
E094          148 ;
E094 A9 99    149          lda #IVALZERO
E096 A0 E0    150          ldy /IVALZERO
E098          151 ;
E098 60       152          rts
E099          153 ;
E099          154 ;
E099 00 00    155 IVALZERO hex 0000
E09B          156 ;
E09B          157 ;
E09B A5 6B    158 ^3      lda ARYTAB
E09D A4 6C    159          ldy ARYTAB+1
E09F          160 ;
E09F 85 9B    161          sta LOWTR
E0A1 84 9C    162          sty LOWTR+1
E0A3          163 ;
E0A3 A5 6D    164          lda STREND
E0A5 A4 6E    165          ldy STREND+1
E0A7          166 ;
E0A7 85 96    167          sta HIGHTR
E0A9 84 97    168          sty HIGHTR+1
E0AB          169 ;
E0AB 18       170          clc
E0AC          171 ;
E0AC 69 07    172          adc #SVARLEN
E0AE 90 01    173          bcc >4
E0B0          174 ;
E0B0 C8       175          iny
E0B1          176 ;
E0B1 85 94    177 ^4      sta HIGHDS
E0B3 84 95    178          sty HIGHDS+1
E0B5          179 ;
E0B5 20 93 D3 180          jsr BLTU
E0B8          181 ;
E0B8 A5 94    182          lda HIGHDS

```

```

E0BA A4 95      183      ldy HIGHDS+1
E0BC           184      ;
E0BC C8        185      iny
E0BD           186      ;
E0BD 85 6B     187      sta ARYTAB
E0BF 84 6C     188      sty ARYTAB+1
E0C1           189      ;
E0C1 A0 00     190      ldy #ZERO
E0C3           191      ;
E0C3 A5 81     192      lda VARNAM
E0C5 91 9B     193      sta (LOWTR),Y
E0C7           194      ;
E0C7 C8        195      iny
E0C8           196      ;
E0C8 A5 82     197      lda VARNAM+1
E0CA 91 9B     198      sta (LOWTR),Y
E0CC           199      ;
E0CC C8        200      iny
E0CD           201      ;
E0CD A9 00     202      lda #ZERO
E0CF           203      ;
E0CF 91 9B     204      sta (LOWTR),Y
E0D1           205      ;
E0D1 C8        206      iny
E0D2           207      ;
E0D2 91 9B     208      sta (LOWTR),Y
E0D4           209      ;
E0D4 C8        210      iny
E0D5           211      ;
E0D5 91 9B     212      sta (LOWTR),Y
E0D7           213      ;
E0D7 C8        214      iny
E0D8           215      ;
E0D8 91 9B     216      sta (LOWTR),Y
E0DA           217      ;
E0DA C8        218      iny
E0DB           219      ;
E0DB 91 9B     220      sta (LOWTR),Y
E0DD           221      ;
E0DD A4 9C     222      ^5 ldy LOWTR+1
E0DF           223      ;
E0DF 18        224      clc
E0E0           225      ;
E0E0 A5 9B     226      lda LOWTR
E0E2 69 02     227      adc #IVARLEN
E0E4 90 01     228      bcc >6
E0E6           229      ;
E0E6 C8        230      iny
E0E7           231      ;
E0E7 85 83     232      ^6 sta VARPNT
E0E9 84 84     233      sty VARPNT+1
E0EB           234      ;
E0EB 60        235      rts
E0EC           236      ;
E0EC           237      ;
E0EC           238      ; Point to first array value.
E0EC           239      ;
E0EC A4 9C     240      PNTARVAL ldy LOWTR+1
E0EE           241      ;
E0EE A5 0F     242      lda NUMDIM
E0F0 0A        243      asl

```

```

E0F1 69 05      244      adc #AHDRLEN
E0F3            245      ;
E0F3 65 9B      246      adc LOWTR
E0F5 90 01      247      bcc >7
E0F7            248      ;
E0F7 C8         249      iny
E0F8            250      ;
E0F8 85 94      251      ^7      sta HIGHDS
E0FA 84 95      252      sty HIGHDS+1
E0FC            253      ;
E0FC 60         254      rts
E0FD            255      ;
E0FD            256      ;
E0FD            257      ; FP8000 is meant to be integer 32768; CHKASCII makes room.
E0FD            258      ;
E0FD 90 80 00   259      FP8000      hex 9080000000      ; FP -32768
E100 00 00
E102            260      ;
E102            261      ;
E102 20 B1 00   262      MAKINT      jsr CHRGET
E105 20 67 DD   263      jsr FRMNUM
E108            264      ;
E108            265      ;
E108 A5 A2      266      AYPOSINT      lda FACSIGN
E10A 30 0D      267      bmi >1
E10C            268      ;
E10C            269      ;
E10C A5 9D      270      AYINT      lda FACEXP
E10E C9 90      271      cmp #$90
E110 90 09      272      bcc >2
E112            273      ;
E112 A9 FD      274      lda #FP8000
E114 A0 E0      275      ldy /FP8000
E116            276      ;
E116 20 B2 EB   277      jsr FPCOMP
E119            278      ;
E119 D0 7E      279      ^1      bne IQ.ERR
E11B            280      ;
E11B 4C F2 EB   281      ^2      jmp FP2INT
E11E            282      ;
E11E            283      ;
E11E            284      dfs 8,ZERO      ; 8 bytes
E126            285      ;
E126            286      ;
E126            287      ; Rewrote this code in order to accelerate this roution.
E126            288      ;
E126 A5 14      289      ARRAY      lda SUBFLG
E128 D0 3F      290      bne >4
E12A            291      ;
E12A A5 10      292      lda DIMFLG
E12C 05 12      293      ora VALTYP+1
E12E 48         294      pha
E12F            295      ;
E12F A5 11      296      lda VALTYP
E131 48         297      pha
E132            298      ;
E132 A9 00      299      lda #ZERO
E134 85 0F      300      sta NUMDIM
E136            301      ;
E136 A5 82      302      ^3      lda VARNAM+1
E138 48         303      pha

```

```

E139          304 ;
E139 A5 81    305     lda VARNAM
E13B 48       306     pha
E13C          307 ;
E13C 20 02 E1 308     jsr MAKINT
E13F          309 ;
E13F 68       310     pla
E140 85 81    311     sta VARNAM
E142          312 ;
E142 68       313     pla
E143 85 82    314     sta VARNAM+1
E145          315 ;
E145 68       316     pla
E146 A8       317     tay
E147          318 ;
E147 68       319     pla
E148 AA       320     tax
E149          321 ;
E149 A5 A0    322     lda VARPTR
E14B 48       323     pha
E14C          324 ;
E14C A5 A1    325     lda VARPTR+1
E14E 48       326     pha
E14F          327 ;
E14F 8A       328     txa
E150 48       329     pha
E151          330 ;
E151 98       331     tya
E152 48       332     pha
E153          333 ;
E153 E6 0F    334     inc NUMDIM
E155          335 ;
E155 20 B7 00 336     jsr CHRGOT
E158          337 ;
E158 C9 2C    338     cmp #', '
E15A F0 DA    339     beq <3
E15C          340 ;
E15C 20 B8 DE 341     jsr CHKCLSP
E15F          342 ;
E15F 68       343     pla
E160 85 11    344     sta VALTYP
E162          345 ;
E162 68       346     pla
E163 85 12    347     sta VALTYP+1
E165          348 ;
E165 29 7F    349     and #MSBCLR
E167 85 10    350     sta DIMFLG
E169          351 ;
E169          352 ;
E169          353 ; Search Array Table for this Array Name.
E169          354 ;
E169 A6 6B    355     ^4     ldx ARYTAB
E16B A5 6C    356     lda ARYTAB+1
E16D          357 ;
E16D 86 9B    358     ^5     stx LOWTR
E16F 85 9C    359     sta LOWTR+1
E171          360 ;
E171 C5 6E    361     cmp STREND+1
E173 D0 04    362     bne >6
E175          363 ;
E175 E4 6D    364     cpx STREND

```

```

E177 F0 3F      365      beq >1
E179           366      ;
E179 A0 00      367      ^6      ldy #ZERO
E17B           368      ;
E17B B1 9B      369      lda (LOWTR),Y
E17D           370      ;
E17D C8         371      iny
E17E           372      ;
E17E C5 81      373      cmp VARNAM
E180 D0 06      374      bne >7
E182           375      ;
E182 B1 9B      376      lda (LOWTR),Y
E184 C5 82      377      cmp VARNAM+1
E186 F0 16      378      beq RA.ERR
E188           379      ;
E188 C8         380      ^7      iny
E189           381      ;
E189 18         382      clc
E18A           383      ;
E18A B1 9B      384      lda (LOWTR),Y
E18C 65 9B      385      adc LOWTR
E18E AA         386      tax
E18F           387      ;
E18F C8         388      iny
E190           389      ;
E190 B1 9B      390      lda (LOWTR),Y
E192 65 9C      391      adc LOWTR+1
E194 90 D7      392      bcc <5
E196           393      ;
E196           394      ;
E196 A2 6D      395      BS.ERR      ldx #MSG09-MESGS      ; Bad Subscript error
E198           396      ;
E198 2C 00 00   397      bit *-*
E19B           398      dfs !-2
E199           399      ;
E199 A2 35      400      IQ.ERR      ldx #MSG05-MESGS      ; Illegal Quantity error
E19B           401      ;
E19B 4C 12 D4   402      ^8      jmp FSCRN
E19E           403      ;
E19E           404      ;
E19E A2 7A      405      RA.ERR      ldx #MSG10-MESGS      ; ReDIM'd Array error
E1A0           406      ;
E1A0 A5 10      407      lda DIMFLG
E1A2 D0 F7      408      bne <8
E1A4           409      ;
E1A4 A5 14      410      lda SUBFLG
E1A6 F0 02      411      beq >9
E1A8           412      ;
E1A8 38         413      sec
E1A9           414      ;
E1A9 60         415      rts
E1AA           416      ;
E1AA 20 EC E0   417      ^9      jsr PNTARVAL
E1AD           418      ;
E1AD A5 0F      419      lda NUMDIM
E1AF           420      ;
E1AF A0 04      421      ldy #4
E1B1           422      ;
E1B1 D1 9B      423      cmp (LOWTR),Y
E1B3 D0 E1      424      bne BS.ERR
E1B5           425      ;

```



```

E1B5 4C 4B E2      426      jmp FINDELE
E1B8              427      ;
E1B8 A5 14        428      ^1      lda SUBFLG
E1BA F0 05        429      beq >2
E1BC              430      ;
E1BC A2 2A        431      ldx #MSG04-MESGS      ; Out of Data error
E1BE              432      ;
E1BE 4C 12 D4     433      jmp FSCRN
E1C1              434      ;
E1C1 20 EC E0     435      ^2      jsr PNTARVAL
E1C4 20 E3 D3     436      jsr CKSTRSIZ
E1C7              437      ;
E1C7 A9 00        438      lda #ZERO
E1C9 85 AE        439      sta STRING2+1
E1CB A8           440      tay
E1CC              441      ;
E1CC A2 05        442      ldx #5
E1CE              443      ;
E1CE A5 81        444      lda VARNAM
E1D0 91 9B        445      sta (LOWTR),Y
E1D2 10 01        446      bpl >3
E1D4              447      ;
E1D4 CA           448      dex
E1D5              449      ;
E1D5 C8           450      ^3      iny
E1D6              451      ;
E1D6 A5 82        452      lda VARNAM+1
E1D8 91 9B        453      sta (LOWTR),Y
E1DA 10 02        454      bpl >4
E1DC              455      ;
E1DC CA           456      dex
E1DD CA           457      dex
E1DE              458      ;
E1DE 86 AD        459      ^4      stx STRING2
E1E0              460      ;
E1E0 C8           461      iny
E1E1 C8           462      iny
E1E2 C8           463      iny
E1E3              464      ;
E1E3 A5 0F        465      lda NUMDIM
E1E5 91 9B        466      sta (LOWTR),Y
E1E7              467      ;
E1E7 A2 0B        468      ^5      ldx #DFLTDIM
E1E9 A9 00        469      lda #ZERO
E1EB              470      ;
E1EB 24 10        471      bit DIMFLG
E1ED 50 08        472      bvc >6
E1EF              473      ;
E1EF 18           474      clc
E1F0              475      ;
E1F0 68           476      pla
E1F1 69 01        477      adc #1
E1F3 AA           478      tax
E1F4              479      ;
E1F4 68           480      pla
E1F5 69 00        481      adc #ZERO
E1F7              482      ;
E1F7 C8           483      ^6      iny
E1F8              484      ;
E1F8 91 9B        485      sta (LOWTR),Y
E1FA              486      ;

```

```

E1FA C8          487      iny
E1FB             488      ;
E1FB 8A          489      txa
E1FC 91 9B       490      sta (LOWTR),Y
E1FE             491      ;
E1FE 20 AD E2    492      jsr MULSUBS
E201             493      ;
E201 86 AD       494      stx STRING2
E203 85 AE       495      sta STRING2+1
E205             496      ;
E205 A4 5E       497      ldy INDEX
E207             498      ;
E207 C6 0F       499      dec NUMDIM
E209 D0 DC       500      bne <5
E20B             501      ;
E20B 65 95       502      adc HIGHDS+1
E20D B0 5D       503      bcs >3
E20F             504      ;
E20F 85 95       505      sta HIGHDS+1
E211             506      ;
E211 A8          507      tay
E212 8A          508      txa
E213             509      ;
E213 65 94       510      adc HIGHDS
E215 90 03       511      bcc >7
E217             512      ;
E217 C8          513      iny
E218 F0 52       514      beq >3
E21A             515      ;
E21A 20 E3 D3    516      ^7      jsr CKSTRSIZ
E21D             517      ;
E21D 85 6D       518      sta STREND
E21F 84 6E       519      sty STREND+1
E221             520      ;
E221 A9 00       521      lda #ZERO
E223             522      ;
E223 E6 AE       523      inc STRING2+1
E225             524      ;
E225 A4 AD       525      ldy STRING2
E227 F0 05       526      beq >9
E229             527      ;
E229 88          528      ^8      dey
E22A             529      ;
E22A 91 94       530      sta (HIGHDS),Y
E22C             531      ;
E22C D0 FB       532      bne <8
E22E             533      ;
E22E C6 95       534      ^9      dec HIGHDS+1
E230             535      ;
E230 C6 AE       536      dec STRING2+1
E232 D0 F5       537      bne <8
E234             538      ;
E234 E6 95       539      inc HIGHDS+1
E236             540      ;
E236 A0 02       541      ldy #2
E238             542      ;
E238 38          543      sec
E239             544      ;
E239 A5 6D       545      lda STREND
E23B E5 9B       546      sbc LOWTR
E23D 91 9B       547      sta (LOWTR),Y

```

```

E23F          548 ;
E23F C8       549      iny
E240          550 ;
E240 A5 6E    551      lda STREND+1
E242 E5 9C    552      sbc LOWTR+1
E244 91 9B    553      sta (LOWTR),Y
E246          554 ;
E246 A5 10    555      lda DIMFLG
E248 D0 62    556      bne >9
E24A          557 ;
E24A C8       558      iny
E24B          559 ;
E24B          560 ;
E24B          561 ; Find Array Element.
E24B          562 ;
E24B B1 9B    563 FINDELE  lda (LOWTR),Y
E24D 85 0F    564      sta NUMDIM
E24F          565 ;
E24F A9 00    566      lda #ZERO
E251 85 AD    567      sta STRING2
E253          568 ;
E253 85 AE    569 ^1      sta STRING2+1
E255          570 ;
E255 C8       571      iny
E256          572 ;
E256 68       573      pla
E257 85 A0    574      sta VARPTR
E259 AA       575      tax
E25A          576 ;
E25A 68       577      pla
E25B 85 A1    578      sta VARPTR+1
E25D          579 ;
E25D D1 9B    580      cmp (LOWTR),Y
E25F 90 0E    581      bcc >4
E261          582 ;
E261 D0 06    583      bne >2
E263          584 ;
E263 C8       585      iny
E264          586 ;
E264 8A       587      txa
E265 D1 9B    588      cmp (LOWTR),Y
E267 90 07    589      bcc >5
E269          590 ;
E269 4C 96 E1 591 ^2      jmp BS.ERR
E26C          592 ;
E26C 4C 10 D4 593 ^3      jmp PRMESG07
E26F          594 ;
E26F C8       595 ^4      iny
E270          596 ;
E270 A5 AE    597 ^5      lda STRING2+1
E272 05 AD    598      ora STRING2
E274          599 ;
E274 18       600      clc
E275          601 ;
E275 F0 0A    602      beq >6
E277          603 ;
E277 20 AD E2 604      jsr MULSUBS
E27A          605 ;
E27A 8A       606      txa
E27B 65 A0    607      adc VARPTR
E27D AA       608      tax

```

```

E27E          609 ;
E27E 98       610 tya
E27F          611 ;
E27F A4 5E    612 ldy INDEX
E281          613 ;
E281 65 A1    614 ^6   adc VARPTR+1
E283          615 ;
E283 86 AD    616 stx STRING2
E285          617 ;
E285 C6 0F    618 dec NUMDIM
E287 D0 CA    619 bne <1
E289          620 ;
E289 85 AE    621 sta STRING2+1
E28B          622 ;
E28B A2 05    623 ldx #5
E28D          624 ;
E28D A5 81    625 lda VARNAM
E28F 10 01    626 bpl >7
E291          627 ;
E291 CA       628 dex
E292          629 ;
E292 A5 82    630 ^7   lda VARNAM+1
E294 10 02    631 bpl >8
E296          632 ;
E296 CA       633 dex
E297 CA       634 dex
E298          635 ;
E298 86 64    636 ^8   stx MULMANT+2
E29A          637 ;
E29A A9 00    638 lda #ZERO
E29C 20 B6 E2 639 jsr MULSUBS2
E29F          640 ;
E29F 8A       641 txa
E2A0 65 94    642 adc HIGHDS
E2A2 85 83    643 sta VARPNT
E2A4          644 ;
E2A4 98       645 tya
E2A5 65 95    646 adc HIGHDS+1
E2A7 85 84    647 sta VARPNT+1
E2A9          648 ;
E2A9 A8       649 tay
E2AA          650 ;
E2AA A5 83    651 lda VARPNT
E2AC          652 ;
E2AC 60       653 ^9   rts
E2AD          654 ;
E2AD          655 ;
E2AD 84 5E    656 MULSUBS sty INDEX
E2AF          657 ;
E2AF B1 9B    658 lda (LOWTR),Y
E2B1 85 64    659 sta MULMANT+2
E2B3          660 ;
E2B3 88       661 dey
E2B4          662 ;
E2B4 B1 9B    663 lda (LOWTR),Y
E2B6          664 ;
E2B6          665 ;
E2B6 85 65    666 MULSUBS2 sta MULMANT+3
E2B8          667 ;
E2B8 A9 10    668 lda #$10
E2BA 85 99    669 sta TEMP2+1

```

```

E2BC          670 ;
E2BC A2 00    671     ldx #ZERO
E2BE A0 00    672     ldy #ZERO
E2C0          673 ;
E2C0 8A      674 ^1     txa
E2C1 0A      675     asl
E2C2 AA      676     tax
E2C3          677 ;
E2C3 98      678     tya
E2C4 2A      679     rol
E2C5 A8      680     tay
E2C6          681 ;
E2C6 B0 A4   682     bcs <3
E2C8          683 ;
E2C8 06 AD   684     asl STRING2
E2CA 26 AE   685     rol STRING2+1
E2CC 90 0B   686     bcc >2
E2CE          687 ;
E2CE 18      688     clc
E2CF          689 ;
E2CF 8A      690     txa
E2D0 65 64   691     adc MULMANT+2
E2D2 AA      692     tax
E2D3          693 ;
E2D3 98      694     tya
E2D4 65 65   695     adc MULMANT+3
E2D6 A8      696     tay
E2D7          697 ;
E2D7 B0 93   698     bcs <3
E2D9          699 ;
E2D9 C6 99   700 ^2     dec TEMP2+1
E2DB D0 E3   701     bne <1
E2DD          702 ;
E2DD 60      703     rts
E2DE          704 ;
E2DE          705 ;
E2DE A5 11   706 FFRE   lda VALTYP
E2E0 F0 03   707     beq >3
E2E2          708 ;
E2E2 20 00 E6 709     jsr FRESTR2
E2E5          710 ;
E2E5 20 84 E4 711 ^3     jsr GARBAG
E2E8          712 ;
E2E8 38      713     sec
E2E9          714 ;
E2E9 A5 6F   715     lda FRETOP
E2EB E5 6D   716     sbc STREND
E2ED A8      717     tay
E2EE          718 ;
E2EE A5 70   719     lda FRETOP+1
E2F0 E5 6E   720     sbc STREND+1
E2F2          721 ;
E2F2 A2 00   722 GIVAYFP ldx #ZERO
E2F4 86 11   723     stx VALTYP
E2F6          724 ;
E2F6 85 9E   725     sta FACMANT
E2F8 84 9F   726     sty FACMANT+1
E2FA          727 ;
E2FA A2 90   728     ldx #$90
E2FC          729 ;
E2FC 4C 9B EB 730     jmp FLOAT2

```

```

E2FF      731 ;
E2FF      732 ;
E2FF A4 24 733 FPOS      ldy CH
E301      734 ;
E301      735 ;
E301 38    736 SNGFLT    sec
E302      737 ;
E302 A9 00 738          lda #ZERO
E304 F0 EC 739          beq GIVAYFP      ; always taken
E306      740 ;
E306      741 ;
E306 A6 76 742 ERRDIR    ldx CURLIN+1
E308      743 ;
E308 E8    744          inx
E309 D0 A1 745          bne <9
E30B      746 ;
E30B A2 97 747          ldx #MSG12-MESGS  ; Illegal Direct error
E30D      748 ;
E30D 2C 00 00 749          bit *-*
E310      750          dfs !-2
E30E      751 ;
E30E A2 E3    752 ^4      ldx #MSG17-MESGS  ; Undefined Function error
E310      753 ;
E310 4C 12 D4 754          jmp FSCRN
E313      755 ;
E313      756 ;
E313 20 41 E3 757 BDEF      jsr GETFNC
E316 20 06 E3 758          jsr ERRDIR
E319 20 BB DE 759          jsr CHKOPNP
E31C      760 ;
E31C A9 80    761          lda #$80
E31E 85 14    762          sta SUBFLG
E320      763 ;
E320 20 E3 DF 764          jsr PTRGET
E323 20 6A DD 765          jsr CHKNUM
E326 20 B8 DE 766          jsr CHKCLSP
E329      767 ;
E329 A9 D0    768          lda #TKEQUAL
E32B 20 C0 DE 769          jsr SYNTAXCHK
E32E      770 ;
E32E 48      771          pha
E32F      772 ;
E32F A5 84    773          lda VARPNT+1
E331 48      774          pha
E332      775 ;
E332 A5 83    776          lda VARPNT
E334 48      777          pha
E335      778 ;
E335 A5 B9    779          lda TXTPTR+1
E337 48      780          pha
E338      781 ;
E338 A5 B8    782          lda TXTPTR
E33A 48      783          pha
E33B      784 ;
E33B 20 95 D9 785          jsr BDATA
E33E      786 ;
E33E 4C AF E3 787          jmp FNCDATA
E341      788 ;
E341      789 ;
E341 A9 C2    790 GETFNC    lda #TKFN
E343 20 C0 DE 791          jsr SYNTAXCHK

```

```

E346          792 ;
E346 09 80    793      ora #$80
E348 85 14    794      sta SUBFLG
E34A          795 ;
E34A 20 EA DF 796      jsr PTRGET3
E34D          797 ;
E34D 85 8A    798      sta FUNCNAM
E34F 84 8B    799      sty FUNCNAM+1
E351          800 ;
E351 4C 6A DD 801      jmp CHKNUM
E354          802 ;
E354          803 ;
E354 20 41 E3 804 CALLFNC jsr GETFNC
E357          805 ;
E357 A5 8B    806      lda FUNCNAM+1
E359 48       807      pha
E35A          808 ;
E35A A5 8A    809      lda FUNCNAM
E35C 48       810      pha
E35D          811 ;
E35D 20 B2 DE 812      jsr PARENCHK
E360 20 6A DD 813      jsr CHKNUM
E363          814 ;
E363 68       815      pla
E364 85 8A    816      sta FUNCNAM
E366          817 ;
E366 68       818      pla
E367 85 8B    819      sta FUNCNAM+1
E369          820 ;
E369 A0 02    821      ldy #2
E36B          822 ;
E36B B1 8A    823      lda (FUNCNAM),Y
E36D 85 83    824      sta VARPNT
E36F AA       825      tax
E370          826 ;
E370 C8       827      iny
E371          828 ;
E371 B1 8A    829      lda (FUNCNAM),Y
E373 F0 99    830      beq <4
E375          831 ;
E375 85 84    832      sta VARPNT+1
E377          833 ;
E377 C8       834      iny
E378          835 ;
E378 B1 83    836 ^5    lda (VARPNT),Y
E37A 48       837      pha
E37B          838 ;
E37B 88       839      dey
E37C 10 FA    840      bpl <5
E37E          841 ;
E37E A4 84    842      ldy VARPNT+1
E380          843 ;
E380 20 2B EB 844      jsr COPYFAC
E383          845 ;
E383 A5 B9    846      lda TXTPTR+1
E385 48       847      pha
E386          848 ;
E386 A5 B8    849      lda TXTPTR
E388 48       850      pha
E389          851 ;
E389 B1 8A    852      lda (FUNCNAM),Y

```

```

E38B 85 B8      853      sta TXTPTR
E38D           854      ;
E38D C8        855      iny
E38E           856      ;
E38E B1 8A     857      lda (FUNCNAM),Y
E390 85 B9     858      sta TXTPTR+1
E392           859      ;
E392 A5 84     860      lda VARPNT+1
E394 48        861      pha
E395           862      ;
E395 A5 83     863      lda VARPNT
E397 48        864      pha
E398           865      ;
E398 20 67 DD  866      jsr FRMNUM
E39B           867      ;
E39B 68        868      pla
E39C 85 8A     869      sta FUNCNAM
E39E           870      ;
E39E 68        871      pla
E39F 85 8B     872      sta FUNCNAM+1
E3A1           873      ;
E3A1 20 B7 00  874      jsr CHRGOT
E3A4 F0 03     875      beq >6
E3A6           876      ;
E3A6 4C C9 DE  877      jmp DOMESG02
E3A9           878      ;
E3A9           879      ;
E3A9           880      ; Retrieve TXTPTR.
E3A9           881      ;
E3A9 68        882      ^6      pla
E3AA 85 B8     883      sta TXTPTR
E3AC           884      ;
E3AC 68        885      pla
E3AD 85 B9     886      sta TXTPTR+1
E3AF           887      ;
E3AF           888      ;
E3AF           889      ; Get five bytes from STACK and save at FUNCNAM.
E3AF           890      ;
E3AF A0 00     891      FNCDATA ldy #ZERO
E3B1           892      ;
E3B1 68        893      pla
E3B2 91 8A     894      sta (FUNCNAM),Y
E3B4           895      ;
E3B4 C8        896      iny
E3B5           897      ;
E3B5 68        898      pla
E3B6 91 8A     899      sta (FUNCNAM),Y
E3B8           900      ;
E3B8 C8        901      iny
E3B9           902      ;
E3B9 68        903      pla
E3BA 91 8A     904      sta (FUNCNAM),Y
E3BC           905      ;
E3BC C8        906      iny
E3BD           907      ;
E3BD 68        908      pla
E3BE 91 8A     909      sta (FUNCNAM),Y
E3C0           910      ;
E3C0 C8        911      iny
E3C1           912      ;
E3C1 68        913      pla

```



```
E3C2 91 8A      914      sta (FUNCNAM),Y
E3C4            915      ;
E3C4 60         916      rts
E3C5            917      ;
E3C5            918      ;
E3C5            919      ; All numeric strings now start at the beginning of the
E3C5            920      ; STACK, so this routine is modified accordingly. FPOUT
E3C5            921      ; returns with A/Y-reg containing the address of the STACK.
E3C5            922      ;
E3C5 20 6A DD   923      FSTR      jsr CHKNUM
E3C8            924      ;
E3C8 68         925      pla
E3C9 68         926      pla
E3CA            927      ;
E3CA 20 34 ED   928      jsr FPOUT
E3CD D0 18      929      bne STRLIT      ; always taken
E3CF            930      ;
E3CF            931      ;
E3CF            932      dfs 6,ZERO      ; 6 bytes
E3D5            933      ;
E3D5            934      ;
E3D5 A6 A0      935      STRINI      ldx VARPTR
E3D7 A4 A1      936      ldy VARPTR+1
E3D9            937      ;
E3D9 86 8C      938      stx DSCPTR
E3DB 84 8D      939      sty DSCPTR+1
E3DD            940      ;
E3DD            941      ;
E3DD 20 52 E4   942      STRSPA      jsr GETSSPC
E3E0            943      ;
E3E0 85 9D      944      sta FACEXP
E3E2 86 9E      945      stx FACMANT
E3E4 84 9F      946      sty FACMANT+1
E3E6            947      ;
E3E6 60         948      rts
E3E7            949      ;
E3E7            950      ;
E3E7            951      icl "E4.L"
```

LLOAD E4.L,A\$4000

```

E3E7          1          ttl "ROM Source Code, E4.L"
E3E7          2          ;
E3E7          3          ;
E3E7          4          ; E4.L
E3E7          5          ;
E3E7          6          ;
E3E7 A2 22     7  STRLIT   ldx #'"'
E3E9 86 0D     8          stx CHARAC
E3EB 86 0E     9          stx ENDCHR
E3ED          10         ;
E3ED          11         ;
E3ED 85 AB    12  STRLIT2  sta STRING1
E3EF 84 AC    13          sty STRING1+1
E3F1          14         ;
E3F1 85 9E    15          sta FACMANT
E3F3 84 9F    16          sty FACMANT+1
E3F5          17         ;
E3F5 A0 FF    18          ldy #NEGONE
E3F7          19         ;
E3F7 C8       20         ^8   iny
E3F8          21         ;
E3F8 B1 AB    22          lda (STRING1),Y
E3FA F0 0C    23          beq >1
E3FC          24         ;
E3FC C5 0D    25          cmp CHARAC
E3FE F0 04    26          beq >9
E400          27         ;
E400 C5 0E    28          cmp ENDCHR
E402 D0 F3    29          bne <8
E404          30         ;
E404          31         ;
E404 C9 22    32         ^9   cmp #'"'
E406 F0 01    33          beq >2
E408          34         ;
E408 18       35         ^1   clc
E409          36         ;
E409 84 9D    37         ^2   sty DSCTMP
E40B 98       38          tya
E40C          39         ;
E40C A6 AC    40          ldx STRING1+1
E40E          41         ;
E40E 65 AB    42          adc STRING1
E410 85 AD    43          sta STRING2
E412 90 01    44          bcc >3
E414          45         ;
E414 E8       46          inx
E415          47         ;
E415 86 AE    48         ^3   stx STRING2+1
E417          49         ;
E417 A5 AC    50          lda STRING1+1
E419 F0 04    51          beq >4
E41B          52         ;
E41B C9 02    53          cmp #2
E41D D0 0B    54          bne PUTNEW
E41F          55         ;
E41F 98       56         ^4   tya
E420          57         ;
E420 20 D5 E3 58          jsr STRINI
E423          59         ;
E423 A6 AB    60          ldx STRING1

```

```

E425 A4 AC      61      ldy STRING1+1
E427           62      ;
E427 20 E2 E5   63      jsr MOVSTR
E42A           64      ;
E42A           65      ;
E42A A6 52      66      PUTNEW    ldx TEMPPT
E42C E0 5E      67      cpx #TEMPST+9      ; max of 3 temp strings
E42E D0 05      68      bne >6
E430           69      ;
E430 A2 C1      70      ldx #MESG15-MESGS      ; Formula too Complex error
E432           71      ;
E432 4C 12 D4   72      ^5      jmp FSCRN
E435           73      ;
E435 A5 9D      74      ^6      lda DSCTMP
E437 95 00      75      sta LOC0,X
E439           76      ;
E439 A5 9E      77      lda DSCTMP+1
E43B 95 01      78      sta LOC1,X
E43D           79      ;
E43D A5 9F      80      lda DSCTMP+2
E43F 95 02      81      sta LOC2,X
E441           82      ;
E441 A0 00      83      ldy #ZERO
E443           84      ;
E443 86 A0      85      stx VARPTR
E445 84 A1      86      sty VARPTR+1
E447           87      ;
E447 88         88      dey
E448           89      ;
E448 84 11      90      sty VALTYP
E44A 86 53      91      stx LASTPT
E44C           92      ;
E44C E8         93      inx
E44D E8         94      inx
E44E E8         95      inx
E44F           96      ;
E44F 86 52      97      stx TEMPPT
E451           98      ;
E451 60         99      rts
E452          100      ;
E452          101      ;
E452          102      ; Get space for A-reg bytes.  Return address in X/Y-reg.
E452          103      ;
E452 46 13      104      GETSSPC  lsr GARFLG
E454          105      ;
E454 48         106      ^7      pha
E455          107      ;
E455 A4 70      108      ldy FRETOP+1
E457          109      ;
E457 38         110      sec
E458          111      ;
E458 49 FF      112      eor #NEGONE
E45A 65 6F      113      adc FRETOP
E45C B0 01      114      bcs >8
E45E          115      ;
E45E 88         116      dey
E45F          117      ;
E45F C4 6E      118      ^8      cpy STREND+1
E461 90 11      119      bcc >1
E463          120      ;
E463 D0 04      121      bne >9

```

```

E465          122 ;
E465 C5 6D    123      cmp STREND
E467 90 0B    124      bcc >1
E469          125 ;
E469 85 6F    126 ^9    sta FRETOP
E46B 84 70    127      sty FRETOP+1
E46D          128 ;
E46D 85 71    129      sta FRESPC
E46F 84 72    130      sty FRESPC+1
E471          131 ;
E471 AA       132      tax
E472          133 ;
E472 68       134      pla
E473          135 ;
E473 60       136      rts
E474          137 ;
E474 A2 4D    138 ^1    ldx #MSG07-MESGS ; Out of Memory error
E476          139 ;
E476 A5 13    140      lda GARFLG
E478 30 B8    141      bmi <5
E47A          142 ;
E47A 20 84 E4 143      jsr GARBAG
E47D          144 ;
E47D A9 80    145      lda #$80
E47F 85 13    146      sta GARFLG
E481          147 ;
E481 68       148      pla
E482 D0 D0    149      bne <7 ; always taken
E484          150 ;
E484          151 ;
E484          152 ; GARBAG has been rewritten according to the concepts that
E484          153 ; were developed by Cornelis Bongers. This routine uses
E484          154 ; the CX space at 0xC600 for the PROCVAR and PROCSPCL
E484          155 ; routines.
E484          156 ;
E484 A9 00    157 GARBAG  lda #ZERO
E486 85 8F    158      sta SPCLFLAG
E488 85 95    159      sta PROCESS
E48A 8D 07 C0 160      sta CXROMON
E48D          161 ;
E48D          162 ;
E48D A5 69    163 CHKVARS  lda VARTAB
E48F A6 6A    164      ldx VARTAB+1
E491          165 ;
E491 85 9B    166      sta LOWTR
E493 86 9C    167      stx LOWTR+1
E495          168 ;
E495 C5 6B    169 ^1    cmp ARYTAB
E497 D0 04    170      bne >2
E499          171 ;
E499 E4 6C    172      cpx ARYTAB+1
E49B F0 25    173      beq CHKARRYS
E49D          174 ;
E49D A0 00    175 ^2    ldy #ZERO
E49F          176 ;
E49F B1 9B    177      lda (LOWTR),Y
E4A1 30 10    178      bmi >3
E4A3          179 ;
E4A3 C8       180      iny
E4A4          181 ;
E4A4 B1 9B    182      lda (LOWTR),Y

```

```

E4A6 10 0B      183      bpl >3
E4A8            184      ;
E4A8 A9 02      185      lda #2
E4AA 20 67 E5    186      jsr NXTVAR
E4AD            187      ;
E4AD 20 00 C6    188      jsr PROCVAR
E4B0            189      ;
E4B0 A9 05      190      lda #SVARLEN-2
E4B2            191      ;
E4B2 2C 00 00    192      bit *-*
E4B5            193      dfs !-2
E4B3            194      ;
E4B3 A9 07      195      ^3      lda #SVARLEN
E4B5            196      ;
E4B5 20 67 E5    197      jsr NXTVAR
E4B8 90 DB      198      bcc <1
E4BA            199      ;
E4BA A5 96      200      ^0      lda HIGHTR
E4BC A6 97      201      ldx HIGHTR+1
E4BE            202      ;
E4BE 85 9B      203      sta LOWTR
E4C0 86 9C      204      stx LOWTR+1
E4C2            205      ;
E4C2            206      ;
E4C2 C5 6D      207      CHKARRYS cmp STREND
E4C4 D0 04      208      bne >1
E4C6            209      ;
E4C6 E4 6E      210      cpx STREND+1
E4C8 F0 33      211      beq >4
E4CA            212      ;
E4CA 18          213      ^1      clc
E4CB            214      ;
E4CB A0 02      215      ldy #2
E4CD            216      ;
E4CD 71 9B      217      adc (LOWTR),Y
E4CF 85 96      218      sta HIGHTR
E4D1            219      ;
E4D1 C8          220      iny
E4D2            221      ;
E4D2 8A          222      txa
E4D3 71 9B      223      adc (LOWTR),Y
E4D5 85 97      224      sta HIGHTR+1
E4D7            225      ;
E4D7 A0 00      226      ldy #ZERO
E4D9            227      ;
E4D9 B1 9B      228      lda (LOWTR),Y
E4DB 30 DD      229      bmi <0
E4DD            230      ;
E4DD C8          231      iny
E4DE            232      ;
E4DE B1 9B      233      lda (LOWTR),Y
E4E0 10 D8      234      bpl <0
E4E2            235      ;
E4E2 A0 04      236      ldy #4
E4E4            237      ;
E4E4 B1 9B      238      lda (LOWTR),Y
E4E6 0A          239      asl
E4E7 69 05      240      adc #AHDRLEN
E4E9            241      ;
E4E9 20 67 E5    242      ^2      jsr NXTVAR
E4EC            243      ;

```

```

E4EC C5 96      244      cmp HIGHTR
E4EE D0 04      245      bne >3
E4F0            246      ;
E4F0 E4 97      247      cpx HIGHTR+1
E4F2 F0 CE      248      beq CHKARRYS
E4F4            249      ;
E4F4 A0 01      250      ^3      ldy #1
E4F6 20 00 C6    251      jsr PROCVAR
E4F9            252      ;
E4F9 A9 03      253      lda #AVARLEN
E4FB D0 EC      254      bne <2      ; always taken
E4FD            255      ;
E4FD 24 95      256      ^4      bit PROCESS
E4FF 10 0B      257      bpl MOVVARS
E501            258      ;
E501            259      ;
E501 A5 8A      260      GARBEXIT lda FUNCNAM
E503 A6 8B      261      ldx FUNCNAM+1
E505            262      ;
E505 85 6F      263      sta FRETOP
E507 86 70      264      stx FRETOP+1
E509            265      ;
E509 4C 3C FA    266      jmp CXOFF
E50C            267      ;
E50C            268      ;
E50C A5 73      269      MOVVARS  lda MEMSIZE
E50E A6 74      270      ldx MEMSIZE+1
E510            271      ;
E510 85 8A      272      sta FUNCNAM
E512 86 8B      273      stx FUNCNAM+1
E514            274      ;
E514 85 9B      275      sta LOWTR
E516 86 9C      276      stx LOWTR+1
E518            277      ;
E518 A0 00      278      ^1      ldy #ZERO
E51A            279      ;
E51A A5 9B      280      ^2      lda LOWTR
E51C C5 6F      281      cmp FRETOP
E51E D0 04      282      bne >3
E520            283      ;
E520 E4 70      284      cpx FRETOP+1
E522 F0 3A      285      beq >4
E524            286      ;
E524 A9 FF      287      ^3      lda #NEGONE
E526 20 8C E5    288      jsr DECPTR
E529            289      ;
E529 B1 9B      290      lda (LOWTR),Y
E52B 10 ED      291      bpl <2
E52D            292      ;
E52D 29 7F      293      and #MSBCLR
E52F 91 9B      294      sta (LOWTR),Y
E531            295      ;
E531 A9 FE      296      lda #NEG TWO
E533 20 8C E5    297      jsr DECPTR
E536            298      ;
E536 B1 9B      299      lda (LOWTR),Y
E538 85 5E      300      sta INDEX
E53A            301      ;
E53A C8          302      iny
E53B            303      ;
E53B B1 9B      304      lda (LOWTR),Y

```

```

E53D 85 5F      305      sta INDEX+1
E53F           306      ;
E53F C8        307      iny
E540           308      ;
E540 B1 5E     309      lda (INDEX),Y
E542           310      ;
E542 88        311      dey
E543           312      ;
E543 91 9B     313      sta (LOWTR),Y
E545           314      ;
E545 B1 5E     315      lda (INDEX),Y
E547           316      ;
E547 88        317      dey
E548           318      ;
E548 91 9B     319      sta (LOWTR),Y
E54A           320      ;
E54A B1 5E     321      lda (INDEX),Y
E54C 85 94     322      sta LEN
E54E           323      ;
E54E 20 73 E5  324      jsr COPYVAR
E551           325      ;
E551 C8        326      iny
E552           327      ;
E552 A5 8A     328      lda FUNCNAM
E554 91 5E     329      sta (INDEX),Y
E556           330      ;
E556 C8        331      iny
E557           332      ;
E557 A5 8B     333      lda FUNCNAM+1
E559 91 5E     334      sta (INDEX),Y
E55B           335      ;
E55B 4C 18 E5  336      jmp <1
E55E           337      ;
E55E 24 8F     338      ^4 bit SPCLFLAG
E560 10 9F     339      bpl GARBEXIT
E562           340      ;
E562 C6 95     341      dec PROCESS
E564           342      ;
E564 4C 8D E4  343      jmp CHKVARS
E567           344      ;
E567           345      ;
E567 18        346      NXTVAR clc
E568           347      ;
E568 65 9B     348      adc LOWTR
E56A 85 9B     349      sta LOWTR
E56C 90 04     350      bcc >1
E56E           351      ;
E56E E6 9C     352      inc LOWTR+1
E570           353      ;
E570 E8        354      inx
E571           355      ;
E571 18        356      clc
E572           357      ;
E572 60        358      ^1 rts
E573           359      ;
E573           360      ;
E573 38        361      COPYVAR sec
E574           362      ;
E574 A5 8A     363      lda FUNCNAM
E576 E5 94     364      sbc LEN
E578 85 8A     365      sta FUNCNAM

```

```

E57A          366 ;
E57A A5 8B    367      lda FUNCNAM+1
E57C E9 00    368      sbc #ZERO
E57E 85 8B    369      sta FUNCNAM+1
E580          370 ;
E580 A4 94    371      ldy LEN
E582          372 ;
E582 88       373 ^1    dey
E583          374 ;
E583 B1 9B    375      lda (LOWTR),Y
E585 91 8A    376      sta (FUNCNAM),Y
E587          377 ;
E587 C0 00    378      cpy #ZERO
E589 D0 F7    379      bne <1
E58B          380 ;
E58B 60       381      rts
E58C          382 ;
E58C          383 ;
E58C 18       384 DECPTR  clc
E58D          385 ;
E58D 65 9B    386      adc LOWTR
E58F 85 9B    387      sta LOWTR
E591 B0 03    388      bcs >1
E593          389 ;
E593 C6 9C    390      dec LOWTR+1
E595          391 ;
E595 CA       392      dex
E596          393 ;
E596 60       394 ^1    rts
E597          395 ;
E597          396 ;
E597 A5 A1    397 CAT2STR  lda VARPTR+1
E599 48       398      pha
E59A          399 ;
E59A A5 A0    400      lda VARPTR
E59C 48       401      pha
E59D          402 ;
E59D 20 63 DE 403      jsr FRMELMNT
E5A0 20 6C DD 404      jsr CHKSTR
E5A3          405 ;
E5A3 68       406      pla
E5A4 85 AB    407      sta STRING1
E5A6          408 ;
E5A6 68       409      pla
E5A7 85 AC    410      sta STRING1+1
E5A9          411 ;
E5A9 A0 00    412      ldy #ZERO
E5AB          413 ;
E5AB 18       414      clc
E5AC          415 ;
E5AC B1 AB    416      lda (STRING1),Y
E5AE 71 A0    417      adc (VARPTR),Y
E5B0 90 05    418      bcc >1
E5B2          419 ;
E5B2 A2 B2    420      ldx #MSG14-MESGS ; String too Long error
E5B4          421 ;
E5B4 4C 12 D4 422      jmp FSCRN
E5B7          423 ;
E5B7 20 D5 E3 424 ^1    jsr STRINI
E5BA 20 D4 E5 425      jsr MOVINS
E5BD          426 ;

```



```

E5BD A5 8C      427      lda DSCPTR
E5BF A4 8D      428      ldy DSCPTR+1
E5C1           429      ;
E5C1 20 04 E6   430      jsr FRETMP
E5C4 20 E6 E5   431      jsr MOVSTR2
E5C7           432      ;
E5C7 A5 AB      433      lda STRING1
E5C9 A4 AC      434      ldy STRING1+1
E5CB           435      ;
E5CB 20 04 E6   436      jsr FRETMP
E5CE 20 2A E4   437      jsr PUTNEW
E5D1           438      ;
E5D1 4C 95 DD   439      jmp FRMEVAL3
E5D4           440      ;
E5D4           441      ;
E5D4 A0 00      442      MOVINS ldy #ZERO
E5D6           443      ;
E5D6 B1 AB      444      lda (STRING1),Y
E5D8 48         445      pha
E5D9           446      ;
E5D9 C8         447      iny
E5DA           448      ;
E5DA B1 AB      449      lda (STRING1),Y
E5DC AA         450      tax
E5DD           451      ;
E5DD C8         452      iny
E5DE           453      ;
E5DE B1 AB      454      lda (STRING1),Y
E5E0 A8         455      tay
E5E1           456      ;
E5E1 68         457      pla
E5E2           458      ;
E5E2           459      ;
E5E2 86 5E      460      MOVSTR stx INDEX
E5E4 84 5F      461      sty INDEX+1
E5E6           462      ;
E5E6           463      ;
E5E6 A8         464      MOVSTR2 tay
E5E7 F0 0A      465      beq >3
E5E9           466      ;
E5E9 48         467      pha
E5EA           468      ;
E5EA 88         469      ^2    dey
E5EB           470      ;
E5EB B1 5E      471      lda (INDEX),Y
E5ED 91 71      472      sta (FRESPC),Y
E5EF           473      ;
E5EF 98         474      tya
E5F0 D0 F8      475      bne <2
E5F2           476      ;
E5F2 68         477      pla
E5F3           478      ;
E5F3 18         479      ^3    clc
E5F4           480      ;
E5F4 65 71      481      adc FRESPC
E5F6 85 71      482      sta FRESPC
E5F8 90 02      483      bcc >4
E5FA           484      ;
E5FA E6 72      485      inc FRESPC+1
E5FC           486      ;
E5FC 60         487      ^4    rts

```

```

E5FD          488 ;
E5FD          489 ;
E5FD 20 6C DD 490 FRESTR    jsr  CHKSTR
E600          491 ;
E600          492 ;
E600 A5 A0    493 FRESTR2   lda  VARPTR
E602 A4 A1    494          ldy  VARPTR+1
E604          495 ;
E604          496 ;
E604 85 5E    497 FRETMP    sta  INDEX
E606 84 5F    498          sty  INDEX+1
E608          499 ;
E608 20 35 E6 500          jsr  FRETMS
E60B          501 ;
E60B 08       502          php
E60C          503 ;
E60C A0 00    504          ldy  #ZERO
E60E          505 ;
E60E B1 5E    506          lda  (INDEX),Y
E610 48       507          pha
E611          508 ;
E611 C8       509          iny
E612          510 ;
E612 B1 5E    511          lda  (INDEX),Y
E614 AA       512          tax
E615          513 ;
E615 C8       514          iny
E616          515 ;
E616 B1 5E    516          lda  (INDEX),Y
E618 A8       517          tay
E619          518 ;
E619 68       519          pla
E61A          520 ;
E61A 28       521          plp
E61B D0 13    522          bne  >6
E61D          523 ;
E61D C4 70    524          cpy  FRETOP+1
E61F D0 0F    525          bne  >6
E621          526 ;
E621 E4 6F    527          cpx  FRETOP
E623 D0 0B    528          bne  >6
E625          529 ;
E625 48       530          pha
E626          531 ;
E626 18       532          clc
E627          533 ;
E627 65 6F    534          adc  FRETOP
E629 85 6F    535          sta  FRETOP
E62B 90 02    536          bcc  >5
E62D          537 ;
E62D E6 70    538          inc  FRETOP+1
E62F          539 ;
E62F 68       540 ^5      pla
E630          541 ;
E630 86 5E    542 ^6      stx  INDEX
E632 84 5F    543          sty  INDEX+1
E634          544 ;
E634 60       545          rts
E635          546 ;
E635          547 ;
E635 C4 54    548 FRETMS    cpy  LASTPT+1

```

```

E637 D0 0C      549      bne >7
E639            550      ;
E639 C5 53      551      cmp LASTPT
E63B D0 08      552      bne >7
E63D            553      ;
E63D 85 52      554      sta TEMPPT
E63F            555      ;
E63F E9 03      556      sbc #3
E641 85 53      557      sta LASTPT
E643            558      ;
E643 A0 00      559      ldy #ZERO
E645            560      ;
E645 60          561      ^7      rts
E646            562      ;
E646            563      ;
E646 20 FB E6    564      FCHR      jsr CONVINT
E649            565      ;
E649 8A          566      txa
E64A 48          567      pha
E64B            568      ;
E64B A9 01      569      lda #1
E64D 20 DD E3    570      jsr STRSPA
E650            571      ;
E650 A0 00      572      ldy #ZERO
E652            573      ;
E652 68          574      pla
E653 91 9E      575      sta (FACMANT),Y
E655            576      ;
E655 68          577      pla
E656 68          578      pla
E657            579      ;
E657 4C 2A E4    580      jmp PUTNEW
E65A            581      ;
E65A            582      ;
E65A 20 B9 E6    583      FLEFT      jsr STRSETUP
E65D            584      ;
E65D D1 8C      585      cmp (DSCPTR),Y
E65F            586      ;
E65F 98          587      tya
E660            588      ;
E660 90 04      589      ^8      bcc >9
E662            590      ;
E662 B1 8C      591      lda (DSCPTR),Y
E664 AA          592      tax
E665            593      ;
E665 98          594      tya
E666            595      ;
E666 48          596      ^9      pha
E667            597      ;
E667 8A          598      ^1      txa
E668            599      ;
E668 48          600      ^2      pha
E669            601      ;
E669 20 DD E3    602      jsr STRSPA
E66C            603      ;
E66C A5 8C      604      lda DSCPTR
E66E A4 8D      605      ldy DSCPTR+1
E670            606      ;
E670 20 04 E6    607      jsr FRETMP
E673            608      ;
E673 18          609      clc

```

```

E674          610 ;
E674 68        611      pla
E675 A8        612      tay
E676          613 ;
E676 68        614      pla
E677 65 5E     615      adc INDEX
E679 85 5E     616      sta INDEX
E67B 90 02     617      bcc >3
E67D          618 ;
E67D E6 5F     619      inc INDEX+1
E67F          620 ;
E67F 98        621 ^3    tya
E680          622 ;
E680 20 E6 E5   623      jsr MOVSTR2
E683          624 ;
E683 4C 2A E4   625      jmp PUTNEW
E686          626 ;
E686          627 ;
E686 20 B9 E6   628 FRIGHT jsr STRSETUP
E689          629 ;
E689 18        630      clc
E68A          631 ;
E68A F1 8C     632      sbc (DSCPTR),Y
E68C 49 FF     633      eor #NEGONE
E68E          634 ;
E68E 4C 60 E6   635      jmp <8
E691          636 ;
E691          637 ;
E691 A9 FF     638 FMID   lda #NEGONE
E693 85 A1     639      sta VARPTR+1
E695          640 ;
E695 20 B7 00   641      jsr CHRGOT
E698          642 ;
E698 C9 29     643      cmp #$29
E69A F0 06     644      beq >4
E69C          645 ;
E69C 20 BE DE   646      jsr CHKCOM
E69F 20 F8 E6   647      jsr GETBYT
E6A2          648 ;
E6A2 20 B9 E6   649 ^4    jsr STRSETUP
E6A5          650 ;
E6A5 CA        651      dex
E6A6          652 ;
E6A6 8A        653      txa
E6A7 48        654      pha
E6A8          655 ;
E6A8 18        656      clc
E6A9          657 ;
E6A9 A2 00     658      ldx #ZERO
E6AB          659 ;
E6AB F1 8C     660      sbc (DSCPTR),Y
E6AD B0 B8     661      bcs <1
E6AF          662 ;
E6AF 49 FF     663      eor #NEGONE
E6B1          664 ;
E6B1 C5 A1     665      cmp VARPTR+1
E6B3 90 B3     666      bcc <2
E6B5          667 ;
E6B5 A5 A1     668      lda VARPTR+1
E6B7          669 ;
E6B7 B0 AF     670      bcs <2

```

; always taken

```

E6B9          671 ;
E6B9          672 ;
E6B9 20 B8 DE 673 STRSETUP jsr CHKCLSP
E6BC          674 ;
E6BC 68       675         pla
E6BD A8       676         tay
E6BE          677 ;
E6BE 68       678         pla
E6BF 85 91    679         sta RTNADLEN
E6C1          680 ;
E6C1 68       681         pla
E6C2 68       682         pla
E6C3          683 ;
E6C3 68       684         pla
E6C4 AA       685         tax
E6C5          686 ;
E6C5 68       687         pla
E6C6 85 8C    688         sta DSCPTR
E6C8          689 ;
E6C8 68       690         pla
E6C9 85 8D    691         sta DSCPTR+1
E6CB          692 ;
E6CB A5 91    693         lda RTNADLEN
E6CD 48       694         pha
E6CE          695 ;
E6CE 98       696         tya
E6CF 48       697         pha
E6D0          698 ;
E6D0 A0 00    699         ldy #ZERO
E6D2          700 ;
E6D2 8A       701         txa
E6D3 F0 1D    702         beq DO.IQ.ER
E6D5          703 ;
E6D5 60       704         rts
E6D6          705 ;
E6D6          706 ;
E6D6 20 DC E6 707 FLEN      jsr GETSTRLN
E6D9          708 ;
E6D9 4C 01 E3 709         jmp SNGFLT
E6DC          710 ;
E6DC          711 ;
E6DC 20 FD E5 712 GETSTRLN jsr FRESTR
E6DF          713 ;
E6DF A2 00    714         ldx #ZERO
E6E1 86 11    715         stx VALTYP
E6E3          716 ;
E6E3 A8       717         tay
E6E4          718 ;
E6E4 60       719         rts
E6E5          720 ;
E6E5          721 ;
E6E5 20 DC E6 722 FASC      jsr GETSTRLN
E6E8 F0 08    723         beq DO.IQ.ER
E6EA          724 ;
E6EA A0 00    725         ldy #ZERO
E6EC          726 ;
E6EC B1 5E    727         lda (INDEX),Y
E6EE A8       728         tay
E6EF          729 ;
E6EF 4C 01 E3 730         jmp SNGFLT
E6F2          731 ;

```

```

E6F2      732 ;
E6F2 4C 99 E1 733 DO.IQ.ER jmp IQ.ERR
E6F5      734 ;
E6F5      735 ;
E6F5 20 B1 00 736 GETBYTC jsr CHRGET
E6F8      737 ;
E6F8      738 ;
E6F8      739 ; Reads a base10 value less than 256 into X-reg.
E6F8      740 ;
E6F8 20 67 DD 741 GETBYT jsr FRMNUM
E6FB      742 ;
E6FB      743 ;
E6FB 20 08 E1 744 CONVINT jsr AYPOSINT
E6FE      745 ;
E6FE A6 A0     746 ldx VARPTR
E700 D0 F0     747 bne DO.IQ.ER
E702      748 ;
E702 A6 A1     749 ldx VARPTR+1
E704      750 ;
E704 4C B7 00 751 jmp CHRGOT
E707      752 ;
E707      753 ;
E707 20 DC E6 754 FVAL jsr GETSTRLN
E70A D0 03     755 bne >6
E70C      756 ;
E70C 4C 4E E8 757 jmp ZEROFAC
E70F      758 ;
E70F A6 B8     759 ^6 ldx TXTPTR
E711 A4 B9     760 ldy TXTPTR+1
E713      761 ;
E713 86 AD     762 stx STRING2
E715 84 AE     763 sty STRING2+1
E717      764 ;
E717 A6 5E     765 ldx INDEX
E719 86 B8     766 stx TXTPTR
E71B      767 ;
E71B 18        768 clc
E71C      769 ;
E71C 65 5E     770 adc INDEX
E71E 85 60     771 sta DEST
E720      772 ;
E720 A6 5F     773 ldx INDEX+1
E722 86 B9     774 stx TXTPTR+1
E724      775 ;
E724 90 01     776 bcc >7
E726      777 ;
E726 E8        778 inx
E727      779 ;
E727 86 61     780 ^7 stx DEST+1
E729      781 ;
E729 A0 00     782 ldy #ZERO
E72B      783 ;
E72B B1 60     784 lda (DEST),Y
E72D 48        785 pha
E72E      786 ;
E72E      787 ;
E72E      788 ; The following would cause a problem if DEST (i.e. HIMEM)
E72E      789 ; equals 0xBFFF. Since HIMEM is 0xBE00 or less in DOS
E72E      790 ; 4.5.08, the following code will not cause a problem.
E72E      791 ;
E72E A9 00     792 lda #ZERO

```

```

E730 91 60      793      sta (DEST),Y
E732           794      ;
E732 20 B7 00   795      jsr CHRGOT
E735 20 4A EC   796      jsr GETINT
E738           797      ;
E738 A0 00      798      ldy #ZERO
E73A           799      ;
E73A 68         800      pla
E73B 91 60      801      sta (DEST),Y
E73D           802      ;
E73D           803      ;
E73D A6 AD      804  STRCOPY  ldx STRING2
E73F A4 AE      805      ldy STRING2+1
E741           806      ;
E741 86 B8      807      stx TXTPTR
E743 84 B9      808      sty TXTPTR+1
E745           809      ;
E745 60         810      rts
E746           811      ;
E746           812      ;
E746 20 67 DD   813  GETASNUM jsr FRMNUM
E749 20 52 E7   814      jsr GETADDR
E74C           815      ;
E74C           816      ;
E74C 20 BE DE   817  COMBYTE  jsr CHKCOM
E74F           818      ;
E74F 4C F8 E6   819      jmp GETBYT
E752           820      ;
E752           821      ;
E752 A5 9D      822  GETADDR  lda FACEXP
E754 C9 91      823      cmp #$91      ; less than 32768?
E756 B0 9A      824      bcs DO.IQ.ER
E758           825      ;
E758 20 F2 EB   826      jsr FP2INT
E75B           827      ;
E75B A5 A0      828      lda VARPTR
E75D A4 A1      829      ldy VARPTR+1
E75F           830      ;
E75F 84 50      831      sty ACL
E761 85 51      832      sta ACH
E763           833      ;
E763 60         834      rts
E764           835      ;
E764           836      ;
E764 A5 50      837  FPEEK    lda ACL
E766 48         838      pha
E767           839      ;
E767 A5 51      840      lda ACH
E769 48         841      pha
E76A           842      ;
E76A 20 52 E7   843      jsr GETADDR
E76D           844      ;
E76D A0 00      845      ldy #ZERO
E76F           846      ;
E76F B1 50      847      lda (ACL),Y
E771 A8         848      tay
E772           849      ;
E772 68         850      pla
E773 85 51      851      sta ACH
E775           852      ;
E775 68         853      pla

```

```

E776 85 50      854      sta ACL
E778           855      ;
E778 4C 01 E3   856      jmp SNGFLT
E77B           857      ;
E77B           858      ;
E77B 20 46 E7   859 BPOKE      jsr GETASNUM
E77E 8A         860      txa
E77F           861      ;
E77F A0 00      862      ldy #ZERO
E781           863      ;
E781 91 50      864      sta (ACL),Y
E783           865      ;
E783 60         866      rts
E784           867      ;
E784           868      ;
E784 20 46 E7   869 BWAIT      jsr GETASNUM
E787 86 85      870      stx FORPNT
E789           871      ;
E789 A2 00      872      ldx #ZERO
E78B           873      ;
E78B 20 B7 00   874      jsr CHRGOT
E78E F0 03      875      beq >8
E790           876      ;
E790 20 4C E7   877      jsr COMBYTE
E793           878      ;
E793 86 86      879 ^8      stx FORPNT+1
E795           880      ;
E795 A0 00      881      ldy #ZERO
E797           882      ;
E797 B1 50      883 ^9      lda (ACL),Y
E799 45 86      884      eor FORPNT+1
E79B           885      ;
E79B 25 85      886      and FORPNT
E79D F0 F8      887      beq <9
E79F           888      ;
E79F 60         889 RTN.E7.9 rts
E7A0           890      ;
E7A0           891      ;
E7A0 A9 57      892 FADDDHALF lda #FP0.5      ; FP 0.5
E7A2 A0 EE      893      ldy /FP0.5
E7A4           894      ;
E7A4 4C BE E7   895      jmp FADD
E7A7           896      ;
E7A7           897      ;
E7A7           898      ; FSUB routine entry point. Calculate ARG + (-FAC) -> FAC.
E7A7           899      ; Make FAC negative and recalculate FACSIGN and XORSIGN.
E7A7           900      ;
E7A7 20 E3 E9   901 FSUB      jsr LOADARG
E7AA           902      ;
E7AA           903 OMINUS:
E7AA A5 A2      904 FSUB2      lda FACSIGN
E7AC 49 FF      905      eor #NEGONE
E7AE 85 A2      906      sta FACSIGN
E7B0           907      ;
E7B0 45 AA      908      eor ARGSIGN
E7B2 85 AB      909      sta XORSIGN
E7B4           910      ;
E7B4 A5 9D      911      lda FACEXP
E7B6           912      ;
E7B6 4C C1 E7   913      jmp OPLUS
E7B9           914      ;

```



```

E7B9      915 ;
E7B9      916 ; Using the difference of ARG and FAC which is less than
E7B9      917 ; -7, shift either ARG or FAC one or more bytes to the
E7B9      918 ; right. Whatever is left over, shift that number of bits
E7B9      919 ; to the right. Then continue with the arithmetic.
E7B9      920 ;
E7B9 20 F0 E8 921 ^1      jsr SHFTBYT
E7BC 90 3C      922          bcc >5          ; always taken
E7BE      923 ;
E7BE      924 ;
E7BE      925 ; FADD routine entry point. Calculate ARG + FAC -> FAC.
E7BE      926 ;
E7BE 20 E3 E9 927 FADD      jsr LOADARG
E7C1      928 ;
E7C1      929 OPLUS:
E7C1 D0 03      930 FADD2      bne >2
E7C3      931 ;
E7C3 4C 53 EB 932          jmp COPYA2F
E7C6      933 ;
E7C6      934 ;
E7C6      935 ; Initially, save a copy of FACGUARD and set X-reg to 0xA5.
E7C6      936 ; Load A-reg with ARGEXP.
E7C6      937 ;
E7C6 A6 AC      938 ^2      ldw FACGUARD
E7C8 86 92      939          stw ARGGUARD
E7CA      940 ;
E7CA A2 A5      941          ldw #ARGEXP
E7CC A5 A5      942          lda ARGEXP
E7CE      943 ;
E7CE      944 ;
E7CE      945 ; Test A-reg and return if zero. Subtract FACEXP and
E7CE      946 ; continue with arithmetic if equal. Branch if FAC > ARG.
E7CE      947 ;
E7CE A8      948 FADD3      tay
E7CF F0 CE      949          beq RTN.E7.9
E7D1      950 ;
E7D1 38      951          sec
E7D2      952 ;
E7D2 E5 9D      953          sbc FACEXP
E7D4 F0 24      954          beq >5
E7D6      955 ;
E7D6 90 12      956          bcc >3
E7D8      957 ;
E7D8      958 ;
E7D8      959 ; ARG > FAC so make 2's compliment of difference, set
E7D8      960 ; FACEXP equal to ARGEXP and FACSIGN equal to ARGSIGN.
E7D8      961 ; Initialize ARGGUARD to zero and set X-reg to 0x9D.
E7D8      962 ;
E7D8 84 9D      963          sty FACEXP
E7DA      964 ;
E7DA A4 AA      965          ldw ARGSIGN
E7DC 84 A2      966          sty FACSIGN
E7DE      967 ;
E7DE 49 FF      968          eor #NEGONE
E7E0 69 00      969          adc #ZERO
E7E2      970 ;
E7E2 A0 00      971          ldw #ZERO
E7E4 84 92      972          sty ARGGUARD
E7E6      973 ;
E7E6 A2 9D      974          ldw #FACEXP
E7E8 D0 04      975          bne >4          ; always taken

```

```
E7EA          976 ;
E7EA          977 ;
E7EA          978 ; FAC > ARG so initialize FACGUARD to zero since X-reg is
E7EA          979 ; already set to 0xA5 and SHFTBYT or SHFTBIT will shift
E7EA          980 ; ARG to the right. ARG has no guard byte. Further
E7EA          981 ; improvements could utilize value at 0x66 or GUARD2 in
E7EA          982 ; order to initialize FACGUARD.
E7EA          983 ;
E7EA A0 00     984 ^3      ldy #ZERO
E7EC 84 AC     985          sty FACGUARD
E7EE          986 ;
E7EE          987 ;
E7EE          988 ; If exponent difference is less than -7, branch.
E7EE          989 ; Otherwise, set Y-reg to difference, set A-reg to
E7EE          990 ; FACGUARD, and shift ARG or FAC that many bits right.
E7EE          991 ;
E7EE C9 F9     992 ^4      cmp #!-7
E7F0 30 C7     993          bmi <1
E7F2          994 ;
E7F2 A8        995          tay
E7F3          996 ;
E7F3 A5 AC     997          lda FACGUARD
E7F5          998 ;
E7F5 56 01     999          lsr MULMANT-OFFSET-0,X
E7F7          1000 ;
E7F7 20 07 E9 1001          jsr SHFTBIT
E7FA          1002 ;
E7FA          1003 ;
E7FA          1004          icl "E8.L"
```

LLOAD E8.L,A\$4000

```

E7FA      1          ttl "ROM Source Code, E8.L"
E7FA      2      ;
E7FA      3      ;
E7FA      4      ; E8.L
E7FA      5      ;
E7FA      6      ;
E7FA      7      ; If XORSIGN is positive, branch.  Set Y-reg to 0x9D.
E7FA      8      ;
E7FA 24 AB      9      ^5          bit XORSIGN
E7FC 10 57     10          bpl >8
E7FE      11      ;
E7FE A0 9D     12          ldy #FACEXP
E800      13      ;
E800      14      ;
E800      15      ; If FAC > ARG, X-reg = 0xA5, so branch.
E800      16      ;
E800 E0 A5     17          cpx #ARGEXP
E802 F0 02     18          beq >6
E804      19      ;
E804      20      ;
E804      21      ; ARG > FAC, X-reg = 0x9D, so set Y-reg to 0xA5.
E804      22      ;
E804 A0 A5     23          ldy #ARGEXP
E806      24      ;
E806      25      ;
E806      26      ; Reg-A contains FACGUARD from SHFTBIT routine.  Make A-reg
E806      27      ; negative and add in ARGGUARD.  ARGGUARD is equal to
E806      28      ; FACGUARD if FAC > ARG or zero if ARG > FAC.  If FAC >
E806      29      ; ARG, then FACGUARD is zero before ARG is shifted to the
E806      30      ; right.  Then subtract the mantissas and save to FAC.
E806      31      ; This subtraction will always leave C-flag set.
E806      32      ;
E806 38        33      ^6          sec
E807      34      ;
E807 49 FF     35          eor #NEGONE
E809 65 92     36          adc ARGGUARD
E80B 85 AC     37          sta FACGUARD
E80D      38      ;
E80D B9 04 00  39          lda FACMANT-FACEXP-3,Y
E810 F5 04     40          sbc ARGMANT-ARGEXP-3,X
E812 85 A1     41          sta FACMANT+3
E814      42      ;
E814 B9 03 00  43          lda FACMANT-FACEXP-2,Y
E817 F5 03     44          sbc ARGMANT-ARGEXP-2,X
E819 85 A0     45          sta FACMANT+2
E81B      46      ;
E81B B9 02 00  47          lda FACMANT-FACEXP-1,Y
E81E F5 02     48          sbc ARGMANT-ARGEXP-1,X
E820 85 9F     49          sta FACMANT+1
E822      50      ;
E822 B9 01 00  51          lda FACMANT-FACEXP-0,Y
E825 F5 01     52          sbc ARGMANT-ARGEXP-0,X
E827 85 9E     53          sta FACMANT
E829      54      ;
E829      55      ;
E829      56      ; If C-flag is clear, make a 2's compliment of FAC.
E829      57      ;
E829 B0 03     58      CFG2COMP bcs NORMFAC1
E82B      59      ;
E82B 20 9E E8  60          jsr FACSCOMP

```

```

E82E      61 ;
E82E      62 ;
E82E      63 ; Initialize the guard byte in Y-reg and the number of bits
E82E      64 ; shifted in A-reg, both to zero. Initialize C-flag.
E82E      65 ;
E82E A0 00 66 NORMFAC1 ldy #ZERO
E830 98    67          tya
E831      68 ;
E831 18    69          clc
E832      70 ;
E832      71 ;
E832      72 ; If FACMANT is not zero, branch. Otherwise, shift all
E832      73 ; bytes up to FACMANT setting the guard byte to zero.
E832      74 ; Can only do this four times at most; set FAC to zero.
E832      75 ;
E832 A6 9E 76 ^7      ldx FACMANT
E834 D0 4A 77          bne >1
E836      78 ;
E836 A6 9F 79          ldx FACMANT+1
E838 86 9E 80          stx FACMANT
E83A      81 ;
E83A A6 A0 82          ldx FACMANT+2
E83C 86 9F 83          stx FACMANT+1
E83E      84 ;
E83E A6 A1 85          ldx FACMANT+3
E840 86 A0 86          stx FACMANT+2
E842      87 ;
E842 A6 AC 88          ldx FACGUARD
E844 86 A1 89          stx FACMANT+3
E846      90 ;
E846 84 AC 91          sty FACGUARD
E848      92 ;
E848 69 08 93          adc #BYTEBITS
E84A C9 20 94          cmp #MANTBITS
E84C D0 E4 95          bne <7
E84E      96 ;
E84E      97 ;
E84E      98 ; Clear FACEXP and FACSIGN to zero.
E84E      99 ;
E84E A9 00 100 ZEROFAC  lda #ZERO
E850      101 ;
E850 85 9D 102 ZEROFAC2 sta FACEXP
E852 85 A2 103          sta FACSIGN
E854      104 ;
E854 60    105          rts
E855      106 ;
E855      107 ;
E855      108 ; Add ARGGUARD and FACGUARD and the FAC and ARG mantissas.
E855      109 ; Branch if final mantissa needs to be normalized.
E855      110 ;
E855 65 92 111 ^8      adc ARGGUARD
E857 85 AC 112          sta FACGUARD
E859      113 ;
E859 A5 A1 114          lda FACMANT+3
E85B 65 A9 115          adc ARGMANT+3
E85D 85 A1 116          sta FACMANT+3
E85F      117 ;
E85F A5 A0 118          lda FACMANT+2
E861 65 A8 119          adc ARGMANT+2
E863 85 A0 120          sta FACMANT+2
E865      121 ;

```

```

E865 A5 9F      122      lda FACMANT+1
E867 65 A7      123      adc ARGMANT+1
E869 85 9F      124      sta FACMANT+1
E86B           125      ;
E86B A5 9E      126      lda FACMANT
E86D 65 A6      127      adc ARGMANT
E86F 85 9E      128      sta FACMANT
E871           129      ;
E871 B0 1C      130      bcs NORMFAC6          ; accelerate code
E873           131      ;
E873 60         132      rts
E874           133      ;
E874           134      ;
E874           135      ; Shift FAC mantissa left and increment A-reg each time
E874           136      ; using bits from FACGUARD.
E874           137      ;
E874 69 01      138      ^9      adc #1
E876           139      ;
E876 06 AC      140      asl FACGUARD
E878           141      ;
E878 26 A1      142      rol FACMANT+3
E87A 26 A0      143      rol FACMANT+2
E87C 26 9F      144      rol FACMANT+1
E87E 26 9E      145      rol FACMANT
E880           146      ;
E880           147      ;
E880           148      ; A-reg = 0, branch if X-reg (or FACMANT) < 0x80.  FACMANT
E880           149      ; needs to have its MSB set.
E880           150      ;
E880 10 F2      151      ^1      bpl <9
E882           152      ;
E882           153      ;
E882           154      ; Recalculate FAC exponent using content of A-reg and 2's
E882           155      ; compliment.
E882           156      ;
E882 38         157      sec
E883           158      ;
E883 E5 9D      159      sbc FACEXP
E885 B0 C7      160      bcs ZEROFAC
E887           161      ;
E887 49 FF      162      eor #NEGONE
E889           163      ;
E889 69 01      164      adc #1
E88B 85 9D      165      sta FACEXP
E88D 90 0E      166      bcc >2          ; remove NORMFAC5
E88F           167      ;
E88F E6 9D      168      NORMFAC6 inc FACEXP
E891 F0 42      169      beq PRMSG06          ; Overflow error
E893           170      ;
E893 66 9E      171      ror FACMANT
E895 66 9F      172      ror FACMANT+1
E897 66 A0      173      ror FACMANT+2
E899 66 A1      174      ror FACMANT+3
E89B           175      ;
E89B 66 AC      176      ror FACGUARD
E89D           177      ;
E89D 60         178      ^2      rts
E89E           179      ;
E89E           180      ;
E89E           181      ; Make 2's compliment of FAC sign and mantissa.
E89E           182      ;

```

```

E89E A5 A2      183  FACSCOMP lda FACSIGN
E8A0 49 FF      184                eor #NEGONE
E8A2 85 A2      185                sta FACSIGN
E8A4           186  ;
E8A4           187  ;
E8A4           188  ; Make 2's compliment of FAC mantissa.
E8A4           189  ;
E8A4 A5 9E      190  FACMCOMP lda FACMANT
E8A6 49 FF      191                eor #NEGONE
E8A8 85 9E      192                sta FACMANT
E8AA           193  ;
E8AA A5 9F      194                lda FACMANT+1
E8AC 49 FF      195                eor #NEGONE
E8AE 85 9F      196                sta FACMANT+1
E8B0           197  ;
E8B0 A5 A0      198                lda FACMANT+2
E8B2 49 FF      199                eor #NEGONE
E8B4 85 A0      200                sta FACMANT+2
E8B6           201  ;
E8B6 A5 A1      202                lda FACMANT+3
E8B8 49 FF      203                eor #NEGONE
E8BA 85 A1      204                sta FACMANT+3
E8BC           205  ;
E8BC A5 AC      206                lda FACGUARD
E8BE 49 FF      207                eor #NEGONE
E8C0 85 AC      208                sta FACGUARD
E8C2           209  ;
E8C2 E6 AC      210                inc FACGUARD
E8C4 D0 0E      211                bne >3
E8C6           212  ;
E8C6           213  ;
E8C6           214  ; Increment FAC mantissa.
E8C6           215  ;
E8C6 E6 A1      216  FACMINC  inc FACMANT+3
E8C8 D0 0A      217                bne >3
E8CA           218  ;
E8CA E6 A0      219                inc FACMANT+2
E8CC D0 06      220                bne >3
E8CE           221  ;
E8CE E6 9F      222                inc FACMANT+1
E8D0 D0 02      223                bne >3
E8D2           224  ;
E8D2 E6 9E      225                inc FACMANT
E8D4           226  ;
E8D4 60         227  ^3      rts
E8D5           228  ;
E8D5           229  ;
E8D5 A2 45      230  PRMSG06 ldx #MSG06-MESGS  ; Overflow error
E8D7           231  ;
E8D7 4C 12 D4   232                jmp FSCRN
E8DA           233  ;
E8DA           234  ;
E8DA           235  ; Called by FMULT when A-reg is zero. OFFSET is set to
E8DA           236  ; MULMANT-1 in order to show that the bytes of MULMANT are
E8DA           237  ; being shifted right. No other routine calls SHFTFMUL.
E8DA           238  ;
E8DA A2 61      239  SHFTFMUL ldx #MULMANT-1
E8DC           240  ;
E8DC B4 04      241  ^1      ldy MULMANT-OFFSET-3,X
E8DE 84 AC      242                sty FACGUARD
E8E0           243  ;

```

```

E8E0 B4 03      244      ldy MULMANT-OFFSET-2,X
E8E2 94 04      245      sty MULMANT-OFFSET-3,X
E8E4            246      ;
E8E4 B4 02      247      ldy MULMANT-OFFSET-1,X
E8E6 94 03      248      sty MULMANT-OFFSET-2,X
E8E8            249      ;
E8E8 B4 01      250      ldy MULMANT-OFFSET-0,X
E8EA 94 02      251      sty MULMANT-OFFSET-1,X
E8EC            252      ;
E8EC A4 A4      253      ldy SAVARGEX
E8EE 94 01      254      sty MULMANT-OFFSET-0,X
E8F0            255      ;
E8F0            256      ;
E8F0            257      ; Show that eight bits have been moved to the right.
E8F0            258      ;
E8F0 69 08      259      SHFTBYT  adc #BYTEBITS
E8F2 30 E8      260            bmi <1
E8F4            261      ;
E8F4 F0 E6      262            beq <1
E8F6            263      ;
E8F6            264      ;
E8F6            265      ; Calculate the remaining bits to be moved to the right.
E8F6            266      ;
E8F6 E9 08      267            sbc #BYTEBITS
E8F8 A8          268            tay
E8F9            269      ;
E8F9 A5 AC      270            lda FACGUARD
E8FB            271      ;
E8FB B0 14      272            bcs >4                ; escape for FMULT
E8FD            273      ;
E8FD 16 01      274      ^2      asl MULMANT-OFFSET-0,X
E8FF 90 02      275            bcc >3
E901            276      ;
E901 F6 01      277            inc MULMANT-OFFSET-0,X
E903            278      ;
E903 76 01      279      ^3      ror MULMANT-OFFSET-0,X
E905 76 01      280            ror MULMANT-OFFSET-0,X
E907            281      ;
E907 76 02      282      SHFTBIT  ror MULMANT-OFFSET-1,X
E909 76 03      283            ror MULMANT-OFFSET-2,X
E90B 76 04      284            ror MULMANT-OFFSET-3,X
E90D            285      ;
E90D 6A          286            ror                ; FACGUARD
E90E            287      ;
E90E C8          288            iny                ; if Y-reg is 0xFF, done
E90F D0 EC      289            bne <2
E911            290      ;
E911 18          291      ^4      clc
E912            292      ;
E912 60          293            rts
E913            294      ;
E913            295      ;
E913            296      ; Changed order of variables.  Removed first FP1.0 since
E913            297      ; there are duplicates in other locations and put FPLOGE
E913            298      ; here.
E913            299      ;
E913            300      ;FP1.0      hex 8100000000      ; FP 1.0
E913 7F 5E 5B    301      FPLOGE      hex 7F5E5BD8AA      ; FP LOG(e) = 0.434294481904
E916 D8 AA       302
E918 80 35 04    302      FPSQR0.5 hex 803504F334      ; FP 0.5^0.5 = 0.707106781
E91B F3 34

```

```

E91D 81 35 04    303  FPSQR2.0 hex 813504F334    ; FP 2.0^0.5 = 1.414213562
E920 F3 34
E922 80 80 00    304  FPN0.5   hex 8080000000    ; FP -0.5
E925 00 00
E927 80 31 72    305  FPLN2    hex 80317217F8    ; FP LN(2) = 0.693147181
E92A 17 F8
E92C              306  ;
E92C              307  ;
E92C 03          308  POLY.LOG hex 03              ; number of polynomials-1
E92D 7F 5E 56    309              hex 7F5E56CB79    ; FP 0.434255942 * X^7 +
E930 CB 79
E932 80 13 9B    310              hex 80139B0B64    ; FP 0.576584541 * X^5 +
E935 0B 64
E937 80 76 38    311              hex 8076389316    ; FP 0.961800759 * X^3 +
E93A 93 16
E93C 82 38 AA    312              hex 8238AA3B20    ; FP 2.88539007 * X
E93F 3B 20
E941              313  ;
E941              314  ;
E941              315  ; This was originally the FLOG entry. However, this code
E941              316  ; calculates the natural logarithm of FAC leaving the
E941              317  ; result in FAC. The actual FLOG routine is elsewhere.
E941              318  ;
E941 20 82 EB    319  FLN      jsr CKFACSGN
E944 F0 02        320              beq >1
E946              321  ;
E946 10 03        322              bpl >2              ; C-flag is clear
E948              323  ;
E948 4C 99 E1     324  ^1      jmp IQ.ERR
E94B              325  ;
E94B A5 9D        326  ^2      lda FACEXP              ; save unbiased exponent
E94D E9 7F        327              sbc #EXPBIAS-1
E94F 48           328              pha
E950              329  ;
E950 A9 80        330              lda #EXPBIAS              ; normalize from 0.5 to 1.0
E952 85 9D        331              sta FACEXP
E954              332  ;
E954 A9 18        333              lda #FPSQR0.5
E956 A0 E9        334              ldy /FPSQR0.5
E958              335  ;
E958 20 BE E7     336              jsr FADD
E95B              337  ;
E95B A9 1D        338              lda #FPSQR2.0
E95D A0 E9        339              ldy /FPSQR2.0
E95F              340  ;
E95F 20 66 EA     341              jsr FDIV
E962              342  ;
E962 A9 04        343              lda #FP1.0
E964 A0 EF        344              ldy /FP1.0
E966              345  ;
E966 20 A7 E7     346              jsr FSUB
E969              347  ;
E969 A9 2C        348              lda #POLY.LOG
E96B A0 E9        349              ldy /POLY.LOG
E96D              350  ;
E96D 20 5C EF     351              jsr POLYPROC
E970              352  ;
E970 A9 22        353              lda #FPN0.5
E972 A0 E9        354              ldy /FPN0.5
E974              355  ;
E974 20 BE E7     356              jsr FADD

```



```

E977          357 ;
E977 68          358      pla
E978 20 D5 EC    359      jsr BYT2FP
E97B          360 ;
E97B A9 27      361      lda #FPLN2          ; FP LN(2)
E97D A0 E9      362      ldy /FPLN2
E97F          363 ;
E97F          364 ;
E97F          365 ; FMULT routine entry point; product = ARG * FAC -> FAC.
E97F          366 ; The entry to OMULT is slightly modified.
E97F          367 ;
E97F 20 E3 E9    368 FMULT      jsr LOADARG
E982          369 ;
E982 F0 5B      370 OMULT      beq >3          ; value from FACEXP
E984          371 ;
E984 20 0E EA    372      jsr PROCEXP          ; process exponents
E987          373 ;
E987 A9 00      374      lda #ZERO          ; initialize MULMANT
E989 85 62      375      sta MULMANT
E98B 85 63      376      sta MULMANT+1
E98D 85 64      377      sta MULMANT+2
E98F 85 65      378      sta MULMANT+3
E991          379 ;
E991          380 ;
E991          381 ; Utilize any content in the FACGUARD from a previous
E991          382 ; calculation. This call multiplies the FAC by all 8 bits.
E991          383 ;
E991 A5 AC      384      lda FACGUARD
E993 20 AD E9    385      jsr TSTMULT
E996          386 ;
E996 A5 A1      387      lda FACMANT+3
E998 20 AD E9    388      jsr TSTMULT
E99B          389 ;
E99B A5 A0      390      lda FACMANT+2
E99D 20 AD E9    391      jsr TSTMULT
E9A0          392 ;
E9A0 A5 9F      393      lda FACMANT+1
E9A2 20 AD E9    394      jsr TSTMULT
E9A5          395 ;
E9A5 A5 9E      396      lda FACMANT          ; multiply FAC by all 8 bits
E9A7 20 B2 E9    397      jsr BYTMULT
E9AA          398 ;
E9AA 4C E6 EA    399      jmp COPYM2F          ; finish, copy MULMANT to FAC
E9AD          400 ;
E9AD          401 ;
E9AD          402 ; If zero, copy MULMANT bytes to the right; faster process.
E9AD          403 ;
E9AD D0 03      404 TSTMULT    bne BYTMULT
E9AF          405 ;
E9AF 4C DA E8    406      jmp SHFTFMUL          ; copy MULMANT bytes right
E9B2          407 ;
E9B2          408 ;
E9B2          409 ; Multiply FAC with bits in A-reg. Use A-reg as an 8-bit
E9B2          410 ; counter by shifting right and setting its MSB.
E9B2          411 ;
E9B2 4A          412 BYTMULT    lsr          ; put LSB into carry
E9B3          413 ;
E9B3 09 80      414      ora #MSBSET          ; setup 8-bit counter
E9B5          415 ;
E9B5 A8          416 ^1      tay          ; save for below; or PHA/PLA
E9B6          417 ;

```

```

E9B6 90 19      418      bcc >2                ; bypass addition
E9B8            419      ;
E9B8 18         420      clc                    ; add ARG to MULMANT
E9B9            421      ;
E9B9 A5 65      422      lda MULMANT+3
E9BB 65 A9      423      adc ARGMANT+3
E9BD 85 65      424      sta MULMANT+3
E9BF            425      ;
E9BF A5 64      426      lda MULMANT+2
E9C1 65 A8      427      adc ARGMANT+2
E9C3 85 64      428      sta MULMANT+2
E9C5            429      ;
E9C5 A5 63      430      lda MULMANT+1
E9C7 65 A7      431      adc ARGMANT+1
E9C9 85 63      432      sta MULMANT+1
E9CB            433      ;
E9CB A5 62      434      lda MULMANT
E9CD 65 A6      435      adc ARGMANT
E9CF 85 62      436      sta MULMANT
E9D1            437      ;
E9D1            438      ;
E9D1            439      ; Shift MULMANT right for next addition.
E9D1            440      ;
E9D1 66 62      441      ^2      ror MULMANT
E9D3 66 63      442      ror MULMANT+1
E9D5 66 64      443      ror MULMANT+2
E9D7 66 65      444      ror MULMANT+3
E9D9            445      ;
E9D9 66 AC      446      ror FACGUARD
E9DB            447      ;
E9DB            448      ;
E9DB            449      ; Recall A-reg, shift right, and put next bit into carry.
E9DB            450      ; When the 8-bit counter is complete, this byte is done.
E9DB            451      ;
E9DB 98         452      tya
E9DC            453      ;
E9DC 4A         454      lsr
E9DD D0 D6      455      bne <1
E9DF            456      ;
E9DF 60         457      ^3      rts
E9E0            458      ;
E9E0            459      ;
E9E0            460      dfs 3,ZERO                ; 3 bytes
E9E3            461      ;
E9E3            462      ;
E9E3            463      ; LOADARG routine entry point.
E9E3            464      ;
E9E3 85 5E      465      LOADARG sta INDEX
E9E5 84 5F      466      sty INDEX+1
E9E7            467      ;
E9E7 A0 04      468      ldy #4
E9E9            469      ;
E9E9 B1 5E      470      lda (INDEX),Y
E9EB 85 A9      471      sta ARGMANT+3
E9ED            472      ;
E9ED 88         473      dey
E9EE            474      ;
E9EE B1 5E      475      lda (INDEX),Y
E9F0 85 A8      476      sta ARGMANT+2
E9F2            477      ;
E9F2 88         478      dey

```

```
E9F3          479 ;
E9F3 B1 5E    480     lda (INDEX),Y
E9F5 85 A7    481     sta ARGMANT+1
E9F7          482 ;
E9F7 88       483     dey
E9F8          484 ;
E9F8 B1 5E    485     lda (INDEX),Y
E9FA 85 AA    486     sta ARGSIGN
E9FC          487 ;
E9FC 45 A2    488     eor FACSIGN
E9FE 85 AB    489     sta XORSIGN
EA00          490 ;
EA00 A5 AA    491     lda ARGSIGN
EA02 09 80    492     ora #EXPBIAS
EA04 85 A6    493     sta ARGMANT
EA06          494 ;
EA06 88       495     dey
EA07          496 ;
EA07 B1 5E    497     lda (INDEX),Y
EA09 85 A5    498     sta ARGEXP
EA0B          499 ;
EA0B A5 9D    500     lda FACEXP
EA0D          501 ;
EA0D 60       502     rts
EA0E          503 ;
EA0E          504 ;
EA0E          505     icl "EA.L"
```

LLOAD EA.L,A\$4000

```

EA0E      1          ttl "ROM Source Code, EA.L"
EA0E      2      ;
EA0E      3      ;
EA0E      4      ; EA.L
EA0E      5      ;
EA0E      6      ;
EA0E      7      ; Process exponents entry point; initialize FACSIGN.
EA0E      8      ;
EA0E A5 A5      9  PROCEXP  lda ARGEXP
EA10     10      ;
EA10 F0 1F     11  PROCEXP2 beq >3
EA12     12      ;
EA12 18       13      clc
EA13     14      ;
EA13 65 9D     15      adc FACEXP
EA15 90 04     16      bcc >1
EA17     17      ;
EA17 30 1D     18      bmi >4                ; Overflow error
EA19     19      ;
EA19 18       20      clc
EA1A     21      ;
EA1A 2C 00 00  22      bit *-*
EA1D     23      dfs !-2
EA1B     24      ;
EA1B 10 14     25      ^1      bpl >3
EA1D     26      ;
EA1D 69 80     27      adc #EXPBIAS
EA1F 85 9D     28      sta FACEXP
EA21 D0 03     29      bne >2
EA23     30      ;
EA23 85 A2     31      sta FACSIGN                ; accelerate code
EA25     32      ;
EA25 60       33      rts
EA26     34      ;
EA26 A5 AB     35      ^2      lda XORSIGN
EA28 85 A2     36      sta FACSIGN
EA2A     37      ;
EA2A 60       38      rts
EA2B     39      ;
EA2B     40      ;
EA2B A5 A2     41  CHKOVERR lda FACSIGN
EA2D 49 FF     42      eor #NEGONE
EA2F 30 05     43      bmi >4
EA31     44      ;
EA31     45      ;
EA31     46      ; Adjust stack pointer; clear FACEXP and FACSIGN to zero.
EA31     47      ;
EA31 68       48      ^3      pla
EA32 68       49      pla
EA33     50      ;
EA33 4C 4E E8  51      jmp ZEROFAC
EA36     52      ;
EA36     53      ;
EA36 4C D5 E8  54      ^4      jmp PRMSG06                ; Overflow error
EA39     55      ;
EA39     56      ;
EA39 20 63 EB  57  MULFAC10 jsr COPYF2A
EA3C     58      ;
EA3C AA       59      tax
EA3D F0 10     60      beq >5

```

```

EA3F      61 ;
EA3F 18    62      clc
EA40      63 ;
EA40 69 02 64      adc #2
EA42 B0 F2 65      bcs <4
EA44      66 ;
EA44 A2 00 67      ldx #ZERO
EA46 86 AB 68      stx XORSIGN
EA48      69 ;
EA48 20 CE E7 70      jsr FADD3
EA4B      71 ;
EA4B E6 9D 72      inc FACEXP
EA4D F0 E7 73      beq <4
EA4F      74 ;
EA4F 60    75      ^5      rts
EA50      76 ;
EA50      77 ;
EA50      78 ;FP10.0      hex 8420000000      ; FP 10.0
EA50 7D 4C CC 79 FP0.1      hex 7D4CCCCCD      ; FP 0.1
EA53 CC CD
EA55      80 ;
EA55      81 ;
EA55      82 ; This routine has been modified to multiply FAC by 0.10
EA55      83 ; rather than move FAC to ARG and divide ARG by 10.0.
EA55      84 ;
EA55      85      .if FPCDMOD
EA55      86 ;
EA55 A9 50 87 DIVFAC10 lda #FP0.1
EA57 A0 EA 88      ldy /FP0.1
EA59      89 ;
EA59 4C 7F E9 90      jmp FMULT
EA5C      91 ;
EA5C      92      .el
EA5C      93 ;
EA5C      94 DIVFAC10 jsr COPYF2A
EA5C      95 ;
EA5C      96      lda #FP10.0
EA5C      97      ldy /FP10.0
EA5C      98 ;
EA5C      99      .fi
EA5C     100 ;
EA5C A2 00 101 DIVFAC.S ldx #ZERO
EA5E     102 ;
EA5E 86 AB 103 DIVFAC      stx XORSIGN
EA60     104 ;
EA60 20 F9 EA 105      jsr LOADFAC
EA63     106 ;
EA63 4C 69 EA 107      jmp ODIVIDE
EA66     108 ;
EA66     109 ;
EA66     110 ; FDIV routine entry point; quotient = ARG / FAC -> FAC.
EA66     111 ; This routine has been rewritten and reorganized in order
EA66     112 ; to produce an 8-bit FACGUARD. ARGGUARD is utilized.
EA66     113 ; Extra space is available for the FLOG routine.
EA66     114 ;
EA66 20 E3 E9 115 FDIV      jsr LOADARG
EA69     116 ;
EA69 F0 71 117 ODIVIDE      beq >5      ; value from ARGEXP
EA6B     118 ;
EA6B 20 72 EB 119      jsr RNDUP      ; process FACGUARD for roundup
EA6E     120 ;

```

```

EA6E      121 ;
EA6E      122 ; Make 2's compliment of FACEXP and then process exponents.
EA6E      123 ;
EA6E 38    124 sec
EA6F      125 ;
EA6F A9 00 126 lda #ZERO
EA71 85 92 127 sta ARGGUARD ; not initialized in LOADARG
EA73      128 ;
EA73 E5 9D 129 sbc FACEXP
EA75 85 9D 130 sta FACEXP
EA77      131 ;
EA77 20 0E EA 132 jsr PROCEXP
EA7A      133 ;
EA7A E6 9D 134 inc FACEXP
EA7C F0 B8 135 beq <4
EA7E      136 ;
EA7E      137 ;
EA7E      138 ; Setup a 5-byte counter in X-reg and an 8-bit counter in
EA7E      139 ; A-reg. Test if FAC can be subtracted from ARG and rol
EA7E      140 ; carry into A-reg as an LSB. If carry is set, perform
EA7E      141 ; the subtraction. Reg-A contains the quotient byte.
EA7E      142 ;
EA7E A2 FB 143 ldx #!-5 ; byte counter for MULMANT
EA80      144 ;
EA80 A9 01 145 lda #1 ; bit counter for byte
EA82      146 ;
EA82 A4 A6 147 ^1 ldy ARGMANT
EA84 C4 9E 148 cpy FACMANT
EA86 D0 16 149 bne >2
EA88      150 ;
EA88 A4 A7 151 ldy ARGMANT+1
EA8A C4 9F 152 cpy FACMANT+1
EA8C D0 10 153 bne >2
EA8E      154 ;
EA8E A4 A8 155 ldy ARGMANT+2
EA90 C4 A0 156 cpy FACMANT+2
EA92 D0 0A 157 bne >2
EA94      158 ;
EA94 A4 A9 159 ldy ARGMANT+3
EA96 C4 A1 160 cpy FACMANT+3
EA98 D0 04 161 bne >2
EA9A      162 ;
EA9A A4 92 163 ldy ARGGUARD
EA9C C4 AC 164 cpy FACGUARD
EA9E      165 ;
EA9E 08    166 ^2 php ; save comparison in carry
EA9F      167 ;
EA9F 2A    168 rol ; save carry value
EAA0 90 07 169 bcc >3 ; counter expired?
EAA2      170 ;
EAA2      171 ;
EAA2      172 ; Bit counter expired; save the quotient byte in MULMANT.
EAA2      173 ;
EAA2 E8    174 inx
EAA3 F0 3E 175 beq >6
EAA5      176 ;
EAA5 95 66 177 sta MULMANT+4,X
EAA7      178 ;
EAA7 A9 01 179 lda #1 ; load bit counter again
EAA9      180 ;
EAA9      181 ;

```

```

EAA9          182 ; Recall carry status; if set, do subtraction.
EAA9          183 ;
EAA9 28       184 ^3      plp
EAAA 90 20    185          bcc >4
EAAC          186 ;
EAAC          187 ;
EAAC          188 ; Save A-reg, subtract FAC from ARC including their guard
EAAC          189 ; bytes, and recall A-reg.
EAAC          190 ;
EAAC A8       191          tay
EAAD          192 ;
EAAD A5 92    193          lda ARGGUARD
EAAF E5 AC    194          sbc FACGUARD
EAB1 85 92    195          sta ARGGUARD
EAB3          196 ;
EAB3 A5 A9    197          lda ARGMANT+3
EAB5 E5 A1    198          sbc FACMANT+3
EAB7 85 A9    199          sta ARGMANT+3
EAB9          200 ;
EAB9 A5 A8    201          lda ARGMANT+2
EABB E5 A0    202          sbc FACMANT+2
EABD 85 A8    203          sta ARGMANT+2
EABF          204 ;
EABF A5 A7    205          lda ARGMANT+1
EAC1 E5 9F    206          sbc FACMANT+1
EAC3 85 A7    207          sta ARGMANT+1
EAC5          208 ;
EAC5 A5 A6    209          lda ARGMANT
EAC7 E5 9E    210          sbc FACMANT
EAC9 85 A6    211          sta ARGMANT
EACB          212 ;
EACB 98       213          tya
EACC          214 ;
EACC          215 ;
EACC          216 ; Shift ARG one bit left.
EACC          217 ;
EACC 06 92    218 ^4      asl ARGGUARD
EACE          219 ;
EACE 26 A9    220          rol ARGMANT+3
EAD0 26 A8    221          rol ARGMANT+2
EAD2 26 A7    222          rol ARGMANT+1
EAD4 26 A6    223          rol ARGMANT
EAD6          224 ;
EAD6 B0 C6    225          bcs <2 ; capture carry to subtract
EAD8          226 ;
EAD8 30 A8    227          bmi <1 ; check if ARG > FAC
EADA 10 C2    228          bpl <2 ; ARG < FAC, capture carry
EADC          229 ;
EADC A2 89    230 ^5      ldx #MSG11-MESGS ; Division by Zero error
EADE          231 ;
EADE 4C 12 D4 232          jmp FSCRN
EAE1          233 ;
EAE1          234 ;
EAE1          235          dfs 2,ZERO ; 2 bytes
EAE3          236 ;
EAE3          237 ;
EAE3          238 ; Save A-reg to FACGUARD. Fall into COPYM2F.
EAE3          239 ;
EAE3 85 AC    240 ^6      sta FACGUARD
EAE5          241 ;
EAE5 28       242          plp ; fix stack pointer

```

```

EAE6          243 ;
EAE6          244 ;
EAE6          245 ; Copy MULMANT to FACMANT.
EAE6          246 ;
EAE6 A5 62    247 COPYM2F  lda MULMANT
EAE8 85 9E    248          sta FACMANT
EAEA          249 ;
EAEA A5 63    250          lda MULMANT+1
EAEC 85 9F    251          sta FACMANT+1
EAAE          252 ;
EAAE A5 64    253          lda MULMANT+2
EAF0 85 A0    254          sta FACMANT+2
EAF2          255 ;
EAF2 A5 65    256          lda MULMANT+3
EAF4 85 A1    257          sta FACMANT+3
EAF6          258 ;
EAF6 4C 2E E8 259          jmp NORMFAC1          ; finalize exponent
EAF9          260 ;
EAF9          261 ;
EAF9          262 ; LOADFAC routine entry point.
EAF9          263 ;
EAF9 85 5E    264 LOADFAC  sta INDEX
EAFB 84 5F    265          sty INDEX+1
EAFD          266 ;
EAFD A0 04    267          ldy #4
EAFF          268 ;
EAFF B1 5E    269          lda (INDEX),Y
EB01 85 A1    270          sta FACMANT+3
EB03          271 ;
EB03 88       272          dey
EB04          273 ;
EB04 B1 5E    274          lda (INDEX),Y
EB06 85 A0    275          sta FACMANT+2
EB08          276 ;
EB08 88       277          dey
EB09          278 ;
EB09 B1 5E    279          lda (INDEX),Y
EB0B 85 9F    280          sta FACMANT+1
EB0D          281 ;
EB0D 88       282          dey
EB0E          283 ;
EB0E B1 5E    284          lda (INDEX),Y
EB10 85 A2    285          sta FACSIGN
EB12          286 ;
EB12 09 80    287          ora #EXPBIAS
EB14 85 9E    288          sta FACMANT
EB16          289 ;
EB16 88       290          dey
EB17          291 ;
EB17 B1 5E    292          lda (INDEX),Y
EB19 85 9D    293          sta FACEXP
EB1B          294 ;
EB1B 84 AC    295          sty FACGUARD
EB1D          296 ;
EB1D 60       297          rts
EB1E          298 ;
EB1E          299 ;
EB1E A2 93    300 COPYF2T1 ldx #TEMP1
EB20          301 ;
EB20 2C 00 00 302          bit *-*
EB23          303          dfs !-2

```



```

EB21          304 ;
EB21 A2 98    305 COPYF2T2 ldx #TEMP2
EB23          306 ;
EB23 A0 00    307             ldy /TEMP1
EB25 F0 04    308             beq COPYFAC             ; always taken
EB27          309 ;
EB27 A6 85    310 COPYF2FR ldx FORPNT
EB29 A4 86    311             ldy FORPNT+1
EB2B          312 ;
EB2B          313 ;
EB2B          314 ; COPYFAC routine entry point.
EB2B          315 ;
EB2B 20 72 EB 316 COPYFAC jsr RNDUP             ; process FACGUARD for roundup
EB2E          317 ;
EB2E 86 5E    318             stx INDEX
EB30 84 5F    319             sty INDEX+1
EB32          320 ;
EB32 A0 04    321             ldy #4
EB34          322 ;
EB34 A5 A1    323             lda FACMANT+3
EB36 91 5E    324             sta (INDEX),Y
EB38          325 ;
EB38 88       326             dey
EB39          327 ;
EB39 A5 A0    328             lda FACMANT+2
EB3B 91 5E    329             sta (INDEX),Y
EB3D          330 ;
EB3D 88       331             dey
EB3E          332 ;
EB3E A5 9F    333             lda FACMANT+1
EB40 91 5E    334             sta (INDEX),Y
EB42          335 ;
EB42 88       336             dey
EB43          337 ;
EB43 A5 A2    338             lda FACSIGN
EB45 09 7F    339             ora #$7F
EB47          340 ;
EB47 25 9E    341             and FACMANT
EB49 91 5E    342             sta (INDEX),Y
EB4B          343 ;
EB4B 88       344             dey
EB4C          345 ;
EB4C A5 9D    346             lda FACEXP
EB4E 91 5E    347             sta (INDEX),Y
EB50          348 ;
EB50 84 AC    349             sty FACGUARD
EB52          350 ;
EB52 60       351             rts
EB53          352 ;
EB53          353 ;
EB53          354 ; Copy ARG to FAC routine entry point. This routine has
EB53          355 ; been expanded and it resides at 0xF67F.
EB53          356 ;
EB53 A5 AA    357 COPYA2F lda ARGSIGN
EB55          358 ;
EB55 85 A2    359 COPYA2F2 sta FACSIGN
EB57          360 ;
EB57 A5 A9    361             lda ARGMANT+3
EB59 85 A1    362             sta FACMANT+3
EB5B          363 ;
EB5B A5 A8    364             lda ARGMANT+2

```

```

EB5D 85 A0      365      sta FACMANT+2
EB5F           366      ;
EB5F 4C 8A F6   367      jmp COPYA2F3
EB62           368      ;
EB62           369      ;
EB62           370      dfs 1,ZERO          ; 1 bytes
EB63           371      ;
EB63           372      ;
EB63           373      ; Copy FAC to ARG routine entry point. Roundup FAC. This
EB63           374      ; routine has been expanded and it resides at 0xF69A.
EB63           375      ;
EB63 20 72 EB   376 COPYF2A jsr RNDUP          ; process FACGUARD for roundup
EB66           377      ;
EB66 A5 A2      378 COPYF2A2 lda FACSIGN
EB68 85 AA      379      sta ARGSIGN
EB6A           380      ;
EB6A A5 A1      381      lda FACMANT+3
EB6C 85 A9      382      sta ARGMANT+3
EB6E           383      ;
EB6E 4C 9B F6   384      jmp COPYF2A3
EB71           385      ;
EB71           386      ;
EB71           387      dfs 1,ZERO          ; 1 bytes
EB72           388      ;
EB72           389      ;
EB72           390      ; Roundup routine entry point; process FACGUARD.
EB72           391      ;
EB72 A5 9D      392 RNDUP      lda FACEXP
EB74 F0 19      393      beq RTN.EB.8
EB76           394      ;
EB76 06 AC      395      asl FACGUARD
EB78 90 15      396      bcc RTN.EB.8
EB7A           397      ;
EB7A 20 C6 E8   398 INCMANT jsr FACMINC          ; increment FAC mantissa
EB7D D0 10      399      bne RTN.EB.8
EB7F           400      ;
EB7F 4C 8F E8   401      jmp NORMFAC6
EB82           402      ;
EB82           403      ;
EB82           404      ; Test FAC for negative, zero, and positive.
EB82           405      ;
EB82 A5 9D      406 CKFACSGN lda FACEXP
EB84 F0 09      407      beq RTN.EB.8
EB86           408      ;
EB86 A5 A2      409 CKFACSG1 lda FACSIGN
EB88           410      ;
EB88 2A         411 CKFACSG2 rol
EB89           412      ;
EB89 A9 FF      413      lda #NEGONE
EB8B           414      ;
EB8B B0 02      415      bcs RTN.EB.8
EB8D           416      ;
EB8D A9 01      417      lda #1
EB8F           418      ;
EB8F 60         419 RTN.EB.8 rts
EB90           420      ;
EB90           421      ;
EB90           422      ; Routine to evaluate FACSIGN and FACMANT for < 0, = 0, or
EB90           423      ; > 0. Also, set C-flag for =< 0 and clear C-flag for > 0.
EB90           424      ;
EB90 20 82 EB   425 FSGN      jsr CKFACSGN

```

```

EB93          426 ;
EB93 85 9E    427 FLOAT    sta FACMANT
EB95          428 ;
EB95 A9 00    429          lda #ZERO
EB97 85 9F    430          sta FACMANT+1
EB99          431 ;
EB99 A2 88    432          ldx #$88
EB9B          433 ;
EB9B A5 9E    434 FLOAT2   lda FACMANT
EB9D 49 FF    435          eor #NEGONE
EB9F          436 ;
EB9F 2A       437          rol
EBA0          438 ;
EBA0 A9 00    439 FLOAT3   lda #ZERO
EBA2 85 A1    440          sta FACMANT+3
EBA4 85 A0    441          sta FACMANT+2
EBA6          442 ;
EBA6 86 9D    443          stx FACEXP
EBA8          444 ;
EBA8 85 AC    445          sta FACGUARD
EBAA 85 A2    446          sta FACSIGN
EBAC          447 ;
EBAC 4C 29 E8 448          jmp CFG2COMP
EBAF          449 ;
EBAF          450 ;
EBAF          451 ; Calculate ABS.
EBAF          452 ;
EBAF 46 A2    453 FABS      lsr FACSIGN
EBB1          454 ;
EBB1 60       455          rts
EBB2          456 ;
EBB2          457 ;
EBB2          458 ; Routine that compares FAC to packed number at A/Y-reg.
EBB2          459 ; Return A=1, 0, or -1 as A/Y-reg is <, =, > FAC.
EBB2          460 ;
EBB2 85 60    461 FPCOMP   sta DEST
EBB4 84 61    462 FPCOMP2  sty DEST+1
EBB6          463 ;
EBB6 A0 00    464          ldy #ZERO
EBB8          465 ;
EBB8 B1 60    466          lda (DEST),Y
EBBA          467 ;
EBBA C8       468          iny
EBBB          469 ;
EBBB AA       470          tax
EBBC F0 C4    471          beq CKFACSGN
EBBE          472 ;
EBBE B1 60    473          lda (DEST),Y
EBC0 45 A2    474          eor FACSIGN
EBC2 30 C2    475          bmi CKFACSG1
EBC4          476 ;
EBC4 E4 9D    477          cpx FACEXP
EBC6 D0 21    478          bne >1
EBC8          479 ;
EBC8 B1 60    480          lda (DEST),Y
EBCA 09 80    481          ora #EXPBIAS
EBCC          482 ;
EBCC C5 9E    483          cmp FACMANT
EBCE D0 19    484          bne >1
EBD0          485 ;
EBD0 C8       486          iny

```

```
EBD1      487 ;
EBD1 B1 60 488      lda (DEST),Y
EBD3 C5 9F 489      cmp FACMANT+1
EBD5 D0 12 490      bne >1
EBD7      491 ;
EBD7 C8    492      iny
EBD8      493 ;
EBD8 B1 60 494      lda (DEST),Y
EBDA C5 A0 495      cmp FACMANT+2
EBDC D0 0B 496      bne >1
EBDE      497 ;
EBDE C8    498      iny
EBDF      499 ;
EBDF A9 7F 500      lda #MSBCLR
EBE1 C5 AC 501      cmp FACGUARD
EBE3      502 ;
EBE3 B1 60 503      lda (DEST),Y
EBE5 E5 A1 504      sbc FACMANT+3
EBE7 F0 A6 505      beq RTN.EB.8
EBE9      506 ;
EBE9 A5 A2 507      ^1 lda FACSIGN
EBEB      508 ;
EBEB 90 9B 509      bcc CKFACSG2      ; accelerate code
EBED      510 ;
EBED 49 FF 511      eor #NEGONE
EBEF      512 ;
EBEF B0 97 513      bcs CKFACSG2      ; always taken
EBF1      514 ;
EBF1      515      dfs 1,ZERO      ; 1 byte
EBF2      516 ;
EBF2      517 ;
EBF2      518      icl "EC.L"
```

LLOAD EC.L,A\$4000

```

EBF2      1          ttl "ROM Source Code, EC.L"
EBF2      2      ;
EBF2      3      ;
EBF2      4      ; EC.L
EBF2      5      ;
EBF2      6      ;
EBF2      7      ; FP2INT quick integer fuction converts FAC to an integer
EBF2      8      ; value by shifting right with sign extension until all
EBF2      9      ; of the fractional bits are out. This routine assumes
EBF2     10      ; that the exponent is less than 32.
EBF2     11      ;
EBF2 A5 9D     12 FP2INT      lda FACEXP
EBF4 F0 4A     13          beq CLRMANT
EBF6      14      ;
EBF6 38       15          sec
EBF7      16      ;
EBF7 E9 A0     17          sbc #$A0
EBF9      18      ;
EBF9 24 A2     19          bit FACSIGN
EBFB 10 09     20          bpl >1
EBFD      21      ;
EBFD AA       22          tax
EBFE      23      ;
EBFE A9 FF     24          lda #NEGONE
EC00 85 A4     25          sta SAVARGEX
EC02      26      ;
EC02 20 A4 E8  27          jsr FACMCOMP
EC05      28      ;
EC05 8A       29          txa
EC06      30      ;
EC06 A2 9D     31      ^1      ldx #FACEXP
EC08      32      ;
EC08 C9 F9     33          cmp #!-7
EC0A 10 06     34          bpl >2
EC0C      35      ;
EC0C 20 F0 E8  36          jsr SHFTBYT
EC0F 84 A4     37          sty SAVARGEX
EC11      38      ;
EC11 60       39          rts
EC12      40      ;
EC12 A8       41      ^2      tay
EC13      42      ;
EC13 A5 A2     43          lda FACSIGN
EC15 29 80     44          and #MSBSET
EC17      45      ;
EC17 46 9E     46          lsr FACMANT
EC19      47      ;
EC19 05 9E     48          ora FACMANT
EC1B 85 9E     49          sta FACMANT
EC1D      50      ;
EC1D 20 07 E9  51          jsr SHFTBIT
EC20 84 A4     52          sty SAVARGEX
EC22      53      ;
EC22 60       54          rts
EC23      55      ;
EC23      56      ;
EC23      57      ; FINT function to convert FAC to integer form and then
EC23      58      ; refloat the integer to floating point. A faster
EC23      59      ; approach would be to simply clear the fractional bits by
EC23      60      ; setting them to zero.

```

```

EC23          61 ;
EC23 A5 9D    62 FINT      lda FACEXP
EC25 C9 A0    63          cmp #$A0
EC27 B0 20    64          bcs RTN.EC.4
EC29          65 ;
EC29 20 F2 EB 66          jsr FP2INT
EC2C 84 AC    67          sty FACGUARD
EC2E          68 ;
EC2E A5 A2    69          lda FACSIGN
EC30 84 A2    70          sty FACSIGN
EC32          71 ;
EC32 49 80    72          eor #EXPBIAS
EC34 2A       73          rol
EC35          74 ;
EC35 A9 A0    75          lda #$A0
EC37 85 9D    76          sta FACEXP
EC39          77 ;
EC39 A5 A1    78          lda FACMANT+3
EC3B 85 0D    79          sta CHARAC
EC3D          80 ;
EC3D 4C 29 E8 81          jmp CFG2COMP
EC40          82 ;
EC40          83 ;
EC40 85 9E    84 CLRMANT   sta FACMANT
EC42 85 9F    85          sta FACMANT+1
EC44 85 A0    86          sta FACMANT+2
EC46 85 A1    87          sta FACMANT+3
EC48          88 ;
EC48 A8       89          tay
EC49          90 ;
EC49 60       91 RTN.EC.4 rts
EC4A          92 ;
EC4A          93 ;
EC4A A0 00    94 GETINT   ldy #ZERO
EC4C A2 0A    95          ldx #10
EC4E          96 ;
EC4E 94 99    97 ^3      sty TEMP2+1,X
EC50          98 ;
EC50 CA       99          dex
EC51 10 FB   100          bpl <3
EC53          101 ;
EC53 90 6C   102          bcc >0
EC55          103 ;
EC55 C9 2D   104          cmp #'-'
EC57 D0 04   105          bne >4
EC59          106 ;
EC59 86 A3   107          stx SERLEN      ; minus sign location
EC5B          108 ;
EC5B F0 04   109          beq INTLOOP     ; always taken
EC5D          110 ;
EC5D C9 2B   111 ^4      cmp #'+'
EC5F D0 05   112          bne >7
EC61          113 ;
EC61 20 B1 00 114 INTLOOP  jsr CHRGET
EC64 90 5B   115          bcc >0
EC66          116 ;
EC66 C9 2E   117 ^7      cmp #'.'
EC68 F0 2E   118          beq >3
EC6A          119 ;
EC6A C9 45   120          cmp #'E'
EC6C D0 30   121          bne >4

```

```

EC6E          122 ;
EC6E 20 B1 00 123      jsr CHRGET
EC71 90 75     124      bcc >5                ; modified
EC73          125 ;
EC73 C9 C9     126      cmp #TKMINUS
EC75 F0 0E     127      beq >8
EC77          128 ;
EC77 C9 2D     129      cmp #'-'
EC79 F0 0A     130      beq >8
EC7B          131 ;
EC7B C9 C8     132      cmp #TKPLUS
EC7D F0 08     133      beq >9
EC7F          134 ;
EC7F C9 2B     135      cmp #'+'
EC81 F0 04     136      beq >9
EC83          137 ;
EC83 D0 07     138      bne >2                ; always taken
EC85          139 ;
EC85 66 9C     140      ^8      ror TEMP2+4
EC87          141 ;
EC87 20 B1 00 142      ^9      jsr CHRGET
EC8A 90 5C     143      bcc >5
EC8C          144 ;
EC8C 24 9C     145      ^2      bit TEMP2+4
EC8E 10 0E     146      bpl >4
EC90          147 ;
EC90 38        148      sec
EC91          149 ;
EC91 A9 00     150      lda #ZERO
EC93 E5 9A     151      sbc TEMP2+2
EC95          152 ;
EC95 4C A0 EC 153      jmp >6
EC98          154 ;
EC98 66 9B     155      ^3      ror TEMP2+3
EC9A          156 ;
EC9A 24 9B     157      bit TEMP2+3
EC9C 50 C3     158      bvc INTLOOP
EC9E          159 ;
EC9E A5 9A     160      ^4      lda TEMP2+2
ECA0          161 ;
ECA0 38        162      ^6      sec
ECA1          163 ;
ECA1 E5 99     164      sbc TEMP2+1
ECA3 85 9A     165      sta TEMP2+2
ECA5 F0 12     166      beq >1
ECA7          167 ;
ECA7 10 09     168      bpl >8
ECA9          169 ;
ECA9 20 55 EA 170      ^7      jsr DIVFAC10
ECAC          171 ;
ECAC E6 9A     172      inc TEMP2+2
ECAE D0 F9     173      bne <7
ECB0          174 ;
ECB0 F0 07     175      beq >1                ; always taken
ECB2          176 ;
ECB2 20 39 EA 177      ^8      jsr MULFAC10
ECB5          178 ;
ECB5 C6 9A     179      dec TEMP2+2
ECB7 D0 F9     180      bne <8
ECB9          181 ;
ECB9 A5 A3     182      ^1      lda SERLEN

```

```

ECBB 10 8C      183      bpl RTN.EC.4      ; accelerate code
ECBD           184      ;
ECBD 4C CD EE   185      jmp NEGFAC
ECC0           186      ;
ECC0           187      dfs 1,ZERO      ; 1 byte
ECC1           188      ;
ECC1 48         189      ^0      pha
ECC2           190      ;
ECC2 24 9B      191      bit TEMP2+3
ECC4 10 02      192      bpl >3
ECC6           193      ;
ECC6 E6 99      194      inc TEMP2+1
ECC8           195      ;
ECC8 20 39 EA   196      ^3      jsr MULFAC10
ECCB           197      ;
ECCB 38         198      sec
ECCC           199      ;
ECCC 68         200      pla
ECCD E9 30      201      sbc #'0'
ECCF           202      ;
ECCF 20 D5 EC   203      jsr BYT2FP
ECD2           204      ;
ECD2 4C 61 EC   205      jmp INTLOOP
ECD5           206      ;
ECD5           207      ;
ECD5 48         208      BYT2FP      pha
ECD6           209      ;
ECD6 20 63 EB   210      jsr COPYF2A
ECD9           211      ;
ECD9 68         212      pla
ECDA           213      ;
ECDA 20 93 EB   214      jsr FLOAT
ECDD           215      ;
ECDD A5 AA      216      lda ARGSIGN
ECDF 45 A2      217      eor FACSIGN
ECE1 85 AB      218      sta XORSIGN
ECE3           219      ;
ECE3 A6 9D      220      ldx FACEXP
ECE5           221      ;
ECE5 4C C1 E7   222      jmp FADD2
ECE8           223      ;
ECE8           224      ;
ECE8 A5 9A      225      ^5      lda TEMP2+2
ECEA C9 0A      226      cmp #10
ECEC 90 09      227      bcc >6
ECEEE          228      ;
ECEEE A9 64      229      lda #100
ECF0           230      ;
ECF0 24 9C      231      bit TEMP2+4
ECF2 30 11      232      bmi >7
ECF4           233      ;
ECF4 4C D5 E8   234      jmp PRMESG06
ECF7           235      ;
ECF7 0A         236      ^6      asl
ECF8 0A         237      asl
ECF9           238      ;
ECF9 18         239      clc
ECFA           240      ;
ECFA 65 9A      241      adc TEMP2+2
ECFC 0A         242      asl
ECFD           243      ;

```



```

ECFD 18          244      clc
ECFE             245      ;
ECFE A0 00       246      ldy #ZERO
ED00             247      ;
ED00 71 B8       248      adc (TXTPTR),Y
ED02             249      ;
ED02 38          250      sec
ED03             251      ;
ED03 E9 30       252      sbc #'0'
ED05             253      ;
ED05 85 9A       254      ^7      sta TEMP2+2
ED07             255      ;
ED07 4C 87 EC    256      jmp <9
ED0A             257      ;
ED0A             258      ;
ED0A 9B 3E BC    259      FP9.9E7  hex 9B3EBC1FFD      ; FP 999999999.9
ED0D 1F FD
ED0F 9E 6E 6B    260      FP9.9E8  hex 9E6E6B27FD      ; FP 999999999.0
ED12 27 FD
ED14 9E 6E 6B    261      FP1.0E9  hex 9E6E6B2800      ; FP 1.0E+09 = 1000000000.0
ED17 28 00
ED19             262      ;
ED19             263      ;
ED19 A9 54       264      PRTMSG19 lda #MSG19
ED1B A0 D3       265      ldy /MSG19
ED1D             266      ;
ED1D 20 3A DB    267      jsr STROUT
ED20             268      ;
ED20 A5 76       269      lda CURLIN+1
ED22 A6 75       270      ldx CURLIN
ED24             271      ;
ED24 85 9E       272      LINPRT  sta FACMANT
ED26 86 9F       273      stx FACMANT+1
ED28             274      ;
ED28 38          275      sec      ; normal FAC
ED29             276      ;
ED29 A2 90       277      ldx #$90      ; FACEXP
ED2B 20 A0 EB    278      jsr FLOAT3
ED2E 20 34 ED    279      jsr FPOUT
ED31             280      ;
ED31 4C 3A DB    281      jmp STROUT
ED34             282      ;
ED34             283      ;
ED34             284      ; Entry point for the PRINT and the STR$ statements. This
ED34             285      ; routine has been rewritten in order to add enhancements.
ED34             286      ; All numerical strings are printed from start of STACK.
ED34             287      ;
ED34 A0 00       288      FPOUT  ldy #ZERO
ED36 84 AD       289      sty SAVY
ED38             290      ;
ED38 A5 A2       291      lda FACSIGN
ED3A 10 07       292      bpl >1
ED3C             293      ;
ED3C 84 A2       294      sty FACSIGN      ; make FACSIGN positive
ED3E             295      ;
ED3E A9 2D       296      lda #'-'
ED40 20 4F EE    297      jsr SAV2STK
ED43             298      ;
ED43 A5 9D       299      ^1      lda FACEXP      ; is FAC zero
ED45 D0 05       300      bne >2
ED47             301      ;

```

```

ED47 A9 30      302      lda #'0'          ; yes, done
ED49            303      ;
ED49 4C 44 EE    304      jmp FPOTEXIT
ED4C            305      ;
ED4C C9 81      306      ^2      cmp #$81
ED4E B0 0A      307      bcs >3          ; FAC > 0x80
ED50            308      ;
ED50            309      ;
ED50            310      ; FAC < 0x81.  Multiply FAC by 1.0E+09.  Set counter to -9.
ED50            311      ;
ED50 A9 14      312      lda #FP1.0E9
ED52 A0 ED      313      ldy /FP1.0E9
ED54            314      ;
ED54 20 7F E9    315      jsr FMULT
ED57            316      ;
ED57 A9 F7      317      lda #!-9
ED59            318      ;
ED59 2C 00 00    319      bit *-*
ED5C            320      dfs !-2
ED5A            321      ;
ED5A            322      ;
ED5A            323      ; FAC > 0x80.  No pre-multiplying.  Set counter to 0.
ED5A            324      ;
ED5A A9 00      325      ^3      lda #ZERO
ED5C            326      ;
ED5C 85 99      327      sta COUNTER          ; multiply/divide counter
ED5E            328      ;
ED5E            329      ;
ED5E            330      ; Multiply FAC by 10 or divide FAC by 10 until FAC is in
ED5E            331      ; the range of 1.0E+08-1 to 1.0E+09-1.
ED5E            332      ;
ED5E A9 0F      333      ^4      lda #FP9.9E8          ; FP 999,999,999.0
ED60 A0 ED      334      ldy /FP9.9E8
ED62            335      ;
ED62 20 B2 EB    336      jsr FPCOMP
ED65 F0 1C      337      beq >8          ; done, bypass ADD0.5
ED67            338      ;
ED67 10 10      339      bpl >6          ; FAC > FP9.9E8
ED69            340      ;
ED69 A9 0A      341      ^5      lda #FP9.9E7          ; FP 99,999,999.9
ED6B A0 ED      342      ldy /FP9.9E7
ED6D            343      ;
ED6D 20 B2 EB    344      jsr FPCOMP
ED70 90 0E      345      bcc >7          ; FAC > FP9.9E7
ED72            346      ;
ED72 20 39 EA    347      jsr MULFAC10
ED75            348      ;
ED75 C6 99      349      dec COUNTER
ED77 D0 F0      350      bne <5          ; always taken
ED79            351      ;
ED79 20 55 EA    352      ^6      jsr DIVFAC10
ED7C            353      ;
ED7C E6 99      354      inc COUNTER
ED7E D0 DE      355      bne <4          ; always taken
ED80            356      ;
ED80 20 A0 E7    357      ^7      jsr FADDHALF          ; a trivial roundup
ED83            358      ;
ED83            359      ;
ED83            360      ; Normalize the mantissa so that its exponent is 0xA0.
ED83            361      ; Initialize X-reg to display at least one whole digit.
ED83            362      ; Up to nine decimal numbers are displayed, then 'E' is

```

```

ED83      363 ; displayed. Calculate exponent value from COUNTER.
ED83      364 ;
ED83 20 F2 EB 365 ^8      jsr FP2INT          ; C-flag clear on exit
ED86      366 ;
ED86 A2 01   367      ldx #1
ED88      368 ;
ED88 A5 99   369      lda COUNTER
ED8A 69 0A   370      adc #10
ED8C 30 08   371      bmi >1                ; if decimal number
ED8E      372 ;
ED8E C9 0B   373      cmp #11
ED90 B0 05   374      bcs >2                ; if > 10 whole digits
ED92      375 ;
ED92      376 ;
ED92      377 ; Number of whole digits < 10, so decrement and display up
ED92      378 ; to 9 whole digits by forcing EXPCOUNT to be zero.
ED92      379 ;
ED92 AA      380      tax
ED93 CA      381      dex
ED94      382 ;
ED94 A9 02   383      lda #2
ED96      384 ;
ED96 38      385 ^1      sec
ED97      386 ;
ED97 E9 02   387 ^2      sbc #2
ED99      388 ;
ED99 86 99   389      stx COUNTER          ; whole digit counter
ED9B 85 9A   390      sta EXPCOUNT        ; exponent value
ED9D      391 ;
ED9D      392 ;
ED9D      393 ; If the whole digit counter is =< zero, write '0.'. The
ED9D      394 ; original code wrote ' '. Reg-X is 0xFF, 0x00, or > 0x00.
ED9D      395 ; The values of 0xFF and 0x00 are only of interest here.
ED9D      396 ;
ED9D CA      397      dex
ED9E 10 12   398      bpl >3
EDA0      399 ;
EDA0 A9 30   400      lda #'0'
EDA2 20 4F EE 401      jsr SAV2STK
EDA5      402 ;
EDA5 A9 2E   403      lda #'.'
EDA7 20 4F EE 404      jsr SAV2STK
EDAA      405 ;
EDAA      406 ;
EDAA      407 ; If COUNTER is still negative after an increment, write
EDAA      408 ; the tenths digit as zero.
EDAA      409 ;
EDAA E8      410      inx
EDAB F0 05   411      beq >3
EDAD      412 ;
EDAD A9 30   413      lda #'0'
EDAF 20 4F EE 414      jsr SAV2STK
EDB2      415 ;
EDB2      416 ;
EDB2      417 ; Digit counting loop. When X-reg is minus, minus values
EDB2      418 ; are added to FAC and X-reg is saved as a digit. When
EDB2      419 ; X-reg is plus, plus values are added to FAC and 10-X-reg
EDB2      420 ; is saved as a digit.
EDB2      421 ;
EDB2 A9 00   422 ^3      lda #ZERO                ; initial FPDECTBL index
EDB4 A2 80   423      ldx #MSBSET          ; initial digit counter

```

```

EDB6          424 ;
EDB6 48       425 ^4      pha                ; save FPDECTBL index
EDB7          426 ;
EDB7 A8       427      tay
EDB8          428 ;
EDB8 18       429 ^5      clc
EDB9          430 ;
EDB9 A5 A1    431      lda FACMANT+3
EDBB 79 5F EE 432      adc FPDECTBL+3,Y
EDBE 85 A1    433      sta FACMANT+3
EDC0          434 ;
EDC0 A5 A0    435      lda FACMANT+2
EDC2 79 5E EE 436      adc FPDECTBL+2,Y
EDC5 85 A0    437      sta FACMANT+2
EDC7          438 ;
EDC7 A5 9F    439      lda FACMANT+1
EDC9 79 5D EE 440      adc FPDECTBL+1,Y
EDCC 85 9F    441      sta FACMANT+1
EDCE          442 ;
EDCE A5 9E    443      lda FACMANT
EDD0 79 5C EE 444      adc FPDECTBL,Y
EDD3 85 9E    445      sta FACMANT
EDD5          446 ;
EDD5 E8       447      inx                ; increment digit counter
EDD6          448 ;
EDD6 B0 04    449      bcs >6                ; when adding - values or
                                         ; done adding + values
EDD8          450 ;
EDD8          451 ;
EDD8 10 DE    452      bpl <5                ; when adding + values
EDDA 30 02    453      bmi >7                ; done adding - values
EDDC          454 ;
EDDC 30 DA    455 ^6      bmi <5                ; not done adding - values
EDDE          456 ;
EDDE 8A       457 ^7      txa                ; recall digit counter
EDDF          458 ;
EDDF 90 04    459      bcc >8                ; when adding - values
EDE1          460 ;
EDE1 49 FF    461      eor #NEGONE            ; make negative
EDE3 69 0A    462      adc #10              ; subtract from 10
EDE5          463 ;
EDE5 69 2F    464 ^8      adc #'0'-1          ; make into ASCII digit
EDE7 29 7F    465      and #MSBCLR          ; save digit
EDE9 20 4F EE 466      jsr SAV2STK
EDEEC        467 ;
EDEEC C6 99   468      dec COUNTER            ; whole digit counter
EDEE D0 05    469      bne >9
EDF0          470 ;
EDF0          471 ;
EDF0          472 ; Done saving whole digits. Write decimal point. Process
EDF0          473 ; remaining table entries.
EDF0          474 ;
EDF0 A9 2E    475      lda #'.´
EDF2 20 4F EE 476      jsr SAV2STK
EDF5          477 ;
EDF5 8A       478 ^9      txa                ; recall digit counter
EDF6 29 80    479      and #MSBSET          ; extract the MSB
EDF8 49 80    480      eor #MSBSET          ; toggle the MSB
EDFA AA       481      tax
EDFB          482 ;
EDFB 68       483      pla                ; recall table index
EDFC 69 04    484      adc #4              ; point to next entry

```

```

EDFE          485 ;
EDFE C9 24     486      cmp #DECTBLEN
EE00 D0 B4     487      bne <4
EE02          488 ;
EE02          489 ;
EE02          490 ; Remove unnecessary zeros. Reg-Y valid from last SAV2STK.
EE02          491 ; Leave an ASCII period/zero if have single digit and 'E'.
EE02          492 ;
EE02 A2 07     493      ldx #7                      ; ASCII zero counter
EE04          494 ;
EE04 B9 00 01  495 ^1    lda STACK,Y
EE07          496 ;
EE07 88        497      dey
EE08          498 ;
EE08 CA        499      dex
EE09 D0 04     500      bne >2
EE0B          501 ;
EE0B A6 9A     502      ldx EXPCOUNT
EE0D D0 09     503      bne >3
EE0F          504 ;
EE0F C9 30     505 ^2    cmp #'0'
EE11 F0 F1     506      beq <1                      ; found ASCII zero
EE13          507 ;
EE13 C9 2E     508      cmp #'.'
EE15 F0 01     509      beq >3                      ; found ASCII period
EE17          510 ;
EE17 C8        511      iny
EE18          512 ;
EE18 C8        513 ^3    iny
EE19          514 ;
EE19 84 AD     515      sty SAVY                      ; insertion location
EE1B          516 ;
EE1B          517 ;
EE1B          518 ; If EXPCOUNT is zero, process is done. Save exponent
EE1B          519 ; value after 'E' and EXPCOUNT sign.
EE1B          520 ;
EE1B A6 9A     521      ldx EXPCOUNT
EE1D F0 28     522      beq >6
EE1F          523 ;
EE1F A9 45     524      lda #'E'
EE21 20 4F EE  525      jsr SAV2STK
EE24          526 ;
EE24 8A        527      txa                      ; recall EXPCOUNT
EE25 10 07     528      bpl >4
EE27          529 ;
EE27 49 FF     530      eor #NEGONE                  ; make 2's compliment
EE29 AA        531      tax
EE2A          532 ;
EE2A E8        533      inx
EE2B          534 ;
EE2B A9 2D     535      lda #'-'                      ; get ASCII minus
EE2D          536 ;
EE2D 2C 00 00  537      bit *-*
EE30          538      dfs !-2
EE2E          539 ;
EE2E A9 2B     540 ^4    lda #'+'                      ; get ASCII plus
EE30          541 ;
EE30 20 4F EE  542      jsr SAV2STK
EE33          543 ;
EE33          544 ;
EE33          545 ; Convert exponent value into base 10.

```

```

EE33          546 ;
EE33 8A       547      txa                ; recall exponent value
EE34          548 ;
EE34 A2 2F    549      ldx #'0'-1        ; init 10's ASCII digit
EE36          550 ;
EE36 38       551      sec
EE37          552 ;
EE37 E8       553      ^5      inx
EE38          554 ;
EE38 E9 0A    555      sbc #10            ; successive subtraction
EE3A B0 FB    556      bcs <5            ; more to subtract
EE3C          557 ;
EE3C 69 3A    558      adc #'0'+10       ; make ASCII 1's digit
EE3E 48       559      pha                ; save
EE3F          560 ;
EE3F 8A       561      txa                ; write 10's digit
EE40 20 4F EE 562      jsr SAV2STK
EE43          563 ;
EE43 68       564      pla                ; write 1's digit
EE44          565 ;
EE44 20 4F EE 566      FPOTEXIT jsr SAV2STK
EE47          567 ;
EE47 A9 00    568      ^6      lda #ZERO          ; terminate ASCII data
EE49 20 4F EE 569      jsr SAV2STK
EE4C          570 ;
EE4C          571 ;
EE4C          572 ; Return to caller with address of stack in A/Y-reg.
EE4C          573 ;
EE4C          574 ;      lda #STACK ; already zero
EE4C A0 01    575      ldy /STACK
EE4E          576 ;
EE4E 60       577      rts
EE4F          578 ;
EE4F          579 ;
EE4F A4 AD    580      SAV2STK ldy SAVY
EE51          581 ;
EE51 99 00 01 582      sta STACK,Y
EE54          583 ;
EE54 E6 AD    584      inc SAVY
EE56          585 ;
EE56 60       586      rts
EE57          587 ;
EE57          588 ;
EE57 80 00 00 589      FP0.5      hex 8000000000      ; FP 0.5
EE5A 00 00
EE5C          590 ;
EE5C FA 0A 1F 591      FPDECTBL hex FA0A1F00      ; -100,000,000
EE5F 00
EE60 00 98 96 592      hex 00989680      ; 10,000,000
EE63 80
EE64 FF F0 BD 593      hex FFF0BDC0      ; - 1,000,000
EE67 C0
EE68 00 01 86 594      hex 000186A0      ; 100,000
EE6B A0
EE6C FF FF D8 595      hex FFFFD8F0      ; - 10,000
EE6F F0
EE70 00 00 03 596      hex 000003E8      ; 1,000
EE73 E8
EE74 FF FF FF 597      hex FFFFFFFF9C      ; - 100
EE77 9C
EE78 00 00 00 598      hex 0000000A      ; 10

```

```

EE7B 0A
EE7C FF FF FF 599          hex FFFFFFFF          ; -          1
EE7F FF
EE80          600          ;
0024          601  DECTBLN equ *-FPDECTBL
EE80          602          ;
EE80          603          ;
EE80          604          dfs 1,ZERO          ; 1 byte
EE81          605          ;
EE81          606          ;
EE81          607          ; FSQR routine entry point; take square root of FAC.
EE81          608          ; Use the Newton-Raphson iteration method which is
EE81          609          ;  $R = [(N/X) + X] / 2$  until  $R = X$ , else  $X = R$ .
EE81          610          ; Calculate exponent of start initial value.
EE81          611          ;
EE81          612          .if FPCDMOD
EE81          613          ;
EE81 20 1E EB 614  FSQR      jsr COPYF2T1
EE84 F0 51      615          beq RTN.EE.D
EE86          616          ;
EE86 84 47      617          sty YREG
EE88          618          ;
EE88 18          619          clc
EE89          620          ;
EE89 49 80      621          eor #EXPBIAS
EE8B 10 01      622          bpl >1
EE8D          623          ;
EE8D 38          624          sec
EE8E          625          ;
EE8E 6A          626          ^1 ror
EE8F          627          ;
EE8F 18          628          clc
EE90          629          ;
EE90 69 80      630          adc #EXPBIAS
EE92 85 9D      631          sta FACEXP
EE94          632          ;
EE94 4C 6B F6   633          jmp FSQR2
EE97          634          ;
EE97          635          .el
EE97          636          ;
EE97          637          dfs 12,ZERO          ; 12 bytes
EE97          638          ;
EE97          639  FSQR      jsr COPYF2A
EE97          640          ;
EE97          641          lda #FP0.5          ; FP 0.5
EE97          642          ldy /FP0.5
EE97          643          ;
EE97          644          jsr LOADFAC
EE97          645          ;
EE97          646          .fi
EE97          647          ;
EE97          648          ;
EE97          649          ; FPWRT exponentiation routine entry point; take FEXP of
EE97          650          ; FLN( ARG ) * FAC leaving the result in FAC, or
EE97          651          ;  $ARG ^ FAC = FEXP( FLN( ARG ) * FAC )$ .
EE97          652          ; This is the ^ operator entry point for Applesoft.
EE97          653          ;
EE97          654  OPOWER:
EE97 F0 70      655  FPWRT      beq FEXP
EE99          656          ;
EE99 A5 A5      657          lda ARGEXP

```

```

EE9B F0 3B      658      beq >3                ; accelerate code
EE9D           659      ;
EE9D A2 8A      660      ldx #TEMP3
EE9F A0 00      661      ldy /TEMP3
EEA1 20 2B EB   662      jsr COPYFAC
EEA4           663      ;
EEA4 A5 AA      664      lda ARGSIGN
EEA6 10 0F      665      bpl >2
EEA8           666      ;
EEA8 20 23 EC   667      jsr FINT
EEAB           668      ;
EEAB A9 8A      669      lda #TEMP3
EEAD A0 00      670      ldy /TEMP3
EEAF           671      ;
EEAF 20 B2 EB   672      jsr FPCOMP
EEB2 D0 03      673      bne >2
EEB4           674      ;
EEB4 98         675      tya
EEB5           676      ;
EEB5 A4 0D      677      ldy CHARAC
EEB7           678      ;
EEB7 20 55 EB   679      ^2      jsr COPYA2F2
EEBA           680      ;
EEBA 98         681      tya                ; negative argument flag
EEBB 48         682      pha
EEBC           683      ;
EEBC 20 41 E9   684      jsr FLN
EEBF           685      ;
EEBF A9 8A      686      lda #TEMP3
EEC1 A0 00      687      ldy /TEMP3
EEC3           688      ;
EEC3 20 7F E9   689      jsr FMULT
EEC6 20 09 EF   690      jsr FEXP
EEC9           691      ;
EEC9 68         692      pla                ; recall flag
EECA           693      ;
EECA 4A         694      lsr
EECB 90 0A      695      bcc RTN.EE.D
EECD           696      ;
EECD           697      ;
EECD           698      ; This routine negates the value in FAC and this routine is
EECD           699      ; also the ">" operator entry point for Applesoft.
EECD           700      ;
EECD           701      OGT:
EECD A5 9D      702      NEGFAC      lda FACEXP
EECF F0 06      703      beq RTN.EE.D
EED1           704      ;
EED1 A5 A2      705      lda FACSIGN
EED3 49 FF      706      eor #NEGONE
EED5 85 A2      707      sta FACSIGN
EED7           708      ;
EED7 60         709      RTN.EE.D rts
EED8           710      ;
EED8           711      ;
EED8 4C 50 E8   712      ^3      jmp ZEROFAC2
EEDB           713      ;
EEDB           714      ;
EEDB 81 38 AA   715      FPINVLN2 hex 8138AA3B29      ; FP 1/ln(2) = 1.44269504
EEDE 3B 29      716      ;
EEEE0          717      POLY.EXP hex 07                ; number of polynomials-1

```



```

EEE1 71 34 58    718          hex 7134583E56          ; FP (LOG(2)^7)/8!
EEE4 3E 56
EEE6 74 16 7E    719          hex 74167EB31B          ; FP (LOG(2)^6)/7!
EEE9 B3 1B
EEEB 77 2F EE    720          hex 772FEEE385          ; FP (LOG(2)^5)/6!
EEEE E3 85
EEF0 7A 1D 84    721          hex 7A1D841C2A          ; FP (LOG(2)^4)/5!
EEF3 1C 2A
EEF5 7C 63 59    722          hex 7C6359580A          ; FP (LOG(2)^3)/4!
EEF8 58 0A
EEFA 7E 75 FD    723          hex 7E75FDE7C6          ; FP (LOG(2)^2)/3!
EEFD E7 C6
EEFF 80 31 72    724          hex 8031721810          ; FP LOG(2)/2!
EF02 18 10
EF04 81 00 00    725  FP1.0    hex 8100000000          ; FP 1.0
EF07 00 00
EF09              726 ;
EF09              727 ;
EF09              728 ; FEXP routine entry point; take e to the power of FAC
EF09              729 ; leaving the result in FAC, or FAC = e ^ FAC.
EF09              730 ;
EF09 A9 DB       731  FEXP      lda #FPINVLN2
EF0B A0 EE       732          ldy /FPINVLN2
EF0D              733 ;
EF0D 20 7F E9    734          jsr FMULT
EF10              735 ;
EF10 A5 AC       736          lda FACGUARD
EF12 69 50       737          adc #$50
EF14 90 03       738          bcc >4
EF16              739 ;
EF16 20 7A EB    740          jsr INCMANT
EF19              741 ;
EF19 85 92       742 ^4      sta ARGGUARD
EF1B              743 ;
EF1B 20 66 EB    744          jsr COPYF2A2
EF1E              745 ;
EF1E A5 9D       746          lda FACEXP
EF20 C9 88       747          cmp #$88
EF22 90 03       748          bcc >6
EF24              749 ;
EF24 20 2B EA    750 ^5      jsr CHKOVERR          ; check zero/overflow error
EF27              751 ;
EF27 20 23 EC    752 ^6      jsr FINT
EF2A              753 ;
EF2A 18          754          clc
EF2B              755 ;
EF2B A5 0D       756          lda CHARAC
EF2D 69 81       757          adc #$81
EF2F F0 F3       758          beq <5
EF31              759 ;
EF31 38          760          sec
EF32              761 ;
EF32 E9 01       762          sbc #1
EF34 48          763          pha
EF35              764 ;
EF35 A2 05       765          ldx #5
EF37              766 ;
EF37 B5 A5       767 ^7      lda ARGEXP,X
EF39 B4 9D       768          ldy FACEXP,X
EF3B              769 ;
EF3B 95 9D       770          sta FACEXP,X

```

```

EF3D 94 A5      771      sty ARGEXP,X
EF3F           772      ;
EF3F CA        773      dex
EF40 10 F5      774      bpl <7
EF42           775      ;
EF42 A5 92      776      lda ARGGUARD
EF44 85 AC      777      sta FACGUARD
EF46           778      ;
EF46 20 AA E7   779      jsr FSUB2
EF49 20 CD EE   780      jsr NEGFAC
EF4C           781      ;
EF4C A9 E0      782      lda #POLY.EXP
EF4E A0 EE      783      ldy /POLY.EXP
EF50 20 72 EF   784      jsr POLYNOM
EF53           785      ;
EF53 A9 00      786      lda #ZERO
EF55 85 AB      787      sta XORSIGN
EF57           788      ;
EF57 68         789      pla
EF58           790      ;
EF58 4C 10 EA   791      jmp PROCEXP2          ; accelerate code
EF5B           792      ;
EF5B           793      ;
EF5B           794      dfs 1,ZERO          ; 1 byte
EF5C           795      ;
EF5C           796      ;
EF5C           797      ; Polynomial processing. First byte is the number of FP
EF5C           798      ; polynomials that must be processed.
EF5C           799      ;
EF5C 85 AD      800      POLYPROC sta STRING2
EF5E 84 AE      801      sty STRING2+1
EF60           802      ;
EF60 20 1E EB   803      jsr COPYF2T1
EF63           804      ;
EF63 A9 93      805      lda #TEMP1          ; rey-Y already zero
EF65           806      ;
EF65 20 7F E9   807      jsr FMULT
EF68 20 76 EF   808      jsr POLYNOM2
EF6B           809      ;
EF6B A9 93      810      lda #TEMP1
EF6D A0 00      811      ldy /TEMP1
EF6F           812      ;
EF6F 4C 7F E9   813      jmp FMULT
EF72           814      ;
EF72           815      ;
EF72           816      ; Routine accelerated in order to fix random values at end.
EF72           817      ;
EF72 85 AD      818      POLYNOM sta STRING2
EF74 84 AE      819      sty STRING2+1
EF76           820      ;
EF76 20 21 EB   821      POLYNOM2 jsr COPYF2T2
EF79           822      ;
EF79 B1 AD      823      lda (STRING2),Y
EF7B 85 A3      824      sta SERLEN
EF7D           825      ;
EF7D E6 AD      826      inc STRING2          ; accelerate code
EF7F D0 02      827      bne >1
EF81           828      ;
EF81 E6 AE      829      inc STRING2+1
EF83           830      ;
EF83 A5 AD      831      ^1 lda STRING2

```

```

EF85 A4 AE      832      ldy STRING2+1
EF87            833      ;
EF87 20 7F E9   834      ^2      jsr FMULT
EF8A            835      ;
EF8A A5 AD      836      lda STRING2
EF8C A4 AE      837      ldy STRING2+1
EF8E            838      ;
EF8E 18         839      clc
EF8F            840      ;
EF8F 69 05      841      adc #5
EF91 90 01      842      bcc >3
EF93            843      ;
EF93 C8         844      iny
EF94            845      ;
EF94 85 AD      846      ^3      sta STRING2
EF96 84 AE      847      sty STRING2+1
EF98 20 BE E7   848      jsr FADD
EF9B            849      ;
EF9B A9 98      850      lda #TEMP2
EF9D A0 00      851      ldy /TEMP2
EF9F            852      ;
EF9F C6 A3      853      dec SERLEN
EFA1 D0 E4      854      bne <2
EFA3            855      ;
EFA3 60         856      RTN.EF.A rts
EFA4            857      ;
EFA4            858      ;
EFA4            859      ; RANDVAL1 hex 9835447A      ; integer value = 2553627770
EFA4            860      ; RANDVAL2 hex 6828B146      ; integer value = 1747497286
EFA4            861      ;
EFA4            862      ; Donald Knuth specified the values to use for this
EFA4            863      ; equation:  $X(n+1) = (X(n) * A + C) \bmod M$ ,  $M = 2^{32}$ .
EFA4            864      ;
EFA4            865      ; Any negative integer value is saved as the new IRAND.
EFA4            866      ; A zero value returns the current integer value of IRAND.
EFA4            867      ; Any positive integer value is used as the Range value.
EFA4            868      ; If the Range value is 1, then a fraction is returned.
EFA4            869      ;
EFA4 12 B9 B0   870      RANDVAL1 hex 12B9B0A5      ; 314159269 integer value A
EFA7 A5         871
EFA8 36 19 62   871      RANDVAL2 hex 361962EB      ; 907633386 integer value C
EFAB EB        872      ;
EFAC A5 A2      873      FRND      lda FACSIGN      ; check to save new IRAND
EFAE 30 06      874      bmi >1      ; branch to save new IRAND
EFB0            875      ;
EFB0 A5 9D      876      lda FACEXP      ; display or generate IRAND
EFB2 F0 1A      877      beq >4      ; branch to display IRAND
EFB4            878      ;
EFB4 D0 2A      879      bne >6      ; always taken, gen IRAND
EFB6            880      ;
EFB6            881      ;
EFB6            882      ; Prepare a new IRAND value. The IRAND is any integer
EFB6            883      ; value from 0 to less than  $2^{32}$ .
EFB6            884      ;
EFB6 A9 00      885      ^1      lda #ZERO      ; make positive
EFB8 85 A2      886      sta FACSIGN
EFBA            887      ;
EFBA A9 A0      888      lda #$A0      ; limit FACEXP to  $< 2^{32}$ 
EFBC C5 9D      889      cmp FACEXP
EFBE B0 02      890      bcs >2

```

```

EFC0      891 ;
EFC0 85 9D      892      sta FACEXP
EFC2      893 ;
EFC2 20 F2 EB   894 ^2      jsr FP2INT          ; convert FAC to an integer
EFC5      895 ;
EFC5      896 ;
EFC5      897 ; Copy the integer that resides in FAC back to IRAND.
EFC5      898 ;
EFC5 A2 03      899      ldx #3
EFC7      900 ;
EFC7 B5 9E      901 ^3      lda FACMANT,X
EFC9 95 C9      902      sta IRAND,X
EFCB      903 ;
EFCB CA        904      dex
EFCC 10 F9      905      bpl <3
EFCE      906 ;
EFCE      907 ;
EFCE      908 ; Display the current IRAND value as an integer.
EFCE      909 ;
EFCE A2 04      910 ^4      ldx #4
EFD0      911 ;
EFD0 B5 C8      912 ^5      lda IRAND-1,X
EFD2 95 9D      913      sta FACMANT-1,X
EFD4      914 ;
EFD4 CA        915      dex
EFD5 D0 F9      916      bne <5
EFD7      917 ;
EFD7 86 AC      918      stx FACGUARD
EFD9      919 ;
EFD9 A9 A0      920      lda #$A0
EFDB 85 9D      921      sta FACEXP
EFDD      922 ;
EFDD 4C 2E E8   923      jmp NORMFAC1
EFE0      924 ;
EFE0      925 ;
EFE0      926 ; Save the value currently in FAC as the desired Range.
EFE0      927 ; Copy the RANDVAL1 multiplier to ARGMANT, the RANDVAL2
EFE0      928 ; constant to FACMANT, and IRAND to MULMANT.
EFE0      929 ;
EFE0 A2 93      930 ^6      ldx #TEMP1
EFE2 A0 00      931      ldy /TEMP1
EFE4 20 2B EB   932      jsr COPYFAC
EFE7      933 ;
EFE7 4C 2D F7   934      jmp FRND2
EFEA      935 ;
EFEA      936 ;
EFEA      937 ; FCOS routine entry point; compute COSINE of FAC -> FAC.
EFEA      938 ;
EFEA A9 66      939 FCOS      lda #FPIDIV2          ; FP PI/2
EFEC A0 F0      940      ldy /FPIDIV2
EFEE      941 ;
EFEE 20 BE E7   942      jsr FADD
EFF1      943 ;
EFF1      944 ;
EFF1      945 ; FSIN routine entry point; compute SINE of FAC -> FAC.
EFF1      946 ;
EFF1 20 63 EB   947 FSIN      jsr COPYF2A
EFF4      948 ;
EFF4 A9 99      949      lda #FPIMUL2          ; FP PI*2
EFF6 A0 F0      950      ldy /FPIMUL2
EFF8      951 ;

```

```
EFF8 A6 AA      952      ldx ARGSIGN
EFFA 20 5E EA    953      jsr DIVFAC
EFFD           954      ;
EFFD 20 63 EB    955      jsr COPYF2A
F000           956      ;
F000           957      ;

BSAVE E0ROM,D1,A$1000,B,L$1000

F000           958      usr E0ROM,D1
F000           959      ;
F000           960      ;
F000           961      icl "F0.L,D2"

LLOAD F0.L,D2,A$4000
```

```

F000          1          ttl "ROM Source Code, F0.L"
F000          2          ;
F000          3          ;
F000          4          ; F0.L
F000          5          ;
F000          6          ;
F000          7          obj PAGE10
F000          8          usr
F000          9          ;
F000         10          ;
F000 20 23 EC         11          jsr FINT
F003          12          ;
F003 A9 00           13          lda #ZERO
F005 85 AB           14          sta XORSIGN
F007          15          ;
F007 20 AA E7        16          jsr FSUB2
F00A          17          ;
F00A A9 6B           18          lda #FP.25
F00C A0 F0           19          ldy /FP.25
F00E          20          ;
F00E 20 A7 E7        21          jsr FSUB
F011          22          ;
F011 A5 A2           23          lda FACSIGN
F013 48              24          pha
F014          25          ;
F014 10 0D           26          bpl FSIN.2
F016          27          ;
F016 20 A0 E7        28          jsr FADDHALF
F019          29          ;
F019 A5 A2           30          lda FACSIGN
F01B 30 09           31          bmi >5
F01D          32          ;
F01D A5 16           33          lda SIGNFLG
F01F 49 FF           34          eor #NEGONE
F021 85 16           35          sta SIGNFLG
F023          36          ;
F023 20 CD EE        37          FSIN.2 jsr NEGFAC
F026          38          ;
F026 A9 6B           39          ^5    lda #FP.25
F028 A0 F0           40          ldy /FP.25
F02A          41          ;
F02A 20 BE E7        42          jsr FADD
F02D          43          ;
F02D 68              44          pla
F02E 10 03           45          bpl >6
F030          46          ;
F030 20 CD EE        47          jsr NEGFAC
F033          48          ;
F033 A9 70           49          ^6    lda #POLY.SIN
F035 A0 F0           50          ldy /POLY.SIN
F037          51          ;
F037 4C 5C EF        52          jmp POLYPROC
F03A          53          ;
F03A          54          ;
F03A          55          ; FTAN routine entry point; compute TANGENT of FAC leaving
F03A          56          ; the result in FAC.
F03A          57          ;
F03A 20 1E EB        58          FTAN  jsr COPYF2T1
F03D          59          ;
F03D A9 00           60          lda #ZERO

```

```

F03F 85 16      61      sta SIGNFLG
F041            62      ;
F041 20 F1 EF    63      jsr FSIN
F044            64      ;
F044 A2 8A      65      ldx #TEMP3
F046 A0 00      66      ldy /TEMP3
F048 20 2B EB    67      jsr COPYFAC
F04B            68      ;
F04B A9 93      69      lda #TEMP1
F04D A0 00      70      ldy /TEMP1
F04F 20 F9 EA    71      jsr LOADFAC
F052            72      ;
F052 A9 00      73      lda #ZERO
F054 85 A2      74      sta FACSIGN
F056            75      ;
F056 A5 16      76      lda SIGNFLG
F058 20 62 F0    77      jsr FTAN.2          ; leave as is
F05B            78      ;
F05B A9 8A      79      lda #TEMP3
F05D A0 00      80      ldy /TEMP3
F05F            81      ;
F05F 4C 66 EA    82      jmp FDIV
F062            83      ;
F062            84      ;
F062 48          85      FTAN.2 pha
F063            86      ;
F063 4C 23 F0    87      jmp FSIN.2
F066            88      ;
F066            89      ;
F066            90      ; Floating point values. FPIMUL2 and FPIMUL2B are the
F066            91      ; same. Only need one and not both.
F066            92      ;
F066 81 49 0F    93      FPIDIV2 hex 81490FDAA2          ; FP PI/2 = 1.57079633
F069 DA A2
F06B            94      ;
F06B            95      .if FPCDMOD
F06B            96      .el
F06B            97      ;
F06B            98      FPIMUL2B hex 83490FDAA2          ; FP PI*2 = 6.28318531
F06B            99      ;
F06B           100      .fi
F06B           101      ;
F06B 7F 00 00    102      FP.25 hex 7F00000000          ; FP 0.25
F06E 00 00
F070           103      ;
F070           104      ;
F070           105      ; SIN function polynomials. The absolute SIN polynomials
F070           106      ; are given using the unreferenced space above and below.
F070           107      ;
F070           108      .if FPCDMOD
F070           109      ;
F070 08          110      POLY.SIN hex 08          ; number of polynomials-1
F071 7D 55 76    111      hex 7D55761956          ; FP (2PI)^17/17!
F074 19 56
F076 80 B7 D6    112      hex 80B7D6DCF8          ; FP -(2PI)^15/15!
F079 DC F8
F07B 82 74 7A    113      hex 82747A1A6A          ; FP (2PI)^13/13!
F07E 1A 6A
F080 84 F1 83    114      hex 84F183A7F2          ; FP -(2PI)^11/11!
F083 A7 F2
F085 86 28 3C    115      hex 86283C1A44          ; FP (2PI)^9/9!

```

```

F088 1A 44
F08A 87 99 69    116          hex 8799696672          ; FP -(2PI)^7/7!
F08D 66 72
F08F 87 23 35    117          hex 872335E33B          ; FP (2PI)^5/5!
F092 E3 3B
F094 86 A5 5D    118          hex 86A55DE731          ; FP -(2PI)^3/3!
F097 E7 31
F099              119          ;
F099              120          .el
F099              121          ;
F099              122          POLY.SIN hex 05          ; number of polynomials-1
F099              123          hex 84E61A2D1B          ; FP -(2PI)^11/11!
F099              124          hex 862807FBF8          ; FP (2PI)^9/9!
F099              125          hex 8799688901          ; FP -(2PI)^7/7!
F099              126          hex 872335DFE1          ; FP (2PI)^5/5!
F099              127          hex 86A55DE728          ; FP -(2PI)^3/3!
F099              128          ;
F099              129          .fi
F099              130          ;
F099 83 49 0F    131          FPIMUL2 hex 83490FDAA2          ; FP (2PI)
F09C DA A2
F09E              132          ;
F09E              133          ;
F09E              134          ; These 10 bytes are never referenced. Simply exclusive-or
F09E              135          ; each byte in ASCII in order to produce the backward
F09E              136          ; string "MICROSOFT!".
F09E              137          ;
F09E              138          .if FPCDMOD
F09E              139          .el
F09E              140          ;
F09E              141          hex A6D3C1C8D4
F09E              142          hex C8D5C4CECA
F09E              143          ;
F09E              144          .fi
F09E              145          ;
F09E              146          ;
F09E              147          ; FATAN routine entry point; compute ARCTANGENT of FAC
F09E              148          ; leaving the result in FAC.
F09E              149          ;
F09E A5 A2      150          FATAN      lda FACSIGN
F0A0 48          151                  pha
F0A1 10 03      152                  bpl >1
F0A3            153          ;
F0A3 20 CD EE   154                  jsr NEGFAC
F0A6            155          ;
F0A6 A5 9D      156          ^1      lda FACEXP
F0A8 48          157                  pha
F0A9            158          ;
F0A9 C9 81      159                  cmp #$81
F0AB 90 07      160                  bcc >2
F0AD            161          ;
F0AD A9 04      162                  lda #FP1.0
F0AF A0 EF      163                  ldy /FP1.0
F0B1 20 66 EA   164                  jsr FDIV
F0B4            165          ;
F0B4 A9 CE      166          ^2      lda #POLY.ATN
F0B6 A0 F0      167                  ldy /POLY.ATN
F0B8 20 5C EF   168                  jsr POLYPROC
F0BB            169          ;
F0BB 68          170                  pla
F0BC C9 81      171                  cmp #$81

```



```

F0BE 90 07      172      bcc >3
F0C0            173      ;
F0C0 A9 66      174      lda #FPIDIV2
F0C2 A0 F0      175      ldy /FPIDIV2
F0C4 20 A7 E7   176      jsr FSUB
F0C7            177      ;
F0C7 68         178      ^3      pla
F0C8 10 03      179      bpl >4
F0CA            180      ;
F0CA 4C CD EE   181      jmp NEGFAC
F0CD            182      ;
F0CD 60         183      ^4      rts
F0CE            184      ;
F0CE            185      ;
F0CE 0B         186      POLY.ATN hex 0B      ; number of polynomials-1
F0CF 76 B3 83   187      hex 76B383BDD3      ; FP -6.84793912E-04
F0D2 BD D3      188
F0D4 79 1E F4   188      hex 791EF4A6F5      ; FP 4.85094216E-03
F0D7 A6 F5      189
F0D9 7B 83 FC   189      hex 7B83FCB010      ; FP -0.0161117018
F0DC B0 10      190
F0DE 7C 0C 1F   190      hex 7C0C1F67CA      ; FP 0.034209638
F0E1 67 CA      191
F0E3 7C DE 53   191      hex 7CDE53CBC1      ; FP -0.0542791328
F0E6 CB C1      192
F0E8 7D 14 64   192      hex 7D1464704C      ; FP 0.0724571965
F0EB 70 4C      193
F0ED 7D B7 EA   193      hex 7DB7EA517A      ; FP -0.0898023954
F0F0 51 7A      194
F0F2 7D 63 30   194      hex 7D6330887E      ; FP 0.110932413
F0F5 88 7E      195
F0F7 7E 92 44   195      hex 7E9244993A      ; FP -0.142839808
F0FA 99 3A      196
F0FC 7E 4C CC   196      hex 7E4CCC91C7      ; FP 0.19999912
F0FF 91 C7      197
F101 7F AA AA   197      hex 7FAAAAAA13      ; FP -0.333333316
F104 AA 13      198
F106 81 00 00   198      hex 8100000000      ; FP 1.000000000
F109 00 00      199
F10B            200      ;
F10B            201      ; CHRGET and CHRGOT routines and FPRAND variable.
F10B            202      ; C-flag is set for 00:2F and 3A:FF, clear for 30:39.
F10B            203      ; Modified this routine and 0xC8 is available.
F10B            204      ;
F10B            205      PGZCODE:
F10B            206      phs CHRGOTADR
00B1            207      ;
00B1 E6 B8      208      CHRGET      inc TXTPTR
00B3 D0 02      209      bne CHRGOT
00B5            210      ;
00B5 E6 B9      211      inc TXTPTR+1
00B7            212      ;
00B7 AD 00 00   213      CHRGOT      lda *-*      ; TXTPTR
00BA C9 3A      214      cmp #'9'+1
00BC B0 0A      215      bcs >1
00BE            216      ;
00BE C9 20      217      cmp #SPACE&MSBCLR
00C0 F0 EF      218      beq CHRGET
00C2            219      ;
00C2 38         220      sec

```

```

00C3      221 ;
00C3 E9 30      222          sbc #'0'
00C5      223 ;
00C5 38        224          sec
00C6      225 ;
00C6 E9 D0      226          sbc #0-'0'
00C8      227 ;
00C8 60        228 ^1      rts
00C9      229 ;
00C9 4F C7 52   230 FPRAND  hex 4FC75258          ; 1338462808 integer value
00CC 58
00CD      231 ;
00CD 00        232          hex 00          ; used for TOKEN, not by FRND
00CE      233 ;
001D      234 ZPCDLEN equ *-CHRGET
00CE      235 ;
00CE      236          phs PGZCODE+ZPCDLEN
F128      237 ;
F128      238 ;
F128      239 ; Many of the COLDSTRT initializations are ridiculous and
F128      240 ; they are of no benefit. These initializations are
F128      241 ; retained, however, but this space could be utilized far
F128      242 ; more effectively.
F128      243 ;
F128 A2 FF      244 COLDSTRT ldx #NEGONE
F12A 86 76      245          stx CURLIN+1
F12C      246 ;
F12C A2 FB      247          ldx #$FB
F12E 9A        248          txs
F12F      249 ;
F12F A9 28      250          lda #COLDSTRT          ; not needed
F131 A0 F1      251          ldy /COLDSTRT
F133      252 ;
F133 85 01      253          sta GOWARM+1          ; not needed
F135 84 02      254          sty GOWARM+2
F137      255 ;
F137 85 04      256          sta GOSTROUT+1        ; not needed
F139 84 05      257          sty GOSTROUT+2
F13B      258 ;
F13B 20 73 F2   259          jsr BNORMAL
F13E      260 ;
F13E A9 00      261          lda #*-*          ; get jmp instruction
F140      262          dfs !-1
F13F 4C 00 00   263          jmp *-*          ; to form jmp vectors
F142      264          dfs !-2
F140      265 ;
F140 85 00      266          sta GOWARM
F142 85 03      267          sta GOSTROUT
F144 85 90      268          sta JMPADRS
F146 85 0A      269          sta GOUSR
F148      270 ;
F148 A9 99      271          lda #IQ.ERR
F14A A0 E1      272          ldy /IQ.ERR
F14C      273 ;
F14C 85 0B      274          sta GOUSR+1
F14E 84 0C      275          sty GOUSR+2
F150      276 ;
F150      277 ;
F150      278 ; Copy CHRGET and FPRAND to pagezero. The original loop
F150      279 ; value was wrong.
F150      280 ;

```

```

F150 A2 1D      281      ldx #ZPCDLEN      ; correct value
F152           282      ;
F152 BD 0A F1   283      ^1      lda PGZCODE-1,X
F155 95 B0      284      sta CHRG TADR-1,X
F157           285      ;
F157 86 F1      286      stx SPEEDBYT
F159           287      ;
F159 CA        288      dex
F15A D0 F6      289      bne <1
F15C           290      ;
F15C 8A        291      txa      ; zero
F15D           292      ;
F15D 85 54      293      sta LASTPT+1
F15F 85 A4      294      sta SAVARGEX
F161 85 F2      295      sta TRACEFLG
F163           296      ;
F163 48        297      pha
F164           298      ;
F164 A9 03      299      lda #3
F166 85 8F      300      sta DSCLEN
F168           301      ;
F168 20 FB DA   302      jsr PRTCR
F16B           303      ;
F16B A9 01      304      lda #1
F16D 8D FD 01   305      sta INPUT-3
F170 8D FC 01   306      sta INPUT-4
F173           307      ;
F173 A2 55      308      ldx #TEMPST
F175 86 52      309      stx TEMPPT
F177           310      ;
F177           311      ;
F177           312      ; Determine amount of memory.
F177           313      ;
F177 A9 00      314      lda #PAGE08
F179 A0 08      315      ldy /PAGE08
F17B           316      ;
F17B 85 50      317      sta ACL
F17D 84 51      318      sty ACH
F17F           319      ;
F17F A0 00      320      ldy #ZERO
F181           321      ;
F181 E6 51      322      ^2      inc ACH
F183           323      ;
F183 B1 50      324      lda (ACL),Y
F185 49 FF      325      eor #NEGONE
F187 91 50      326      sta (ACL),Y
F189           327      ;
F189 D1 50      328      cmp (ACL),Y
F18B D0 08      329      bne >3
F18D           330      ;
F18D 49 FF      331      eor #NEGONE
F18F 91 50      332      sta (ACL),Y
F191           333      ;
F191 D1 50      334      cmp (ACL),Y
F193 F0 EC      335      beq <2
F195           336      ;
F195 A4 50      337      ^3      ldy ACL
F197           338      ;
F197 A5 51      339      lda ACH
F199 29 F0      340      and #$F0
F19B           341      ;

```

```

F19B 84 73      342      sty MEMSIZE
F19D 85 74      343      sta MEMSIZE+1
F19F           344      ;
F19F 84 6F      345      sty FRETOP
F1A1 85 70      346      sta FRETOP+1
F1A3           347      ;
F1A3           348      ;
F1A3           349      ; Initialize Applesoft program start location to 0x0801.
F1A3           350      ; This code has been modified. Any address may replace
F1A3           351      ; PAGE08 as the start address for Applesoft.
F1A3           352      ;
F1A3 A9 01      353      lda #PAGE08+1
F1A5 A0 08      354      ldy /PAGE08+1
F1A7           355      ;
F1A7 85 67      356      sta PRGTAB
F1A9 84 68      357      sty PRGTAB+1
F1AB           358      ;
F1AB A2 00      359      ldx #ZERO
F1AD 86 D6      360      stx RUNFLAG          ; also cleared in SCRTCH
F1AF 8E 00 08   361      stx PAGE08
F1B2           362      ;
F1B2 20 E3 D3   363      jsr CKSTRSIZ
F1B5 20 4B D6   364      jsr SCRTCH
F1B8           365      ;
F1B8 A9 3A      366      lda #STROUT
F1BA A0 DB      367      ldy /STROUT
F1BC           368      ;
F1BC 85 04      369      sta GOSTROUT+1
F1BE 84 05      370      sty GOSTROUT+2
F1C0           371      ;
F1C0 A9 3C      372      lda #RESTART
F1C2 A0 D4      373      ldy /RESTART
F1C4           374      ;
F1C4 85 01      375      sta GOWARM+1
F1C6 84 02      376      sty GOWARM+2
F1C8           377      ;
F1C8           378      ; jmp (LOC1)
F1C8           379      ;
F1C8 4C 3C D4   380      jmp RESTART          ; accelerate code
F1CB           381      ;
F1CB           382      ;
F1CB           383      ; Convert natural log LN to base-10 log LOG.
F1CB           384      ;
F1CB 20 41 E9   385      FLOG      jsr FLN
F1CE           386      ;
F1CE A9 13      387      lda #FPLOGE
F1D0 A0 E9      388      ldy /FPLOGE
F1D2           389      ;
F1D2 4C 7F E9   390      jmp FMULT
F1D5           391      ;
F1D5           392      ;
F1D5 20 67 DD   393      BCALL      jsr FRMNUM
F1D8 20 52 E7   394      jsr GETADDR
F1DB           395      ;
F1DB 6C 50 00   396      jmp (ACL)
F1DE           397      ;
F1DE           398      ;
F1DE 20 F8 E6   399      BIN      jsr GETBYT
F1E1           400      ;
F1E1 8A         401      txa
F1E2           402      ;

```

```

F1E2 4C 8B FE      403          jmp INPORT
F1E5              404          ;
F1E5              405          ;
F1E5 20 F8 E6      406 BPR      jsr GETBYT
F1E8              407          ;
F1E8 8A            408          txa
F1E9              409          ;
F1E9 4C 95 FE      410          jmp OUTPORT
F1EC              411          ;
F1EC              412          ;
F1EC              413          ; LORES comma separated coordinates for H2 and V2.
F1EC              414          ;
F1EC 20 F8 E6      415 PLOTFNS  jsr GETBYT
F1EF              416          ;
F1EF E0 30         417          cpx #'0'
F1F1 B0 13         418          bcs GOIQERR
F1F3              419          ;
F1F3 86 F0         420          stx FIRST
F1F5              421          ;
F1F5 A9 2C         422          lda #', '
F1F7 20 C0 DE      423          jsr SYNTAXCHK
F1FA              424          ;
F1FA 20 F8 E6      425          jsr GETBYT
F1FD              426          ;
F1FD E0 30         427          cpx #'0'
F1FF B0 05         428          bcs GOIQERR
F201              429          ;
F201 86 2C         430          stx H2
F203 86 2D         431          stx V2
F205              432          ;
F205 60            433          rts
F206              434          ;
F206              435          ;
F206 4C 99 E1      436 GOIQERR  jmp IQ.ERR
F209              437          ;
F209              438          ;
F209 20 EC F1      439 ODRCOOR  jsr PLOTFNS
F20C              440          ;
F20C E4 F0         441          cpx FIRST
F20E B0 08         442          bcs >5
F210              443          ;
F210 A5 F0         444          lda FIRST
F212 85 2C         445          sta H2
F214 85 2D         446          sta V2
F216              447          ;
F216 86 F0         448          stx FIRST
F218              449          ;
F218 A9 C5         450 ^5      lda #TKAT
F21A 20 C0 DE      451          jsr SYNTAXCHK
F21D              452          ;
F21D 20 F8 E6      453          jsr GETBYT
F220              454          ;
F220 E0 30         455          cpx #$30
F222 B0 E2         456          bcs GOIQERR
F224              457          ;
F224 60            458          rts
F225              459          ;
F225              460          ;
F225 20 EC F1      461 BPLOT    jsr PLOTFNS
F228              462          ;
F228 8A            463          txa

```

; > 12288 line number?

```

F229          464 ;
F229 A4 F0    465      ldy FIRST
F22B C0 28    466      cpy #40
F22D B0 D7    467      bcs GOIQERR
F22F          468 ;
F22F 4C 00 F8 469      jmp PLOT
F232          470 ;
F232          471 ;
F232 20 09 F2 472 BHLIN jsr ODRCOOR
F235          473 ;
F235 8A       474      txa
F236          475 ;
F236 A4 2C    476      ldy H2
F238 C0 28    477      cpy #40
F23A B0 CA    478      bcs GOIQERR
F23C          479 ;
F23C A4 F0    480      ldy FIRST
F23E          481 ;
F23E 4C 19 F8 482      jmp HLINE
F241          483 ;
F241          484 ;
F241 20 09 F2 485 BVLIN jsr ODRCOOR
F244          486 ;
F244 8A       487      txa
F245 A8       488      tay
F246          489 ;
F246 C0 28    490      cpy #40
F248 B0 BC    491      bcs GOIQERR
F24A          492 ;
F24A A5 F0    493      lda FIRST
F24C          494 ;
F24C 4C 28 F8 495      jmp VLINE
F24F          496 ;
F24F          497 ;
F24F 20 F8 E6 498 BCOLOR jsr GETBYT
F252          499 ;
F252 8A       500      txa
F253          501 ;
F253 4C 64 F8 502      jmp SETCOL
F256          503 ;
F256          504 ;
F256 20 F8 E6 505 BVTAB  jsr GETBYT
F259          506 ;
F259 CA       507      dex
F25A          508 ;
F25A 8A       509      txa
F25B C9 18    510      cmp #24
F25D B0 A7    511      bcs GOIQERR
F25F          512 ;
F25F 4C 5B FB 513      jmp TABV
F262          514 ;
F262          515 ;
F262 20 F8 E6 516 BSPEED jsr GETBYT
F265          517 ;
F265 8A       518      txa
F266 49 FF    519      eor #NEGONE
F268 AA       520      tax
F269          521 ;
F269 E8       522      inx
F26A 86 F1    523      stx SPEEDBYT
F26C          524 ;

```

```

F26C 60          525          rts
F26D          526          ;
F26D          527          ;
F26D 38          528 BTRACE   sec
F26E          529          ;
F26E 90 00       530          bcc *+2
F270          531          dfs !-1
F26F          532          ;
F26F 18          533 BNOTRACE clc
F270          534          ;
F270 66 F2       535          ror TRACEFLG
F272          536          ;
F272 60          537          rts
F273          538          ;
F273          539          ;
F273 A9 FF       540 BNORMAL  lda #NEGONE
F275          541          ;
F275 2C 00 00    542          bit *-*
F278          543          dfs !-2
F276          544          ;
F276 A9 3F       545 BINVERSE lda #INVERSE
F278          546          ;
F278 A2 00       547          ldx #ZERO          ; accelerate code
F27A          548          ;
F27A 85 32       549 ^7          sta INVFLG
F27C 86 F3       550          stx FLASHBYT
F27E          551          ;
F27E 60          552          rts
F27F          553          ;
F27F          554          ;
F27F          555          dfs 1,ZERO          ; 1 byte
F280          556          ;
F280          557          ;
F280 A9 7F       558 BFLASH   lda #FLASH
F282          559          ;
F282 A2 40       560          ldx #$40
F284 D0 F4       561          bne <7          ; always taken
F286          562          ;
F286          563          ;
F286 20 67 DD     564 BHIMEM   jsr FRMNUM
F289 20 52 E7     565          jsr GETADDR
F28C          566          ;
F28C A5 50       567          lda ACL
F28E C5 6D       568          cmp STREND
F290          569          ;
F290 A5 51       570          lda ACH
F292 E5 6E       571          sbc STREND+1
F294 B0 03       572          bcs >9
F296          573          ;
F296 4C 10 D4     574 ^8          jmp PRMSG07          ; Out of Memory error
F299          575          ;
F299 A5 50       576 ^9          lda ACL
F29B 85 73       577          sta MEMSIZE
F29D 85 6F       578          sta FRETOP
F29F          579          ;
F29F A5 51       580          lda ACH
F2A1 85 74       581          sta MEMSIZE+1
F2A3 85 70       582          sta FRETOP+1
F2A5          583          ;
F2A5 60          584          rts
F2A6          585          ;

```

```

F2A6          586 ;
F2A6 20 67 DD 587 BLOMEM      jsr FRMNUM
F2A9 20 52 E7 588             jsr GETADDR
F2AC          589 ;
F2AC A5 50    590             lda ACL
F2AE C5 73    591             cmp MEMSIZE
F2B0          592 ;
F2B0 A5 51    593             lda ACH
F2B2 E5 74    594             sbc MEMSIZE+1
F2B4 B0 E0    595             bcs <8
F2B6          596 ;
F2B6 A5 50    597             lda ACL
F2B8 C5 69    598             cmp VARTAB
F2BA          599 ;
F2BA A5 51    600             lda ACH
F2BC E5 6A    601             sbc VARTAB+1
F2BE 90 D6    602             bcc <8
F2C0          603 ;
F2C0 A5 50    604             lda ACL
F2C2 85 69    605             sta VARTAB
F2C4          606 ;
F2C4 A5 51    607             lda ACH
F2C6 85 6A    608             sta VARTAB+1
F2C8          609 ;
F2C8 4C 6C D6 610             jmp CLEARC
F2CB          611 ;
F2CB          612 ;
F2CB A9 AB    613 BONERR      lda #$80+BSGOTO/2
F2CD 20 C0 DE 614             jsr SYNTAXCHK
F2D0          615 ;
F2D0 A5 B8    616             lda TXTPTR
F2D2 85 F4    617             sta TXTPTRSV
F2D4          618 ;
F2D4 A5 B9    619             lda TXTPTR+1
F2D6 85 F5    620             sta TXTPTRSV+1
F2D8          621 ;
F2D8 38       622             sec
F2D9 66 D8    623             ror ERRFLG
F2DB          624 ;
F2DB A5 75    625             lda CURLIN
F2DD 85 F6    626             sta CURLINSV
F2DF          627 ;
F2DF A5 76    628             lda CURLIN+1
F2E1 85 F7    629             sta CURLINSV+1
F2E3          630 ;
F2E3 20 A6 D9 631             jsr DATSCAN2
F2E6          632 ;
F2E6 4C 98 D9 633             jmp BDATA2
F2E9          634 ;
F2E9          635 ;
F2E9 86 DE    636 HANDLERR  stx ERRNUM
F2EB          637 ;
F2EB A6 F8    638             ldx REMSTK
F2ED 86 DF    639             stx ERRSTK
F2EF          640 ;
F2EF A5 75    641             lda CURLIN
F2F1 85 DA    642             sta ERRLIN
F2F3          643 ;
F2F3 A5 76    644             lda CURLIN+1
F2F5 85 DB    645             sta ERRLIN+1
F2F7          646 ;

```



```

F2F7 A5 79      647      lda TEXTPTR
F2F9 85 DC      648      sta ERRPOS
F2FB           649      ;
F2FB A5 7A      650      lda TEXTPTR+1
F2FD 85 DD      651      sta ERRPOS+1
F2FF           652      ;
F2FF A5 F4      653      lda TXTPTRSV
F301 85 B8      654      sta TXTPTR
F303           655      ;
F303 A5 F5      656      lda TXTPTRSV+1
F305 85 B9      657      sta TXTPTR+1
F307           658      ;
F307 A5 F6      659      lda CURLINSV
F309 85 75      660      sta CURLIN
F30B           661      ;
F30B A5 F7      662      lda CURLINSV+1
F30D 85 76      663      sta CURLIN+1
F30F           664      ;
F30F 20 B7 00   665      jsr CHRGOT
F312 20 3E D9   666      jsr BGOTO
F315           667      ;
F315 4C D2 D7   668      jmp NEWSTT
F318           669      ;
F318           670      ;
F318 A5 DA      671      BRESUME  lda ERRLIN
F31A 85 75      672      sta CURLIN
F31C           673      ;
F31C A5 DB      674      lda ERRLIN+1
F31E 85 76      675      sta CURLIN+1
F320           676      ;
F320 A5 DC      677      lda ERRPOS
F322 85 B8      678      sta TXTPTR
F324           679      ;
F324 A5 DD      680      lda ERRPOS+1
F326 85 B9      681      sta TXTPTR+1
F328           682      ;
F328 A6 DF      683      ldx ERRSTK
F32A 9A         684      txs
F32B           685      ;
F32B 4C D2 D7   686      jmp NEWSTT
F32E           687      ;
F32E           688      ;
F32E 4C C9 DE   689      ^1      jmp DOMESG02      ; Syntax error
F331           690      ;
F331 B0 FB      691      BDEL    bcs <1
F333           692      ;
F333 A6 AF      693      ldx PRGEND
F335 86 69      694      stx VARTAB
F337           695      ;
F337 A6 B0      696      ldx PRGEND+1
F339 86 6A      697      stx VARTAB+1
F33B           698      ;
F33B 20 0C DA   699      jsr LINGET
F33E 20 1A D6   700      jsr FNDLIN
F341           701      ;
F341 A5 9B      702      lda LOWTR
F343 85 60      703      sta DEST
F345           704      ;
F345 A5 9C      705      lda LOWTR+1
F347 85 61      706      sta DEST+1
F349           707      ;

```

```

F349 A9 2C      708      lda #', '
F34B 20 C0 DE   709      jsr SYNTAXCHK
F34E           710      ;
F34E 20 0C DA   711      jsr LINGET
F351           712      ;
F351 E6 50      713      inc ACL
F353 D0 02      714      bne >2
F355           715      ;
F355 E6 51      716      inc ACH
F357           717      ;
F357 20 1A D6   718      ^2      jsr FNDLIN
F35A           719      ;
F35A A5 9B      720      lda LOWTR
F35C C5 60      721      cmp DEST
F35E           722      ;
F35E A5 9C      723      lda LOWTR+1
F360 E5 61      724      sbc DEST+1
F362 B0 01      725      bcs >3
F364           726      ;
F364 60         727      rts
F365           728      ;
F365 A0 00      729      ^3      ldy #ZERO
F367           730      ;
F367 B1 9B      731      ^4      lda (LOWTR),Y
F369 91 60      732      sta (DEST),Y
F36B           733      ;
F36B E6 9B      734      inc LOWTR
F36D D0 02      735      bne >5
F36F           736      ;
F36F E6 9C      737      inc LOWTR+1
F371           738      ;
F371 E6 60      739      ^5      inc DEST
F373 D0 02      740      bne >6
F375           741      ;
F375 E6 61      742      inc DEST+1
F377           743      ;
F377 A5 69      744      ^6      lda VARTAB
F379 C5 9B      745      cmp LOWTR
F37B           746      ;
F37B A5 6A      747      lda VARTAB+1
F37D E5 9C      748      sbc LOWTR+1
F37F B0 E6      749      bcs <4
F381           750      ;
F381 A6 61      751      ldx DEST+1
F383           752      ;
F383 A4 60      753      ldy DEST
F385 D0 01      754      bne >7
F387           755      ;
F387 CA         756      dex
F388           757      ;
F388 88         758      ^7      dey
F389           759      ;
F389 86 6A      760      stx VARTAB+1
F38B 84 69      761      sty VARTAB
F38D           762      ;
F38D 4C F2 D4   763      jmp ASENTER
F390           764      ;
F390           765      ;
F390 AD 56 C0   766      BGR      lda HIRESOFF
F393 AD 53 C0   767      lda MIXEDON
F396           768      ;

```

```

F396 4C 40 FB      769          jmp SETGR
F399              770          ;
F399              771          ;
F399              772          ; Modified routine with substantially better logic.
F399              773          ;
F399 AD 53 C0      774 BTEXT      lda MIXEDON
F39C              775          ;
F39C 4C 33 FB      776          jmp INIT2
F39F              777          ;
F39F              778          ;
F39F              779          ; Removed STORE and RECALL statements since they depend on
F39F              780          ; reading and writing data to and from the cassette ports
F39F              781          ; that are no longer useful to the Apple //e user.
F39F              782          ;
F39F              783          ;
F39F              784          ; Read audio waveform for HEADER, SYNC, and binary DATA to
F39F              785          ; the address in A1 until the address in A2. Wait until
F39F              786          ; three seconds of HEADER has past before looking for SYNC.
F39F              787          ; This routine was originally found at 0xC5D1 and modified.
F39F              788          ;
F39F 20 CA D8      789 CXREAD      jsr RD2BIT              ; wait for waveform to change
F3A2              790          ;
F3A2 A9 FF         791          lda #NEGONE              ; initialize CHECKSUM
F3A4 85 2E         792          sta CHKSUM
F3A6              793          ;
F3A6 A2 12         794          ldx #18
F3A8              795          ;
F3A8 20 A8 FC      796 ^1          jsr WAIT              ; waste 167309 cycles
F3AB              797          ;
F3AB CA           798          dex
F3AC D0 FA         799          bne <1
F3AE              800          ;
F3AE 20 CA D8      801          jsr RD2BIT              ; wait for waveform to change
F3B1              802          ;
F3B1 A0 24         803 ^2          ldy #$24              ; waveform change, 432 cycles
F3B3              804          ;
F3B3 20 CD D8      805          jsr RDBIT
F3B6 B0 F9         806          bcs <2
F3B8              807          ;
F3B8 20 CD D8      808          jsr RDBIT
F3BB              809          ;
F3BB A0 3B         810          ldy #$3B              ; waveform change, 708 cycles
F3BD              811          ;
F3BD 20 BC D8      812 ^3          jsr RDBYTE              ; read DATA, MSB first
F3C0              813          ;
F3C0 81 3C         814          sta (A1L,X)              ; save data, X-reg = ZERO
F3C2              815          ;
F3C2 45 2E         816          eor CHKSUM
F3C4 85 2E         817          sta CHKSUM              ; update CHECKSUM
F3C6              818          ;
F3C6 20 BA FC      819          jsr NEXTA1              ; increment A1, compare to A2
F3C9              820          ;
F3C9 A0 35         821          ldy #$35              ; waveform change, 636 cycles
F3CB              822          ;
F3CB 90 F0         823          bcc <3              ; from NEXTA1
F3CD              824          ;
F3CD 20 BC D8      825          jsr RDBYTE              ; read CHECKSUM data byte
F3D0              826          ;
F3D0 C5 2E         827          cmp CHKSUM              ; is the data good?
F3D2              828          ;
F3D2 60           829          rts              ; return to CALLER

```

```

F3D3      830 ;
F3D3      831 ;
F3D3      832      dfs 1,ZERO      ; 1 byte
F3D4      833 ;
F3D4      834 ;
F3D4      835 ; The HGR2 and HGR routines have been modified.
F3D4      836 ;
F3D4 2C 52 C0 837 ^1      bit MIXEDOFF
F3D7      838 ;
F3D7 60      839      rts
F3D8      840 ;
F3D8 A9 40      841 BHGR2      lda /PAGE40
F3DA 20 EE F3    842      jsr CLRHIREs
F3DD      843 ;
F3DD 2C 55 C0    844      bit PAGE2ON
F3E0      845 ;
F3E0 90 F2      846      bcc <1      ; always taken
F3E2      847 ;
F3E2      848 ;
F3E2 A9 20      849 BHGR      lda /PAGE20
F3E4 20 EE F3    850      jsr CLRHIREs
F3E7      851 ;
F3E7 2C 54 C0    852      bit PAGE1ON
F3EA 2C 53 C0    853      bit MIXEDON
F3ED      854 ;
F3ED 60      855      rts
F3EE      856 ;
F3EE      857 ;
F3EE      858 ; Discretely and quickly clear the selected HIRES screen.
F3EE      859 ;
F3EE 85 1B      860 CLRHIREs sta SHAPE+1
F3F0 85 E6      861      sta HRPAG      ; selected HIRES screen
F3F2      862 ;
F3F2 A0 00      863      ldy #ZERO
F3F4 84 1A      864      sty SHAPE
F3F6 84 1C      865      sty COLBITS
F3F8      866 ;
F3F8 A2 20      867      ldx /PAGE40-PAGE20
F3FA      868 ;
F3FA A5 1C      869 ^5      lda COLBITS
F3FC 91 1A      870      sta (SHAPE),Y
F3FE      871 ;
F3FE 20 7E F4    872      jsr COLSHIFT
F401      873 ;
F401 C8      874      iny
F402 D0 F6      875      bne <5
F404      876 ;
F404 E6 1B      877      inc SHAPE+1
F406      878 ;
F406 CA      879      dex
F407 D0 F1      880      bne <5
F409      881 ;
F409 2C 57 C0    882      bit HIRESON
F40C 2C 50 C0    883      bit TEXTOFF
F40F      884 ;
F40F 18      885      clc      ; for branch always taken
F410      886 ;
F410 60      887      rts
F411      888 ;
F411      889 ;
F411      890      icl "F4.L"

```

LLOAD F4.L,A\$4000

```

F411      1      ttl "ROM Source Code, F4.L"
F411      2      ;
F411      3      ;
F411      4      ; F4.L
F411      5      ;
F411      6      ;
F411      7      ;      ---A-reg--  -GBASL--  -GBASH--
F411  86  E0      8  HPOSN      stx  HRXCOOR
F413  84  E1      9              sty  HRXCOOR+1
F415  85  E2     10              sta  HRYCOOR      ;  --ABCDEFGH  -----
F417      11      ;
F417  48          12      pha      ;  --ABCDEFGH  -----
F418      13      ;
F418  29  C0     14      and  #$C0      ;  --AB000000  -----
F41A  85  26     15      sta  GBASL      ;  --AB000000  AB000000  -----
F41C      16      ;
F41C  4A          17      lsr      ;  0-0AB00000  AB000000  -----
F41D  4A          18      lsr      ;  0-00AB0000  AB000000  -----
F41E      19      ;
F41E  05  26     20      ora  GBASL      ;  0-ABAB0000  AB000000  -----
F420  85  26     21      sta  GBASL      ;  0-ABAB0000  ABAB0000  -----
F422      22      ;
F422  68          23      pla      ;  0-ABCDEFGH  ABAB0000  -----
F423  85  27     24      sta  GBASH      ;  0-ABCDEFGH  ABAB0000  ABCDEFGH
F425      25      ;
F425  0A          26      asl      ;  A-BCDEFGH0  ABAB0000  ABCDEFGH
F426  0A          27      asl      ;  B-CDEFGH00  ABAB0000  ABCDEFGH
F427      28      ;
F427  0A          29      asl      ;  C-DEFGH000  ABAB0000  ABCDEFGH
F428  26  27     30      rol  GBASH      ;  A-DEFGH000  ABAB0000  BCDEFGHC
F42A      31      ;
F42A  0A          32      asl      ;  D-EFGH0000  ABAB0000  BCDEFGHC
F42B  26  27     33      rol  GBASH      ;  B-EFGH0000  ABAB0000  CDEFGHCD
F42D      34      ;
F42D  0A          35      asl      ;  E-FGH00000  ABAB0000  CDEFGHCD
F42E  66  26     36      ror  GBASL      ;  0-FGH00000  EABAB000  CDEFGHCD
F430      37      ;
F430  A5  27     38      lda  GBASH      ;  0-CDEFGHCD  EABAB000  CDEFGHCD
F432  29  1F     39      and  #$1F      ;  0-000FGHCD  EABAB000  CDEFGHCD
F434      40      ;
F434  05  E6     41      ora  HRPAG      ;  0-PPPPFGHCD  EABAB000  CDEFGHCD
F436  85  27     42      sta  GBASH      ;  0-PPPPFGHCD  EABAB000  PPPFGHCD
F438      43      ;
F438  8A          44      txa
F439      45      ;
F439  C0  00     46      cpy  #ZERO
F43B  F0  05     47      beq  >2      ; carry set either way
F43D      48      ;
F43D      49      ;
F43D      50      ; Pixel 256 is 256/7 = line byte #36 with remainder 4 bits.
F43D      51      ;
F43D  A0  23     52      ldy  #35
F43F      53      ;
F43F  69  04     54      adc  #4
F441      55      ;
F441  C8          56      ^1      iny
F442      57      ;
F442  E9  07     58      ^2      sbc  #PXLSBYTE
F444  B0  FB     59      bcs  <1
F446      60      ;

```

```

F446 84 E5      61      sty HRHORZ
F448           62      ;
F448 AA        63      tax
F449           64      ;
F449           65      ;
F449           66      ; X-reg ranges from $F9 to $FF so it can be used as an
F449           67      ; index of 0 to 6, respectively.
F449           68      ;
F449 BD 39 F4   69      lda BITABLE-$F9,X
F44C 85 30      70      sta COLOR
F44E           71      ;
F44E 98        72      tya
F44F 4A        73      lsr
F450           74      ;
F450 A5 E4      75      lda HRCOLOR
F452 85 1C      76      sta COLBITS
F454           77      ;
F454 B0 28      78      bcs COLSHIFT
F456           79      ;
F456 60        80      rts
F457           81      ;
F457           82      ;
F457           83      ; Enter HPLOT routine with the following start coordinates:
F457           84      ;
F457           85      ; X-reg, HIRES X-coordinate, low order HORZ byte (0-255)
F457           86      ; Y-reg, HIRES X-coordinate, high order HORZ byte (256-279)
F457           87      ; A-reg, HIRES Y-coordinate VERT byte (0-191)
F457           88      ;
F457 20 11 F4   89      HPLOT      jsr HPOSN
F45A           90      ;
F45A B1 26      91      lda (GBASL),Y
F45C 45 1C      92      eor COLBITS
F45E 25 30      93      and COLOR
F460           94      ;
F460 51 26      95      eor (GBASL),Y
F462 91 26      96      sta (GBASL),Y
F464           97      ;
F464 60        98      rts
F465           99      ;
F465          100      ;
F465 10 23     101      HRMOVLf  bpl HRMOVRT      ; Y-reg = HRHORZ
F467           102      ;
F467 A5 30     103      lda COLOR
F469           104      ;
F469 4A        105      lsr      ; bit 0 pixel test
F46A B0 05     106      bcs >4
F46C           107      ;
F46C 49 C0     108      eor #$C0      ; add bit 7 pixel
F46E           109      ;
F46E 85 30     110      ^3      sta COLOR
F470           111      ;
F470 60        112      rts
F471           113      ;
F471 88        114      ^4      dey      ; move left
F472 10 02     115      bpl >5
F474           116      ;
F474 A0 27     117      ldy #39      ; continue on screen right
F476           118      ;
F476 A9 C0     119      ^5      lda #$C0      ; get bit 7 pixel
F478           120      ;
F478 85 30     121      ^6      sta COLOR

```

```

F47A 84 E5      122      sty HRHORZ
F47C           123      ;
F47C A5 1C      124      lda COLBITS
F47E           125      ;
F47E           126      ;
F47E           127      ; No branch if 0x1F<COLBITS<0xE0.
F47E           128      ;
F47E 0A         129      COLSHIFT asl
F47F           130      ;
F47F C9 C0      131      cmp #$C0
F481 10 06      132      bpl >7
F483           133      ;
F483 A5 1C      134      lda COLBITS
F485 49 7F      135      eor #MSBCLR          ; make inverse for odd byte
F487 85 1C      136      sta COLBITS
F489           137      ;
F489 60         138      ^7      rts
F48A           139      ;
F48A           140      ;
F48A A5 30      141      HRMOVRT  lda COLOR
F48C 0A         142      asl          ; bit 7 pixel test
F48D           143      ;
F48D 49 80      144      eor #MSBSET
F48F 30 DD      145      bmi <3          ; all except bit 7 pixel
F491           146      ;
F491 A9 81      147      lda #$81      ; get bit 0 pixel
F493           148      ;
F493 C8         149      iny          ; move right
F494           150      ;
F494 C0 28      151      cpy #40
F496 90 E0      152      bcc <6
F498           153      ;
F498 A0 00      154      ldy #ZERO      ; continue on screen left
F49A           155      ;
F49A B0 DC      156      bcs <6          ; always taken
F49C           157      ;
F49C           158      ;
F49C           159      dfs 2,ZERO      ; 2 bytes
F49E           160      ;
F49E           161      ;
F49E           162      ; The Draw Shape routines have been heavy modified. The
F49E           163      ; DRAWHDR common header uses the MSB of OPRND to fall into
F49E           164      ; either XDRAWIT or DRAWIT. Each has specific code before
F49E           165      ; falling into common code. The XDRAWIT header was flawed
F49E           166      ; and generated anomalies. The common code was duplicated.
F49E           167      ;
F49E A5 D0      168      DRAWHDR  lda SHPVAL
F4A0 29 04      169      and #PLOTMASK
F4A2 F0 24      170      beq >3
F4A4           171      ;
F4A4 24 44      172      bit OPRND          ; XDRAWIT/DRAWIT flag
F4A6 10 12      173      bpl DRAWIT
F4A8           174      ;
F4A8           175      ;
F4A8           176      ; XDRAW code.
F4A8           177      ;
F4A8 B1 26      178      XDRAWIT  lda (GBASL),Y
F4AA 25 1C      179      and COLBITS          ; added to support color
F4AC 25 30      180      and COLOR
F4AE 29 7F      181      and #MSBCLR
F4B0 D0 12      182      bne >2

```



```

F4B2          183 ;
F4B2 A5 1C    184         lda COLBITS           ; added to support color
F4B4 25 30    185         and COLOR
F4B6 29 7F    186         and #MSBCLR
F4B8 10 08    187         bpl >1                 ; always taken
F4BA          188 ;
F4BA          189 ;
F4BA          190 ; DRAW code.
F4BA          191 ;
F4BA B1 26    192 DRAWIT   lda (GBASL),Y
F4BC 45 1C    193         eor COLBITS
F4BE 25 30    194         and COLOR
F4C0 D0 02    195         bne >2
F4C2          196 ;
F4C2 E6 EA    197 ^1      inc HRCOLCNT           ; collision counter
F4C4          198 ;
F4C4          199 ;
F4C4          200 ; Common XDRAW/DRAW code.
F4C4          201 ;
F4C4          202 ^2      eor (GBASL),Y
F4C6 91 26    203         sta (GBASL),Y
F4C8          204 ;
F4C8          205 ;
F4C8          206 ; C-flag clear in horz summation and set in vert summation.
F4C8          207 ;
F4C8 A5 D0    208 ^3      lda SHPVAL
F4CA 65 D1    209         adc ROTQVAL
F4CC          210 ;
F4CC 29 03    211         and #MOVEMASK
F4CE C9 02    212         cmp #2                 ; for down or left
F4D0          213 ;
F4D0          214 ;
F4D0          215 ; ROR sets MSB if move direction is down or left and C-flag
F4D0          216 ; is set if move direction is right or left.
F4D0          217 ;
F4D0 6A      218         ror
F4D1 B0 92    219         bcs HRMOVLF
F4D3          220 ;
F4D3          221 ;
F4D3 30 30    222 HRMOVUP  bmi HRMOVND           ; branch if down
F4D5          223 ;
F4D5          224 ;
F4D5          225 ; Move up one scan line.  Enters with C-flag clear.
F4D5          226 ; Scan line 00 at 0x2000 -> 0x3FD0.
F4D5          227 ;
F4D5 A5 27    228         lda GBASH
F4D7 2C 04 F5 229         bit BITBYT1C
F4DA D0 22    230         bne >8
F4DC          231 ;
F4DC 06 26    232         asl GBASL           ; 00, 08, 10, 18, 20, 28 ...
F4DE B0 1A    233         bcs >6
F4E0          234 ;
F4E0 2C 03 F5 235         bit BITBYT03       ; 00, 10, 20, 30, 40, 50 ...
F4E3 F0 05    236         beq >4
F4E5          237 ;
F4E5 69 1F    238         adc #$1F           ; 10, 20, 30, 50, 60, 70 ...
F4E7          239 ;
F4E7 38      240         sec
F4E8 B0 12    241         bcs >7           ; always taken
F4EA          242 ;
F4EA 69 23    243 ^4      adc #$23           ; 00, 40, 80 only

```

```

F4EC 48          244          pha
F4ED          245          ;
F4ED A5 26      246          lda GBASL
F4EF 69 B0      247          adc #$B0
F4F1 B0 02      248          bcs >5
F4F3          249          ;
F4F3 69 F0      250          adc #$F0          ; 00 only
F4F5          251          ;
F4F5 85 26      252          ^5          sta GBASL
F4F7          253          ;
F4F7 68          254          pla
F4F8          255          ;
F4F8 B0 02      256          bcs >7          ; always taken
F4FA          257          ;
F4FA 69 1F      258          ^6          adc #$1F
F4FC          259          ;
F4FC 66 26      260          ^7          ror GBASL
F4FE          261          ;
F4FE 69 FC      262          ^8          adc #!-4          ; subtract 4 effectively
F500          263          ;
F500 85 27      264          sta GBASH
F502          265          ;
F502 60          266          rts
F503          267          ;
F503          268          ;
F503 03          269          BITBYT03 hex 03
F504 1C          270          BITBYT1C hex 1C
F505          271          ;
F505          272          ;
F505          273          ; Move down one scan line. Enters with C-flag clear.
F505          274          ; Scan line BF at 0x3FD0 -> 0x2000.
F505          275          ;
F505 A5 27      276          HRMOVDN  lda GBASH
F507 69 04      277          adc #4
F509          278          ;
F509 2C 04 F5    279          bit BITBYT1C
F50C D0 20      280          bne >4
F50E          281          ;
F50E 06 26      282          asl GBASL          ; 07, 0F, 17, 1F, 27, 2F ...
F510 90 18      283          bcc >2
F512          284          ;
F512 69 E0      285          adc #$E0          ; 0F, 1F, 2F, 3F, 4F, 5F ...
F514          286          ;
F514 18          287          clc
F515          288          ;
F515 2C 31 F5    289          bit BITBYT04
F518 F0 12      290          beq >3
F51A          291          ;
F51A A5 26      292          lda GBASL          ; 3F, 7F, BF only
F51C 69 50      293          adc #$50
F51E          294          ;
F51E 49 F0      295          eor #$F0
F520 F0 02      296          beq >1
F522          297          ;
F522 49 F0      298          eor #$F0          ; 3F, 7F only
F524          299          ;
F524 85 26      300          ^1          sta GBASL
F526          301          ;
F526 A5 E6      302          lda HRPAG
F528          303          ;
F528 90 02      304          bcc >3

```

```

F52A      305 ;
F52A 69 E0 306 ^2      adc #$E0
F52C      307 ;
F52C 66 26 308 ^3      ror GBASL
F52E      309 ;
F52E 85 27 310 ^4      sta GBASH
F530      311 ;
F530 60    312          rts
F531      313 ;
F531      314 ;
F531 04    315 BITBYT04 hex 04
F532      316 ;
F532      317 ;
F532 81    318 BITABLE   byt %10000001
F533 82    319          byt %10000010
F534 84    320          byt %10000100
F535 88    321          byt %10001000
F536 90    322          byt %10010000
F537 A0    323          byt %10100000
F538 C0    324          byt %11000000
F539      325 ;
F539      326 ;
F539      327          dfs 1,ZERO          ; 1 byte
F53A      328 ;
F53A      329 ;
F53A      330 ; Enter HLIN routine with the following end coordinates:
F53A      331 ;
F53A      332 ; A-reg, HIRES X-coordinate, low order byte (0-255)
F53A      333 ; X-reg, HIRES X-coordinate, high order byte (256-279)
F53A      334 ; Y-reg, HIRES Y-coordinate byte (0-191)
F53A      335 ;
F53A      336 ; This routine is modified using the HLINMOD directive as
F53A      337 ; documented in DOS 4.5 Volume and File Disk Management
F53A      338 ; System Second Edition.
F53A      339 ;
F53A 48    340 HLIN      pha
F53B      341 ;
F53B 38    342          sec
F53C      343 ;
F53C E5 E0 344          sbc HRXCOOR
F53E 48    345          pha
F53F      346 ;
F53F 8A    347          txa
F540      348 ;
F540 E5 E1 349          sbc HRXCOOR+1
F542 85 D3 350          sta HRXEND
F544 B0 0A 351          bcs >1
F546      352 ;
F546 68    353          pla          ; make 2's compliment
F547 49 FF 354          eor #NEGONE
F549 69 01 355          adc #1
F54B 48    356          pha
F54C      357 ;
F54C A9 00 358          lda #ZERO
F54E E5 D3 359          sbc HRXEND
F550      360 ;
F550 85 D1 361 ^1      sta HRXSTRT+1
F552 85 D5 362          sta HRYEND
F554      363 ;
F554 68    364          pla
F555 85 D0 365          sta HRXSTRT

```

```

F557 85 D4      366      sta HRXEND+1
F559           367      ;
F559 68         368      pla
F55A 85 E0      369      sta HRXCOOR
F55C 86 E1      370      stx HRXCOOR+1
F55E           371      ;
F55E 18         372      clc
F55F           373      ;
F55F 98         374      tya
F560 E5 E2      375      sbc HRYCOOR
F562 90 04      376      bcc >2
F564           377      ;
F564 49 FF      378      eor #NEGONE
F566 69 FE      379      adc #!-2
F568           380      ;
F568 85 D2      381      ^2      sta HRYSTRT
F56A 84 E2      382      sty HRYCOOR
F56C           383      ;
F56C 66 D3      384      ror HRXEND
F56E           385      ;
F56E 38         386      sec
F56F           387      ;
F56F E5 D0      388      sbc HRXSTRT
F571 AA         389      tax
F572           390      ;
F572 A9 FF      391      lda #NEGONE
F574 E5 D1      392      sbc HRXSTRT+1
F576 85 1D      393      sta COLCOUNT
F578           394      ;
F578 A4 E5      395      ldy HRHORZ
F57A           396      ;
F57A           397      .if HLINMOD
F57A           398      ;
F57A B0 04      399      bcs >4
F57C           400      ;
F57C 0A         401      ^3      asl
F57D           402      ;
F57D 20 65 F4   403      jsr HRMOVLF
F580           404      ;
F580 18         405      ^4      clc
F581           406      ;
F581 A5 D4      407      lda HRXEND+1
F583           408      ;
F583           409      .el
F583           410      ;
F583           411      bcs >4
F583           412      ;
F583           413      ^3      asl
F583           414      ;
F583           415      jsr HRMOVLF
F583           416      ;
F583           417      sec
F583           418      ;
F583           419      ^4      lda HRXEND+1
F583           420      ;
F583           421      .fi
F583           422      ;
F583 65 D2      423      adc HRYSTRT
F585 85 D4      424      sta HRXEND+1
F587           425      ;
F587 A5 D5      426      lda HRYEND

```

```

F589 E9 00      427      sbc #ZERO
F58B           428      ;
F58B 85 D5      429      ^5      sta HRYEND
F58D           430      ;
F58D B1 26      431      lda (GBASL),Y
F58F 45 1C      432      eor COLBITS
F591 25 30      433      and COLOR
F593           434      ;
F593 51 26      435      eor (GBASL),Y
F595 91 26      436      sta (GBASL),Y
F597           437      ;
F597 E8         438      inx
F598 D0 04      439      bne >6
F59A           440      ;
F59A E6 1D      441      inc COLCOUNT
F59C F0 14      442      beq >7
F59E           443      ;
F59E A5 D3      444      ^6      lda HRXEND
F5A0           445      ;
F5A0 B0 DA      446      bcs <3
F5A2           447      ;
F5A2 20 D3 F4   448      jsr HRMOVUP
F5A5           449      ;
F5A5           450      .if HLINMOD
F5A5           451      ;
F5A5 38         452      sec
F5A6           453      ;
F5A6           454      .el
F5A6           455      ;
F5A6           456      clc
F5A6           457      ;
F5A6           458      .fi
F5A6           459      ;
F5A6 A5 D4      460      lda HRXEND+1
F5A8 65 D0      461      adc HRXSTRT
F5AA 85 D4      462      sta HRXEND+1
F5AC           463      ;
F5AC A5 D5      464      lda HRYEND
F5AE 65 D1      465      adc HRXSTRT+1
F5B0           466      ;
F5B0 50 D9      467      bvc <5          ; always taken
F5B2           468      ;
F5B2 60         469      ^7      rts          ; added, HFIND removed
F5B3           470      ;
F5B3           471      ;
F5B3           472      ; Rotational values in 5.625 degrees based on 360/64.
F5B3           473      ; These values are COS( 90 * X/16 ) * 0x100 rather than
F5B3           474      ; * 0xFF. These value are more precise.
F5B3           475      ;
F5B3 00         476      ROTATBL hex 00          ; no rotation
F5B4 FF         477      hex FF
F5B5 FB         478      hex FB
F5B6 F5         479      hex F5
F5B7 ED         480      hex ED
F5B8 E2         481      hex E2
F5B9 D5         482      hex D5
F5BA C6         483      hex C6
F5BB B5         484      hex B5          ; 45 degrees
F5BC A2         485      hex A2
F5BD 8E         486      hex 8E
F5BE 79         487      hex 79

```

```

F5BF 62          488          hex 62
F5C0 4A          489          hex 4A
F5C1 32          490          hex 32
F5C2 19          491          hex 19
F5C3 00          492          hex 00          ; no rotation
F5C4            493          ;
F5C4            494          ;
F5C4            495          ; DRAW and XDRAW establish OPRND and fall into DRAWCMD.
F5C4            496          ; DRAWCMD is followed by DRAWSHP which was duplicated.
F5C4            497          ; Now, DRAWHDR uses two separate headers that is based
F5C4            498          ; on the MSB of OPRND which was set by DRAW or XDRAW.
F5C4            499          ;
F5C4 18          500  BDRAW    clc
F5C5            501          ;
F5C5 B0 00       502          bcs  *+2
F5C7            503          dfs  !-1
F5C6            504          ;
F5C6 38          505  BXDRAW   sec
F5C7            506          ;
F5C7 66 44       507          ror  OPRND
F5C9            508          ;
F5C9            509          ;
F5C9            510          ; This is the DRAWCMD routine.
F5C9            511          ;
F5C9 20 F8 E6    512          jsr  GETBYT          ; get requested SHAPE number
F5CC            513          ;
F5CC A5 E8       514          lda  HRSHTPTBL
F5CE 85 1A       515          sta  SHAPE
F5D0            516          ;
F5D0 A5 E9       517          lda  HRSHTPTBL+1
F5D2 85 1B       518          sta  SHAPE+1
F5D4            519          ;
F5D4 8A          520          txa          ; SHAPE number to draw
F5D5            521          ;
F5D5 A2 00       522          ldx  #ZERO
F5D7            523          ;
F5D7 C1 1A       524          cmp  (SHAPE,X)
F5D9 F0 05       525          beq  >3
F5DB            526          ;
F5DB 90 03       527          bcc  >3
F5DD            528          ;
F5DD 4C 06 F2    529          jmp  GOIQERR
F5E0            530          ;
F5E0 0A          531  ^3      asl
F5E1 90 03       532          bcc  >4
F5E3            533          ;
F5E3 E6 1B       534          inc  SHAPE+1
F5E5            535          ;
F5E5 18          536          clc
F5E6            537          ;
F5E6 A8          538  ^4      tay
F5E7            539          ;
F5E7 B1 1A       540          lda  (SHAPE),Y
F5E9 65 E8       541          adc  HRSHTPTBL
F5EB AA          542          tax
F5EC            543          ;
F5EC C8          544          iny
F5ED            545          ;
F5ED B1 1A       546          lda  (SHAPE),Y
F5EF 65 E9       547          adc  HRSHTPTBL+1
F5F1            548          ;

```

```

F5F1 86 1A      549      stx SHAPE
F5F3 85 1B      550      sta SHAPE+1
F5F5           551      ;
F5F5 20 B7 00   552      jsr CHRGOT
F5F8           553      ;
F5F8 C9 C5      554      cmp #TKAT
F5FA D0 09      555      bne >5
F5FC           556      ;
F5FC 20 C0 DE   557      jsr SYNTAXCHK
F5FF 20 B9 F6   558      jsr GETFNS          ; get coordinants
F602 20 11 F4   559      jsr HPOSN          ; set GBASL/GBASH, E0:E2, E5
F605           560      ;
F605           561      ;
F605           562      ; This is the DRAWSHP routine.
F605           563      ;
F605 A5 F9      564      ^5      lda HRROT
F607           565      ;
F607 4A         566      lsr
F608 4A         567      lsr
F609 4A         568      lsr
F60A 4A         569      lsr
F60B           570      ;
F60B 85 D1      571      sta ROTQVAL
F60D           572      ;
F60D A5 F9      573      lda HRROT
F60F 29 0F      574      and #SROTMASK
F611 AA         575      tax
F612           576      ;
F612 BC B3 F5   577      ldy ROTATBL,X
F615 88         578      dey
F616 84 D2      579      sty ROTHVAL
F618           580      ;
F618 49 0F      581      eor #SROTMASK
F61A AA         582      tax
F61B           583      ;
F61B BC B4 F5   584      ldy ROTATBL+1,X
F61E 84 D3      585      sty ROTVVAL
F620           586      ;
F620 A4 E5      587      ldy HRHORZ
F622           588      ;
F622 A2 FF      589      ldx #NEGONE
F624 86 D7      590      stx SHPOLD
F626           591      ;
F626 E8         592      inx
F627 86 EA      593      stx HRCOLCNT          ; initialize to zero
F629           594      ;
F629 A1 1A      595      lda (SHAPE,X)
F62B           596      ;
F62B           597      ;
F62B           598      ; If there is no change in direction, keep summing axes.
F62B           599      ;
F62B 85 D0      600      ^1      sta SHPVAL
F62D 29 07      601      and #SCMDMASK
F62F           602      ;
F62F C5 D7      603      cmp SHPOLD
F631 F0 08      604      beq >2
F633           605      ;
F633 85 D7      606      sta SHPOLD
F635           607      ;
F635 A9 00      608      lda #ZERO
F637 85 D4      609      sta ROTHSUM

```

```

F639 85 D5      610      sta ROTVSUM
F63B           611      ;
F63B A6 E7      612      ^2      ldx HRSCALE
F63D           613      ;
F63D           614      ;
F63D           615      ; Horizontal summation.
F63D           616      ;
F63D 38         617      ^3      sec
F63E           618      ;
F63E A5 D4      619      lda ROTHSUM
F640 65 D2      620      adc ROTHVAL
F642 85 D4      621      sta ROTHSUM
F644 90 05      622      bcc >4
F646           623      ;
F646 18         624      clc
F647           625      ;
F647 20 9E F4   626      jsr DRAWHDR
F64A           627      ;
F64A 18         628      clc
F64B           629      ;
F64B           630      ;
F64B           631      ; Vertical summation.
F64B           632      ;
F64B A5 D5      633      ^4      lda ROTVSUM
F64D 65 D3      634      adc ROTVVAL
F64F 85 D5      635      sta ROTVSUM
F651 90 03      636      bcc >5
F653           637      ;
F653 20 9E F4   638      jsr DRAWHDR
F656           639      ;
F656 CA         640      ^5      dex
F657 D0 E4      641      bne <3
F659           642      ;
F659 A5 D0      643      lda SHPVAL
F65B           644      ;
F65B 4A         645      lsr
F65C 4A         646      lsr
F65D 4A         647      lsr
F65E           648      ;
F65E D0 CB      649      bne <1
F660           650      ;
F660           651      ;
F660           652      ; Point to next shape in table.
F660           653      ;
F660 E6 1A      654      inc SHAPE
F662 D0 02      655      bne >6
F664           656      ;
F664 E6 1B      657      inc SHAPE+1
F666           658      ;
F666 A1 1A      659      ^6      lda (SHAPE,X)
F668 D0 C1      660      bne <1
F66A           661      ;
F66A 60         662      rts
F66B           663      ;
F66B           664      ;
F66B 20 21 EB   665      FSQR2   jsr COPYF2T2
F66E           666      ;
F66E A9 93      667      lda #TEMP1
F670 A0 00      668      ldy /TEMP1
F672 20 66 EA   669      jsr FDIV
F675           670      ;

```



```

F675 A9 98      671      lda #TEMP2
F677 A0 00      672      ldy /TEMP2
F679 20 BE E7   673      jsr FADD
F67C           674      ;
F67C C6 9D      675      dec FACEXP
F67E E6 47      676      inc YREG
F680           677      ;
F680 A9 98      678      lda #TEMP2
F682 A0 00      679      ldy /TEMP2
F684           680      ;
F684 20 B2 EB   681      jsr FPCOMP
F687 D0 E2      682      bne FSQR2
F689           683      ;
F689 60         684      rts
F68A           685      ;
F68A           686      ;
F68A A5 A7      687 COPYA2F3 lda ARGMANT+1
F68C 85 9F      688      sta FACMANT+1
F68E           689      ;
F68E A5 A6      690      lda ARGMANT
F690 85 9E      691      sta FACMANT
F692           692      ;
F692 A5 A5      693      lda ARGEXP
F694 85 9D      694      sta FACEXP
F696           695      ;
F696 A2 00      696      ldx #ZERO
F698 86 AC      697      stx FACGUARD
F69A           698      ;
F69A 60         699      rts
F69B           700      ;
F69B           701      ;
F69B A5 A0      702 COPYF2A3 lda FACMANT+2
F69D 85 A8      703      sta ARGMANT+2
F69F           704      ;
F69F A5 9F      705      lda FACMANT+1
F6A1 85 A7      706      sta ARGMANT+1
F6A3           707      ;
F6A3 A5 9E      708      lda FACMANT
F6A5 85 A6      709      sta ARGMANT
F6A7           710      ;
F6A7 A5 9D      711      lda FACEXP
F6A9 85 A5      712      sta ARGEXP
F6AB           713      ;
F6AB A2 00      714      ldx #ZERO
F6AD 86 AC      715      stx FACGUARD
F6AF           716      ;
F6AF 60         717      rts
F6B0           718      ;
F6B0           719      ;
F6B0           720      dfs 9,ZERO          ; 9 bytes
F6B9           721      ;
F6B9           722      ;
F6B9           723      ; Get HPOSN coordinates: X-reg/Y-reg = HORZ, A-reg = VERT.
F6B9           724      ;
F6B9 20 67 DD   725 GETFNS  jsr FRMNUM
F6BC 20 52 E7   726      jsr GETADDR
F6BF           727      ;
F6BF A6 50      728      ldx ACL
F6C1 A4 51      729      ldy ACH
F6C3           730      ;
F6C3 C0 01      731      cpy #1

```

```

F6C5 90 06      732      bcc >1
F6C7            733      ;
F6C7 D0 1D      734      bne DOIQERR
F6C9            735      ;
F6C9 E0 18      736      cpx #24
F6CB B0 19      737      bcs DOIQERR
F6CD            738      ;
F6CD 8A         739      ^1      txa
F6CE 48         740      pha
F6CF            741      ;
F6CF 98         742      tya
F6D0 48         743      pha
F6D1            744      ;
F6D1 A9 2C      745      lda #', '
F6D3 20 C0 DE   746      jsr SYNTAXCHK
F6D6            747      ;
F6D6 20 F8 E6   748      jsr GETBYT
F6D9            749      ;
F6D9 E0 C0      750      cpx #192      ; maximum vertical line
F6DB B0 09      751      bcs DOIQERR
F6DD            752      ;
F6DD 86 9D      753      stx DSCTMP
F6DF            754      ;
F6DF 68         755      pla
F6E0 A8         756      tay
F6E1            757      ;
F6E1 68         758      pla
F6E2 AA         759      tax
F6E3            760      ;
F6E3 A5 9D      761      lda DSCTMP
F6E5            762      ;
F6E5 60         763      rts
F6E6            764      ;
F6E6            765      ;
F6E6 4C 06 F2   766      DOIQERR  jmp GOIQERR
F6E9            767      ;
F6E9            768      ;
F6E9 20 F8 E6   769      BHCOLOR  jsr GETBYT
F6EC            770      ;
F6EC E0 08      771      cpx #8
F6EE B0 F6      772      bcs DOIQERR
F6F0            773      ;
F6F0 BD F6 F6   774      lda HRCOLTBL,X
F6F3 85 E4      775      sta HRCOLOR
F6F5            776      ;
F6F5 60         777      RTN.F6.F  rts
F6F6            778      ;
F6F6            779      ;
F6F6 00 2A 55   780      HRCOLTBL  hex 002A557F
F6F9 7F         781
F6FA 80 AA D5   781      hex 80AAD5FF
F6FD FF
F6FE            782      ;
F6FE            783      ;
F6FE C9 C1      784      BHPlot    cmp #TKTO
F700 F0 0D      785      beq >2
F702            786      ;
F702 20 B9 F6   787      jsr GETFNS
F705 20 57 F4   788      jsr HPlot
F708            789      ;
F708 20 B7 00   790      BHPlot2  jsr CHRGOT

```

```

F70B          791  ;
F70B C9 C1    792      cmp #TKTO
F70D D0 E6    793      bne RTN.F6.F
F70F          794  ;
F70F 20 C0 DE 795  ^2      jsr SYNTAXCHK
F712 20 B9 F6 796      jsr GETFNS
F715          797  ;
F715 84 9D    798      sty DSCTMP
F717          799  ;
F717 A8       800      tay
F718 8A       801      txa
F719          802  ;
F719 A6 9D    803      ldx DSCTMP
F71B          804  ;
F71B 20 3A F5 805      jsr HLIN
F71E          806  ;
F71E 4C 08 F7 807      jmp BHPLLOT2
F721          808  ;
F721          809  ;
F721 20 F8 E6 810  BROT      jsr GETBYT
F724 86 F9    811      stx HRROT
F726          812  ;
F726 60       813      rts
F727          814  ;
F727          815  ;
F727 20 F8 E6 816  BSCALE      jsr GETBYT
F72A 86 E7    817      stx HRSCALE
F72C          818  ;
F72C 60       819      rts
F72D          820  ;
F72D          821  ;
F72D          822  ; Continuation of the FRND routine.
F72D          823  ;
F72D A2 03    824  FRND2      ldx #3
F72F          825  ;
F72F BD A4 EF 826  ^7      lda RANDVAL1,X
F732 95 62    827      sta MULMANT,X
F734          828  ;
F734 BD A8 EF 829      lda RANDVAL2,X
F737 95 9E    830      sta FACMANT,X
F739          831  ;
F739 B5 C9    832      lda IRAND,X
F73B 95 A6    833      sta ARGMANT,X
F73D          834  ;
F73D CA       835      dex
F73E 10 EF    836      bpl <7
F740          837  ;
F740          838  ;
F740          839  ; Multiply ARGMANT and MULMANT using the Peasant algorithm
F740          840  ; for multiplication and save result into FAC and IRAND.
F740          841  ;
F740 A0 20    842      ldy #32
F742          843  ;
F742 46 62    844  ^8      lsr MULMANT
F744 66 63    845      ror MULMANT+1
F746 66 64    846      ror MULMANT+2
F748 66 65    847      ror MULMANT+3
F74A          848  ;
F74A 90 0E    849      bcc >1
F74C          850  ;
F74C 18       851      clc

```

```

F74D      852 ;
F74D A2 03 853      ldx #3
F74F      854 ;
F74F B5 9E 855 ^9      lda FACMANT,X
F751 75 A6 856      adc ARGMANT,X
F753 95 9E 857      sta FACMANT,X
F755 95 C9 858      sta IRAND,X
F757      859 ;
F757 CA    860      dex
F758 10 F5 861      bpl <9
F75A      862 ;
F75A 06 A9 863 ^1      asl ARGMANT+3
F75C 26 A8 864      rol ARGMANT+2
F75E 26 A7 865      rol ARGMANT+1
F760 26 A6 866      rol ARGMANT
F762      867 ;
F762 88    868      dey
F763 D0 DD 869      bne <8
F765      870 ;
F765 84 A2 871      sty FACSIGN
F767 84 AC 872      sty FACGUARD
F769      873 ;
F769      874 ;
F769      875 ; Convert the value in FAC to a floating point fraction.
F769      876 ;
F769 A9 80 877      lda #$80
F76B 85 9D 878      sta FACEXP
F76D      879 ;
F76D 20 2E E8 880      jsr NORMFAC1
F770      881 ;
F770      882 ;
F770      883 ; If Range is less than two, return the value in FAC as is,
F770      884 ; otherwise multiply by Range and return an integer.
F770      885 ;
F770 A5 93 886      lda TEMP1
F772 C9 82 887      cmp #$82
F774 90 0A 888      bcc >2 ; branch if Range < 2
F776      889 ;
F776 A9 93 890      lda #TEMP1
F778 A0 00 891      ldy /TEMP1
F77A 20 7F E9 892      jsr FMULT
F77D      893 ;
F77D 20 23 EC 894      jsr FINT
F780      895 ;
F780 60    896 ^2      rts
F781      897 ;
F781      898 ;
F781      899      dfs $F78F-*,ZERO
F78F      900 ;
F78F      901 ;
F78F C1 F0 F0 902 TITLE asc "Apple //e+"
F792 EC E5 A0
F795 AF AF E5
F798 AB
F799      903 ;
000A      904 TITLEN equ *-TITLE
001C      905 DELTITLE equ 40-TITLEN+2
000E      906 OFFTITLE equ DELTITLE/2
F799      907 ;
F799      908 ;
F799      909 ; Modifications to Applesoft in order to support an 80

```

```

F799          910 ; column display.
F799          911 ;
F799 BD 01 02 912 PARSIEX1 lda INPUT+1,X
F79C 10 11    913          bpl >1
F79E          914 ;
F79E A5 0E    915 PARSIEX2 lda ENDCHR
F7A0 F0 16    916          beq >3
F7A2          917 ;
F7A2 C9 22    918          cmp #'"'
F7A4 F0 12    919          beq >3
F7A6          920 ;
F7A6 A5 13    921          lda DATAFLG
F7A8 C9 49    922          cmp #$49
F7AA F0 0C    923          beq >3
F7AC          924 ;
F7AC BD 00 02 925 PARSIEX3 lda INPUT,X
F7AF          926 ;
F7AF 08       927 ^1      php
F7B0          928 ;
F7B0 C9 61    929          cmp #$61
F7B2 90 02    930          bcc >2
F7B4          931 ;
F7B4 29 5F    932          and #$5F
F7B6          933 ;
F7B6 28       934 ^2      plp
F7B7          935 ;
F7B7 60       936          rts
F7B8          937 ;
F7B8 BD 00 02 938 ^3      lda INPUT,X
F7BB          939 ;
F7BB 60       940          rts
F7BC          941 ;
F7BC          942 ;
F7BC 48       943 BLISTEX1 pha
F7BD          944 ;
F7BD A9 20    945          lda #SPACE&MSBCLR
F7BF 20 5C DB 946          jsr OUTCHR
F7C2          947 ;
F7C2 68       948          pla
F7C3          949 ;
F7C3 4C 24 ED 950          jmp LINPRT
F7C6          951 ;
F7C6          952 ;
F7C6          953 BLISTEX2:
F7C6 A5 24    954 PRTCREX1 lda CH
F7C8 C9 21    955          cmp #$21
F7CA          956 ;
F7CA 2C 1F C0 957          bit RDVID80
F7CD 10 05    958          bpl >4
F7CF          959 ;
F7CF AD 7B 05 960          lda OURCH
F7D2 C9 49    961          cmp #$49
F7D4          962 ;
F7D4 60       963 ^4      rts
F7D5          964 ;
F7D5          965 ;
F7D5 8A       966 PRTCREX2 txa
F7D6          967 ;
F7D6 2C 1F C0 968          bit RDVID80
F7D9 30 08    969          bmi >5
F7DB          970 ;

```

```

F7DB 2C 00 00    971          bit  *-*
F7DE             972          dfs  !-2
F7DC             973          ;
F7DC 85 24       974  BHTABEX1 sta  CH
F7DE             975          ;
F7DE 38          976          sec
F7DF             977          ;
F7DF 8A          978          txa
F7E0 E5 24       979          sbc  CH
F7E2             980          ;
F7E2 60          981          rts
F7E3             982          ;
F7E3 ED 7B 05    983  ^5      sbc  OURCH
F7E6             984          ;
F7E6 60          985  RTN.F7.E rts
F7E7             986          ;
F7E7             987          ;
F7E7 20 F8 E6    988  BHTAB    jsr  GETBYT
F7EA             989          ;
F7EA CA          990          dex
F7EB             991          ;
F7EB A9 28       992  ^1      lda  #40
F7ED C5 21       993          cmp  WNDWDTH
F7EF B0 02       994          bcs  >2
F7F1             995          ;
F7F1 A5 21       996          lda  WNDWDTH
F7F3             997          ;
F7F3 20 DC F7    998  ^2      jsr  BHTABEX1
F7F6             999          ;
F7F6 86 24      1000          stx  CH
F7F8             1001         ;
F7F8 90 EC      1002          bcc  RTN.F7.E
F7FA             1003         ;
F7FA AA        1004          tax
F7FB             1005         ;
F7FB 20 FB DA   1006          jsr  PRTCR
F7FE D0 EB      1007          bne  <1          ; always taken
F800             1008         ;
F800             1009         ;
F800             1010          icl  "F8.L"

```

```

LLOAD F8.L,A$4000

```

```

F800          1          ttl "ROM Source Code, F8.L"
F800          2          ;
F800          3          ;
F800          4          ; F8.L
F800          5          ;
F800          6          ;
F800          7          F8SPACE:
F800          8          ;
F800          9          ;
F800 4A       10         PLOT      lsr
F801 08       11          php                      ; save LSB in C-flag
F802         12          ;
F802 20 47 F8 13          jsr GBASCALC
F805         14          ;
F805 28       15          plp                      ; recall LSB
F806         16          ;
F806 A9 0F    17          lda #$0F                  ; mask 0x0F if even
F808         18          ;
F808 90 02    19          bcc RTMASK
F80A         20          ;
F80A 69 E0    21          adc #$E0                  ; mask 0xF0 if odd
F80C         22          ;
F80C 85 2E    23         RTMASK   sta MASK
F80E         24          ;
F80E B1 26    25         PLOT1    lda (GBASL),Y      ; get data
F810 45 30    26          eor COLOR                 ; apply color
F812         27          ;
F812 25 2E    28          and MASK                 ; select bits
F814         29          ;
F814 51 26    30          eor (GBASL),Y             ; apply data
F816 91 26    31          sta (GBASL),Y            ; save it
F818         32          ;
F818 60       33          rts
F819         34          ;
F819         35          ;
F819 20 00 F8 36         HLINE    jsr PLOT          ; plot square
F81C         37          ;
F81C C4 2C    38         HLINE1   cpy H2
F81E B0 11    39          bcs >2
F820         40          ;
F820 C8       41          iny
F821         42          ;
F821 20 0E F8 43          jsr PLOT1                 ; plot next square
F824 90 F6    44          bcc HLINE1
F826         45          ;
F826 69 01    46          ^1      adc #1             ; next Y-coord
F828         47          ;
F828 48       48         VLINE    pha
F829         49          ;
F829 20 00 F8 50          jsr PLOT          ; plot square
F82C         51          ;
F82C 68       52          pla
F82D C5 2D    53          cmp V2
F82F 90 F5    54          bcc <1
F831         55          ;
F831 60       56          ^2      rts
F832         57          ;
F832         58          ;
F832 A0 2F    59         CLRSCR   ldy #$2F          ; max Y, full scrn clr
F834 D0 02    60          bne >3

```

```

F836          61 ;
F836 A0 27    62 CLRTOP    ldy #$27          ; max Y, top scrn clr
F838          63 ;
F838 84 2D    64 ^3      sty V2
F83A          65 ;
F83A A0 27    66          ldy #$27          ; rightmost X-coord (column)
F83C          67 ;
F83C A9 00    68 ^4      lda #ZERO          ; top coord for VLINE
F83E 85 30    69          sta COLOR        ; clear color to black
F840          70 ;
F840 20 28 F8 71          jsr VLINE        ; draw VLINE
F843          72 ;
F843 88       73          dey
F844 10 F6    74          bpl <4
F846          75 ;
F846 60       76          rts
F847          77 ;
F847          78 ;
F847 48       79 GBASCALC pha          ; for input ODDEFGH
F848          80 ;
F848 4A       81          lsr
F849          82 ;
F849 29 03    83          and #3
F84B 09 04    84          ora #4
F84D 85 27    85          sta GBASH        ; = 000001FG
F84F          86 ;
F84F 68       87          pla          ; = HDEDE000
F850 29 18    88          and #$18
F852          89 ;
F852 90 02    90          bcc >5
F854          91 ;
F854 69 7F    92          adc #$7F
F856          93 ;
F856 85 26    94 ^5      sta GBASL
F858          95 ;
F858 0A       96          asl
F859 0A       97          asl
F85A          98 ;
F85A 05 26    99          ora GBASL
F85C 85 26   100          sta GBASL
F85E          101 ;
F85E 60       102          rts
F85F          103 ;
F85F          104 ;
F85F A5 30   105 NXTCOL    lda COLOR          ; increment by 3
F861          106 ;
F861 18       107          clc
F862          108 ;
F862 69 03    109          adc #3
F864          110 ;
F864 29 0F    111 SETCOL    and #$0F          ; COLOR=17*A MOD 16m 114
F866 85 30    112          sta COLOR
F868          113 ;
F868 0A       114          asl          ; both half bytes equal
F869 0A       115          asl
F86A 0A       116          asl
F86B 0A       117          asl
F86C          118 ;
F86C 05 30    119          ora COLOR
F86E 85 30    120          sta COLOR
F870          121 ;

```



```

F870 60          122          rts
F871          123          ;
F871          124          ;
F871 4A          125  SCRN    lsr                ; screen Y-coord/2
F872 08          126          php                ; save LSB
F873          127          ;
F873 20 47 F8    128          jsr GBASCALC        ; calc base address
F876          129          ;
F876 B1 26       130          lda (GBASL),Y      ; get data
F878          131          ;
F878 28          132          plp                ; recall LSB
F879          133          ;
F879 90 04       134  SCRN2   bcc >6            ; if even use LO half
F87B          135          ;
F87B 4A          136          lsr                ; shift HI half down
F87C 4A          137          lsr
F87D 4A          138          lsr
F87E 4A          139          lsr
F87F          140          ;
F87F 29 0F       141  ^6      and #$0F          ; mask 4 bits
F881          142          ;
F881 60          143          rts
F882          144          ;
F882          145          ;
F882 A6 3A       146  INSDS1   ldx PCL            ; print PCL,PCH
F884 A4 3B       147          ldy PCH
F886          148          ;
F886 20 96 FD    149          jsr PRXY
F889 20 48 F9    150          jsr PRBLNK
F88C          151          ;
F88C A1 3A       152          lda (PCL,X)        ; get opcode
F88E          153          ;
F88E A8          154  INSDS2   tay
F88F          155          ;
F88F 4A          156          lsr                ; even/odd test
F890 90 05       157          bcc >7
F892          158          ;
F892 6A          159          ror                ; bit 1 test
F893 B0 0C       160          bcs >8            ; XXXXXX11 invalid OP
F895          161          ;
F895 29 87       162          and #$87          ; mask bits
F897          163          ;
F897 4A          164  ^7      lsr                ; C-flag = LSB for later
F898 AA          165          tax
F899          166          ;
F899 BD 62 F9    167          lda FMT1,X        ; get FORMAT index
F89C          168          ;
F89C 20 79 F8    169          jsr SCRN2
F89F D0 04       170          bne GETFMT
F8A1          171          ;
F8A1 A0 03       172  ^8      ldy #3            ; invalid OP
F8A3 A9 00       173          lda #ZERO         ; error FORMAT
F8A5          174          ;
F8A5          175          ;
F8A5          176          ; Move remaining code to 0xC1-0xC2 because the code that
F8A5          177          ; tests the ROM in slot 3 must be in the 0xF8 ROM space.
F8A5          178          ;
F8A5 AA          179  GETFMT   tax
F8A6          180          ;
F8A6 BD C7 FB    181          lda FMT2,X        ; get instruction format
F8A9 C9 49       182          cmp #$49         ; is it relative (zpage)

```

```

F8AB D0 01      183      bne >1
F8AD           184      ;
F8AD 88         185      dey                      ; correct index
F8AE           186      ;
F8AE AA         187      ^1      tax
F8AF           188      ;
F8AF 84 2A      189      sty BAS2L
F8B1           190      ;
F8B1 A0 10      191      ldy #16
F8B3           192      ;
F8B3 4C B4 FB   193      jmp GOTOROM
F8B6           194      ;
F8B6           195      ;
F8B6           196      ; Test slot 3 for a card containing a ROM.  If there is one
F8B6           197      ; the internal slot 3 firmware for 80 columns will not be
F8B6           198      ; used.  Y-reg is set to a reasonable value.
F8B6           199      ; Bytes checked are 0xC305 and 0xC307.
F8B6           200      ;
F8B6 8D 06 C0   201      TESTROM sta CXROMOFF          ; enable slots
F8B9           202      ;
F8B9 A0 08      203      ldy #8
F8BB           204      ;
F8BB A2 02      205      ^1      ldx #2
F8BD           206      ;
F8BD BD 05 C3   207      ^2      lda BASICIN,X
F8C0 DD 9C FC   208      cmp CLREOL,X
F8C3 D0 07      209      bne >3
F8C5           210      ;
F8C5 CA         211      dex
F8C6 CA         212      dex
F8C7           213      ;
F8C7 10 F4      214      bpl <2
F8C9           215      ;
F8C9 88         216      dey
F8CA D0 EF      217      bne <1
F8CC           218      ;
F8CC 8D 07 C0   219      ^3      sta CXROMON          ; swap in internal ROM
F8CF           220      ;
F8CF 60         221      rts
F8D0           222      ;
F8D0           223      ;
F8D0           224      ; Disassemble instruction.
F8D0           225      ;
F8D0 20 82 F8   226      INSTDSP jsr INSDS1          ; get format, length
F8D3           227      ;
F8D3 48         228      pha
F8D4           229      ;
F8D4 B1 3A      230      PRNTOP  lda (PCL),Y
F8D6           231      ;
F8D6 20 DA FD   232      jsr PRBYTE
F8D9           233      ;
F8D9 A2 01      234      ldx #1                      ; print 2 blanks
F8DB           235      ;
F8DB 20 4A F9   236      PRNTBL  jsr PRBL2
F8DE           237      ;
F8DE C4 2F      238      cpy LENGTH
F8E0           239      ;
F8E0 C8         240      iny
F8E1           241      ;
F8E1 90 F1      242      bcc PRNTOP
F8E3           243      ;

```

```

F8E3 A2 03      244      ldx #3                ; 12 chr field
F8E5            245      ;
F8E5 C0 04      246      cpy #4
F8E7 90 F2      247      bcc PRNTBL
F8E9            248      ;
F8E9 68          249      pla
F8EA A8          250      tay
F8EB            251      ;
F8EB B9 A6 F9    252      lda MNEML,Y          ; get mnemonic
F8EE 85 2C      253      sta LMNEM
F8F0            254      ;
F8F0 B9 EB F9    255      lda MNEMR,Y
F8F3 85 2D      256      sta RMNEM
F8F5            257      ;
F8F5 A9 00      258      ^4      lda #ZERO
F8F7 A0 05      259      ldy #5
F8F9            260      ;
F8F9 06 2D      261      ^5      asl RMNEM
F8FB 26 2C      262      rol LMNEM
F8FD            263      ;
F8FD 2A          264      rol
F8FE            265      ;
F8FE 88          266      dey
F8FF D0 F8      267      bne <5
F901            268      ;
F901 69 BF      269      adc #"@"-1          ; add in offset
F903            270      ;
F903 20 ED FD    271      jsr COUT
F906            272      ;
F906 CA          273      dex
F907 D0 EC      274      bne <4
F909            275      ;
F909 20 48 F9    276      jsr PRBLNK          ; print 3 blanks
F90C            277      ;
F90C A4 2F      278      ldy LENGTH
F90E A2 06      279      ldx #6                ; 6 format bits
F910            280      ;
F910 E0 03      281      ^1      cpx #3
F912 F0 1C      282      beq >5
F914            283      ;
F914 06 2E      284      ^2      asl FORMAT
F916 90 0E      285      bcc >3
F918            286      ;
F918 BD 2F FA    287      lda CHAR1-1,X
F91B            288      ;
F91B 20 ED FD    289      jsr COUT
F91E            290      ;
F91E BD 35 FA    291      lda CHAR2-1,X
F921 F0 03      292      beq >3
F923            293      ;
F923 20 ED FD    294      jsr COUT
F926            295      ;
F926 CA          296      ^3      dex
F927 D0 E7      297      bne <1
F929            298      ;
F929 60          299      rts
F92A            300      ;
F92A            301      ;
F92A 88          302      ^4      dey
F92B 30 E7      303      bmi <2
F92D            304      ;

```

```

F92D 20 DA FD      305      jsr PRBYTE
F930                306      ;
F930 A5 2E        307      ^5      lda FORMAT
F932 C9 E8        308      cmp #$E8      ; relative address mode
F934                309      ;
F934 B1 3A        310      lda (PCL),Y      ; print target, not offset
F936                311      ;
F936 90 F2        312      bcc <4
F938                313      ;
F938 20 56 F9     314      jsr PCADJ3
F93B                315      ;
F93B AA          316      tax
F93C                317      ;
F93C E8          318      inx
F93D D0 01       319      bne PRNTYX
F93F                320      ;
F93F C8          321      iny
F940                322      ;
F940 98          323      PRNTYX      tya
F941                324      ;
F941 20 DA FD     325      PRNTAX      jsr PRBYTE
F944                326      ;
F944 8A          327      PRNTAX      txa
F945                328      ;
F945 4C DA FD     329      jmp PRBYTE
F948                330      ;
F948                331      ;
F948 A2 03       332      PRBLNK      ldx #3      ; blank count
F94A                333      ;
F94A A9 A0       334      PRBL2      lda #SPACE
F94C 20 ED FD     335      jsr COUT
F94F                336      ;
F94F CA          337      dex
F950 D0 F8       338      bne PRBL2
F952                339      ;
F952 60          340      rts
F953                341      ;
F953                342      ;
F953                343      ; A-reg = PCL + LENGTH + 1, carry into Y-reg (PCH).
F953                344      ;
F953 38          345      PCADJ      sec
F954                346      ;
F954 A5 2F       347      PCADJ2      lda LENGTH
F956                348      ;
F956 A4 3B       349      PCADJ3      ldy PCH
F958                350      ;
F958 AA          351      tax
F959 10 01       352      bpl >6
F95B                353      ;
F95B 88          354      dey
F95C                355      ;
F95C 65 3A       356      ^6      adc PCL
F95E 90 01       357      bcc RTS2
F960                358      ;
F960 C8          359      iny
F961                360      ;
F961 60          361      RTS2      rts
F962                362      ;
F962                363      ;
F962                364      ; Opcodes of the form XXXX XXY0:
F962                365      ;

```

```

F962      366 ; Use XXXXXX for index into FMT1.
F962      367 ;
F962      368 ; If Y=0, use lower nibble for index into FMT2.
F962      369 ; If Y=1, use upper nibble for index into FMT2.
F962      370 ;
F962      371 FMT1:
F962 04 22 54 372      hex 04225433ED824493
F965 33 ED 82
F968 44 93
F96A 03 22 54 373      hex 03225433ED884499
F96D 33 ED 88
F970 44 99
F972 04 20 54 374      hex 04205433ED804490
F975 33 ED 80
F978 44 90
F97A 04 22 54 375      hex 0422543BED88449F
F97D 3B ED 88
F980 44 9F
F982 0D 22 44 376      hex 0D224433EDC84493
F985 33 ED C8
F988 44 93
F98A 11 22 44 377      hex 11224433EDC844A9
F98D 33 ED C8
F990 44 A9
F992 01 22 44 378      hex 01224433ED804490
F995 33 ED 80
F998 44 90
F99A 01 22 44 379      hex 01224433ED804490
F99D 33 ED 80
F9A0 44 90
F9A2      380 ;
F9A2      381 ;
F9A2      382 ; Opcodes of the form ZZXX XY01:
F9A2      383 ;
F9A2 26 31 87 384      hex 2631879A
F9A5 9A
F9A6      385 ;
F9A6      386 ;
F9A6      387 ; Mnemonic extraction from MNEML and MNEMR:
F9A6      388 ;
F9A6      389 ; Table: <MNEML-><MNEMR->
F9A6      390 ; Bit: 7654321076543210
F9A6      391 ; -----
F9A6      392 ; Chr1 Chr2 Chr3
F9A6      393 ;
F9A6      394 ; Mnemonic = Chr1+0xBF Chr2+0xBF Chr3+0xBF
F9A6      395 ;
F9A6      396 MNEML:
F9A6 1C 8A 1C 397      hex 1C8A1C235D8B1BA1
F9A9 23 5D 8B
F9AC 1B A1
F9AE 9D 8A 1D 398      hex 9D8A1D239D8B1DA1
F9B1 23 9D 8B
F9B4 1D A1
F9B6 1C 29 19 399      hex 1C2919AE69A81923
F9B9 AE 69 A8
F9BC 19 23
F9BE 24 53 1B 400      hex 24531B23245319A1
F9C1 23 24 53
F9C4 19 A1
F9C6 AD 1A 5B 401      hex AD1A5B5BA5692424

```

```

F9C9 5B A5 69
F9CC 24 24
F9CE AE AE A8    402          hex AEAEA8AD298A7C8B
F9D1 AD 29 8A
F9D4 7C 8B
F9D6 15 9C 6D    403          hex 159C6D9CA5692953
F9D9 9C A5 69
F9DC 29 53
F9DE 84 13 34    404          hex 84133411A56923A0
F9E1 11 A5 69
F9E4 23 A0
F9E6 00 8A 8B    405          hex 008A8BACA5
F9E9 AC A5
F9EB              406          ;
F9EB              407          MNEMR:
F9EB D8 62 5A    408          hex D8625A4826629488
F9EE 48 26 62
F9F1 94 88
F9F3 54 44 C8    409          hex 5444C8546844E894
F9F6 54 68 44
F9F9 E8 94
F9FB C4 B4 08    410          hex C4B4088474B4286E
F9FE 84 74 B4
FA01 28 6E
FA03 74 F4 CC    411          hex 74F4CC4A72F2A48A
FA06 4A 72 F2
FA09 A4 8A
FA0B 06 AA A2    412          hex 06AAA2A274747472
FA0E A2 74 74
FA11 74 72
FA13 44 68 B2    413          hex 4468B232B2722272
FA16 32 B2 72
FA19 22 72
FA1B 1A 1A 26    414          hex 1A1A2626727288C8
FA1E 26 72 72
FA21 88 C8
FA23 C4 CA 26    415          hex C4CA26484444A2C8
FA26 48 44 44
FA29 A2 C8
FA2B 00 74 74    416          hex 007474C676
FA2E C6 76
FA30              417          ;
FA30              418          ;
FA30              419          ; Characters used to disassemble instructions.
FA30              420          ;
FA30 AC A9 AC    421          CHAR1    hex ACA9ACA3A8A4    ; ,),#($
FA33 A3 A8 A4
FA36 D9 00 D8    422          CHAR2    hex D900D8A4A400    ; Y X$$
FA39 A4 A4 00
FA3C              423          ;
FA3C              424          ;
FA3C 8D 06 C0    425          CXOFF    sta CXROMOFF    ; enable slots
FA3F              426          ;
FA3F 60          427          CXRTN    rts
FA40              428          ;
FA40              429          ;
FA40              430          ; This code is useless, and probably only supports the old
FA40              431          ; BREAK vector at 0xFA40. It assumes that RDCXROM is off.
FA40              432          ;
FA40 85 45        433          OLDIRQ    sta AREG
FA42 A5 45        434          lda AREG

```

```

FA44      435 ;
FA44 4C FA C3 436      jmp IRQRTN
FA47      437 ;
FA47      438 ;
FA47 8D 06 C0 439 NEWBREAK sta CXROMOFF      ; enable slots
FA4A      440 ;
FA4A 85 45      441      sta AREG
FA4C      442 ;
FA4C 28      443 BREAK      plp
FA4D      444 ;
FA4D 20 4C FF 445      jsr SAVE2      ; save registers
FA50      446 ;
FA50 68      447      pla
FA51 85 3A      448      sta PCL
FA53      449 ;
FA53 68      450      pla
FA54 85 3B      451      sta PCH
FA56      452 ;
FA56 6C F0 03 453      jmp (AUTOBRK+1)
FA59      454 ;
FA59      455 ;
FA59 20 82 F8 456 OLDBRK      jsr INSDS1      ; print PC
FA5C 20 DA FA 457      jsr REGDSP2      ; print registers
FA5F      458 ;
FA5F 4C 65 FF 459      jmp MON      ; enter Monitor
FA62      460 ;
FA62      461 ;
FA62 AD 58 C0 462 RESET      lda ANN1OFF
FA65 AD 5A C0 463      lda ANN2OFF
FA68      464 ;
FA68 AD 5D C0 465      lda ANN3ON
FA6B AD 5F C0 466      lda ANN4ON
FA6E      467 ;
FA6E A0 09      468      ldy #9      ; //e init
FA70 D0 0C      469      bne RESET2      ; always taken
FA72      470 ;
FA72      471 ;
FA72 8D 07 C0 472 SWEET16      sta CXROMON      ; enable CX ROM
FA75      473 ;
FA75 4C 70 C6 474      jmp SW16
FA78      475 ;
FA78 8D 06 C0 476 SW16RTN      sta CXROMOFF      ; enable slots
FA7B      477 ;
FA7B 6C 1E 00 478      jmp (R15L)
FA7E      479 ;
FA7E      480 ;
FA7E 20 B4 FB 481 RESET2      jsr GOTOROM
FA81      482 ;
FA81      483 ;
FA81 D8      484 NEWMON      cld
FA82      485 ;
FA82 20 3A FF 486      jsr BELL
FA85      487 ;
FA85 AD F3 03 488      lda AUTORSET+1      ; check for power up
FA88 49 A5      489      eor #PWRUPBYT
FA8A      490 ;
FA8A CD F4 03 491      cmp PWRSTATE
FA8D D0 17      492      bne PWRUP
FA8F      493 ;
FA8F AD F2 03 494      lda AUTORSET      ; BASIC coldstart?
FA92 D0 0F      495      bne >1

```

```

FA94          496 ;
FA94 A9 E0    497      lda /BASIC
FA96 CD F3 03 498      cmp AUTORSET+1
FA99 D0 08    499      bne >1
FA9B          500 ;
FA9B A0 03    501 NEWMON2 ldy #BASIC2
FA9D 8C F2 03 502      sty AUTORSET
FAA0          503 ;
FAA0 4C 00 E0 504      jmp BASIC
FAA3          505 ;
FAA3 6C F2 03 506 ^1      jmp (AUTORSET)
FAA6          507 ;
FAA6          508 ;
FAA6          509 ; Setup PAGE03 vectors.
FAA6          510 ;
FAA6 20 60 FB 511 PWRUP    jsr APPLE2
FAA9          512 ;
FAA9 A2 05    513      ldx #PWRCNLEN
FAAB          514 ;
FAAB BD FC FA 515 ^1      lda PWRCON-1,X
FAAE 9D EF 03 516      sta AUTOBRK,X
FAB1          517 ;
FAB1 CA       518      dex
FAB2 D0 F7    519      bne <1
FAB4          520 ;
FAB4 A9 C8    521      lda /PAGEC8          ; high slot + 1
FAB6          522 ;
FAB6 86 00    523      stx LOC0          ; X-reg = 0
FAB8 85 01    524      sta LOC1
FABA          525 ;
FABA          526 ;
FABA          527 ; Check 3 ID bytes, not 4, to find all bootable devices.
FABA          528 ;
FABA A0 03    529 ^2      ldy #DISKIDLN-3
FABC          530 ;
FABC C6 01    531      dec LOC1
FABE          532 ;
FABE A5 01    533      lda LOC1
FAC0 8D F8 07 534      sta MSLOT
FAC3          535 ;
FAC3 C9 C0    536      cmp /PAGEC0          ; last slot?
FAC5 F0 D4    537      beq NEWMON2
FAC7          538 ;
FAC7 B1 00    539 ^3      lda (LOC0),Y
FAC9 D9 02 FB 540      cmp DISKID,Y
FACC D0 EC    541      bne <2
FACE          542 ;
FACE 88       543      dey
FACF 88       544      dey
FAD0          545 ;
FAD0 10 F5    546      bpl <3
FAD2          547 ;
FAD2 6C 00 00 548      jmp (LOC0)
FAD5          549 ;
FAD5          550 ;
FAD5          551      dfs 2,ZERO          ; 2 bytes
FAD7          552 ;
FAD7          553 ;
FAD7          554 ; Display register contents with labels.
FAD7          555 ;
FAD7 20 8E FD 556 REGDSP    jsr CROUT

```



```

FADA          557 ;
FADA A9 45     558 REGDSP2  lda #AREG
FADC 85 40     559          sta A3L
FADE          560 ;
FADE A9 00     561          lda /AREG
FAE0 85 41     562          sta A3H
FAE2          563 ;
FAE2 A2 FB     564          ldx #$100-REGTBLEN
FAE4          565 ;
FAE4 A9 A0     566 ^1      lda #SPACE
FAE6 20 ED FD  567          jsr COUT
FAE9          568 ;
FAE9 BD 1E FA  569          lda REGTBL-$100-REGTBLEN,X
FAEC 20 ED FD  570          jsr COUT
FAEF          571 ;
FAEF A9 BD     572          lda #""
FAF1 20 ED FD  573          jsr COUT
FAF4          574 ;
FAF4 B5 4A     575          lda AREG+5,X          ; REGTBLEN
FAF6 20 DA FD  576          jsr PRBYTE
FAF9          577 ;
FAF9 E8        578          inx
FAFA 30 E8     579          bmi <1
FAFC          580 ;
FAFC 60        581          rts
FAFD          582 ;
FAFD          583 ;
FAFD 59 FA     584 PWRCON   adr OLDBRK
FAFF 00 E0     585          adr BASIC
FB01 45        586          byt PWRUPBYT^BASIC/PAGESIZE
FB02          587 ;
0005          588 PWRCNLEN equ *-PWRCON
FB02          589 ;
FB02          590 ;
FB02          591 ; Only 3 of 4 ID bytes are now verified.
FB02          592 ;
FB02 FF 20     593 DISKID   hex FF20
FB04 FF 00     594          hex FF00
FB06 FF 03     595          hex FF03
FB08          596          hex FF3C
FB08          597 ;
0006          598 DISKIDLN equ *-DISKID
FB08          599 ;
FB08          600 ;
FB08 20 84 FE  601 RSETINIT jsr SETNORM
FB0B 20 2F FB  602          jsr INIT
FB0E 20 93 FE  603          jsr SETVID
FB11          604 ;
FB11 4C 89 FE  605          jmp SETKBD
FB14          606 ;
FB14          607 ;
FB14          608 ; Translate table for IJKLM to DBALC.
FB14          609 ;
FB14 C4 C2 C1  610 XLATBL   asc "DBALC"
FB17 CC C3
FB19          611 ;
FB19          612 ;
FB19          613 ; Display register table.
FB19          614 ;
FB19 C1 D8 D9  615 REGTBL   asc "AXYPS"
FB1C D0 D3

```

```

FB1E      616 ;
0005      617 REGTBLEN equ *-REGTBL
FB1E      618 ;
FB1E      619 ;
FB1E      620 ; Paddle read routine.  Count Y-reg every 11 usec.
FB1E      621 ;
FB1E AD 70 C0 622 PREAD      lda GCTOGL
FB21      623 ;
FB21 A0 00    624          ldy #ZERO
FB23      625 ;
FB23 EA      626          nop                ; compensate for 1st count
FB24 EA      627          nop
FB25      628 ;
FB25 BD 64 C0 629 ^1      lda GC1IN,X
FB28 10 04    630          bpl >2
FB2A      631 ;
FB2A C8      632          iny
FB2B D0 F8    633          bne <1
FB2D      634 ;
FB2D 88      635          dey
FB2E      636 ;
FB2E 60      637 ^2      rts
FB2F      638 ;
FB2F      639 ;
FB2F      640 ; Initialize status, text mode, graphics mode, and window.
FB2F      641 ;
FB2F A9 00    642 INIT      lda #ZERO
FB31 85 48    643          sta PREG
FB33      644 ;
FB33 AD 56 C0 645 INIT2     lda HIRESOFF
FB36 AD 54 C0 646          lda PAGE1ON
FB39      647 ;
FB39 AD 51 C0 648 SETEXT    lda TEXTON
FB3C      649 ;
FB3C A9 00    650          lda #ZERO
FB3E F0 0B    651          beq SETWND                ; always taken
FB40      652 ;
FB40 AD 50 C0 653 SETGR     lda TEXTOFF
FB43 AD 53 C0 654          lda MIXEDON
FB46      655 ;
FB46 20 36 F8 656          jsr CLRTOP
FB49      657 ;
FB49 A9 14    658          lda #20
FB4B      659 ;
FB4B 85 22    660 SETWND    sta WNDTOP
FB4D      661 ;
FB4D A9 00    662          lda #ZERO
FB4F 85 20    663          sta WNDLFT
FB51      664 ;
FB51 A0 0C    665          ldy #12
FB53 D0 5F    666          bne GOTOROM                ; always taken
FB55      667 ;
FB55      668 ;
FB55      669          dfs 6,ZERO                ; 6 bytes
FB5B      670 ;
FB5B      671 ;
FB5B 85 25    672 TABV      sta CV
FB5D      673 ;
FB5D 4C 22 FC 674          jmp VTAB
FB60      675 ;
FB60      676 ;

```

```

FB60          677 ; TITLE was moved and is 1 byte longer.
FB60          678 ;
FB60 20 58 FC  679 APPLE2    jsr HOME
FB63          680 ;
FB63 A0 0A     681          ldy #TITLEN
FB65          682 ;
FB65 B9 8E F7  683 STITLE    lda TITLE-1,Y
FB68 99 0E 04  684          sta TEXTPG1+OFFTITL,Y
FB6B          685 ;
FB6B 88        686          dey
FB6C D0 F7     687          bne STITLE
FB6E          688 ;
FB6E 60        689          rts
FB6F          690 ;
FB6F          691 ;
FB6F          692 ; Routine to calculate power-up byte from RESET vector.
FB6F          693 ; Not referenced.
FB6F          694 ;
FB6F AD F3 03  695 SETPWRUP  lda AUTORSET+1
FB72 49 A5     696          eor #PWRUPBYT
FB74 8D F4 03  697          sta PWRSTATE
FB77          698 ;
FB77 60        699          rts
FB78          700 ;
FB78          701 ;
FB78 C9 8D     702 VIDWAIT   cmp #RETURN
FB7A D0 18     703          bne >1
FB7C          704 ;
FB7C AC 00 C0  705          ldy KEY
FB7F 10 13     706          bpl >1
FB81          707 ;
FB81 C0 93     708          cpy #CTRLS          ; ctrl-S pressed?
FB83 D0 0F     709          bne >1
FB85          710 ;
FB85 2C 10 C0  711          bit CLRKEY
FB88          712 ;
FB88          713 ;
FB88          714 ; Nothing is done with CTRLC.
FB88          715 ;
FB88 AC 00 C0  716 KBDWAIT   ldy KEY
FB8B 10 FB     717          bpl KBDWAIT
FB8D          718 ;
FB8D C0 83     719          cpy #CTRLC          ; ctrl-C pressed?
FB8F F0 03     720          beq >1
FB91          721 ;
FB91 2C 10 C0  722          bit CLRKEY
FB94          723 ;
FB94 4C FD FB  724 ^1        jmp VIDOUT
FB97          725 ;
FB97          726 ;
FB97 38        727 ESCOLD    sec          ; insure carry set
FB98          728 ;
FB98 4C 2C FC  729          jmp ESCOLD2
FB9B          730 ;
FB9B          731 ;
FB9B A8        732 ^1        tay          ; character is index
FB9C          733 ;
FB9C B9 4B FA  734          lda XLATBL-"I",Y    ; IJKLM -> CBALD
FB9F          735 ;
FB9F 20 97 FB  736          jsr ESCOLD
FBA2 20 21 FD  737          jsr RDESC

```

```

FBA5          738 ;
FBA5          739 ;
FBA5 C9 CE    740 ESCNEW    cmp #"N"
FBA7 B0 EE    741          bcs ESCOLD          ; exclude >M
FBA9          742 ;
FBA9 C9 C9    743          cmp #"I"
FBAB 90 EA    744          bcc ESCOLD          ; exclude <I
FBAD          745 ;
FBAD C9 CC    746          cmp #"L"
FBAF F0 E6    747          beq ESCOLD          ; exclude L
FBB1          748 ;
FBB1 D0 E8    749          bne <1              ; always taken
FBB3          750 ;
FBB3          751 ;
FBB3 06       752 SIGROM    hex 06              ; ROM ID byte
FBB4          753 ;
FBB4          754 ;
FBB4          755 ; Save state of CX ROM.  If +, return via CXOFF and turn CX
FBB4          756 ; ROM off.  If -, return via CXRTN and leave CX ROM on.
FBB4          757 ;
FBB4 2C 15 C0 758 GOTOROM  bit RDCXROM          ; get CX ROM state
FBB7          759 ;
FBB7 08       760          php                  ; save state
FBB8          761 ;
FBB8 8D 07 C0 762          sta CXROMON          ; enable CX ROM
FBBB          763 ;
FBBB 4C 00 C1 764          jmp DOCXCMD
FBBE          765 ;
FBBE          766 ;
FBBE          767          dfs 2,ZERO          ; 2 bytes
FBC0          768 ;
FBC0          769 ;
FBC0          770 ; Signature byte.  Original Apple //e uses 0xEA.
FBC0          771 ; Enhanced Apple //e uses 0xE0.
FBC0          772 ;
FBC0 E0       773 SIGBYTE  hex E0              ; //e ROM ID byte
FBC1          774 ;
FBC1          775 ;
FBC1          776 ; The same routine is at 0xCABA, so use that routine.
FBC1          777 ;
FBC1 84 28    778 BASCALC  sty BASL
FBC3          779 ;
FBC3 A0 02    780          ldy #2
FBC5 D0 ED    781          bne GOTOROM          ; always taken
FBC7          782 ;
FBC7          783 ;
FBC7          784 ; FMT2 bits: 6543 21LL
FBC7          785 ;
FBC7          786 ; LL sets value/address length:
FBC7          787 ;
FBC7          788 ; 0 - no extra bytes (1 byte)
FBC7          789 ; 1 - value or zpage (2 bytes)
FBC7          790 ; 2 - absolute address (3 bytes)
FBC7          791 ; 3 - 1 byte relative branch for abs. addr. (2 bytes)
FBC7          792 ;
FBC7          793 ; Xreg index into CHAR1/CHAR2 tables for bit set:
FBC7          794 ;
FBC7          795 ; 6 - $ |
FBC7          796 ; 5 - ($ } left of value/address
FBC7          797 ; 4 - #$ |
FBC7          798 ;

```

```

FBC7          799 ;      3 - ,X |
FBC7          800 ;      2 - )   } right of value/address
FBC7          801 ;      1 - ,Y |
FBC7          802 ;
FBC7          803 ;
FBC7          804 FMT2:
FBC7 00        805      byt %00000000      ; error
FBC8 21        806      byt %00100001      ; immediate; 1 byte
FBC9 81        807      byt %10000001      ; zpage; 1 byte
FBCA 82        808      byt %10000010      ; absolute; 2 bytes
FBCB 00        809      byt %00000000      ; implied; 0 bytes
FBCC 00        810      byt %00000000      ; accumulator; 0 bytes
FBCD 59        811      byt %01011001      ; (zpage,X); 1 byte
FBCE 4D        812      byt %01001101      ; (zpage),Y; 1 byte
FBCF 91        813      byt %10010001      ; zpage,X; 1 byte
FBD0 92        814      byt %10010010      ; absolute,X; 2 bytes
FBD1 86        815      byt %10000110      ; absolute,Y; 2 bytes
FBD2 4A        816      byt %01001010      ; (absolute); 2 bytes
FBD3 85        817      byt %10000101      ; zpage,Y; 1 byte
FBD4 9D        818      byt %10011101      ; relative; 1 byte
FBD5 49        819      byt %01001001      ; (zpage); 1 byte (was 0x4B)
FBD6 5A        820      byt %01011010      ; (absolute,X); 2 bytes
FBD7          821 ;
FBD7          822 ;
FBD7          823      dfs 2,ZERO          ; 2 bytes
FBD9          824 ;
FBD9          825 ;
FBD9 C9 87     826 BELL2      cmp #ASCIBELL      ; ^G
FBDB D0 12     827          bne >2
FBDD          828 ;
FBDD A9 40     829 RINGBELL   lda #$40          ; delay 0.01 seconds
FBDF 20 A8 FC  830          jsr WAIT
FBE2          831 ;
FBE2 A0 C0     832          ldy #$C0
FBE4          833 ;
FBE4 A9 0C     834 ^1      lda #12          ; 1 KHz for 1 second
FBE6 20 A8 FC  835          jsr WAIT
FBE9          836 ;
FBE9 AD 30 C0  837          lda SPKRTOGL
FBEC          838 ;
FBEC 88        839          dey
FBED D0 F5     840          bne <1
FBEF          841 ;
FBEF 60        842 ^2      rts
FBF0          843 ;
FBF0          844 ;
FBF0          845 ; Store character on screen and advance cursor.
FBF0          846 ;
FBF0 A4 24     847 STORADV   ldy CH
FBF2          848 ;
FBF2 91 28     849          sta (BASL),Y
FBF4          850 ;
FBF4 E6 24     851 ADVANCE   inc CH
FBF6          852 ;
FBF6 A5 24     853          lda CH
FBF8 C5 21     854          cmp WNDWDTH
FBFA B0 66     855          bcs CR
FBFC          856 ;
FBFC 60        857 RTN.FB.F rts
FBFD          858 ;
FBFD          859 ;

```

FBFD 860 icl "FC.L"

LLOAD FC.L,A\$4000

```

FBFD      1          ttl "ROM Source Code, FC.L"
FBFD      2      ;
FBFD      3      ;
FBFD      4      ; FC.L
FBFD      5      ;
FBFD      6      ;
FBFD      7      ; Check for NORMAL, INVERSE, and control characters.
FBFD      8      ;
FBFD C9 A0      9  VIDOUT    cmp #SPACE
FBFF B0 EF     10          bcs STORADV
FC01      11      ;
FC01 A8       12          tay
FC02 10 EC     13          bpl STORADV
FC04      14      ;
FC04 C9 8D     15          cmp #RETURN
FC06 F0 5A     16          beq CR
FC08      17      ;
FC08 C9 8A     18          cmp #DARROW
FC0A F0 5A     19          beq LF
FC0C      20      ;
FC0C C9 88     21          cmp #LARROW
FC0E D0 C9     22          bne BELL2
FC10      23      ;
FC10 C6 24     24  BS        dec CH
FC12 10 E8     25          bpl RTN.FB.F
FC14      26      ;
FC14 A5 21     27          lda WNDWDTH
FC16 85 24     28          sta CH
FC18      29      ;
FC18 C6 24     30          dec CH
FC1A      31      ;
FC1A A5 22     32  UP        lda WNDTOP
FC1C C5 25     33          cmp CV
FC1E B0 DC     34          bcs RTN.FB.F
FC20      35      ;
FC20 C6 25     36          dec CV
FC22      37      ;
FC22      38      ;
FC22 A5 25     39  VTAB      lda CV
FC24      40      ;
FC24 84 28     41  VTAB2     sty BASL
FC26      42      ;
FC26 A0 04     43          ldy #4
FC28 D0 8A     44          bne GOTOROM          ; always taken
FC2A      45      ;
FC2A      46      ;
FC2A      47          dfs 2,ZERO          ; 2 bytes
FC2C      48      ;
FC2C      49      ;
FC2C 49 C0     50  ESCOLD2   eor #"@"          ; check esc-@
FC2E F0 28     51          beq HOME
FC30      52      ;
FC30 69 FD     53          adc #!-3
FC32 90 C0     54          bcc ADVANCE          ; check esc-A
FC34      55      ;
FC34 F0 DA     56          beq BS          ; check esc-B
FC36      57      ;
FC36 69 FD     58          adc #!-3
FC38 90 2C     59          bcc LF          ; check esc-C
FC3A      60      ;

```

```

FC3A F0 DE      61          beq UP          ; check esc-D
FC3C            62      ;
FC3C 69 FD      63          adc #!-3
FC3E 90 5C      64          bcc CLREOL      ; check esc-E
FC40            65      ;
FC40 D0 BA      66          bne RTN.FB.F    ; not esc-F
FC42            67      ;
FC42 A0 0A      68  CLREOP    ldy #10        ; handle esc-F
FC44 D0 14      69          bne GOTOROM2    ; always taken
FC46            70      ;
FC46            71      ;
FC46            72      ; New VIDWAIT to handle 40 and 80 columns.
FC46            73      ;
FC46 2C 1F C0   74  NEWVW     bit RDVID80      ; 80 column enabled?
FC49 10 04      75          bpl >1
FC4B            76      ;
FC4B A0 00      77          ldy #ZERO        ; print 80 column
FC4D F0 0B      78          beq GOTOROM2    ; always taken
FC4F            79      ;
FC4F            80      ;
FC4F            81      ; Print 40 column.
FC4F            82      ;
FC4F 98          83      ^1      tya
FC50 48          84          pha
FC51            85      ;
FC51 20 78 FB   86          jsr VIDWAIT
FC54            87      ;
FC54 68          88          pla
FC55            89      ;
FC55 A4 35      90          ldy YSAV1
FC57            91      ;
FC57 60          92          rts
FC58            93      ;
FC58            94      ;
FC58 A0 05      95  HOME     ldy #5
FC5A            96      ;
FC5A 4C B4 FB   97  GOTOROM2 jmp GOTOROM
FC5D            98      ;
FC5D            99      ;
FC5D            100         dfs 2,ZERO        ; 2 bytes
FC5F            101      ;
FC5F            102      ;
FC5F            103      ; Return from BRK instruction in STEP/TRACE.
FC5F            104      ;
FC5F 4C 59 FA   105  STEPRTN2 jmp OLDBRK
FC62            106      ;
FC62            107      ;
FC62 A9 00      108  CR       lda #ZERO
FC64 85 24      109          sta CH
FC66            110      ;
FC66 E6 25      111  LF       inc CV
FC68            112      ;
FC68 A5 25      113          lda CV
FC6A C5 23      114          cmp WNDBTM
FC6C 90 B6      115          bcc VTAB2
FC6E            116      ;
FC6E C6 25      117          dec CV
FC70            118      ;
FC70 A0 06      119  SCROLL   ldy #6
FC72 D0 E6      120          bne GOTOROM2    ; always taken
FC74            121      ;

```



```

FC74      122 ;
FC74 8D 06 C0 123 GOTOIRQ sta CXROMOFF ; enable slots
FC77      124 ;
FC77 6C FE 03 125 jmp (MASKIRQ)
FC7A      126 ;
FC7A      127 ;
FC7A      128 ; After an interrupt, IRQDONE (0xC3F4) jumps here because
FC7A      129 ; this code cannot execute in CX ROM space.
FC7A      130 ;
FC7A 68      131 IRQDONE2 pla
FC7B 8D F8 07 132 sta MSLOT
FC7E      133 ;
FC7E C9 C1    134 cmp /PAGEC1
FC80 90 0D    135 bcc >1
FC82      136 ;
FC82 8D FF CF 137 sta CLRROM
FC85      138 ;
FC85 A0 00    139 ldy #ZERO
FC87 A6 01    140 ldx LOC1
FC89      141 ;
FC89 85 01    142 sta LOC1
FC8B      143 ;
FC8B B1 00    144 lda (LOC0),Y
FC8D      145 ;
FC8D 86 01    146 stx LOC1
FC8F      147 ;
FC8F 8D 07 C0 148 ^1 sta CXROMON ; enable CX ROM
FC92      149 ;
FC92 4C 7C C4 150 jmp NEWIRQ ; restore the machine
FC95      151 ;
FC95      152 ;
FC95 90 02    153 CHKINV bcc >1
FC97      154 ;
FC97 25 32    155 and INVFLG
FC99      156 ;
FC99 4C F7 FD 157 ^1 jmp COUTZ2
FC9C      158 ;
FC9C      159 ;
FC9C 38      160 CLREOL sec
FC9D      161 ;
FC9D 90 00    162 bcc *+2
FC9F      163 dfs !-1
FC9E      164 ;
FC9E 18      165 CLREOL2 clc
FC9F      166 ;
FC9F 84 2A    167 sty BAS2L
FCA1      168 ;
FCA1 A0 07    169 ldy #7
FCA3      170 ;
FCA3 B0 B5    171 bcs GOTOROM2
FCA5      172 ;
FCA5 C8      173 iny
FCA6 D0 B2    174 bne GOTOROM2 ; always taken
FCA8      175 ;
FCA8      176 ;
FCA8      177 ; A-reg = delay value. Clock is 1,020,484 cycles/second.
FCA8      178 ;
FCA8      179 ; Delay = 2.5*A*A + 13.5*A + 13 cycles.
FCA8      180 ;
FCA8      181 ; A = ( Delay / 2.5 + 2.09 ) ^ 0.5 - 2.7.
FCA8      182 ;

```

```

FCA8 38          183 WAIT      sec
FCA9          184 ;
FCA9 48          185 ^1      pha
FCAA          186 ;
FCAA E9 01       187 ^2      sbc #1
FCAC D0 FC       188          bne <2
FCAE          189 ;
FCAE 68          190          pla
FCAF          191 ;
FCAF E9 01       192          sbc #1
FCB1 D0 F6       193          bne <1
FCB3          194 ;
FCB3 60          195          rts
FCB4          196 ;
FCB4          197 ;
FCB4          198 ; Increment A4, compare A1 to A2, increment A1.
FCB4          199 ;
FCB4 E6 42       200 NEXTA4    inc A4L
FCB6 D0 02       201          bne NEXTA1
FCB8          202 ;
FCB8 E6 43       203          inc A4H
FCBA          204 ;
FCBA A5 3C       205 NEXTA1    lda A1L
FCBC C5 3E       206          cmp A2L
FCBE          207 ;
FCBE A5 3D       208          lda A1H
FCC0 E5 3F       209          sbc A2H
FCC2          210 ;
FCC2 E6 3C       211          inc A1L
FCC4 D0 02       212          bne >1
FCC6          213 ;
FCC6 E6 3D       214          inc A1H
FCC8          215 ;
FCC8 60          216 ^1      rts
FCC9          217 ;
FCC9          218 ;
FCC9 60          219 HEADR    rts                ; cassette HEADR removed
FCCA          220 ;
FCCA          221 ;
FCCA          222 ; Return from STEP/TRACE. If carry set, then BRK.
FCCA          223 ;
FCCA 8D 06 C0    224 STEPRTN  sta CXROMOFF
FCCD          225 ;
FCCD B0 90       226          bcs STEPRTN2
FCCF          227 ;
FCCF 4C 73 FF    228          jmp NEXTITM
FCD2          229 ;
FCD2          230 ;
FCD2 8D 06 C0    231 XERR    sta CXROMOFF                ; enable slots
FCD5          232 ;
FCD5 20 4A F9    233          jsr PRBL2                ; tab to error
FCD8          234 ;
FCD8 A9 DE       235          lda #"^"
FCDA 20 ED FD    236          jsr COUT
FCDD          237 ;
FCDD 20 3A FF    238          jsr BELL
FCE0          239 ;
FCE0 4C F0 FC    240          jmp NXTLINE
FCE3          241 ;
FCE3          242 ;
FCE3          243 ; End of Mini-Assembler instruction processing.

```

```

FCE3      244 ;
FCE3 8D 06 C0 245 FINDOP      sta CXROMOFF      ; enable slots
FCE6      246 ;
FCE6 20 D0 F8 247             jsr INSTDSP
FCE9 20 53 F9 248             jsr PCADJ
FCEC      249 ;
FCEC 84 3B 250             sty PCH
FCEE 85 3A 251             sta PCL
FCF0      252 ;
FCF0      253 ;
FCF0      254 ; Start of Mini-Assembler instruction processing.
FCF0      255 ;
FCF0 A9 A1 256 NXTLINE      lda #"!"
FCF2 85 33 257             sta PROMPT
FCF4      258 ;
FCF4 20 67 FD 259             jsr GETLINE2
FCF7      260 ;
FCF7 8D 07 C0 261             sta CXROMON      ; enable CX ROM
FCFA      262 ;
FCFA 4C 9C CF 263             jmp NXTLINE2
FCFD      264 ;
FCFD      265 ;
FCFD      266 ; Change lowercase to uppercase.
FCFD      267 ;
FCFD B9 00 02 268 UPRMON      lda INPUT,Y
FD00      269 ;
FD00 C8 270             iny
FD01      271 ;
FD01 C9 E1 272 UPRCASE      cmp #"a"
FD03 90 06 273             bcc >1
FD05      274 ;
FD05 C9 FB 275             cmp #"z"+1
FD07 B0 02 276             bcs >1
FD09      277 ;
FD09 29 DF 278             and #LWRMASK
FD0B      279 ;
FD0B 60 280             ^1      rts
FD0C      281 ;
FD0C      282 ;
FD0C 4C 13 FD 283 RDKEY      jmp RDKEY2
FD0F      284 ;
FD0F      285 ;
FD0F      286             dfs 1,ZERO      ; 1 byte
FD10      287 ;
FD10      288 ;
FD10      289 ; Preserve this entry point, but do nothing.
FD10      290 ;
FD10 6C 38 00 291             jmp (KSWL)
FD13      292 ;
FD13      293 ;
FD13 A0 0B 294 RDKEY2      ldy #11
FD15 20 B4 FB 295             jsr GOTOROM
FD18      296 ;
FD18 6C 38 00 297 RDKEY3      jmp (KSWL)
FD1B      298 ;
FD1B      299 ;
FD1B A0 03 300 KEYIN      ldy #3
FD1D      301 ;
FD1D 4C B4 FB 302 GOTOROM4  jmp GOTOROM
FD20      303 ;
FD20      304 ;

```

```

FD20          305          dfs 1,ZERO          ; 1 byte
FD21          306          ;
FD21          307          ;
FD21 20 13 FD 308 RDESC      jsr RDKEY2          ; get a key
FD24          309          ;
FD24 A0 01     310          ldy #1
FD26 D0 F5     311          bne GOTOROM4        ; always taken
FD28          312          ;
FD28          313          ;
FD28          314          ; All slot cards must save their slot number in MSLOT and
FD28          315          ; set its MSB. If the MSB of MSLOT is clear, that is a
FD28          316          ; flag to the video firmware that escapes are allowed.
FD28          317          ;
FD28 4E F8 07 318 NEWRDKEY lsr MSLOT            ; allow escapes
FD2B          319          ;
FD2B 4C 13 FD 320          jmp RDKEY2          ; now read the key
FD2E          321          ;
FD2E          322          ;
FD2E          323          dfs 1,ZERO          ; 1 byte
FD2F          324          ;
FD2F          325          ;
FD2F 20 21 FD 326 ESC        jsr RDESC          ; remap keys
FD32 20 A5 FB 327          jsr ESCNEW          ; do ESC function
FD35          328          ;
FD35          329          ;
FD35 20 28 FD 330 RDCHAR     jsr NEWRDKEY
FD38          331          ;
FD38 C9 9B     332          cmp #ESCAPE
FD3A F0 F3     333          beq ESC
FD3C          334          ;
FD3C 60        335          rts
FD3D          336          ;
FD3D          337          ;
FD3D          338          ; Do 80 column pick and fix.
FD3D          339          ;
FD3D A0 0F     340 PICKFIX   ldy #15
FD3F 20 B4 FB 341          jsr GOTOROM
FD42          342          ;
FD42 A4 24     343          ldy CH
FD44          344          ;
FD44 9D 00 02 345          sta INPUT,X
FD47          346          ;
FD47 20 ED FD 347 NOTCR     jsr COUT
FD4A          348          ;
FD4A BD 00 02 349          lda INPUT,X
FD4D C9 88     350          cmp #LARROW
FD4F F0 20     351          beq BCKSPC
FD51          352          ;
FD51 C9 98     353          cmp #CTRLX          ; ctrl-X pressed?
FD53 F0 0D     354          beq CANCEL
FD55          355          ;
FD55 E0 F8     356          cpx #$F8
FD57 90 06     357          bcc >1            ; margin?
FD59          358          ;
FD59 20 3A FF 359          jsr BELL
FD5C          360          ;
FD5C EA       361          nop
FD5D EA       362          nop
FD5E EA       363          nop
FD5F          364          ;
FD5F E8       365          ^1      inx

```

```

FD60 D0 13      366          bne NXTCHAR          ; always taken
FD62            367          ;
FD62            368          ;
FD62            369          ; Print a backslash after a cancelled line.
FD62            370          ;
FD62 A9 DC      371 CANCEL    lda #"\ "
FD64 20 ED FD    372          jsr COUT
FD67            373          ;
FD67 20 8E FD    374 GETLINE2 jsr CROUT
FD6A            375          ;
FD6A A5 33      376 GETLINE    lda PROMPT
FD6C 20 ED FD    377          jsr COUT
FD6F            378          ;
FD6F A2 01      379          ldx #1
FD71            380          ;
FD71 8A          381 BCKSPC    txa
FD72 F0 F3      382          beq GETLINE2
FD74            383          ;
FD74 CA          384          dex
FD75            385          ;
FD75 20 35 FD    386 NXTCHAR    jsr RDCHAR
FD78            387          ;
FD78            388          ;
FD78            389          ; For ^U processing use the screen character, either 40
FD78            390          ; column PICK or 80 column PICKFIX.
FD78            391          ;
FD78 C9 95      392          cmp #CTRLU
FD7A D0 08      393          bne ADDINP
FD7C            394          ;
FD7C B1 28      395          lda (BASL),Y          ; 40 column pick
FD7E            396          ;
FD7E            397          ;
FD7E            398          ; CAPTST entry gone.
FD7E            399          ;
FD7E            400          ;
FD7E 2C 1F C0    401          bit RDVID80
FD81 30 BA      402          bmi PICKFIX          ; 80 column pick
FD83            403          ;
FD83 EA          404          nop
FD84            405          ;
FD84            406          ;
FD84            407          ; Add to input buffer.
FD84            408          ;
FD84 9D 00 02    409 ADDINP    sta INPUT,X
FD87 C9 8D      410          cmp #RETURN
FD89 D0 BC      411          bne NOTCR
FD8B            412          ;
FD8B 20 9C FC    413          jsr CLREOL          ; clear to EOL after CR
FD8E            414          ;
FD8E            415          ;
FD8E A9 8D      416 CROUT     lda #RETURN
FD90 D0 5B      417          bne COUT          ; always taken
FD92            418          ;
FD92            419          ;
FD92            420          ; Print CR, then A1 in HEX.
FD92            421          ;
FD92 A6 3C      422 PRA1      ldx A1L
FD94 A4 3D      423          ldy A1H
FD96            424          ;
FD96 20 8E FD    425 PRXY      jsr CROUT
FD99 20 40 F9    426          jsr PRNTYX

```

```

FD9C          427 ;
FD9C A0 00    428      ldy #ZERO
FD9E          429 ;
FD9E A9 AD    430      lda #"-"
FDA0          431 ;
FDA0 4C ED FD 432      jmp COUT
FDA3          433 ;
FDA3          434 ;
FDA3 A5 3C    435 XAM8   lda A1L
FDA5 09 07    436      ora #7           ; set to finish at MOD 8=7
FDA7 85 3E    437      sta A2L
FDA9          438 ;
FDA9 A5 3D    439      lda A1H
FDAB 85 3F    440      sta A2H
FDAD          441 ;
FDAD A5 3C    442 MOD8CHK lda A1L
FDAF 29 07    443      and #7
FDB1 D0 03    444      bne DATAOUT
FDB3          445 ;
FDB3 20 92 FD 446 XAM     jsr PRA1
FDB6          447 ;
FDB6 A9 A0    448 DATAOUT lda #SPACE
FDB8 20 ED FD 449      jsr COUT
FDBB          450 ;
FDBB B1 3C    451      lda (A1L),Y
FDBD 20 DA FD 452      jsr PRBYTE
FDC0          453 ;
FDC0 20 BA FC 454      jsr NEXTA1
FDC3 90 E8    455      bcc MOD8CHK
FDC5          456 ;
FDC5 60       457      rts
FDC6          458 ;
FDC6          459 ;
FDC6          460 ; Determine if monitor mode is examine, add, or subtract.
FDC6          461 ;
FDC6 4A       462 XAMPM   lsr
FDC7 90 EA    463      bcc XAM
FDC9          464 ;
FDC9 4A       465      lsr
FDCA 4A       466      lsr
FDCB          467 ;
FDCB A5 3E    468      lda A2L
FDCD          469 ;
FDCD 90 02    470      bcc ADD
FDCF          471 ;
FDCF 49 FF    472      eor #NEGONE           ; 2's complement for subtract
FDD1          473 ;
FDD1 65 3C    474 ADD     adc A1L
FDD3 48       475      pha
FDD4          476 ;
FDD4 A9 BD    477      lda #"="
FDD6 20 ED FD 478      jsr COUT
FDD9          479 ;
FDD9 68       480      pla
FDDA          481 ;
FDDA          482 ;
FDDA          483 ; Print byte as 2 HEX digits.
FDDA          484 ;
FDDA 48       485 PRBYTE  pha
Fddb          486 ;
Fddb 4A       487      lsr

```

```

FDDC 4A          488          lsr
FDDD 4A          489          lsr
FDDE 4A          490          lsr
FDDF           491          ;
FDDF 20 E5 FD    492          jsr PRHEX2
FDE2           493          ;
FDE2 68          494          pla
FDE3           495          ;
FDE3           496          ;
FDE3           497          ; Print HEX digit in A-reg.
FDE3           498          ;
FDE3 29 0F       499 PRHEX      and #$0F
FDE5           500          ;
FDE5 09 B0       501 PRHEX2    ora #"0"
FDE7           502          ;
FDE7 C9 BA       503          cmp #"9"+1
FDE9 90 02       504          bcc COUT
FDEB           505          ;
FDEB 69 06       506          adc #6                ; for HEX letters
FDED           507          ;
FDED           508          ;
FDED           509          ; Vector to user output routing.
FDED           510          ;
FDED 6C 36 00    511 COUT      jmp (CSWL)
FDF0           512          ;
FDF0           513          ;
FDF0 48          514 COUT2     pha
FDF1           515          ;
FDF1 C9 A0       516          cmp #SPACE
FDF3           517          ;
FDF3 4C 95 FC    518          jmp CHKINV
FDF6           519          ;
FDF6           520          ;
FDF6 48          521 COUTZ     pha
FDF7           522          ;
FDF7 84 35       523 COUTZ2    sty YSAV1
FDF9           524          ;
FDF9 A8          525          tay                ; masked character
FDFA           526          ;
FDFA 68          527          pla                ; original character
FDFB           528          ;
FDFB 4C 46 FC    529          jmp NEWVW
FD FE           530          ;
FD FE           531          ;
FD FE           532          dfs 2,ZERO          ; 2 bytes
FE00           533          ;
FE00           534          ;
FE00 C6 34       535 BL1      dec YSAV
FE02 F0 9F       536          beq XAM8
FE04           537          ;
FE04 CA          538 BLANK     dex
FE05 D0 16       539          bne SETMDZ
FE07           540          ;
FE07 C9 BA       541          cmp #":"
FE09 D0 BB       542          bne XAMPM
FE0B           543          ;
FE0B 85 31       544 STOR      sta MODE
FE0D           545          ;
FE0D A5 3E       546          lda A2L
FE0F           547          ;
FE0F 91 40       548 STOR2     sta (A3L),Y

```

```

FE11          549 ;
FE11 E6 40    550 NEXTA3    inc A3L
FE13 D0 02    551          bne >1
FE15          552 ;
FE15 E6 41    553          inc A3H
FE17          554 ;
FE17 60       555 ^1      rts
FE18          556 ;
FE18          557 ;
FE18          558 ; Save converted `:`, `+`, `-`, or `.` as MODE.
FE18          559 ;
FE18 A4 34    560 SETMODE   ldy YSAV
FE1A          561 ;
FE1A B9 FF 01 562          lda INPUT-1,Y
FE1D          563 ;
FE1D 85 31    564 SETMDZ    sta MODE
FE1F          565 ;
FE1F 60       566          rts
FE20          567 ;
FE20          568 ;
FE20          569 ; Copy A2 to A4 and A5.
FE20          570 ;
FE20 A2 01    571 LT      ldx #1
FE22          572 ;
FE22 B5 3E    573 ^1      lda A2L,X
FE24 95 42    574          sta A4L,X
FE26 95 44    575          sta A5L,X
FE28          576 ;
FE28 CA       577          dex
FE29 10 F7    578          bpl <1
FE2B          579 ;
FE2B 60       580          rts
FE2C          581 ;
FE2C          582 ;
FE2C          583 ; Move A1 through A2 to A4.
FE2C          584 ;
FE2C B1 3C    585 MOVE     lda (A1L),Y
FE2E 91 42    586          sta (A4L),Y
FE30          587 ;
FE30 20 B4 FC 588          jsr NEXTA4
FE33 90 F7    589          bcc MOVE
FE35          590 ;
FE35 60       591          rts
FE36          592 ;
FE36          593 ;
FE36          594 ; Verify A1 through A2 with A4.
FE36          595 ;
FE36 B1 3C    596 VERIFY   lda (A1L),Y
FE38 D1 42    597          cmp (A4L),Y
FE3A F0 1C    598          beq >1
FE3C          599 ;
FE3C 20 92 FD 600          jsr PRA1
FE3F          601 ;
FE3F B1 3C    602          lda (A1L),Y
FE41 20 DA FD 603          jsr PRBYTE
FE44          604 ;
FE44 A9 A0    605          lda #SPACE
FE46 20 ED FD 606          jsr COUT
FE49          607 ;
FE49 A9 A8    608          lda #"("
FE4B 20 ED FD 609          jsr COUT

```



```

FE4E          610 ;
FE4E B1 42    611      lda (A4L),Y
FE50 20 DA FD 612      jsr PRBYTE
FE53          613 ;
FE53 A9 A9    614      lda #")"
FE55 20 ED FD 615      jsr COUT
FE58          616 ;
FE58 20 B4 FC 617 ^1    jsr NEXTA4
FE5B 90 D9    618      bcc VERIFY
FE5D          619 ;
FE5D 60       620      rts
FE5E          621 ;
FE5E          622 ;
FE5E          623 ; Move A1 to PC if specified and disassemble 20
FE5E          624 ; instructions.
FE5E          625 ;
FE5E 20 75 FE 626 LIST   jsr A1PC
FE61          627 ;
FE61 A9 14    628      lda #20
FE63          629 ;
FE63 48       630 LIST2   pha
FE64          631 ;
FE64 20 D0 F8 632      jsr INSTDSP
FE67 20 53 F9 633      jsr PCADJ
FE6A          634 ;
FE6A 85 3A    635      sta PCL
FE6C 84 3B    636      sty PCH
FE6E          637 ;
FE6E 68       638      pla
FE6F          639 ;
FE6F 38       640      sec
FE70          641 ;
FE70 E9 01    642      sbc #1
FE72 D0 EF    643      bne LIST2
FE74          644 ;
FE74 60       645      rts
FE75          646 ;
FE75          647 ;
FE75          648 ; If an address was specified copy it from A1 to PC.
FE75          649 ;
FE75 8A       650 A1PC     txa
FE76 F0 07    651      beq >1
FE78          652 ;
FE78 B5 3C    653 A1PCLP   lda A1L,X
FE7A 95 3A    654      sta PCL,X
FE7C          655 ;
FE7C CA       656      dex
FE7D 10 F9    657      bpl A1PCLP
FE7F          658 ;
FE7F 60       659 ^1     rts
FE80          660 ;
FE80          661 ;
FE80          662 ; Set INVERSE/NORMAL video.
FE80          663 ;
FE80 A0 3F    664 SETINV   ldy #INVERSE
FE82 D0 02    665      bne >1          ; always taken
FE84          666 ;
FE84 A0 FF    667 SETNORM   ldy #NEGONE
FE86          668 ;
FE86 84 32    669 ^1     sty INVFLG
FE88          670 ;

```

```

FE88 60          671          rts
FE89          672          ;
FE89          673          ;
FE89 A9 00      674 SETKBD    lda #ZERO          ; IN#0
FE8B          675          ;
FE8B 85 3E      676 INPORT   sta A2L          ; IN#n
FE8D          677          ;
FE8D A2 38      678 INPORT2  ldx #KSWL
FE8F A0 1B      679          ldy #KEYIN
FE91          680          ;
FE91 D0 08      681          bne IOPORT          ; always taken
FE93          682          ;
FE93          683          ;
FE93 A9 00      684 SETVID    lda #ZERO          ; PR#0
FE95          685          ;
FE95 85 3E      686 OUTPORT   sta A2L          ; PR#n
FE97          687          ;
FE97 A2 36      688 OUTPORT2  ldx #CSWL
FE99 A0 F0      689          ldy #COUT2
FE9B          690          ;
FE9B A5 3E      691 IOPORT    lda A2L
FE9D 29 0F      692          and #$0F
FE9F F0 04      693          beq >1
FEA1          694          ;
FEA1 09 C0      695          ora /PAGEC0
FEA3 A0 00      696          ldy #ZERO
FEA5          697          ;
FEA5 94 00      698 ^1      sty LOC0,X
FEA7 95 01      699          sta LOC1,X
FEA9          700          ;
FEA9 A0 0E      701          ldy #14
FEAB          702          ;
FEAB 4C B4 FB   703          jmp GOTOROM
FEAE          704          ;
FEAE          705          ;
FEAE F0 55      706 ZAPMEM    beq ZAPMEM2          ; Y-reg = 0, always taken
FEB0          707          ;
FEB0          708          ;
FEB0 4C 00 E0   709 XBASIC    jmp BASIC          ; cold start
FEB3          710          ;
FEB3          711          ;
FEB3 4C 03 E0   712 BASCONT   jmp BASIC2          ; warm start
FEB6          713          ;
FEB6          714          ;
FEB6 20 75 FE   715 GO        jsr A1PC
FEB9 20 3F FF   716          jsr RESTORE
FEBF          717          ;
FEBF 6C 3A 00   718          jmp (PCL)
FEBF          719          ;
FEBF          720          ;
FEBF 4C D7 FA   721 REGZ      jmp REGDSP
FEC2          722          ;
FEC2          723          ;
FEC2 C6 34      724 TRACE     dec YSAV
FEC4          725          ;
FEC4 8D 07 C0   726 STEPZ     sta CXROMON
FEC7          727          ;
FEC7 4C 08 C5   728          jmp CXSTEP
FECA          729          ;
FECA          730          ;
FECA          731          ; Jump to user ^Y vector.

```

```

FECA          732 ;
FECA 4C F8 03 733 USR      jmp USRYHAND
FECD          734 ;
FECD          735 ;
FECD          736      dfs 1,ZERO          ; 1 byte
FECE          737 ;
FECE          738 ;
FECE C9 F1    739 CHRTBLX3 cmp #$89+$B0^"X"      ; SEARCH command
FED0 D0 3C    740      bne RTN.FF.0
FED2          741 ;
FED2          742 ; New 'X' command, used in the following ways:
FED2          743 ;
FED2          744 ; `char<$strt.$end X <cr>      "char<$strt.$end X <cr>
FED2          745 ; `char`char<$strt.$end X <cr> "char"char<$strt.$end X <cr>
FED2          746 ; `char"char<$strt.$end X <cr> "char`char<$strt.$end X <cr>
FED2          747 ;
FED2          748 ; $val<$strt.$end X <cr> where $val is MSB/LSB
FED2          749 ;
FED2 A0 01    750 SEARCH   ldy #1
FED4          751 ;
FED4 A5 43    752      lda A4H
FED6 F0 04    753      beq >1
FED8          754 ;
FED8 D1 3C    755      cmp (A1L),Y
FEDA D0 0A    756      bne >2
FEDC          757 ;
FEDC 88       758 ^1     dey
FEDD          759 ;
FEDD A5 42    760      lda A4L
FEDF D1 3C    761      cmp (A1L),Y
FEE1 D0 03    762      bne >2
FEE3          763 ;
FEE3 20 92 FD 764      jsr PRA1
FEE6          765 ;
FEE6 20 BA FC 766 ^2     jsr NEXTA1
FEE9 90 E7    767      bcc SEARCH
FEEB          768 ;
FEEB 20 8E FD 769      jsr CROUT
EEEE          770 ;
EEEE 4C F9 FE 771      jmp CRMON2          ; fix program counter
FEF1          772 ;
FEF1          773 ;
FEF1          774 ; Enter the Mini-Assembler.  Fall into CRMON.
FEF1          775 ;
FEF1 A0 0D    776 MINIASM  ldy #13
FEF3          777 ;
FEF3 20 B4 FB 778      jsr GOTOROM
FEF6          779 ;
FEF6          780 ;
FEF6 20 00 FE 781 CRMON   jsr BL1
FEF9          782 ;
FEF9 68       783 CRMON2  pla
FEFA 68       784      pla
FEFB          785 ;
FEFB D0 6C    786      bne MON2          ; always taken
FEFD          787 ;
FEFD          788 ;
FEFD          789 ; Enter the cassette READ routine.  If CHKSUM compares then
FEFD          790 ; sound BELL and return to CALLER or enter PRNTERR.
FEFD          791 ;
FEFD          792 ; This command is used in the following way:

```

```

FEFD      793 ;
FEFD      794 ; $strt.$end R <cr>
FEFD      795 ;
FEFD 20 9F F3 796 READ      jsr CXREAD
FF00      797 ;
FF00 F0 38   798          beq BELL
FF02      799 ;
FF02 D0 29   800          bne PRNTERR          ; always taken
FF04      801 ;
FF04      802 ;
FF04      803          dfs 1,ZERO          ; 1 byte
FF05      804 ;
FF05      805 ;
FF05      806 ; New 'Z' command, used in the following way:
FF05      807 ;
FF05      808 ; $val<$strt.$end Z <cr>
FF05      809 ;
FF05 A5 42   810 ZAPMEM2   lda A4L
FF07 91 3C   811          sta (A1L),Y
FF09      812 ;
FF09 20 BA FC 813          jsr NEXTA1
FF0C 90 F7   814          bcc ZAPMEM2
FF0E      815 ;
FF0E 60      816 RTN.FF.0 rts
FF0F      817 ;
FF0F      818 ;
FF0F C9 EB   819 CHRTBLX2  cmp #$89+$B0^"R"          ; READ command
FF11 F0 EA   820          beq READ
FF13      821 ;
FF13 C9 A0   822          cmp #$89+$B0^"'"          ; APOSTROPHE command
FF15 D0 B7   823          bne CHRTBLX3
FF17      824 ;
FF17 18      825          clc
FF18      826 ;
FF18 08      827 LOOKASC   php
FF19      828 ;
FF19 B9 00 02 829          lda INPUT,Y
FF1C      830 ;
FF1C C9 8D   831          cmp #RETURN
FF1E F0 0A   832          beq >2
FF20      833 ;
FF20 28      834          plp
FF21 B0 02   835          bcs >1
FF23      836 ;
FF23 29 7F   837          and #MSBCLR
FF25      838 ;
FF25 A2 07   839 ^1      ldx #7
FF27      840 ;
FF27 C8      841          iny
FF28 D0 66   842          bne NEXTBIT          ; always taken
FF2A      843 ;
FF2A 28      844 ^2      plp
FF2B F0 7A   845          beq GETNUM          ; always taken
FF2D      846 ;
FF2D      847 ;
FF2D      848 ; Print "ERR" and fall into BELL.
FF2D      849 ;
FF2D A9 C5   850 PRNTERR   lda #"E"
FF2F 20 ED FD 851          jsr COUT
FF32      852 ;
FF32 A9 D2   853          lda #"R"

```

```

FF34 20 ED FD      854          jsr COUT
FF37 20 ED FD      855          jsr COUT
FF3A              856          ;
FF3A              857          ;
FF3A A9 87         858 BELL      lda #ASCIBELL
FF3C              859          ;
FF3C 4C ED FD      860          jmp COUT
FF3F              861          ;
FF3F              862          ;
FF3F              863          ; Restore the 65C02 registers.
FF3F              864          ;
FF3F A5 48         865 RESTORE   lda PREG
FF41 48            866          pha
FF42              867          ;
FF42 A5 45         868          lda AREG
FF44 A6 46         869          ldx XREG
FF46 A4 47         870          ldy YREG
FF48              871          ;
FF48 28            872          plp
FF49              873          ;
FF49 60            874          rts
FF4A              875          ;
FF4A              876          ;
FF4A              877          ; Save the 65C02 registers.
FF4A              878          ;
FF4A 85 45         879 SAVE      sta AREG
FF4C              880          ;
FF4C 86 46         881 SAVE2     stx XREG
FF4E 84 47         882          sty YREG
FF50              883          ;
FF50 08            884          php
FF51              885          ;
FF51 68            886          pla
FF52 85 48         887          sta PREG
FF54              888          ;
FF54 BA           889          tsx
FF55 86 49         890          stx SPNT
FF57              891          ;
FF57 D8            892          cld
FF58              893          ;
FF58 60            894 IORTS     rts
FF59              895          ;
FF59              896          ;
FF59 20 08 FB      897 OLDRST   jsr RSETINIT          ; not referenced
FF5C              898          ;
FF5C 4C 65 FF      899          jmp MON
FF5F              900          ;
FF5F              901          ;
FF5F C9 9B         902 CHRTBLX1 cmp #$89+$B0^""          ; QUOTE command
FF61 F0 B5         903          beq LOOKASC
FF63              904          ;
FF63 D0 AA         905          bne CHRTBLX2          ; always taken
FF65              906          ;
FF65              907          ;
FF65              908          ; Monitor entry point.
FF65              909          ;
FF65 D8            910 MON       cld
FF66              911          ;
FF66 20 3A FF      912          jsr BELL
FF69              913          ;
FF69 A9 AA         914 MON2     lda #"*"

```

```

FF6B 85 33      915      sta PROMPT
FF6D            916      ;
FF6D 20 67 FD   917      jsr GETLINE2
FF70 20 C7 FF   918      jsr ZMODE
FF73            919      ;
FF73            920      ;
FF73 20 A7 FF   921  NEXTITM  jsr GETNUM
FF76            922      ;
FF76 84 34      923      sty YSAV
FF78            924      ;
FF78 A0 17      925      ldy #SUBTBL-CHRTBL
FF7A            926      ;
FF7A            927      ;
FF7A 88         928  CHRSRCH  dey
FF7B 30 E8      929      bmi MON
FF7D            930      ;
FF7D D9 CC FF   931      cmp CHRTBL,Y
FF80 D0 F8      932      bne CHRSRCH
FF82            933      ;
FF82 20 BE FF   934      jsr TOSUBR
FF85            935      ;
FF85 A4 34      936      ldy YSAV          ; process next entry
FF87            937      ;
FF87 4C 73 FF   938      jmp NEXTITM
FF8A            939      ;
FF8A            940      ;
FF8A A2 03      941  DIG      ldx #3
FF8C            942      ;
FF8C 0A         943      asl
FF8D 0A         944      asl
FF8E 0A         945      asl
FF8F 0A         946      asl
FF90            947      ;
FF90            948      ;
FF90 0A         949  NEXTBIT  asl
FF91            950      ;
FF91 26 3E      951      rol A2L
FF93 26 3F      952      rol A2H
FF95            953      ;
FF95 CA        954      dex
FF96 10 F8      955      bpl NEXTBIT
FF98            956      ;
FF98            957      ;
FF98            958      ; If MODE is 0x00 copy A2 to A1 and A3.
FF98            959      ;
FF98 A5 31      960      ^1      lda MODE
FF9A D0 06      961      bne >2
FF9C            962      ;
FF9C B5 3F      963      lda A2H,X
FF9E 95 3D      964      sta A1H,X
FFA0 95 41      965      sta A3H,X
FFA2            966      ;
FFA2 E8         967      ^2      inx
FFA3 F0 F3      968      beq <1
FFA5            969      ;
FFA5 D0 06      970      bne NEXTCHR          ; always taken
FFA7            971      ;
FFA7            972      ;
FFA7 A2 00      973  GETNUM  ldx #ZERO
FFA9 86 3E      974      stx A2L
FFAB 86 3F      975      stx A2H

```

```

FFAD          976 ;
FFAD          977 ;
FFAD 20 FD FC 978 NEXTCHR jsr UPRMON
FFB0          979 ;
FFB0 49 B0    980          eor #"0"
FFB2          981 ;
FFB2 C9 0A    982          cmp #10
FFB4 90 D4    983          bcc DIG
FFB6          984 ;
FFB6 69 88    985          adc #$88
FFB8 C9 FA    986          cmp #$FA
FFBA B0 CE    987          bcs DIG
FFBC          988 ;
FFBC 90 A1    989          bcc CHRTBLX1          ; always taken
FFBE          990 ;
FFBE          991 ;
FFBE          992 ; Get address of PAGEFE subroutine.
FFBE          993 ;
FFBE A9 FE    994 TOSUBR   lda /PAGEFE
FFC0 48       995          pha
FFC1          996 ;
FFC1 B9 E3 FF 997          lda SUBTBL,Y
FFC4 48       998          pha
FFC5          999 ;
FFC5 A5 31   1000          lda MODE
FFC7          1001 ;
FFC7          1002 ;
FFC7 A0 00   1003 ZMODE    ld y #ZERO
FFC9 84 31   1004          sty MODE
FFCB          1005 ;
FFCB 60      1006          rts
FFCC          1007 ;
FFCC          1008 ;
FFCC          1009 CHRTBL:
FFCC BC      1010          byt $89+$B0^CTRLC      ; ctrl-C, jmp 0xE003
FFCD B2      1011          byt $89+$B0^CTRLY      ; ctrl-Y, jmp 0x3F8
FFCE BE      1012          byt $89+$B0^CTRLE      ; ctrl-E, display registers
FFCF ED      1013          byt $89+$B0^"T"        ; T, trace
FFD0 EF      1014          byt $89+$B0^"V"        ; V, verify memory
FFD1 C4      1015          byt $89+$B0^CTRLK      ; ctrl-K, set input device
FFD2 EC      1016          byt $89+$B0^"S"        ; S, step
FFD3 A9      1017          byt $89+$B0^CTRLP      ; ctrl-P, set output device
FFD4 BB      1018          byt $89+$B0^CTRLB      ; ctrl-B, jmp 0xE000
FFD5 A6      1019          byt $89+$B0^"- "       ; -, math operator
FFD6 A4      1020          byt $89+$B0^"+ "       ; +, math operator
FFD7 06      1021          byt $89+$B0^"M"        ; M, move memory
FFD8 95      1022          byt $89+$B0^"<"        ; <, move direction
FFD9 07      1023          byt $89+$B0^"N"        ; N, set normal
FFDA 02      1024          byt $89+$B0^"I"        ; I, set inverse
FFDB 05      1025          byt $89+$B0^"L"        ; L, list memory
FFDC 9A      1026          byt $89+$B0^"! "       ; !, enter mini-assembler
FFDD 00      1027          byt $89+$B0^"G"        ; G, jmp to memory
FFDE F3      1028          byt $89+$B0^"Z"        ; Z, fill memory with value
FFDF 93      1029          byt $89+$B0^": "       ; :, input instruction mode
FFE0 A7      1030          byt $89+$B0^"."        ; ., memory delimiter
FFE1 C6      1031          byt $89+$B0^RETURN    ; CR, carriage return
FFE2 99      1032          byt $89+$B0^" "       ; space, input instruct mode
FFE3          1033 ;
FFE3          1034 ;
FFE3          1035 SUBTBL:
FFE3 B2      1036          byt BASCONT-1          ; ctrl-C <cr>

```

```

FFE4 C9      1037      byt  USR-1      ; ctrl-Y <cr>
FFE5 BE      1038      byt  REGZ-1     ; ctrl-E <cr>
FFE6 C1      1039      byt  TRACE-1    ; trace
FFE7 35      1040      byt  VERIFY-1   ; $dst<$strt.$end V <cr>
FFE8 8C      1041      byt  INPORT2-1  ; #slot ctrl-K <cr>
FFE9 C3      1042      byt  STEPZ-1    ; step
FFEA 96      1043      byt  OUTPORT2-1  ; #slot ctrl-P <cr>
FFEB AF      1044      byt  XBASIC-1   ; ctrl-B <cr>
FFEC 17      1045      byt  SETMODE-1   ; $val-$val <cr>
FFED 17      1046      byt  SETMODE-1   ; $val+$val <cr>
FFEE 2B      1047      byt  MOVE-1     ; $dst<$strt.$end M <cr>
FFEF 1F      1048      byt  LT-1       ; $dst<$strt ::: <cr>
FFF0 83      1049      byt  SETNORM-1   ; N <cr>
FFF1 7F      1050      byt  SETINV-1    ; I <cr>
FFF2 5D      1051      byt  LIST-1     ; $adr L <cr>
FFF3 F0      1052      byt  MINIASM-1   ; mini-assembler
FFF4 B5      1053      byt  GO-1       ; $adr G <cr>
FFF5 AD      1054      byt  ZAPMEM-1    ; $val<$strt.$end Z <cr>
FFF6 17      1055      byt  SETMODE-1   ; $adr:$val ... <cr>
FFF7 17      1056      byt  SETMODE-1   ; $strt.$end
FFF8 F5      1057      byt  CRMON-1     ; <cr>
FFF9 03      1058      byt  BLANK-1     ; <space>
FFFA         1059      ;
FFFA         1060      ;
FFFA FB 03   1061      adr  NMASKIRQ
FFFC 62 FA   1062      adr  RESET
FFFE FA C3   1063      adr  IRQRTN
0000         1064      ;
0000         1065      ;

```

BSAVE F0ROM,D1,A\$1000,B,L\$1000

```

0000      1066      usr  F0ROM,D1
0000      1067      ;
0000      1068      ;      dcm "CD D2"
0000      1069      ;
0000      1070      ;
0000      1071      stt  "ROM2E Symbol Table"
0000      1072      ;
0000      1073      ;
0000      1074      end  111

```

*** End of Assembly

Symbol List starts at 0x7800, ends at 0xA6D6, used 0x2ED6, remaining 0x05EA

Symbols unsorted:

LOC0	0000	LOC1	0001	LOC2	0002	R0L	0000	R0H	0001
R12L	0018	R12H	0019	R14L	001C	R14H	001D	R15L	001E
R15H	001F	GOWARM	0000	GOSTROUT	0003	GOUSR	000A	JMPADRS	0090
CHARAC	000D	ENDCHR	000E	EOLPTR	000F	NUMDIM	000F	TOKNCNTR	000F
DIMFLG	0010	VALTYP	0011	DATAFLG	0013	GARFLG	0013	SUBFLG	0014
INPUTFLG	0015	CPRMASK	0016	SIGNFLG	0016	SHAPE	001A	COLBITS	001C
COLCOUNT	001D	WNDLFT	0020	WNDWDTH	0021	WNDTOP	0022	WNCBTM	0023
CH	0024	CV	0025	GBASL	0026	GBASH	0027	BASL	0028
BASH	0029	BAS2L	002A	BAS2H	002B	H2	002C	LMNEM	002C
V2	002D	RMNEM	002D	CHKSUM	002E	FORMAT	002E	MASK	002E
LENGTH	002F	LASTIN	002F	SIGN	002F	COLOR	0030	HMASK	0030
MODE	0031	INVFLG	0032	PROMPT	0033	YSAV	0034	YSAV1	0035
CSWL	0036	CSWH	0037	KSWL	0038	KSWH	0039	PCL	003A
PCH	003B	A1L	003C	A1H	003D	A2L	003E	A2H	003F
A3L	0040	A3H	0041	A4L	0042	A4H	0043	A5L	0044
MACSTAT	0044	OPRND	0044	AREG	0045	XREG	0046	YREG	0047
PREG	0048	SPNT	0049	RNDL	004E	RNDH	004F	LINNUM	0050
ACL	0050	ACH	0051	TEMPPT	0052	LASTPT	0053	TEMPST	0055
INDEX	005E	DEST	0060	OFFSET	0061	MULMANT	0062	MULGUARD	0066
PRGTAB	0067	VARTAB	0069	ARYTAB	006B	STREND	006D	FRETOP	006F
FRESPEC	0071	MEMSIZE	0073	CURLIN	0075	OLDLIN	0077	TEXTPTR	0079
DATLIN	007B	DATPTR	007D	SRCPTR	007F	VARNAM	0081	VARPNT	0083
FORPNT	0085	LASTOP	0087	TXPTRSAV	0087	CPRTYPE	0089	FUNCNAM	008A
TEMP3	008A	DSCPTR	008C	DSCLN	008F	SPCLFLAG	008F	RTNADLEN	0091
ARGGUARD	0092	TEMP1	0093	HIGHDS	0094	LEN	0094	PROCESS	0095
HIGHTR	0096	TEMP2	0098	COUNTER	0099	EXPCOUNT	009A	LOWTR	009B
DSCTMP	009D	FACEXP	009D	FACMANT	009E	VARPTR	00A0	FACSIGN	00A2
SERLEN	00A3	SAVARGEX	00A4	ARGEXP	00A5	ARGMANT	00A6	ARGSIGN	00AA
XORSIGN	00AB	FACGUARD	00AC	STRING1	00AB	STRING2	00AD	SAVY	00AD
PRGEND	00AF	CHRGTRDR	00B1	TXTPTR	00B8	IRAND	00C9	TOKEN	00CD
HRXSTR	00D0	HRYSTRT	00D2	HRXEND	00D3	HRYEND	00D5	SHVAL	00D0
ROTQVAL	00D1	ROTHVAL	00D2	ROTVVAL	00D3	ROTHSUM	00D4	ROTVSUM	00D5
SHPOL	00D7	RUNFLAG	00D6	ERRFLG	00D8	ERRLIN	00DA	ERRPOS	00DC
ERRNUM	00DE	ERRSTK	00DF	HRXCOOR	00E0	HRYCOOR	00E2	HRCOLOR	00E4
HRHORZ	00E5	HRPAG	00E6	HRSCALE	00E7	HRSHPTBL	00E8	HRCOLCNT	00EA
FIRST	00F0	SPEEDBYT	00F1	TRACEFLG	00F2	FLASHBYT	00F3	TXTPTRSV	00F4
CURLINSV	00F6	REMSTK	00F8	HRROT	00F9	HLINMOD	0001	FPCDMOD	0001
ZERO	0000	IVARLEN	0002	AVARLEN	0003	MANTLEN	0004	FACLEN	0005
AHDLEN	0005	SVARLEN	0007	PXLSBYTE	0007	BYTEBITS	0008	DFLTDIM	000B
MANTBITS	0020	MOVEMASK	0003	PLOTMASK	0004	SCMDMASK	0007	SROTMASK	000F
INVERSE	003F	FLASH	007F	INVRSE80	007F	MSBCLR	007F	MSBSET	0080
EXPBIAS	0080	LWRMASK	00DF	MAXINPUT	00EF	NEG TWO	00FE	NEGONE	00FF
CTRLB	0082	CTRLC	0083	CTRLE	0085	ASCIBELL	0087	CTRLH	0088
LARROW	0088	DARROW	008A	CTRLJ	008A	CTRLK	008B	UARROW	008B
RETURN	008D	CTRLP	0090	CTRLS	0093	CTRLU	0095	RARROW	0095
CTRLX	0098	CTRLY	0099	ESCAPE	009B	SPACE	00A0	GOODF8	0006
PWRUPBYT	00A5	ERROR.1	00FF	ERROR.2	00FE	TKTAB	00C0	TKTO	00C1
TKFN	00C2	TKSPC	00C3	TKTHEN	00C4	TKAT	00C5	TKNOT	00C6
TKSTEP	00C7	TKPLUS	00C8	TKMINUS	00C9	TKGRTR	00CF	TKEQUAL	00D0
TKSGN	00D2	TKSCRN	00D7	OTVPLUS	0079	OTVMINUS	0079	OTVMULT	007B
OTVDIVID	007B	OTVPOWER	007D	OTVAND	0050	OTVOR	0046	OTVGREAT	007F
OTVEQUAL	007F	OTVLESS	0064	PAGESIZE	0100	STACK	0100	INPUT	0200
XFERADR	03ED	AUTOBRK	03EF	AUTORSET	03F2	PWRSTATE	03F4	USRAHAND	03F5
USRYHAND	03F8	NMASKIRQ	03FB	MASKIRQ	03FE	TEXTPG1	0400	PG1TXLOC	05B0
OLDCH	047B	XMODE	04FB	OURCH	057B	OURCV	05FB	XCHAR	067B
XCOORD	06FB	XTEMP1	077B	OLDBASL	077B	XTEMP2	07FB	OLDBASH	07FB

MSLOT	07F8	PAGE08	0800	PAGE0C	0C00	PAGE10	1000	PAGE20	2000
PAGE40	4000	PAGEBD	BD00	PAGEBF	BF00	PAGEC0	C000	PAGEC1	C100
PAGEC8	C800	PAGEF0	F000	PAGEFE	FE00	KEY	C000	STR80OFF	C000
STR80ON	C001	RAMRDOFF	C002	RAMRDON	C003	RAMWROFF	C004	RAMWRON	C005
CXROMOFF	C006	CXROMON	C007	AUXZPOFF	C008	AUXZPON	C009	C3ROMOFF	C00A
C3ROMON	C00B	VID80OFF	C00C	VID80ON	C00D	ALTCHOFF	C00E	ALTCHON	C00F
CLRKEY	C010	RDBANK2	C011	RDLCRAM	C012	RDRAMRD	C013	RDRAMWR	C014
RDCXROM	C015	RDAUXZP	C016	RDC3ROM	C017	RDSTR80	C018	RDVRTBLK	C019
RDTEXT	C01A	RDMIXED	C01B	RDPAGE2	C01C	RDHIRES	C01D	RDALTCH	C01E
RDVID80	C01F	TAPEOUT	C020	SPKRTOGL	C030	TEXTOFF	C050	TEXTON	C051
MIXEDOFF	C052	MIXEDON	C053	PAGE1ON	C054	PAGE2ON	C055	HIRESOFF	C056
HIRESON	C057	ANN1OFF	C058	ANN2OFF	C05A	ANN3ON	C05D	ANN4ON	C05F
TAPEIN	C060	PB1IN	C061	PB2IN	C062	GC1IN	C064	GCTOGL	C070
RAM2WP	C080	ROM2WE	C081	ROM2WP	C082	RAM2WE	C083	RAM1WP	C088
RAM1WE	C08B	PAGED0	D000	M.6	0040	M.CTL2	0020	M.4	0010
M.CTL	0008	M.2	0004	M.1	0002	M.MOUSE	0001	M.PASCAL	0080
M.CURSOR	0010	M.GOXY	0008	M.VMODE	0004	M.PAS1.0	0002	IOSPACE	C000
DOCXCMD	C100	XCLREOP	C103	XHOME	C119	XSCROLL	C123	XSETWND	C152
XCLREOL2	C160	YSCROLL	C165	YCLREOL	C168	YCLREOL2	C16B	YCLREOP	C170
YSETWND	C173	XRESET	C176	YRESET	C176	YRDKEY	C179	YHOME	C17C
YIOPORT	C18A	XIOPORT	C195	ISO	C1A0	XRDKEY	C1A9	XBASCLC	C1B6
XRDESC	C1BD	XNEWVW	C1CF	XGETFMT	C1D5	YGETFMT	C1D5	XGOMINI	C1E1
YGOMINI	C1E1	XPICKFIX	C1E8	YPICKFIX	C1E8	XCLREOL	C1F1	XCLREOL1	C1F3
XVTAB	C203	CXEXIT	C208	DOCMD	C211	YREGTBLX	C243	YREGTBLX	C24F
XKEYIN	C25B	XSETWNDX	C2A5	XRESETX	C2B5	CXRESET	C2D9	KBDTBL	C2EB
KBDOUT	C2EF	C3SPACE	C300	BASICINT	C300	BASICIN	C305	BASICOUT	C307
AUXMOVE	C311	XFER	C314	BASICENT	C317	JC8	C344	JBASINIT	C347
JPINIT	C34A	JPREAD	C350	JPWRITE	C356	JPSTAT	C35C	SETC8	C36D
DOMOVE	C376	DOXFER	C3C3	IRQDONE	C3F4	IRQRTN	C3FA	NEWIRQ	C47C
RAMSWTBL	C4C1	SWTBLEN	0006	FORM	C4C8	GETNSP	C500	CXSTEP	C508
XJMPX	C575	XRTI	C584	XRTS	C588	PCINC2	C58C	PCINC3	C58E
XJSR	C598	XJMP	C5A3	XJMPAT	C5A4	BRANCH	C5B9	BRANCH2	C5C7
INITBL	C5CD	INITBLEN	0008	XGETFMT2	C5D5	PROCVAR	C600	PROCSPCL	C64E
SW16	C670	SW16B	C679	SW16C	C67F	TOBR	C6A2	SETCMD2	C6B2
RSCMD2	C6C6	BSNSCMD2	C6D7	BRTBL	C6E2	OPTBL	C6E3	RTNCMD	C701
SETCMD	C709	RSCMD	C70B	BSNSCMD	C70D	LDCMD	C70F	STCMD	C718
LD@CMD	C721	ST@CMD	C72B	ST@CMD2	C72D	INRCMD	C733	LDD@CMD	C73A
STD@CMD	C743	POP@CMD	C74C	POP@CMD	C753	STP@CMD	C763	ADDCMD	C76C
SUBCMD	C77A	CPRCMD	C77C	BSCMD	C790	BRCMD	C79C	BNCCMD	C79D
BCCMD	C7AE	BPCMD	C7B1	BMCMD	C7B6	BZCMD	C7BB	BNZCMD	C7C2
BM1CMD	C7C9	BNM1CMD	C7D2	SOUTCMD	C7DB	RSNSCMD	C7E0	SJMPCMD	C7E9
DCRCMD	C7F7	PXINIT	C800	BASCINIT	C803	C3HOOKS	C82A	C3IN	C832
PXREAD	C84D	CSETUP	C850	CXNEWVW	C870	CXNEWVW2	C874	CXVIDCK3	C87C
CXVIDCK4	C87E	CTRLON	C8BD	BIORET	C8C5	XINPUT	C8E6	ESCCHAR	C96B
ESCTABL	C97C	CXKEYIN	C98D	PXWRITE	C9AA	PXINIT2	C9B0	PPINIT	C9B4
PPREAD	C9D6	PPWRITE	C9F0	DOBASL	CA1F	PWRITER	CA29	BITBYT60	CA2E
OPTBLC	CA71	OPTBL	CA7D	TSTROMCD	CA89	TESTCARD	CA90	XBASCALC	CABA
CTRLCHR2	CAD2	CTRLCHR	CAD6	CTRLXFER	CB07	PCURON	CB0D	PCUROFF	CB18
XBS	CB40	XCR	CB51	XEM	CB5F	XFS	CB6B	XUS	CB79
XSO	CB84	XSI	CB8F	CTRLADRL	CB9E	CTRLADRH	CBB9	SCROLLDN	CBD4
XLFF	CBD8	SCROLLUP	CBEB	XFF	CC74	XVT	CC77	XSUB	CC93
XGS	CC97	XGSEOLZ	CC9A	CLR40	CCA5	CLRHAF	CCAD	CLR80	CCBA
CLR2	CCD2	XDC1	CCE7	XDC1.2	CCEC	XDC2	CCF9	DO40	CD2B
SETTOP	CD2E	MOUSEOFF	CD3A	MOUSEON	CD41	XNAK	CD4A	SETKEYIN	CD58
SETCOUT2	CD61	FULL40	CD6A	FULL80	CD6E	QUIT	CD7D	SCRN84	CD8E
SCRN48	CDC1	SCRNRET	CDF5	XVTAB2	CDFB	XVTAB3	CE00	XRDKEY2	CE11
PASINV	CE1F	INVERT	CE26	STORCHAR	CE38	PICK	CE44	STORIT	CE70
ESCON	CEB1	ESCOFF	CEC4	ESCRTN	CECD	PSETUP	CED4	COPYROM	CEF4
REL	CF3A	TRYNEXT	CF6B	NXTLINE2	CF9C	CLRROM	CFFF	BASADDR	D000
BSFOR	D002	BSDATA	D006	BSPOP	D042	BSGOTO	D056	BSGOSUB	D060
BSREM	D064	BSPRINT	D074	FN1ADDR	D080	FS1SGN	D080	FS2LEFT	D0AC

FS3LN	D0B2	TAGADDR	D0B4	OTLESS	D0CF	BASNAME	D0D2	MESGS	D257
MESG01	D257	MESG02	D267	MESG03	D26D	MESG04	D281	MESG05	D28C
MESG06	D29C	MESG07	D2A4	MESG08	D2B1	MESG09	D2C4	MESG10	D2D1
MESG11	D2E0	MESG12	D2EE	MESG13	D2FC	MESG14	D309	MESG15	D318
MESG16	D32B	MESG17	D33A	MESG18	D34C	MESG19	D354	MESG20	D359
GTFORPNT	D362	BLTU	D393	CKSTKSIZ	D3D6	CKSTRSIZ	D3E3	PRMESG07	D410
FSCRN	D412	PR.LINUM	D431	RESTART	D43C	ASENTER	D4F2	INLIN	D52C
INLIN2	D52E	PARSINPT	D559	FNDLIN	D61A	FNDLIN2	D61E	BNEW	D649
SCRTH	D64B	SETPTRS	D665	BCLEAR	D66A	CLEARC	D66C	STKINIT	D683
STXTPTR	D697	BLIST	D6A5	GETCHR	D72C	BFOR	D766	FP1.0ADR	D7AF
NEWSTT	D7D2	DOTRACE	D805	DOSTAMT	D828	DOSTAMT2	D82A	BRESTORE	D849
SETDAPTR	D853	RTN.D8.5	D857	ISCNTLC	D858	DOCTRL.C	D863	BSTOP	D86E
BEND	D870	BCONT	D896	DOHANDER	D8B0	RDBYTE	D8BC	RD2BIT	D8CA
RDBIT	D8CD	BLOAD	D8DC	BRUN	D912	BGOSUB	D921	BGOSUB2	D935
BGOTO	D93E	RTN.D9.6	D96A	BRETURN	D96B	BPOP	D96B	DOMESG08	D97C
PRMESG02	D981	BDATA	D995	BDATA2	D998	DATSCAN	D9A3	DATSCAN2	D9A6
BIF	D9C9	BREM	D9DC	BON	D9EC	RTN.DA.0	DA0B	LINGET	DA0C
LINGET2	DA12	BLET	DA46	BLET2	DA63	PUTSTR	DA7B	COPYSTR	DAB7
PRSTRING	DACF	BPRINT	DAD5	BPRINT2	DAD7	PRTCR	DAFB	RTN.DB.0	DB02
STROUT	DB3A	STRPRT	DB3D	OUTSPC	DB57	OUTQSTN	DB5A	OUTCHR	DB5C
INPUTERR	DB71	INPERR	DB86	BGET	DBA0	BINPUT	DBB2	HEXTIN	DBDC
BREAD	DBE2	DO.LIST	DBEB	DO.ITEM	DBF1	INSTART	DC2B	MESG21	DCDF
MESG22	DCEF	BNEXT	DCF9	BNEXT2	DCFF	FRMNUM	DD67	CHKNUM	DD6A
CHKSTR	DD6C	CHKVAL	DD6D	FRMEVAL	DD7B	FRMEVAL2	DD86	FRMEVAL3	DD95
SAVOP	DDD7	FRMRECUR	DDFD	FRMSTAK	DE10	FRMSTAK2	DE15	FRMSTAK3	DE23
NOTMATH	DE38	FRMELMNT	DE63	STRTXT	DE84	OEQUAL	DE9B	PARENCHK	DEB2
CHKCLSP	DEB8	CHKOPNP	DEBB	CHKCOM	DEBE	SYNTAXCHK	DEC0	DOMESG02	DEC9
GETIVAL	DED5	ORR	DF4F	OAND	DF55	OLT	DF65	FPDL	DFCD
BDIM	DFD9	PTRGET	DFE3	PTRGET2	DFE8	PTRGET3	DFEA	GOMESG02	DFE4
BASIC	E000	BASIC2	E003	PTRCONT	E007	CHKASCI	E07D	IVALZERO	E099
PNTARVAL	E0EC	FP8000	E0FD	MAKINT	E102	AYPOSINT	E108	AYINT	E10C
ARRAY	E126	BS.ERR	E196	IQ.ERR	E199	RA.ERR	E19E	FINDELE	E24B
MULSUBS	E2AD	MULSUBS2	E2B6	FFRE	E2DE	GIVAYFP	E2F2	FPOS	E2FF
SNGFLT	E301	ERRDIR	E306	BDEF	E313	GETFNC	E341	CALLFNC	E354
FNCDATA	E3AF	FSTR	E3C5	STRINI	E3D5	STRSPA	E3DD	STRLIT	E3E7
STRLIT2	E3ED	PUTNEW	E42A	GETSSPC	E452	GARBAG	E484	CHKVARS	E48D
CHKARRYS	E4C2	GARBEXIT	E501	MOVVARS	E50C	NXTVAR	E567	COPYVAR	E573
DECPTR	E58C	CAT2STR	E597	MOVINS	E5D4	MOVSTR	E5E2	MOVSTR2	E5E6
FRESTR	E5FD	FRESTR2	E600	FRETMP	E604	FRETMS	E635	FCHR	E646
FLEFT	E65A	FRIGHT	E686	FMID	E691	STRSETUP	E6B9	FLEN	E6D6
GETSTRLN	E6DC	FASC	E6E5	DO.IQ.ER	E6F2	GETBYTC	E6F5	GETBYT	E6F8
CONVINT	E6FB	FVAL	E707	STRCOPY	E73D	GETASNUM	E746	COMBYTE	E74C
GETADDR	E752	FPEEK	E764	BPOKE	E77B	BWAIT	E784	RTN.E7.9	E79F
FADDDHALF	E7A0	FSUB	E7A7	OMINUS	E7AA	FSUB2	E7AA	FADD	E7BE
OPLUS	E7C1	FADD2	E7C1	FADD3	E7CE	CFG2COMP	E829	NORMFAC1	E82E
ZEROFAC	E84E	ZEROFAC2	E850	NORMFAC6	E88F	FACSCOMP	E89E	FACMCOMP	E8A4
FACMINC	E8C6	PRMESG06	E8D5	SHFTFMUL	E8DA	SHFTBYT	E8F0	SHFTBIT	E907
FPLOGE	E913	FPSQR0.5	E918	FPSQR2.0	E91D	FPN0.5	E922	FPLN2	E927
POLY.LOG	E92C	FLN	E941	FMULT	E97F	OMULT	E982	TSTMULT	E9AD
BYTMULT	E9B2	LOADARG	E9E3	PROCEXP	EA0E	PROCEXP2	EA10	CHKOVERR	EA2B
MULFAC10	EA39	FP0.1	EA50	DIVFAC10	EA55	DIVFAC.S	EA5C	DIVFAC	EA5E
FDIV	EA66	ODIVIDE	EA69	COPYM2F	EAE6	LOADFAC	EAF9	COPYF2T1	EB1E
COPYF2T2	EB21	COPYF2FR	EB27	COPYFAC	EB2B	COPYA2F	EB53	COPYA2F2	EB55
COPYF2A	EB63	COPYF2A2	EB66	RNDUP	EB72	INCMANT	EB7A	CKFACSGN	EB82
CKFACSG1	EB86	CKFACSG2	EB88	RTN.EB.8	EB8F	FSGN	EB90	FLOAT	EB93
FLOAT2	EB9B	FLOAT3	EBA0	FABS	EBAF	FPCOMP	EBB2	FPCOMP2	EBB4
FP2INT	EBF2	FINT	EC23	CLRMANT	EC40	RTN.EC.4	EC49	GETINT	EC4A
INTLOOP	EC61	BYT2FP	ECD5	FP9.9E7	ED0A	FP9.9E8	ED0F	FP1.0E9	ED14
PRTMSG19	ED19	LINPRT	ED24	FPOUT	ED34	FPOTEXIT	EE44	SAV2STK	EE4F
FP0.5	EE57	FPDECTBL	EE5C	DECTBLEN	0024	FSQR	EE81	OPOWER	EE97
FPWRT	EE97	OGT	EECD	NEGFAC	EECD	RTN.EE.D	EED7	FPINVLN2	EEDB

POLY.EXP	EEE0	FP1.0	EF04	FEXP	EF09	POLYPROC	EF5C	POLYNOM	EF72
POLYNOM2	EF76	RTN.EF.A	EFA3	RANDVAL1	EFA4	RANDVAL2	EFA8	FRND	EFAC
FCOS	EFEA	FSIN	EFF1	FSIN.2	F023	FTAN	F03A	FTAN.2	F062
FPIDIV2	F066	FP.25	F06B	POLY.SIN	F070	FPIMUL2	F099	FATAN	F09E
POLY.ATN	F0CE	PGZCODE	F10B	CHRGET	00B1	CHRGOT	00B7	FPRAND	00C9
ZPCDLEN	001D	COLDSTRT	F128	FLOG	F1CB	BCALL	F1D5	BIN	F1DE
BPR	F1E5	PLOTFNS	F1EC	GOIQERR	F206	ODRCOOR	F209	BPLOT	F225
BHLIN	F232	BVLIN	F241	BCOLOR	F24F	BVTAB	F256	BSPEED	F262
BTRACE	F26D	BNOTRACE	F26F	BNORMAL	F273	BINVERSE	F276	BFLASH	F280
BHIMEM	F286	BLOMEM	F2A6	BONERR	F2CB	HANDLERR	F2E9	BRESUME	F318
BDEL	F331	BGR	F390	BTEXT	F399	CXREAD	F39F	BHGR2	F3D8
BHGR	F3E2	CLRHIRE	F3EE	HPOSN	F411	HPLOT	F457	HRMOVLF	F465
COLSHIFT	F47E	HRMOVRT	F48A	DRAWHDR	F49E	XDRAWIT	F4A8	DRAWIT	F4BA
HRMOVUP	F4D3	BITBYT03	F503	BITBYT1C	F504	HRMOVDN	F505	BITBYT04	F531
BITABLE	F532	HLIN	F53A	ROTATBL	F5B3	BDRAW	F5C4	BXDRAW	F5C6
FSQR2	F66B	COPYA2F3	F68A	COPYF2A3	F69B	GETFNS	F6B9	DOIQERR	F6E6
BHCOLOR	F6E9	RTN.F6.F	F6F5	HRCOLTBL	F6F6	BHPLOT	F6FE	BHPLOT2	F708
BROT	F721	BSCALE	F727	FRND2	F72D	TITLE	F78F	TITLEN	000A
DELTITLE	001C	OFFTITLE	000E	PARSIEX1	F799	PARSIEX2	F79E	PARSIEX3	F7AC
BLISTEX1	F7BC	BLISTEX2	F7C6	PRTCRESX1	F7C6	PRTCRESX2	F7D5	BHTABEX1	F7DC
RTN.F7.E	F7E6	BHTAB	F7E7	F8SPACE	F800	PLOT	F800	RTMASK	F80C
PLOT1	F80E	HLINE	F819	HLINE1	F81C	VLIN	F828	CLRSCR	F832
CLRTOP	F836	GBASCALC	F847	NXTCOL	F85F	SETCOL	F864	SCRN	F871
SCRN2	F879	INSDS1	F882	INSDS2	F88E	GETFMT	F8A5	TESTROM	F8B6
INSTDSP	F8D0	PRNTOP	F8D4	PRNTBL	F8DB	PRNTYX	F940	PRNTAX	F941
PRNTX	F944	PRBLNK	F948	PRBL2	F94A	PCADJ	F953	PCADJ2	F954
PCADJ3	F956	RTS2	F961	FMT1	F962	MNEML	F9A6	MNEMR	F9EB
CHAR1	FA30	CHAR2	FA36	CXOFF	FA3C	CXRTN	FA3F	OLDIRQ	FA40
NEWBREAK	FA47	BREAK	FA4C	OLDBRK	FA59	RESET	FA62	SWEET16	FA72
SW16RTN	FA78	RESET2	FA7E	NEWMON	FA81	NEWMON2	FA9B	PWRUP	FAA6
REGDSP	FAD7	REGDSP2	FADA	PWRCON	FAFD	PWRCNLEN	0005	DISKID	FB02
DISKIDLN	0006	RSETINIT	FB08	XLATBL	FB14	REGTBL	FB19	REGTBLEN	0005
PREAD	FB1E	INIT	FB2F	INIT2	FB33	SETEXT	FB39	SETGR	FB40
SETWND	FB4B	TABV	FB5B	APPLE2	FB60	STITLE	FB65	SETPWRUP	FB6F
VIDWAIT	FB78	KBDWAIT	FB88	ESCOLD	FB97	ESCNEW	FBA5	SIGROM	FB83
GOTOROM	FBB4	SIGBYTE	FBC0	BASCALC	FBC1	FMT2	FBC7	BELL2	FBD9
RINGBELL	FBDD	STORADV	FBF0	ADVANCE	FBF4	RTN.FB.F	FBFC	VIDOUT	FBFD
BS	FC10	UP	FC1A	VTAB	FC22	VTAB2	FC24	ESCOLD2	FC2C
CLREOP	FC42	NEWVW	FC46	HOME	FC58	GOTOROM2	FC5A	STEPRTN2	FC5F
CR	FC62	LF	FC66	SCROLL	FC70	GOTOIRQ	FC74	IRQDONE2	FC7A
CHKINV	FC95	CLREOL	FC9C	CLREOL2	FC9E	WAIT	FCA8	NEXTA4	FCB4
NEXTA1	FCBA	HEADR	FCC9	STEPRTN	FCCA	XERR	FCD2	FINDOP	FCE3
NXTLINE	FCF0	UPRMON	FCFD	UPRCASE	FD01	RDKEY	FD0C	RDKEY2	FD13
RDKEY3	FD18	KEYIN	FD1B	GOTOROM4	FD1D	RDESC	FD21	NEWRDKEY	FD28
ESC	FD2F	RDCHAR	FD35	PICKFIX	FD3D	NOTCR	FD47	CANCEL	FD62
GETLINE2	FD67	GETLINE	FD6A	BCKSPC	FD71	NXTCHAR	FD75	ADDINP	FD84
CROUT	FD8E	PRA1	FD92	PRXY	FD96	XAM8	FDA3	MOD8CHK	FDAD
XAM	FDB3	DATAOUT	FDB6	XAMPM	FDC6	ADD	FDD1	PRBYTE	FDDA
PRHEX	FDE3	PRHEX2	FDE5	COUT	FDED	COUT2	FDF0	COUTZ	FDF6
COUTZ2	FDF7	BL1	FE00	BLANK	FE04	STOR	FE0B	STOR2	FE0F
NEXTA3	FE11	SETMODE	FE18	SETMDZ	FE1D	LT	FE20	MOVE	FE2C
VERIFY	FE36	LIST	FE5E	LIST2	FE63	A1PC	FE75	A1PCLP	FE78
SETINV	FE80	SETNORM	FE84	SETKBD	FE89	INPORT	FE8B	INPORT2	FE8D
SETVID	FE93	OUTPORT	FE95	OUTPORT2	FE97	IOPORT	FE9B	ZAPMEM	FEAE
XBASIC	FEB0	BASCONT	FEB3	GO	FEB6	REGZ	FEBF	TRACE	FEC2
STEPZ	FEC4	USR	FECA	CHRTBLX3	FECE	SEARCH	FED2	MINIASM	FEF1
CRMON	FEF6	CRMON2	FEF9	READ	FEFD	ZAPMEM2	FF05	RTN.FF.0	FF0E
CHRTBLX2	FF0F	LOOKASC	FF18	PRNTERR	FF2D	BELL	FF3A	RESTORE	FF3F
SAVE	FF4A	SAVE2	FF4C	IORTS	FF58	OLDRST	FF59	CHRTBLX1	FF5F
MON	FF65	MON2	FF69	NEXTITM	FF73	CHRSRCH	FF7A	DIG	FF8A
NEXTBIT	FF90	GETNUM	FFA7	NEXTCHR	FFAD	TOSUBR	FFBE	ZMODE	FFC7

CHRTBL FFCC SUBTBL FFE3

Symbols alphabetically sorted:

A1H	003D	A1L	003C	A1PC	FE75	A1PCLP	FE78	A2H	003F
A2L	003E	A3H	0041	A3L	0040	A4H	0043	A4L	0042
A5L	0044	ACH	0051	ACL	0050	ADD	FDD1	ADDCMD	C76C
ADDINP	FD84	ADVANCE	FBF4	AHDRLEN	0005	ALTCHOFF	C00E	ALTCHON	C00F
ANN1OFF	C058	ANN2OFF	C05A	ANN3ON	C05D	ANN4ON	C05F	APPLE2	FB60
AREG	0045	ARGEXP	00A5	ARGGUARD	0092	ARGMANT	00A6	ARGSIGN	00AA
ARRAY	E126	ARYTAB	006B	ASCIBELL	0087	ASENTER	D4F2	AUTOBRK	03EF
AUTORSET	03F2	AUXMOVE	C311	AUXZPOFF	C008	AUXZPON	C009	AVARLEN	0003
AYINT	E10C	AYPOSINT	E108	BAS2H	002B	BAS2L	002A	BASADDR	D000
BASCALC	FBC1	BASCINIT	C803	BASCONT	FEB3	BASH	0029	BASIC	E000
BASIC2	E003	BASICENT	C317	BASICIN	C305	BASICINT	C300	BASICOUT	C307
BASL	0028	BASNAME	D0D2	BCALL	F1D5	BCCMD	C7AE	BCKSPC	FD71
BCLEAR	D66A	BCOLOR	F24F	BCONT	D896	BDATA	D995	BDATA2	D998
BDEF	E313	BDEL	F331	BDIM	DFD9	BDRAW	F5C4	BELL	FF3A
BELL2	FBD9	BEND	D870	BFLASH	F280	BFOR	D766	BGET	DBA0
BGOSUB	D921	BGOSUB2	D935	BGOTO	D93E	BGR	F390	BHCOLOR	F6E9
BHGR	F3E2	BHGR2	F3D8	BHIMEM	F286	BHLIN	F232	BHPLOT	F6FE
BHPLOT2	F708	BHTAB	F7E7	BHTABEX1	F7DC	BIF	D9C9	BIN	F1DE
BINPUT	DBB2	BINVERSE	F276	BIORET	C8C5	BITABLE	F532	BITBYT03	F503
BITBYT04	F531	BITBYT1C	F504	BITBYT60	CA2E	BL1	FE00	BLANK	FE04
BLET	DA46	BLET2	DA63	BLIST	D6A5	BLISTEX1	F7BC	BLISTEX2	F7C6
BLOAD	D8DC	BLOMEM	F2A6	BLTU	D393	BM1CMD	C7C9	BMCMD	C7B6
BNCCMD	C79D	BNEW	D649	BNEXT	DCF9	BNEXT2	DCFF	BNM1CMD	C7D2
BNORMAL	F273	BNOTRACE	F26F	BNZCMD	C7C2	BON	D9EC	BONERR	F2CB
BPCMD	C7B1	BPLOT	F225	BPOKE	E77B	BPOP	D96B	BPR	F1E5
BPRINT	DAD5	BPRINT2	DAD7	BRANCH	C5B9	BRANCH2	C5C7	BRCMD	C79C
BREAD	DBE2	BREAK	FA4C	BREM	D9DC	BRESTORE	D849	BRESUME	F318
BRETURN	D96B	BROT	F721	BRTBL	C6E2	BRUN	D912	BS	FC10
BS.ERR	E196	BSCALE	F727	BSCMD	C790	BSDATA	D006	BSFOR	D002
BSGOSUB	D060	BSGOTO	D056	BSNSCMD	C70D	BSNSCMD2	C6D7	BSPEED	F262
BSPOP	D042	BSPRINT	D074	BSREM	D064	BSTOP	D86E	BTEXT	F399
BTRACE	F26D	BVLIN	F241	BVTAB	F256	BWAIT	E784	BXDRAW	F5C6
BYT2FP	ECD5	BYTEBITS	0008	BYTMULT	E9B2	BZCMD	C7BB	C3HOOKS	C82A
C3IN	C832	C3ROMOFF	C00A	C3ROMON	C00B	C3SPACE	C300	CALLFNC	E354
CANCEL	FD62	CAT2STR	E597	CFG2COMP	E829	CH	0024	CHAR1	FA30
CHAR2	FA36	CHARAC	000D	CHKARRYS	E4C2	CHKASCI	E07D	CHKCLSP	DEB8
CHKCOM	DEBE	CHKINV	FC95	CHKNUM	DD6A	CHKOPNP	DEBB	CHKOVERR	EA2B
CHKSTR	DD6C	CHKSUM	002E	CHKVAL	DD6D	CHKVARS	E48D	CHRGET	00B1
CHRGOT	00B7	CHRGTADR	00B1	CHRSRCH	FF7A	CHRTBL	FFCC	CHRTBLX1	FF5F
CHRTBLX2	FF0F	CHRTBLX3	FECE	CKFACSG1	EB86	CKFACSG2	EB88	CKFACSGN	EB82
CKSTKSIZ	D3D6	CKSTRSIZ	D3E3	CLEARC	D66C	CLR2	CCD2	CLR40	CCA5
CLR80	CCBA	CLREOL	FC9C	CLREOL2	FC9E	CLREOP	FC42	CLRHAF	CCAD
CLRHIRE	F3EE	CLRKEY	C010	CLRMANT	EC40	CLRROM	CFFF	CLRSCR	F832
CLRTOP	F836	COLBITS	001C	COLCOUNT	001D	COLDSTRT	F128	COLOR	0030
COLSHIFT	F47E	COMBYTE	E74C	CONVINT	E6FB	COPYA2F	EB53	COPYA2F2	EB55
COPYA2F3	F68A	COPYF2A	EB63	COPYF2A2	EB66	COPYF2A3	F69B	COPYF2FR	EB27
COPYF2T1	EB1E	COPYF2T2	EB21	COPYFAC	EB2B	COPYM2F	EAE6	COPYROM	CEF4
COPYSTR	DAB7	COPYVAR	E573	COUNTER	0099	COUT	FDED	COUT2	FDF0
COUTZ	FDF6	COUTZ2	FDF7	CPRCMD	C77C	CPRMASK	0016	CPRTYPE	0089
CR	FC62	CRMON	FEF6	CRMON2	FEF9	CROUT	FD8E	CSETUP	C850
CSWH	0037	CSWL	0036	CTRLADR	CBB9	CTRLADRL	CB9E	CTRLB	0082
CTRLC	0083	CTRLCHR	CAD6	CTRLCHR2	CAD2	CTRLE	0085	CTRLH	0088
CTRLJ	008A	CTRLK	008B	CTRLON	C8BD	CTRLP	0090	CTRLS	0093
CTRLU	0095	CTRLX	0098	CTRLXFER	CB07	CTRLY	0099	CURLIN	0075
CURLINSV	00F6	CV	0025	CXEXIT	C208	CXKEYIN	C98D	CXNEWVW	C870
CXNEWVW2	C874	CXOFF	FA3C	CXREAD	F39F	CXRESET	C2D9	CXROMOFF	C006

CXROMON	C007	CXRTN	FA3F	CXSTEP	C508	CXVIDCK3	C87C	CXVIDCK4	C87E
DARROW	008A	DATAFLG	0013	DATAOUT	FDB6	DATLIN	007B	DATPTR	007D
DATSCAN	D9A3	DATSCAN2	D9A6	DCRCMD	C7F7	DECPTR	E58C	DECTBLN	0024
DELTITLE	001C	DEST	0060	DFLTDIM	000B	DIG	FF8A	DIMFLG	0010
DISKID	FB02	DISKIDLN	0006	DIVFAC	EA5E	DIVFAC.S	EA5C	DIVFAC10	EA55
DO.IQ.ER	E6F2	DO.ITEM	DBF1	DO.LIST	DBEB	DO40	CD2B	DOBASL	CA1F
DOCMD	C211	DOCTRL.C	D863	DOCXCMD	C100	DOHANDER	D8B0	DOIQERR	F6E6
DOMESG02	DEC9	DOMESG08	D97C	DOMOVE	C376	DOSTAMT	D828	DOSTAMT2	D82A
DOTRACE	D805	DOXFER	C3C3	DRAWHDR	F49E	DRAWIT	F4BA	DSCLEN	008F
DSCPTR	008C	DSCTMP	009D	ENDCHR	000E	EOLPTR	000F	ERRDIR	E306
ERRFLG	00D8	ERRLIN	00DA	ERRNUM	00DE	ERROR.1	00FF	ERROR.2	00FE
ERRPOS	00DC	ERRSTK	00DF	ESC	FD2F	ESCAPE	009B	ESCCHAR	C96B
ESCNEW	FBA5	ESCOFF	CEC4	ESCOLD	FB97	ESCOLD2	FC2C	ESCON	CEB1
ESCRTN	CECD	ESCTABL	C97C	EXPBIAS	0080	EXPCOUNT	009A	F8SPACE	F800
FABS	EBAF	FACEXP	009D	FACGUARD	00AC	FACLEN	0005	FACMANT	009E
FACMCOMP	E8A4	FACMINC	E8C6	FACSCOMP	E89E	FACSIGN	00A2	FADD	E7BE
FADD2	E7C1	FADD3	E7CE	FADDHALF	E7A0	FASC	E6E5	FATAN	F09E
FCHR	E646	FCOS	EFEA	FDIV	EA66	FEXP	EF09	FFRE	E2DE
FINDELE	E24B	FINDOP	FCE3	FINT	EC23	FIRST	00F0	FLASH	007F
FLASHBYT	00F3	FLEFT	E65A	FLEN	E6D6	FLN	E941	FLOAT	EB93
FLOAT2	EB9B	FLOAT3	EBA0	FLOG	F1CB	FMID	E691	FMT1	F962
FMT2	FBC7	FMULT	E97F	FN1ADDR	D080	FNCDATA	E3AF	FNDLIN	D61A
FNDLIN2	D61E	FORM	C4C8	FORMAT	002E	FORPNT	0085	FP.25	F06B
FP0.1	EA50	FP0.5	EE57	FP1.0	EF04	FP1.0ADR	D7AF	FP1.0E9	ED14
FP2INT	EBF2	FP8000	E0FD	FP9.9E7	ED0A	FP9.9E8	ED0F	FPCDMOD	0001
FPCOMP	EBB2	FPCOMP2	EBB4	FPDECTBL	EE5C	FPDL	DFCD	FPEEK	E764
FPIDIV2	F066	FPIMUL2	F099	FPINVLN2	EEDB	FPLN2	E927	FPLOGE	E913
FPN0.5	E922	FPOS	E2FF	FPOTEXIT	EE44	FPOUT	ED34	FPRAND	00C9
FPSQR0.5	E918	FPSQR2.0	E91D	FPWRT	EE97	FRESPC	0071	FRESTR	E5FD
FRESTR2	E600	FRETMP	E604	FRETMS	E635	FRETOP	006F	FRIGHT	E686
FRMELMNT	DE63	FRMEVAL	DD7B	FRMEVAL2	DD86	FRMEVAL3	DD95	FRNUM	DD67
FRMRECUR	DDFD	FRMSTAK	DE10	FRMSTAK2	DE15	FRMSTAK3	DE23	FRND	EFAC
FRND2	F72D	FS1SGN	D080	FS2LEFT	D0AC	FS3LN	D0B2	FSCRN	D412
FSGN	EB90	FSIN	EFF1	FSIN.2	F023	FSQR	EE81	FSQR2	F66B
FSTR	E3C5	FSUB	E7A7	FSUB2	E7AA	FTAN	F03A	FTAN.2	F062
FULL40	CD6A	FULL80	CD6E	FUNCNAM	008A	FVAL	E707	GARBAG	E484
GARBEXIT	E501	GARFLG	0013	GBASCALC	F847	GBASH	0027	GBASL	0026
GC1IN	C064	GCTOGL	C070	GETADDR	E752	GETASNUM	E746	GETBYT	E6F8
GETBYTC	E6F5	GETCHR	D72C	GETFMT	F8A5	GETFNC	E341	GETFNS	F6B9
GETINT	EC4A	GETIVAL	DED5	GETLINE	FD6A	GETLINE2	FD67	GETNSP	C500
GETNUM	FFA7	GETSSPC	E452	GETSTRLN	E6DC	GIVAYFP	E2F2	GO	FEB6
GOIQERR	F206	GOMESG02	DFE4	GOODF8	0006	GOSTROUT	0003	GOTOIRQ	FC74
GOTOROM	FBB4	GOTOROM2	FC5A	GOTOROM4	FD1D	GOUSR	000A	GOWARM	0000
GTFORPNT	D362	H2	002C	HANDLERR	F2E9	HEADR	FCC9	HEXTIN	DBDC
HIGHDS	0094	HIGHTR	0096	HIRESOFF	C056	HIRESON	C057	HLIN	F53A
HLINE	F819	HLINE1	F81C	HLINMOD	0001	HMASK	0030	HOME	FC58
HPLOT	F457	HPOSN	F411	HRCOLCNT	00EA	HRCOLOR	00E4	HRCOLTBL	F6F6
HRHORZ	00E5	HRMOVDN	F505	HRMOVLF	F465	HRMOVRT	F48A	HRMOVUP	F4D3
HRPAG	00E6	HRROT	00F9	HRSCALE	00E7	HRSHPTBL	00E8	HRXCOOR	00E0
HRXEND	00D3	HRXSTRT	00D0	HRYCOOR	00E2	HRYEND	00D5	HRYSTRT	00D2
INCMANT	EB7A	INDEX	005E	INIT	FB2F	INIT2	FB33	INITBL	C5CD
INITBLN	0008	INLIN	D52C	INLIN2	D52E	INPERR	DB86	INPORT	FE8B
INPORT2	FE8D	INPUT	0200	INPUTERR	DB71	INPUTFLG	0015	INRCMD	C733
INSDS1	F882	INSDS2	F88E	INSTART	DC2B	INSTDSP	F8D0	INTLOOP	EC61
INVERSE	003F	INVERT	CE26	INVFLG	0032	INVRSE80	007F	IOPORT	FE9B
IORTS	FF58	IOSPACE	C000	IQ.ERR	E199	IRAND	00C9	IRQDONE	C3F4
IRQDONE2	FC7A	IRQRTN	C3FA	ISCNTLC	D858	ISO	C1A0	IVALZERO	E099
IVARLEN	0002	JBASINIT	C347	JC8	C344	JMPADRS	0090	JPINIT	C34A
JPREAD	C350	JPSTAT	C35C	JPWRITE	C356	KBDOUT	C2EF	KBDTBL	C2EB
KBDWAIT	FB88	KEY	C000	KEYIN	FD1B	KSWH	0039	KSWL	0038
LARROW	0088	LASTIN	002F	LASTOP	0087	LASTPT	0053	LD@CMD	C721

LDCMD	C70F	LDD@CMD	C73A	LEN	0094	LENGTH	002F	LF	FC66
LINGET	DA0C	LINGET2	DA12	LINNUM	0050	LINPRT	ED24	LIST	FE5E
LIST2	FE63	LMNEM	002C	LOADARG	E9E3	LOADFAC	EAF9	LOC0	0000
LOC1	0001	LOC2	0002	LOOKASC	FF18	LOWTR	009B	LT	FE20
LWRMASK	00DF	M.1	0002	M.2	0004	M.4	0010	M.6	0040
M.CTL	0008	M.CTL2	0020	M.CURSOR	0010	M.GOXY	0008	M.MOUSE	0001
M.PAS1.0	0002	M.PASCAL	0080	M.VMODE	0004	MACSTAT	0044	MAKINT	E102
MANTBITS	0020	MANTLEN	0004	MASK	002E	MASKIRQ	03FE	MAXINPUT	00EF
MEMSIZE	0073	MESG01	D257	MESG02	D267	MESG03	D26D	MESG04	D281
MESG05	D28C	MESG06	D29C	MESG07	D2A4	MESG08	D2B1	MESG09	D2C4
MESG10	D2D1	MESG11	D2E0	MESG12	D2EE	MESG13	D2FC	MESG14	D309
MESG15	D318	MESG16	D32B	MESG17	D33A	MESG18	D34C	MESG19	D354
MESG20	D359	MESG21	DCDF	MESG22	DCEF	MESGS	D257	MINIASM	FEF1
MIXEDOFF	C052	MIXEDON	C053	MNEML	F9A6	MNEMR	F9EB	MOD8CHK	FDAD
MODE	0031	MON	FF65	MON2	FF69	MOUSEOFF	CD3A	MOUSEON	CD41
MOVE	FE2C	MOVEMASK	0003	MOVINS	E5D4	MOVSTR	E5E2	MOVSTR2	E5E6
MOVVARS	E50C	MSBCLR	007F	MSBSET	0080	MSLOT	07F8	MULFAC10	EA39
MULGUARD	0066	MULMANT	0062	MULSUBS	E2AD	MULSUBS2	E2B6	NEGFAC	EECD
NEGONE	00FF	NEGTWO	00FE	NEWBREAK	FA47	NEWIRQ	C47C	NEWMON	FA81
NEWMON2	FA9B	NEWRDKEY	FD28	NEWSTT	D7D2	NEWVW	FC46	NEXTA1	FCBA
NEXTA3	FE11	NEXTA4	FCB4	NEXTBIT	FF90	NEXTCHR	FFAD	NEXTITM	FF73
NMASKIRQ	03FB	NORMFAC1	E82E	NORMFAC6	E88F	NOTCR	FD47	NOTMATH	DE38
NUMDIM	000F	NXTCHAR	FD75	NXTCOL	F85F	NXTLINE	FCF0	NXTLINE2	CF9C
NXTVAR	E567	OAND	DF55	ODIVIDE	EA69	ODRCOOR	F209	OEQUAL	DE9B
OFFSET	0061	OFFTITLE	000E	OGT	EECD	OLDBASH	07FB	OLDBASL	077B
OLDBRK	FA59	OLDCH	047B	OLDIRQ	FA40	OLDLIN	0077	OLDRST	FF59
OLT	DF65	OMINUS	E7AA	OMULT	E982	OOR	DF4F	OPLUS	E7C1
OPOWER	EE97	OPRND	0044	OPTBL	C6E3	OPTBLC	CA71	OPTBLL	CA7D
OTLESS	D0CF	OTVAND	0050	OTVDIVID	007B	OTVEQUAL	007F	OTVGREAT	007F
OTVLESS	0064	OTVMINUS	0079	OTVMULT	007B	OTVOR	0046	OTVPLUS	0079
OTVPOWER	007D	OURCH	057B	OURCV	05FB	OUTCHR	DB5C	OUTPORT	FE95
OUTPORT2	FE97	OUTQSTN	DB5A	OUTSPC	DB57	PAGE08	0800	PAGE0C	0C00
PAGE10	1000	PAGE1ON	C054	PAGE20	2000	PAGE2ON	C055	PAGE40	4000
PAGEBD	BD00	PAGEBF	BF00	PAGEC0	C000	PAGEC1	C100	PAGEC8	C800
PAGED0	D000	PAGEF0	F000	PAGEFE	FE00	PAGESIZE	0100	PARENCHK	DEB2
PARSIEX1	F799	PARSIEX2	F79E	PARSIEX3	F7AC	PARSINPT	D559	PASINV	CE1F
PB1IN	C061	PB2IN	C062	PCADJ	F953	PCADJ2	F954	PCADJ3	F956
PCH	003B	PCINC2	C58C	PCINC3	C58E	PCL	003A	PCUROFF	CB18
PCURON	CB0D	PG1TXLOC	05B0	PGZCODE	F10B	PICK	CE44	PICKFIX	FD3D
PLOT	F800	PLOT1	F80E	PLOTFNS	F1EC	PLOTMASK	0004	PNTARVAL	E0EC
POLY.ATN	F0CE	POLY.EXP	EEE0	POLY.LOG	E92C	POLY.SIN	F070	POLYNOM	EF72
POLYNOM2	EF76	POLYPROC	EF5C	POP@CMD	C753	POPD@CMD	C74C	PPINIT	C9B4
PPREAD	C9D6	PPWRITE	C9F0	PR.LINUM	D431	PRA1	FD92	PRBL2	F94A
PRBLNK	F948	PRBYTE	FDDA	PREAD	FB1E	PREG	0048	PRGEND	00AF
PRGTAB	0067	PRHEX	FDE3	PRHEX2	FDE5	PRMESG02	D981	PRMESG06	E8D5
PRMESG07	D410	PRNTAX	F941	PRNTBL	F8DB	PRNTERR	FF2D	PRNTOP	F8D4
PRNTX	F944	PRNTYX	F940	PROCESS	0095	PROCEXP	EA0E	PROCEXP2	EA10
PROCSPCL	C64E	PROCVAR	C600	PROMPT	0033	PRSTRING	DACF	PRTCR	DAFB
PRTCRESX1	F7C6	PRTCRESX2	F7D5	PRTMSG19	ED19	PRXY	FD96	PSETUP	CED4
PTRCONT	E007	PTRGET	DFE3	PTRGET2	DFE8	PTRGET3	DFEA	PUTNEW	E42A
PUTSTR	DA7B	PWRCNLEN	0005	PWRCON	FAFD	PWRITER	CA29	PWRSTATE	03F4
PWRUP	FAA6	PWRUPBYT	00A5	PXINIT	C800	PXINIT2	C9B0	PXLSBYTE	0007
PXREAD	C84D	PXWRITE	C9AA	QUIT	CD7D	R0H	0001	R0L	0000
R12H	0019	R12L	0018	R14H	001D	R14L	001C	R15H	001F
R15L	001E	RA.ERR	E19E	RAM1WE	C08B	RAM1WP	C088	RAM2WE	C083
RAM2WP	C080	RAMRDOFF	C002	RAMRDON	C003	RAMSWTBL	C4C1	RAMWROFF	C004
RAMWRON	C005	RANDVAL1	EFA4	RANDVAL2	EFA8	RARROW	0095	RD2BIT	D8CA
RDALTCH	C01E	RDAUXZP	C016	RDBANK2	C011	RDBIT	D8CD	RDBYTE	D8BC
RDC3ROM	C017	RDCHAR	FD35	RDCXROM	C015	RDESC	FD21	RDHIRES	C01D
RDKEY	FD0C	RDKEY2	FD13	RDKEY3	FD18	RDLGRAM	C012	RDMIXED	C01B
RDPAGE2	C01C	RDRAMRD	C013	RDRAMWR	C014	RDSTR80	C018	RDTEXT	C01A

RDVID80	C01F	RDVRTBLK	C019	READ	FEFD	REGDSP	FAD7	REGDSP2	FADA
REGTBL	FB19	REGTBLEN	0005	REGZ	FEBF	REL	CF3A	REMSTK	00F8
RESET	FA62	RESET2	FA7E	RESTART	D43C	RESTORE	FF3F	RETURN	008D
RINGBELL	FBDD	RMNEM	002D	RNDH	004F	RNDL	004E	RNDUP	EB72
ROM2WE	C081	ROM2WP	C082	ROTATBL	F5B3	ROTHSUM	00D4	ROTHVAL	00D2
ROTQVAL	00D1	ROTVSUM	00D5	ROTVVAL	00D3	RSCMD	C70B	RSCMD2	C6C6
RSETINIT	FB08	RSNSCMD	C7E0	RTMASK	F80C	RTN.D8.5	D857	RTN.D9.6	D96A
RTN.DA.0	DA0B	RTN.DB.0	DB02	RTN.E7.9	E79F	RTN.EB.8	EB8F	RTN.EC.4	EC49
RTN.EE.D	EED7	RTN.EF.A	EFA3	RTN.F6.F	F6F5	RTN.F7.E	F7E6	RTN.FB.F	FBFC
RTN.FF.0	FF0E	RTNADLEN	0091	RTNCMD	C701	RTS2	F961	RUNFLAG	00D6
SAV2STK	EE4F	SAVARGEX	00A4	SAVE	FF4A	SAVE2	FF4C	SAVOP	DDD7
SAVY	00AD	SCMDMASK	0007	SCRN	F871	SCRN2	F879	SCRN48	CDC1
SCRN84	CD8E	SCRNRET	CDF5	SCROLL	FC70	SCROLLDN	CBD4	SCROLLUP	CBEB
SCRTH	D64B	SEARCH	FED2	SERLEN	00A3	SETC8	C36D	SETCMD	C709
SETCMD2	C6B2	SETCOL	F864	SETCOUT2	CD61	SETDAPTR	D853	SETEXT	FB39
SETGR	FB40	SETINV	FE80	SETKBD	FE89	SETKEYIN	CD58	SETMDZ	FE1D
SETMODE	FE18	SETNORM	FE84	SETPTRS	D665	SETPWRUP	FB6F	SETTOP	CD2E
SETVID	FE93	SETWND	FB4B	SHAPE	001A	SHFTBIT	E907	SHFTBYT	E8F0
SHFTFMUL	E8DA	SHPOLD	00D7	SHPVAL	00D0	SIGBYTE	FBC0	SIGN	002F
SIGNFLG	0016	SIGROM	FBB3	SJMPCMD	C7E9	SNGFLT	E301	SOUTCMD	C7DB
SPACE	00A0	SPCLFLAG	008F	SPEEDBYT	00F1	SPKRTOGL	C030	SPNT	0049
SRCPTR	007F	SROTMASK	000F	ST@CMD	C72B	ST@CMD2	C72D	STACK	0100
STCMD	C718	STD@CMD	C743	STEPRTN	FCCA	STEPRTN2	FC5F	STEPZ	FEC4
STITLE	FB65	STKINIT	D683	STOR	FE0B	STOR2	FE0F	STORADV	FBF0
STORCHAR	CE38	STORIT	CE70	STP@CMD	C763	STR80OFF	C000	STR80ON	C001
STRCOPY	E73D	STREND	006D	STRING1	00AB	STRING2	00AD	STRINI	E3D5
STRLIT	E3E7	STRLIT2	E3ED	STROUT	DB3A	STRPRT	DB3D	STRSETUP	E6B9
STRSPA	E3DD	STRTXT	DE84	STXTPTR	D697	SUBCMD	C77A	SUBFLG	0014
SUBTBL	FFE3	SVARLEN	0007	SW16	C670	SW16B	C679	SW16C	C67F
SW16RTN	FA78	SWEET16	FA72	SWTBLEN	0006	SYNTAXCHK	DEC0	TABV	FB5B
TAGADDR	D0B4	TAPEIN	C060	TAPEOUT	C020	TEMP1	0093	TEMP2	0098
TEMP3	008A	TEMPPT	0052	TEMPST	0055	TESTCARD	CA90	TESTROM	F8B6
TEXTOFF	C050	TEXTON	C051	TEXTPG1	0400	TEXTPTR	0079	TITLE	F78F
TITLEN	000A	TKAT	00C5	TKEQUAL	00D0	TKFN	00C2	TKGRTR	00CF
TKMINUS	00C9	TKNOT	00C6	TKPLUS	00C8	TKSCRN	00D7	TKSGN	00D2
TKSPC	00C3	TKSTEP	00C7	TKTAB	00C0	TKTHEN	00C4	TKTO	00C1
TOBR	C6A2	TOKEN	00CD	TOKNCNTR	000F	TOSUBR	FFBE	TRACE	FEC2
TRACEFLG	00F2	TRYNEXT	CF6B	TSTMULT	E9AD	TSTROMCD	CA89	TXPTRSAV	0087
TXTPTR	00B8	TXTPTRSV	00F4	UARROW	008B	UP	FC1A	UPRCASE	FD01
UPRMON	FCFD	USR	FECA	USRAHAND	03F5	USRYHAND	03F8	V2	002D
VALTYP	0011	VARNAM	0081	VARPNT	0083	VARPTR	00A0	VARTAB	0069
VERIFY	FE36	VID80OFF	C00C	VID80ON	C00D	VIDOUT	FBFD	VIDWAIT	FB78
VLINE	F828	VTAB	FC22	VTAB2	FC24	WAIT	FCA8	WNBDM	0023
WNDLFT	0020	WNDTOP	0022	WNDWDTH	0021	XAM	FDB3	XAM8	FDA3
XAMPM	FDC6	XBASCALC	CABA	XBASCLC	C1B6	XBASIC	FEB0	XBS	CB40
XCHAR	067B	XCLREOL	C1F1	XCLREOL1	C1F3	XCLREOL2	C160	XCLREOP	C103
XCOORD	06FB	XCR	CB51	XDC1	CCE7	XDC1.2	CCEC	XDC2	CCF9
XDRAWIT	F4A8	XEM	CB5F	XERR	FCD2	XFER	C314	XFERADR	03ED
XFF	CC74	XFS	CB6B	XGETFMT	C1D5	XGETFMT2	C5D5	XGOMINI	C1E1
XGS	CC97	XGSEOLZ	CC9A	XHOME	C119	XINPUT	C8E6	XIOPORT	C195
XJMP	C5A3	XJMPAT	C5A4	XJMPX	C575	XJSR	C598	XKEYIN	C25B
XLATBL	FB14	XLF	CBD8	XMODE	04FB	XNAK	CD4A	XNEWVW	C1CF
XORSIGN	00AB	XPICKFIX	C1E8	XRDESC	C1BD	XRDKEY	C1A9	XRDKEY2	CE11
XREG	0046	XRESET	C176	XRESETX	C2B5	XRTI	C584	XRTS	C588
XSCROLL	C123	XSETWND	C152	XSETWNDX	C2A5	XSI	CB8F	XSO	CB84
XSUB	CC93	XTEMP1	077B	XTEMP2	07FB	XUS	CB79	XVT	CC77
XVTAB	C203	XVTAB2	CDFB	XVTAB3	CE00	YCLREOL	C168	YCLREOL2	C16B
YCLREOP	C170	YGETFMT	C1D5	YGOMINI	C1E1	YHOME	C17C	YIOPORT	C18A
YPICKFIX	C1E8	YRDKEY	C179	YREG	0047	YREGTBLX	C243	YREGTBLV	C24F
YRESET	C176	YSAV	0034	YSAV1	0035	YSCROLL	C165	YSETWND	C173
ZAPMEM	FEAE	ZAPMEM2	FF05	ZERO	0000	ZEROFAC	E84E	ZEROFAC2	E850

ZMODE FFC7 ZPCDLEN 001D

Symbols numerically sorted:

ZERO	0000	ROL	0000	LOC0	0000	GOWARM	0000	ROH	0001
M.MOUSE	0001	LOC1	0001	HLINMOD	0001	FPCDMOD	0001	M.PAS1.0	0002
M.1	0002	LOC2	0002	IVARLEN	0002	MOVEMASK	0003	GOSTROUT	0003
AVARLEN	0003	PLOTMASK	0004	MANTLEN	0004	M.VMODE	0004	M.2	0004
REGBLEN	0005	PWRCNLEN	0005	FACLEN	0005	AHDRLEN	0005	SWTBLEN	0006
GOODF8	0006	DISKIDLN	0006	SVARLEN	0007	SCMDMASK	0007	PXLSBYTE	0007
M.GOXY	0008	M.CTL	0008	INITBLEN	0008	BYTEBITS	0008	TITLEN	000A
GOUSR	000A	DFLTDIM	000B	CHARAC	000D	OFFTITLE	000E	ENDCHR	000E
TOKNCNTR	000F	SROTMASK	000F	NUMDIM	000F	EOLPTR	000F	M.CURSOR	0010
M.4	0010	DIMFLG	0010	VALTYP	0011	GARFLG	0013	DATAFLG	0013
SUBFLG	0014	INPUTFLG	0015	SIGNFLG	0016	CPRMASK	0016	R12L	0018
R12H	0019	SHAPE	001A	R14L	001C	DELTITLE	001C	COLBITS	001C
ZPCDLEN	001D	R14H	001D	COLCOUNT	001D	R15L	001E	R15H	001F
WNDLFT	0020	MANTBITS	0020	M.CTL2	0020	WNDWDTH	0021	WNDTOP	0022
WNCBTM	0023	DECTBLEN	0024	CH	0024	CV	0025	GBASL	0026
GBASH	0027	BASL	0028	BASH	0029	BAS2L	002A	BAS2H	002B
LMNEM	002C	H2	002C	V2	002D	RMNEM	002D	MASK	002E
FORMAT	002E	CHKSUM	002E	SIGN	002F	LENGTH	002F	LASTIN	002F
HMASK	0030	COLOR	0030	MODE	0031	INVFLG	0032	PROMPT	0033
YSAV	0034	YSAV1	0035	CSWL	0036	CSWH	0037	KSWL	0038
KSWH	0039	PCL	003A	PCH	003B	A1L	003C	A1H	003D
A2L	003E	INVERSE	003F	A2H	003F	M.6	0040	A3L	0040
A3H	0041	A4L	0042	A4H	0043	OPRND	0044	MACSTAT	0044
A5L	0044	AREG	0045	XREG	0046	OTVOR	0046	YREG	0047
PREG	0048	SPNT	0049	RNDL	004E	RNDH	004F	OTVAND	0050
LINNUM	0050	ACL	0050	ACH	0051	TEMPPT	0052	LASTPT	0053
TEMPST	0055	INDEX	005E	DEST	0060	OFFSET	0061	MULMANT	0062
OTVLESS	0064	MULGUARD	0066	PRGTAB	0067	VARTAB	0069	ARYTAB	006B
STREND	006D	FRETOP	006F	FRESPC	0071	MEMSIZE	0073	CURLIN	0075
OLDLIN	0077	TEXTPTR	0079	OTVPLUS	0079	OTVMINUS	0079	OTVMULT	007B
OTVDIVID	007B	DATLIN	007B	OTVPOWER	007D	DATPTR	007D	SRCPTR	007F
OTVGREAT	007F	OTVEQUAL	007F	MSBCLR	007F	INVRSE80	007F	FLASH	007F
MSBSET	0080	M.PASCAL	0080	EXPBIAS	0080	VARNAM	0081	CTRLB	0082
VARPNT	0083	CTRLC	0083	FORPNT	0085	CTRLE	0085	TXPTRSAV	0087
LASTOP	0087	ASCIBELL	0087	LARROW	0088	CTRLH	0088	CPRTYPE	0089
TEMP3	008A	FUNCNAM	008A	DARROW	008A	CTRLJ	008A	UARROW	008B
CTRLK	008B	DSCPTR	008C	RETURN	008D	SPCLFLAG	008F	DSCLEN	008F
JMPADRS	0090	CTRLP	0090	RTNADLEN	0091	ARGGUARD	0092	TEMP1	0093
CTRLS	0093	LEN	0094	HIGHDS	0094	RARROW	0095	PROCESS	0095
CTRLU	0095	HIGHTR	0096	TEMP2	0098	CTRLX	0098	CTRLY	0099
COUNTER	0099	EXPCOUNT	009A	LOWTR	009B	ESCAPE	009B	FACEXP	009D
DSCTMP	009D	FACMANT	009E	VARPTR	00A0	SPACE	00A0	FACSIGN	00A2
SERLEN	00A3	SAVARGEX	00A4	PWRUPBYT	00A5	ARGEXP	00A5	ARGMANT	00A6
ARGSIGN	00AA	XORSIGN	00AB	STRING1	00AB	FACGUARD	00AC	STRING2	00AD
SAVY	00AD	PRGEND	00AF	CHRGADR	00B1	CHRGET	00B1	CHRGOT	00B7
TXTPTR	00B8	TKTAB	00C0	TKTO	00C1	TKFN	00C2	TKSPC	00C3
TKTHEN	00C4	TKAT	00C5	TKNOT	00C6	TKSTEP	00C7	TKPLUS	00C8
TKMINUS	00C9	IRAND	00C9	FPRAND	00C9	TOKEN	00CD	TKGRTR	00CF
TKEQUAL	00D0	SHPVAL	00D0	HRXSTR	00D0	ROTQVAL	00D1	TKSGN	00D2
ROTHVAL	00D2	HRYSTRT	00D2	ROTVVAL	00D3	HRXEND	00D3	ROTHSUM	00D4
ROTVSUM	00D5	HRYEND	00D5	RUNFLAG	00D6	TKSCRN	00D7	SHPOLD	00D7
ERRFLG	00D8	ERRLIN	00DA	ERRPOS	00DC	ERRNUM	00DE	LWRMASK	00DF
ERRSTK	00DF	HRXCOOR	00E0	HRYCOOR	00E2	HRCOLOR	00E4	HRHORZ	00E5
HRPAG	00E6	HRSCALE	00E7	HRSHPTBL	00E8	HRCOLCNT	00EA	MAXINPUT	00EF
FIRST	00F0	SPEEDBYT	00F1	TRACEFLG	00F2	FLASHBYT	00F3	TXTPTRSV	00F4
CURLINSV	00F6	REMSTK	00F8	HRROT	00F9	NEG TWO	00FE	ERROR.2	00FE

NEGONE	00FF	ERROR.1	00FF	STACK	0100	PAGESIZE	0100	INPUT	0200
XFERADR	03ED	AUTOBRK	03EF	AUTORSET	03F2	PWRSTATE	03F4	USRAHAND	03F5
USRYHAND	03F8	NMASKIRQ	03FB	MASKIRQ	03FE	TEXTPG1	0400	OLDCH	047B
XMODE	04FB	OURCH	057B	PG1TXLOC	05B0	OURCV	05FB	XCHAR	067B
XCOORD	06FB	XTEMP1	077B	OLDBASL	077B	MSLOT	07F8	XTEMP2	07FB
OLDBASH	07FB	PAGE08	0800	PAGE0C	0C00	PAGE10	1000	PAGE20	2000
PAGE40	4000	PAGEBD	BD00	PAGEBF	BF00	STR80OFF	C000	PAGEC0	C000
KEY	C000	IOSPACE	C000	STR80ON	C001	RAMRDOFF	C002	RAMRDON	C003
RAMWROFF	C004	RAMWRON	C005	CXROMOFF	C006	CXROMON	C007	AUXZPOFF	C008
AUXZPON	C009	C3ROMOFF	C00A	C3ROMON	C00B	VID80OFF	C00C	VID80ON	C00D
ALTCHOFF	C00E	ALTCHON	C00F	CLRKEY	C010	RDBANK2	C011	RDLCRAM	C012
RDRAMRD	C013	RDRAMWR	C014	RDCXROM	C015	RDAUXZP	C016	RDC3ROM	C017
RDSTR80	C018	RDVRTBLK	C019	RDTEXT	C01A	RDMIXED	C01B	RDPAGE2	C01C
RDHIRES	C01D	RDALTCH	C01E	RDVID80	C01F	TAPEOUT	C020	SPKRTOGL	C030
TEXTOFF	C050	TEXTON	C051	MIXEDOFF	C052	MIXEDON	C053	PAGE1ON	C054
PAGE2ON	C055	HIRESOFF	C056	HIRESON	C057	ANN1OFF	C058	ANN2OFF	C05A
ANN3ON	C05D	ANN4ON	C05F	TAPEIN	C060	PB1IN	C061	PB2IN	C062
GC1IN	C064	GCTOGL	C070	RAM2WP	C080	ROM2WE	C081	ROM2WP	C082
RAM2WE	C083	RAM1WP	C088	RAM1WE	C08B	PAGEC1	C100	DOCXCMD	C100
XCLREOP	C103	XHOME	C119	XSCROLL	C123	XSETWND	C152	XCLREOL2	C160
YSCROLL	C165	YCLREOL	C168	YCLREOL2	C16B	YCLREOP	C170	YSETWND	C173
YRESET	C176	XRESET	C176	YRDKEY	C179	YHOME	C17C	YIOPORT	C18A
XIOPORT	C195	ISO	C1A0	XRDKEY	C1A9	XBASCLC	C1B6	XRDESC	C1BD
XNEWVW	C1CF	YGETFMT	C1D5	XGETFMT	C1D5	YGOMINI	C1E1	XGOMINI	C1E1
YPICKFIX	C1E8	XPICKFIX	C1E8	XCLREOL	C1F1	XCLREOL1	C1F3	XVTAB	C203
CXEXIT	C208	DOCMD	C211	YREGTBLX	C243	YREGTBLY	C24F	XKEYIN	C25B
XSETWNDX	C2A5	XRESETX	C2B5	CXRESET	C2D9	KBDTBL	C2EB	KBDOUT	C2EF
C3SPACE	C300	BASICINT	C300	BASICIN	C305	BASICOUT	C307	AUXMOVE	C311
XFER	C314	BASICENT	C317	JC8	C344	JBASINIT	C347	JPINIT	C34A
JPREAD	C350	JPWRITE	C356	JPSTAT	C35C	SETC8	C36D	DOMOVE	C376
DOXFER	C3C3	IRQDONE	C3F4	IRQRTN	C3FA	NEWIRQ	C47C	RAMSWTBL	C4C1
FORM	C4C8	GETNSP	C500	CXSTEP	C508	XJMPX	C575	XRTI	C584
XRTS	C588	PCINC2	C58C	PCINC3	C58E	XJSR	C598	XJMP	C5A3
XJMPAT	C5A4	BRANCH	C5B9	BRANCH2	C5C7	INITBL	C5CD	XGETFMT2	C5D5
PROCVAR	C600	PROCSPCL	C64E	SW16	C670	SW16B	C679	SW16C	C67F
TOBR	C6A2	SETCMD2	C6B2	RSCMD2	C6C6	BSNSCMD2	C6D7	BRTBL	C6E2
OPTBL	C6E3	RTNCMD	C701	SETCMD	C709	RSCMD	C70B	BSNSCMD	C70D
LDCMD	C70F	STCMD	C718	LD@CMD	C721	ST@CMD	C72B	ST@CMD2	C72D
INRCMD	C733	LDD@CMD	C73A	STD@CMD	C743	POPD@CMD	C74C	POP@CMD	C753
STP@CMD	C763	ADDCMD	C76C	SUBCMD	C77A	CPRCMD	C77C	BSCMD	C790
BRCMD	C79C	BNCCMD	C79D	BCCMD	C7AE	BPCMD	C7B1	BMCMD	C7B6
BZCMD	C7BB	BNZCMD	C7C2	BM1CMD	C7C9	BNM1CMD	C7D2	SOUTCMD	C7DB
RSNSCMD	C7E0	SJMPCMD	C7E9	DCRCMD	C7F7	PXINIT	C800	PAGEC8	C800
BASCINIT	C803	C3HOOKS	C82A	C3IN	C832	PXREAD	C84D	CSETUP	C850
CXNEWVW	C870	CXNEWVW2	C874	CXVIDCK3	C87C	CXVIDCK4	C87E	CTRLON	C8BD
BIORET	C8C5	XINPUT	C8E6	ESCCHAR	C96B	ESCTABL	C97C	CXKEYIN	C98D
PXWRITE	C9AA	PXINIT2	C9B0	PPINIT	C9B4	PPREAD	C9D6	PPWRITE	C9F0
DOBASL	CA1F	PWRITER	CA29	BITBYT60	CA2E	OPTBLC	CA71	OPTBLL	CA7D
TSTROMCD	CA89	TESTCARD	CA90	XBASCALC	CABA	CTRLCHR2	CAD2	CTRLCHR	CAD6
CTRLXFER	CB07	PCURON	CB0D	PCUROFF	CB18	XBS	CB40	XCR	CB51
XEM	CB5F	XFS	CB6B	XUS	CB79	XSO	CB84	XSI	CB8F
CTRLADRL	CB9E	CTRLADRH	CBB9	SCROLLDN	CBD4	XLF	CBD8	SCROLLUP	CBEB
XFF	CC74	XVT	CC77	XSUB	CC93	XGS	CC97	XGSEOLZ	CC9A
CLR40	CCA5	CLRHAF	CCAD	CLR80	CCBA	CLR2	CCD2	XDC1	CCE7
XDC1.2	CCEC	XDC2	CCF9	DO40	CD2B	SETTOP	CD2E	MOUSEOFF	CD3A
MOUSEON	CD41	XNAK	CD4A	SETKEYIN	CD58	SETCOUT2	CD61	FULL40	CD6A
FULL80	CD6E	QUIT	CD7D	SCRN84	CD8E	SCRN48	CDC1	SCRNRET	CDF5
XVTAB2	CDFB	XVTAB3	CE00	XRDKEY2	CE11	PASINV	CE1F	INVERT	CE26
STORCHAR	CE38	PICK	CE44	STORIT	CE70	ESCON	CEB1	ESCOFF	CEC4
ESCRTN	CECD	PSETUP	CED4	COPYROM	CEF4	REL	CF3A	TRYNEXT	CF6B
NXTLINE2	CF9C	CLRROM	CFFF	PAGED0	D000	BASADDR	D000	BSFOR	D002

BSDATA	D006	BSPOP	D042	BSGOTO	D056	BSGOSUB	D060	BSREM	D064
BSPRINT	D074	FS1SGN	D080	FN1ADDR	D080	FS2LEFT	D0AC	FS3LN	D0B2
TAGADDR	D0B4	OTLESS	D0CF	BASNAME	D0D2	MESGS	D257	MESG01	D257
MESG02	D267	MESG03	D26D	MESG04	D281	MESG05	D28C	MESG06	D29C
MESG07	D2A4	MESG08	D2B1	MESG09	D2C4	MESG10	D2D1	MESG11	D2E0
MESG12	D2EE	MESG13	D2FC	MESG14	D309	MESG15	D318	MESG16	D32B
MESG17	D33A	MESG18	D34C	MESG19	D354	MESG20	D359	GTFORPNT	D362
BLTU	D393	CKSTKSIZ	D3D6	CKSTRSIZ	D3E3	PRMESG07	D410	FSCRN	D412
PR.LINUM	D431	RESTART	D43C	ASENTER	D4F2	INLIN	D52C	INLIN2	D52E
PARSINPT	D559	FNDLIN	D61A	FNDLIN2	D61E	BNEW	D649	SCRATCH	D64B
SETPTRS	D665	BCLEAR	D66A	CLEARC	D66C	STKINIT	D683	STXTPTR	D697
BLIST	D6A5	GETCHR	D72C	BFOR	D766	FP1.0ADR	D7AF	NEWSTT	D7D2
DOTRACE	D805	DOSTAMT	D828	DOSTAMT2	D82A	BRESTORE	D849	SETDAPTR	D853
RTN.D8.5	D857	ISCNTLC	D858	DOCTRL.C	D863	BSTOP	D86E	BEND	D870
BCONT	D896	DOHANDER	D8B0	RDBYTE	D8BC	RD2BIT	D8CA	RDBIT	D8CD
BLOAD	D8DC	BRUN	D912	BGOSUB	D921	BGOSUB2	D935	BGOTO	D93E
RTN.D9.6	D96A	BRETURN	D96B	BPOP	D96B	DOMESG08	D97C	PRMESG02	D981
BDATA	D995	BDATA2	D998	DATSCAN	D9A3	DATSCAN2	D9A6	BIF	D9C9
BREM	D9DC	BON	D9EC	RTN.DA.0	DA0B	LINGET	DA0C	LINGET2	DA12
BLET	DA46	BLET2	DA63	PUTSTR	DA7B	COPYSTR	DAB7	PRSTRING	DACF
BPRINT	DAD5	BPRINT2	DAD7	PRTCR	DAFB	RTN.DB.0	DB02	STROUT	DB3A
STRPRT	DB3D	OUTSPC	DB57	OUTQSTN	DB5A	OUTCHR	DB5C	INPUTERR	DB71
INPERR	DB86	BGET	DBA0	BINPUT	DBB2	HEXTIN	DBDC	BREAD	DBE2
DO.LIST	DBEB	DO.ITEM	DBF1	INSTART	DC2B	MESG21	DCDF	MESG22	DCEF
BNEXT	DCF9	BNEXT2	DCFF	FRMNUM	DD67	CHKNUM	DD6A	CHKSTR	DD6C
CHKVAL	DD6D	FRMEVAL	DD7B	FRMEVAL2	DD86	FRMEVAL3	DD95	SAVOP	DDD7
FRMRECUR	DDFD	FRMSTAK	DE10	FRMSTAK2	DE15	FRMSTAK3	DE23	NOTMATH	DE38
FRMELMNT	DE63	STRTXT	DE84	OEQUAL	DE9B	PARENCHK	DEB2	CHKCLSP	DEB8
CHKOPNP	DEBB	CHKCOM	DEBE	SYNTAXCHK	DEC0	DOMESG02	DEC9	GETIVAL	DED5
OR	DF4F	OAND	DF55	OLT	DF65	FPDL	DFCD	BDIM	DFD9
PTRGET	DFE3	PTRGET2	DFE8	PTRGET3	DFFA	GOMESG02	DFF4	BASIC	E000
BASIC2	E003	PTRCONT	E007	CHKASCI	E07D	IVALZERO	E099	PNTARVAL	E0EC
FP8000	E0FD	MAKINT	E102	AYPOSINT	E108	AYINT	E10C	ARRAY	E126
BS.ERR	E196	IQ.ERR	E199	RA.ERR	E19E	FINDELE	E24B	MULSUBS	E2AD
MULSUBS2	E2B6	FFRE	E2DE	GIVAYFP	E2F2	FPOS	E2FF	SNGFLT	E301
ERRDIR	E306	BDEF	E313	GETFNC	E341	CALLFNC	E354	FNCDATA	E3AF
FSTR	E3C5	STRINI	E3D5	STRSPA	E3DD	STRLIT	E3E7	STRLIT2	E3ED
PUTNEW	E42A	GETSSPC	E452	GARBAG	E484	CHKVARS	E48D	CHKARRYS	E4C2
GARBEXIT	E501	MOVVAR	E50C	NXTVAR	E567	COPYVAR	E573	DECPTR	E58C
CAT2STR	E597	MOVINS	E5D4	MOVSTR	E5E2	MOVSTR2	E5E6	FRESTR	E5FD
FRESTR2	E600	FRETMP	E604	FRETMS	E635	FCHR	E646	FLEFT	E65A
FRIGHT	E686	FMID	E691	STRSETUP	E6B9	FLEN	E6D6	GETSTRLN	E6DC
FASC	E6E5	DO.IQ.ER	E6F2	GETBYTC	E6F5	GETBYT	E6F8	CONVINT	E6FB
FVAL	E707	STRCOPY	E73D	GETASNUM	E746	COMBYTE	E74C	GETADDR	E752
FPEEK	E764	BPOKE	E77B	BWAIT	E784	RTN.E7.9	E79F	FADDDHALF	E7A0
FSUB	E7A7	OMINUS	E7AA	FSUB2	E7AA	FADD	E7BE	OPLUS	E7C1
FADD2	E7C1	FADD3	E7CE	CFG2COMP	E829	NORMFAC1	E82E	ZEROFAC	E84E
ZEROFAC2	E850	NORMFAC6	E88F	FACSCOMP	E89E	FACMCOMP	E8A4	FACMINC	E8C6
PRMESG06	E8D5	SHFTFMUL	E8DA	SHFTBYT	E8F0	SHFTBIT	E907	FPLOGE	E913
FPSQR0.5	E918	FPSQR2.0	E91D	FPN0.5	E922	FPLN2	E927	POLY.LOG	E92C
FLN	E941	FMULT	E97F	OMULT	E982	TSTMULT	E9AD	BYTMULT	E9B2
LOADARG	E9E3	PROCEXP	EA0E	PROCEXP2	EA10	CHKOVERR	EA2B	MULFAC10	EA39
FP0.1	EA50	DIVFAC10	EA55	DIVFAC.S	EA5C	DIVFAC	EA5E	FDIV	EA66
ODIVIDE	EA69	COPYM2F	EAE6	LOADFAC	EAF9	COPYF2T1	EB1E	COPYF2T2	EB21
COPYF2FR	EB27	COPYFAC	EB2B	COPYA2F	EB53	COPYA2F2	EB55	COPYF2A	EB63
COPYF2A2	EB66	RNDUP	EB72	INCMANT	EB7A	CKFACSGN	EB82	CKFACSG1	EB86
CKFACSG2	EB88	RTN.EB.8	EB8F	FSGN	EB90	FLOAT	EB93	FLOAT2	EB9B
FLOAT3	EBA0	FABS	EBAF	FPCOMP	EBB2	FPCOMP2	EBB4	FP2INT	EBF2
FINT	EC23	CLRMANT	EC40	RTN.EC.4	EC49	GETINT	EC4A	INTLOOP	EC61
BYT2FP	ECD5	FP9.9E7	ED0A	FP9.9E8	ED0F	FP1.0E9	ED14	PRTMSG19	ED19
LINPRT	ED24	FPOUT	ED34	FPOTEXIT	EE44	SAV2STK	EE4F	FP0.5	EE57

FPDECTBL	EE5C	FSQR	EE81	OPOWER	EE97	FPWRT	EE97	OGT	EECD
NEGFAC	EECD	RTN.EE.D	EED7	FPINVLN2	EEDB	POLY.EXP	EEE0	FP1.0	EF04
FEXP	EF09	POLYPROC	EF5C	POLYNOM	EF72	POLYNOM2	EF76	RTN.EF.A	EFA3
RANDVAL1	EFA4	RANDVAL2	EFA8	FRND	EFAC	FCOS	EFEA	FSIN	EFF1
PAGEF0	F000	FSIN.2	F023	FTAN	F03A	FTAN.2	F062	FPIDIV2	F066
FP.25	F06B	POLY.SIN	F070	FPIMUL2	F099	FATAN	F09E	POLY.ATN	F0CE
PGZCODE	F10B	COLDSTRT	F128	FLOG	F1CB	BCALL	F1D5	BIN	F1DE
BPR	F1E5	PLOTFNS	F1EC	GOIQERR	F206	ODRCOOR	F209	BPLOT	F225
BHLIN	F232	BVLIN	F241	BCOLOR	F24F	BVTAB	F256	BSPEED	F262
BTRACE	F26D	BNOTRACE	F26F	BNORMAL	F273	BINVERSE	F276	BFLASH	F280
BHIMEM	F286	BLOMEM	F2A6	BONERR	F2CB	HANDLERR	F2E9	BRESUME	F318
BDEL	F331	BGR	F390	BTEXT	F399	CXREAD	F39F	BHGR2	F3D8
BHGR	F3E2	CLRHIRE	F3EE	HPOSN	F411	HPLOT	F457	HRMOVL	F465
COLSHIFT	F47E	HRMOVRT	F48A	DRAWHDR	F49E	XDRAWIT	F4A8	DRAWIT	F4BA
HRMOVUP	F4D3	BITBYT03	F503	BITBYT1C	F504	HRMOVDN	F505	BITBYT04	F531
BITABLE	F532	HLIN	F53A	ROTATBL	F5B3	BDRAW	F5C4	BXDRAW	F5C6
FSQR2	F66B	COPYA2F3	F68A	COPYF2A3	F69B	GETFNS	F6B9	DOIQERR	F6E6
BHCOLOR	F6E9	RTN.F6.F	F6F5	HRCOLTBL	F6F6	BHPLOT	F6FE	BHPLOT2	F708
BROT	F721	BSCALE	F727	FRND2	F72D	TITLE	F78F	PARSIEX1	F799
PARSIEX2	F79E	PARSIEX3	F7AC	BLISTEX1	F7BC	PRTCRESX1	F7C6	BLISTEX2	F7C6
PRTCRESX2	F7D5	BHTABEX1	F7DC	RTN.F7.E	F7E6	BHTAB	F7E7	PLOT	F800
F8SPACE	F800	RTMASK	F80C	PLOT1	F80E	HLINE	F819	HLINE1	F81C
VLINE	F828	CLRSCR	F832	CLRTOP	F836	GBASCALC	F847	NXTCOL	F85F
SETCOL	F864	SCRN	F871	SCRN2	F879	INSDS1	F882	INSDS2	F88E
GETFMT	F8A5	TESTROM	F8B6	INSTDSP	F8D0	PRNTOP	F8D4	PRNTBL	F8DB
PRNTYX	F940	PRNTAX	F941	PRNTX	F944	PRBLNK	F948	PRBL2	F94A
PCADJ	F953	PCADJ2	F954	PCADJ3	F956	RTS2	F961	FMT1	F962
MNEML	F9A6	MNEMR	F9EB	CHAR1	FA30	CHAR2	FA36	CXOFF	FA3C
CXRTN	FA3F	OLDIRQ	FA40	NEWBREAK	FA47	BREAK	FA4C	OLDBRK	FA59
RESET	FA62	SWEET16	FA72	SW16RTN	FA78	RESET2	FA7E	NEWMON	FA81
NEWMON2	FA9B	PWRUP	FAA6	REGDSP	FAD7	REGDSP2	FADA	PWRCON	FAFD
DISKID	FB02	RSETINIT	FB08	XLATBL	FB14	REGTBL	FB19	PREAD	FB1E
INIT	FB2F	INIT2	FB33	SETEXT	FB39	SETGR	FB40	SETWND	FB4B
TABV	FB5B	APPLE2	FB60	STITLE	FB65	SETPWRUP	FB6F	VIDWAIT	FB78
KBDWAIT	FB88	ESCOLD	FB97	ESCNEW	FBA5	SIGROM	FBB3	GOTOROM	FBB4
SIGBYTE	FBC0	BASCALC	FBC1	FMT2	FBC7	BELL2	FBD9	RINGBELL	FBDD
STORADV	FBF0	ADVANCE	FBF4	RTN.FB.F	FBFC	VIDOUT	FBFD	BS	FC10
UP	FC1A	VTAB	FC22	VTAB2	FC24	ESCOLD2	FC2C	CLREOP	FC42
NEWVW	FC46	HOME	FC58	GOTOROM2	FC5A	STEPRTN2	FC5F	CR	FC62
LF	FC66	SCROLL	FC70	GOTOIRQ	FC74	IRQDONE2	FC7A	CHKINV	FC95
CLREOL	FC9C	CLREOL2	FC9E	WAIT	FCA8	NEXTA4	FCB4	NEXTA1	FCBA
HEADR	FCC9	STEPRTN	FCCA	XERR	FCD2	FINDOP	FCE3	NXTLINE	FCF0
UPRMON	FCFD	UPRCASE	FD01	RDKEY	FD0C	RDKEY2	FD13	RDKEY3	FD18
KEYIN	FD1B	GOTOROM4	FD1D	RDESC	FD21	NEWRDKEY	FD28	ESC	FD2F
RDCHAR	FD35	PICKFIX	FD3D	NOTCR	FD47	CANCEL	FD62	GETLINE2	FD67
GETLINE	FD6A	BCKSPC	FD71	NXTCHAR	FD75	ADDINP	FD84	CROUT	FD8E
PRA1	FD92	PRXY	FD96	XAM8	FDA3	MOD8CHK	FDAD	XAM	FDB3
DATAOUT	FDB6	XAMPM	FDC6	ADD	FDD1	PRBYTE	FDDA	PRHEX	FDE3
PRHEX2	FDE5	COUT	FDED	COUT2	FDF0	COUTZ	FDF6	COUTZ2	FDF7
PAGEFE	FE00	BL1	FE00	BLANK	FE04	STOR	FE0B	STOR2	FE0F
NEXTA3	FE11	SETMODE	FE18	SETMDZ	FE1D	LT	FE20	MOVE	FE2C
VERIFY	FE36	LIST	FE5E	LIST2	FE63	A1PC	FE75	A1PCLP	FE78
SETINV	FE80	SETNORM	FE84	SETKBD	FE89	INPORT	FE8B	INPORT2	FE8D
SETVID	FE93	OUTPORT	FE95	OUTPORT2	FE97	IOPORT	FE9B	ZAPMEM	FEAE
XBASIC	FEB0	BASCONT	FEB3	GO	FEB6	REGZ	FEBF	TRACE	FEC2
STEPZ	FEC4	USR	FECA	CHRTBLX3	FECE	SEARCH	FED2	MINIASM	FEF1
CRMON	FEF6	CRMON2	FEF9	READ	FEFD	ZAPMEM2	FF05	RTN.FF.0	FF0E
CHRTBLX2	FF0F	LOOKASC	FF18	PRNTERR	FF2D	BELL	FF3A	RESTORE	FF3F
SAVE	FF4A	SAVE2	FF4C	IORTS	FF58	OLDRST	FF59	CHRTBLX1	FF5F
MON	FF65	MON2	FF69	NEXTITM	FF73	CHRSRCH	FF7A	DIG	FF8A
NEXTBIT	FF90	GETNUM	FFA7	NEXTCHR	FFAD	TOSUBR	FFBE	ZMODE	FFC7

CHRTBL	FFCC	SUBTBL	FFE3
--------	------	--------	------