TSC 6800 Scientific Functions Package

SL68-20

ECHNICAL SYSTEMS CONSULTANTS

SCIENTIFIC FUNCTIONS

```
*
*
*    COPYRIGHT (C) 1976 BY
*  TECHNICAL SYSTEMS CONSULTANTS
* BOX 2574  W. LAFAYETTE  IN  47906
*
*  THE TSC SCIENTIFIC FUNCTIONS PACKAGE IS DESIGNED
* TO UTILIZE THE TSC FLOATING POINT PACKAGE FOR
* ARITHMETIC PROCESSING.  THE ROUTINES IMPLEMENT
* SOME OF THE MOST USED SCIENTIFIC FUNCTIONS USING
* THE CORDIC (COORDINATE ROTATION DIGITAL COMPUTER)
* ALGORITHM, AN OBSCURE BUT USEFUL ALGORITHM BECAUSE
* IT INVOLVES FEWER MULTIPLICATIONS (BUT MORE ADDITIONS)
* THAN OTHER APPROXIMATION TECHNIQUES, AND THEREFORE
* RUNS FASTER.  THE TECHNIQUE INVOLVES ROTATING A
* COORDINATE SYSTEM TO OBTAIN A MAPPING FROM THE FIRST
* SYSTEM INTO A SECOND COORDINATE SYSTEM WHICH IS
* ROTATIONALLY DISPLACED FROM THE FIRST.  THE ROTATION
* IS ACCOMPLISHED IN A SEQUENCE OF CAREFULLY CALCULATED
* DISPLACEMENTS SUCH THAT THE INTERMEDIATE MAPPINGS
* CAN BE PERFORMED BY MERELY ADDING AND SHIFTING, OR
* IN THIS CASE, ADDING AND DECREMENTING EXPONENTS.
* BE REMINDED THAT THIS PACKAGE CONTAINS SUBROUTINES ONLY.
*  THIS PACKAGE ALSO CONTAINS SEVERAL USEFUL CONSTANTS
* FOR THE USER.  AMONG THESE ARE PI, E, AND LN10.  SEE
* THE DATA BLOCKS BELOW.
*  THIS PROGRAM IS DESIGN SUCH THAT THE USER DESIRING
* ONLY TRIG FUNCTIONS CAN DELETE THE SECTION OF CODE
* BEYOND 0700 AS THAT CODE IS ONLY USED FOR THE
* HYPERBOLIC FAMILY OF FUNCTIONS (LOGS, ETC. ).
*  THE PRECISION OF THE APPROXIMATIONS IS LIMITED TO
* SIX DIGITS (MORE DIGITS WOULD TAKE PROPORTIONATELY
* LONGER).  THE ENTRY ADDRESSES, ARGUMENT AND RESULT
* LOCATIONS, AND RANGE OF ARGUMENTS ARE SHOWN IN THE
* TABLES BELOW:
*
*    OPERATION    ARGUMENT    RESULT      ARGUMENT LIMITS
*      SIN(X)     XSIGN-XEX  RSIGN-ACEXP   .1<X<INF   (ABS VAL)
*      COS(X)     XSIGN-XEX  RSIGN-ACEXP  .1< X <INF (ABS VAL)
*      TAN(X)     XSIGN-XEX  RSIGN-ACEXP  .1< X <INF (ABS VAL)
*    ARCSIN(X)    XSIGN-XEX  RSIGN-ACEXP .01< X <1.0 (ABS VAL)
*    ARCCOS(X)    XSIGN-XEX  RSIGN-ACEXP .01< X <1.0 (ABS VAL)
*    ARCTAN(X)    XSIGN-XEX  RSIGN-ACEXP .01< X <INF (ABS VAL)
*     SINH(X)     XSIGN-XEX  RSIGN-ACEXP  .1< X <LN(10) (ABS VAL)
*     COSH(X)     XSIGN-XEX  RSIGN-ACEXP  .1< X <LN(10) (ABS VAL)
*     TANH(X)     XSIGN-XEX  RSIGN-ACEXP  .1< X <LN(10) (ABS VAL)
*       E↑X       XSIGN-XEX  RSIGN-ACEXP  0 < X < 227 (ABS VAL)
*      10↑X       XSIGN-XEX  RSIGN-ACEXP  0 < X < 98  (ABS VAL)
*      LN(X)      XSIGN-XEX  RSIGN-ACEXP    ANY POSITIVE VALUE
*     LOG(X)      XSIGN-XEX  RSIGN-ACEXP    ANY POSITIVE VALUE
*    SQRT(X)      XSIGN-XEX  RSIGN-ACEXP  ANY POSITIVE VALUE AND 0
*       X↑Y       XSIGN-XEX                 ANY POSITIVE VALUE
*                 YSIGN-YEX  RSIGN-ACEXP    ANY VALUE
```

```
*     1/X          XSIGN-XEX  RSIGN-ACEXP    X NOT ZERO
*
*
*  NOTES:
*
*  1)  TRIG FUNCTIONS CAN BE COMPUTED IN DEGREE OR RADIAN MODE
*     THE INDICATOR BYTE "MODE" (0076 HEX) SPECIFIES THE MODE
*     00 = DEGREE MODE,   FF = RADIAN MODE
*
*  2)  E↑X  IS THE NUMBER E RAISED TO THE X POWER.
*
*  3)  LN(X) IS THE NATURAL LOG (BASE E) OF X
*
*  4)  LOG(X) IS THE COMMON LOG (BASE 10) OF X
*
*  5)  SQRT(X) IS THE SQUARE ROOT OF X
*
*  6)  X↑Y IS X RAISED TO THE Y POWER.   X IS IN XSIGN-XEX
*     Y IS IN YSIGN-YEX.
*
*  7)  1/X  IS THE INVERSE OF X
*
*  8)  OVERFLOW IS INDICATED BY "OVFL" (003A HEX) NOT ZERO
*
*  9)  ILLEGAL OPERATION IS INDICATED BY "NLEGAL" (0072)
*     NOT ZERO.  THE USER MUST CLEAR NLEGAL BEFORE
*     CALLING A PARTICULAR ROUTINE.
*
* 10)  INF ABOVE MEANS .999999999 X 10↑99
*     -INF ABOVE MEANS .100000000 X 10↑-99
*
*
*
*   OPERATION      ENTRY ADDRESS
*     SIN             0460
*     COS             0459
*     TAN             0496
*   ARCSIN            05B7
*   ARCCOS            059C
*   ARCTAN            0647
*    SINH             07BE
*    COSH             07C8
*    TANH             07CF
*     E↑X             075F
*    10↑X             0751
*    LN(X)            0836
*   LOG(X)            0820
*     1/X             0744
*    SQRT             0739
*    X↑Y              0701
*
*
*
*  EXTERNAL ROUTINES FROM FP PACKAGE
```

```
LOCN B1 B2 B3
     0194       FPDIV    EQU     $0194
     0262       XOPT01   EQU     $0262
     0275       ZCHK     EQU     $0275
     014A       NORM     EQU     $014A
     01CD       BCDADD   EQU     $01CD
                *
                *
                *   STORAGE ALSO USED BY FP PACKAGE
     0020       RSIGN    EQU     $0020
     002C       XSIGN    EQU     $002C
     0032       XEX      EQU     $0032
     002D       XOP      EQU     $002D
     0033       YSIGN    EQU     $0033
     0034       YOP      EQU     $0034
     0039       YEX      EQU     $0039
     003A       OVFL     EQU     $003A
     003F       XTEMP    EQU     $003F
     0041       XTEMP2   EQU     $0041
                *
                *
                *  TEMPORARY STORAGE FOR SCIENTIFIC PACKAGE
                         ORG     $0045
                *
0045            XPRIME   RMB     7
004C            YPRIME   RMB     7
0053            ZPRIME   RMB     7
005A            TPRIME   RMB     7
0061            ANGLE    RMB     7
0068            AFACTX   RMB     2
006A            XOPN     RMB     1
006B            YOPN     RMB     1
006C            AOPN     RMB     1
006D            LSTSGN   RMB     1
006E            SIGN     RMB     1
006F            OPX      RMB     2
0071            ITER     RMB     1
0072            NLEGAL   RMB     1
0073            C4       RMB     2
0075            QUADRT   RMB     1
                *
0076 00         MODE     FCB     0          SET DEGREE MODE
                *
                *
                         ORG     $0300
                *
                *
                * FUNDAMENTAL CONSTANTS
                *
0300 00         PI       FCB     $00,$03,$14,$15,$92,$65,$01
0307 00         LN10     FCB     $00,$02,$30,$25,$85,$09,$01
030E 00         E        FCB     $00,$02,$71,$82,$81,$83,$01
0314 01         *
                * MISCELLANEOUS NUMERICAL CONSTANTS
0315 00         NINETY   FCB     $00,$09,$00,$00,$00,$00,$02
031C 00         C360     FCB     $00,$03,$60,$00,$00,$00,$03
```

```
LOCN B1 B2 B3
0323 00           TWO     FCB     $00,$02,$00,$00,$00,$00,$01
032A 00           ONE     FCB     $00,$01,$00,$00,$00,$00,$01
0331 00           ZERO    FCB     $00,$00,$00,$00,$00,$00,$00
0338 00           INF     FCB     $00,$09,$99,$99,$99,$99,$63
033F 00           RCON    FCB     $00,$00,$00,$00,$05,$00
0345 00           RTODEG  FCB     $00,$05,$72,$95,$77,$95,$02
034B 02           *  CONSTANTS FOR THE ALGORITHM
034C 00           TRIGX   FCB     $00,$06,$75,$83,$61,$59,$00
0353 04           ANGFAC  FCB         $04,$50,$00,$00,$00,$02
0359 05                   FCB         $05,$71,$05,$93,$14,$01
035F 05                   FCB         $05,$72,$93,$86,$98,$00
0365 05                   FCB         $05,$72,$95,$76,$04,$FF
036B 05                   FCB         $05,$72,$95,$77,$89,$FE
0371 05                   FCB         $05,$72,$95,$77,$95,$FD
0377 05                   FCB         $05,$72,$95,$77,$95,$FC
037D 05                   FCB         $05,$72,$95,$77,$95,$FB
0383 05                   FCB         $05,$72,$95,$77,$95,$FA
0389 00           HYPCON  FCB     $00,$01,$11,$81,$37,$93,$01
0390 01           HYPANG  FCB         $01,$00,$33,$53,$48,$00
0396 01                   FCB         $01,$00,$00,$33,$33,$FF
039C 01                   FCB         $01,$00,$00,$00,$33,$FF
03A2 01                   FCB         $01,$00,$00,$00,$00,$FD
03A8 01                   FCB         $01,$00,$00,$00,$00,$FC
03AE 01                   FCB         $01,$00,$00,$00,$00,$FB
03B4 01                   FCB         $01,$00,$00,$00,$00,$FA
03BA 01                   FCB         $01,$00,$00,$00,$00,$F9
03C0 01                   FCB         $01,$00,$00,$00,$00,$F8
03C5 F8           *
                  *
                  * SIGN DETERMINANTS FOR SINE AND TANGENT
      0030        SSBYTE  EQU     00110000B
      0050        TSBYTE  EQU     01010000B
                  *
                  ** INTEGER
                  * THIS ROUTINE INTEGERIZES THE OPERAND POINTED TO BY X
03C6 A6 06        INTEGER LDA A   6,X         GET THE EXPONENT
03C8 C6 05                LDA B   #5
03CA 08                   INX
03CB 81 00        TESTIT  CMP A   #0          CHECK FOR 0
03CD 2F 06                BLE     GOTIT
03CF 08                   INX                 ADVANCE POINTER
03D0 5A                   DEC B
03D1 80 02                SUB A   #2
03D3 20 F6                BRA     TESTIT      DO AGAIN
03D5 26 0A        GOTIT   BNE     ALLZER      CHECK WHICH DIGIT
03D7 A6 00                LDA A   0,X         GET BYTE
03D9 84 F0                AND A   #$F0        MASK OUT LOW
03DB A7 00                STA A   0,X         PUT BACK
03DD 08           NEXDIG  INX                 ADVANCE POINTER
03DE 5A                   DEC B               KICK COUNTER
03DF 27 04                BEQ     INTDON
03E1 6F 00        ALLZER  CLR     0,X         SET 0
03E3 20 F8                BRA     NEXDIG
03E5 A6 00        INTDON  LDA A   0,X         GET EXPONENT
03E7 2A 02                BPL     ALLOK
```

```
LOCN B1 B2 B3
03E9 6F 00              CLR    0,X        SET 0
03EB 39          ALLOK  RTS               DONE
                 *
                 *
                 ** EXTRCT
                 * THIS ROUTINE EXTRACTS FACTORS FROM ARGUMENTS.  THE
                 * ARGUMENT IS ASSUMED IN XSIGN-XOP-XEX WHILE THE
                 * INDEX REGISTER POINTS TO THE FACTOR TO BE EXTRACTED.
                 * THE INTEGER NUMBER OF FACTORS IS RETURNED IN ZPRIME.
03EC DF 73       EXTRCT STX    C4         SAVE FACTOR POINTER
03EE BD 06 E5           JSR    MOVE3
03F1 BD 01 94           JSR    FPDIV      GO REMOVE FACTOR
03F4 CE 00 20           LDX    #RSIGN
03F7 8D CD              BSR    INTEGER    GO INTEGERIZE
03F9 CE 00 53           LDX    #ZPRIME
03FC BD 06 D5           JSR    MOVE1      SAVE THE FACTOR
03FF CE 00 2C           LDX    #XSIGN
0402 BD 06 EC           JSR    MOVE4      SAVE OPERAND
0405 CE 00 2C           LDX    #XSIGN
0408 BD 06 D5           JSR    MOVE1
040B DE 73              LDX    C4         GET FACTOR POINTER
040D BD 06 E5           JSR    MOVE3      SET YOP
0410 BD 01 80           JSR    FPMULT     GO MULTIPLY BACK
0413 CE 00 33           LDX    #YSIGN
0416 BD 06 D5           JSR    MOVE1      GET RESULT
0419 CE 00 45           LDX    #XPRIME    GET OPERAND
041C BD 06 DE           JSR    MOVE2
041F 7E 01 00           JMP    FPSUB      GO SUBTRACT
                 *
                 *
                 ** TRGFAC
                 * THIS ROUTINE IS THE FRONT END ROUTINE FOR
                 * TRIGONOMETRIC FUNCTIONS.  CONVERSION FROM RADIAN
                 * TO DEGREE MODE IS DONE HERE.  THE ANGLES ARE
                 * THEN REDUCED TO LESS THAN 90 DEGREES IN MAG-
                 * NITUDE AND MADE POSITIVE.  THIS ROUTINE THEN
                 * CALLS THE BASIC ROUTINE FOR TRIGONOMETRIC CORDIC.
0422 96 2C       TRGFAC LDA A  XSIGN      GET THE SIGN
0424 97 6E              STA A  SIGN       SAVE THE SIGN
0426 7F 00 2C           CLR    XSIGN      SET =00
0429 96 76              LDA A  MODE       GO GET MODE INDICATOR
042B 27 0F              BEQ    DEGREE     IF 0, THEN DEGREE
042D CE 03 45           LDX    #RTODEG    PT TO CONSTANT
0430 BD 06 E5           JSR    MOVE3      MOVE TO YOP
0433 BD 01 80           JSR    FPMULT     GO CONVERT TO DEGREES
0436 CE 00 2C           LDX    #XSIGN     POINT TO XOP
0439 BD 06 D5           JSR    MOVE1      MOVE RESULT THERE
043C CE 03 1C    DEGREE LDX    #C360      POINT TO CONSTANT
043F 8D AB              BSR    EXTRCT     GO REMOVE FACTORS OF 360
0441 CE 00 2C           LDX    #XSIGN
0444 BD 06 D5           JSR    MOVE1      MOVE THE RESULT
0447 CE 03 15           LDX    #NINETY
044A 8D A0              BSR    EXTRCT     REMOVE FACTORS OF 90
044C 96 54              LDA A  ZPRIME+1   GET QUADRANT
044E 97 75              STA A  QUADRT
```

```
LOCN B1 B2 B3
0450 96 21                      LDA A  RSIGN+1
0452 27 04                      BEQ    EXCEPT     IF 0, EXCEPTION
0454 BD 04 DB                   JSR    TRIGN      GO COMPUTE
0457 43                         COM A             SET FLAG
0458 39                 EXCEPT  RTS               DONE
                        *
                        *
                        *
                        ** COS
                        * ENTRY POINT FOR COSINE
0459 86 01              COS     LDA A  #01        SET PHASE DIFFERENCE
045B 7F 00 2C                   CLR    XSIGN
045E 20 01                      BRA    SIN0       GO DO SINE
                        *
                        *
                        ** SIN
                        * ENTRY POINT FOR SINE
0460 4F                 SIN     CLR A             SET BYTE
0461 36                 SIN0    PSH A             SAVE
0462 8D BE                      BSR    TRGFAC     GO COMPUTE
0464 07                         TPA               SAVE STATUS
0465 33                         PUL B             GET PHASE
0466 DB 75                      ADD B  QUADRT     ADD TO QUADRANT
0468 C4 03                      AND B  #3         MASK TO 0-3
046A D7 75                      STA B  QUADRT
046C 06                         TAP               RESTORE STATUS
046D 27 1C                      BEQ    SPECL      IF Z, SPECIAL CASE
046F CE 00 4C                   LDX    #YPRIME    POINT TO ANSWER
0472 56                         ROR B              CHECK QUADRANT
0473 24 03                      BCC    SIN1       IF 1 OR 3, THEN OK
0475 CE 00 45                   LDX    #XPRIME    ELSE SWITCH TO COS
0478 C6 30              SIN1    LDA B  #SSBYTE    GET SINE SIGN BYTE
                        *
                        *
                        ** TRGSGN
                        * THIS ROUTINE SELECTS THE SIGN OF THE RESULT
                        * OF TRIG FUNCTIONS AND THEN GOES TO THE
                        * ROUNDING PROCESSOR.
047A 96 75              TRGSGN  LDA A  QUADRT     GET QUADRANT INFO
047C 27 04              TESTSG  BEQ    GOTBIT     CHECK Q1
047E 58                         ASL B             SHIFT SIGN BYTE
047F 4A                         DEC A             COUNT DOWN QUADRANT
0480 20 FA                      BRA    TESTSG     DO AGAIN
0482 96 6E              GOTBIT  LDA A  SIGN       RETRIEVE SIGN
0484 5D                         TST B             CHECK NEW SIGN
0485 2A 01                      BPL    TRGDON     IF POS USE SAME SIGN
0487 43                         COM A             ELSE  USE OTHER
0488 7E 06 97           TRGDON  JMP    ROUND0     GO ROUND OFF
                        *
                        *
                        ** SPECL
                        * THIS ROUTINE HANDLES THE SPECIAL CASES OF
                        * TRIG FUNCTIONS.  IE.  0,90,180,270, AND 360
048B CE 03 31           SPECL   LDX    #ZERO      SPECIAL  CASE 0
048E 56                         ROR B             CHECK QUADRANT
```

```
LOCN B1 B2 B3
048F 24 E7              BCC     SIN1       IF 1 OR 3 THEN 0
0491 CE 03 2A           LDX     #ONE       SET FOR 1
0494 20 E2              BRA     SIN1
               *
               *
               ** TAN
               * ENTRY POINT FOR TANGENT
0496 BD 04 22  TAN      JSR     TRGFAC     GO COMPUTE
0499 07                 TPA                SAVE STATUS
049A D6 75              LDA B   QUADRT     GET QYADRANT INFO
049C 06                 TAP                RESTORE STATUS
049D 27 25              BEQ     SPECLT     IF Z, SPECIAL CASE
049F 56                 ROR B              CHECK QUADRANT
04A0 24 0D              BCC     TAN1       IF 1 OR 3 DO TAN
04A2 CE 00 45           LDX     #XPRIME    ELSE DO COTAN
04A5 BD 06 DE           JSR     MOVE2      MOVE TO XOP
04A8 CE 00 4C           LDX     #YPRIME
04AB 8D 28              BSR     TAN3       GO DIVIDE
04AD 20 02              BRA     TAN4
04AF 8D 1B     TAN1     BSR     TAN2       GO DIVIDE
04B1 CE 00 20  TAN4     LDX     #RSIGN     POINT TO RESULT
04B4 C6 50              LDA B   #TSBYTE    GET TAN SIGN BYTE
04B6 96 3A              LDA A   OVFL       CHECK OVERFLOW
04B8 27 C0              BEQ     TRGSGN     IF NO GO FIX SIGNS
04BA CE 03 38  INFIN    LDX     #INF       POINT TO  INFINITY
04BD 86 FF              LDA A   #$FF
04BF 97 3A              STA A   OVFL       SET OVERFLOW
04C1 7E 06 C7  TAN5     JMP     MOVE0      SET
04C4 56        SPECLT   ROR B              CHECK QUADRANT
04C5 25 F3              BCS     INFIN      IF 90 OR 270, THEN INFINITY
04C7 CE 03 31           LDX     #ZERO      ELSE 0
04CA 20 F5              BRA     TAN5
04CC CE 00 4C  TAN2     LDX     #YPRIME    POINT TO SORE
04CF BD 06 DE           JSR     MOVE2      MOVE TO XOP
04D2 CE 00 45           LDX     #XPRIME    POINT TO X STORE
04D5 BD 06 E5  TAN3     JSR     MOVE3      MOVE TO YOP
04D8 7E 01 94           JMP     FPDIV      GO DIVIDE
               *
               *
               ** TRIGN
               * THIS IS THE IMPLEMENTATION OF THE ALGORITHM
               * OF TRIGONOMETRIC CORDIC.
04DB CE 00 61  TRIGN    LDX     #ANGLE     POINT TO ANGLE STORE
04DE DF 6F              STX     OPX        SAVE PTR
04E0 BD 06 D5           JSR     MOVE1      MOVE ARGUMENT THERE
04E3 CE 03 4C           LDX     #TRIGX     POINT TO INIT CONST.
04E6 BD 06 EC           JSR     MOVE4      SET UP
04E9 CE 03 4C           LDX     #TRIGX
04EC BD 06 F3           JSR     MOVE5
04EF 4F        TRIG     CLR A
04F0 97 6B              STA A   YOPN       SET FOR Y ADDITION
04F2 97 6D              STA A   LSTSGN
04F4 43                 COM A
04F5 97 6A              STA A   XOPN        SET FOR X SUBTRACTION
04F7 97 6C              STA A   AOPN       SET FOR ANGLE SUBTRACTION
```

```
LOCN B1 B2 B3
04F9 86 01                LDA A  #01
04FB 97 71                STA A  ITER       SET ITERATION COUNTER
04FD CE 03 52             LDX    #ANGFAC-1  POINT TO ANGLE FACTOR
0500 DF 68                STX    AFACTX     STORE POINTER
0502 8D 1D                BSR    NEWAN1     COMPUTE NEW ANGLE
0504 BD 05 8F    TRIG1    JSR    NEXANG     GET TO NEXT FACTOR
0507 86 09                LDA A  #9         SET FOR 9 ITERATIONS
0509 36          TRIG2    PSH A             SAVE
050A 8D 2B                BSR    NEWOPN     SET NEW OPERATIONS
050C 8D 3D                BSR    NEWXY      CALCULATE NEW X AND Y
050E BD 05 1F             JSR    NEWANG     COMPUTE NEW ANGLE
0511 32                   PUL A             GET ITERATION
0512 4A                   DEC A             ONCE DONE
0513 26 F4                BNE    TRIG2      CHECK IF DONE
0515 96 71       TRIG4    LDA A  ITER       GET COUNT
0517 4C                   INC A
0518 97 71                STA A  ITER       BUMP ITERATION COUNT
051A 81 09                CMP A  #9         CHECK DONE ?
051C 26 E6                BNE    TRIG1      CHECK IF DONE
051E 39          TRIG5    RTS               DONE
                 *
                 *
                 ** NEWANG
                 * THIS ROUTINE CALCULATES THE NEW ANGLE BASED
                 * ON THE SIGN DETERMINANT FOR CORDIC.
051F DE 68       NEWANG   LDX    AFACTX     GET ANGLE POINTER
0521 BD 06 E5    NEWAN1   JSR    MOVE3      MOVE IT
0524 CE 00 61             LDX    #ANGLE     POINT TO ANGLE ACC.
0527 BD 06 DE             JSR    MOVE2      SET UP
052A 96 6C                LDA A  AOPN       GET ANGLE OPERATION (+ OR -)
052C 97 33                STA A  YSIGN      FIX UP THE SIGN ACCORDINGLY
052E BD 01 03             JSR    FPADD      GO ADD TOGETHER
0531 CE 00 61    NEW1     LDX    #ANGLE     POINT TO ANGLE STORE
0534 7E 06 D5             JMP    MOVE1      RESTORE NEW VALUE
                 *
                 *
                 ** NEWOPN
                 * THIS ROUTINE SELECTS NEW OPERATIONS FOR X,Y
                 * AND ANGLE BASED ON THE SIGN DETERMINANT.
                 * OPX HOLDS THE POINTER TO THE DETERMINANT.
0537 DE 6F       NEWOPN   LDX    OPX        GET OPERATION POINTER
0539 A6 00       NEWOP1   LDA A  0,X        GET NEW SIGN
053B 91 6D       NEWOP3   CMP A  LSTSGN     COMPARE WITH LAST SIGN
053D 27 0B                BEQ    NEWOP2     IF SAME, DON'T CHANGE OPER.
053F 97 6D                STA A  LSTSGN     SAVE FOR NEXT TIME
0541 73 00 6A    NEWOP4   COM    XOPN
0544 73 00 6B             COM    YOPN
0547 73 00 6C             COM    AOPN       CHANGE OPERATIONS
054A 39          NEWOP2   RTS
                 *
                 *
                 ** NEWXY
                 * THIS ROUTINE CALCULATES THE NEW X AND Y VALUES
                 * FOR THE CORDIC ALGORITHM BASED ON THE SIGN
                 * DETERMINANTS FOR THE INDIVIDUAL QUANTITIES.
```

```
LOCN B1 B2 B3
054B CE 00 45  NEWXY   LDX     #XPRIME    POINT TO X
054E BD 06 DE          JSR     MOVE2      MOVE TO XOP
0551 CE 00 4C          LDX     #YPRIME    POINT TO Y
0554 BD 06 E5          JSR     MOVE3      MOVE TO YOP
0557 96 39             LDA A   YEX        GET EXPONENT
0559 90 71             SUB A   ITER       SCALE FOR ITERATION
055B 97 39             STA A   YEX        PUT BACK
055D 96 6A             LDA A   XOPN       CHECK X OPERATION
055F 2A 03             BPL     NEWXY1
0561 73 00 33          COM     YSIGN      CHANGE SIGNS
0564 BD 01 03  NEWXY1  JSR     FPADD      GO ADD IN
0567 CE 00 45          LDX     #XPRIME
056A BD 06 E5          JSR     MOVE3      MOVE TO YOP
056D CE 00 4C          LDX     #YPRIME
0570 BD 06 DE          JSR     MOVE2      MOVE TO XOP
0573 CE 00 45          LDX     #XPRIME
0576 BD 06 D5          JSR     MOVE1      MOVE NEW VALUE
0579 96 39             LDA A   YEX        GET EXPONENT
057B 90 71             SUB A   ITER       SCALE FOR ITERATION
057D 97 39             STA A   YEX        PUT BACK
057F 96 6B             LDA A   YOPN       GET OPERATION
0581 2A 03             BPL     NEWXY2
0583 73 00 33          COM     YSIGN
0586 BD 01 03  NEWXY2  JSR     FPADD      GO COMPUTE NEW VALUE
0589 CE 00 4C          LDX     #YPRIME
058C 7E 06 D5          JMP     MOVE1      RESTORE NEW VALUE
               *
               *
               ** NEXANG
               * THIS SECTION MOVES THE POINTER TO THE NEXT
               * ANGLE FACTOR.
058F 86 06     NEXANG  LDA A   #6         LOAD OFFSET
0591 9B 69             ADD A   AFACTX+1   ADD IN
0593 97 69             STA A   AFACTX+1   SAVE BACK
0595 96 68             LDA A   AFACTX     GET MS BYTE
0597 89 00             ADC A   #0         ADD IN  CARRY
0599 97 68             STA A   AFACTX     STORE BACK
059B 39                RTS
               *
               *
               ** ARCCOS
               * ENTRY POINT FOR ARCCOSINE
059C 8D 21     ARCCOS  BSR     ARC        GO DO ARC FUNCTION
059E BD 06 E5          JSR     MOVE3      MOVE TO YOP
05A1 96 6E             LDA A   SIGN       GET SIGN
05A3 97 33             STA A   YSIGN      SET SIGN
05A5 CE 03 15          LDX     #NINETY
05A8 BD 06 DE          JSR     MOVE2      SET X=90
05AB BD 01 00          JSR     FPSUB      SUBTRACT Y FROM 90
05AE CE 00 20          LDX     #RSIGN
05B1 A6 00             LDA A   0,X        GET THE SIGN
05B3 36                PSH A              SAVE SIGN
05B4 7E 06 82          JMP     ATAN3      GO FIX
               *
               *
```

```
LOCN B1 B2 B3
                    ** ARCSIN
                    * ENTRY POINT FOR ARCSINE
05B7 8D 06          ARCSIN  BSR     ARC         GO DO ARC FUNCTION
05B9 96 6E                  LDA A   SIGN        GET SIGN
05BB 36                     PSH A               SAVE
05BC 7E 06 82               JMP     ATAN3       GO FIX
                    *
                    *
                    ** ARC
                    * INVERSE TRIGONOMETRIC CORDIC
05BF 96 2C          ARC     LDA A   XSIGN       GET SIGN
05C1 97 6E                  STA A   SIGN        SAVE SIGN
05C3 7F 00 2C               CLR     XSIGN       SET +
05C6 CE 03 2A               LDX     #ONE
05C9 BD 03 EC               JSR     EXTRCT      CHECK >1.0
05CC CE 00 20               LDX     #RSIGN      POINT TO REMAINDER
05CF 96 59                  LDA A   ZPRIME+6
05D1 81 01                  CMP A   #1          IF SO, ERROR
05D3 22 0C                  BHI     ARC5
05D5 96 54                  LDA A   ZPRIME+1    CHECK FACTOR
05D7 27 0F                  BEQ     ARC3        ZERO?
05D9 81 01                  CMP A   #1          CHECK FACTOR
05DB 22 04                  BHI     ARC5        TOO MANY?
05DD 96 21                  LDA A   RSIGN+1     CHECK REMAINDER
05DF 27 03                  BEQ     ARC4        IF 1.0, SPECIAL CASE
05E1 7E 08 F5       ARC5    JMP     ILLEGL
05E4 CE 03 15       ARC4    LDX     #NINETY
05E7 39                     RTS                 DONE
05E8 96 21          ARC3    LDA A   RSIGN+1     CHECK FOR 0.0
05EA 26 01                  BNE     ARC6
05EC 39                     RTS
05ED CE 00 53       ARC6    LDX     #ZPRIME
05F0 BD 06 D5               JSR     MOVE1       SAVE THE SCALED ARG.
05F3 CE 03 52               LDX     #ANGFAC-1
05F6 DF 68                  STX     AFACTX      SAVE NEW PTR
05F8 BD 06 FA               JSR     MOVE6       SET ANGLE = 45 DEG.
05FB CE 03 4C               LDX     #TRIGX
05FE BD 06 EC               JSR     MOVE4       SET X
0601 CE 03 4C               LDX     #TRIGX
0604 BD 06 F3               JSR     MOVE5       SET Y
0607 86 01                  LDA A   #01
0609 97 71                  STA A   ITER        SET UP ITERATION COUNTER
060B 8D 82          ARC1    BSR     NEXANG      GO GET NEXT ANGLE
060D 86 09                  LDA A   #9
060F 36             ARC2    PSH A               SET FOR 9 LOOPS
0610 CE 00 53       ARCCHK  LDX     #ZPRIME     POINT TO SCALED ARG.
0613 BD 06 DE               JSR     MOVE2
0616 CE 00 4C               LDX     #YPRIME
0619 BD 06 E5               JSR     MOVE3       SET UP FOR COMPARISON
061C BD 01 00               JSR     FPSUB       GO SUBTRACT
061F 5F                     CLR B
0620 96 45                  LDA A   XPRIME      CHECK SIGN OF X
0622 26 04                  BNE     ARCC3       IF MINUS, REVERSE
0624 96 20                  LDA A   RSIGN       GET COMPARISON INDICATOR
0626 2A 01                  BPL     ARCC2
```

```
LOCN B1 B2 B3
0628 53         ARCC3   COM B           CHANGE
0629 D7 6B      ARCC2   STA B   YOPN
062B D7 6C              STA B   AOPN
062D 53                 COM B
062E D7 6A              STA B   XOPN      FIX OPERATORS
0630 BD 05 4B   ARCC5   JSR     NEWXY     COMPUTE X & Y CHANGES
0633 BD 05 1F           JSR     NEWANG    COMPUTE NEW ANGLE
0636 32                 PUL A
0637 4A                 DEC A             DECREMENT COUNTER
0638 26 D5              BNE     ARC2      IF NOT DONE, DO AGAIN
063A 96 71              LDA A   ITER
063C 4C                 INC A             KICK ITERATION COUNT
063D 97 71              STA A   ITER
063F 81 09              CMP A   #9        CHECK IF DONE
0641 26 C8              BNE     ARC1
0643 CE 00 61           LDX     #ANGLE    POINT TO ANSWER
0646 39                 RTS               DONE
                *
                *
                ** ARCTAN
                * ENTRY POINT FOR ARCTANGENT
0647 96 2C      ARCTAN  LDA A   XSIGN
0649 36                 PSH A             SAVE SIGN
064A 7F 00 2C           CLR     XSIGN     SET SIGN = +
064D CE 00 2C           LDX     #XSIGN
0650 A6 01              LDA A   1,X       GET FIRST BYTE
0652 27 44              BEQ     ROUND     IF 0, SPECIAL CASE
0654 BD 06 F3           JSR     MOVE5     MOVE ARG. TO Y
0657 CE 03 2A           LDX     #ONE
065A BD 06 EC           JSR     MOVE4     SET X=1
065D CE 03 52           LDX     #ANGFAC-1
0660 DF 68              STX     AFACTX    STORE PTR
0662 BD 06 FA           JSR     MOVE6     SET ANGLE=45
0665 CE 00 4C           LDX     #YPRIME
0668 DF 6F              STX     OPX
066A 4F                 CLR A
066B 97 6C              STA A   AOPN
066D 97 6A              STA A   XOPN
066F 97 71              STA A   ITER
0671 97 6D              STA A   LSTSGN    SET UP CONSTANTS
0673 43                 COM A
0674 97 6B              STA A   YOPN      SET Y SUBTRACT
0676 BD 05 4B           JSR     NEWXY     GET INITIAL X AND Y
0679 7C 00 71           INC     ITER      SET COUNTER
067C BD 05 04           JSR     TRIG1     GO COMPUTE
067F CE 00 61           LDX     #ANGLE    POINT TO RESULT
0682 96 76      ATAN3   LDA A   MODE      CHECK MODE
0684 27 0F              BEQ     ATAN5     IF DEGREE, DONE
0686 BD 06 DE           JSR     MOVE2     PUT ANSWER IN XOP
0689 CE 03 45           LDX     #RTODEG
068C BD 06 E5           JSR     MOVE3     SET Y= CONV. CONST.
068F BD 01 94           JSR     FPDIV     GO SCALE
0692 CE 00 20           LDX     #RSIGN    POINT TO ANSWER
0695 20 01      ATAN5   BRA     ROUND
                *
```

```
LOCN B1 B2 B3
                  *
                  ** ROUND
                  * THIS IS THE ROUNDING PROCESSOR.  THE ROUNDING
                  * CONSTANT IS RCON.
0697 36           ROUND0  PSH A            SAVE THE SIGN
0698 A6 06        ROUND   LDA A  6,X       GET EXPONENT
069A 36                   PSH A            SAVE
069B BD 06 C7             JSR    MOVE0     MOVE TARGET TO AC
069E CE 03 3F     ROUND1  LDX    #RCON
06A1 8D 42                BSR    MOVE3     MOVE ROUNDING CONSTANT TO YOP
06A3 BD 01 CD             JSR    BCDADD    GO ADD IN ROUNDING
06A6 32                   PUL A            GET EXPONENT
06A7 97 26                STA A  RSIGN+6   INSTALL
06A9 32                   PUL A            GET SIGN
06AA 97 20                STA A  RSIGN     SET SIGN
06AC 96 24                LDA A  RSIGN+4
06AE 84 F0                AND A  #$F0      MASK OFF
06B0 97 24                STA A  RSIGN+4   STORE BACK
06B2 4F                   CLR A
06B3 97 25                STA A  RSIGN+5   ZERO OUT LAST BYTE
06B5 BD 01 4A             JSR    NORM      GO NORMALIZE
06B8 5F                   CLR B
06B9 96 26                LDA A  RSIGN+6   GET EXPONENT
06BB 81 63                CMP A  #$63      CHECK EXPONENT FOR >63
06BD 2E 04                BGT    ERROR
06BF 81 9C                CMP A  #$9C      CHECK <-63
06C1 2E 01                BGT    ANSOK
06C3 53           ERROR   COM B
06C4 D7 3A        ANSOK   STA B  OVFL      SET OVERFLOW
06C6 39                   RTS              DONE
                  *
                  *
                  ** THIS SECTION OF CODE PERFORMS ALL OF THE
                  * OPERAND TRANSPORTATION.
                  *
                  * MOVE X TO  RESULT
06C7 DF 41        MOVE0   STX    XTEMP2    SAVE SOURCE PTR.
06C9 CE 00 20             LDX    #RSIGN    POINT TO RESULT
06CC DF 3F        MOVE01  STX    XTEMP     SAVE DESTINATION PTR
06CE DE 41                LDX    XTEMP2
06D0 C6 07        MOVE02  LDA B  #7        SET FOR 7 BYTES
06D2 7E 02 62             JMP    XOPT01    MOVE IT
                  *
                  *   MOVE RESULT TO X
06D5 DF 3F        MOVE1   STX    XTEMP     SAVE DESTINATION PTR
06D7 CE 00 20             LDX    #RSIGN    POINT TO RESULT
06DA DF 41                STX    XTEMP2    SET SOURCE PTR
06DC 20 F2                BRA    MOVE02
                  *
                  *   MOVE X TO XOP
06DE DF 41        MOVE2   STX    XTEMP2    SAVE SOURCE PTR
06E0 CE 00 2C             LDX    #XSIGN    POINT TO DESTINATION
06E3 20 E7                BRA    MOVE01
                  *
                  *   MOVE X TO YOP
```

```
LOCN B1 B2 B3
06E5 DF 41     MOVE3    STX     XTEMP2     SAVE SOURCE PTR
06E7 CE 00 33           LDX     #YSIGN     POINT TO DESTINATION
06EA 20 E0              BRA     MOVE01
               *
               *   MOVE X TO XPRIME
06EC DF 41     MOVE4    STX     XTEMP2     SAVE SOURCE PTR
06EE CE 00 45           LDX     #XPRIME    POINT TO DESTINATION
06F1 20 D9              BRA     MOVE01     GO MOVE
               *
               *   MOVE X TO YPRIME
06F3 DF 41     MOVE5    STX     XTEMP2     SAVE SOURCE POINTER
06F5 CE 00 4C           LDX     #YPRIME    POINT TO DESTINATION
06F8 20 D2              BRA     MOVE01     MOVE IT
               *
               *   MOVE X TO ANGLE
06FA DF 41     MOVE6    STX     XTEMP2     SAVE SOURCE
06FC CE 00 61           LDX     #ANGLE     POINT
06FF 20 CB              BRA     MOVE01
               *
               *
               ** XTOY
               * THIS SECTION PERFORMS THE FUNCTION X**Y
               * OR X TO Y POWER
0701 CE 00 33  XTOY     LDX     #YSIGN     POINT TO POWER
0704 8D C1              BSR     MOVE0
0706 CE 00 5A  XTOY0    LDX     #TPRIME
0709 8D CA              BSR     MOVE1      SAVE IN TPRIME
070B CE 00 2C           LDX     #XSIGN     POINT TO ARG
070E A6 01              LDA A   1,X        GET MS BYTE
0710 26 03              BNE     XTOY3
0712 7E 07 C3           JMP     SINH1
0715 BD 08 36  XTOY3    JSR     NATLOG     TAKE LN(X)
0718 96 72              LDA A   NLEGAL     CHECK ILLEGAL OPER.
071A 27 01              BEQ     XTOY2
071C 39        XTOY1    RTS
071D 96 3A     XTOY2    LDA A   OVFL       CHECK OVERFLOW
071F 26 FB              BNE     XTOY1
0721 CE 00 2C           LDX     #XSIGN
0724 8D AF              BSR     MOVE1      MOVE RESULT TO X
0726 CE 00 5A           LDX     #TPRIME
0729 8D BA              BSR     MOVE3      MOVE POWER TO Y
072B BD 01 80           JSR     FPMULT     GO MULTIPLY
072E 96 3A              LDA A   OVFL       CHECK OVERFLOW
0730 26 EA              BNE     XTOY1
0732 CE 00 2C           LDX     #XSIGN
0735 8D 9E              BSR     MOVE1      MOVE RESULT BACK TO X
0737 20 26              BRA     EXP        GO EXPONENTIATE
               *
               *
               ** SQRT
               * THIS ROUTINE IMPLEMENTS  SQRT(X)
               * AS SPECIAL CASE OF X**Y
0739 CE 03 31  SQRT     LDX     #ZERO
073C 8D 89              BSR     MOVE0
073E 86 05              LDA A   #$05
```

```
LOCN B1 B2 B3
0740 97 21             STA A   RSIGN+1    SET RESULT TO 0.5
0742 20 C2             BRA     XTOYO
               *
               *
               ** INVERS
               * THIS ROUTINE IMPLEMENTS 1/X
0744 CE 00 2C  INVERS  LDX     #XSIGN     POINT TO ARG
0747 8D 9C             BSR     MOVE3      MOVE TO YOP
0749 CE 03 2A          LDX     #ONE       POINT TO ONE
074C 8D 90             BSR     MOVE2      SET XOP=1
074E 7E 01 94          JMP     FPDIV      GO DIVIDE
               *
               *
               ** ALOG10
               * ENTRY POINT FOR COMMON ANTILOG   (OR 10**X)
0751 CE 03 07  ALOG10  LDX     #LN10      POINT TO CONST
0754 8D 8F             BSR     MOVE3      SET Y=LN10
0756 BD 01 80          JSR     FPMULT     GO SCALE
0759 CE 00 2C          LDX     #XSIGN
075C BD 06 D5          JSR     MOVE1      MOVE SCALED ARGUMENT
               *
               *
               ** EXP
               * IMPLEMENTATION OF E**X  (EXPONENTIATION)
075F 96 32     EXP     LDA A   XSIGN+6    GET EXPONENT OF ARG
0761 81 03             CMP A   #3         CHECK FOR OVFL
0763 2F 03             BLE     EXP1       IF SO, OK
0765 7E 04 BA  EXP11   JMP     INFIN
0768 81 F9     EXP1    CMP A   #$F9       CHECK <-6
076A 2E 06             BGT     EXP2
076C CE 03 2A  EXP10   LDX     #ONE
076F 7E 06 C7  EXP12   JMP     MOVE0      SET ANSWER
0772 CE 03 07  EXP2    LDX     #LN10
0775 BD 03 EC          JSR     EXTRCT     REMOVE FACTORS OF LN10
0778 CE 00 2C          LDX     #XSIGN
077B BD 06 D5          JSR     MOVE1      MOVE REMAINDER TO XOP
077E BD 07 D9          JSR     HYP        GO DO HYPERBOLIC CORDIC
0781 CE 00 45          LDX     #XPRIME
0784 BD 06 DE          JSR     MOVE2      SET UP XOP
0787 CE 00 4C          LDX     #YPRIME
078A BD 06 E5          JSR     MOVE3      SET UP YOP
078D BD 01 03          JSR     FPADD      EXP=SINH+COSH
0790 D6 59             LDA B   ZPRIME+6   GET FACTOR
0792 C1 02             CMP B   #2         CHECK EXPONENT
0794 23 09             BLS     EXP21
0796 CE 03 31          LDX     #ZERO
0799 96 53             LDA A   ZPRIME     GET SIGN
079B 2B D2             BMI     EXP12      IF MINUS THEN 0
079D 20 C6             BRA     EXP11      ELSE INFINITY
079F 96 54     EXP21   LDA A   ZPRIME+1
07A1 56                ROR B
07A2 25 0C             BCS     EXPOK1
07A4 48                ASL A
07A5 16                TAB
07A6 48                ASL A
```

```
LOCN B1 B2 B3
07A7 48                ASL A
07A8 1B                ABA               MULTIPLY BY 10
07A9 D6 55             LDA B  ZPRIME+2  GET ONES
07AB 54                LSR B
07AC 54                LSR B
07AD 54                LSR B
07AE 54                LSR B            MOVE TO LS HALF
07AF 1B                ABA               MERGE
07B0 D6 53     EXPOK1  LDA B  ZPRIME    GET SIGN
07B2 27 01             BEQ    ADDON     IF + JUST ADD
07B4 40                NEG A            CHANGE SIGN
07B5 9B 26     ADDON   ADD A  RSIGN+6   ADD IN
07B7 97 26             STA A  RSIGN+6   PUT BACK
07B9 CE 00 20          LDX    #RSIGN    POINT TO RESULT
07BC 20 05             BRA    SINH1     GO ROUND
               *
               *
               *
               ** SINH
               * ENTRY POINT FOR HYPERBOLIC SINE
07BE 8D 19     SINH    BSR    HYP       CALCULATE
07C0 CE 00 4C          LDX    #YPRIME   POINT TO ANSWER
07C3 A6 00     SINH1   LDA A  0,X       GET SIGN
07C5 7E 06 97          JMP    ROUND0    GO ROUND
               *
               *
               ** COSH
               * ENTRY POINT FOR HYPERBOLIC COSINE
07C8 8D 0F     COSH    BSR    HYP       CALCULATE HYPERBOLIC
07CA CE 00 45          LDX    #XPRIME   POINT TO ANSWER
07CD 20 F4             BRA    SINH1     MOVE IT
               *
               *
               ** TANH
               * ENTRY POINT FOR HYPERBOLIC TANGENT
07CF 8D 08     TANH    BSR    HYP       GO CALCULATE
07D1 BD 04 CC          JSR    TAN2      TANH=SINH/COSH
07D4 CE 00 20          LDX    #RSIGN    POINT TO RESULT
07D7 20 EA             BRA    SINH1     GO ROUND
               *
               *
               ** HYP
               * THIS ROUTINE IMPLEMENTS HYPERBOLIC CORDIC.
07D9 CE 03 89  HYP     LDX    #HYPCON   POINT TO CONSTANT
07DC BD 06 EC          JSR    MOVE4     SET UP XPRIME
07DF CE 03 31          LDX    #ZERO
07E2 BD 06 F3          JSR    MOVE5     SET YPRIME=0
07E5 CE 00 2C          LDX    #XSIGN    POINT TO ARGUMENT
07E8 BD 06 FA          JSR    MOVE6     MOVE TO ANGLE
07EB CE 00 61          LDX    #ANGLE
07EE DF 6F             STX    OPX       SET OPERATION DETERMINANT
07F0 CE 03 8F          LDX    #HYPANG-1
07F3 DF 68             STX    AFACTX    SET ANGLE FACTOR POINTER
07F5 4F                CLR A
07F6 97 6D             STA A  LSTSGN    SET LAST SIGN
```

```
LOCN B1 B2 B3
07F8 97 6A              STA A  XOPN       SET FOR ADD ON X
07FA 97 6B              STA A  YOPN       SET FOR ADD ON Y
07FC 43                 COM A
07FD 97 6C              STA A  AOPN       SET FOR SUB. ON ANGLE
07FF 86 01     HYP0     LDA A  #1
0801 97 71              STA A  ITER       SET ITERATION COUNTER
0803 86 16     HYP1     LDA A  #22
0805 36        HYP2     PSH A
0806 BD 05 37           JSR    NEWOPN     SET NEW OPERATIONS
0809 BD 05 4B           JSR    NEWXY      CALCULATE NEW X AND Y
080C BD 05 1F           JSR    NEWANG     CALCULATE NEW ANGLE
080F 32                 PUL A
0810 4A                 DEC A             DECREMENT COUNTER
0811 26 F2              BNE    HYP2
0813 BD 05 8F           JSR    NEXANG     POINT TO NEXT FACTOR
0816 96 71              LDA A  ITER
0818 4C                 INC A             KICK ITERATION COUNT
0819 97 71              STA A  ITER
081B 81 0A              CMP A  #10        CHECK IF DONE
081D 26 E4              BNE    HYP1       LOOP
081F 39                 RTS               DONE
               *
               *
               ** LOG10
               * IMPLEMENTATION OF COMMON LOGARITHM (BASE 10)
0820 8D 14     LOG10    BSR    NATLOG     GO DO NATLOG
0822 CE 00 2C           LDX    #XSIGN
0825 BD 06 D5           JSR    MOVE1      MOVE RESULT TO X
0828 CE 03 07           LDX    #LN10
082B BD 06 E5           JSR    MOVE3      SET Y= LN10
082E BD 01 94           JSR    FPDIV      SCALE
0831 CE 00 20  LOG10A   LDX    #RSIGN     POINT TO RESULT
0834 20 8D              BRA    SINH1      GO ROUND
               *
               *
               ** NATLOG
               * IMPLEMENTATION OF NATURAL LOGARITHM (BASE E)
0836 CE 00 2C  NATLOG   LDX    #XSIGN
0839 BD 06 F3           JSR    MOVE5      SAVE IN YPRIME
083C CE 03 2A           LDX    #ONE
083F BD 06 E5           JSR    MOVE3      SET Y=1
0842 BD 01 00           JSR    FPSUB      SUBTRACT 1
0845 96 21              LDA A  RSIGN+1    GET MS BYTE
0847 26 06              BNE    NATL2
0849 CE 03 31           LDX    #ZERO      SET ZERO
084C 7E 07 C3           JMP    SINH1      GO FIX
084F CE 03 2A  NATL2    LDX    #ONE
0852 BD 06 DE           JSR    MOVE2      USE CLEAR TRICK
0855 96 52              LDA A  YPRIME+6   GET ARGS EXP.
0857 2A 04              BPL    SIGNOK     IF + NO TRICKS
0859 40                 NEG A             IF MINUS MAKE PLUS
085A 73 00 2C           COM    XSIGN      CHANGE SIGNS TO COMP.
085D C6 FF     SIGNOK   LDA B  #$FF       SET -1
085F 5C        SUBT     INC B             KICK COUNTER
0860 80 0A              SUB A  #10        TAKE OUT TEN
```

```
LOCN B1 B2 B3
0862 24 FB              BCC     SUBT       IF NO BORROW OK
0864 8B 0A              ADD A   #10        ELSE ADD BACK
0866 97 2D              STA A   XSIGN+1    STORE ONES
0868 5D                 TST B              CHECK IF HAVE TENS
0869 27 0C              BEQ     FACTOK     IF NOT,ALL DONE FOOLING
086B D7 2D              STA B   XSIGN+1    SAVE ONES
086D 48                 ASL A
086E 48                 ASL A
086F 48                 ASL A
0870 48                 ASL A              GET TO MS HALF
0871 97 2E              STA A   XSIGN+2    SAVE ONES
0873 86 02              LDA A   #02        SET EXP=2
0875 97 32              STA A   XEX        STORE IT
0877 CE 03 07   FACTOK  LDX     #LN10
087A BD 06 E5           JSR     MOVE3      SET Y=LN10
087D BD 01 80           JSR     FPMULT     GO SCALE
0880 CE 00 53           LDX     #ZPRIME
0883 BD 06 D5           JSR     MOVE1      SAVE SCALING FACTOR
0886 96 4D              LDA A   YPRIME+1   CHECK FOR ZERO
0888 27 6B              BEQ     ILLEGL     IF 0 ERROR
088A CE 00 4C           LDX     #YPRIME
088D A6 00      INVHYP  LDA A   0,X        GET SIGN
088F 2B 64              BMI     ILLEGL     IF NEGATIVE, ILLEGAL
0891 6F 06              CLR     6,X        SET EXP = 0
0893 BD 06 DE           JSR     MOVE2      RETRIEVE ARGUMENT
0896 CE 03 2A           LDX     #ONE
0899 BD 06 E5           JSR     MOVE3      SET Y=1
089C BD 01 03           JSR     FPADD
089F CE 00 45           LDX     #XPRIME
08A2 BD 06 D5           JSR     MOVE1      SET X=ARG+1
08A5 CE 00 4C           LDX     #YPRIME
08A8 BD 06 DE           JSR     MOVE2      SET XOP=ARG
08AB CE 03 2A           LDX     #ONE
08AE BD 06 E5           JSR     MOVE3      SET YOP=1
08B1 BD 01 00           JSR     FPSUB
08B4 CE 00 4C           LDX     #YPRIME
08B7 DF 6F              STX     OPX        SET DECISION PTR
08B9 BD 06 D5           JSR     MOVE1      SET Y=ARG-1
08BC CE 03 31           LDX     #ZERO
08BF BD 06 FA           JSR     MOVE6      SET ANG=0
08C2 4F                 CLR A
08C3 97 6C              STA A   AOPN
08C5 97 6D              STA A   LSTSGN
08C7 43                 COM A
08C8 97 6B              STA A   YOPN       SET UP OPERATIONS
08CA 97 6A              STA A   XOPN
08CC CE 03 8F           LDX     #HYPANG-1
08CF DF 68              STX     AFACTX     SET CONSTANT PTR.
08D1 BD 07 FF           JSR     HYPO       GO COMPUTE
08D4 CE 00 61           LDX     #ANGLE
08D7 BD 06 D5           JSR     MOVE1      SET X=RESULT
08DA CE 03 23           LDX     #TWO
08DD BD 06 E5           JSR     MOVE3      SET Y= 2
08E0 BD 01 80           JSR     FPMULT     GO SCALE
08E3 CE 00 2C           LDX     #XSIGN
```

```
LOCN B1 B2 B3
08E6 BD 06 D5          JSR    MOVE1      GET RESULT
08E9 CE 00 53          LDX    #ZPRIME
08EC BD 06 E5          JSR    MOVE3      SET Y= ADDON
08EF BD 01 03          JSR    FPADD      GO ADD IT ON
08F2 7E 08 31          JMP    LOG10A     GO ROUND
                *
                *
                ** THIS ROUTINE SETS ILLEGAL OPERATION INDICATOR
08F5 86 FF      ILLEGL LDA A  #$FF
08F7 97 72             STA A  NLEGAL     SET NOT LEGAL
08F9 39                RTS               DONE


SYMBOL TABLE:

ADDON   07B5   AFACTX 0068   ALLOK  03EB   ALLZER 03E1   ALOG10 0751
ANGFAC 0353   ANGLE  0061   ANSOK  06C4   AOPN   006C   ARC    05BF
ARCCHK 0610   ARCCOS 059C   ARCC2  0629   ARCC3  0628   ARCC5  0630
ARCSIN 05B7   ARCTAN 0647   ARC1   060B   ARC2   060F   ARC3   05E8
ARC4   05E4   ARC5   05E1   ARC6   05ED   ARGSTR 0960   ATAN3  0682
ATAN5  0695   BCDADD 01CD   BYTE   E055   COS    0459   COSH   07C8
C360   031C   C4     0073   DEGREE 043C   E      030E   ERROR  06C3
EXCEPT 0458   EXP    075F   EXPOK1 0780   EXP1   0768   EXP10  076C
EXP11  0765   EXP12  076F   EXP2   0772   EXP21  079F   EXTRCT 03EC
FACTOK 0877   FPADD  0103   FPDIV  0194   FPMULT 0180   FPSUB  0100
GOTBIT 0482   GOTIT  03D5   HYP    07D9   HYPANG 0390   HYPCON 0389
HYPO   07FF   HYP1   0803   HYP2   0805   ILLEGL 08F5   INEEE  E1AC
INF    0338   INFIN  04BA   INTDON 03E5   INTEGE 03C6   INVERS 0744
INVHYP 088D   ITER   0071   LN10   0307   LOAD   090E   LOG10  0820
LOG10A 0831   LSTSGN 006D   MODE   0076   MOVE0  06C7   MOVE01 06CC
MOVE02 06D0   MOVE1  06D5   MOVE2  06DF   MOVE3  06E5   MOVE4  06EC
MOVE5  06F3   MOVE6  06FA   NATLOG 0836   NATL2  084F   NEWANG 051F
NEWAN1 0521   NEWOPN 0537   NEWOP1 0539   NEWOP2 054A   NEWOP3 053B
NEWOP4 0541   NEWXY  054B   NEWXY1 0564   NEWXY2 0586   NEW1   0531
NEXANG 058F   NEXDIG 03DD   NINETY 0315   NLEGAL 0072   NORM   014A
ONE    032A   OPS    096F   OPX    006F   OUTHL  E067   OUTHR  E06B
OUTS   E0CC   OVFL   003A   PBYTE  0958   PDATA1 E07E   PI     0300
PRT    0935   QUADRT 0075   RCON   033F   ROUND  0698   ROUND0 0697
ROUND1 069E   RSIGN  0020   RTODEG 0345   SIGN   006E   SIGNOK 085D
SIN    0460   SINE   097D   SINH   07BE   SINH1  07C3   SIN0   0461
SIN1   0478   SPECL  048B   SPECLT 04C4   SQRT   0739   SSBYTE 0030
START  0900   SUBT   085F   TABOFF 092E   TAN    0496   TANH   07CF
TAN1   04AF   TAN2   04CC   TAN3   04D5   TAN4   04B1   TAN5   04C1
TESTIT 03CB   TESTSG 047C   TPRIME 005A   TRGDON 0488   TRGFAC 0422
TRGSGN 047A   TRIG   04EF   TRIGN  04DB   TRIGX  034C   TRIG1  0504
TRIG2  0509   TRIG4  0515   TRIGS  051E   TSBYTE 0050   TWO    0323
XEX    0032   XOP    002D   XOPN   006A   XOPTO1 0262   XPRIME 0045
XSIGN  002C   XTEMP  003F   XTEMP2 0041   XTOY   0701   XTOY0  0706
XTOY1  071C   XTOY2  071D   XTOY3  0715   YEX    0039   YOP    0034
YOPN   006B   YPRIME 004C   YSIGN  0033   ZCHK   0275   ZERO   0331
ZPRIME 0053
```

```
S10400760085
S113030000031415926501000230258509010002DD
S11303107182818301000900000000020003600073
S113032000000300020000000000010001000000C2
S113033001000000000000000009999999996300E8
S1130340000000050000057295779502000675838C
S11303506159000450000000020571059314010561
S11303607293869800057295760 4FF0572957789D5
S1130370FE0572957795FD0572957795FC05729546
S11303807795FB0572957795FA00011181379301F2
S11303900100335348000100003333FF0100000023
S11303A033FE0100000000FD0100000000FC01001C
S11303B0000000FB0100000000FA0100000000F949
S11303C00100000000F8A606C6050881002F0608F3
S11303D05A800220F6260AA60084F0A700085A27AD
S11303E0046F0020F8A6002A026F0039DF73BD06EF
S11303F0E5BD0194CE00208DCDCE0053BD06D5CEF3
S1130400002CBD06ECCE002CBD06D5DE73BD06E582
S1130410BD0180CE0033BD06D5CE0045BD06DE7ECF
S11304200100962C976E7F002C9676270FCE0345FD
S1130430BD06E5BD0180CE002CBD06D5CE031C8DC6
S1130440ABCE002CBD06D5CE03158DA09654977562
S113045096212704BD04DB433986017F002C20014B
S11304604F368DBE0733DB75C403D77506271CCE04
S1130470004C562403CE0045C63096752704584ACE
S113048020FA966E5D2A01437E0697CE03315624E8
S1130490E7CE032A20E2BD042207D675062725569 7
S11304A0240DCE0045BD06DECE004C8D2820028DE5
S11304B01BCE0020C650963A27C0CE033886FF973D
S11304C03A7E06C75625F3CE033120F5CE004CBD47
S11304D006DECE0045BD06E57E0194CE0061DF6FE9
S11304E0BD06D5CE034CBD06ECCE034CBD06F34F82
S11304F0976B976D43976A976C86019771CE0352F9
S1130500DF688D1DBD058F8609368D2B8D3DBD059C
S11305101F324A26F496714C9771810926E639DE1A
S113052068BD06E5CE0061BD06DE966C9733BD015D
S113053003CE00617E06D5DE6FA600916D270B9772
S11305406D73006A73006B73006C39CE0045BD0691
S1130550DECE004CBD06E5963990719739966A2A2D
S113056003730033BD0103CE0045BD06E5CE004C48
S1130570BD06DECE0045BD06D596399071973996F5
S11305806B2A03730033BD0103CE004C7E06D5866F
S1130590069B699769966889009768398D21BD061D
S11305A0E5966E9733CE0315BD06DEBD0100CE0081
S11305B020A600367E06828D06966E367E068296CC
S11305C02C976E7F002CCE032ABD03ECCE00209620
S11305D0598101220C9654270F8101220496212768
S11305E0037E08F5CE0315399621260139CE005332
S11305F0BD06D5CE0352DF68BD06FACE034CBD0658
S1130600ECCE034CBD06F3860197718D82860936C4
S1130610CE0053BD06DECE004CBD06E5BD01005F35
S11306209645260496202A0153D76BD76C53D76A74
S1130630BD054BBD051F324A26D596714C97718175
S11306400926C8CE006139962C367F002CCE002CAA
S1130650A6012744BD06F3CE032ABD06ECCE035201
S1130660DF68BD06FACE004CDF6F4F976C976A9730
S113067071976D43976BBD054B7C0071BD0504CE2E
S113068000619676270FBD06DECE0345BD06E5BDA7
S11306900194CE0020200136A60636BD06C7CE033F
S11306A03F8D42BD01CD329726329720962484F0A7
```

```
S11306B097244F9725BD014A5F962681632E0481B6
S11306C09C2E0153D73A39DF41CE0020DF3FDE4173
S11306D0C6077E0262DF3FCE0020DF4120F2DF4109
S11306E0CE002C20E7DF41CE003320E0DF41CE00F6
S11306F04520D9DF41CE004C20D2DF41CE0061201D
S1130700CBCE00338DC1CE005A8DCACE002CA601AB
S113071026037E07C3BD083696722270139963A260A
S1130720FBCE002C8DAFCE005A8DBABD0180963A17
S113073026EACE002C8D9E2026CE03318D89860597
S113074097212C2CE002C8D9CCE032A8D907E0151
S113075094CE03078D8FBD0180CE002CBD06D596A7
S11307603281032F037E04BA81F92E06CE032A7E3A
S113077006C7CE0307BD03ECCE002CBD06D5BD07CE
S1130780D9CE0045BD06DECE004CBD06E5BD010355
S1130790D659C1022309CE033196532BD220C696D3
S11307A05456250C481648481BD65554545454541BCB
S11307B0D6532701409B269726CE002020058D196D
S11307C0CE004CA6007E06978D0FCE004520F48DFA
S11307D008BD04CCCE002020EACE0389BD06ECCEB1
S11307E00331BD06F3CE002CBD06FACE0061DF6FE7
S11307F0CE038FDF684F976D976A976B43976C862C
S1130800019771861636BD0537BD054BBD051F32F0
S11308104A26F2BD058F96714C9771810A26E439F8
S11308208D14CE002CBD06D5CE0307BD06E5BD0153
S113083094CE0020208DCE002CBD06F3CE032ABD1D
S113084006E5BD0100962126 06CE03317E07C3CE00
S1130850032ABD06DE96522A044073002CC6FF5CB0
S11308600800A24FB8B0A972D5D270CD72D48484816
S113087048972E86029732CE0307BD06E5BD018058
S1130880CE0053BD06D5964D276BCE004CA6002B4B
S11308900646F06BD06DECE032ABD06E5BD0103CEA8
S11308A00045BD06D5CE004CBD06DECE032ABD06EE
S11308B0E5BD0100CE004CDF6FBD06D5CE0331BDD2
S11308C006FA4F976C976D43976B976ACE038FDF49
S11308D068BD07FFCE0061BD06D5CE0323BD06E586
S11308E0BD0180CE002CBD06D5CE0053BD06E5BDAE
S10D08F001037E083186FF97723978
```

```
 LOCN B1 B2 B3
                     *
                     *
                     * EXTERNAL ROUTINES (MIKBUG)
       E067          OUTHL     EQU     $E067
       E06B          OUTHR     EQU     $E06B
       E0CC          OUTS      EQU     $E0CC
       E07E          PDATA1    EQU     $E07E
       E1AC          INEEE     EQU     $E1AC
       E055          BYTE      EQU     $E055
   *
   *
   *
   *
   *
   *  THE FOLLOWING IS A VERY CRUDE DRIVER USED FOR TESTING
   * THE SCIENTIFIC PACKAGE.  NOTE THAT THIS CODE IS NOT
   * A PART OF THE PACKAGE.  THE PROGRAM IS ENTERED AT 0900
   * AND THE PROMPT "ARGUMENT" IS PRINTED.  AT THIS TIME YOU
   * ARE TO ENTER THE ARGUMENT IN FULL INTERNAL FLOATING POINT
   * FORMAT (SIGN,MANTISSA,EXPONENT).  AFTER THE EXPONENT IS
   * ENTERED THE PROMPT "OPERATION" IS PRINTED AND YOU TYPE
   * ONE OF THE OPERATION CODES FOUND IN THE TABLE BELOW (0-?)
   * NOTE THAT NO CHECKING FOR INPUT NOT IN RANGE IS DONE
   * (WE SAID IT WAS CRUDE).  AFTER TIME FOR THE CALCULATION
   * HAS ELAPSED THE ANSWER WILL BE PRINTED IN FULL INTERNAL
   * FLOATING POINT FORMAT NORMALIZED FORM. THE OVFL AND NLEGAL
   * BYTES ARE ALSO PRINTED OUT FOR YOUR INFORMATION.  THE
   * FUNCTION X^Y MUST BE TREATED SPECIALLY BECAUSE NO PROVISIONS
   * ARE MADE FOR ENTERING THE SECOND OPERAND (THE VALUE
   * FOR Y MUST BE INSTALLED IN YSIGN-YEX BY THE USER BEFORE
   * CALLING THE FUNCTION.
   *   THIS DRIVER USES MANY ROUTINES FROM MIKBUG (MOT. TRADEMARK)
   * IT IS NOT INTENDED FOR SALE BUT IS INCLUDED HERE FOR THE
   * CONVENIENCE OF THOSE WHO MAY FIND USE FOR IT.
   *
                               ORG     $900
0900 8E A0 7F   START          LDS     #$A07F
0903 CE 09 60                  LDX     #ARGSTR
0906 BD E0 7E                  JSR     PDATA1
0909 CE 00 2C                  LDX     #XSIGN
090C C6 07                     LDA B   #7
090E BD E0 55   LOAD           JSR     BYTE
0911 A7 00                     STA A   0,X
0913 08                        INX
0914 8C 00 33                  CPX     #XSIGN+7
0917 26 F5                     BNE     LOAD
0919 CE 09 6F                  LDX     #OPS
091C BD E0 7E                  JSR     PDATA1
091F CE 09 7D                  LDX     #SINE
0922 BD E1 AC                  JSR     INEEE
0925 80 30                     SUB A   #$30
0927 48                        ASL A
0928 B7 09 2F                  STA A   TABOFF+1
092B BD E0 CC                  JSR     OUTS
```

```
LOCN B1 B2 B3
092E EE 00       TABOFF  LDX      0,X
0930 AD 00               JSR      0,X
0932 CE 00 20            LDX      #RSIGN
0935 A6 00       PRT     LDA A    0,X
0937 BD 09 58            JSR      PBYTE
093A 08                  INX
093B 8C 00 27            CPX      #RSIGN+7
093E 26 F5               BNE      PRT
0940 BD E0 CC            JSR      OUTS
0943 96 3A               LDA A    OVFL
0945 BD 09 58            JSR      PBYTE
0948 BD E0 CC            JSR      OUTS
094B 96 72               LDA A    NLEGAL
094D BD 09 58            JSR      PBYTE
0950 7F 00 72            CLR      NLEGAL
0953 7F 00 3A            CLR      OVFL
0956 20 A8               BRA      START
0958 36          PBYTE   PSH A
0959 BD E0 67            JSR      OUTHL
095C 32                  PUL A
095D 7E E0 6B            JMP      OUTHR
0960 0D 0A       ARGSTR  FDB      $0D0A,$0000
0964 41 00               FCC      ;ARGUMENT? ;
096E 04                  FCB      4
096F 20          OPS     FCC      ;  OPERATION? ;
097C 04                  FCB      4
                 *
                 *
                 ** OPERATION CODE TABLE
097D 04 60       SINE    FDB      SIN          (0)
097F 04 59               FDB      COS          (1)
0981 04 96               FDB      TAN          (2)
0983 05 B7               FDB      ARCSIN       (3)
0985 05 9C               FDB      ARCCOS       (4)
0987 06 47               FDB      ARCTAN       (5)
0989 07 5F               FDB      EXP          (6)
098B 07 BE               FDB      SINH         (7)
098D 07 C8               FDB      COSH         (8)
098F 07 CF               FDB      TANH         (9)
0991 08 36               FDB      NATLOG       (:)
0993 08 20               FDB      LOG10        (;)
0995 07 51               FDB      ALOG10       (<)
0997 07 44               FDB      INVERS       (=)
0999 07 39               FDB      SQRT         (>)
099B 07 01               FDB      XTOY         (?)
                 *
S11309008EA07FCE0960BDF07ECE002CC607BDE080
S113091055A700088C003326F5CE096FBDE07ECEC6
S1130920097DBDE1AC803048B7092FBDE0CCEE00B5
S1130930AD00CE0020A600BD0958088C002726F57E
S1130940BDE0CC963ABD0958BDE0CC9672BD0958BD
S11309507F00727F003A20A836BDE067327EE06BEC
S11309600D0A000041524755D454E543F20042086
S1130970204F5045524154494F4E3F200404600 4D7
S113098059049605B7059C0647075F07BE07C807C5
S1100990CF0836082007510744073907013 6
S9030000FC
```