```
!A

LLOAD CXFPMATH.L,A$4000

*** End of Pass 1

LLOAD CXFPMATH.1.L,A$4000

LLOAD CXFPMATH.2.L,A$4000

LLOAD CXFPMATH.3.L,A$4000

LLOAD CXFPMATH.4.L,A$4000

LLOAD CXFPMATH.5.L,A$4000

LLOAD CXFPMATH.L,A$4000

*** End of Pass 2
```

```
0800                      ttl "CXFPMATH Routines, CXFPMATH.L"
0800                      src "CXFPMATH.L"
0800            ;
0800            ;
0800            ; CXFPMATH.L
0800            ;
0800            ;
0008            REALPTRS epz $08                  ; 0x08:0x0D
0018            IMAGPTRS epz $18                  ; 0x18:0x1D
001E            COUNT    epz $1E
001F            OFFSET   epz $1F
006F            FRETOP   epz $6F
0073            HIMEM    epz $73
0092            ARGGUARD epz $92
009D            FACEXP   epz $9D                  ; FAC exponent
009E            FACMANT  epz $9E                  ; FAC mantissa byte 1
00A2            FACSIGN  epz $A2
00AA            ARGSIGN  epz $AA
00AB            XORSIGN  epz $AB
00AC            FACGUARD epz $AC
00B1            CHRGET   epz $B1
00B7            CHRGOT   epz $B7
0800            ;
0800                     enz
0800            ;
0000            ZERO     equ $00
007F            MSBCLR   equ $7F
0080            MSBSET   equ $80
00FF            NEGONE   equ $FF
0800            ;
0004            ARRAYSIZ equ $04                  ; number of 2x2 array elements
0004            VARS2    equ $04                  ; number of bytes for 2 values
0005            IMAGOFF  equ $05                  ; imaginary to real FP offset
0005            FPLEN    equ $05                  ; floating point number length
0005            CMDLEN   equ $05                  ; length of all input commands
0006            ARGLEN   equ $06                  ; real and imaginary arguments
0006            VARS3    equ $06                  ; number of bytes for 3 values
000A            CXFPOFF  equ IMAGOFF*2            ; next complex array value
0800            ;
008D            RETURN   equ $8D
009B            ESCAPE   equ $9B
00A0            SPACE    equ $A0
00A2            QUOTE    equ $A2
00AC            COMMA    equ $AC
0800            ;
0100            PAGESIZE equ $100
0800            ;
03D0            DOSWARM  equ $3D0
0800            ;
C000            KEY      equ $C000
C010            CLRKEY   equ $C010
0800            ;
C082            ROM2WP   equ $C082
0800            ;
DFE3            PTRGET   equ $DFE3
0800            ;
E7A7            FSUB     equ $E7A7                ; add ARG to negative FAC
E7BE            FADD     equ $E7BE                ; add ARG to FAC
E97F            FMULT    equ $E97F                ; multiply ARG and FAC
E9E3            LOADARG  equ $E9E3                ; load ARG from memory
```

```
EA66              FDIV     equ $EA66          ; divide FAC by ARG
EAF9              LOADFAC  equ $EAF9          ; load FAC from memory
EB2B              COPYFAC  equ $EB2B          ; copy FAC to memory
EB53              MOVFA    equ $EB53          ; copy ARG to FAC
EB90              FSGN     equ $EB90          ; evaluate FAC for 0<,=0,>0
EE8D              FSQR     equ $EE8D          ; compute square root of FAC
EE97              FEXP     equ $EE97          ; raise ARG to power of FAC
EFEA              FCOS     equ $EFEA          ; compute cosine of FAC in rad
EFF1              FSIN     equ $EFF1          ; compute sine of FAC in rad
0800              ;
F03A              FTAN     equ $F03A          ; calc tangent of FAC in rad
F09E              FATAN    equ $F09E          ; compute arctangent of FAC
0800              ;
FD8E              CROUT    equ $FD8E
FDDA              PRBYTE   equ $FDDA
FDE3              PRHEX    equ $FDE3
FDED              COUT     equ $FDED
0800              ;
0800              ;
B000                       org $B000
B000                       obj $900
B000                       usr
B000              ;
B000              ;
B000              ; Initialization Table INITBL and Jump Table JMPTBL for all
B000              ; Input Commands.
B000              ;
B000 4C 00 B1     INITBL   jmp LOADER
B003 4C 37 B1              jmp LIBENT
B006 4C 7A B1              jmp LINKER
B009              ;
B009 4C 4D B2     JMPTBL   jmp LGNOT
B00C 4C 5D B2              jmp LGAND
B00F 4C 61 B2              jmp LGORA
B012 4C 65 B2              jmp LGEOR
B015              ;
B015 4C 7D B2              jmp CXADD
B018 4C A7 B2              jmp CXSUB
B01B 4C D1 B2              jmp CXMUL
B01E 4C 33 B3              jmp CXDIV
B021              ;
B021 4C F0 B3              jmp CX2PL
B024 4C 6D B4              jmp PL2CX
B027              ;
B027 4C 9D B4              jmp AYADD
B02A 4C B2 B4              jmp AYSUB
B02D 4C C7 B4              jmp AYMUL
B030 4C 28 B5              jmp AYINV
B033              ;
B033 4C 0D B6              jmp CSMUL
B036              ;
B036              ;
B036                       dfs $40-*&NEGONE,ZERO
B040              ;
B040              ;
B040              ; Hash Value Table HASHTBL for all Input Commands.
B040              ;
B040 36           HASHTBL  byt $36            ; LGNOT
B041 EF                    byt $EF            ; LGAND
B042 2D                    byt $2D            ; LGORA
B043 0F                    byt $0F            ; LGEOR
```

```
B044          ;
B044 D3                byt $D3              ; CXADD
B045 3C                byt $3C              ; CXSUB
B046 2E                byt $2E              ; CXMUL
B047 FB                byt $FB              ; CXDIV
B048          ;
B048 B7                byt $B7              ; CX2PL
B049 19                byt $19              ; PL2CX
B04A          ;
B04A BB                byt $BB              ; AYADD
B04B 24                byt $24              ; AYSUB
B04C 15                byt $15              ; AYMUL
B04D 01                byt $01              ; AYINV
B04E          ;
B04E 05                byt $05              ; CSMUL
B04F          ;
000F          HASHSIZ  equ *-HASHTBL
B04F          ;
B04F          ;
B04F          ; Pointer Length Table PTRLNTBL for all Input Commands.
B04F          ;
B04F 04       PTRLNTBL byt VARS2            ; LGNOT
B050 06                byt VARS3            ; LGAND
B051 06                byt VARS3            ; LGORA
B052 06                byt VARS3            ; LGEOR
B053          ;
B053 06                byt VARS3            ; CXADD
B054 06                byt VARS3            ; CXSUB
B055 06                byt VARS3            ; CXMUL
B056 06                byt VARS3            ; CXDIV
B057          ;
B057 04                byt VARS2            ; CX2PL
B058 04                byt VARS2            ; PL2CX
B059          ;
B059 06                byt VARS3            ; AYADD
B05A 06                byt VARS3            ; AYSUB
B05B 06                byt VARS3            ; AYMUL
B05C 06                byt VARS3            ; AYINV
B05D          ;
B05D 06                byt VARS3            ; CSMUL
B05E          ;
B05E          ;
B05E                   dfs $70-*&NEGONE,ZERO
B070          ;
B070          ;
B070          ; Working variables.
B070          ;
B070          FIRSTIME dfs 1,ZERO
B071          HASHVAL  dfs 1,ZERO
B072          CMDNDX   dfs 1,ZERO
B073          PTRNUM   dfs 1,ZERO
B074          PTRNDX   dfs 1,ZERO
B075          PTRLEN   dfs 1,ZERO
B076          SGNFLG1  dfs 1,ZERO
B077          SGNFLG2  dfs 1,ZERO
B078          ;
B078          ;
B078          ; Real and Imaginary coefficient addresses.
B078          ;
B078          FPREAL:
B078          REAL1L   dfs 1,ZERO
```

```
B079          REAL1H   dfs 1,ZERO
B07A          REAL2L   dfs 1,ZERO
B07B          REAL2H   dfs 1,ZERO
B07C          REAL3L   dfs 1,ZERO
B07D          REAL3H   dfs 1,ZERO
B07E          ;
B07E          FPIMAG:
B07E          IMAG1L   dfs 1,ZERO
B07F          IMAG1H   dfs 1,ZERO
B080          IMAG2L   dfs 1,ZERO
B081          IMAG2H   dfs 1,ZERO
B082          IMAG3L   dfs 1,ZERO
B083          IMAG3H   dfs 1,ZERO
B084          ;
B084          ;
B084          ; Temporary locations for intermediate results.
B084          ;
B084          TEMP1    dfs FPLEN,ZERO
B089          TEMP2    dfs FPLEN,ZERO
B08E          ;
B08E          TEMPR1   dfs FPLEN,ZERO
B093          TEMPR2   dfs FPLEN,ZERO
B098          TEMPI1   dfs FPLEN,ZERO
B09D          TEMPI2   dfs FPLEN,ZERO
B0A2          ;
B0A2          ;
B0A2          ; Offsets that are used by AYADD and AYSUB in BLDARGS1.
B0A2          ;
B0A2 00 14 0A ADDSUB   byt ZERO,CXFPOFF*2,CXFPOFF,CXFPOFF*3 ; X and Y
B0A5 1E
B0A6          ;
B0A6          ;
B0A6          ; Offsets and temporary memory that is used by AYMUL and
B0A6          ; BLDARGS2.
B0A6          ;
B0A6 00 14 00 MULOFF1  byt ZERO,CXFPOFF*2,ZERO,CXFPOFF*2 ; X
B0A9 14
B0AA 0A 1E 0A          byt CXFPOFF,CXFPOFF*3,CXFPOFF,CXFPOFF*3
B0AD 1E
B0AE 00 0A 14 MULOFF2  byt ZERO,CXFPOFF,CXFPOFF*2,CXFPOFF*3 ; Y
B0B1 1E
B0B2 00 0A 14          byt ZERO,CXFPOFF,CXFPOFF*2,CXFPOFF*3
B0B5 1E
B0B6          ;
B0B6 8E 93 8E MULLSB1  byt TEMPR1,TEMPR2,TEMPR1,TEMPR2 ; real values
B0B9 93
B0BA 8E 93 8E          byt TEMPR1,TEMPR2,TEMPR1,TEMPR2
B0BD 93
B0BE 98 9D 98 MULLSB2  byt TEMPI1,TEMPI2,TEMPI1,TEMPI2 ; imaginary values
B0C1 9D
B0C2 98 9D 98          byt TEMPI1,TEMPI2,TEMPI1,TEMPI2
B0C5 9D
B0C6          ;
B0C6 00 14 0A MULOUT   byt ZERO,CXFPOFF*2,CXFPOFF,CXFPOFF*3 ; Z
B0C9 1E
B0CA          ;
B0CA          ;
B0CA          ; Offsets and temporary memory that is used by AYINV and
B0CA          ; BLDARGS3 for computing a determinant.
B0CA          ;
B0CA 00 0A    DETOFF1  byt ZERO,CXFPOFF    ; a and c
```

```
B0CC 1E 14      DETOFF2  byt CXFPOFF*3,CXFPOFF*2 ; d and b
B0CE            ;
B0CE 8E 93      DETLSB1  byt TEMPR1,TEMPR2   ; real values
B0D0 98 9D      DETLSB2  byt TEMPI1,TEMPI2   ; imaginary values
B0D2            ;
B0D2            ;
B0D2            ; Offsets and temporary memory that is used by AYINV and
B0D2            ; BLDARGS4 for computing an inverse matrix.
B0D2            ;
B0D2 1E 14 0A   INVOFF1  byt CXFPOFF*3,CXFPOFF*2,CXFPOFF,ZERO ; X
B0D5 00
B0D6 00 14 0A   INVOFF2  byt ZERO,CXFPOFF*2,CXFPOFF,CXFPOFF*3 ; Z
B0D9 1E
B0DA            ;
B0DA 00 E2 E2   INVLSB   byt ZERO,MINUSONE,MINUSONE,ZERO ; minus values
B0DD 00
B0DE            ;
B0DE            ;
B0DE            ; Offsets that are used by CSMUL and BLDARGS5 for
B0DE            ; multiplying a complex array by a complex scalar.
B0DE            ;
B0DE 00 0A 14   CSOFF    byt ZERO,CXFPOFF,CXFPOFF*2,CXFPOFF*3 ; X and Z
B0E1 1E
B0E2            ;
B0E2            ;
B0E2            ; Floating point values that are needed for computations.
B0E2            ;
B0E2 81 80 00   MINUSONE hex 8180000000
B0E5 00 00
B0E7 E0 7F FF   HIGHVAL  hex E07FFFFFFF
B0EA FF FF
B0EC 81 49 0F   PIDIV2   hex 81490FDAA2
B0EF DA A2
B0F1 82 49 0F   PI       hex 82490FDAA2
B0F4 DA A2
B0F6            ;
B0F6            ;
B0F6                     dfs $FE-*&NEGONE,ZERO
B0FE            ;
B0FE            ;
B0FE 45 06      VERSION  byt $45,$06         ; Version #4.5.06
B100            ;
B100            ;
B100            ; Parse the Applesoft command line for the CXFPMATH
B100            ; routine name and obtain the memory addresses for two
B100            ; input complex variables and for one output complex
B100            ; variable.  Enter the selected CXFPMATH routine and
B100            ; process the data.  FIRSTIME initializes FRETOP in order
B100            ; to protect CXMATH.
B100            ;
B100 AD 70 B0   LOADER   lda FIRSTIME
B103 D0 09               bne >1
B105            ;
B105 CE 70 B0            dec FIRSTIME
B108            ;
B108 85 6F               sta FRETOP
B10A            ;
B10A A9 B0               lda /INITBL
B10C 85 70               sta FRETOP+1
B10E            ;
B10E 20 37 B1   ^1       jsr LIBENT
```

```
B111             ;
B111 AE 72 B0              ldx CMDNDX
B114             ;
B114 BD 4F B0              lda PTRLNTBL,X
B117 8D 75 B0              sta PTRLEN
B11A             ;
B11A 20 7A B1              jsr LINKER
B11D             ;
B11D AD 72 B0              lda CMDNDX
B120 0A                    asl
B121             ;
B121 6D 72 B0              adc CMDNDX
B124 AA                    tax
B125             ;
B125 BD 0A B0              lda JMPTBL+1,X
B128 8D 35 B1              sta TEMPMOD+1
B12B             ;
B12B BD 0B B0              lda JMPTBL+2,X
B12E 8D 36 B1              sta TEMPMOD+2
B131             ;
B131 2C 82 C0              bit ROM2WP
B134             ;
B134 4C 00 00    TEMPMOD   jmp *-*
B137             ;
B137             ;
B137             ; Parse the Applesoft command line for the CXFPMATH
B137             ; routine name and create a hash value from that name.
B137             ; Match the hash value in HASHTBL and save its index value.
B137             ;
B137             ; Terminate processing if there are no variables, if the
B137             ; routine name length is wrong, or if the hash value is not
B137             ; found in HASHTBL.
B137             ;
B137 A2 00       LIBENT    ldx #ZERO
B139 8E 71 B0              stx HASHVAL
B13C             ;
B13C 20 B1 00    ^1        jsr CHRGET
B13F F0 2E                 beq >5
B141             ;
B141 C9 2C                 cmp #COMMA&MSBCLR
B143 F0 13                 beq >2
B145             ;
B145 C9 22                 cmp #QUOTE&MSBCLR
B147 F0 F3                 beq <1
B149             ;
B149 18                    clc
B14A             ;
B14A E9 30                 sbc #"0"&MSBCLR
B14C             ;
B14C 0E 71 B0              asl HASHVAL
B14F 6D 71 B0              adc HASHVAL
B152 8D 71 B0              sta HASHVAL
B155             ;
B155 E8                    inx
B156 D0 E4                 bne <1                 ; always taken
B158             ;
B158 E0 05       ^2        cpx #CMDLEN
B15A D0 16                 bne >6
B15C             ;
B15C A2 0E                 ldx #HASHSIZ-1
B15E             ;
```

```
B15E AD 71 B0                 lda HASHVAL
B161             ;
B161 DD 40 B0    ^3           cmp HASHTBL,X
B164 F0 05                    beq >4
B166             ;
B166 CA                       dex
B167 10 F8                    bpl <3
B169             ;
B169 30 0A                    bmi >7
B16B             ;
B16B 8E 72 B0    ^4           stx CMDNDX
B16E             ;
B16E 60                       rts
B16F             ;
B16F A0 00       ^5           ldy #ERR1-ERRS
B171             ;
B171 2C 00 00                 bit *-*
B174                          dfs !-2
B172             ;
B172 A0 26       ^6           ldy #ERR2-ERRS
B174             ;
B174 2C 00 00                 bit *-*
B177                          dfs !-2
B175             ;
B175 A0 3F       ^7           ldy #ERR3-ERRS
B177             ;
B177 4C C5 B1                 jmp PRNTERR
B17A             ;
B17A             ;
B17A             ; Parse the Applesoft command line after the CXFPMATH
B17A             ; routine name to obtain the memory addresses for two input
B17A             ; complex variables and for one output complex variable.
B17A             ;
B17A             ; Terminate processing if the number of complex variables
B17A             ; is wrong.
B17A             ;
B17A A2 00       LINKER       ldx #ZERO
B17C             ;
B17C 8E 74 B0    ^1           stx PTRNDX
B17F             ;
B17F 20 B7 00                 jsr CHRGOT
B182 F0 36                    beq >4
B184             ;
B184 20 B1 00                 jsr CHRGET
B187 F0 31                    beq >4
B189             ;
B189 C9 2C                    cmp #COMMA&MSBCLR
B18B F0 29                    beq >3
B18D             ;
B18D 20 E3 DF                 jsr PTRGET
B190             ;
B190 AE 74 B0                 ldx PTRNDX
B193 EC 75 B0                 cpx PTRLEN
B196 B0 1E                    bcs >3
B198             ;
B198 95 08                    sta REALPTRS+0,X
B19A 94 09                    sty REALPTRS+1,X
B19C             ;
B19C 9D 78 B0                 sta FPREAL+0,X
B19F 48                       pha
B1A0             ;
```

```
B1A0 98                   tya
B1A1 9D 79 B0             sta FPREAL+1,X
B1A4               ;
B1A4 18                   clc
B1A5               ;
B1A5 68                   pla
B1A6 69 05                adc #IMAGOFF
B1A8 90 01                bcc >2
B1AA               ;
B1AA C8                   iny
B1AB               ;
B1AB 95 18         ^2     sta IMAGPTRS+0,X
B1AD 94 19                sty IMAGPTRS+1,X
B1AF               ;
B1AF 9D 7E B0             sta FPIMAG+0,X
B1B2               ;
B1B2 98                   tya
B1B3 9D 7F B0             sta FPIMAG+1,X
B1B6               ;
B1B6 E8            ^3     inx
B1B7 E8                   inx
B1B8               ;
B1B8 D0 C2                bne <1                ; always taken
B1BA               ;
B1BA AE 74 B0      ^4     ldx PTRNDX
B1BD EC 75 B0             cpx PTRLEN
B1C0 D0 01                bne >5
B1C2               ;
B1C2 60                   rts
B1C3               ;
B1C3 A0 5D         ^5     ldy #ERR4-ERRS        ; fall into PRNTERR
B1C5               ;
B1C5               ;
B1C5               ; Print the Error Message in ERRS given by Y-reg.
B1C5               ;
B1C5 B9 D3 B1      PRNTERR lda ERRS,Y
B1C8 F0 06                beq >1
B1CA               ;
B1CA 20 ED FD             jsr COUT
B1CD               ;
B1CD C8                   iny
B1CE D0 F5                bne PRNTERR
B1D0               ;
B1D0 4C D0 03      ^1     jmp DOSWARM
B1D3               ;
B1D3               ;
B1D3               ; Error messages.
B1D3               ;
B1D3               ERRS:
B1D3               ;
B1D3 8D            ERR1   byt RETURN
B1D4 CE EF A0             asc "No variables included with command."
B1D7 F6 E1 F2
B1DA E9 E1 E2
B1DD EC E5 F3
B1E0 A0 E9 EE
B1E3 E3 EC F5
B1E6 E4 E5 E4
B1E9 A0 F7 E9
B1EC F4 E8 A0
B1EF E3 EF ED
```

```
B1F2 ED E1 EE
B1F5 E4 AE
B1F7 8D 00      461              byt RETURN,ZERO
B1F9            462  ;
B1F9 8D         463  ERR2        byt RETURN
B1FA C3 EF ED   464              asc "Command size is wrong."
B1FD ED E1 EE
B200 E4 A0 F3
B203 E9 FA E5
B206 A0 E9 F3
B209 A0 F7 F2
B20C EF EE E7
B20F AE
B210 8D 00      465              byt RETURN,ZERO
B212            466  ;
B212 8D         467  ERR3        byt RETURN
B213 C3 EF ED   468              asc "Command not found in table."
B216 ED E1 EE
B219 E4 A0 EE
B21C EF F4 A0
B21F E6 EF F5
B222 EE E4 A0
B225 E9 EE A0
B228 F4 E1 E2
B22B EC E5 AE
B22E 8D 00      469              byt RETURN,ZERO
B230            470  ;
B230 8D         471  ERR4        byt RETURN
B231 D7 F2 EF   472              asc "Wrong number of arguments."
B234 EE E7 A0
B237 EE F5 ED
B23A E2 E5 F2
B23D A0 EF E6
B240 A0 E1 F2
B243 E7 F5 ED
B246 E5 EE F4
B249 F3 AE
B24B 8D 00      473              byt RETURN,ZERO
B24D            474  ;
B24D            475  ;
B24D            476              icl "CXFPMATH.1.L"

LLOAD CXFPMATH.1.L,A$4000
```

```
B24D                       ttl "CXFPMATH.1 Routines, CXFPMATH.1.L"
B24D                 ;
B24D                 ;
B24D                 ; CXFPMATH.1.L
B24D                 ;
B24D                 ;
B24D                 ; Logical operation routines.  These routines operate on
B24D                 ; integers only and they are written as:
B24D                 ;
B24D                 ; CALL "LGNOT",X%,Z%                 where X% NOT form Z%
B24D                 ;
B24D                 ; CALL "LGAND",X%,Y%,Z%              where X% AND Y% form Z%
B24D                 ; CALL "LGORA",X%,Y%,Z%              where X% ORA Y% form Z%
B24D                 ; CALL "LGEOR",X%,Y%,Z%              where X% EOR Y% form Z%
B24D                 ;
B24D                 ;
B24D                 ; Calculate the NOT logical operator.
B24D                 ;
B24D A0 00           LGNOT     ldy #ZERO
B24F                 ;
B24F B1 08                     lda (REALPTRS+0),Y  ; get MSB of X
B251 49 FF                     eor #NEGONE         ; operate on MSB of X
B253 91 0A                     sta (REALPTRS+2),Y  ; save result to MSB of Z
B255                 ;
B255 C8                        iny
B256                 ;
B256 B1 08                     lda (REALPTRS+0),Y  ; get LSB of X
B258 49 FF                     eor #NEGONE         ; operate on LSB of X
B25A 91 0A                     sta (REALPTRS+2),Y  ; save result to LSB of Z
B25C                 ;
B25C 60                        rts
B25D                 ;
B25D                 ;
B25D                 ; Get the logical operator AND and branch to LGCALC.
B25D                 ;
B25D A9 00           LGAND     lda #ZERO
B25F                           dfs !-1
B25E                 ;
B25E 31 08                     and (REALPTRS),Y
B260                           dfs !-1
B25F                 ;
B25F D0 06                     bne LGCALC          ; always taken
B261                 ;
B261                 ;
B261                 ; Get the logical operator ORA and branch to LGCALC.
B261                 ;
B261 A9 00           LGORA     lda #ZERO
B263                           dfs !-1
B262                 ;
B262 11 08                     ora (REALPTRS),Y
B264                           dfs !-1
B263                 ;
B263 D0 02                     bne LGCALC          ; always taken
B265                 ;
B265                 ;
B265                 ; Get the logical operator EOR and fall into LGCALC.
B265                 ;
B265 A9 00           LGEOR     lda #ZERO
B267                           dfs !-1
B266                 ;
```

```
B266 51 08                eor (REALPTRS),Y
B268                      dfs !-1
B267            ;
B267            ;
B267 8D 71 B2   LGCALC    sta LGMOD1
B26A 8D 78 B2             sta LGMOD2
B26D            ;
B26D A0 00                ldy #ZERO
B26F            ;
B26F            ;
B26F            ; The values in Integer Arrays are in MSB,LSB order.
B26F            ;
B26F B1 08                lda (REALPTRS+0),Y  ; get MSB of X
B271 31 0A      LGMOD1    and (REALPTRS+2),Y  ; operate on MSB of Y
B273 91 0C                sta (REALPTRS+4),Y  ; save result to MSB of Z
B275            ;
B275 C8                   iny
B276            ;
B276 B1 08                lda (REALPTRS+0),Y  ; get LSB of X
B278 31 0A      LGMOD2    and (REALPTRS+2),Y  ; operate on LSB of Y
B27A 91 0C                sta (REALPTRS+4),Y  ; save result to LSB of Z
B27C            ;
B27C 60                   rts
B27D            ;
B27D            ;
B27D                      icl "CXFPMATH.2.L"
```

LLOAD CXFPMATH.2.L,A$4000

```
B27D                        ttl "CXFPMATH.2 Routines, CXFPMATH.2.L"
B27D               ;
B27D               ;
B27D               ; CXFPMATH.2.L
B27D               ;
B27D               ;
B27D               ; Complex CXADD routine.  This routine adds two complex
B27D               ; floating point numbers and is written as:
B27D               ;
B27D               ; CALL "CXADD",X(0),Y(0),Z(0)            where X + Y form Z
B27D               ;
B27D               ; The real coefficients are added separately from the
B27D               ; imaginary coefficients.  The imaginary coefficients
B27D               ; follow the real coefficients in memory.  All floating
B27D               ; point numbers are five bytes in size.
B27D               ;
B27D               ; Put Y into FAC, add in X, and save to Z.
B27D               ;
B27D A5 0A         CXADD    lda REALPTRS+2
B27F A4 0B                  ldy REALPTRS+3
B281               ;
B281 20 F9 EA               jsr LOADFAC
B284               ;
B284 A5 08                  lda REALPTRS+0
B286 A4 09                  ldy REALPTRS+1
B288               ;
B288 20 BE E7               jsr FADD
B28B               ;
B28B A6 0C                  ldx REALPTRS+4
B28D A4 0D                  ldy REALPTRS+5
B28F               ;
B28F 20 2B EB               jsr COPYFAC
B292               ;
B292               ;
B292               ; Add the imaginary coefficients.
B292               ;
B292 A5 1A                  lda IMAGPTRS+2
B294 A4 1B                  ldy IMAGPTRS+3
B296               ;
B296 20 F9 EA               jsr LOADFAC
B299               ;
B299 A5 18                  lda IMAGPTRS+0
B29B A4 19                  ldy IMAGPTRS+1
B29D               ;
B29D 20 BE E7               jsr FADD
B2A0               ;
B2A0 A6 1C                  ldx IMAGPTRS+4
B2A2 A4 1D                  ldy IMAGPTRS+5
B2A4               ;
B2A4 4C 2B EB               jmp COPYFAC
B2A7               ;
B2A7               ;
B2A7               ; Complex CXSUB routine.  This routine subtracts two
B2A7               ; complex floating point numbers and is written as:
B2A7               ;
B2A7               ; CALL "CXSUB",X(0),Y(0),Z(0)            where X - Y form Z
B2A7               ;
B2A7               ; The real coefficients are subtracted separately from the
B2A7               ; imaginary coefficients.  The imaginary coefficients
B2A7               ; follow the real coefficients in memory.  All floating
```

```
B2A7          ; point numbers are five bytes in size.
B2A7          ;
B2A7          ; Put Y into FAC, negate FAC, add in X, and save to Z.
B2A7          ;
B2A7 A5 0A    CXSUB    lda REALPTRS+2
B2A9 A4 0B             ldy REALPTRS+3
B2AB          ;
B2AB 20 F9 EA          jsr LOADFAC
B2AE          ;
B2AE A5 08             lda REALPTRS+0
B2B0 A4 09             ldy REALPTRS+1
B2B2          ;
B2B2 20 A7 E7          jsr FSUB
B2B5          ;
B2B5 A6 0C             ldx REALPTRS+4
B2B7 A4 0D             ldy REALPTRS+5
B2B9          ;
B2B9 20 2B EB          jsr COPYFAC
B2BC          ;
B2BC          ;
B2BC          ; Subtract the imaginary coefficients.
B2BC          ;
B2BC A5 1A             lda IMAGPTRS+2
B2BE A4 1B             ldy IMAGPTRS+3
B2C0          ;
B2C0 20 F9 EA          jsr LOADFAC
B2C3          ;
B2C3 A5 18             lda IMAGPTRS+0
B2C5 A4 19             ldy IMAGPTRS+1
B2C7          ;
B2C7 20 A7 E7          jsr FSUB
B2CA          ;
B2CA A6 1C             ldx IMAGPTRS+4
B2CC A4 1D             ldy IMAGPTRS+5
B2CE          ;
B2CE 4C 2B EB          jmp COPYFAC
B2D1          ;
B2D1          ;
B2D1          ; Complex CXMUL routine.  This routine multiplies two
B2D1          ; complex floating point numbers and is written as:
B2D1          ;
B2D1          ; CALL "CXMUL",X(0),Y(0),Z(0)            where X * Y form Z
B2D1          ;
B2D1          ; The real and imaginary coefficients are combined and
B2D1          ; processed according to the following procedure:
B2D1          ;
B2D1          ; (a + bi) * (c + di) -> (ac - bd) + (ad + bc)i
B2D1          ;
B2D1          ; Put Y into FAC, multiply by X, and save to Z.
B2D1          ;
B2D1 A5 0A    CXMUL    lda REALPTRS+2
B2D3 A4 0B             ldy REALPTRS+3
B2D5          ;
B2D5 20 F9 EA          jsr LOADFAC
B2D8          ;
B2D8 A5 08             lda REALPTRS+0
B2DA A4 09             ldy REALPTRS+1
B2DC          ;
B2DC 20 7F E9          jsr FMULT
B2DF          ;
B2DF A2 84             ldx #TEMP1
```

```
B2E1 A0 B0          ldy /TEMP1
B2E3            ;
B2E3 20 2B EB       jsr COPYFAC
B2E6            ;
B2E6            ;
B2E6            ; Multiply imaginary coefficients b and d.
B2E6            ;
B2E6 A5 1A          lda IMAGPTRS+2
B2E8 A4 1B          ldy IMAGPTRS+3
B2EA            ;
B2EA 20 F9 EA       jsr LOADFAC
B2ED            ;
B2ED A5 18          lda IMAGPTRS+0
B2EF A4 19          ldy IMAGPTRS+1
B2F1            ;
B2F1 20 7F E9       jsr FMULT
B2F4            ;
B2F4            ;
B2F4            ; Combine real coefficients ac - bd.
B2F4            ;
B2F4 A9 84          lda #TEMP1
B2F6 A0 B0          ldy /TEMP1
B2F8            ;
B2F8 20 A7 E7       jsr FSUB
B2FB            ;
B2FB A6 0C          ldx REALPTRS+4
B2FD A4 0D          ldy REALPTRS+5
B2FF            ;
B2FF 20 2B EB       jsr COPYFAC
B302            ;
B302            ;
B302            ; Multiply coefficients b and c.
B302            ;
B302 A5 0A          lda REALPTRS+2
B304 A4 0B          ldy REALPTRS+3
B306            ;
B306 20 F9 EA       jsr LOADFAC
B309            ;
B309 A5 18          lda IMAGPTRS+0
B30B A4 19          ldy IMAGPTRS+1
B30D            ;
B30D 20 7F E9       jsr FMULT
B310            ;
B310 A2 84          ldx #TEMP1
B312 A0 B0          ldy /TEMP1
B314            ;
B314 20 2B EB       jsr COPYFAC
B317            ;
B317            ;
B317            ; Multiply coefficients a and d.
B317            ;
B317 A5 1A          lda IMAGPTRS+2
B319 A4 1B          ldy IMAGPTRS+3
B31B            ;
B31B 20 F9 EA       jsr LOADFAC
B31E            ;
B31E A5 08          lda REALPTRS+0
B320 A4 09          ldy REALPTRS+1
B322            ;
B322 20 7F E9       jsr FMULT
B325            ;
```

```
B325              ;
B325              ; Combine imaginary coefficients ad + bc.
B325              ;
B325 A9 84                  lda #TEMP1
B327 A0 B0                  ldy /TEMP1
B329              ;
B329 20 BE E7               jsr FADD
B32C              ;
B32C A6 1C                  ldx IMAGPTRS+4
B32E A4 1D                  ldy IMAGPTRS+5
B330              ;
B330 4C 2B EB               jmp COPYFAC
B333              ;
B333              ;
B333              ; Complex CXDIV routine.  This routine divides two complex
B333              ; floating point numbers and is written as:
B333              ;
B333              ; CALL "CXDIV",X(0),Y(0),Z(0)            where X / Y form Z
B333              ;
B333              ; The real and imaginary coefficients are combined and
B333              ; processed according to the following procedure:
B333              ;
B333              ; (a + bi) / (c + di) =
B333              ; (ac + bd) / (c*c + d*d) + (bc - ad)i / (c*c + d*d)
B333              ;
B333              ; Calculate the denominator (c*c + d*d).
B333              ;
B333 A5 1A        CXDIV     lda IMAGPTRS+2
B335 A4 1B                  ldy IMAGPTRS+3
B337              ;
B337 20 F9 EA               jsr LOADFAC
B33A              ;
B33A A5 1A                  lda IMAGPTRS+2
B33C A4 1B                  ldy IMAGPTRS+3
B33E              ;
B33E 20 7F E9               jsr FMULT
B341              ;
B341 A2 84                  ldx #TEMP1
B343 A0 B0                  ldy /TEMP1
B345              ;
B345 20 2B EB               jsr COPYFAC
B348              ;
B348 A5 0A                  lda REALPTRS+2
B34A A4 0B                  ldy REALPTRS+3
B34C              ;
B34C 20 F9 EA               jsr LOADFAC
B34F              ;
B34F A5 0A                  lda REALPTRS+2
B351 A4 0B                  ldy REALPTRS+3
B353              ;
B353 20 7F E9               jsr FMULT
B356              ;
B356 A9 84                  lda #TEMP1
B358 A0 B0                  ldy /TEMP1
B35A              ;
B35A 20 BE E7               jsr FADD
B35D              ;
B35D A2 84                  ldx #TEMP1
B35F A0 B0                  ldy /TEMP1
B361              ;
B361 20 2B EB               jsr COPYFAC
```

```
B364              ;
B364              ;
B364              ; Combine coefficients for real numerator (ac + bd) and
B364              ; divide by the denominator that is saved in TEMP1.
B364              ;
B364              ; Multiply real coefficients a and c.
B364              ;
B364 A5 0A                lda REALPTRS+2
B366 A4 0B                ldy REALPTRS+3
B368              ;
B368 20 F9 EA             jsr LOADFAC
B36B              ;
B36B A5 08                lda REALPTRS+0
B36D A4 09                ldy REALPTRS+1
B36F              ;
B36F 20 7F E9             jsr FMULT
B372              ;
B372 A2 89                ldx #TEMP2
B374 A0 B0                ldy /TEMP2
B376              ;
B376 20 2B EB             jsr COPYFAC
B379              ;
B379              ;
B379              ; Multiply imaginary coefficients b and d.
B379              ;
B379 A5 1A                lda IMAGPTRS+2
B37B A4 1B                ldy IMAGPTRS+3
B37D              ;
B37D 20 F9 EA             jsr LOADFAC
B380              ;
B380 A5 18                lda IMAGPTRS+0
B382 A4 19                ldy IMAGPTRS+1
B384              ;
B384 20 7F E9             jsr FMULT
B387              ;
B387              ;
B387              ; Combine numerator coefficients (ac + bd) and save.
B387              ;
B387 A9 89                lda #TEMP2
B389 A0 B0                ldy /TEMP2
B38B              ;
B38B 20 BE E7             jsr FADD
B38E              ;
B38E A2 89                ldx #TEMP2
B390 A0 B0                ldy /TEMP2
B392              ;
B392 20 2B EB             jsr COPYFAC
B395              ;
B395              ;
B395              ; Recall the denominator and divide it into (ac + bc).
B395              ;
B395 A9 84                lda #TEMP1
B397 A0 B0                ldy /TEMP1
B399              ;
B399 20 F9 EA             jsr LOADFAC
B39C              ;
B39C A9 89                lda #TEMP2
B39E A0 B0                ldy /TEMP2
B3A0              ;
B3A0 20 66 EA             jsr FDIV
B3A3              ;
```

```
B3A3 A6 0C                 ldx REALPTRS+4
B3A5 A4 0D                 ldy REALPTRS+5
B3A7                  ;
B3A7 20 2B EB              jsr COPYFAC
B3AA                  ;
B3AA                  ;
B3AA                  ; Combine coefficients for imaginary numerator (bc - ad)
B3AA                  ; and divide by the denominator that is saved in TEMP1.
B3AA                  ;
B3AA                  ; Multiply the coefficients b and c.
B3AA                  ;
B3AA A5 0A                 lda REALPTRS+2
B3AC A4 0B                 ldy REALPTRS+3
B3AE                  ;
B3AE 20 F9 EA              jsr LOADFAC
B3B1                  ;
B3B1 A5 18                 lda IMAGPTRS+0
B3B3 A4 19                 ldy IMAGPTRS+1
B3B5                  ;
B3B5 20 7F E9              jsr FMULT
B3B8                  ;
B3B8 A2 89                 ldx #TEMP2
B3BA A0 B0                 ldy /TEMP2
B3BC                  ;
B3BC 20 2B EB              jsr COPYFAC
B3BF                  ;
B3BF                  ;
B3BF                  ; Multiply the coefficients a and d.
B3BF                  ;
B3BF A5 1A                 lda IMAGPTRS+2
B3C1 A4 1B                 ldy IMAGPTRS+3
B3C3                  ;
B3C3 20 F9 EA              jsr LOADFAC
B3C6                  ;
B3C6 A5 08                 lda REALPTRS+0
B3C8 A4 09                 ldy REALPTRS+1
B3CA                  ;
B3CA 20 7F E9              jsr FMULT
B3CD                  ;
B3CD                  ;
B3CD                  ; Combine numerator coefficients (bc - ad) and save.
B3CD                  ;
B3CD A9 89                 lda #TEMP2
B3CF A0 B0                 ldy /TEMP2
B3D1                  ;
B3D1 20 A7 E7              jsr FSUB
B3D4                  ;
B3D4 A2 89                 ldx #TEMP2
B3D6 A0 B0                 ldy /TEMP2
B3D8                  ;
B3D8 20 2B EB              jsr COPYFAC
B3DB                  ;
B3DB                  ;
B3DB                  ; Recall the denominator and divide it into (bc - ad).
B3DB                  ;
B3DB A9 84                 lda #TEMP1
B3DD A0 B0                 ldy /TEMP1
B3DF                  ;
B3DF 20 F9 EA              jsr LOADFAC
B3E2                  ;
B3E2 A9 89                 lda #TEMP2
```

```
B3E4 A0 B0               ldy /TEMP2
B3E6            ;
B3E6 20 66 EA            jsr FDIV
B3E9            ;
B3E9 A6 1C               ldx IMAGPTRS+4
B3EB A4 1D               ldy IMAGPTRS+5
B3ED            ;
B3ED 4C 2B EB            jmp COPYFAC
B3F0            ;
B3F0            ;
B3F0                     icl "CXFPMATH.3.L"
```

LLOAD CXFPMATH.3.L,A$4000

```
B3F0                     ttl "CXFPMATH.3 Routines, CXFPMATH.3.L"
B3F0            ;
B3F0            ;
B3F0            ; CXFPMATH.3.L
B3F0            ;
B3F0            ;
B3F0            ; Complex CX2PL routine.  This routine converts a complex
B3F0            ; number into its polar coordinates and is written as:
B3F0            ;
B3F0            ; CALL "CX2PL",X(0),Z(0)              where X CX2PL forms Z
B3F0            ;
B3F0            ; The real and imaginary coefficients are processed
B3F0            ; according to the following procedure:
B3F0            ;
B3F0            ; magnitude = root(real*real + imag*imag)
B3F0            ;
B3F0            ; phase = atn(imag / real), when real > 0
B3F0            ;       = (pi/2) * sgn(imag), when real = 0
B3F0            ;       = atn(imag / real) + pi, when real < 0
B3F0            ;
B3F0            ; Compute the sign flag of the real number as 0, <0, or >0.
B3F0            ;
B3F0 A0 00      CX2PL    ldy #ZERO
B3F2            ;
B3F2 B1 08               lda (REALPTRS),Y
B3F4 29 7F               and #MSBCLR
B3F6            ;
B3F6 C8                  iny
B3F7            ;
B3F7 11 08               ora (REALPTRS),Y
B3F9 8D 76 B0            sta SGNFLG1
B3FC            ;
B3FC            ;
B3FC            ; Calculate the magnitude of the complex number.
B3FC            ;
B3FC A5 18               lda IMAGPTRS+0
B3FE A4 19               ldy IMAGPTRS+1
B400            ;
B400 20 F9 EA            jsr LOADFAC
B403            ;
B403 A5 18               lda IMAGPTRS+0
B405 A4 19               ldy IMAGPTRS+1
B407            ;
B407 20 7F E9            jsr FMULT
B40A            ;
B40A A2 84               ldx #TEMP1
B40C A0 B0               ldy /TEMP1
B40E            ;
B40E 20 2B EB            jsr COPYFAC
B411            ;
B411 A5 08               lda REALPTRS+0
B413 A4 09               ldy REALPTRS+1
B415            ;
B415 20 F9 EA            jsr LOADFAC
B418            ;
B418 A5 08               lda REALPTRS+0
B41A A4 09               ldy REALPTRS+1
B41C            ;
B41C 20 7F E9            jsr FMULT
B41F            ;
```

```
B41F A9 84              lda #TEMP1
B421 A0 B0              ldy /TEMP1
B423            ;
B423 20 BE E7           jsr FADD
B426 20 8D EE           jsr FSQR
B429            ;
B429 A6 0A              ldx REALPTRS+2
B42B A4 0B              ldy REALPTRS+3
B42D            ;
B42D 20 2B EB           jsr COPYFAC
B430            ;
B430            ;
B430            ; Calculate the phase of the complex number when the
B430            ; magnitude of the complex number is zero.
B430            ;
B430 AD 76 B0           lda SGNFLG1
B433 D0 14              bne >1
B435            ;
B435 A5 18              lda IMAGPTRS+0
B437 A4 19              ldy IMAGPTRS+1
B439            ;
B439 20 F9 EA           jsr LOADFAC
B43C 20 90 EB           jsr FSGN
B43F            ;
B43F A9 EC              lda #PIDIV2
B441 A0 B0              ldy /PIDIV2
B443            ;
B443 20 7F E9           jsr FMULT
B446            ;
B446 4C 66 B4           jmp C2PEXIT
B449            ;
B449            ;
B449            ; Calculate the phase of the complex number when the
B449            ; magnitude of the complex number is not zero.
B449            ;
B449 A5 08      ^1      lda REALPTRS+0
B44B A4 09              ldy REALPTRS+1
B44D            ;
B44D 20 F9 EA           jsr LOADFAC
B450            ;
B450 A5 18              lda IMAGPTRS+0
B452 A4 19              ldy IMAGPTRS+1
B454            ;
B454 20 66 EA           jsr FDIV
B457 20 9E F0           jsr FATAN
B45A            ;
B45A            ;
B45A            ; Add the value PI if SGNFLG1 < 0.
B45A            ;
B45A 2C 76 B0           bit SGNFLG1
B45D 10 07              bpl C2PEXIT
B45F            ;
B45F A9 F1              lda #PI
B461 A0 B0              ldy /PI
B463            ;
B463 20 BE E7           jsr FADD
B466            ;
B466            ;
B466            ; Save FAC to the imaginary coefficient.
B466            ;
B466 A6 1A      C2PEXIT ldx IMAGPTRS+2
```

```
B468 A4 1B              ldy IMAGPTRS+3
B46A               ;
B46A 4C 2B EB           jmp COPYFAC
B46D               ;
B46D               ;
B46D               ; Complex PL2CX routine.  This routine converts polar
B46D               ; coordinates into a complex number and is written as:
B46D               ;
B46D               ; CALL "PL2CX",X(0),Z(0)            where X PL2CX forms Z
B46D               ;
B46D               ; The magnitude at REALPTRS and phase at IMAGPTRS values
B46D               ; are processed according to the following procedure:
B46D               ;
B46D               ; real number = magnitude * cos(phase)
B46D               ; imag number = magnitude * sin(phase)
B46D               ;
B46D               ; Compute the real coefficient.
B46D               ;
B46D A5 18         PL2CX    lda IMAGPTRS+0
B46F A4 19              ldy IMAGPTRS+1
B471               ;
B471 20 F9 EA           jsr LOADFAC
B474               ;
B474 20 EA EF           jsr FCOS
B477               ;
B477 A5 08              lda REALPTRS+0
B479 A4 09              ldy REALPTRS+1
B47B               ;
B47B 20 7F E9           jsr FMULT
B47E               ;
B47E A6 0A              ldx REALPTRS+2
B480 A4 0B              ldy REALPTRS+3
B482               ;
B482 20 2B EB           jsr COPYFAC
B485               ;
B485               ;
B485               ; Compute the imaginary coefficient.
B485               ;
B485 A5 18              lda IMAGPTRS+0
B487 A4 19              ldy IMAGPTRS+1
B489               ;
B489 20 F9 EA           jsr LOADFAC
B48C               ;
B48C 20 F1 EF           jsr FSIN
B48F               ;
B48F A5 08              lda REALPTRS+0
B491 A4 09              ldy REALPTRS+1
B493               ;
B493 20 7F E9           jsr FMULT
B496               ;
B496 A6 1A              ldx IMAGPTRS+2
B498 A4 1B              ldy IMAGPTRS+3
B49A               ;
B49A 4C 2B EB           jmp COPYFAC
B49D               ;
B49D               ;
B49D                    icl "CXFPMATH.4.L"
```

LLOAD CXFPMATH.4.L,A$4000

```
B49D                        ttl "CXFPMATH.4 Routines, CXFPMATH.4.L"
B49D              ;
B49D              ;
B49D              ; CXFPMATH.4.L
B49D              ;
B49D              ;
B49D              ; Complex AYADD routine.  This routine adds two complex
B49D              ; 2x2 floating point arrays and is written as:
B49D              ;
B49D              ; CALL "AYADD",X(0,0,0),Y(0,0,0),Z(0,0,0)      X + Y form Z
B49D              ;
B49D              ; The real coefficients are added separately from the
B49D              ; imaginary coefficients.  The four imaginary coefficients
B49D              ; follow the four real coefficients for each array in
B49D              ; memory.  All complex floating point array numbers are
B49D              ; five bytes in size.
B49D              ;
B49D              ; Put Y into FAC, add in X using CXADD, and save to Z.
B49D              ;
B49D A9 00        AYADD     lda #ZERO          ; initialize offset counter
B49F 85 1E                  sta COUNT
B4A1 F0 03                  beq >2             ; always taken
B4A3              ;
B4A3 20 22 B6     ^1        jsr BLDARGS1
B4A6              ;
B4A6 20 7D B2     ^2        jsr CXADD
B4A9              ;
B4A9 E6 1E                  inc COUNT
B4AB              ;
B4AB A5 1E                  lda COUNT
B4AD C9 04                  cmp #ARRAYSIZ
B4AF D0 F2                  bne <1
B4B1              ;
B4B1 60                     rts
B4B2              ;
B4B2              ;
B4B2              ; Complex AYSUB routine.  This routine subtracts two
B4B2              ; complex 2x2 floating point arrays and is written as:
B4B2              ;
B4B2              ; CALL "AYSUB",X(0,0,0),Y(0,0,0),Z(0,0,0)      X - Y form Z
B4B2              ;
B4B2              ; The real coefficients are subtracted separately from the
B4B2              ; imaginary coefficients.  The four imaginary coefficients
B4B2              ; follow the four real coefficients for each array in
B4B2              ; memory.  All complex floating point array numbers are
B4B2              ; five bytes in size.
B4B2              ;
B4B2              ; Put Y into FAC, negate FAC, add in X using CXSUB, and
B4B2              ; save to Z.
B4B2              ;
B4B2 A9 00        AYSUB     lda #ZERO          ; initialize offset counter
B4B4 85 1E                  sta COUNT
B4B6 F0 03                  beq >2             ; always taken
B4B8              ;
B4B8 20 22 B6     ^1        jsr BLDARGS1
B4BB              ;
B4BB 20 A7 B2     ^2        jsr CXSUB
B4BE              ;
B4BE E6 1E                  inc COUNT
B4C0              ;
```

```
B4C0 A5 1E                  lda COUNT
B4C2 C9 04                  cmp #ARRAYSIZ
B4C4 D0 F2                  bne <1
B4C6               ;
B4C6 60                     rts
B4C7               ;
B4C7               ;
B4C7               ; Complex AYMUL routine.  This routine multiplies two
B4C7               ; complex 2x2 floating point arrays and is written as:
B4C7               ;
B4C7               ; CALL "AYMUL",X(0,0,0),Y(0,0,0),Z(0,0,0)      X * Y form Z
B4C7               ;
B4C7               ; The real and imaginary coefficients are multiplied by
B4C7               ; processing the dot product of the rows of array X and
B4C7               ; the columns of array Y.
B4C7               ;
B4C7               ; Put Y into FAC, multiply by X using CXMUL, and save to Z.
B4C7               ;
B4C7 A9 00         AYMUL    lda #ZERO            ; initialize offset counter
B4C9 85 1E                  sta COUNT
B4CB               ;
B4CB 20 52 B6      ^1       jsr BLDARGS2
B4CE 20 D1 B2               jsr CXMUL
B4D1               ;
B4D1 E6 1E                  inc COUNT
B4D3               ;
B4D3 20 52 B6               jsr BLDARGS2
B4D6 20 D1 B2               jsr CXMUL
B4D9               ;
B4D9               ;
B4D9               ; Add TEMPR1 and TEMPR2.
B4D9               ;
B4D9 A9 93                  lda #TEMPR2
B4DB A0 B0                  ldy /TEMPR2
B4DD               ;
B4DD 20 F9 EA               jsr LOADFAC
B4E0               ;
B4E0 A9 8E                  lda #TEMPR1
B4E2 A0 B0                  ldy /TEMPR1
B4E4               ;
B4E4 20 BE E7               jsr FADD
B4E7               ;
B4E7               ;
B4E7               ; Save the computed value.
B4E7               ;
B4E7 A5 1E                  lda COUNT
B4E9 4A                     lsr
B4EA AA                     tax
B4EB               ;
B4EB 18                     clc
B4EC               ;
B4EC AD 7C B0               lda REAL3L
B4EF AC 7D B0               ldy REAL3H
B4F2               ;
B4F2 7D C6 B0               adc MULOUT,X
B4F5 90 01                  bcc >2
B4F7               ;
B4F7 C8                     iny
B4F8               ;
B4F8 AA            ^2       tax
B4F9               ;
```

```
B4F9 20 2B EB          jsr COPYFAC
B4FC              ;
B4FC              ;
B4FC              ; Add TEMPI1 and TEMPI2.
B4FC              ;
B4FC A9 9D             lda #TEMPI2
B4FE A0 B0             ldy /TEMPI2
B500              ;
B500 20 F9 EA          jsr LOADFAC
B503              ;
B503 A9 98             lda #TEMPI1
B505 A0 B0             ldy /TEMPI1
B507              ;
B507 20 BE E7          jsr FADD
B50A              ;
B50A              ;
B50A              ; Save the computed value.
B50A              ;
B50A A5 1E             lda COUNT
B50C 4A                lsr
B50D AA                tax
B50E              ;
B50E 18                clc
B50F              ;
B50F AD 82 B0          lda IMAG3L
B512 AC 83 B0          ldy IMAG3H
B515              ;
B515 7D C6 B0          adc MULOUT,X
B518 90 01             bcc >3
B51A              ;
B51A C8                iny
B51B              ;
B51B AA           ^3   tax
B51C              ;
B51C 20 2B EB          jsr COPYFAC
B51F              ;
B51F              ;
B51F              ; Multiply the next row and column.
B51F              ;
B51F E6 1E             inc COUNT
B521              ;
B521 A5 1E             lda COUNT
B523 C9 08             cmp #ARRAYSIZ*2
B525 D0 A4             bne <1
B527              ;
B527 60                rts
B528              ;
B528              ;
B528              ; Complex AYINV routine.  This routine computes the inverse
B528              ; of a complex 2x2 floating point array and is written as:
B528              ;
B528              ; CALL "AYINV",X(0,0,0),Y(0),Z(0,0,0)       X forms Y and Z
B528              ;
B528              ; The determinant of X is computed and saved to Y.  If the
B528              ; determinant is not zero, each element of X is divided
B528              ; by Y to form Z.  Otherwise, all elements of Z are set to
B528              ; 7.922816252E+28.
B528              ;
B528              ;         | a  b |                                 -1
B528              ; Let A = |      |, then detA = ad - bc, and A     * A = I
B528              ;         | c  d |
```

```
B528              ;
B528              ;  -1      1  | d -b |            | 1  0 |
B528              ; A   = ----  |      |  and I =   |      |
B528              ;       detA  |-c  a |            | 0  1 |
B528              ;
B528              ;
B528              ; Compute the determinant detA = ad - bc.
B528              ;
B528 A9 00        AYINV    lda #ZERO           ; initialize offset counter
B52A 85 1E                 sta COUNT
B52C              ;
B52C 20 AB B6              jsr BLDARGS3
B52F 20 D1 B2              jsr CXMUL
B532              ;
B532 E6 1E                 inc COUNT
B534              ;
B534 20 AB B6              jsr BLDARGS3
B537 20 D1 B2              jsr CXMUL
B53A              ;
B53A              ;
B53A              ; Subtract TEMP2 from TEMP1, compute SGNFLG1, and save
B53A              ; the real coefficient to the determinant Y.
B53A              ;
B53A A9 93                 lda #TEMPR2
B53C A0 B0                 ldy /TEMPR2
B53E              ;
B53E 20 F9 EA              jsr LOADFAC
B541              ;
B541 A9 8E                 lda #TEMPR1
B543 A0 B0                 ldy /TEMPR1
B545              ;
B545 20 A7 E7              jsr FSUB
B548              ;
B548 A5 9D                 lda FACEXP
B54A 29 7F                 and #MSBCLR
B54C              ;
B54C 05 9E                 ora FACMANT
B54E 8D 76 B0              sta SGNFLG1
B551              ;
B551 AE 7A B0              ldx FPREAL+2
B554 AC 7B B0              ldy FPREAL+3
B557              ;
B557 86 0A                 stx REALPTRS+2
B559 84 0B                 sty REALPTRS+3
B55B              ;
B55B 20 2B EB              jsr COPYFAC
B55E              ;
B55E              ;
B55E              ; Subtract TEMP2 from TEMP1, compute SGNFLG2, and save
B55E              ; the imaginary coefficient to the determinant Y.
B55E              ;
B55E A9 9D                 lda #TEMPI2
B560 A0 B0                 ldy /TEMPI2
B562              ;
B562 20 F9 EA              jsr LOADFAC
B565              ;
B565 A9 98                 lda #TEMPI1
B567 A0 B0                 ldy /TEMPI1
B569              ;
B569 20 A7 E7              jsr FSUB
B56C              ;
```

```
B56C A5 9D               lda FACEXP
B56E 29 7F               and #MSBCLR
B570            ;
B570 05 9E               ora FACMANT
B572 8D 77 B0            sta SGNFLG2
B575            ;
B575 AE 80 B0            ldx FPIMAG+2
B578 AC 81 B0            ldy FPIMAG+3
B57B            ;
B57B 86 1A               stx IMAGPTRS+2
B57D 84 1B               sty IMAGPTRS+3
B57F            ;
B57F 20 2B EB            jsr COPYFAC
B582            ;
B582            ;
B582            ; Determine if either the real or imaginary coefficients
B582            ; are zero, and if so, set the inverse matrix coefficients
B582            ; to HIGHVAL or 7.922816252E+28.
B582            ;
B582 AD 76 B0            lda SGNFLG1
B585 F0 05               beq INVZERO
B587            ;
B587 AD 77 B0            lda SGNFLG2
B58A D0 3C               bne NONZERO
B58C            ;
B58C            ;
B58C            ; One or both of the sign flags were found to be zero.  Set
B58C            ; all coefficients of Z to HIGHVAL.
B58C            ;
B58C AD 7C B0   INVZERO  lda REAL3L
B58F AC 7D B0            ldy REAL3H
B592            ;
B592 85 08               sta REALPTRS
B594 84 09               sty REALPTRS+1
B596            ;
B596 AD 82 B0            lda IMAG3L
B599 AC 83 B0            ldy IMAG3H
B59C            ;
B59C 85 18               sta IMAGPTRS
B59E 84 19               sty IMAGPTRS+1
B5A0            ;
B5A0 A2 04               ldx #ARRAYSIZ
B5A2 D0 14               bne >4                ; always taken
B5A4            ;
B5A4 A5 08      ^1       lda REALPTRS
B5A6 69 05               adc #IMAGOFF
B5A8 90 02               bcc >2
B5AA            ;
B5AA E6 09               inc REALPTRS+1
B5AC            ;
B5AC 85 08      ^2       sta REALPTRS
B5AE            ;
B5AE A5 18               lda IMAGPTRS
B5B0 69 05               adc #IMAGOFF
B5B2 90 02               bcc >3
B5B4            ;
B5B4 E6 19               inc IMAGPTRS+1
B5B6            ;
B5B6 85 18      ^3       sta IMAGPTRS
B5B8            ;
B5B8 A0 04      ^4       ldy #FPLEN-1
```

```
B5BA                ;
B5BA B9 E7 B0       ^5       lda HIGHVAL,Y
B5BD                ;
B5BD 91 08                   sta (REALPTRS),Y
B5BF 91 18                   sta (IMAGPTRS),Y
B5C1                ;
B5C1 88                      dey
B5C2 10 F6                   bpl <5
B5C4                ;
B5C4 CA                      dex
B5C5 D0 DD                   bne <1
B5C7                ;
B5C7 60                      rts
B5C8                ;
B5C8                ;
B5C8                ; Both sign flags were found to be not zero.  Compute the
B5C8                ; inverse matrix coefficients of matrix X and save them
B5C8                ; to matrix Z.  The determinant coefficients are still at
B5C8                ; REALPTRS2/3 and IMAGPTRS2/3.
B5C8                ;
B5C8 A9 00          NONZERO  lda #ZERO           ; initialize offset counter
B5CA 85 1E                   sta COUNT
B5CC                ;
B5CC                ;
B5CC                ; Compute real inverse.
B5CC                ;
B5CC 20 04 B7       ^1       jsr BLDARGS4
B5CF 20 33 B3                jsr CXDIV
B5D2                ;
B5D2 A6 1E                   ldx COUNT
B5D4                ;
B5D4 BD DA B0                lda INVLSB,X
B5D7 F0 2B                   beq >2
B5D9                ;
B5D9                ;
B5D9                ; Multiply real coefficient by -1.
B5D9                ;
B5D9 A0 B0                   ldy /INVLSB
B5DB                ;
B5DB 20 F9 EA                jsr LOADFAC
B5DE                ;
B5DE A5 0C                   lda REALPTRS+4
B5E0 A4 0D                   ldy REALPTRS+5
B5E2                ;
B5E2 20 7F E9                jsr FMULT
B5E5                ;
B5E5 A6 0C                   ldx REALPTRS+4
B5E7 A4 0D                   ldy REALPTRS+5
B5E9                ;
B5E9 20 2B EB                jsr COPYFAC
B5EC                ;
B5EC                ;
B5EC                ; Multiply imaginary coefficient by -1.
B5EC                ;
B5EC A6 1E                   ldx COUNT
B5EE                ;
B5EE BD DA B0                lda INVLSB,X
B5F1 A0 B0                   ldy /INVLSB
B5F3                ;
B5F3 20 F9 EA                jsr LOADFAC
B5F6                ;
```

```
B5F6 A5 1C                   lda IMAGPTRS+4
B5F8 A4 1D                   ldy IMAGPTRS+5
B5FA               ;
B5FA 20 7F E9                jsr FMULT
B5FD               ;
B5FD A6 1C                   ldx IMAGPTRS+4
B5FF A4 1D                   ldy IMAGPTRS+5
B601               ;
B601 20 2B EB                jsr COPYFAC
B604               ;
B604 E6 1E         ^2        inc COUNT
B606               ;
B606 A5 1E                   lda COUNT
B608 C9 04                   cmp #ARRAYSIZ
B60A D0 C0                   bne <1
B60C               ;
B60C 60                      rts
B60D               ;
B60D               ;
B60D               ; Complex CSMUL routine.  This routine multiplies the
B60D               ; complex 2x2 floating point array with a complex floating
B60D               ; point scalar variable and is written as:
B60D               ;
B60D               ; CALL "CSMUL", X(0,0,0),Y(0),Z(0,0,0)         X * Y form Z
B60D               ;
B60D               ; The real and imaginary coefficients of each array element
B60D               ; are multiplied by the real and imaginary coefficients of
B60D               ; the scalar variable.
B60D               ;
B60D               ; Put Y into FAC, multiply by X using CXMUL, and save to Z.
B60D               ;
B60D A9 00         CSMUL     lda #ZERO            ; initialize offset counter
B60F 85 1E                   sta COUNT
B611 F0 03                   beq >2               ; always taken
B613               ;
B613 20 4B B7      ^1        jsr BLDARGS5
B616               ;
B616 20 D1 B2      ^2        jsr CXMUL
B619               ;
B619 E6 1E                   inc COUNT
B61B               ;
B61B A5 1E                   lda COUNT
B61D C9 04                   cmp #ARRAYSIZ
B61F D0 F2                   bne <1
B621               ;
B621 60                      rts
B622               ;
B622               ;
B622                         icl "CXFPMATH.5.L"
```

LLOAD CXFPMATH.5.L,A$4000

```
B622                        ttl "CXFPMATH.5 Routines, CXFPMATH.5.L"
B622                ;
B622                ;
B622                ; CXFPMATH.5.L
B622                ;
B622                ;
B622                ; Point to the next working real, imaginary, and output
B622                ; arguments.  BLDARDS1 is used by AYADD and AYSUB.
B622                ;
B622 A6 1E          BLDARGS1 ldx COUNT
B624                ;
B624 BD A2 B0                lda ADDSUB,X
B627 85 1F                   sta OFFSET
B629                ;
B629 A2 00                   ldx #ZERO
B62B                ;
B62B 18             ^1       clc
B62C                ;
B62C BD 78 B0                lda FPREAL+0,X
B62F BC 79 B0                ldy FPREAL+1,X
B632                ;
B632 65 1F                   adc OFFSET
B634 90 01                   bcc >2
B636                ;
B636 C8                      iny
B637                ;
B637 95 08          ^2       sta REALPTRS+0,X
B639 94 09                   sty REALPTRS+1,X
B63B                ;
B63B 18                      clc
B63C                ;
B63C BD 7E B0                lda FPIMAG+0,X
B63F BC 7F B0                ldy FPIMAG+1,X
B642                ;
B642 65 1F                   adc OFFSET
B644 90 01                   bcc >3
B646                ;
B646 C8                      iny
B647                ;
B647 95 18          ^3       sta IMAGPTRS+0,X
B649 94 19                   sty IMAGPTRS+1,X
B64B                ;
B64B E8                      inx
B64C E8                      inx
B64D                ;
B64D E0 06                   cpx #6
B64F 90 DA                   bcc <1
B651                ;
B651 60                      rts
B652                ;
B652                ;
B652                ; Point to the next working real, imaginary, and output
B652                ; arguments.  BLDARDS2 is used by AYMUL.
B652                ;
B652 A6 1E          BLDARGS2 ldx COUNT
B654                ;
B654                ;
B654                ; Setup real coefficients.
B654                ;
B654 18                      clc
```

```
B655            ;
B655 AD 78 B0           lda FPREAL+0
B658 AC 79 B0           ldy FPREAL+1
B65B            ;
B65B 7D A6 B0           adc MULOFF1,X
B65E 90 01              bcc >1
B660            ;
B660 C8                 iny
B661            ;
B661 85 08      ^1      sta REALPTRS+0
B663 84 09              sty REALPTRS+1
B665            ;
B665 18                 clc
B666            ;
B666 AD 7A B0           lda FPREAL+2
B669 AC 7B B0           ldy FPREAL+3
B66C            ;
B66C 7D AE B0           adc MULOFF2,X
B66F 90 01              bcc >2
B671            ;
B671 C8                 iny
B672            ;
B672 85 0A      ^2      sta REALPTRS+2
B674 84 0B              sty REALPTRS+3
B676            ;
B676 BD B6 B0           lda MULLSB1,X
B679 A0 B0              ldy /MULLSB1
B67B            ;
B67B 85 0C              sta REALPTRS+4
B67D 84 0D              sty REALPTRS+5
B67F            ;
B67F            ;
B67F            ; Setup up imaginary coefficients.
B67F            ;
B67F 18                 clc
B680            ;
B680 AD 7E B0           lda FPIMAG+0
B683 AC 7F B0           ldy FPIMAG+1
B686            ;
B686 7D A6 B0           adc MULOFF1,X
B689 90 01              bcc >3
B68B            ;
B68B C8                 iny
B68C            ;
B68C 85 18      ^3      sta IMAGPTRS+0
B68E 84 19              sty IMAGPTRS+1
B690            ;
B690 18                 clc
B691            ;
B691 AD 80 B0           lda FPIMAG+2
B694 AC 81 B0           ldy FPIMAG+3
B697            ;
B697 7D AE B0           adc MULOFF2,X
B69A 90 01              bcc >4
B69C            ;
B69C C8                 iny
B69D            ;
B69D 85 1A      ^4      sta IMAGPTRS+2
B69F 84 1B              sty IMAGPTRS+3
B6A1            ;
B6A1 BD BE B0           lda MULLSB2,X
```

```
B6A4 A0 B0              ldy /MULLSB2
B6A6            ;
B6A6 85 1C              sta IMAGPTRS+4
B6A8 84 1D              sty IMAGPTRS+5
B6AA            ;
B6AA 60                 rts
B6AB            ;
B6AB            ;
B6AB            ; Point to the next working real, imaginary, and output
B6AB            ; arguments.  BLDARDS3 is used by AYINV to compute a
B6AB            ; determinant.
B6AB            ;
B6AB A6 1E      BLDARGS3 ldx COUNT
B6AD            ;
B6AD            ;
B6AD            ; Setup real coefficients.
B6AD            ;
B6AD 18                 clc
B6AE            ;
B6AE AD 78 B0           lda FPREAL+0
B6B1 AC 79 B0           ldy FPREAL+1
B6B4            ;
B6B4 7D CA B0           adc DETOFF1,X
B6B7 90 01              bcc >1
B6B9            ;
B6B9 C8                 iny
B6BA            ;
B6BA 85 08      ^1      sta REALPTRS+0
B6BC 84 09              sty REALPTRS+1
B6BE            ;
B6BE 18                 clc
B6BF            ;
B6BF AD 78 B0           lda FPREAL+0
B6C2 AC 79 B0           ldy FPREAL+1
B6C5            ;
B6C5 7D CC B0           adc DETOFF2,X
B6C8 90 01              bcc >2
B6CA            ;
B6CA C8                 iny
B6CB            ;
B6CB 85 0A      ^2      sta REALPTRS+2
B6CD 84 0B              sty REALPTRS+3
B6CF            ;
B6CF BD CE B0           lda DETLSB1,X
B6D2 A0 B0              ldy /DETLSB1
B6D4            ;
B6D4 85 0C              sta REALPTRS+4
B6D6 84 0D              sty REALPTRS+5
B6D8            ;
B6D8            ;
B6D8            ; Setup up imaginary coefficients.
B6D8            ;
B6D8 18                 clc
B6D9            ;
B6D9 AD 7E B0           lda FPIMAG+0
B6DC AC 7F B0           ldy FPIMAG+1
B6DF            ;
B6DF 7D CA B0           adc DETOFF1,X
B6E2 90 01              bcc >3
B6E4            ;
B6E4 C8                 iny
```

```
B6E5             ;
B6E5 85 18       ^3       sta IMAGPTRS+0
B6E7 84 19                sty IMAGPTRS+1
B6E9             ;
B6E9 18                   clc
B6EA             ;
B6EA AD 7E B0             lda FPIMAG+0
B6ED AC 7F B0             ldy FPIMAG+1
B6F0             ;
B6F0 7D CC B0             adc DETOFF2,X
B6F3 90 01                bcc >4
B6F5             ;
B6F5 C8                   iny
B6F6             ;
B6F6 85 1A       ^4       sta IMAGPTRS+2
B6F8 84 1B                sty IMAGPTRS+3
B6FA             ;
B6FA BD D0 B0             lda DETLSB2,X
B6FD A0 B0                ldy /DETLSB2
B6FF             ;
B6FF 85 1C                sta IMAGPTRS+4
B701 84 1D                sty IMAGPTRS+5
B703             ;
B703 60                   rts
B704             ;
B704             ;
B704             ; Point to the next working real, imaginary, and output
B704             ; arguments.  BLDARDS4 is used by AYINV to compute an
B704             ; inverse matrix.
B704             ;
B704 A6 1E       BLDARGS4 ldx COUNT
B706             ;
B706             ;
B706             ; Setup real coefficients.
B706             ;
B706 18                   clc
B707             ;
B707 AD 78 B0             lda FPREAL+0
B70A AC 79 B0             ldy FPREAL+1
B70D             ;
B70D 7D D2 B0             adc INVOFF1,X
B710 90 01                bcc >1
B712             ;
B712 C8                   iny
B713             ;
B713 85 08       ^1       sta REALPTRS+0
B715 84 09                sty REALPTRS+1
B717             ;
B717 18                   clc
B718             ;
B718 AD 7C B0             lda FPREAL+4
B71B AC 7D B0             ldy FPREAL+5
B71E             ;
B71E 7D D6 B0             adc INVOFF2,X
B721 90 01                bcc >2
B723             ;
B723 C8                   iny
B724             ;
B724 85 0C       ^2       sta REALPTRS+4
B726 84 0D                sty REALPTRS+5
B728             ;
```

```
B728            ;
B728            ; Setup up imaginary coefficients.
B728            ;
B728 18                  clc
B729            ;
B729 AD 7E B0            lda FPIMAG+0
B72C AC 7F B0            ldy FPIMAG+1
B72F            ;
B72F 7D D2 B0            adc INVOFF1,X
B732 90 01               bcc >3
B734            ;
B734 C8                  iny
B735            ;
B735 85 18      ^3       sta IMAGPTRS+0
B737 84 19               sty IMAGPTRS+1
B739            ;
B739 18                  clc
B73A            ;
B73A AD 82 B0            lda FPIMAG+4
B73D AC 83 B0            ldy FPIMAG+5
B740            ;
B740 7D D6 B0            adc INVOFF2,X
B743 90 01               bcc >4
B745            ;
B745 C8                  iny
B746            ;
B746 85 1C      ^4       sta IMAGPTRS+4
B748 84 1D               sty IMAGPTRS+5
B74A            ;
B74A 60                  rts
B74B            ;
B74B            ;
B74B            ; Point to the next working real, imaginary, and output
B74B            ; arguments.  BLDARDS5 is used by CSMUL to multiply a
B74B            ; complex array and a complex scalar.
B74B            ;
B74B A6 1E      BLDARGS5 ldx COUNT
B74D            ;
B74D            ;
B74D            ; Setup real coefficients.
B74D            ;
B74D 18                  clc
B74E            ;
B74E AD 78 B0            lda FPREAL+0
B751 AC 79 B0            ldy FPREAL+1
B754            ;
B754 7D DE B0            adc CSOFF,X
B757 90 01               bcc >1
B759            ;
B759 C8                  iny
B75A            ;
B75A 85 08      ^1       sta REALPTRS+0
B75C 84 09               sty REALPTRS+1
B75E            ;
B75E 18                  clc
B75F            ;
B75F AD 7C B0            lda FPREAL+4
B762 AC 7D B0            ldy FPREAL+5
B765            ;
B765 7D DE B0            adc CSOFF,X
B768 90 01               bcc >2
```

```
B76A            ;
B76A C8                 iny
B76B            ;
B76B 85 0C      ^2      sta REALPTRS+4
B76D 84 0D              sty REALPTRS+5
B76F            ;
B76F            ;
B76F            ; Setup up imaginary coefficients.
B76F            ;
B76F 18                 clc
B770            ;
B770 AD 7E B0           lda FPIMAG+0
B773 AC 7F B0           ldy FPIMAG+1
B776            ;
B776 7D DE B0           adc CSOFF,X
B779 90 01              bcc >3
B77B            ;
B77B C8                 iny
B77C            ;
B77C 85 18      ^3      sta IMAGPTRS+0
B77E 84 19              sty IMAGPTRS+1
B780            ;
B780 18                 clc
B781            ;
B781 AD 82 B0           lda FPIMAG+4
B784 AC 83 B0           ldy FPIMAG+5
B787            ;
B787 7D DE B0           adc CSOFF,X
B78A 90 01              bcc >4
B78C            ;
B78C C8                 iny
B78D            ;
B78D 85 1C      ^4      sta IMAGPTRS+4
B78F 84 1D              sty IMAGPTRS+5
B791            ;
B791 60                 rts
B792            ;
B792            ;

BSAVE CXFPMATH,A$0900,B,L$0792

B792                    usr CXFPMATH
B792            ;
B792            ;
B792                    end 111

*** End of Assembly
```

Symbol List starts at 0x7800, ends at 0x7DBE, used 0x05BE, remaining 0x392A

Symbols unsorted:

REALPTRS	0008	IMAGPTRS	0018	COUNT	001E	OFFSET	001F	FRETOP	006F
HIMEM	0073	ARGGUARD	0092	FACEXP	009D	FACMANT	009E	FACSIGN	00A2
ARGSIGN	00AA	XORSIGN	00AB	FACGUARD	00AC	CHRGET	00B1	CHRGOT	00B7
ZERO	0000	MSBCLR	007F	MSBSET	0080	NEGONE	00FF	ARRAYSIZ	0004
VARS2	0004	IMAGOFF	0005	FPLEN	0005	CMDLEN	0005	ARGLEN	0006
VARS3	0006	CXFPOFF	000A	RETURN	008D	ESCAPE	009B	SPACE	00A0
QUOTE	00A2	COMMA	00AC	PAGESIZE	0100	DOSWARM	03D0	KEY	C000
CLRKEY	C010	ROM2WP	C082	PTRGET	DFE3	FSUB	E7A7	FADD	E7BE
FMULT	E97F	LOADARG	E9E3	FDIV	EA66	LOADFAC	EAF9	COPYFAC	EB2B
MOVFA	EB53	FSGN	EB90	FSQR	EE8D	FEXP	EE97	FCOS	EFEA
FSIN	EFF1	FTAN	F03A	FATAN	F09E	CROUT	FD8E	PRBYTE	FDDA
PRHEX	FDE3	COUT	FDED	INITBL	B000	JMPTBL	B009	HASHTBL	B040
HASHSIZ	000F	PTRLNTBL	B04F	FIRSTIME	B070	HASHVAL	B071	CMDNDX	B072
PTRNUM	B073	PTRNDX	B074	PTRLEN	B075	SGNFLG1	B076	SGNFLG2	B077
FPREAL	B078	REAL1L	B078	REAL1H	B079	REAL2L	B07A	REAL2H	B07B
REAL3L	B07C	REAL3H	B07D	FPIMAG	B07E	IMAG1L	B07E	IMAG1H	B07F
IMAG2L	B080	IMAG2H	B081	IMAG3L	B082	IMAG3H	B083	TEMP1	B084
TEMP2	B089	TEMPR1	B08E	TEMPR2	B093	TEMPI1	B098	TEMPI2	B09D
ADDSUB	B0A2	MULOFF1	B0A6	MULOFF2	B0AE	MULLSB1	B0B6	MULLSB2	B0BE
MULOUT	B0C6	DETOFF1	B0CA	DETOFF2	B0CC	DETLSB1	B0CE	DETLSB2	B0D0
INVOFF1	B0D2	INVOFF2	B0D6	INVLSB	B0DA	CSOFF	B0DE	MINUSONE	B0E2
HIGHVAL	B0E7	PIDIV2	B0EC	PI	B0F1	VERSION	B0FE	LOADER	B100
TEMPMOD	B134	LIBENT	B137	LINKER	B17A	PRNTERR	B1C5	ERRS	B1D3
ERR1	B1D3	ERR2	B1F9	ERR3	B212	ERR4	B230	LGNOT	B24D
LGAND	B25D	LGORA	B261	LGEOR	B265	LGCALC	B267	LGMOD1	B271
LGMOD2	B278	CXADD	B27D	CXSUB	B2A7	CXMUL	B2D1	CXDIV	B333
CX2PL	B3F0	C2PEXIT	B466	PL2CX	B46D	AYADD	B49D	AYSUB	B4B2
AYMUL	B4C7	AYINV	B528	INVZERO	B58C	NONZERO	B5C8	CSMUL	B60D
BLDARGS1	B622	BLDARGS2	B652	BLDARGS3	B6AB	BLDARGS4	B704	BLDARGS5	B74B

Symbols alphabetically sorted:

ADDSUB	B0A2	ARGGUARD	0092	ARGLEN	0006	ARGSIGN	00AA	ARRAYSIZ	0004
AYADD	B49D	AYINV	B528	AYMUL	B4C7	AYSUB	B4B2	BLDARGS1	B622
BLDARGS2	B652	BLDARGS3	B6AB	BLDARGS4	B704	BLDARGS5	B74B	C2PEXIT	B466
CHRGET	00B1	CHRGOT	00B7	CLRKEY	C010	CMDLEN	0005	CMDNDX	B072
COMMA	00AC	COPYFAC	EB2B	COUNT	001E	COUT	FDED	CROUT	FD8E
CSMUL	B60D	CSOFF	B0DE	CX2PL	B3F0	CXADD	B27D	CXDIV	B333
CXFPOFF	000A	CXMUL	B2D1	CXSUB	B2A7	DETLSB1	B0CE	DETLSB2	B0D0
DETOFF1	B0CA	DETOFF2	B0CC	DOSWARM	03D0	ERR1	B1D3	ERR2	B1F9
ERR3	B212	ERR4	B230	ERRS	B1D3	ESCAPE	009B	FACEXP	009D
FACGUARD	00AC	FACMANT	009E	FACSIGN	00A2	FADD	E7BE	FATAN	F09E
FCOS	EFEA	FDIV	EA66	FEXP	EE97	FIRSTIME	B070	FMULT	E97F
FPIMAG	B07E	FPLEN	0005	FPREAL	B078	FRETOP	006F	FSGN	EB90
FSIN	EFF1	FSQR	EE8D	FSUB	E7A7	FTAN	F03A	HASHSIZ	000F
HASHTBL	B040	HASHVAL	B071	HIGHVAL	B0E7	HIMEM	0073	IMAG1H	B07F
IMAG1L	B07E	IMAG2H	B081	IMAG2L	B080	IMAG3H	B083	IMAG3L	B082
IMAGOFF	0005	IMAGPTRS	0018	INITBL	B000	INVLSB	B0DA	INVOFF1	B0D2
INVOFF2	B0D6	INVZERO	B58C	JMPTBL	B009	KEY	C000	LGAND	B25D
LGCALC	B267	LGEOR	B265	LGMOD1	B271	LGMOD2	B278	LGNOT	B24D
LGORA	B261	LIBENT	B137	LINKER	B17A	LOADARG	E9E3	LOADER	B100
LOADFAC	EAF9	MINUSONE	B0E2	MOVFA	EB53	MSBCLR	007F	MSBSET	0080
MULLSB1	B0B6	MULLSB2	B0BE	MULOFF1	B0A6	MULOFF2	B0AE	MULOUT	B0C6
NEGONE	00FF	NONZERO	B5C8	OFFSET	001F	PAGESIZE	0100	PI	B0F1

```
 PIDIV2    B0EC   PL2CX     B46D   PRBYTE    FDDA   PRHEX     FDE3   PRNTERR   B1C5
 PTRGET    DFE3   PTRLEN    B075   PTRLNTBL  B04F   PTRNDX    B074   PTRNUM    B073
 QUOTE     00A2   REAL1H    B079   REAL1L    B078   REAL2H    B07B   REAL2L    B07A
 REAL3H    B07D   REAL3L    B07C   REALPTRS  0008   RETURN    008D   ROM2WP    C082
 SGNFLG1   B076   SGNFLG2   B077   SPACE     00A0   TEMP1     B084   TEMP2     B089
 TEMPI1    B098   TEMPI2    B09D   TEMPMOD   B134   TEMPR1    B08E   TEMPR2    B093
 VARS2     0004   VARS3     0006   VERSION   B0FE   XORSIGN   00AB   ZERO      0000
```

Symbols numerically sorted:

```
 ZERO      0000   VARS2     0004   ARRAYSIZ  0004   IMAGOFF   0005   FPLEN     0005
 CMDLEN    0005   VARS3     0006   ARGLEN    0006   REALPTRS  0008   CXFPOFF   000A
 HASHSIZ   000F   IMAGPTRS  0018   COUNT     001E   OFFSET    001F   FRETOP    006F
 HIMEM     0073   MSBCLR    007F   MSBSET    0080   RETURN    008D   ARGGUARD  0092
 ESCAPE    009B   FACEXP    009D   FACMANT   009E   SPACE     00A0   QUOTE     00A2
 FACSIGN   00A2   ARGSIGN   00AA   XORSIGN   00AB   FACGUARD  00AC   COMMA     00AC
 CHRGET    00B1   CHRGOT    00B7   NEGONE    00FF   PAGESIZE  0100   DOSWARM   03D0
 INITBL    B000   JMPTBL    B009   HASHTBL   B040   PTRLNTBL  B04F   FIRSTIME  B070
 HASHVAL   B071   CMDNDX    B072   PTRNUM    B073   PTRNDX    B074   PTRLEN    B075
 SGNFLG1   B076   SGNFLG2   B077   REAL1L    B078   FPREAL    B078   REAL1H    B079
 REAL2L    B07A   REAL2H    B07B   REAL3L    B07C   REAL3H    B07D   IMAG1L    B07E
 FPIMAG    B07E   IMAG1H    B07F   IMAG2L    B080   IMAG2H    B081   IMAG3L    B082
 IMAG3H    B083   TEMP1     B084   TEMP2     B089   TEMPR1    B08E   TEMPR2    B093
 TEMPI1    B098   TEMPI2    B09D   ADDSUB    B0A2   MULOFF1   B0A6   MULOFF2   B0AE
 MULLSB1   B0B6   MULLSB2   B0BE   MULOUT    B0C6   DETOFF1   B0CA   DETOFF2   B0CC
 DETLSB1   B0CE   DETLSB2   B0D0   INVOFF1   B0D2   INVOFF2   B0D6   INVLSB    B0DA
 CSOFF     B0DE   MINUSONE  B0E2   HIGHVAL   B0E7   PIDIV2    B0EC   PI        B0F1
 VERSION   B0FE   LOADER    B100   TEMPMOD   B134   LIBENT    B137   LINKER    B17A
 PRNTERR   B1C5   ERRS      B1D3   ERR1      B1D3   ERR2      B1F9   ERR3      B212
 ERR4      B230   LGNOT     B24D   LGAND     B25D   LGORA     B261   LGEOR     B265
 LGCALC    B267   LGMOD1    B271   LGMOD2    B278   CXADD     B27D   CXSUB     B2A7
 CXMUL     B2D1   CXDIV     B333   CX2PL     B3F0   C2PEXIT   B466   PL2CX     B46D
 AYADD     B49D   AYSUB     B4B2   AYMUL     B4C7   AYINV     B528   INVZERO   B58C
 NONZERO   B5C8   CSMUL     B60D   BLDARGS1  B622   BLDARGS2  B652   BLDARGS3  B6AB
 BLDARGS4  B704   BLDARGS5  B74B   KEY       C000   CLRKEY    C010   ROM2WP    C082
 PTRGET    DFE3   FSUB      E7A7   FADD      E7BE   FMULT     E97F   LOADARG   E9E3
 FDIV      EA66   LOADFAC   EAF9   COPYFAC   EB2B   MOVFA     EB53   FSGN      EB90
 FSQR      EE8D   FEXP      EE97   FCOS      EFEA   FSIN      EFF1   FTAN      F03A
 FATAN     F09E   CROUT     FD8E   PRBYTE    FDDA   PRHEX     FDE3   COUT      FDED
```