TSC 6800 Floating Point Package

SL68-4

TECHNICAL SYSTEMS CONSULTANTS

```
*  TSC FLOATING POINT PACKAGE
*  VER 2.3
*
*  COPYRIGHT (C) 1976 BY
*    TECHNICAL SYSTEMS CONSULTANTS
*    BOX 2574  W. LAFAYETTE IN.   47906
*
*      THE TSC FLOATING POINT PACKAGE PROVIDES
* THE BASIC ARITHMETIC FUNCTIONS ADD, SUBTRACT,
* MULTIPLY, AND DIVIDE.  THE NORMALIZED
* FORM OF THE X AND Y OPERANDS IS A MIXED
* FORMAT.  THE MANTISSA IS SIGN PLUS MAGNI-
* TUDE PACKED BCD NOTATION.  THIS IMPLIES
* THAT THE SIGN BYTE (XSIGN OR YSIGN) IS
* EITHER ALL ZEROS FOR POSITIVE OR ALL ONES
* FOR NEGATIVE.  THE MANTISSA ITSELF (XOP OR
* YOP) IS VIEWED AS BEING A 9 DIGIT FRAC-
* TIONAL NUMBER WITH A ZERO TO THE LEFT OF
* THE DECIMAL POINT (IN THE UPPER HALF OF THE
* MOST SIGNIFICANT BYTE OF XOP OR YOP).  THIS
* IS DONE TO SIMPLIFY THE ARITHMETIC AND NOR-
* MALIZATION OPERATIONS.  PACKED BCD IMPLIES
* THAT THERE ARE 2 BCD DIGITS PER BYTE.
*      THE EXPONENT IS AN 8 BIT 2'S COMPLEMENT
* NUMBER WITH A RANGE OF +128 TO -127.  HOW-
* EVER, BECAUSE OF THE WAY IN WHICH THE
* MULTIPLY, DIVIDE, AND NORMALIZE OPERATIONS
* ARE DONE THE PRACTICAL RANGE IS SLIGHTLY
* SMALLER.  ONE CAN TOTALLY AVOID DEALING
* WITH THIS PROBLEM BY RESTRICTING THE EX-
* PONENT RANGE TO +99 AND -99.  THIS SHOULD
* NOT PROVE TO BE AN UNREASONABLE CONSTRAINT.
*      THE RESULT OF ALL ARITHMETIC OPERATIONS
* IS RETURNED IN NORMALIZED FORM WITH THE SIGN
* IN RSIGN, THE MANTISSA IN FPAC, AND THE
* EXPONENT IN ACEXP.
*      EXAMPLES OF THE NORMALIZED FORM ARE GIVEN
* BELOW:
*       NUMBER     SIGN    MANTISSA   EXPONENT
*     +100          00    0100000000   03
*     -1163.12      FF    0116312000   04
*     +0.125        00    0125000000   00
*     +0.000213     00    0213000000   FD
*     -0.000213     FF    0213000000   FD
*
*      THIS PACKAGE CONTAINS SUBOUTINES ONLY.
* AN EXTERNAL DRIVER PROGRAM MUST BE USED TO
* EXERCISE THEM.  THE ROUTINE ADDRESSES
* ARE GIVEN BELOW:
*    OPERATION     NAME       ADDRESS
*     ADD          FPADD        0103
*     SUBTRACT     FPSUB        0100
*     MULTIPLY     FPMUL        0180
```

```
                  *    DIVIDE      FPDIV       0194
                  *
                  *
                  *
                  *
                  * STORAGE SPACE
0020                      ORG      $20
0020              RSIGN   RMB      1          RESULT SIGN BYTE
0021              FPAC    RMB      5          FLOATING POINT ACCUMULATOR
0026              ACEXP   RMB      1          ACCUMULATOR EXPONENT
0027              FPMQ    RMB      5          FLOATING POINT MQ REGISTER
002C              XSIGN   RMB      1          X SIGN BYTE
002D              XOP     RMB      5          X OPERAND MANTISSA
0032              XEX     RMB      1          X OPERAND EXPONENT
0033              YSIGN   RMB      1          Y SIGN BYTE
0034              YOP     RMB      5          Y OPERAND MANTISSA
0039              YEX     RMB      1          Y OPERAND EXPONENT
003A              OVFL    RMB      1
003B              ATEMP   RMB      1
003C              ATEMP2  RMB      1
003D              BTEMP   RMB      1
003E              BTEMP2  RMB      1
003F              XTEMP   RMB      2          TEMPORARY X STORAGE
0041              XTEMP2  RMB      2
                  *
0005              AC      EQU      05         OPERAND BYTE COUNT
                  *
0100                      ORG      $100
                  *
                  *
                  *FPSUB
                  * FLOATING POINT SUBTRACT
                  * SUBTRACTS YOP*YEX FROM XOP*XEX
0100 73 00 33     FPSUB   COM      YSIGN      CHANGE SIGN
                  *  GO INTO FPADD
                  *
                  *FPADD
                  * FLOATING POINT ADD ROUTINE
                  * ADDS YOP*YEX TO XOP*XEX
0103 8D 71        FPADD   BSR      SETSIN     SET SIGN HOLDER
0105 BD 02 CB             JSR      EXPADJ     ADJUST EXPONENTS
0108 CE 00 21             LDX      #FPAC
010B BD 02 5B             JSR      XOPTOX     MOVE XOP TO FPAC
010E CE 00 21             LDX      #FPAC
0111 BD 02 75             JSR      ZCHK       CHECK XOP FOR =0
0114 27 1D                BEQ      FPAD01
0116 CE 00 34             LDX      #YOP
0119 BD 02 75             JSR      ZCHK       CHECK YOP FOR =0
011C 27 19                BEQ      FPADD1
011E 96 20                LDA A    RSIGN
0120 2A 15                BPL      FPADD1     USE EITHER SIGN
0122 BD 01 C7             JSR      BCDSUB     SUBTRACT
0125 25 10                BCS      FPADD1     USE X SIGN
0127 CE 00 34             LDX      #YOP       POINT TO YOP
```

```
012A BD 02 AB            JSR    LTC       RECOMPLEMENT YOP
012D CE 06 2D            LDX    #YOP
0130 BD 02 AB            JSR    LTC       COMPLEMENT XOP
0133 96 33      FPAD01   LDA A  YSIGN     USE Y SIGN
0135 20 02               BRA    FPADD2
0137 96 2C      FPADD1   LDA A  XSIGN
0139 97 20      FPADD2   STA A  RSIGN
013B CE 00 21   FPAD21   LDX    #FPAC
013E BD 02 5D            JSR    XOPTOX    MOVE XOP TO FPAC
0141 BD 01 CD            JSR    BCDADD    ADD
0144 CE 00 27            LDX    #FPMQ
0147 BD 02 80            JSR    CLROP     CLEAR THE MQ
                * GO INTO NORM
                *
                *NORM
                * NORMALIZE FLOATING POINT RESULTS
                * MQ MUST CONTAIN VALID DATA
014A CE 00 21   NORM     LDX    #FPAC
014D BD 02 75            JSR    ZCHK      CHECK FOR ZERO
0150 26 07               BNE    NORM2
0152 7F 00 26            CLR    ACEXP
0155 7F 00 20            CLR    RSIGN
0158 39                  RTS
0159 CE 00 21   NORM2    LDX    #FPAC
015C A6 00               LDA A  0,X
015E 27 0C               BEQ    NORM3
0160 84 F0               AND A  #$F0
0162 27 1B               BEQ    SETSI1
0164 7C 00 26            INC    ACEXP
0167 29 5A               BVS    FPDIV3    CHECK FOR OVERFLOW
0169 7E 02 2F            JMP    EL4RR
016C BD 02 45   NORM3    JSR    EL4RL
016F 7A 00 26            DEC    ACEXP
0172 29 4F               BVS    FPDIV3    CHECK FOR OVERFLOW
0174 20 E3               BRA    NORM2
                *
                *SETSIN
                * CALCULATE XSIGN.XOR.YSIGN
                * STORE IN RSIGN
0176 96 2C      SETSIN   LDA A  XSIGN
0178 98 33               EOR A  YSIGN
017A 97 20               STA A  RSIGN
017C 7F 00 3A            CLR    OVFL
017F 39         SETSI1   RTS
                *
                *
                *FPMUL
                * FLOATING POINT MULTIPLY ROUTINE
                * MULTIPLIES XOP*XEX BY YOP*YEX
                * TRUNCATES PRODUCT TO BC*2-1 BCD DIGITS
0180 8D F4      FPMUL    BSR    SETSIN    STORE OPERAND SIGNS
0182 96 32               LDA A  XEX
0184 9B 39               ADD A  YEX       CALCULATE EXPONENT
0186 29 3B               BVS    FPDIV3    CHECK FOR OVERFLOW
```

```
0188 97 26               STA A   ACEXP      SAVE EXPONENT
018A CE 00 27            LDX     #FPMQ
018D BD 02 5B            JSR     XOPTOX     MOVE XOP TO MQ
0190 8D 50               BSR     BCDMUL     MULTIPLY
0192 20 B6               BRA     NORM
                *
                *FPDIV
                *FLOATING POINT DIVIDE ROUTINE
                *DIVIDES XOP*XEX BY YOP*YEX
                *TRUNCATES THE REMAINDER
0194 8D E0      FPDIV    BSR     SETSIN     STORE SIGNS
0196 96 32               LDA A   XEX
0198 90 39               SUB A   YEX        CALCULATE EXPONENT
019A 29 27               BVS     FPDIV3     CHECK FOR OVERFLOW
019C CE 00 21            LDX     #FPAC
019F BD 02 5B            JSR     XOPTOX     MOVE XOP TO THE AC
01A2 CE 00 27            LDX     #FPMQ
01A5 BD 02 80            JSR     CLROP      CLEAR THE MQ
01A8 BD 02 2F            JSR     EL4RR      SHIFT ACMQ TO AVOID OVFL
01AB 4C                  INC A              COMPENSATE EXPONENT
01AC 29 15               BVS     FPDIV3     CHECK FOR OVERFLOW
01AE 97 26               STA A   ACEXP      STORE EXPONENT
01B0 8D 4B               BSR     BCDDIV     DIVIDE
01B2 25 0F               BCS     FPDIV3     CHECK FOR OVERFLOW
01B4 C6 05               LDA B   #BC
01B6 CE 00 21            LDX     #FPAC
01B9 A6 06      FPDIV1   LDA A   BC+1,X
01BB A7 00               STA A   0,X        MOVE QUOTIENT TO THE AC
01BD 08                  INX
01BE 5A                  DEC B
01BF 26 F8               BNE     FPDIV1
01C1 20 87               BRA     NORM
01C3 73 00 3A   FPDIV3   COM     OVFL
01C6 39                  RTS
                *
                *BCDSUB
                * SUBTRACTS YOP FROM FPAC
01C7 CE 00 34   BCDSUB   LDX     #YOP
01CA BD 02 AB            JSR     LTC        TAKE TENS COMP
                *    GO INTO BCDADD
                *
                *BCDADD
                * ADDS YOP TO FPAC
                * USES ZERO INITIAL CARRY
01CD 8D 59      BCDADD   BSR     SAVREG     SAVE REGISTER CONTENTS
01CF CE 00 21            LDX     #FPAC
01D2 0C                  CLC
01D3 C6 05               LDA B   #BC        SET COUNTER
01D5 A6 04      BCDAD1   LDA A   BC-1,X
01D7 A9 17               ADC A   BC*4+3,X
01D9 19                  DAA                ADJUST FOR BCD
01DA A7 04               STA A   BC-1,X
01DC 09                  DEX
01DD 5A                  DEC B              ONCE DONE
```

```
01DE 26 F5              BNE     BCDAD1
01E0 20 40              BRA     RSTREG     RESTORE REGISTERS
                *
                *BCDMUL
                * MULTIPLIES FPMQ BY YOP
                * ANSWER IN FPAC AND FPMQ
01E2 CE 00 21   BCDMUL  LDX     #FPAC
01E5 BD 02 80           JSR     CLROP      CLEAR FPAC
01E8 C6 09              LDA B   #BC*2-1    SET CTR
01EA 96 2B      BCDMU1  LDA A   FPMQ+BC-1  GET LS BYTE
01EC 84 0F              AND A   #$0F       MASK OFF LS BCD
01EE 27 07              BEQ     BCDMU3
01F0 8D DB      BCDMU2  BSR     BCDADD     ADD IN OPERAND
01F2 7A 00 2B           DEC     FPMQ+BC-1
01F5 20 F3              BRA     BCDMU1
01F7 8D 36      BCDMU3  BSR     EL4RR      SHIFT ACMQ 1 BCD RIGHT
01F9 5A                 DEC B
01FA 26 EE              BNE     BCDMU1
01FC 39                 RTS
                *
                *BCDDIV
                * DIVIDES FPAC AND FPMQ BY YOP
                * QUOTIENT RETURNED IN FPMQ, REMAINDER IN FPAC
                * CARRY RETURNED SET ON OVERFLOW
01FD CE 00 34   BCDDIV  LDX     #YOP
0200 8D 73              BSR     ZCHK
0202 26 02              BNE     BCDD15     CHECK FOR DIV BY 0
0204 0D                 SEC
0205 39         BCDDI1  RTS
0206 C6 0A      BCDD15  LDA B   #BC*2
0208 8D BD              BSR     BCDSUB     SUBTRACT OPERAND
020A 24 0C              BCC     BCDDI3     CHECK FOR OVFL
020C 39                 RTS
020D 8D 36      BCDD16  BSR     EL4RL      SHIFT ACMQ 1 BCD LEFT
020F 8D BC      BCDDI2  BSR     BCDADD     SUBTRACT OPERAND
0211 24 05              BCC     BCDDI3     IF NO CARRY, TOO SMALL
0213 7C 00 2B           INC     FPMQ+BC-1  TALLY ONE
0216 20 F7              BRA     BCDDI2
0218 8D AD      BCDDI3  BSR     BCDSUB     COMPENSATE REMAINDER
021A BD 02 AB           JSR     LTC        RECOMPLEMENT
021D 5A                 DEC B              DEC LOOP CTR
021E 26 ED              BNE     BCDD16
0220 0C                 CLC
0221 39                 RTS
                *
                *RSTREG
                * RESTORE REGISTERS X,A,B
0222 96 3B      RSTREG  LDA A   ATEMP
0224 D6 3D              LDA B   BTEMP
0226 DE 3F              LDX     XTEMP
                *       GO INTO SAVREG
                *
                *SAVREG
                * SAVE REGISTERS X,A,B
```

```
0228 97 3B       SAVREG  STA A  ATEMP
022A D7 3C               STA B  BTEMP
022C DF 3F               STX    XTEMP
022E 39                  RTS
                 *
                 *EL4RR
                 * EXTRA LONG 4 ROTATE RIGHT
                 * ROTATES ACMQ RIGHT ONE BCD
022F 8D F7       EL4RR   BSR    SAVREG    SAVE PTRS
0231 86 04               LDA A  #04
0233 CE 00 21    EL4RR1  LDX    #FPAC     POINT TO AC
0236 C6 01               LDA B  #01
0238 0C                  CLC
0239 8D 4E               BSR    LRR       SHIFT AC
023B CE 00 27            LDX    #FPMQ     POINT TO MQ
023E 8D 49               BSR    LRR       SHIFT MQ
0240 4A                  DEC A
0241 26 F0               BNE    EL4RR1
0243 20 DD               BRA    RSTREG
                 *
                 *EL4RL
                 * EXTRA LONG 4 ROTATE LEFT
                 * ROTATES ACMQ LEFT ONE BCD DIGIT
0245 8D E1       EL4RL   BSR    SAVREG
0247 86 04               LDA A  #04       SET SHIFT COUNT
0249 CE 00 27    EL4RL1  LDX    #FPMQ
024C 0C                  CLC
024D C6 01               LDA B  #01
024F 8D 49               BSR    LRL       SHIFT
0251 CE 00 21            LDX    #FPAC
0254 8D 44               BSR    LRL       SHIFT AC
0256 4A                  DEC A
0257 26 F0               BNE    EL4RL1
0259 20 C7       EL4RL2  BRA    RSTREG    RESTORE POINTERS
                 *
                 *XOPTOX
                 * MOVE XOP TO LOCN POINTED TO BY X
                 * MODIFIES X
025B 8D CB       XOPTOX  BSR    SAVREG
025D CE 00 2D            LDX    #XOP      POINT TO XOP
0260 C6 05               LDA B  #BC       SET CTR
0262 A6 00       XOPTO1  LDA A  0,X
0264 08                  INX
0265 DF 41               STX    XTEMP2    STORE PTR
0267 DE 3F               LDX    XTEMP     LOAD DEST PTR
0269 A7 00               STA A  0,X
026B 08                  INX
026C DF 3F               STX    XTEMP
026E DE 41               LDX    XTEMP2
0270 5A                  DEC B
0271 26 EF               BNE    XOPTO1
0273 20 AD               BRA    RSTREG
                 *
                 *ZCHK
```

```
                  * CHECK A 5 BYTE BCD FOR =0
                  * OPERAND POINTED TO BY X
                  *  MODIFIES B,X
 0275 C6 05       ZCHK     LDA B   #BC
 0277 6D 00       ZCHK1    TST     0,X
 0279 26 04                BNE     ZCHK2
 027B 08                   INX
 027C 5A                   DEC B
 027D 26 F8                BNE     ZCHK1
 027F 39          ZCHK2    RTS
                  *
                  *CLROP
                  * CLEAR 5 BYTE OPERAND POINTED TO BY X
                  * MODIFIES B,X
 0280 C6 05       CLROP    LDA B   #BC
 0282 6F 00       CLROP1   CLR     0,X
 0284 08                   INX
 0285 5A                   DEC B
 0286 26 FA                BNE     CLROP1
 0288 39                   RTS
                  *
                  *LRR
                  * LONG ROTATE RIGHT WITH CARRY
                  * X POINTS TO THE MS BYTE
                  * B CONTAINS AMOUNT OF SHIFT
 0289 8D 65       LRR      BSR     SVRG2
 028B DE 41       LRR1     LDX     XTEMP2
 028D 86 05                LDA A   #BC
 028F 66 00       LRR2     ROR     0,X
 0291 08                   INX
 0292 4A                   DEC A
 0293 26 FA                BNE     LRR2
 0295 5A                   DEC B
 0296 26 F3                BNE     LRR1
 0298 20 5C       LRR3     BRA     RSRG2
                  *
                  *LRL
                  * LONG ROTATE LEFT WITH CARRY
                  * X POINTS TO THE MS BYTE
                  * B CONTAINS THE AMOUNT OF SHIFT
 029A 8D 54       LRL      BSR     SVRG2
 029C DE 41       LRL1     LDX     XTEMP2
 029E 86 05                LDA A   #BC
 02A0 69 04       LRL2     ROL     BC-1,X
 02A2 09                   DEX
 02A3 4A                   DEC A
 02A4 26 FA                BNE     LRL2
 02A6 5A                   DEC B
 02A7 26 F3                BNE     LRL1
 02A9 20 4B                BRA     RSRG2
                  *
                  *LTC
                  * LONG TENS COMPLEMENT OF OPERAND
                  * POINTED TO BY X
```

```
02AB 8D 43      LTC      BSR     SVRG2
02AD C6 05               LDA B   #BC
02AF 86 99      LTC1     LDA A   #$99
02B1 A0 04               SUB A   BC-1,X
02B3 A7 04               STA A   BC-1,X
02B5 09                  DEX
02B6 5A                  DEC B
02B7 26 F6               BNE     LTC1
02B9 0D                  SEC
02BA C6 05               LDA B   #BC
02BC DE 41               LDX     XTEMP2
02BE 86 00      LTC2     LDA A   #00
02C0 A9 04               ADC A   BC-1,X
02C2 19                  DAA
02C3 A7 04               STA A   BC-1,X
02C5 09                  DEX
02C6 5A                  DEC B
02C7 26 F5               BNE     LTC2
02C9 20 2B      LTC4     BRA     RSRG2
                *
                *EXPADJ
                * ADJUSTS EXPONENTS FOR ADD AND SUBTRACT
                * OPERATES ON XOP AND YOP
                * MODIFIES A,B,X
02CB 96 32      EXPADJ   LDA A   XEX          LOAD X EXPONENT
02CD 91 39               CMP A   YEX          COMPARE WITH Y EXP
02CF 27 1C               BEQ     EXP3         EXPONENTS SAME?
02D1 2E 07               BGT     EXP1         XEX>YEX?
02D3 CE 00 2D            LDX     #XOP         POINT TO XOP
02D6 96 39               LDA A   YEX          GET YEX
02D8 20 03               BRA     EXP2
02DA CE 00 34   EXP1     LDX     #YOP         POINT TO YOP
02DD C6 04      EXP2     LDA B   #04
02DF 8D A8               BSR     LRR
02E1 E6 00               LDA B   0,X
02E3 C4 0F               AND B   #$0F         MASK OFF GOOD BCD
02E5 E7 00               STA B   0,X          SHIFTED 1 BCD RIGHT
02E7 6C 05               INC     BC,X         INCREMENT EXPONENT
02E9 A1 05               CMP A   BC,X         SAME YET?
02EB 26 F0               BNE     EXP2         IF NOT  DO AGAIN
02ED 97 26      EXP3     STA A   ACEXP        STORE NEW EXPONENT
02EF 39                  RTS                  DONE
                *
                *SVRG2
                * LEVEL2 REGISTER SAVE
02F0 DF 41      SVRG2    STX     XTEMP2
02F2 97 3C               STA A   ATEMP2
02F4 D7 3E               STA B   BTEMP2
                *  GO INTO RSRG2
                *
                *RSRG2
                * LEVEL2 REGISTER RESTORE
02F6 DE 41      RSRG2    LDX     XTEMP2
02F8 96 3C               LDA A   ATEMP2
```

```
 02FA D6 3E                 LDA B  BTEMP2
 02FC 39                    RTS
                     *
                     *
                            END

   SYMBOL TABLE:

ACEXP   0026    ATEMP   003B    ATEMP2 003C    BC       0005    BCDAD1 01D5
BCDADD  01CD    BCDD15  C206    BCDD16 020D    BCDDI1  0205    BCDDI2 020F
BCDDI3  0218    BCDDIV  01FD    BCDMU1 01EA    BCDMU2  01F0    BCDMU3 01F7
BCDMUL  01E2    BCDSUB  01C7    BTEMP   003D   BTEMP2  003E    CLROP   0280
CLROP1  0282    EL4RL   0245    EL4RL1 0249    EL4RL2  0259    EL4RR   022F
EL4RR1  0233    EXP1    02DA    EXP2    02DD   EXP3     02ED    EXPADJ 02CB
FPAC    0021    FPAD01  0133    FPAD21 013B    FPADD    0103    FPADD1 0137
FPADD2  0139    FPDIV   0194    FPDIV1 01B9    FPDIV3  01C3    FPMQ    0027
FPMUL   0180    FPSUB   0100    LRL     029A   LRL1     029C    LRL2    02A0
LRR     0289    LRR1    028E    LRR2    028F   LRR3     0298    LTC     02AB
LTC1    02AF    LTC2    02BE    LTC4    02C9   NORM     014A    NORM2   0159
NORM3   016C    OVFL    003A    RSIGN   0020   RSRG2    02F6    RSTREG 0222
SAVREG  0228    SETSI1  017F    SETSIN 0176    SVRG2    02F0    XEX     0032
XOP     002D    XOPT01  0262    XOPTOX 025B    XSIGN    002C    XTEMP   003F
XTEMP2  0041    YEX     0039    YOP     0034   YSIGN    0033    ZCHK    0275
ZCHK1   0277    ZCHK2   027F

S11301007300338D71BD02CBCE0021BD025BCE00E6
S113011021BD0275271DCE0034BD02752719962016
S11301202A158D01C72510CE0034BD02ABCE002D6B
S1130130BD02AB96332002962C9720CE0021BD023F
S11301405BBD01CDCE0027BD0280CE0021BD02756E
S113015026077F00267F002039CE0021A600270C29
S113016084F0271B7C0026295A7E022FBD02457A83
S11301700026294F20E3962C983397207F003A39A4
S11301808DF496329B39293B9726CE0027BD025B1E
S11301908D5020B68DE0963290392927CE0021BDAE
S11301A0025BCE0027BD0280BD022F4C2915972685
S11301B08D4B250FC605CE0021A606A700085A269A
S11301C0F8208773003A39CE0034BD02AB8D59CE86
S11301D000210CC605A604A91719A704095A26F577
S11301E02040CE0021BD0280C609962B840F27072C
S11301F08DD87A002B20F38D365A26EE39CE00346F
S11302008D7326020D39C60A8DBD240C398D368DA9
S1130210BC24057C002B20F78DADBD02AB5A26ED26
S11302200C39963BD63DDE3F973BD73DDF3F398DBA
S1130230F78604CE0021C6010C8D4ECE00278D49D1
S11302404A26F020DD8DE18604CE00270CC6018D00
S113025049CE00218D444A26F020C78DCBCE002DF7
S1130260C605A60008DF41DE3FA70008DF3FDE41E8
S11302705A26EF20ADC6056D002604085A26F83923
S1130280C6056F00085A26FA398D65DE4186056673
S113029000084A26FA5A26F3205C8D54DE4186056E
S11302A06904094A26FA5A26F3204B8D43C605866B
S11302B099A004A704095A26F60DC605DE41860056
S11302C0A90419A704095A26F5202B963291392737
S11302D01C2E07CE002D96392003CE0034C6048D83
S11302E0A8E600C40FE7006C05A10526F09726399F
S11002F0DF41973CD73EDE41963CD63E39B7
S9030000FC
```

```
* TSC FLOATING POINT PACKAGE DRIVER
*
*
*  COPYRIGHT (C) 1976 BY
*     TECHNICAL SYSTEMS CONSULTANTS
*     BOX 2574  W. LAFAYETTE  IN.  47906
*
*     THE TSC FLOATING POINT PACKAGE DRIVER, WHEN
* USED IN CONJUNCTION WITH THE TSC FLOATING POINT
* PACKAGE, IMPLEMENTS A BASIC FOUR-FUNCTION
* SCIENTIFIC NOTATION CALCULATOR.  THIS PROGRAM
* ACCEPTS INPUT FROM THE KEYBOARD, IN A FORM
* TO BE DESCRIBED LATER, INITIATES THE CALCULATION
* AND THEN OUTPUTS THE RESULT.
*     THE USER IS PROMPTED WITH THE SYMBOL  >
* AT WHICH POINT THE FIRST OPERAND IS TYPED.  THE
* OPERANDS ARE SUBJECT TO FORMAT RESTRICTIONS
* AS NOTED BELOW.  DIRECTLY FOLLOWING THE FIRST
* OPERAND THE USER TYPES THE OPERATOR, EITHER A
*   +,-,*, OR /   FOR ADD, SUBTRACT, MULTIPLY, OR
* DIVIDE, RESPECTIVELY.  DIRECTLY FOLLOWING THE
* OPERATOR, THE USER TYPES THE SECOND OPERAND,
* SUBJECT TO THE SAME RESTRICTIONS AS THE FIRST.
* NEXT A CARRIAGE RETURN IS TYPED TO INITIATE
* THE CALCULATION AND THEN THE ANSWER IS TYPED
* OUT AND THE USER IS PROMPTED FOR THE NEXT
* CALCULATION.
*     THE RESTRICTIONS ON THE FORMAT OF THE
* OPERANDS ARE AS FOLLOWS:
*  1)  THE OPERAND MUST BEGIN WITH A PLUS,
*    A MINUS, A DECIMAL POINT (PERIOD), OR ANY
*    DECIMAL DIGIT.
*  2)  THE DECIMAL POINT, IF IT APPEARS, MAY
*    BE ANY WHERE IN THE NUMBER AFTER THE SIGN
*    (IF ANY) AND BEFORE THE EXPONENT (IF ANY).
*  3)  THE EXPONENT, INDICATED BY THE LETTER
*    E, MAY BE PRECEEDED BY A PLUS OR MINUS SIGN
*    AND IS LIMITED TO TWO DIGITS.
*
*  THE CALCULATOR TRUNCATES ALL DIGITS IN EXCESS
* OF 9 SIGNIFICANT DIGITS.
*     SOME POSSIBLE FORMS ARE SHOWN BELOW:
*   >12*1.3
*   >.001-6
*   >-12+3E2
*   >+5.6E-21/-21E+00
*   >123456.789+.987654321
*   >+1.2--3.1E-1
*   >4*-5
*
* DEPARTURE FROM THE FORMAT RESTRICTIONS WILL
* CAUSE A SYNTAX ERROR MESSAGE TO BE PRINTED.
*     OPERATIONS RESULTING IN ARITHMETIC OVER-
```

```
                         * FLOW OR UNDERFLOW WILL CAUSE AN OVERFLOW
                         * MESSAGE TO BE PRINTED.
                         *     THE STARTING ADDRESS OF THIS PROGRAM
                         * IS 0300.
                         *
                         * MIKBUG  ROUTINES
                         *   (MIKBUG IS A REGISTERED
                         *     TRADEMARK OF MOTOROLA INC. )
E07E                     PDATA1  EQU    $E07E
E1AC                     INEEE   EQU    $E1AC
E1D1                     OUTEEE  EQU    $E1D1
E067                     OUTHL   EQU    $E067
E06B                     OUTHR   EQU    $E06B
                         *
                         * STORAGE
0050                             ORG    $0050
0050                     INBUF   RMB    6
0056                     INEXP   RMB    1
0057                     SIGDIG  RMB    1
0058                     DECFLG  RMB    1
0059                     EXPNEG  RMB    1
005A                     EXP     RMB    1
005B                     SYNTAX  RMB    1
005C                     OPER    RMB    1
003F                     XTEMP   EQU    $003F
0041                     XTEMP2  EQU    $0041
003D                     CTR     EQU    $003D
003E                     TOGGLE  EQU    $003E
002C                     XSIGN   EQU    $002C
0033                     YSIGN   EQU    $0033
0026                     ACEXP   EQU    $0026
0020                     RSIGN   EQU    $0020
003A                     OVFL    EQU    $003A
                         *
                         *
                         *FLOATING POINT PACKAGE ROUTINES
0103                     FPADD   EQU    $0103
0100                     FPSUB   EQU    $0100
0180                     FPMUL   EQU    $0180
0194                     FPDIV   EQU    $0194
                         *
                         *
                         *
A048                             ORG    $A048
A048 03 00                       FDB    BEG
0300                             ORG    $0300
0300 8E A0 42            BEG     LDS    #$A042    INITIALIZE SP
0303 CE 04 DB            START   LDX    #PROM
0306 BD E0 7E                    JSR    PDATA1
0309 4F                          CLR A
030A 97 5B                       STA A  SYNTAX    CLEAR SYNTAX ERROR
030C 97 5C                       STA A  OPER      CLEAR OPERATOR FLAG
030E BD 03 EE                    JSR    INPUT     FILL THE INPUT BUFFER
0311 96 5B                       LDA A  SYNTAX
```

```
0313 26 3C                BNE    SYNERR    CHECK FOR SYNTAX ERROR
0315 CE 00 2C             LDX    #XSIGN
0318 BD 03 D3             JSR    BUFTOX    TRANSFER INPUT TO XOP
031B BD 03 EE  NXTOP      JSR    INPUT     FILL BUFFER AGAIN
031E 96 5B                LDA A  SYNTAX
0320 26 2F                BNE    SYNERR    CHECK FOR SYNTAX ERROR
0322 CE 00 33             LDX    #YSIGN
0325 BD 03 D3             JSR    BUFTOX    TRANSFER TO YOP
0328 96 5C                LDA A  OPER      GET OPERATOR
032A 4A                   DEC A
032B 27 15                BEQ    ADDOP     IS IT AN ADD REQUEST
032D 4A                   DEC A
032E 27 0D                BEQ    SUBOP     IS IT A SUBTRACT REQ.
0330 4A                   DEC A
0331 27 05                BEQ    MULOP     IS IT A MULT.  REQUEST
0333 BD 01 94             JSR    FPDIV     ASSUME IT IS A DIVIDE
0336 20 0D                BRA    PRINT
0338 BD 01 80  MULOP      JSR    FPMUL     GO MULTIPLY
033B 20 08                BRA    PRINT
033D BD 01 00  SUBOP      JSR    FPSUB     GO SUBTRACT
0340 20 03                BRA    PRINT
0342 BD 01 03  ADDOP      JSR    FPADD     GO ADD
0345 96 3A     PRINT      LDA A  OVFL
0347 27 0D                BEQ    NOVFL     TEST FOR OVERFLOW
0349 CE 04 E2  OV         LDX    #OVER
034C BD E0 7E  PRTMES     JSR    PDATA1    GIVE HIM MESSAGE
034F 20 B2                BRA    START     DO AGAIN
0351 CE 04 EF  SYNERR     LDX    #SYNT
0354 20 F6                BRA    PRTMES    PRINT MESSAGE
0356 96 21     NOVFL      LDA A  RSIGN+1   GET FIRST DIGIT
0358 27 03                BEQ    OVCHK     CHECK FOR =0
035A 7A 00 26             DEC    ACEXP     ADJUST FOR OUTPUT
035D 96 26     OVCHK      LDA A  ACEXP
035F 81 63                CMP A  #99
0361 2E E6                BGT    OV        TEST FOR EXP OVERFLOW
0363 81 9D                CMP A  #$9D
0365 2D E2                BLT    OV        TEST FOR EXP UNDERFLOW
0367 CE 04 FA             LDX    #EQUAL
036A BD E0 7E             JSR    PDATA1    PRINT CRLF =
036D 96 20                LDA A  RSIGN     GET SIGN
036F 27 05                BEQ    POS       IS IT POSITIVE
0371 86 2D                LDA A  #'-
0373 BD E1 D1             JSR    OUTEEE    PRINT A MINUS
0376 96 21     POS        LDA A  RSIGN+1   GET BYTE
0378 BD E0 6B             JSR    OUTHR     PRINT LS BCD
037B C6 08                LDA B  #8        SET FOR COUNTING OFF
037D CE 00 25             LDX    #ACEXP-1  POINT TO LAST BYTE
0380 A6 00     CNTOFF     LDA A  0,X       GET THE BYTE
0382 85 0F                BIT A  #$0F
0384 26 09                BNE    GOTCNT    CHECK FOR LS ZERO
0386 5A                   DEC B            COUNT OFF DIGIT
0387 85 F0                BIT A  #$F0
0389 26 04                BNE    GOTCNT    CHECK FOR MS ZERO
038B 09                   DEX              POINT TO NEXT
```

```
038C 5A                    DEC B              COUNT OFF ONE DIGIT
038D 26 F1                 BNE    CNTOFF      IF NOT 8 DO AGAIN
038F CE 00 22    GOTCNT    LDX    #RSIGN+2    POINT TO SEC. BYTE
0392 5D                    TST B              CHECK FOR ZERO
0393 27 16                 BEQ    PRTEXP      IF SO, GO PRINT EXP.
0395 86 2E                 LDA A  #'.
0397 BD E1 D1              JSR    OUTEEE      PRINT DECIMAL POINT
039A A6 00       PRTLOP    LDA A  0,X         GET NEXT CHAR
039C BD E0 67              JSR    OUTHL       PRINT MS BCD
039F 5A                    DEC B              CHECK IF DONE
03A0 27 09                 BEQ    PRTEXP      IF SO GO PRINT EXP.
03A2 A6 00                 LDA A  0,X         GET BYTE AGAIN
03A4 BD E0 6B              JSR    OUTHR       PRINT LS BCD
03A7 08                    INX
03A8 5A                    DEC B              ONE BYTE DONE
03A9 26 EF                 BNE    PRTLOP
03AB D6 26       PRTEXP    LDA B  ACEXP
03AD 27 21                 BEQ    NOPRT       CHECK FOR EXP =0
03AF 86 45                 LDA A  #'E
03B1 BD E1 D1              JSR    OUTEEE      PRINT AN E
03B4 86 2B                 LDA A  #'+         GET ASCII FOR +
03B6 5D                    TST B              CHECK THE SIGN
03B7 2A 03                 BPL    PRTEXS      TEST SIGN
03B9 50                    NEG B              COMPLEMENT THE EXP.
03BA 86 2D                 LDA A  #'-
03BC BD E1 D1    PRTEXS    JSR    OUTEEE      PRINT EXPONENT SIGN
03BF 4F                    CLR A              CONVERT TO BCD AND PRINT
03C0 C0 0A       SUBT      SUB B  #10         SUBTRACT 10
03C2 25 03                 BCS    TOOMAN      SHOULDN'T SUBTRACT?
03C4 4C                    INC A              COUNT ONCE
03C5 20 F9                 BRA    SUBT
03C7 BD E0 6B    TOOMAN    JSR    OUTHR       PRINT MS DIGIT
03CA 86 0A                 LDA A  #10
03CC 1B                    ABA                COMPENSATE REMAINDER
03CD BD E0 6B              JSR    OUTHR       PRINT LS DIGIT
03D0 7E 03 03    NOPRT     JMP    START
                 *
                 *BUFTOX
                 * MOVE INPUT BUFFER CONTENTS TO X
03D3 DF 3F       BUFTOX    STX    XTEMP       SAVE X
03D5 CE 00 50              LDX    #INBUF
03D8 A6 00       BUF1      LDA A  0,X         GET CHAR OF BUFFER
03DA 08                    INX
03DB 8C 00 58              CPX    #INEXP+2    DONE YET?
03DE 27 0D                 BEQ    DONE
03E0 DF 41                 STX    XTEMP2
03E2 DE 3F                 LDX    XTEMP
03E4 A7 00                 STA A  0,X
03E6 08                    INX
03E7 DF 3F                 STX    XTEMP
03E9 DE 41                 LDX    XTEMP2
03EB 20 EB                 BRA    BUF1
03ED 39          DONE      RTS
                 *
```

```
                    *INPUT
                    * FILL THE INPUT BUFFER AND SET FLAGS
03EE CE 00 5A       INPUT   LDX     #EXP
03F1 6F 00          STUF    CLR     0,X         CLEAR THE BUFFER
03F3 09                     DEX
03F4 8C 00 4F               CPX     #INBUF-1
03F7 26 F8                  BNE     STUF
03F9 08                     INX
03FA 7F 00 3D               CLR     CTR         CLEAR FULL FLAG
03FD C6 FF                  LDA B   #$FF
03FF D7 3E                  STA B   TOGGLE      SET BYTE TOGGLE
0401 BD 04 D3       INCH    JSR     INCHAR      GET A CHAR
0404 81 2B                  CMP A   #'+
0406 27 11                  BEQ     INNEXT      IGNORE PLUS SIGN
0408 81 2D                  CMP A   #'-
040A 26 04                  BNE     NOTNEG      IF NOT MINUS PROCEED
040C 63 00                  COM     0,X         SET SIGN INDICATOR
040E 20 09                  BRA     INNEXT      GET NEXT CHAR
0410 81 2E          NOTNEG  CMP A   #'.
0412 27 03                  BEQ     ISPT        CHECK FOR DEC. POINT
0414 08             NOTPT   INX                 POINT NEXT BYTE
0415 20 06                  BRA     CRCHK       GO CHECK FOR CR
0417 97 58          ISPT    STA A   DECFLG      SET DECIMAL FLAG
0419 08             INNEXT  INX
041A BD 04 D3       GETIN   JSR     INCHAR      GET CHAR
041D 81 0D          CRCHK   CMP A   #$D
041F 27 71                  BEQ     REL         CHECK FOR CR
0421 80 30                  SUB A   #'0         REMOVE ASCII BIAS
0423 27 08                  BEQ     GOTZER      CHECK FOR ZERO INPUT
0425 2B 3A                  BMI     NOTYET      CHECK FOR <0
0427 81 09                  CMP A   #'9-'0
0429 22 36                  BHI     NOTYET      CHECK FOR >9
042B 97 57                  STA A   SIGDIG      SET SIGNIFICANT FLAG
042D D6 3D          GOTZER  LDA B   CTR
042F 26 E9                  BNE     GETIN       CHECK FOR BUFF. FULL
0431 D6 57                  LDA B   SIGDIG      HAD SIG. DIGITS?
0433 26 09                  BNE     TSTNXT
0435 D6 58                  LDA B   DECFLG      HAD DECIMAL PT?
0437 27 E1                  BEQ     GETIN       IF NOT & NOT SIG.
0439 7A 00 56               DEC     INEXP       IF SO BACK UP EXP.
043C 20 DC                  BRA     GETIN
043E D6 58          TSTNXT  LDA B   DECFLG      HAD DECIMAL PT?
0440 26 03                  BNE     STORIT      IF SO EXP. OK
0442 7C 00 56               INC     INEXP       KICK EXPONENT
0445 D6 3E          STORIT  LDA B   TOGGLE      CHECK FOR WHICH DIGIT
0447 26 04                  BNE     LOHALF
0449 48                     ASL A
044A 48                     ASL A
044B 48                     ASL A
044C 48                     ASL A               GET TO TOP HALF
044D AA 00          LOHALF  ORA A   0,X         MERGE
044F A7 00                  STA A   0,X         RE-STORE IT
0451 73 00 3E               COM     TOGGLE      SET FOR NEXT DIGIT
0454 26 01                  BNE     NOTNXT      CHECK FOR NEXT BYTE
```

```
0456 08                      INX               POINT TO NEXT BYTE
0457 8C 00 56   NOTNXT       CPX      #INEXP   CHECK FOR END OF BUFF
045A 26 BE                   BNE      GETIN    IF NOT GET MORE
045C 73 00 3D                COM      CTR      SET BUFFER END FLG
045F 20 B9                   BRA      GETIN    GET NEXT CHAR
0461 8B 30      NOTYET       ADD A    #'0      RESTORE ASCII
0463 81 45      FULL         CMP A    #'E
0465 27 2D                   BEQ      EXPIN    CHECK FOR EXP IND.
0467 C6 01                   LDA B    #1       SET OPER FLAG
0469 81 2B                   CMP A    #'+
046B 27 1F                   BEQ      GOTOP    CHECK FOR ADD OPER.
046D 5C                      INC B
046E 81 2D                   CMP A    #'-
0470 27 1A                   BEQ      GOTOP    CHECK FOR SUB.  OPER.
0472 5C                      INC B
0473 81 2A                   CMP A    #'*
0475 27 15                   BEQ      GOTOP    CHECK FOR MUL.  OPER.
0477 5C                      INC B
0478 81 2F                   CMP A    #'/
047A 27 10                   BEQ      GOTOP    CHECK FOR DIV.  OPER.
047C 81 2E                   CMP A    #'.      CHECK FOR DEC.  PT
047E 26 08                   BNE      SYNERF
0480 D6 58                   LDA B    DECFLG   CHECK FOR ALREADY DEC.  PT.
0482 26 04                   BNE      SYNERF
0484 97 58                   STA A    DECFLG   FLAG A DEC.  PT.
0486 20 92                   BRA      GETIN
0488 97 5B      SYNERF       STA A    SYNTAX   FLAG A SYNTAX ERROR
048A 20 8E                   BRA      GETIN    GET MORE CHARS.
048C 96 5C      GOTOP        LDA A    OPER     CHECK FOR ALREADY OPER.
048E 26 F8                   BNE      SYNERF   IF SO, FLAG AN ERROR
0490 D7 5C                   STA B    OPER     SET OPER FLG
0492 20 27      REL          BRA      GOTDIG
0494 8D 3D      EXPIN        BSR      INCHAR
0496 81 2B                   CMP A    #'+
0498 27 FA                   BEQ      EXPIN    IGNORE PLUS
049A 81 2D                   CMP A    #'-
049C 26 05                   BNE      CHKNXT
049E 73 00 59                COM      EXPNEG   SET EXPONENT SIGN
04A1 8D 30      EXINP        BSR      INCHAR   GET A CHAR
04A3 80 30      CHKNXT       SUB A    #'0
04A5 2B 04                   BMI      SYNEXP   CHECK FOR <0
04A7 81 09                   CMP A    #9
04A9 23 05                   BLS      EXPOK    CHECK FOR >9
04AB 8B 30      SYNEXP       ADD A    #'0      RESTORE ASCII
04AD 7E 04 1D                JMP      CRCHK    GO CHECK FOR CR
04B0 D6 5A      EXPOK        LDA B    EXP
04B2 58                      ASL B
04B3 58                      ASL B
04B4 58                      ASL B
04B5 58                      ASL B
04B6 1B                      ABA               MERGE
04B7 97 5A                   STA A    EXP      STUFF EXP
04B9 20 E6                   BRA      EXINP
04BB 96 5A      GOTDIG       LDA A    EXP
```

```
 04BD 84 F0             AND A   #$F0      MASK MS 4 BITS
 04BF 44                LSR A
 04C0 16                TAB
 04C1 44                LSR A
 04C2 44                LSR A
 04C3 1B                ABA               MULTIPLY BY 10
 04C4 D6 5A             LDA B   EXP       GET OLD EXP BACK
 04C6 C4 0F             AND B   #$0F      GET LS DIGIT
 04C8 1B                ABA               ADD IN
 04C9 D6 59             LDA B   EXPNEG    CHECK FOR EXP SIGN
 04CB 27 01             BEQ     POSEXP
 04CD 40                NEG A
 04CE 9B 56     POSEXP  ADD A   INEXP     GET RESULTING EXP.
 04D0 97 56             STA A   INEXP     STORE IT
 04D2 39                RTS
 04D3 BD E1 AC  INCHAR  JSR     INEEE     GET A CHAR
 04D6 81 20             CMP A   #$20
 04D8 27 F9             BEQ     INCHAR    IGNORE BLANKS
 04DA 39                RTS
 04DB 0D        PROM    FCB     $D,$A,0,0
 04DF 3E                FCC     ;> ;
 04E1 04                FCB     4
 04E2 0D        OVER    FCB     $D,$A,0,0
 04E6 4F                FCC     ;OVERFLOW;
 04EE 04                FCB     4
 04EF 0D        SYNT    FCB     $D,$A,0,0
 04F3 53                FCC     ;SYNTAX;
 04F9 04                FCB     4
 04FA 0A        EQUAL   FCB     $A,0,0
 04FD 20                FCC     ; =;
 04FF 04                FCB     4
                        END

    SYMBOL TABLE:

ACEXP   0026   ADDOP  0342   BEG    0300   BUF1    03D8   BUFTOX 03D3
CHKNXT  04A3   CNTOFF 0380   CRCHK  041D   CTR     003D   DECFLG 0058
DONE    03ED   EQUAL  04FA   EXINP  04A1   EXP     005A   EXPIN  0494
EXPNEG  0059   EXPOK  04B0   FPADD  0103   FPDIV   0194   FPMUL  0180
FPSUB   0100   FULL   0463   GETIN  041A   GOTCNT  038F   GOTDIG 04BB
GOTOP   048C   GOTZER 042D   INBUF  0050   INCH    0401   INCHAR 04D3
INEEE   E1AC   INEXP  0056   INNEXT 0419   INPUT   03EE   ISPT   0417
LOHALF  044D   MULOP  0338   NOPRT  03D0   NOTNEG  0410   NOTNXT 0457
NOTPT   0414   NOTYET 0461   NOVFL  0356   NXTOP   031B   OPER   005C
OUTEEE  E1D1   OUTHL  E067   OUTHR  E06B   OV      0349   OVCHK  035D
OVER    04E2   OVFL   003A   PDATA1 E07E   POS     0376   POSEXP 04CE
PRINT   0345   PROM   04DB   PRTEXP 03AB   PRTEXS  03BC   PRTLOP 039A
PRTMES  034C   REL    0492   RSIGN  0020   SIGDIG  0057   START  0303
STORIT  0445   STUF   03F1   SUBOP  033D   SUBT    03C0   SYNERF 0488
SYNERR  0351   SYNEXP 04AB   SYNT   04EF   SYNTAX  005B   TOGGLE 003E
TOOMAN  03C7   TSTNXT 043E   XSIGN  002C   XTEMP   003F   XTEMP2 0041
YSIGN   0033
```

```
S105A04803000F
S11303008EA042CE04DBBDE07E4F975B975CBD03BD
S1130310EE965B263CCE002CBD03D3BD03EE965B6C
S1130320262FCE0033BD03D3965C4A27154A270DEA
S11303304A2705BD0194200DBD01802008BD0100A0
S11303402003BD0103963A270DCE04E2BDE07E20D2
S1130350B2CE04EF20F6962127037A002696268152
S1130360632EE6819D2DE2CE04FABDE07E962C2721
S113037005862DBDE1D19621BDE06BC608CE0025D2
S1130380A600847F26095A85F02604095A26F1CEBF
S113039000225D2716862EBDE1D1A600BDE0675A76
S11303A02709A600BDE068085A26EFD6262721862A
S11303B045BDE1D1862B5D2A0350862DBDE1D14F89
S11303C0C00A25034C20F9BDE06B860A1BBDE06B17
S11303D07E0303DF3FCE0050A600088C0058270D93
S11303E0DF41DE3FA70008DF3FDE4120EB39CE00CE
S11303F05A6F00098C004F26F8087F003DC6FFD7CE
S11304003EBD04D3812B2711812D260463002009CE
S1130410812E2703082006975808BD04D3810D2791
S113042071803027082B3A810922369757D63D260A
S1130430E9D6572609D65827E17A005620DCD65843
S113044026037C0056D63E260448484848AA00A7FE
S11304500073003E2601088C005626BE73003D2022
S1130460B98B308145272DC601812B271F5C812D37
S1130470271A5C812A27155C812F2710812E2608D4
S1130480D658260497582092975B208E965C26F8BF
S1130490D75C20278D3D812B27FA812D26057300FB
S11304A0598D3080302B04810923058B307E041D47
S11304B0D65A585858581B975A20E6965A84F044EE
S11304C01644441BD65AC40F1BD6592701409B56C9
S11304D0975639BDE1AC812027F9390D0A00003E59
S11304E020040D0A00004F564552464C4F57040D48
S11304F00A000053594E544158040A0000203D0498
S9030000FC
```

```
        *R E1 EA E1 AF08 0300 A042
        *G
        > 12*12
         =1.44E+02
        > 355/113
         =3.14159292
        > 3.55E2/1.33E2
         =2.66917293
        > 100/3
         =3.3333333E+01
        > 12--5
         =1.7E+01
        > +13.123456789*1
         =1.31234567E+01
        > 1../
        SYNTAX
        > 1/0
        OVERFLOW
        > 1E60*1E60
        OVERFLOW
        > 5.28E3/3
         =1.76E+03
        >
```