```
                                00100   SUBTTL  FLOATING POINT MATH PACKAGE CONFIGURATION
                                00120   TITLE   MATHPK FOR BASIC MCS 8080  GATES/ALLEN/DAVIDOFF

                                00160   IFNDEF  LENGTH,<
                                00180           PRINTX  !!! MUST HAVE COM !!!
                                00200           END>

                                00240   RADIX   8                       ;!!!! ALERT !!!!
                                00260                                   ;THROUGHOUT THE MATHPACKAGE!!

                    000000      00300           .P==0

                                00340   INTERNAL        ZERO,FLOAT,FLOATR,MOVE,FADD,FADDS,FSUB,FMULT,FDIV,FIN,FOUT
                                00360   INTERNAL        PUSHF,ABS,INT,QINT,SGN,SQR,RND,SIN,FCOMP,SIGNC,OVERR
                                00380   INTERNAL        INPRT,LINPRT,MOVFM,MOVMF,MOVFR,MOVRF,MOVRM,NEG,INKART,INXHRT
                                00400   IFN     EXTFNC,<
                                00420   INTERNAL        FPWR,EXP,LOG,COS,TAN,ATN,FONE>
                                00440   IFN     MULDIM&<LENGTH-2>,<
                                00460   INTERNAL        DMULT>
                                00480   IFN     STRING,<
                                00500   INTERNAL        SIGNS>
                                00520   IFN     LENGTH-2,<
                                00540   INTERNAL        FADDT,FSUBT,FMULTT,FDIVT>
                                00560   IFE     LENGTH-1,<
                                00580   INTERNAL        FPWRT>
                                00600   IFE     LENGTH-2,<
                                00620   INTERNAL        VMOVMF,VMOVFM,FRCINT,FRCSNG,FRCDBL,VNEG,PUFOUT,DCXBRT,IADD
                                00640   INTERNAL        ISUB,IMULT,IDIV,ICOMP,INEG,DADD,DSUB,DMULT,DDIV,DCOMP,INTFNC>


                                00700   EXTERNAL        FAC,FACLO,FBUFFR,MINUTK,PLUSTK,ERROR,DV0ERR,ERROV,FCERR,SIGN
                                00720   EXTERNAL        SCODE
                                00740   IFE     LENGTH-2,<
                                00760   EXTERNAL        DFACLO,ARG,ARGLO,VALTYP,TMERR,TEMP2,TEMP3>


                                00820   COMMENT %
                                00840           EXTERNAL LOCATIONS USED BY THE MATH-PACKAGE
                                00860           ;THE FLOATING ACCUMULATOR
                                00880   IFE     LENGTH-2,<
                                00900           BLOCK   1               ;(TEMPORARY LEAST SIGNIFICANT BYTE)
                                00920   DFACLO: BLOCK   4>              ;(FOUR LOWEST ORDERS FOR DOUBLE PRECISION)
                                00940   FACLO:  BLOCK   3               ;(LOW ORDER OF MANTISSA (LO))
                                00960                                   ;(MIDDLE ORDER OF MANTISSA (MO))
                                00980                                   ;(HIGH ORDER OF MANTISSA (MO))
                                01000   FAC:    BLOCK   2               ;(EXPONENT)
                                01020                                   ;(TEMPORARY COMPLEMENT OF SIGN IN MSB)
                                01040   IFE     LENGTH-2,<
                                01060   ARGLO:  BLOCK   7               ;(LOCATION OF SECOND ARGUMENT FOR DOUBLE
                                01080   ARG:    BLOCK   1>              ; PRECISION)
                                01100   FBUFFR: BLOCK   ^D13            ;BUFFER FOR FOUT
                                01120   IFE     LENGTH-2,<BLOCK ^D<30-13>>
```



```

                                01180   THE FLOATING POINT FORMAT IS AS FOLLOWS:

                                01220   THE SIGN IS THE FIRST BIT OF THE MANTISSA
                                01240   THE MANTISSA IS 24 BITS LONG
                                01260   THE BINARY POINT IS TO THE LEFT OF THE MSB
                                01280   NUMBER = MANTISSA * 2 ^ EXPONENT
                                01300   THE MANTISSA IS POSITIVE, WITH A ONE ASSUMED TO BE WHERE THE SIGN BIT IS
                                01320   THE SIGN OF THE EXPONENT IS THE FIRST BIT OF THE EXPONENT
                                01340   THE EXPONENT IS STORED IN EXCESS 200 I.E, WITH A BIAS OF 200
                                01360   SO, THE EXPONENT IS A SIGNED 8-BIT NUMBER WITH 200 ADDED TO IT
                                01380   AN EXPONENT OF ZERO MEANS THE NUMBER IS ZERO, THE OTHER BYTES ARE IGNORED
                                01400   TO KEEP THE SAME NUMBER IN THE FAC WHILE SHIFTING:
                                01420           TO SHIFT RIGHT, EXP:=EXP+1
                                01440           TO SHIFT LEFT,  EXP:=EXP-1

                                01480   SO, IN MEMORY THE NUMBER LOOKS LIKE THIS:
                                01500           (BITS 17-24 OF THE MANTISSA)
                                01520           (BITS 9-16 OF THE MANTISSA)
                                01540           (THE SIGN IN BIT 7, BITS 2-8 OF THE MANTISSA ARE IN BITS 6-0)
                                01560           (THE EXPONENT AS A SIGNED NUMBER + 200)
                                01580   (REMEMBER THAT BIT 1 OF THE MANTISSA IS ALWAYS A ONE)

                                01620   ARITHMETIC ROUTINE CALLING CONVENTIONS:

                                01660   FOR ONE ARGUMENT FUNCTIONS:
                                01680           THE ARGUMENT IS IN THE FAC, THE RESULT IS LEFT IN THE FAC
                                01700   FOR TWO ARGUMENT OPERATIONS:
                                01720           THE FIRST ARGUMENT IS IN B,C,D,E I,E, THE "REGISTERS"
                                01740           THE SECOND ARGUMENT IS IN THE FAC
                                01760           THE RESULT IS LEFT IN THE FAC

                                01800   THE "S" ENTRY POINTS TO THE TWO ARGUMENT OPERATIONS HAVE (HL) POINTING TO
                                01820   THE FIRST ARGUMENT INSTEAD OF THE FIRST ARGUMENT BEING IN THE REGISTERS,
                                01840   MOVRM IS CALLED TO GET THE ARGUMENT IN THE REGISTERS,
                                01860   THE "T" ENTRY POINTS ASSUME THE FIRST ARGUMENT IS ON THE STACK,
                                01880   POPR IS USED TO GET THE ARGUMENT IN THE REGISTERS,
                                01900   NOTE: THE "T" ENTRY POINTS SHOULD ALWAYS BE JUMPED TO AND NEVER CALLED
                                01920   BECAUSE THE RETURN ADDRESS ON THE STACK WILL BE CONFUSED WITH THE NUMBER,

                                01960   ON THE STACK, THE TWO LO'S ARE PUSHED ON FIRST AND THEN THE MO AND SIGN,
                                01980   THIS IS DONE SO IF A NUMBER IS STORED IN MEMORY, IT CAN BE PUSHED ON THE
                                02000   STACK WITH TWO PUSHM'S,  THE LOWER BYTE OF EACH PART IS IN THE LOWER
                                02020   MEMORY ADDRESS SO WHEN THE NUMBER IS POPPED INTO THE REGISTERS, THE HIGHER
                                02040   ORDER BYTE WILL BE IN THE HIGHER ORDER REGISTER OF THE REGISTER PAIR, I,E,
                                02060   THE HIGHER ORDER BYTE WILL BE POPPED INTO B, D OR H,
                                02080   %
                                02100   PAGE
```

```
                                                02120   SUBTTL FLOATING POINT ADDITION AND SUBTRACTION
                                                02140           ;ENTRY TO FADD WITH POINTER TO ARG IN (HL)
000000' 001000  000041          02160   FADDM:  LXI     H,FHALF         ;ENTRY TO ADD 1/2
000001' 000000  002312'
000002' 000000  000000
000003' 001000  000315          02180   FADDS:  CALL    MOVRM           ;GET ARGUMENT INTO THE REGISTERS
000004' 000000  001243'
000005' 000000  000001'
000006' 001000  000303          02200           JMP     FADD            ;DO THE ADDITION
000007' 000000  000025'
000010' 000000  000004'

                                                02260           ;SUBTRACTION    FAC:=ARG-FAC
                                02280   IFN     EXTFNC,<
000011' 001000  000315          02300   FSUBS:  CALL    MOVRM>          ;ENTRY IF POINTER TO ARG IS IN (HL)
000012' 000000  001243'
000013' 000000  000007'
                                02320   IFE     LENGTH-1,<
000014' 001000  000041          02340           XWD     1000,041>       ;"LXI   H" AROUND NEXT 2 BYTES
                                02360   IFN     LENGTH-2,<
000015' 001000  000301          02380   FSUBT:  POPR>                   ;ENTRY IF ARGUMENT IS ON THE STACK
000016' 001000  000321
000017' 001000  000315          02400   FSUB:   CALL    NEG             ;NEGATE SECOND ARGUMENT
000020' 000000  001175'
000021' 000000  000012'
                                02420                                   ;FALL INTO FADD

                                02460           ;ADDITION       FAC:=ARG+FAC
                                02500           ;ALTERS A,B,C,D,E,H,L
                                02520   IFN     LENGTH-2,<
000022' 001000  000041          02540           XWD     1000,041        ;"LXI   H" AROUND NEXT 2 BYTES
000023' 001000  000301          02560   FADDT:  POPR>                   ;ENTRY IF ARGUMENT IS ON THE STACK
000024' 001000  000321
000025' 001000  000170          02580   FADD:   MOV     A,B             ;CHECK IF FIRST ARGUMENT IS ZERO
000026' 001000  000267          02600           ORA     A               ;GET EXPONENT
000027' 001000  000310          02620           RZ                      ;IT IS, RESULT IS NUMBER IN FAC
000030' 001000  000072          02640           LDA     FAC             ;GET EXPONENT
000031' 000000  000000*
000032' 000000  000026'
000033' 001000  000267          02660           ORA     A               ;SEE IF THE NUMBER IS ZERO
000034' 001000  000312          02680           JZ      MOVFR           ;IT IS, ANSWER IS IN REGISTERS
000035' 000000  001225'
000036' 000000  000031'

                                02720   ;WE WANT TO GET THE SMALLER NUMBER IN THE REGISTERS SO WE CAN SHIFT IT RIGHT
                                02740   ;AND ALIGN THE BINARY POINTS OF THE TWO NUMBERS,  THEN WE CAN JUST ADD OR
                                02760   ;SUBTRACT THEM (DEPENDING ON THEIR SIGNS) BYTEWISE,
000037' 001000  000220          02780           SUB     B               ;CHECK RELATIVE SIZES
000040' 001000  000322          02800           JNC     FADD1           ;IS FAC SMALLER?
000041' 000002  000057'
000042' 000000  000035'
```



```
000043' 001000  000057          02820           CMA                     ;YES, NEGATE SHIFT COUNT
000044' 001000  000074          02840           INR     A
000045' 001000  000353          02860           XCHG                    ;SWITCH FAC AND REGISTERS, SAVE (DE)
000046' 001000  000315          02880           CALL    PUSHF           ;PUT FAC ON STACK
000047' 000000  001205'
000050' 000000  000041'
000051' 001000  000353          02900           XCHG                    ;GET (DE) BACK WHERE IT BELONGS
000052' 001000  000315          02920           CALL    MOVFR           ;PUT REGISTERS IN THE FAC
000053' 000000  001225'
000054' 000000  000047'
000055' 001000  000301          02940           POPR                    ;GET THE OLD FAC IN THE REGISTERS
000056' 001000  000321
000057'                         02960   FADD1:
                                02980   IFN     LENGTH,<
000057' 001000  000376          03000           CPI     31              ;ARE WE WITHIN 24 BITS?
000060' 000000  000031
000061' 001000  000320          03020           RNC>                    ;NO, ALL DONE
000062' 001000  000365          03040           PUSH    PSW             ;SAVE SHIFT COUNT
000063' 001000  000315          03060           CALL    UNPACK          ;UNPACK THE NUMBERS
000064' 000000  001272'
000065' 000000  000053'
000066' 001000  000147          03080           MOV     M,A             ;SAVE SUBTRACTION FLAG
000067' 001000  000361          03100           POP     PSW             ;GET SHIFT COUNT BACK
000070' 001000  000315          03120           CALL    SHIFTR          ;SHIFT REGISTERS RIGHT THE RIGHT AMOUNT
000071' 000000  000334'
000072' 000000  000064'

                                03160   ;IF THE NUMBERS HAVE THE SAME SIGN, THEN WE ADD THEM,  IF THE SIGNS ARE
                                03180   ;DIFFERENT, THEN WE HAVE TO SUBTRACT THEM,  WE HAVE TO DO THIS BECAUSE THE
                                03200   ;MANTISSAS ARE POSITIVE,  JUDGING BY THE EXPONENTS, THE LARGER NUMBER IS IN
                                03220   ;THE FAC, SO IF WE SUBTRACT, THE SIGN OF THE RESULT SHOULD BE THE SIGN OF THE
                                03240   ;FAC; HOWEVER, IF THE EXPONENTS ARE THE SAME, THE NUMBER IN THE REGISTERS
                                03260   ;COULD BE BIGGER, SO AFTER WE SUBTRACT THEM, WE HAVE TO CHECK IF THE RESULT
                                03280   ;WAS NEGATIVE,  IF IT WAS, WE NEGATE THE NUMBER IN THE REGISTERS AND
                                03300   ;COMPLEMENT THE SIGN OF THE FAC,  (HERE THE FAC IS UNPACKED)
                                03320   ;IF WE HAVE TO ADD THE NUMBERS, THE SIGN OF THE RESULT IS THE SIGN OF THE
                                03340   ;FAC,  SO, IN EITHER CASE, WHEN WE ARE ALL DONE, THE SIGN OF THE RESULT
                                03360   ;WILL BE THE SIGN OF THE FAC,
000073' 001000  000264          03380           ORA     H               ;GET SUBTRACTION FLAG
000074' 001000  000041          03400           LXI     H,FACLO         ;SET POINTER TO LO'S
000075' 000000  000006*
000076' 000000  000071'
000077' 001000  000362          03420           JP      FADD3           ;SUBTRACT IF THE SIGNS WERE DIFFERENT
000100' 000000  000125'
000101' 000000  000075'
000102' 001000  000315          03440           CALL    FADDA           ;ADD THE NUMBERS
000103' 000000  000274'
000104' 000000  000100'
000105' 001000  000322          03460           JNC     ROUND           ;ROUND RESULT IF THERE WAS NO OVERFLOW
000106' 000000  000233'
000107' 000000  000103'
                                03480                                   ;THE MOST IT CAN OVERFLOW IS ONE BIT
000110' 001000  000043          03500           INX     H               ;THERE WAS OVERFLOW
```

```
                                                03520           INR     M               ;INCREMENT EXPONENT
                                                03540           JZ      OVERR           ;CHECK FOR OVERFLOW
                                                03560   IFE     LENGTH,<
                                                03580           CALL    SHFTRO>         ;SHIFT RESULT RIGHT ONE, SHIFT CARRY IN
                                                03600   IFN     LENGTH,<
                                                03620           MVI     L,1             ;SHIFT RESULT RIGHT ONE, SHIFT CARRY IN
                                                03640           CALL    SHRADD>
                                                03660           JMP     ROUND           ;ROUND RESULT AND WE ARE DONE
                                                03680           ;HERE TO SUBTRACT C,D,E,B FROM ((HL)+0,1,2),0
                                                03700   FADD3:  XRA     A               ;SUBTRACT NUMBERS, NEGATE UNDERFLOW BYTE
                                                03720           SUB     B
                                                03740           MOV     B,A             ;SAVE IT
                                                03760           MOV     A,M             ;SUBTRACT LOW ORDERS
                                                03780           SBB     E
                                                03800           MOV     E,A
                                                03820           INX     H               ;UPDATE POINTER TO NEXT BYTE
                                                03840           MOV     A,M             ;SUBTRACT MIDDLE ORDERS
                                                03860           SBB     D
                                                03880           MOV     D,A
                                                03900           INX     H               ;UPDATE POINTER TO HIGH ORDERS
                                                03920           MOV     A,M             ;SUBTRACT HIGH ORDERS
                                                03940           SBB     C
                                                03960           MOV     C,A
                                                03980           ;BECAUSE WE WANT A POSITIVE MANTISSA, CHECK IF WE HAVE TO NEGATE THE
                                                04000           ; NUMBER
                                                04020   FADFLT: CC      NEGR            ;ENTRY FROM FLOATR, INT: NEGATE NUMBER IF IT
                                                04040                                   ; WAS NEGATIVE, FALL INTO NORMALIZE

                                                04100           ;NORMALIZE C,D,E,B
                                                04120           ;ALTERS A,B,C,D,E,H,L
                                                04140           ;HERE WE SHIFT THE MANTISSA LEFT UNTIL THE MSB IS A ONE,
                                                04160           ;EXCEPT IN 4K, THE IDEA IS TO SHIFT LEFT BY 8 AS MANY TIMES AS
                                                04180           ;POSSIBLE.
000146'                                         04200   NORMAL:
                                                04220   IFE     LENGTH,<
                                                04240           MVI     H,0             ;CLEAR SHIFT COUNT
                                                04260           MOV     A,C             ;IS THE NUMBER NORMALIZED?
                                                04280           ORA     A
                                                04300           JM      ROUND           ;YES, WE ARE DONE
                                                04320   NORM2:  CPI     340             ;IS THE RESULT ZERO?
                                                04340           JZ      ZERO            ;YES, ZERO THE FAC
                                                04360           DCR     H               ;NO, DECREMENT SHIFT COUNT
                                                04380           MOV     A,B             ;SHIFT THE LO LEFT
```



```
                                                04400           ADD     A               ;SHIFT IN A ZERO
                                                04420           MOV     B,A
                                                04440           CALL    SHFTLO          ;SHIFT THE REST OF THE NUMBER LEFT ONE
                                                04460           MOV     A,M             ;GET THE SHIFT COUNT
                                                04480           JP      NORM2>          ;CONTINUE IF NUMBER IS NOT NORMALIZED
                                                04500   IFN     LENGTH,<
000146' 001000  000150                          04520           MOV     L,B             ;PUT LOWEST 2 BYTES IN (HL)
000147' 001000  000143                          04540           MOV     H,E
000150' 001000  000257                          04560           XRA     A               ;ZERO SHIFT COUNT
000151' 001000  000107                          04580   NORM1:  MOV     B,A             ;SAVE SHIFT COUNT
000152' 001000  000171                          04600           MOV     A,C             ;DO WE HAVE 1 BYTE OF ZEROS
000153' 001000  000267                          04620           ORA     A
000154' 001000  000302                          04640           JNZ     NORM3           ;NO, SHIFT ONE PLACE AT A TIME
000155' 000000  000210'
000156' 000000  000144'
                                                04660           ;THIS LOOP SPEEDS THINGS UP BY SHIFTING 8 PLACES AT ONE TIME
000157' 001000  000112                          04680           MOV     C,D             ;YES, SHIFT OVER 1 BYTE
000160' 001000  000124                          04700           MOV     D,M
000161' 001000  000145                          04720           MOV     H,L
000162' 001000  000157                          04740           MOV     L,A             ;SHIFT IN 8 ZEROS FOR THE LOW ORDER
000163' 001000  000170                          04760           MOV     A,B             ;UPDATE SHIFT COUNT
000164' 001000  000326                          04780           SUI     10
000165' 000000  000010
000166' 001000  000376                          04800           CPI     340             ;DID WE SHIFT IN 4 BYTES OF ZEROS?
000167' 000000  000340
000170' 001000  000302                          04820           JNZ     NORM1           ;NO, TRY TO SHIFT OVER 8 MORE
000171' 000000  000151'
000172' 000000  000155'
                                                04840                                   ;YES, NUMBER WAS ZERO,  FALL INTO ZERO

                                                04900           ;ZERO FAC
                                                04920           ;ALTERS A ONLY
                                                04940           ;EXITS WITH A=0
                                                04960           ;BY OUR FLOATING POINT FORMAT, THE NUMBER IS ZERO IF THE EXPONENT IS
                                                04980           ; ZERO
000173' 001000  000257                          05000   ZERO:   XRA     A               ;ZERO A
000174' 001000  000062                          05020   ZERO0:  STA     FAC             ;ZERO THE FAC'S EXPONENT, ENTRY IF A=0
000175' 000000  000031*
000176' 000000  000171'
000177' 001000  000311                          05040           RET                     ;ALL DONE

000200' 001000  000005                          05100   NORM2:  DCR     B               ;DECREMENT SHIFT COUNT
000201' 001000  000051                          05120           DAD     H               ;ROTATE (HL) LEFT ONE, SHIFT IN A ZERO
000202' 001000  000172                          05140           MOV     A,D             ;ROTATE NEXT HIGHER ORDER LEFT ONE
000203' 001000  000027                          05160           RAL
000204' 001000  000127                          05180           MOV     D,A
000205' 001000  000171                          05200           MOV     A,C             ;ROTATE HIGH ORDER LEFT ONE
000206' 001000  000217                          05220           ADC     A               ;SET CONDITION CODES
000207' 001000  000117                          05240           MOV     C,A
000210' 001000  000362                          05260   NORM3:  JP      NORM2           ;WE HAVE MORE NORMALIZATION TO DO
000211' 000000  000200'
```

```
000212' 000000  000175'
000213' 001000  000170          05280           MOV     A,B             ;ALL NORMALIZED, GET SHIFT COUNT
000214' 001000  000134          05300           MOV     E,M             ;PUT LO'S BACK IN E,B
000215' 001000  000105          05320           MOV     B,L
000216' 001000  000267          05340           ORA     A               ;CHECK IF WE DID NO SHIFTING
000217' 001000  000312          05360           JZ      ROUND>
000220' 000000  000233'
000221' 000000  000211'
000222' 001000  000041          05380           LXI     H,FAC           ;LOOK AT FAC'S EXPONENT
000223' 000000  000175*
000224' 000000  000220'
000225' 001000  000206          05400           ADD     M               ;UPDATE EXPONENT
000226' 001000  000167          05420           MOV     M,A
000227' 001000  000322          05440           JNC     ZERO            ;CHECK FOR UNDERFLOW
000230' 000000  000173'
000231' 000000  000223'
000232' 001000  000310          05460           RZ                      ;NUMBER IS ZERO, ALL DONE
                                05480                                   ;FALL INTO ROUND AND WE ARE DONE

                                05540           ;ROUND RESULT IN C,D,E,B AND PUT NUMBER IN THE FAC
                                05560           ;ALTERS A,B,C,D,E,H,L
                                05580           ;WE ROUND C,D,E UP OR DOWN DEPENDING UPON THE MSB OF B
000233' 001000  000170  ROUND:  05600   ROUND:  MOV     A,B             ;SEE IF WE SHOULD ROUND UP
000234' 001000  000041          05620   ROUNDB: LXI     H,FAC           ;ENTRY FROM FDIV, GET POINTER TO EXPONENT
000235' 000000  000223*
000236' 000000  000230'
000237' 001000  000267          05640           ORA     A
000240' 001000  000374          05660           CM      ROUNDA          ;DO IT IF NECESSARY
000241' 000000  000255'
000242' 000000  000235'
000243' 001000  000106          05680           MOV     B,M             ;PUT EXPONENT IN B
                                05700           ;HERE WE PACK THE MO AND SIGN
000244' 001000  000043          05720           INX     H               ;POINT TO SIGN
000245' 001000  000176          05740           MOV     A,M             ;GET SIGN
000246' 001000  000346          05760           ANI     200             ;GET RID OF UNWANTED BITS
000247' 000000  000200
000250' 001000  000251          05780           XRA     C               ;PACK SIGN AND MO
000251' 001000  000117          05800           MOV     C,A             ;SAVE IT IN C
000252' 001000  000303          05820           JMP     MOVFR           ;SAVE NUMBER IN FAC
000253' 000000  001225'
000254' 000000  000241'


                                05880   IFE     LENGTH,<
                                05900           ;SHIFT C,D,E LEFT ONE
                                05920           ;THIS IS USED BY NORMAL, FDIV
                                05940           ;ALTERS A,C,D,E
                                05960   SHFTLO: MOV     A,E             ;GET THE LO
                                05980           RAL                     ;SHIFT IT
                                06000           MOV     E,A             ;SAVE IT
                                06020           MOV     A,D             ;SHIFT THE NEXT HIGHER ORDER
                                06040           RAL
```



```
                                06060           MOV     D,A
                                06080           MOV     A,C             ;SHIFT THE HIGHEST ORDER
                                06100           ADC     A               ;ROTATE A LEFT AND SET CONDITION CODES
                                06120           MOV     C,A
                                06140           RET>                    ;ALL DONE


                                06200           ;SUBROUTNE FOR ROUND:  ADD ONE TO C,D,E
000255' 001000  000034          06220   ROUNDA: INR     E               ;ADD ONE TO THE LOW ORDER, ENTRY FROM QINT
000256' 001000  000300          06240           RNZ                     ;ALL DONE IF IT IS NOT ZERO
000257' 001000  000024          06260           INR     D               ;ADD ONE TO NEXT HIGHER ORDER
000260' 001000  000300          06280           RNZ                     ;ALL DONE IF NO OVERFLOW
000261' 001000  000014          06300           INR     C               ;ADD ONE TO THE HIGHEST ORDER
000262' 001000  000300          06320           RNZ                     ;RETURN IF NO OVEFLOW
000263' 001000  000016          06340           MVI     C,200           ;THE NUMBER OVERFLOWED, SET NEW HIGH ORDER
000264' 000000  000220
000265' 001000  000064          06360           INR     M               ;UPDATE EXPONENT
000266' 001000  000300          06380           RNZ                     ;RETURN IF IT DID NOT OVERFLOW
                                06400                                   ;IT DID, FALL INTO OVERR

                                06440           ;OVERFLOW ERROR
000267' 001000  000036  OVERR:  06460   OVERR:  MVI     E,ERROV         ;SET OVERFLOW ERROR CODE
000270' 000000  000000*
000271' 001000  000303          06480           JMP     ERROR           ;GO TO IT!!
000272' 000000  000000*
000273' 000000  000253'


                                06540           ;ADD (HL)+2,1,0 TO C,D,E
                                06560           ;THIS CODE IS USED BY FADD, FOUT
000274' 001000  000176  FADDA:  06580   FADDA:  MOV     A,M             ;GET LOWEST ORDER
000275' 001000  000203          06600           ADD     E               ;ADD IN OTHER LOWEST ORDER
000276' 001000  000137          06620           MOV     E,A             ;SAVE IT
000277' 001000  000043          06640           INX     H               ;UPDATE POINTER TO NEXT BYTE
000300' 001000  000176          06660           MOV     A,M             ;ADD MIDDLE ORDERS
000301' 001000  000212          06680           ADC     D
000302' 001000  000127          06700           MOV     D,A
000303' 001000  000043          06720           INX     H               ;UPDATE POINTER TO HIGH ORDER
000304' 001000  000176          06740           MOV     A,M             ;ADD HIGH ORDERS
000305' 001000  000211          06760           ADC     C
000306' 001000  000117          06780           MOV     C,A
000307' 001000  000311          06800           RET                     ;ALL DONE


                                06860           ;NEGATE NUMBER IN C,D,E,B
                                06880           ;THIS CODE IS USED BY FADD, QINT
                                06900           ;ALTERS A,B,C,D,E,L
000310' 001000  000041  NEGN:   06920   NEGN:   LXI     H,FAC+1         ;NEGATE FAC
000311' 000000  000001*
000312' 000000  000272'
000313' 001000  000176          06940           MOV     A,M             ;GET SIGN
000314' 001000  000057          06960           CMA                     ;COMPLEMENT IT
000315' 001000  000167          06980           MOV     M,A             ;SAVE IT AGAIN
```

```
000316' 001000  000257          07000           XRA     A               ;ZERO A
000317' 001000  000157          07020           MOV     L,A             ;SAVE ZERO IN L
000320' 001000  000220          07040           SUB     B               ;NEGATE LOWEST ORDER
000321' 001000  000107          07060           MOV     B,A             ;SAVE IT
000322' 001000  000175          07080           MOV     A,L             ;GET A ZERO
000323' 001000  000233          07100           SBB     E               ;NEGATE NEXT HIGHEST ORDER
000324' 001000  000137          07120           MOV     E,A             ;SAVE IT
000325' 001000  000175          07140           MOV     A,L             ;GET A ZERO
000326' 001000  000232          07160           SBB     D               ;NEGATE NEXT HIGHEST ORDER
000327' 001000  000127          07180           MOV     D,A             ;SAVE IT
000330' 001000  000175          07200           MOV     A,L             ;GET ZERO BACK
000331' 001000  000231          07220           SBB     C               ;NEGATE HIGHEST ORDER
000332' 001000  000117          07240           MOV     C,A             ;SAVE IT
000333' 001000  000311          07260           RET                     ;ALL DONE

                                07320           ;SHIFT C,D,E RIGHT
                                07340           ;A = SHIFT COUNT
                                07360           ;ALTERS A,B,C,D,E,L
                                07380           ;THE IDEA (EXCEPT IN 4K) IS TO SHIFT RIGHT 8 PLACES AS MANY TIMES AS
                                07400           ; POSSIBLE
000334' 001000  000006          07420   SHIFTR: MVI     B,0             ;ZERO OVERFLOW BYTE
000335' 000000  000000          07440   IFE     LENGTH,<
                                07460           INR     A>              ;ADD ONE TO SHIFT COUNT
                                07480   IFN     LENGTH,<
000336' 001000  000326          07500   SHFTR1: SUI     10              ;CAN WE SHIFT IT 8 RIGHT?
000337' 000000  000010
000340' 001000  000332          07520           JC      SHFTR2          ;NO, SHIFT IT ONE PLACE AT A TIME
000341' 000000  000353'
000342' 000000  000311'
                                07540           ;THIS LOOP SPEEDS THINGS UP BY SHIFTING 8 PLACES AT ONE TIME
000343' 001000  000103          07560           MOV     B,E             ;SHIFT NUMBER 1 BYTE RIGHT
000344' 001000  000132          07580           MOV     E,D
000345' 001000  000121          07600           MOV     D,C
000346' 001000  000016          07620           MVI     C,0             ;PUT 0 IN MO
000347' 000000  000000
000350' 001000  000303          07640           JMP     SHFTR1          ;TRY TO SHIFT 8 RIGHT AGAIN
000351' 000000  000336'
000352' 000000  000341'
000353' 001000  000306          07660   SHFTR2: ADI     11>             ;CORRECT SHIFT COUNT
000354' 000000  000011
000355' 001000  000157          07680           MOV     L,A             ;SAVE SHIFT COUNT
000356' 001000  000257          07700   SHFTR3: XRA     A               ;CLEAR CARRY
000357' 001000  000055          07720           DCR     L               ;ARE WE DONE SHIFTING?
000360' 001000  000310          07740           RZ                      ;RETURN IF WE ARE
                                07760   IFE     LENGTH,<
                                07780   SHRADD: CALL    SHFTRO>         ;SHIFT THE NUMBER RIGHT ONE
                                07800   IFN     LENGTH,<
000361' 001000  000171          07820           MOV     A,C             ;GET MO
000362' 001000  000037          07840   SHRADD: RAR                     ;ENTRY FROM FADD, SHIFT IT RIGHT
000363' 001000  000117          07860           MOV     C,A             ;SAVE IT
000364' 001000  000172          07880           MOV     A,D             ;SHIFT NEXT BYTE RIGHT
```



```
000365' 001000  000037          07900           RAR
000366' 001000  000127          07920           MOV     D,A
000367' 001000  000173          07940           MOV     A,E             ;SHIFT LOW ORDER RIGHT
000370' 001000  000037          07960           RAR
000371' 001000  000137          07980           MOV     E,A
000372' 001000  000170          08000           MOV     A,B             ;SHIFT OVERFLOW BYTE RIGHT
000373' 001000  000037          08020           RAR
000374' 001000  000107          08040           MOV     B,A>
000375' 001000  000303          08060           JMP     SHFTR3          ;SEE IF WE ARE DONE
000376' 000000  000356'
000377' 000000  000351'

                                08120   IFE     LENGTH,<
                                08140           ;SHIFT C,D,E,B RIGHT ONE
                                08160           ;THIS IS USED BY SHIFTR, FMULT, FADD
                                08180           ;ALTERS A,B,C,D,E
                                08200   SHFTRO: MOV     A,C             ;GET THE MO
                                08220   SHFROA: RAR                     ;SHIFT IT RIGHT, ENTRY FROM FMULT
                                08240           MOV     C,A
                                08260           MOV     A,D             ;SHIFT THE MO RIGHT
                                08280           RAR
                                08300           MOV     D,A
                                08320           MOV     A,E             ;SHIFT THE LO
                                08340           RAR
                                08360           MOV     E,A
                                08380           MOV     A,B             ;SHIFT THE EXTRA LO BYTE
                                08400           RAR
                                08420           MOV     B,A
                                08440           RET>                    ;ALL DONE
                                08460   PAGE
```

```
                                        08480   SUBTTL  NATURAL LOG FUNCTION
                                        08500   IFN     EXTFNC,<
                                        08520           ;CALCULATION IS BY:
                                        08540           ; LN(F*2**N)=(N+LOG2(F))*LN(2)
                                        08560           ;AN APPROXIMATION POLYNOMIAL IS USED TO CALCULATE LOG2(F)
                                        08600           ;CONSTANTS USED BY LOG
000400' 000000  000000                  08620   FONE:   000     ; 1
000401' 000000  000000                  08640           000
000402' 000000  000000                  08660           000
000403' 000000  000201                  08680           201
000404' 000000  000003                  08700   LOGCN2: 3       ;DEGREE+1
000405' 000000  000252                  08720           252     ; 0.598978650
000406' 000000  000126                  08740           126
000407' 000000  000031                  08760           031
000410' 000000  000200                  08780           200
000411' 000000  000361                  08800           361     ; 0.961470632
000412' 000000  000042                  08820           042
000413' 000000  000166                  08840           166
000414' 000000  000200                  08860           200
000415' 000000  000105                  08880           105     ; 2.88539129
000416' 000000  000252                  08900           252     ; NOTE: THE REFERENCE FOR THIS CONSTANT HAS 100 NOT 105
000417' 000000  000070                  08920           070     ; IN THE LOW ORDER BYTE.
000420' 000000  000202                  08940           202
                                        08960   LOG:    FSIGN                   ;CHECK FOR A NEGATIVE OR ZERO ARGUMENT
000422' 001000  000352                  09000           JPE     FCERR           ;FAC .LE. 0, BLOW HIM OUT OF THE WATER
000423' 000000  000000*
000424' 000000  000376'
                                        09020                                   ;FSIGN ONLY RETURNS 0,1 OR 377 IN A
                                        09040                                   ;THE PARITY WILL BE EVEN IF A HAS 0 OR 377
000425' 001000  000041                  09060           LXI     H,FAC           ;GET POINTER TO EXPONENT
000426' 000000  000235*
000427' 000000  000423'
000430' 001000  000176                  09080           MOV     A,M             ;GET EXPONENT IN A
000431' 001000  000001                  09100           MOVRI   200,065,004,363 ;GET SQR(,5)
000432' 000000  000065
000433' 000000  000200
000434' 001000  000021
000435' 000000  000363
000436' 000000  000004
000437' 001000  000220                  09120           SUB     B               ;REMOVE EXCESS 200
000440' 001000  000365                  09140           PUSH    PSW             ;SAVE EXPONENT FOR LATER
000441' 001000  000160                  09160           MOV     M,B             ;SET EXP TO 200, RESULT IS NUM IN (,5,1)
000442' 001000  000325                  09180           PUSHR                   ;SAVE SQR(,5)
000443' 001000  000305
000444' 001000  000315                  09200           CALL    FADD            ;CALCULATE (F-SQR(,5))/(F+SQR(,5))
000445' 000000  000025'
000446' 000000  000426'
000447' 001000  000301                  09220           POPR                    ;GET SQR(,5) BACK
000450' 001000  000321
000451' 001000  000004                  09240           INR     B               ;GET SQR(2)
000452' 001000  000315                  09260           CALL    FDIV            ;WHERE F=NUMBER LEFT IN FAC
```



```
000453' 000000  000655'
000454' 000000  000445'
000455' 001000  000041                  09280           LXI     H,FONE          ;THE CALCULATION IS EQUIVALENT TO THE ABOVE,
000456' 000000  000400'
000457' 000000  000453'
000460' 001000  000315                  09300           CALL    FSUBS           ; BUT DONE IN A DIFFERENT ORDER
000461' 000000  000011'
000462' 000000  000456'
000463' 001000  000041                  09320           LXI     H,LOGCN2        ;EVALUATE APPROXIMATION POLYNOMIAL
000464' 000000  000404'
000465' 000000  000461'
000466' 001000  000315                  09340           CALL    POLYX
000467' 000000  002615'
000470' 000000  000464'
000471' 001000  000001                  09360           MOVRI   200,200,000,000 ;GET -1/2
000472' 000000  000200
000473' 000000  000200
000474' 001000  000021
000475' 000000  000000
000476' 000000  000000
000477' 001000  000315                  09380           CALL    FADD            ;ADD IN LAST CONSTANT
000500' 000000  000025'
000501' 000000  000467'
000502' 001000  000361                  09400           POP     PSW             ;RETRIEVE ORIGINAL EXPONENT
000503' 001000  000315                  09420           CALL    FINLOG          ;ADD IT TO ORIGINAL NUMBER
000504' 000000  001731'
000505' 000000  000500'
000506' 001000  000001                  09440   MULLN2: MOVRI   200,261,162,030>;GET LN(2)
000507' 000000  000061
000510' 000000  000200
000511' 001000  000021
000512' 000000  000030
000513' 000000  000162
                                        09460   ;       JMP     FMULT           ;MULTIPLY BY LN(2)
                                        09480   PAGE
```

```
                                                09500   SUBTTL  FLOATING MULTIPLICATION AND DIVISION
                                                09520           ;MULTIPLICATION                 FAC:=ARG*FAC
                                                09540           ;ALTERS A,B,C,D,E,H,L
                                                09560   IFE     EXTFNC,<
                                                09580   FMULTS: CALL    MOVRM>                  ;ENTRY WITH POINTER TO ARG IN (HL)
                                                09600   IFN     LENGTH-2,<
0000514' 001000  000041                         09620           XWD     1000,041        ;"LXI   H" AROUND NEXT 2 BYTES
0000515' 001000  000301                         09640   FMULTT: POPR>                   ;ENTRY IF ARGUMENT IS ON THE STACK
0000516' 001000  000321
0000517' 001000  000357                         09660   FMULT:  FSIGN                   ;CHECK IF FAC IS ZERO
0000520' 001000  000310                         09680           RZ                      ;IF IT IS, RESULT IS ZERO
0000521' 001000  000056                         09700           MVI     L,0             ;ADD THE TWO EXPONENTS, L IS A FLAG
0000522' 000000  000000
0000523' 001000  000315                         09720           CALL    MULDIV          ;FIX UP THE EXPONENTS
0000524' 000000  001035'
0000525' 000000  000504'
                                                09740           ;SAVE THE NUMBER IN THE REGISTERS SO WE CAN ADD IT FAST
0000526' 001000  000171                         09760           MOV     A,C             ;GET MO
0000527' 001000  000062                         09780           STA     FMULTA+1        ;STORE MO OF REGISTERS
0000530' 000000  000606'
0000531' 000000  000524'
0000532' 001000  000353                         09800           XCHG                    ;STORE THE TWO LO'S OF THE REGISTERS
0000533' 001000  000042                         09820           SHLD    FMULTB+1
0000534' 000000  000601'
0000535' 000000  000530'
0000536' 001000  000001                         09840           LXI     B,$CODE         ;ZERO THE PRODUCT REGISTERS
0000537' 000000  000000*
0000540' 000000  000534'
0000541' 001000  000120                         09860           MOV     D,B
0000542' 001000  000130                         09880           MOV     E,B
0000543' 001000  000041                         09900           LXI     H,NORMAL        ;PUT ADDRESS OF NORMAL, WERE WE FINISH UP,
0000544' 000000  000146'
0000545' 000000  000537'
0000546' 001000  000345                         09920           PUSH    H               ; ON THE STACK
0000547' 001000  000041                         09940           LXI     H,FMULT2        ;PUT FMULT2 ON THE STACK TWICE, SO AFTER
0000550' 000000  000557'
0000551' 000000  000544'
0000552' 001000  000345                         09960           PUSH    H               ; WE MULTIPLY BY THE LO BYTE, WE WILL
0000553' 001000  000345                         09980           PUSH    H               ; MULTIPLY BY THE MO AND HO
0000554' 001000  000041                         10000           LXI     H,FACLO         ;GET ADDRESS OF LO OF FAC
0000555' 000000  000075*
0000556' 000000  000550'
0000557' 001000  000176                         10020   FMULT2: MOV     A,M             ;GET BYTE TO MULTIPLY BY
0000560' 001000  000043                         10040           INX     H               ;MOVE POINTER TO NEXT BYTE
                                                10060   IFN     LENGTH,<
                                                10080           ORA     A
0000561' 001000  000267                         10100           JZ      FMULT3>         ;ARE WE MULTIPLYING BY ZERO?
0000562' 001000  000312
0000563' 000000  000632'
0000564' 000000  000555'
0000565' 001000  000345                         10120           PUSH    H               ;SAVE POINTER
                                                10140   IFE     LENGTH,<
                                                10160           MVI     L,10>           ;SET UP A COUNT
                                                10180   IFN     LENGTH,<
```



```
0000566' 001000  000353                         10200           XCHG                    ;GET LO'S IN (HL)
0000567' 001000  000036                         10220           MVI     E,10>           ;SET UP A COUNT
0000570' 000000  000010
                                                10240
                                                10260   ;THE PRODUCT WILL BE FORMED IN C,D,E,B, THIS WILL BE IN C,M,L,B PART OF THE
                                                10280   ;TIME IN ORDER TO USE THE "DAD" INSTRUCTION,  AT FMULT2, WE GET THE NEXT
                                                10300   ;BYTE OF THE MANTISSA IN THE FAC TO MULTIPLY BY,  ((HL) POINTS TO IT)
                                                10320   ;(THE FMULT2 SUBROUTINE PRESERVES (HL))  IN 8K, IF THE BYTE IS ZERO, WE JUST
                                                10340   ;SHIFT THE PRODUCT 8 RIGHT,  THIS BYTE IS THEN SHIFTED RIGHT AND SAVED IN D
                                                10360   ;(M IN 4K),  THE CARRY DETERMINES IF WE SHOULD ADD IN THE SECOND FACTOR
                                                10380   ;IF WE DO, WE ADD IT TO C,M,L,  B IS ONLY USED TO DETERMINE WHICH WAY WE
                                                10400   ;ROUND,  WE THEN SHIFT C,M,L,B (C,D,E,B) IN 4K RIGHT ONE TO GET READY FOR THE
                                                10420   ;NEXT TIME THROUGH THE LOOP,  NOTE THAT THE CARRY IS SHIFTED INTO THE MSB OF
                                                10440   ;C,  E HAS A COUNT (L IN 4K) TO DETERMINE WHEN WE HAVE LOOKED AT ALL THE BITS
                                                10460   ;OF D (H IN 4K),
0000571' 001000  000037                         10480   FMULT4: RAR                     ;ROTATE BYTE RIGHT
                                                10500   IFE     LENGTH,<
                                                10520           MOV     M,A>            ;SAVE THE COUNT
                                                10540   IFN     LENGTH,<
0000572' 001000  000127                         10560           MOV     D,A>            ;SAVE IT
0000573' 001000  000171                         10580           MOV     A,C             ;GET MO
0000574' 001000  000322                         10600           JNC     FMULT5          ;DON'T ADD IN NUMBER IF BIT WAS ZERO
0000575' 000000  000627'
0000576' 000000  000563'
                                                10620   IFE     LENGTH,<
                                                10640           XCHG>                   ;PUT THE LO'S IN (HL)
0000577' 001000  000325                         10660           PUSH    D               ;SAVE COUNTERS
0000600' 001000  000021                         10680   FMULTB: LXI     D,$CODE         ;GET LO'S OF NUMBER TO ADD, THIS IS SET ABOVE
0000601' 000000  000537*
0000602' 000000  000575'
0000603' 001000  000031                         10700           DAD     D               ;ADD THEM IN
0000604' 001000  000321                         10720           POP     D               ;GET COUNTERS BACK
0000605' 001000  000316                         10740   FMULTA: ACI     0               ;ADD IN MO, THIS IS SET UP ABOVE
0000606' 000000  000000
                                                10760   IFE     LENGTH,<
                                                10780           XCHG                    ;PUT THE LO'S BACK IN (DE)
                                                10800   FMULTS: CALL    SHFROA          ;SHIFT THE RESULT RIGHT ONE
                                                10820           DCR     L               ;ARE WE DONE?
                                                10840           MOV     A,M>            ;GET NUMBER WE ARE MULTIPLYING BY
                                                10860   IFN     LENGTH,<
0000607' 001000  000037                         10880   FMULTS: RAR                     ;ROTATE RESULT RIGHT ONE
0000610' 001000  000117                         10900           MOV     C,A
0000611' 001000  000174                         10920           MOV     A,M             ;ROTATE NEXT BYTE
0000612' 001000  000037                         10940           RAR
0000613' 001000  000147                         10960           MOV     M,A
0000614' 001000  000175                         10980           MOV     A,L             ;ROTATE NEXT LOWER ORDER
0000615' 001000  000037                         11000           RAR
0000616' 001000  000157                         11020           MOV     L,A
0000617' 001000  000170                         11040           MOV     A,B             ;ROTATE LO
0000620' 001000  000037                         11060           RAR
0000621' 001000  000107                         11080           MOV     B,A
0000622' 001000  000035                         11100           DCR     E               ;ARE WE DONE?
0000623' 001000  000172                         11120           MOV     A,D>            ;GET NUMBER WE ARE MULTIPLYING BY
```

```
000624' 001000  000302          11140           JNZ     FMULT4          ;MULTIPLY AGAIN IF WE ARE NOT DONE
000625' 000000  000571'
000626' 000000  000601'
                                11160   IFN     LENGTH,<
000627' 001000  000353          11180           XCHG>                   ;GET LO'S IN (HL)
000630' 001000  000341          11200   POPMRT: POP     H               ;GET POINTER TO NUMBER TO MULTIPLY BY
000631' 001000  000311          11220           RET                     ;ALL DONE
                                11240   IFN     LENGTH,<
000632' 001000  000103          11260   FMULT3: MOV     B,E             ;MULTIPLY BY ZERO: SHIFT EVERYTHING 8 RIGHT
000633' 001000  000132          11280           MOV     E,D
000634' 001000  000121          11300           MOV     D,C
000635' 001000  000117          11320           MOV     C,A             ;SHIFT IN 8 ZEROS ON THE LEFT
000636' 001000  000311          11340           RET>                    ;ALL DONE

                                11400           ;DIVIDE FAC BY 10
                                11420           ;ALTERS A,B,C,D,E,H,L
000637' 001000  000315          11440   DIV10:  CALL    PUSHF           ;SAVE NUMBER
000640' 000000  001205'
000641' 000000  000625'
                                11460   IFN     LENGTH-2,<
000642' 001000  000001          11480           MOVRI   204,040,000,000 ;LOAD CONSTANT '10' INTO REGISTERS
000643' 000000  000040
000644' 000000  000204
000645' 001000  000021
000646' 000000  000000
000647' 000000  000000
000650' 001000  000315          11500           CALL    MOVFR>          ;MOVE THE CONSTANT TO THE FAC
000651' 000000  001225'
000652' 000000  000640'
                                11520   IFE     LENGTH-2,<
                                11540           LXI     H,FTEN          ;GET POINTER TO THE CONSTANT '10'
                                11560           CALL    MOVFM>          ;MOVE TEN INTO THE FAC
000653' 001000  000301          11580   FDIVT:  POPR                    ;GET NUMBER BACK IN REGISTERS
000654' 001000  000321
                                11600                                   ;FALL INTO DIVIDE AND WE ARE DONE

                                11660           ;DIVISION       FAC:=ARG/FAC
                                11680           ;ALTERS A,B,C,D,E,H,L
000655' 001000  000357          11700   FDIV:   FSIGN                   ;CHECK FOR DIVISION BY ZERO
000656' 001000  000312          11720           JZ      DV0ERR          ;HE IS TRYING TO GET AWAY WITH IT
000657' 000000  000000*
000660' 000000  000651'
000661' 001000  000056          11740           MVI     L,377           ;SUBTRACT THE TWO EXPONENTS, L IS A FLAG
000662' 000000  000377
000663' 001000  000315          11760           CALL    MULDIV          ;FIX UP THE EXPONENTS AND THINGS
000664' 000000  001035'
000665' 000000  000657'
000666' 001000  000064          11780           INR     M               ;ADD 2 TO EXPONENT TO CORRECT SCALING
000667' 001000  000064          11800           INR     M
                                11820           ;HERE WE SAVE THE FAC IN MEMORY SO WE CAN SUBTRACT IT FROM THE NUMBER
                                11840           ;IN THE REGISTERS QUICKLY.
```



```
000670' 001000  000053          11860           DCX     H               ;POINT TO MO
000671' 001000  000176          11880           MOV     A,M             ;GET MO
000672' 001000  000062          11900           STA     FDIVA+1         ;SAVE IT
000673' 000000  000734'
000674' 000000  000664'
000675' 001000  000053          11920           DCX     H               ;SAVE MIDDLE ORDER
000676' 001000  000176          11940           MOV     A,M
000677' 001000  000062          11960           STA     FDIVB+1         ;PUT IT WHERE NOTHING WILL HURT IT
000700' 000000  000730'
000701' 000000  000673'
000702' 001000  000053          11980           DCX     H               ;SAVE LO
000703' 001000  000176          12000           MOV     A,M
000704' 001000  000062          12020           STA     FDIVC+1
000705' 000000  000724'
000706' 000000  000700'

                                12060   ;THE NUMERATOR WILL BE KEPT IN B,H,L.  THE QUOTIENT WILL BE FORMED IN C,D,E,
                                12080   ;TO GET A BIT OF THE QUOTIENT, WE FIRST SAVE B,H,L ON THE STACK, THEN
                                12100   ;SUBTRACT THE DENOMINATOR THAT WE SAVED IN MEMORY.  THE CARRY INDICATES
                                12120   ;WHETHER OR NOT B,H,L WAS BIGGER THAN THE DENOMINATOR.  IF B,H,L WAS BIGGER,
                                12140   ;THE NEXT BIT OF THE QUOTIENT IS A ONE.  TO GET THE OLD B,H,L OFF THE STACK,
                                12160   ;WE POP THEM INTO THE PSW.  IF THE DENOMINATOR WAS BIGGER, THE NEXT BIT OF
                                12180   ;THE QUOTIENT IS ZERO, AND WE GET THE OLD B,H,L BACK BY POPPING IT OFF THE
                                12200   ;STACK.  WE HAVE TO KEEP AN EXTRA BIT OF THE QUOTIENT IN FDIVG+1 IN CASE THE
                                12220   ;DENOMINATOR WAS BIGGER.  THEN B,H,L WILL GET SHIFTED LEFT.  IF THE MSB  OF
                                12240   ;B WAS ONE, IT HAS TO BE STORED SOMEWHERE, SO WE STORE IT IN FDIVG+1.  THEN
                                12260   ;THE NEXT TIME THROUGH THE LOOP B,H,L WILL LOOK BIGGER BECAUSE IT HAS AN
                                12280   ;EXTRA MO BIT IN FDIVG+1, WE ARE DONE DIVIDING WHEN THE MSB OF C IS A ONE.
                                12300   ;THIS OCCURS WHEN WE HAVE CALCULATED 24 BITS OF THE QUOTIENT.  WHEN WE JUMP
                                12320   ;TO ROUND, THE 25TH BIT OF THE QUOTIENT DETERMINES WHETHER WE ROUND OR NOT.
                                12340   ;IT IS IN THE MSB OF A.  IF INITIALLY THE DENOMINATOR IS BIGGER THAN THE
                                12360   ;NUMERATOR, THE FIRST BIT OF THE QUOTIENT WILL BE ZERO.  THIS MEANS WE
                                12380   ;WILL GO THROUGH THE DIVIDE LOOP 26 TIMES, SINCE IT STOPS ON THE 25TH BIT
                                12400   ;AFTER THE FIRST NON-ZERO BIT OF THE EXPONENT.  SO, THIS QUOTIENT WILL LOOK
                                12420   ;SHIFTED LEFT ONE FROM THE QUOTIENT OF TWO NUMBERS IN WHICH THE NUMERATOR IS
                                12440   ;BIGGER.  THIS CAN ONLY OCCUR ON THE FIRST TIME THROUGH THE LOOP, SO C,D,E
                                12460   ;ARE ALL ZERO.  SO, IF WE FINISH THE LOOP AND C,D,E ARE ALL ZERO, THEN WE
                                12480   ;MUST DECREMENT THE EXPONENT TO CORRECT FOR THIS.
000707' 001000  000101          12500           MOV     B,C             ;GET NUMBER IN B,H,L
000710' 001000  000353          12520           XCHG
000711' 001000  000257          12540           XRA     A               ;ZERO C,D,E AND HIGHEST ORDER
000712' 001000  000117          12560           MOV     C,A
000713' 001000  000127          12580           MOV     D,A
000714' 001000  000137          12600           MOV     E,A
000715' 001000  000062          12620           STA     FDIVG+1
000716' 000000  000737'
000717' 000000  000705'
000720' 001000  000345          12640   FDIV1:  PUSH    H               ;SAVE LO'S OF NUMBER
000721' 001000  000305          12660           PUSH    B               ;SAVE MO OF NUMBER
000722' 001000  000175          12680           MOV     A,L             ;SUBTRACT NUMBER THAT WAS IN FAC
000723' 001000  000326          12700   FDIVC:  SUI     0               ;SUBTRACT LO
000724' 000000  000000
000725' 001000  000157          12720           MOV     L,A             ;SAVE IT
```

```
  879  000726' 001000  000174          12740          MOV     A,H             ;SUBTRACT MIDDLE ORDER
  880  000727' 001000  000336          12760  FDIVB:  SBI     0
  881  000730' 000000  000000
  882  000731' 001000  000147          12780          MOV     H,A
  883  000732' 001000  000170          12800          MOV     A,B             ;SUBTRACT HO
  884  000733' 001000  000336          12820  FDIVA:  SBI     0
  885  000734' 001000  000000
  886  000735' 001000  000107          12840          MOV     B,A
  887  000736' 001000  000076          12860  FDIVG:  MVI     A,0             ;GET HIGHEST ORDER
  888  000737' 000000  000000
  889  000740' 001000  000336          12880          SBI     0               ;SUBTRACT THE CARRY FROM IT
  890  000741' 000000  000000
  891  000742' 001000  000077          12900          CMC                     ;SET CARRY TO CORESPOND TO NEXT QUOTIENT BIT
  892  000743' 001000  000322          12920          JNC     FDIV2           ;GET OLD NUMBER BACK IF WE SUBTRACTED TOO MUCH
  893  000744' 000000  000755'
  894  000745' 000000  000716'
  895  000746' 001000  000062          12940          STA     FDIVG+1         ;UPDATE HIGHEST ORDER
  896  000747' 000000  000737'
  897  000750' 000000  000744'
  898  000751' 001000  000361          12960          POP     PSW             ;THE SUBTRACTION WAS GOOD
  899  000752' 001000  000361          12980          POP     PSW             ;GET PREVIOUS NUMBER OFF STACK
  900  000753' 001000  000067          13000          STC                     ;NEXT BIT IN QUOTIENT IS A ONE
  901  000754' 001000  000322          13020          XNO     1000,322        ;"JNC" AROUND NEXT 2 BYTES
  902  000755' 001000  000301          13040  FDIV2:  POP     B               ;WE SUBTRACTED TOO MUCH
  903  000756' 001000  000341          13060          POP     H               ;GET OLD NUMBER BACK
  904  000757' 001000  000171          13080          MOV     A,C             ;ARE WE DONE?
  905  000760' 001000  000074          13100          INR     A               ;SET SIGN FLAG WITHOUT AFFECTING CARRY
  906  000761' 001000  000075          13120          DCR     A
  907  000762' 001000  000037          13140          RAR                     ;PUT CARRY IN MSB
  908  000763' 001000  000372          13160          JM      ROUNDB          ;WE ARE DONE
  909  000764' 000000  000234'
  910  000765' 000000  000747'
  911  000766' 001000  000027          13180          RAL                     ;WE AREN'T, GET OLD CARRY BACK
  912                                   13200  IFE     LENGTH,<
  913                                   13220          CALL    SHFTLO>         ;ROTATE EVERYTHING LEFT ONE
  914                                   13240  IFN     LENGTH,<
  915  000767' 001000  000173          13260          MOV     A,E             ;ROTATE EVERYTHING LEFT ONE
  916  000770' 001000  000027          13280          RAL                     ;ROTATE NEXT BIT OF QUOTIENT IN
  917  000771' 001000  000137          13300          MOV     E,A
  918  000772' 001000  000172          13320          MOV     A,D
  919  000773' 001000  000027          13340          RAL
  920  000774' 001000  000127          13360          MOV     D,A
  921  000775' 001000  000171          13380          MOV     A,C
  922  000776' 001000  000027          13400          RAL
  923  000777' 001000  000117          13420          MOV     C,A>
  924  001000' 001000  000051          13440          DAD     H               ;ROTATE A ZERO INTO RIGHT END OF NUMBER
  925  001001' 001000  000170          13460          MOV     A,B             ;THE HO BYTE, FINALLY!
  926  001002' 001000  000027          13480          RAL
  927  001003' 001000  000107          13500          MOV     B,A
  928  001004' 001000  000072          13520          LDA     FDIVG+1         ;ROTATE THE HIGHEST ORDER
  929  001005' 000000  000737'
  930  001006' 000000  000764'
  931  001007' 001000  000027          13540          RAL
```



```
  932  001010' 001000  000062          13560          STA     FDIVG+1
  933  001011' 000000  000737'
  934  001012' 000000  001025'
  935  001013' 001000  000171          13580          MOV     A,C             ;ADD ONE TO EXPONENT IF THE FIRST SUBTRACTION
  936  001014' 001000  000262          13600          ORA     D               ; DID NOT WORK
  937  001015' 001000  000263          13620          ORA     E
  938  001016' 001000  000302          13640          JNZ     FDIV1           ;THIS ISN'T THE CASE
  939  001017' 000000  000720'
  940  001020' 000000  001011'
  941  001021' 001000  000345          13660          PUSH    H               ;SAVE PART OF NUMBER
  942  001022' 001000  000041          13680          LXI     H,FAC           ;GET POINTER TO FAC
  943  001023' 000000  000426*
  944  001024' 000000  001017'
  945  001025' 001000  000065          13700          DCR     M               ;DECREMENT EXPONENT
  946  001026' 001000  000341          13720          POP     H               ;GET NUMBER BACK
  947  001027' 001000  000302          13740          JNZ     FDIV1           ;DIVIDE MORE IF NO OVERFLOW OCCURED
  948  001030' 000000  000720'
  949  001031' 000000  001023'
  950  001032' 001000  000303          13760          JMP     OVERR           ;OVERFLOW!!
  951  001033' 000000  000267'
  952  001034' 000000  001030'
  953
  954
  955                                   13820          ;CHECK SPECIAL CASES AND ADD EXPONENTS FOR FMULT, FDIV
  956                                   13840          ;ALTERS A,B,H,L
  957                                   13860  IFE     LENGTH-2,<
  958                                   13880  MULDVS: MVI     A,377           ;ENTRY FROM DDIV, SUBTRACT EXPONENTS
  959                                   13900          XWD     1000,056        ;"MVI  L" AROUND NEXT BYTE
  960                                   13920  MULDVA: XRA     A               ;ENTRY FROM DMULT, ADD EXPONENTS
  961                                   13940          LXI     H,ARG-1         ;GET POINTER TO SIGN AND HO OF ARG
  962                                   13960          MOV     C,M             ;GET HO AND SIGN FOR UNPACKING
  963                                   13980          INX     H               ;INCREMENT POINTER TO EXPONENT
  964                                   14000          MOV     B,M             ;GET EXPONENT FOR BELOW
  965                                   14020          MOV     L,A>            ;SAVE ADD OR SUBTRACT FLAG
  966  001035' 001000  000170          14040  MULDIV: MOV     A,B             ;IS NUMBER IN REGISTERS ZERO?
  967  001036' 001000  000267          14060          ORA     A
  968  001037' 001000  000312          14080          JZ      MULDV2          ;IT IS, ZERO FAC AND WE ARE DONE
  969  001040' 000000  001077'
  970  001041' 000000  001033'
  971  001042' 001000  000175          14100          MOV     A,L             ;GET ADD OR SUBTRACT FLAG
  972  001043' 001000  000041          14120          LXI     H,FAC           ;GET POINTER TO EXPONENT
  973  001044' 000000  001023*
  974  001045' 000000  001040'
  975  001046' 001000  000256          14140          XRA     M               ;GET EXPONENT
  976  001047' 001000  000200          14160          ADD     B               ;ADD IN REGISTER EXPONENT
  977  001050' 001000  000107          14180          MOV     B,A             ;SAVE IT
  978  001051' 001000  000037          14200          RAR                     ;CHECK FOR OVERFLOW
  979  001052' 001000  000250          14220          XRA     B               ;OVERFLOW IF SIGN IS THE SAME AS CARRY
  980  001053' 001000  000170          14240          MOV     A,B             ;GET SUM
  981  001054' 001000  000362          14260          JP      MULDV1          ;WE HAVE OVERFLOW!!
  982  001055' 000000  001076'
  983  001056' 000000  001044'
  984  001057' 001000  000306          14280          ADI     200             ;PUT EXPONENT IN EXCESS 200
```

```
  985  001060' 000000  000200
  986  001061' 001000  000167          14300            MOV     M,A             ;SAVE IT IN THE FAC
  987  001062' 001000  000316          14320            JZ      POPHRT          ;WE HAVE UNDEFLOW!! RETURN.
  988  001065' 000000  000630'
  989  001064' 000000  001055'
  990  001065' 001000  000315          14340            CALL    UNPACK          ;UNPACK THE ARGUMENTS
  991  001066' 000000  001272'
  992  001067' 000000  001063'
  993  001070' 001000  000167          14360            MOV     M,A             ;SAVE THE NEW SIGN
  994  001071' 001000  000053          14380            DCX     H               ;POINT TO EXPONENT
  995  001072' 001000  000311          14400            RET                     ;ALL DONE, LEAVE NO IN A
  996                                   14420  IFN       EXTFNC,<
  997  001073' 001000  000357          14440  MLOVEX:   FSIGN                   ;ENTRY FROM EXP, PICK UNDERFLOW IF NEGATIVE
  998  001074' 001000  000057          14460            CMA                     ;PICK OVERFLOW IF POSITIVE
  999  001075' 001000  000341          14480            POP     H>              ;DON'T SCREW UP STACK
 1000  001076' 001000  000267          14500  MULOV1:   ORA     A               ;IS ERROR OVERFLOW OR UNDEFLOW?
 1001  001077' 001000  000341          14520  MULOV2:   POP     H               ;GET OLD RETURN ADDRESS OFF STACK
 1002                                   14540  IFE       LENGTH,<
 1003                                   14560            JM      OVERR           ;OVERFLOW
 1004                                   14580                                    ;UNDERFLOW -- FALL INTO ZERO
 1005                                   14600
 1006                                   14620
 1007                                   14640            ;ZERO FAC
 1008                                   14660            ;ALTERS A ONLY
 1009                                   14680            ;EXITS WITH A=0
 1010                                   14700  ZERO:     XRA     A               ;ZERO A
 1011                                   14720            STA     FAC             ;ZERO FAC
 1012                                   14740            RET>                    ;ALL DONE
 1013
 1014
 1015                                   14800  IFN       LENGTH,<
 1016  001100' 001000  000362          14820            JP      ZERO            ;UNDERFLOW
 1017  001101' 000000  000173'
 1018  001102' 000000  001066'
 1019  001103' 001000  000303          14840            JMP     OVERR>          ;OVERFLOW
 1020  001104' 000000  000267'
 1021  001105' 000000  001101'
 1022
 1023
 1024                                   14900            ;MULTIPLY FAC BY 10
 1025                                   14920            ;ALTERS A,B,C,D,E,M,L
 1026  001106' 001000  000315          14940  MUL10:    CALL    MOVRF           ;GET NUMBER IN REGISTERS
 1027  001107' 000000  001240'
 1028  001110' 000000  001104'
 1029  001111' 001000  000170          14960            MOV     A,B             ;GET EXPONENT
 1030  001112' 001000  000267          14980            ORA     A               ;RESULT IS ZERO IF ARG IS ZERO
 1031  001113' 001000  000310          15000            RZ                      ;IT IS
 1032  001114' 001000  000306          15020            ADI     2               ;MULTIPLY BY 4 BY ADDING 2 TO EXPONENT
 1033  001115' 000000  000002
 1034  001116' 001000  000332          15040            JC      OVERR           ;OVERFLOW!!
 1035  001117' 000000  000267'
 1036  001120' 000000  001107'
 1037  001121' 001000  000107          15060            MOV     B,A             ;RESTORE EXPONENT
```



```
 1038  001122' 001000  000315          15080            CALL    FADD            ;ADD IN ORIGINAL NUMBER TO GET 5 TIMES IT
 1039  001123' 000000  000025'
 1040  001124' 000000  001117'
 1041  001125' 001000  000041          15100            LXI     H,FAC           ;ADD 1 TO EXPONENT TO MULTIPLY NUMBER BY
 1042  001126' 000000  001044*
 1043  001127' 000000  001125'
 1044  001130' 001000  000064          15120            INR     M               ; 2 TO GET 10 TIMES ORIGINAL NUMBER
 1045  001131' 001000  000300          15140            RNZ                     ;ALL DONE IF NO OVERFLOW
 1046  001132' 001000  000303          15160            JMP     OVERR           ;OVERFLOW!!
 1047  001133' 000000  000267'
 1048  001134' 000000  001126'
 1049                                   15180  PAGE
```

```
                                        15200   SUBTTL  SIGN, SGN, FLOAT, NEG AND ABS
                                        15220           ;PUT SIGN OF FAC IN A
                                        15240           ;ALTERS A ONLY
                                        15260           ;LEAVES FAC ALONE
                                        15280           ;NOTE: TO TAKE ADVANTAGE OF THE RST INSTRUCTIONS TO SAVE BYTES,
                                        15300           ;FSIGN IS DEFINED TO BE AN RST.  "FSIGN" IS EQUIVALENT TO "CALL SIGN"
                                        15320           ;THE FIRST FEW INSTRUCTIONS OF SIGN (THE ONES BEFORE SIGNC) ARE DONE
                                        15340           ;IN THE 8 BYTES AT THE RST LOCATION.
                                        15360   REPEAT  0,<                      ;FSIGN IS ALWAYS AN RST
                                        15380   SIGN:   LDA     FAC             ;CHECK IF THE NUMBER IS ZERO
                                        15400           ORA     A
                                        15420           RZ>                     ;IT IS, A IS ZERO
001135' 001000  000072                  15440   SIGNC:  LDA     FAC-1           ;GET SIGN OF FAC, IT IS NON-ZERO
001136' 777777  777777*
001137' 000000  001133'
                                        15460           XWD     1000,376        ;"CPI" AROUND NEXT BYTE
001140' 001000  000057                  15480   FCOMPS: CMA                     ;ENTRY FROM FCOMP, COMPLEMENT SIGN
001142' 001000  000027                  15500   ICOMPS: RAL                     ;ENTRY FROM ICOMP, PUT SIGN BIT IN CARRY
001143' 001000  000237                  15520   SIGNS:  SBB     A               ;A=0 IF CARRY WAS 0, A=377 IF CARRY WAS 1
001144' 001000  000300                  15540           RNZ                     ;RETURN IF NUMBER WAS NEGATIVE
001145' 001000  000074                  15560   INRART: INR     A               ;PUT ONE IN A IF NUMBER WAS POSITIVE
001146' 001000  000311                  15580           RET                     ;ALL DONE
                                        15600
                                        15620           ;SGN FUNCTION
                                        15640           ;ALTERS A,B,C,D,E,H,L
                                        15660   IFN     LENGTH=2,<
001147' 001000  000357                  15700   SGN:    FSIGN>                  ;GET SIGN OF FAC IN A
                                        15720                                   ;FALL INTO FLOAT
                                        15740
                                        15760
                                        15780           ;FLOAT THE SIGNED INTEGER IN A
                                        15800           ;ALTERS A,B,C,D,E,H,L
001150' 001000  000006                  15820   FLOAT:  MVI     B,210           ;SET EXPONENT CORRECTLY
001151' 000000  000210
001152' 001000  000021                  15840           LXI     D,$CODE         ;ZERO D,E
001153' 000000  000601*
001154' 000000  001136'
                                        15860                                   ;FALL INTO FLOATR
                                        15880
                                        15900
                                        15920           ;FLOAT THE SIGNED NUMBER IN B,A,D,E
                                        15940           ;ALTERS A,B,C,D,E,H,L
001155' 001000  000041                  15960   FLOATR: LXI     H,FAC           ;GET POINTER TO FAC
001156' 000000  001126*
001157' 000000  001153'
001160' 001000  000117                  15980           MOV     C,A             ;PUT MO IN C
001161' 001000  000160                  16000           MOV     M,B             ;PUT EXPONENT IN THE FAC
001162' 001000  000006                  16020           MVI     B,0             ;ZERO OVERFLOW BYTE
001163' 000000  000000
001164' 001000  000043                  16040           INX     H               ;POINT TO SIGN
001165' 001000  000066                  16060           MVI     M,200           ;ASSUME A POSITIVE NUMBER
001166' 000000  000200
```



```
001167' 001000  000027                  16080           RAL                     ;PUT SIGN IN CARRY
001170' 001000  000303                  16100           JMP     FADFLT          ;GO AND FLOAT THE NUMBER
001171' 000000  000143'
001172' 000000  001156'
                                        16120
                                        16140
                                        16160           ;ABSOLUTE VALUE OF FAC
                                        16180           ;ALTERS A,H,L
001173'                                 16200   ABS:
                                        16220   IFE     LENGTH=2,<
                                        16240           CPI     2               ;IS THE ARGUMENT AN INTEGER?
                                        16260           JZ      IABS>           ;YES, USE THE INTEGER ABSOLUTE VALUE
001173' 001000  000357                  16280           FSIGN                   ;GET THE SIGN OF FAC
001174' 001000  000360                  16300           RP                      ;ALL DONE IF IT IS POSITIVE
                                        16320                                   ;FALL INTO NEG
                                        16340
                                        16360           ;NEGATE NUMBER IN THE FAC
                                        16400           ;ALTERS A,H,L
                                        16420           ;NOTE: THE NUMBER MUST BE PACKED
001175' 001000  000041                  16440   NEG:    LXI     H,FAC+1         ;GET POINTER TO SIGN
001176' 777777  777777*
001177' 000000  001171'
001200' 001000  000176                  16460           MOV     A,M             ;GET SIGN
001201' 001000  000356                  16480           XRI     200             ;COMPLEMENT SIGN BIT
001202' 000000  000200
001203' 001000  000167                  16500           MOV     M,A             ;SAVE IT
001204' 001000  000311                  16520           RET                     ;ALL DONE
                                        16540
                                        16560
                                        16580   IFE     LENGTH=2,<
                                        16600           ;NEGATE ANY TYPE VALUE IN THE FAC
                                        16620           ;ALTERS A,B,C,D,E,H,L
                                        16640   VNEG:   LDA     VALTYP          ;SEE WHAT KIND OF NUMBER WE HAVE
                                        16660           CPI     2
                                        16680           JZ      INEG            ;WE HAVE AN INTEGER, NEGATE IT THAT WAY
                                        16700           JM      TMERR           ;BLOW UP ON STRINGS
                                        16720           JMP     NEG             ;NEGATE SNG AND DBL THE SAME
                                        16740
                                        16760
                                        16780           ;SGN FUNCTION
                                        16800           ;ALTERS A,H,L
                                        16820   SGN:    CALL    VSIGN           ;GET THE SIGN OF THE FAC IN A
                                        16840           MOV     L,A             ;PUT IT IN THE LO POSITION
                                        16860           RAL                     ;EXTEND THE SIGN TO THE HO
                                        16880           SBB     A
                                        16900           MOV     H,A
                                        16920           JMP     CONISS          ;RETURN THE RESULT AND SET VALTYP
                                        16940
                                        16960
                                        16980           ;GET THE SIGN OF THE VALUE IN THE FAC IN A
                                        17000           ;ASSUMES A HAS THE VALTYP WHEN CALLED
                                        17020           ;ALTERS A,H,L
```

```
                                17040   VSIGN:  CPI     2               ;IS THE ARGUMENT AN INTEGER?
                                17060           JNZ     SIGN            ;NO, SINGLE AND DOUBLE PREC, WORK THE SAME
                                17080           LHLD    FACLO           ;GET THE INTEGER ARGUMENT
                                17100           MOV     A,H             ;GET ITS SIGN
                                17120           ORA     L               ;CHECK IF THE NUMBER IS ZERO
                                17140           RZ                      ;IT IS, WE ARE DONE
                                17160           MOV     A,H             ;IT ISN'T, SIGN IS THE SIGN OF H
                                17180           JMP     ICOMPS>         ;GO SET A CORRECTLY
                                17200   PAGE
```



```
                                17220   SUBTTL  FLOATING POINT MOVEMENT ROUTINES
                                17240           ;PUT FAC ON STACK
                                17260           ;ALTERS D,E
001205' 001000  000353          17280   PUSHF:  XCHG                    ;SAVE (HL)
001206' 001000  000052          17300           LHLD    FACLO           ;GET LO'S
001207' 000000  000555*
001210' 000000  001176'
001211' 001000  000343          17320           XTHL                    ;SWITCH LO'S AND RET ADDR
001212' 001000  000345          17340           PUSH    H               ;PUT RET ADDR BACK ON STACK
001213' 001000  000052          17360           LHLD    FAC-1           ;GET MO'S
001214' 777777  777777*
001215' 000000  001207'
001216' 001000  000343          17380           XTHL                    ;SWITCH MO'S AND RET ADDR
001217' 001000  000345          17400           PUSH    H               ;PUT RET ADDR BACK ON STACK
001220' 001000  000353          17420           XCHG                    ;GET OLD (HL) BACK
001221' 001000  000311          17440           RET                     ;ALL DONE

                                17500           ;MOVE NUMBER FROM MEMORY [(HL)] TO FAC
                                17520           ;ALTERS B,C,D,E,H,L
                                17540           ;AT EXIT NUMBER IS IN B,C,D,E
                                17560           ;AT EXIT (HL):=(HL)+4
001222' 001000  000315          17580   MOVFM:  CALL    MOVRM           ;GET NUMBER IN REGISTERS
001223' 000000  001243'
001224' 000000  001214'
                                17600                                   ;FALL INTO MOVFR AND PUT IT IN FAC


                                17660           ;MOVE REGISTERS (B,C,D,E) TO FAC
                                17680           ;ALTERS D,E
001225' 001000  000353          17700   MOVFR:  XCHG                    ;GET LO'S IN (HL)
001226' 001000  000042          17720           SHLD    FACLO           ;PUT THEM WHERE THEY BELONG
001227' 000000  001207*
001230' 000000  001223'
001231' 001000  000140          17740           MOV     H,B             ;GET MO'S IN (HL)
001232' 001000  000151          17760           MOV     L,C
001233' 001000  000042          17780           SHLD    FAC-1           ;PUT MO'S WHERE THEY BELONG
001234' 777777  777777*
001235' 000000  001227'
001236' 001000  000353          17800           XCHG                    ;GET OLD (HL) BACK
001237' 001000  000311          17820           RET                     ;ALL DONE


                                17880           ;MOVE FAC TO REGISTERS (B,C,D,E)
                                17900           ;ALTERS B,C,D,E,H,L
001240' 001000  000041          17920   MOVRF:  LXI     H,FACLO         ;GET POINTER TO FAC
001241' 000000  001227*
001242' 000000  001234'
                                17940                                   ;FALL INTO MOVRM


                                18000           ;GET NUMBER IN REGISTERS (B,C,D,E) FROM MEMORY [(HL)]
                                18020           ;ALTERS B,C,D,E,H,L
```

```
                                        ;AT EXIT (HL)+=(HL)+4
001243' 001000  000136          MOVRM:  MOV     E,M             ;GET LO
001244' 001000  000043                  INX     H               ;POINT TO MO
001245' 001000  000126                  MOV     D,M             ;GET MO
001246' 001000  000043                  INX     H               ;POINT TO MO
001247' 001000  000116                  MOV     C,M             ;GET MO
001250' 001000  000043                  INX     H               ;POINT TO EXPONENT
001251' 001000  000106                  MOV     B,M             ;GET EXPONENT
001252' 001000  000043          INXHRT: INX     H               ;INC POINTER TO BEGINNING OF NEXT NUMBER
001253' 001000  000311                  RET                     ;ALL DONE


                                        ;MOVE NUMBER FROM FAC TO MEMORY [(HL)]
                                        ;ALTERS A,B,D,E,H,L
001254' 001000  000021          MOVMF:  LXI     D,FACLO         ;GET POINTER TO FAC
001255' 000000  001241*
001256' 000000  001241'
                                        ;FALL INTO MOVE


                                        ;MOVE NUMBER FROM (DE) TO (HL)
                                        ;ALTERS A,B,D,E,H,L
                                        ;EXITS WITH (DE)+=(DE)+4, (HL)+=(HL)+4
001257' 001000  000006          MOVE:   MVI     B,4             ;SET COUNTER
001260' 000000  000004
                                IFE     LENGTH-2,<
                                        XWD     1000,076        ;"MVI   A" OVER NEXT BYTE
                                MOVVFM: XCHG>                   ;MOVE NUMBERS INTO THE FAC
001261' 001000  000032          MOVE1:  LDAX    D               ;GET WORD, ENTRY FROM VMOVMF
001262' 001000  000167                  MOV     M,A             ;PUT IT WHERE IT BELONGS
001263' 001000  000023                  INX     D               ;INCREMENT POINTERS TO NEXT WORD
001264' 001000  000043                  INX     H
001265' 001000  000005                  DCR     B               ;SEE IF DONE
001266' 001000  000302                  JNZ     MOVE1
001267' 000000  001261'
001270' 000000  001255'
001271' 001000  000311                  RET


                                        ;UNPACK THE FAC AND THE REGISTERS
                                        ;ALTERS A,C,H,L
                                        ;WHEN THE NUMBER IN THE FAC IS UNACKED, THE ASSUMED ONE IN THE
                                        ;MANTISSA IS RESTORED, AND THE COMPLEMENT OF THE SIGN IS PLACED
                                        ;IN FAC+1
001272' 001000  000041          UNPACK: LXI     H,FAC-1         ;POINT TO MO AND SIGN
001273' 777777  777777*
001274' 000000  001267'
001275' 001000  000176                  MOV     A,M             ;GET MO AND SIGN
001276' 001000  000365                  PUSH    PSW             ;SAVE SIGN
001277' 001000  000366                  ORI     200             ;RESTORE THE HIDDEN ONE
001300' 000000  000200
001301' 001000  000167                  MOV     M,A             ;SAVE MO
001302' 001000  000361                  POP     PSW             ;GET SIGN
```



```
001303' 001000  000256          XRA     M               ;GET COMPLEMENT OF SIGN IN MSB
001304' 001000  000043          INX     H               ;POINT TO TEMPORARY SIGN BYTE
001305' 001000  000043          INX     H
001306' 001000  000167          MOV     M,A             ;SAVE COMPLEMENT OF SIGN
001307' 001000  000171          MOV     A,C             ;GET MO AND SIGN OF THE REGISTERS
001310' 001000  000365          PUSH    PSW             ;SAVE SIGN
001311' 001000  000366          ORI     200             ;RESTORE THE HIDDEN ONE
001312' 000000  000200
001313' 001000  000117          MOV     C,A             ;SAVE THE MO
001314' 001000  000361          POP     PSW             ;GET THE SIGN BACK
001315' 001000  000256          XRA     M               ;COMPARE SIGN OF FAC AND SIGN OF REGISTERS
001316' 001000  000311          RET                     ;ALL DONE


                                IFE     LENGTH-2,<
                                REPEAT  0,<             ;VPUSHF WILL BE IN-LINE IN F3
                                        ;PUT ANY TYPE VALUE ON THE STACK FROM THE FAC
                                        ;STRINGS ARE TREATED AS INTEGERS
                                        ;ALTERS A,B,C,H,L
                                VPUSHF: LDA     VALTYP          ;GET THE VALUE TYPE
                                        CPI     4               ;SET FLAGS ACCORDING TO VALTYP
                                        LXI     H,FACLO         ;GET POINTER TO LO IN FAC
                                        PUSHM                   ;PUSH FACLO+0,1 ON THE STACK
                                        JM      VPUSHD          ;RETURN IF THE DATA WAS AN INTEGER OR A STRING
                                        PUSHM                   ;PUSH FAC+1,0 ON THE STACK
                                        JZ      VPUSHD          ;RETURN IF WE HAD A SINGLE PRECISION NUMBER
                                        LXI     D,DFACLO        ;WE HAVE A DOUBLE PRECISON NUMBER
                                        PUSHM                   ;PUSH ITS 4 LO BYTES ON THE STACK
                                        PUSHM
                                VPUSHD:>                        ;ALL DONE


                                        ;MOVE ANY TYPE VALUE FROM MEMORY [(HL)] TO FAC
                                        ;ALTERS A,B,D,E,H,L
                                VMOVFA: LXI     H,ARGLO         ;ENTRY FROM DADD, MOVE ARG TO FAC
                                VMOVFM: LXI     D,MOVVFM        ;GET ADDRESS OF LOCATION THAT DOES
                                        JMP     VMVVFM          ; AN "XCHG" AND FALLS INTO MOVE1


                                        ;MOVE ANY TYPE VALUE FROM FAC TO MEMORY [(HL)]
                                        ;ALTERS A,B,D,E,H,L
                                VMOVAF: LXI     H,ARGLO         ;ENTRY FROM FIN, DMUL10, DDIV10
                                        ;MOVE FAC TO ARG
                                VMOVMF: LXI     D,MOVE1         ;GET ADDRESS OF MOVE SUBROUTINE
                                VMVVFM: PUSH    D               ;SHOVE IT ON THE STACK
                                        LXI     D,FACLO         ;GET FIRST ADDRESS FOR INT, SNG
                                        LDA     VALTYP          ;GET THE VALUE TYPE
                                        ANI     177             ;STRINGS LOOK LIKE REALS
                                        MOV     B,A             ;SET UP THE COUNT
                                        CPI     10              ;DO WE HAVE DBL?
                                        RNZ                     ;WE DO NOT, GO DO THE MOVE
                                        LXI     D,DFACLO        ;WE DO, GET LO ADDR OF THE DBL NUMBER
                                        RET>                    ;GO DO THE MOVE
```

```
 1324                                   19980   PAGE
```



```
 1325                                   20000   SUBTTL  COMPARE TWO NUMBERS
 1326                                   20020           ;COMPARE TWO SINGLE PRECISION NUMBERS
 1327                                   20040           ;A=1 IF ARG ,LT, FAC
 1328                                   20060           ;A=0 IF ARG=FAC
 1329                                   20080           ;A=-1 IF ARG ,GT, FAC
 1330                                   20100           ;DOREL DEPENDS UPON THE FACT THAT FCOMP RETURNS WITH CARRY ON
 1331                                   20120           ; IFF A HAS 377
 1332                                   20140           ;ALTERS A,H,L
 1333  001317' 001000  000170           20160   FCOMP:  MOV     A,B             ;CHECK IF ARG IS ZERO
 1334  001320' 001000  000267           20180           ORA     A
 1335  001321' 001000  000512           20200           JZ      SIGN
 1336  001322' 000000  000000*
 1337  001323' 000000  001273'
 1338  001324' 001000  000041           20220           LXI     H,FCOMPS        ;WE JUMP TO FCOMPS WHEN WE ARE DONE
 1339  001325' 000000  001141'
 1340  001326' 000000  001322'
 1341  001327' 001000  000345           20240           PUSH    H               ;PUT THE ADDRESS ON THE STACK
 1342  001330' 001000  000357           20260           FSIGN                   ;CHECK IF FAC IS ZERO
 1343  001331' 001000  000171           20280           MOV     A,C             ;IF IT IS, RESULT IS MINUS THE SIGN OF ARG
 1344  001332' 001000  000510           20300           RZ                      ;IT IS
 1345  001333' 001000  000041           20320           LXI     H,FAC+1         ;POINT TO SIGN OF FAC
 1346  001334' 777777  777777*
 1347  001335' 000000  001325'
 1348  001336' 001000  000256           20340           XRA     M               ;SEE IF THE SIGNS ARE THE SAME
 1349  001337' 001000  000171           20360           MOV     A,C             ;IF THEY ARE DIFFERENT, RESULT IS SIGN OF ARG
 1350  001340' 001000  000370           20380           RM                      ;THEY ARE DIFFERENT
 1351  001341' 001000  003315           20400           CALL    FCOMP2          ;CHECK THE REST OF THE NUMBER
 1352  001342' 000000  001347'
 1353  001343' 000000  001334'
 1354  001344' 001000  000037           20420   FCOMPD: RAR                     ;NUMBERS ARE DIFFERENT, CHANGE SIGN IF
 1355  001345' 001000  000251           20440           XRA     C               ; BOTH NUMBERS ARE NEGATIVE
 1356  001346' 001000  000311           20460           RET                     ;GO SET UP A
 1357
 1358  001347' 001000  000043           20500   FCOMP2: INX     H               ;POINT TO EXPONENT
 1359  001350' 001000  000170           20520           MOV     A,B             ;GET EXPONENT OF ARG
 1360  001351' 001000  000276           20540           CMP     M               ;COMPARE THE TWO
 1361  001352' 001000  000300           20560           RNZ                     ;NUMBERS ARE DIFFERENT
 1362  001353' 001000  000053           20580           DCX     H               ;POINT TO MO
 1363  001354' 001000  000171           20600           MOV     A,C             ;GET MO OF ARG
 1364  001355' 001000  000276           20620           CMP     M               ;COMPARE WITH MO OF FAC
 1365  001356' 001000  000300           20640           RNZ                     ;THEY ARE DIFFERENT
 1366  001357' 001000  000053           20660           DCX     H               ;POINT TO MO OF FAC
 1367  001360' 001000  000172           20680           MOV     A,D             ;GET MO OF ARG
 1368  001361' 001000  000276           20700           CMP     M               ;COMPARE WITH MO OF FAC
 1369  001362' 001000  000300           20720           RNZ                     ;THE NUMBERS ARE DIFFERENT
 1370  001363' 001000  000053           20740           DCX     H               ;POINT TO LO OF FAC
 1371  001364' 001000  000173           20760           MOV     A,E             ;GET LO OF ARG
 1372  001365' 001000  000226           20780           SUB     M               ;SUBTRACT LO OF FAC
 1373  001366' 001000  000300           20800           RNZ                     ;NUMBERS ARE DIFFERENT
 1374  001367' 001000  000341           20820           POP     H               ;NUMBERS ARE THE SAME, DON'T SCREW UP STACK
 1375  001370' 001000  000341           20840           POP     H
 1376  001371' 001000  000311           20860           RET                     ;ALL DONE
 1377
```

```
                        20920   IFE     LENGTH-2,<
                        20940           ;COMPARE TWO INTEGERS
                        20960           ;A=1 IF (DE) ,LT, (HL)
                        20980           ;A=0 IF (DE)=(HL)
                        21000           ;A=-1 IF (DE) ,GT, (HL)
                        21020           ;ALTERS A ONLY
                        21040   ICOMP:  MOV     A,D             ;ARE THE SIGNS THE SAME?
                        21060           XRA     H
                        21080           MOV     A,H             ;IF NOT, ANSWER IS THE SIGN OF (HL)
                        21100           JM      ICOMPS          ;THEY ARE DIFFERENT
                        21120           CMP     D               ;THEY ARE THE SAME, COMPARE THE HO'S
                        21140           JNZ     SIGNS           ;GO SET UP A
                        21160           MOV     A,L             ;COMPARE THE LO'S
                        21180           SUB     E
                        21200           JNZ     SIGNS           ;GO SET UP A
                        21220           RET                     ;ALL DONE, THEY ARE THE SAME
                        21240
                        21260
                        21280           ;COMPARE TWO DOUBLE PRECISION NUMBERS
                        21300           ;A=1 IF ARG ,LT, FAC
                        21320           ;A=0 IF ARG=FAC
                        21340           ;A=-1 IF ARG ,GT, FAC
                        21360           ;ALTERS A,B,C,D,E,H,L
                        21380   DCOMPD: LXI     H,ARGLO         ;ENTRY WITH POINTER TO ARG IN (DE)
                        21400           MVI     B,10            ;SET UP COUNT TO MOVE DBL NUMBERS
                        21420           CALL    MOVE1           ;MOVE THE ARGUMENT INTO ARG
                        21440   DCOMP:  LXI     D,ARG           ;GET POINTER TO ARG
                        21460           LDAX    D               ;SEE IF ARG=0
                        21480           ORA     A
                        21500           JZ      SIGN            ;ARG=0, GO SET UP A
                        21520           LXI     H,FCOMPS        ;PUSH FCOMPS ON STACK SO WE WILL RETURN TO
                        21540           PUSH    H               ; TO IT AND SET UP A
                        21560           FSIGN                   ;SEE IF FAC=0
                        21580           DCX     D               ;POINT TO SIGN OF ARG
                        21600           LDAX    D               ;GET SIGN OF ARG
                        21620           MOV     C,A             ;SAVE IT FOR LATER
                        21640           RZ                      ;FAC=0, SIGN OF RESULT IS SIGN OF ARG
                        21660           LXI     H,FAC-1         ;POINT TO SIGN OF FAC
                        21680           XRA     M               ;SEE IF THE SIGNS ARE THE SAME
                        21700           MOV     A,C             ;IF THEY ARE, GET THE SIGN OF THE NUMBERS
                        21720           RM                      ;THE SIGNS ARE DIFFERENT, GO SET A
                        21740           INX     D               ;POINT BACK TO EXPONENT OF ARG
                        21760           INX     H               ;POINT TO EXPONENT OF FAC
                        21780           MVI     B,10            ;SET UP A COUNT
                        21800   DCOMP1: LDAX    D               ;GET A BYTE FROM ARG
                        21820           SUB     M               ;COMPARE IT WITH THE FAC
                        21840           JNZ     FCOMPD          ;THEY ARE DIFFERENT, GO SET UP A
                        21860           DCX     D               ;THEY ARE THE SAME, EXAMINE THE NEXT LOWER
                        21880           DCX     H               ; ORDER BYTES
                        21900           DCR     B               ;ARE WE DONE?
                        21920           JNZ     DCOMP1          ;NO, COMPARE THE NEXT BYTES
                        21940           POP     B               ;THEY ARE THE SAME, GET FCOMPS OFF STACK
```



```
                        21960           RET>                    ;ALL DONE
                        21980   PAGE
```

```
1433                    22000  SUBTTL  CONVERSION ROUTINES BETWEEN INTEGER, SINGLE AND DOUBLE PRECISION
1434                    22020  IFE     LENGTH-2,<
1435                    22040          ;FORCE THE FAC TO BE AN INTEGER
1436                    22060          ;ALTERS A,B,C,D,E,H,L
1437                    22080  FRCINT: LDA     VALTYP          ;SEE WHAT WE HAVE
1438                    22100          CPI     4
1439                    22120          LHLD    FACLO           ;GET FACLO+0,1 IN CASE WE HAVE AN INTEGER
1440                    22140          RC                      ;WE HAVE AN INTEGER, ALL DONE
1441                    22160          JM      TMERR           ;WE HAVE A STRING, THAT IS A "NO-NO"
1442                    22180          CNZ     CONSD           ;IF WE HAVE A DBL, CONVERT IT TO A SNG
1443                    22200          LXI     H,OVERR         ;PUT OVERR ON THE STACK SO WE WILL GET ERROR
1444                    22220          PUSH    H               ; IF NUMBER IS TOO BIG
1445                    22240                                  ;FALL INTO CONIS
1446                    22260
1447                    22280
1448                    22300          ;CONVERT SINGLE PRECISION NUMBER TO INTEGER
1449                    22320          ;ALTERS A,B,C,D,E,H,L
1450                    22340  CONIS:  LDA     FAC             ;GET THE EXPONENT
1451                    22360          CPI     220             ;SEE IF IT IS TOO BIG
1452                    22380          JNC     CONIS2          ;IT IS, BUT IT MIGHT BE -32768
1453                    22400          CALL    QINT            ;IT ISN'T, CONVERT IT TO AN INTEGER
1454                    22420          XCHG                    ;PUT IT IN (HL)
1455                    22440  CONIS1: POP     D               ;GET ERROR ADDRESS OFF STACK
1456                    22460          ;ENTRY FROM SGN, FIN, LINPRT
1457                    22480  CONISS: SHLD    FACLO           ;STORE THE NUMBER IN FACLO
1458                    22500          MVI     A,2             ;SET VALTYP TO "INTEGER"
1459                    22520  CONISD: STA     VALTYP          ;ENTRY FROM CONDS
1460                    22540          RET                     ;ALL DONE
1461                    22560  CONIS2: MOVRI   220,200,000,000 ;CHECK IF NUMBER IS -32768, ENTRY FROM FIN
1462                    22580          CALL    FCOMP
1463                    22600          RNZ                     ;ERROR!  IT CAN'T BE CONVERTED TO AN INTEGER
1464                    22620          MOV     H,C             ;IT IS -32768, PUT IT IN (HL)
1465                    22640          MOV     L,D
1466                    22660          JMP     CONIS1          ;STORE IT IN THE FAC AND SET VALTYP
1467                    22680
1468                    22700
1469                    22720          ;FORCE THE FAC TO BE A SINGLE PRECISION NUMBER
1470                    22740          ;ALTERS A,B,C,D,E,H,L
1471                    22760  FRCSNG: LDA     VALTYP          ;SEE WHAT KIND OF NUMBER WE HAVE
1472                    22780          CPI     4
1473                    22800          RZ                      ;WE ALREADY HAVE AN INTEGER, ALL DONE
1474                    22820          JC      CONSI           ;WE HAVE AN INTEGER, CONVERT IT
1475                    22840          JM      TMERR           ;STRINGS!! -- ERROR!!
1476                    22860                                  ;DBL PREC -- FALL INTO CONSD
1477                    22880
1478                    22900
1479                    22920          ;CONVERT DOUBLE PRECISION NUMBER TO A SINGLE PRECISON ONE
1480                    22940          ;ALTERS A,B,C,D,E,H,L
1481                    22960  CONSD:  CALL    MOVRF           ;GET THE HO'S IN THE REGISTERS
1482                    22980          MVI     A,4             ;SET VALTYP TO "SINGLE PRECISON"
1483                    23000          STA     VALTYP
1484                    23020          MOV     A,B             ;CHECK IF THE NUMBER IS ZERO
1485                    23040          ORA     A
```



```
1486                    23060          RZ                      ;IF IT IS, WE ARE DONE
1487                    23080          CALL    UNPACK          ;UNPACK THE NUMBER
1488                    23100          LXI     H,FACLO-1       ;GET FIRST BYTE BELOW A SNG NUMBER
1489                    23120          MOV     B,M             ;PUT IT IN B FOR ROUND
1490                    23140          JMP     ROUND           ;ROUND THE DBL NUMBER UP AND WE ARE DONE
1491                    23160
1492                    23180
1493                    23200          ;CONVERT AN INTEGER TO A SINGLE PRECISION NUMBER
1494                    23220          ;ALTERS A,B,C,D,E,H,L
1495                    23240  CONSI:  LHLD    FACLO           ;GET THE INTEGER
1496                    23260  CONSIH: MVI     A,4             ;SET VALTYP TO "SINGLE PRECISION"
1497                    23280          STA     VALTYP
1498                    23300          MOV     A,H             ;SET UP REGISTERS FOR FLOATR
1499                    23320          MOV     D,L
1500                    23340          MVI     E,0
1501                    23360          MVI     B,220
1502                    23380          JMP     FLOATR          ;GO FLOAT THE NUMBER
1503                    23400
1504                    23420
1505                    23440          ;FORCE THE FAC TO BE A DOUBLE PRECISION NUMBER
1506                    23460          ;ALTERS A,B,C,D,E,H,L
1507                    23480  FRCDBL: LDA     VALTYP          ;SEE WHAT KIND OF NUMBER WE HAVE
1508                    23500          CPI     10
1509                    23520          RZ                      ;WE ALREADY HAVE A DBL, WE ARE DONE
1510                    23540          JNC     TMERR           ;GIVE AN ERROR IF WE HAVE A STRING
1511                    23560          CPI     2               ;SEE IF WE HAVE A SNG OR INT
1512                    23580          CZ      CONSI           ;CONVERT TO SNG IF WE HAVE AN INT
1513                    23600                                  ;FALL INTO CONDS AND CONVERT TO DBL
1514                    23620
1515                    23640
1516                    23660          ;CONVERT A SINGLE PRECISION NUMBER TO A DOUBLE PRECISION ONE
1517                    23680          ;ALTERS A,H,L
1518                    23700  CONDS:  LXI     H,$CODE         ;ZERO H,L
1519                    23720          SHLD    DFACLO          ;CLEAR THE FOUR LOWER BYTES IN THE DOUBLE
1520                    23740          SHLD    DFACLO+2        ; PRECISION NUMBER
1521                    23760          MVI     A,10            ;SET VALTYP TO "DOUBLE PRECISION"
1522                    23780          JMP     CONISD>         ;GO TO IT
1523                    23800  PAGE
```

```
 1524                                   23820  SUBTTL  GREATEST INTEGER FUNCTION
 1525                                   23840          ;QUICK GREATEST INTEGER FUNCTION
 1526                                   23860          ;LEAVES INT(FAC) IN C,D,E (SIGNED)
 1527                                   23880          ;ASSUMES FAC .LT. 2**23 = 8388608
 1528                                   23900          ;ASSUMES THE EXPONENT OF FAC IS IN A
 1529                                   23920          ;ALTERS A,B,C,D,E
 1530  001372' 001000  000107          23940  QINT:   MOV     B,A             ;ZERO B,C,D,E IN CASE THE NUMBER IS ZERO
 1531  001373' 001000  000117          23960          MOV     C,A
 1532  001374' 001000  000127          23980          MOV     D,A
 1533  001375' 001000  000137          24000          MOV     E,A
 1534  001376' 001000  000267          24020          ORA     A               ;SET CONDITION CODES
 1535  001377' 001000  000310          24040          RZ                      ;IT IS ZERO, WE ARE DONE
 1536
 1537                                   24080  ;THE HARD CASE IN QINT IS NEGATIVE NON-INTEGERS.  TO HANDLE THIS, IF THE
 1538                                   24100  ;NUMBER IS NEGATIVE, WE REGARD THE 3-BYTE MANTISSA AS A 3-BYTE INTEGER AND
 1539                                   24120  ;SUBTARCT ONE,  THEN ALL THE FRACTIONAL BITS ARE SHIFTED OUT BY SHIFTING THE
 1540                                   24140  ;MANTISSA RIGHT.  THEN, IF THE NUMBER WAS NEGATIVE, WE ADD ONE.  SO, IF WE
 1541                                   24160  ;HAD A NEGATIVE INTEGER, ALL THE BITS TO THE RIGHT OF THE BINARY POINT WERE
 1542                                   24180  ;ZERO.  SO THE NET EFFECT IS WE HAVE THE ORIGINAL NUMBER IN C,D,E.  IF THE
 1543                                   24200  ;NUMBER WAS A NEGATIVE NON-INTEGER, THERE IS AT LEAST ONE NON-ZERO BIT TO THE
 1544                                   24220  ;RIGHT OF THE BINARY POINT.  SO THE NET EFFECT IS THAT WE GET THE ABSOLUTE
 1545                                   24240  ;VALUE OF INT(FAC) IN C,D,E.  C,D,E IS THEN NEGATED IF THE ORIGINAL NUMBER WAS
 1546                                   24260  ;NEGATIVE SO THE RESULT WILL BE SIGNED.
 1547  001400' 001000  000345          24280          PUSH    H               ;SAVE (HL)
 1548  001401' 001000  000315          24300          CALL    MOVRF           ;GET NUMBER IN THE REGISTERS
 1549  001402' 000000  001240'
 1550  001403' 000000  001342'
 1551  001404' 001000  000315          24320          CALL    UNPACK          ;UNPACK THE NUMBER
 1552  001405' 000000  001272'
 1553  001406' 000000  00140E'
 1554  001407' 001000  000256          24340          XRA     M               ;GET SIGN OF NUMBER
 1555  001410' 001000  000147          24360          MOV     M,A             ;DON'T LOSE IT
 1556  001411' 001200  000374          24380          CM      QINTA           ;SUBTRACT 1 FROM LO IF NUMBER IS NEGATIVE
 1557  001412' 000000  001436'
 1558  001413' 000000  001405'
 1559  001414' 001000  000076          24400          MVI     A,230           ;SEE HOW MANY WE HAVE TO SHIFT TO CHANGE
 1560  001415' 000000  000230
 1561  001416' 001000  000220          24420          SUB     B               ; NUMBER TO AN INTEGER
 1562  001417' 001000  000315          24440          CALL    SHIFTR          ;SHIFT NUMBER TO GET RID OF FRACTIONAL BITS
 1563  001420' 000000  000334'
 1564  001421' 000000  001412'
 1565  001422' 001000  000174          24460          MOV     A,H             ;GET SIGN
 1566  001423' 001000  000027          24480          RAL                     ;PUT SIGN IN CARRY SO IT WILL NOT BE CHANGED
 1567  001424' 001000  000334          24500          CC      ROUNDA          ;IF NUMBER WAS NEGATIVE, ADD ONE
 1568  001425' 000000  000255'
 1569  001426' 000000  001420'
 1570  001427' 001000  000006          24520          MVI     B,0             ;FORGET THE BITS WE SHIFTED OUT
 1571  001430' 000000  000000
 1572  001431' 001000  000334          24540          CC      NEGR            ;NEGATE NUMBER IF IT WAS NEGATIVE BECAUSE WE
 1573  001432' 000000  000310'
 1574  001433' 000000  001425'
 1575                                   24560                                  ; WANT A SIGNED MANTISSA
 1576  001434' 001000  000341          24580          POP     H               ;GET OLD (HL) BACK
```



```
 1577  001435' 001000  000311          24600          RET                     ;ALL DONE
 1578
 1579  001436' 001000  000033          24640  QINTA:  DCX     D               ;SUBTRACT ONE FROM C,D,E
 1580  001437' 001000  000172          24660          MOV     A,D             ;WE HAVE TO SUBTRACT ONE FROM C IF
 1581  001440' 001000  000243          24680          ANA     E               ; D AND E ARE BOTH ALL ONES
 1582  001441' 001000  000074          24700          INR     A               ;SEE IF BOTH WERE -1
 1583  001442' 001000  000300          24720          RNZ                     ;THEY WERE NOT, WE ARE DONE
 1584                                   24740  IFN     LENGTH=2,<
 1585  001443' 001000  000015          24760          DCR     C>              ;THEY WERE, SUBTRACT ONE FROM C
 1586                                   24780  IFE     LENGTH=2,<
 1587                                   24800  DCX8RT: DCX     B>              ;THIS IS FOR BILL,  C WILL NEVER BE ZERO
 1588                                   24820                                  ; (THE MSB WILL ALWAYS BE ONE) SO "DCX  B"
 1589                                   24840                                  ; AND "DCR      C" ARE FUNCTIONALLY EQUIVALENT
 1590  001444' 001000  000311          24860          RET                     ;ALL DONE
 1591
 1592
 1593                                   24920          ;GREATEST INTEGER FUNCTION
 1594                                   24940          ;ALTERS A,B,C,D,E,H,L
 1595                                   24960  IFE     LENGTH=2,<
 1596                                   24980  INTFNC: CPI     4               ;SEE WHAT KIND OF NUMBER WE HAVE
 1597                                   25000          RC                      ;IT IS AN INTEGER, ALL DONE
 1598                                   25020          JNZ     DINT            ;CONVERT THE DOUBLE PRECISION NUMBER
 1599                                   25040          CALL    CONIS>          ;TRY TO CONVERT THE NUMBER TO AN INTEGER
 1600                                   25060                                  ;IF WE CAN'T, WE WILL RETURN HERE TO GIVE A
 1601                                   25080                                  ; SINGLE PRECISION RESULT
 1602  001445' 001000  000041          25100  INT:    LXI     H,FAC           ;GET EXPONENT
 1603  001446' 000000  001156*
 1604  001447' 000000  001432'
 1605  001450' 001000  000176          25120          MOV     A,M
 1606  001451' 001000  000376          25140          CPI     230             ;SEE IF NUMBER HAS ANY FRACTIONAL BITS
 1607  001452' 000000  000230
 1608                                   25160  IFN     EXTFNC,<                ;THE ONLY GUY WHO NEEDS THIS DOESN'T CARE
 1609  001453' 001000  000072          25180          LDA     FACLO>          ; ABOUT THE SIGN
 1610  001454' 000000  001255*
 1611  001455' 000000  001446'
 1612  001456' 001000  000320          25200          RNC                     ;IT DOES NOT
 1613                                   25220  IFN     EXTFNC,<
 1614  001457' 001000  000176          25240          MOV     A,M>            ;GET EXPONENT BACK
 1615  001460' 001000  000315          25260          CALL    QINT            ;IT DOES, SHIFT THEM OUT
 1616  001461' 000000  001372'
 1617  001462' 000000  001454'
 1618  001463' 001000  000066          25280          MVI     M,230           ;CHANGE EXPONENT SO IT WILL BE CORRECT
 1619  001464' 000000  000230
 1620                                   25300                                  ; AFTER NORMALIZATION
 1621                                   25320  IFN     EXTFNC,<
 1622  001465' 001000  000173          25340          MOV     A,E             ;GET LO
 1623  001466' 001000  000365          25360          PUSH    PSW>            ;SAVE IT
 1624  001467' 001000  000171          25380          MOV     A,C             ;NEGATE NUMBER IF IT IS NEGATIVE
 1625  001470' 001000  000027          25400          RAL                     ;PUT SIGN IN CARRY
 1626                                   25420  IFE     EXTFNC,<
 1627                                   25440          JMP     FADFLT>         ;REFLOAT NUMBER
 1628                                   25460  IFN     EXTFNC,<
 1629  001471' 001000  000315          25480          CALL    FADFLT          ;REFLOAT NUMBER
```

```
1630  001472' 000000  000143'
1631  001473' 000000  001461'
1632  001474' 001000  000361          25500  POPPRT: POP     PSW             ;GET LO BACK
1633  001475' 001000  000311          25520          RET>                    ;ALL DONE
1634
1635
1636                                   25580  IFE     LENGTH=2,<
1637                                   25600          ;GREATEST INTEGER FUNCTION FOR DOUBLE PRECISION NUMBERS
1638                                   25620          ;ALTERS A,B,C,D,E,H,L
1639                                   25640  DINT:   LXI     H,FAC           ;GET POINTER TO FAC
1640                                   25660          MOV     A,M             ;GET EXPONENT
1641                                   25680          CPI     220             ;CAN WE CONVERT IT TO AN INTEGER?
1642                                   25700          JC      FRCINT          ;THEN DO SO
1643                                   25720          JNZ     DINT2           ;CHECK FOR =32768
1644                                   25740          MOV     C,A             ;SAVE EXPONENT IN C
1645                                   25760          DCX     H               ;GET POINTER TO SIGN AND MO
1646                                   25780          MOV     A,M             ;GET SIGN AND MO
1647                                   25800          XRI     200             ;CHECK IF IT IS 200
1648                                   25820          MVI     B,6             ;SET UP A COUNT TO CHECK IF THE REST OF
1649                                   25840  DINT1:  DCX     H               ; THE NUMBER IS ZERO, POINT TO NEXT BYTE
1650                                   25860          ORA     M               ;IF ANY BITS ARE NON=ZERO, A WILL BE NON=ZERO
1651                                   25880          DCR     B               ;ARE WE DONE?
1652                                   25900          JNZ     DINT1           ;NO, CHECK THE NEXT LOWER ORDER BYTE
1653                                   25920          ORA     A               ;IS A NOW ZERO?
1654                                   25940          LXI     H,200*400+$CODE ;GET =32768 JUST IN CASE
1655                                   25960          JZ      CONISS          ;A IS ZERO SO WE HAVE =32768
1656                                   25980          MOV     A,C             ;GET EXPONENT
1657                                   26000  DINT2:  CPI     270             ;ARE THERE ANY FRACTIONAL BITS?
1658                                   26020          RNC                     ;NO, THE NUMBER IS ALREADY AN INTEGER
1659                                   26040  DINTFO: PUSH    PSW             ;ENTRY FROM FOUT, CARRY IS ZERO IF WE COME
1660                                   26060                                  ; HERE FROM FOUT
1661                                   26080          CALL    MOVRF           ;GET HO'S OF NUMBER IN REGISTERS FOR UNPACKING
1662                                   26100          CALL    UNPACK          ;UNPACK IT
1663                                   26120          XRA     M               ;GET ITS SIGN BACK
1664                                   26140          DCX     H               ;SET THE EXPONENT TO NORMALIZE CORRECTLY
1665                                   26160          MVI     M,270
1666                                   26180          PUSH    PSW             ;SAVE THE SIGN
1667                                   26200          CM      DINTA           ;SUBTRACT 1 FROM LO IF NUMBER IS NEGATIVE
1668                                   26220          MVI     A,270           ;GET HOW MANY BITS WE HAVE TO SHIFT OUT
1669                                   26240          SUB     B
1670                                   26260          CALL    DSHFTR          ;SHIFT THEM OUT!!
1671                                   26280          POP     PSW             ;GET THE SIGN BACK
1672                                   26300          CM      DROUNA          ;IF NUMBER WAS NEGATIVE, ADD ONE
1673                                   26320          XRA     A               ;PUT A ZERO IN THE EXTRA LO BYTE SO WHEN
1674                                   26340          STA     DFACLO-1        ; WE NORMALIZE, WE WILL SHIFT IN ZEROS
1675                                   26360          POP     PSW             ;IF WE WERE CALLED FROM FOUT, DON'T NORMALIZE,
1676                                   26380          RNC                     ; JUST RETURN
1677                                   26400          JMP     DNORML          ;RE=FLOAT THE INTEGER
1678                                   26420
1679                                   26440  DINTA:  LXI     H,DFACLO        ;SUBTRACT ONE FROM FAC, GET POINTER TO LO
1680                                   26460  DINTA1: MOV     A,M             ;GET A BYTE OF FAC
1681                                   26480          DCR     M               ;SUBTRACT ONE FROM IT
1682                                   26500          ORA     A               ;CONTINUE ONLY IF THE BYTE USED TO BE ZERO
```



```
1683                                   26520          INX     H               ;INCREMENT POINTER TO NEXT BYTE
1684                                   26540          JZ      DINTA1          ;CONTINUE IF NECESSARY
1685                                   26560          RET>                    ;ALL DONE
1686                                   26580  PAGE
```

```
                                        26600   SUBTTL  INTEGER ARITHMETIC ROUTINES
                                        26620   IFN     MULDIM8<LENGTH-2>,<
                                        26640           ;TWO BYTE UNSIGNED INTEGER MULTIPLY
                                        26660           ; (HL):=(BC)*(DE)
                                        26680           ;A,D,E,H,L ARE CHANGED
001476' 001000  000041                  26700   DMULT:  LXI     H,$CODE         ;ZERO PRODUCT REGISTERS
001477' 000000  001153*
001500' 000000  001472'
001501' 001000  000170                  26720           MOV     A,B             ;CHECK IF (BC) IS ZERO
001502' 001000  000261                  26740           ORA     C               ;IF SO, JUST RETURN, (HL) IS ALREADY ZERO
001503' 001000  000310                  26760           RZ                      ;THIS IS DONE FOR SPEED
001504' 001000  000076                  26780           MVI     A,20            ;SET UP A COUNT
001505' 000000  000020
001506' 001000  000051                  26800   DMULT1: DAD     H               ;ROTATE (HL) LEFT ONE
001507' 001000  000332                  26820           JC      BSERR##         ;CHECK FOR OVERFLOW, IF SO,
001510' 000000  000000*
001511' 000000  001477'
001512' 001000  000353                  26840           XCHG                    ; BAD SUBSCRIPT (BS) ERROR
001513' 001000  000051                  26860           DAD     H               ;ROTATE (DE) LEFT ONE
001514' 001000  000353                  26880           XCHG
001515' 001000  000322                  26900           JNC     DMULT2          ;ADD IN (BC) IF NO WAS 1
001516' 000000  001524'
001517' 000000  001516'
001520' 001000  000011                  26920           DAD     B
001521' 001000  000332                  26940           JC      BSERR           ;CHECK FOR OVERFLOW
001522' 000000  001510*
001523' 000000  001516'
001524' 001000  000075                  26960   DMULT2: DCR     A               ;SEE IF DONE
001525' 001000  000302                  26980           JNZ     DMULT1
001526' 000000  001506'
001527' 000000  001522'
001530' 001000  000311                  27000           RET>                    ;ALL DONE

                                        27020
                                        27040   IFE     LENGTH-2,<
                                        27060   COMMENT %
                                        27080           INTEGER ARITHMETIC CONVENTIONS
                                        27100
                                        27120   INTEGER VARIABLES ARE 2 BYTE, SIGNED NUMBERS
                                        27140           THE LO BYTE COMES FIRST IN MEMORY
                                        27160
                                        27180   CALLING CONVENTIONS:
                                        27200   FOR ONE ARGUMENT FUNCTIONS:
                                        27220           THE ARGUMENT IS IN (HL), THE RESULT IS LEFT IN (HL)
                                        27240   FOR TWO ARGUMENT OPERATIONS:
                                        27260           THE FIRST ARGUMENT IS IN (DE)
                                        27280           THE SECOND ARGUMENT IS IN (HL)
                                        27300           THE RESULT IS LEFT IN (HL)
                                        27320   IF OVERFLOW OCCURS, THE ARGUMENTS ARE CONVERTED TO SINGLE PRECISION
                                        27340   WHEN INTEGERS ARE STORED IN THE FAC, THEY ARE STORED AT FACLO+0,1
                                        27360   VALTYP(INTEGER)=2
                                        27380   %
                                        27400
                                        27420
```



```
                                        27440
                                        27460           ;INTEGER SUBTRTACTION   (HL):=(DE)-(HL)
                                        27480           ;ALTERS A,B,C,D,E,H,L
                                        27500   ISUB:   MOV     A,H             ;EXTEND THE SIGN OF (HL) TO B
                                        27520           RAL                     ;GET SIGN IN CARRY
                                        27540           SBB     A
                                        27560           MOV     B,A
                                        27580           CALL    INEGHL          ;NEGATE (HL)
                                        27600           MOV     A,C             ;GET A ZERO
                                        27620           SBB     B               ;NEGATE SIGN
                                        27640           JMP     IADDS           ;GO ADD THE NUMBERS
                                        27660
                                        27680
                                        27700           ;INTEGER ADDITION       (HL):=(DE)+(HL)
                                        27720           ;ALTERS A,B,C,D,E,H,L
                                        27740   IADD:   MOV     A,H             ;EXTEND THE SIGN OF (HL) TO B
                                        27760           RAL                     ;GET SIGN IN CARRY
                                        27780           SBB     A
                                        27800   IADDS:  MOV     B,A             ;SAVE THE SIGN
                                        27820           PUSH    H               ;SAVE THE SECOND ARGUMENT IN CASE OF OVERFLOW
                                        27840           MOV     A,D             ;EXTEND THE SIGN OF (DE) TO A
                                        27860           RAL                     ;GET SIGN IN CARRY
                                        27880           SBB     A
                                        27900           DAD     D               ;ADD THE TWO LO'S
                                        27920           ADC     B               ;ADD THE EXTRA HO
                                        27940           RRC                     ;IF THE LSB OF A IS DIFFERENT FROM THE MSB OF
                                        27960           XRA     H               ; H, THEN OVERFLOW OCCURED
                                        27980           JP      POPPRT          ;NO OVERFLOW, GET OLD (HL) OFF STACK AND WE
                                        28000                                   ; ARE DONE
                                        28020           PUSH    B               ;OVERFLOW -- SAVE EXTENDED SIGN OF (HL)
                                        28040           XCHG                    ;GET (DE) IN (HL)
                                        28060           CALL    CONSIH          ;FLOAT IT
                                        28080           POP     PSW             ;GET SIGN OF (HL) IN A
                                        28100           POP     H               ;GET OLD (HL) BACK
                                        28120           CALL    PUSHF           ;PUT FIRST ARGUMENT ON STACK
                                        28140           XCHG                    ;PUT SECOND ARGUMENT IN (DE) FOR FLOATR
                                        28160           CALL    INEGAD          ;FLOAT IT
                                        28180           POPR                    ;GET FIRST ARGUMENT OFF STACK
                                        28200           JMP     FADD            ;ADD THE TWO NUMBERS USING SINGLE PRECISION
                                        28220
                                        28240
                                        28260           ;INTEGER MULTIPLICATION         (HL):=(DE)*(HL)
                                        28280           ;ALTERS A,B,C,D,E,H,L
                                        28300   IMULT:   PUSH    H               ;SAVE SECOND ARGUMENT IN CASE OF OVERFLOW
                                        28320           PUSH    D               ;SAVE FIRST ARGUMENT
                                        28340           CALL    IMULOV          ;FIX UP THE SIGNS
                                        28360           PUSH    B               ;SAVE THE SIGN OF THE RESULT
                                        28380           MOV     B,H             ;COPY SECOND ARGUMENT INTO (BC)
                                        28400           MOV     C,L
                                        28420           LXI     H,$CODE         ;ZERO (HL), THAT IS WHERE THE PRODUCT GOES
                                        28440           MVI     A,20            ;SET UP A COUNT
                                        28460   IMULT1: DAD     H               ;ROTATE PRODUCT LEFT ONE
                                        28480           JC      IMULTS          ;CHECK FOR OVERLFOW
```

```
 1793                    28500          XCHG                    ;ROTATE FIRST ARGUMENT LEFT ONE TO SEE IF
 1794                    28520          DAD     H               ; WE ADD IN (BC) OR NOT
 1795                    28540          XCHG
 1796                    28560          JNC     IMULT2          ;DON'T ADD IN ANYTHING
 1797                    28580          DAD     B               ;ADD IN (BC)
 1798                    28600          JC      IMULT5          ;CHECK FOR OVERLFOW
 1799                    28620  IMULT2: DCR     A               ;ARE WE DONE?
 1800                    28640          JNZ     IMULT1          ;NO, DO IT AGAIN
 1801                    28660          POP     B               ;WE ARE DONE, GET SIGN OF RESULT
 1802                    28680          POP     D               ;GET ORIGINAL FIRST ARGUMENT
 1803                    28700  IMLDIV: MOV     A,H             ;ENTRY FROM IDIV, IS RESULT ,GE, 32768?
 1804                    28720          ORA     A
 1805                    28740          JM      IMULT3          ;IT IS, CHECK FOR SPECIAL CASE OF -32768
 1806                    28760          POP     D               ;RESULT IS OK, GET SECOND ARGUMENT OFF STACK
 1807                    28780          MOV     A,B             ;GET THE SIGN OF RESULT IN A
 1808                    28800          JMP     INEGA           ;NEGATE THE RESULT IF NECESSARY
 1809                    28820  IMULT3: XRI     200             ;IS RESULT 327687
 1810                    28840          ORA     L               ;NOTE: IF WE GET HERE FROM IDIV, THE RESULT
 1811                    28860          JZ      IMULT4          ; MUST BE 32768, IT CANNOT BE GREATER
 1812                    28880          XCHG                    ;IT IS ,GT, 32768, WE HAVE OVERFLOW
 1813                    28900          XWD     1000,001        ;"LXI   B" OVER NEXT 2 BYTES
 1814                    28920  IMULT5: POP     B               ;GET SIGN OF RESULT OFF STACK
 1815                    28940          POP     H               ;GET THE ORIGINAL FIRST ARGUMENT
 1816                    28960          CALL    CONSIH          ;FLOAT IT
 1817                    28980          POP     H               ;GET THE ORIGINAL SECOND ARGUMENT
 1818                    29000          CALL    PUSHF           ;SAVE FLOATED FIRST ARUMENT
 1819                    29020          CALL    CONSIH          ;FLOAT SECOND ARGUMENT
 1820                    29040  FMULTT: POPR                    ;GET FIRST ARGUMENT OFF STACK, ENTRY FROM POLYX
 1821
 1822                    29060          JMP     FMULT           ;MULTIPLY THE ARGUMENTS USING SINGLE PRECISION
 1823                    29080  IMULT4: MOV     A,B             ;IS RESULT +32768 OR -32768?
 1824                    29100          ORA     A               ;SET ITS SIGN
 1825                    29120          POP     B               ;DISCARD ORIGINAL SECOND ARGUMENT
 1826                    29140          RM                      ;THE RESULT SHOULD BE NEGATIVE, IT IS OK
 1827                    29160          PUSH    D               ;IT IS POSITIVE, SAVE REMAINDER FOR MOD
 1828                    29180          CALL    CONSIH          ;FLOAT -32768
 1829                    29200          POP     D               ;GET MOD'S REMAINDER BACK
 1830                    29220          JMP     NEG             ;NEGATE -32768 TO GET 32768, WE ARE DONE
 1831                    29240
 1832                    29260
 1833                    29280          ;INTEGER DIVISION       (HL)I=(DE)/(HL)
 1834                    29300          ;REMAINDER IS IN (DE), QUOTIENT IN (HL)
 1835                    29320          ;ALTERS A,B,C,D,E,H,L
 1836                    29340  IDIV:   MOV     A,H             ;CHECK FOR DIVISION BY ZERO
 1837                    29360          ORA     L
 1838                    29380          JZ      DV0ERR          ;WE HAVE DIVISION BY ZERO!!
 1839                    29400          CALL    IMULDV          ;FIX UP THE SIGNS
 1840                    29420          PUSH    B               ;SAVE THE SIGN OF THE RESULT
 1841                    29440          XCHG                    ;GET DENOMINATOR IN (HL)
 1842                    29460          CALL    INEGHL          ;NEGATE IT
 1843                    29480          MOV     B,H             ;SAVE NEGATED DENOMINATOR IN (BC)
 1844                    29500          MOV     C,L
 1845                    29520          LXI     H,$CODE         ;ZERO WHERE WE DO THE SUBTRACTION
```



```
 1846                    29540          MVI     A,21            ;SET UP A COUNT
 1847                    29560          PUSH    PSW             ;SAVE IT
 1848                    29580          ORA     A               ;CLEAR CARRY
 1849                    29600          JMP     IDIV3           ;GO DIVIDE
 1850                    29620  IDIV1:  PUSH    PSW             ;SAVE COUNT
 1851                    29640          PUSH    H               ;SAVE (HL) I,E, CURRENT NUMERATOR
 1852                    29660          DAD     B               ;SUBTRACT DENOMINATOR
 1853                    29680          JNC     IDIV2           ;WE SUBTRACTED TOO MUCH, GET OLD (HL) BACK
 1854                    29700          POP     PSW             ;THE SUBTRACTION WAS GOOD, DISCARD OLD (HL)
 1855                    29720          STC                     ;NEXT BIT IN QUOTIENT IS A ONE
 1856                    29740          XWD     1000,076        ;"MVI   A" OVER NEXT BYTE
 1857                    29760  IDIV2:  POP     H               ;IGNORE THE SUBTRACTION, WE COULDN'T DO IT
 1858                    29780  IDIV3:  MOV     A,E             ;SHIFT IN THE NEXT QUOTIENT BIT
 1859                    29800          RAL
 1860                    29820          MOV     E,A
 1861                    29840          MOV     A,D             ;SHIFT THE MQ
 1862                    29860          RAL
 1863                    29880          MOV     D,A
 1864                    29900          MOV     A,L             ;SHIFT IN THE NEXT BIT OF THE NUMERATOR
 1865                    29920          RAL
 1866                    29940          MOV     L,A
 1867                    29960          MOV     A,H             ;DO THE MQ
 1868                    29980          RAL
 1869                    30000          MOV     H,A             ;SAVE THE MQ
 1870                    30020          POP     PSW             ;GET COUNT BACK
 1871                    30040          DCR     A               ;ARE WE DONE?
 1872                    30060          JNZ     IDIV1           ;NO, DIVIDE AGAIN
 1873                    30080          XCHG                    ;GET QUOTIENT IN (HL), REMAINDER IN (DE)
 1874                    30100          POP     B               ;GET SIGN OF RESULT
 1875                    30120          PUSH    D               ;SAVE REMAINDER SO STACK WILL BE ALRIGHT
 1876                    30140          JMP     IMLDIV          ;CHECK FOR SPECIAL CASE OF 32768
 1877                    30160
 1878                    30180
 1879                    30200          ;GET READY TO MULTIPLY OR DIVIDE
 1880                    30220          ;ALTERS A,B,C,D,E,H,L
 1881                    30240  IMULDV: MOV     A,H             ;GET SIGN OF RESULT
 1882                    30260          XRA     D
 1883                    30280          MOV     B,A             ;SAVE IT IN B
 1884                    30300          CALL    INEGH           ;NEGATE SECOND ARGUMENT IF NECESARY
 1885                    30320          XCHG                    ;PUT (DE) IN (HL), FALL IN AND NEGATE FIRST
 1886                    30340                                  ; ARGUMENT IF NECESSARY
 1887                    30360
 1888                    30380
 1889                    30400          ;NEGATE H,L
 1890                    30420          ;ALTERS A,C,H,L
 1891                    30440  INEGH:  MOV     A,H             ;GET SIGN OF (HL)
 1892                    30460  INEGA:  ORA     A               ;SET CONDITION CODES
 1893                    30480          RP                      ;WE DON'T HAVE TO NEGATE, IT IS POSITIVE
 1894                    30500  INEGHL: XRA     A               ;CLEAR A
 1895                    30520          MOV     C,A             ;STORE A ZERO (WE USE THIS METHOD FOR ISUB)
 1896                    30540          SUB     L               ;NEGATE LO
 1897                    30560          MOV     L,A             ;SAVE IT
 1898                    30580          MOV     A,C             ;GET A ZERO BACK
```

```
 1899                           30600          SBB     H               ;NEGATE HO
 1900                           30620          MOV     H,A             ;SAVE IT
 1901                           30640          RET                     ;ALL DONE
 1902                           30660
 1903                           30680
 1904                           30700          ;INTEGER ABSOLUTE VALUE
 1905                           30720          ;ALTERS A,B,C,D,E,H,L
 1906                           30740  IABS:   LDA     FACLO+1         ;GET SIGN OF INTEGER IN FAC
 1907                           30760          ORA     A               ;CHECK ITS SIGN
 1908                           30780          RP                      ;IT IS POSITIVE, LEAVE IT ALONE
 1909                           30800                                  ;FALL INTO INEG AND NEGATE IT
 1910                           30820
 1911                           30840
 1912                           30860          ;INTEGER NEGATION
 1913                           30880          ;ALTERS A,B,C,D,E,H,L
 1914                           30900  INEG:   LHLD    FACLO           ;GET THE INTEGER
 1915                           30920          CALL    INEGHL          ;NEGATE IT
 1916                           30940          SHLD    FACLO           ;STORE IT BACK IN THE FAC
 1917                           30960          XRI     200             ;CHECK FOR SPECIAL CASE OF 32768
 1918                           30980          ORA     L
 1919                           31000          RNZ                     ;IT DID NOT OCCUR, EVERYTHING IS FINE
 1920                           31020          XCHG                    ;WE HAVE IT, FLOAT 32768
 1921                           31040          MVI     A,4             ;CHANGE VALTYP TO "SINGLE PRECISION"
 1922                           31060          STA     VALTYP
 1923                           31080  INEGAD: MVI     B,230           ;ENTRY FROM IADD, SET EXPONENT
 1924                           31100          JMP     FLOATR          ;GO FLOAT THE NUMBER
 1925                           31120
 1926                           31140
 1927                           31160          ;MOD OPERATOR
 1928                           31180          ;(ML)I=(DE)-(DE)/(ML)*(ML),  (DE)=QUOTIENT
 1929                           31200          ;ALTERS A,B,C,D,E,H,L
 1930                           31220  MOD:    PUSH    D               ;SAVE (DE) FOR ITS SIGN
 1931                           31240          CALL    IDIV            ;DIVIDE AND GET THE REMAINDER
 1932                           31260          XCHG                    ;PUT REMAINDER IN (DE)
 1933                           31280          MVI     A,2             ;SET VALTYP TO "INTEGER" IN CASE RESULT OF
 1934                           31300          STA     VALTYP          ; THE DIVISION WAS 32768
 1935                           31320          POP     PSW             ;GET THE SIGN OF (DE) BACK
 1936                           31340          JMP     INEGA>          ;NEGATE THE REMAINDER IF NECESSARY
 1937                           31360  PAGE
```



```
 1938                           31380  SUBTTL  DOUBLE PRECISION ARITHMETIC ROUTINES
 1939                           31400  IFE     LENGTH-2,<
 1940                           31420  COMMENT %
 1941                           31440          DOUBLE PRECISION ARITHMETIC CONVENTIONS
 1942                           31460
 1943                           31480  DOUBLE PRECISION NUMBERS ARE 8 BYTE QUANTITIES
 1944                           31500  THE LAST 4 BYTES IN MEMORY ARE IN THE SAME FORMAT AS SINGLE PRECISION NUMBERS
 1945                           31520  THE FIRST 4 BYTES ARE 32 MORE LOW ORDER BITS OF PRECISION
 1946                           31540  THE LOWEST ORDER BYTE COMES FIRST IN MEMORY
 1947                           31560
 1948                           31580  CALLING CONVENTIONS:
 1949                           31600  FOR ONE ARGUMENT FUNCTIONS:
 1950                           31620          THE ARGUMENT IS IN THE FAC, THE RESULT IS LEFT IN THE FAC
 1951                           31640  FOR TWO ARGUMENT OPERATIONS:
 1952                           31660          THE FIRST ARGUMENT IS IN ARG-7,6,5,4,3,2,1,0  (NOTE: ARGLO=ARG-7)
 1953                           31680          THE SECOND ARGUMENT IS IN THE FAC
 1954                           31700          THE RESULT IS LEFT IN THE FAC
 1955                           31720  VALTYP(DOUBLE PRECISION)=10 OCTAL
 1956                           31740  %
 1957                           31760
 1958                           31780
 1959                           31800          ;DOUBLE PRECISION SUBTRACTION   FAC:=ARG-FAC
 1960                           31820          ;ALTERS ALL REGISTERS
 1961                           31840  DSUB:   CALL    NEG             ;NEGATE THE SECOND ARGUMENT
 1962                           31860                                  ;FALL INTO DADD
 1963                           31880
 1964                           31900
 1965                           31920          ;DOUBLE PRECISION ADDITION      FAC:=ARG+FAC
 1966                           31940          ;ALTERS ALL REGISTERS
 1967                           31960  DADD:   LXI     H,ARG           ;GET  POINTER TO EXPONENT OF FIRST ARGUMENT
 1968                           31980          MOV     A,M             ;CHECK IF IT IS ZERO
 1969                           32000          ORA     A
 1970                           32020          RZ                      ;IT IS, RESULT IS ALREADY IN FAC
 1971                           32040          MOV     B,A             ;SAVE EXPONENT FOR UNPACKING
 1972                           32060          DCX     H               ;POINT TO MO AND SIGN
 1973                           32080          MOV     C,M             ;GET MO AND SIGN FOR UNPACKING
 1974                           32100          LXI     D,FAC           ;GET POINTER TO EXPONENT OF SECOND ARGUMENT
 1975                           32120          LDAX    D               ;GET EXPONENT
 1976                           32140          ORA     A               ;SEE IF IT IS ZERO
 1977                           32160          JZ      VMOVFA          ;IT IS, MOVE ARG TO FAC AND WE ARE DONE
 1978                           32180          SUB     B               ;SUBTRACT EXPONENTS TO GET SHIFT COUNT
 1979                           32200          JNC     DADD2           ;PUT THE SMALLER NUMBER IN FAC
 1980                           32220          CMA                     ;NEGATE SHIFT COUNT
 1981                           32240          INR     A
 1982                           32260          PUSH    PSW             ;SAVE SHIFT COUNT
 1983                           32280          PUSH    B               ;SAVE MO TO UNPACK LATER
 1984                           32300          MVI     C,10            ;SWITCH FAC AND ARG, SET UP A COUNT
 1985                           32320          INX     H               ;POINT TO ARG
 1986                           32340  DADD1:  LDAX    D               ;GET A BYTE OF THE FAC
 1987                           32360          MOV     B,M             ;GET A BYTE OF ARG
 1988                           32380          MOV     M,A             ;PUT THE FAC BYTE IN ARG
 1989                           32400          MOV     A,B             ;PUT THE ARG BYTE IN A
 1990                           32420          STAX    D               ;PUT THE ARG BYTE IN FAC
```

```
                                32440           DCX     D               ;POINT TO THE NEXT LO BYTE OF FAC
                                32460           DCX     H               ;POINT TO THE NEXT LO BYTE OF ARG
                                32480           DCR     C               ;ARE WE DONE?
                                32500           JNZ     DADD1           ;NO, DO THE NEXT LO BYTE
                                32520           POP     B               ;GET THE HO BACK
                                32540           POP     PSW             ;GET THE SHIFT COUNT BACK
                                32560   DADD2:  CPI     71              ;ARE WE WITHIN 56 BITS?
                                32580           RNC                     ;NO, ALL DONE
                                32600           PUSH    PSW             ;SAVE SHIFT COUNT
                                32620           CALL    UNPACK          ;UNPACK THE NUMBERS
                                32640           MOV     B,A             ;SAVE SUBTRACTION FLAG
                                32660           MOV     A,C             ;SAVE THE UNPACKED HO
                                32680           STA     ARG+1
                                32700           POP     PSW             ;GET SHIFT COUNT
                                32720           CALL    DSHFTR          ;SHIFT FAC RIGHT THE RIGHT NUMBER OF TIMES
                                32740           ORA     B               ;GET SUBTRACTION FLAG, HERE A=0
                                32760           JP      DADD3           ;SUBTRACT NUMBERS IF THEIR SIGNS ARE DIFFERENT
                                32780           CALL    DADDAA          ;SIGNS ARE THE SAME, ADD THE NUMBERS
                                32800           JNC     DROUND          ;ROUND THE RESULT IF NO CARRY
                                32820           INR     M               ;WE HAVE OVERFLOW, ADD ONE TO THE EXPONENT
                                32840           JZ      OVERR           ;CHECK FOR OVERFLOW
                                32860           MVI     D,1             ;SHIFT NUMBER RIGHT ONE, SHIFT IN CARRY
                                32880           CALL    DSHFRA
                                32900           JMP     DROUND          ;ROUND THE RESULT
                                32920   DADD3:  XHD     1000,076        ;"MVI   A", SUBTRACT THE NUMBERS
                                32940           SBB     M               ;GET THE SUBTRACT INSTRUCTION IN A
                                32960           CALL    DADDA           ;SUBTRACT THE NUMBERS
                                32980           INX     H               ;POINT TO THE UNPACKED SIGN
                                33000           MOV     A,M             ;COMPLEMENT IT, SINCE THE FAC WAS SMALLER
                                33020           CMA
                                33040           MOV     M,A
                                33060           CC      DNEGR           ;NEGATE THE RESULT IF IT WAS NEATIVE
                                33080                                   ;FALL INTO DNORML
                                33100
                                33120
                                33140           ;NORMALIZE FAC
                                33160           ;ALTERS A,B,C,D,H,L
                                33180   DNORML: XRA     A               ;CLEAR SHIFT COUNT
                                33200   DNORM1: MOV     B,A             ;SAVE SHIFT COUNT
                                33220           LDA     FAC+1           ;GET HO
                                33240           ORA     A               ;SEE IF WE CAN SHIFT 8 LEFT
                                33260           JNZ     DNORM5          ;WE CAN'T, SEE IF NUMBER IS NORMALIZED
                                33280           LXI     H,DFACLO+1      ;WE CAN, GET POINTER TO LO
                                33300           MVI     C,10            ;SET UP A COUNT
                                33320   DNORM2: MOV     D,M             ;GET A BYTE OF FAC
                                33340           MOV     M,A             ;PUT IN BYTE FROM LAST LOCATION, THE FIRST
                                33360                                   ; TIME THROUGH A IS ZERO
                                33380           MOV     A,D             ;PUT THE CURRENT BYTE IN A FOR NEXT TIME
                                33400           INX     H               ;INCREMENT POINTER TO NEXT HIGHER ORDER
                                33420           DCR     C               ;ARE WE DONE?
                                33440           JNZ     DNORM2          ;NO, DO THE NEXT BYTE
                                33460           MOV     A,B             ;SUBTRACT 8 FROM SHIFT COUNT
                                33480           SUI     10
```



```
                                33500           CPI     300             ;HAVE WE SHIFTED ALL BYTES TO ZERO?
                                33520           JNZ     DNORM1          ;NO, TRY TO SHIFT 8 MORE
                                33540           JMP     ZERO            ;YES, THE NUMBER IS ZERO
                                33560   DNORM3: DCR     B               ;DECREMENT SHIFT COUNT
                                33580           LXI     H,DFACLO+1      ;GET POINTER TO LO
                                33600           CALL    DSHFLC          ;SHIFT THE FAC LEFT
                                33620           ORA     A               ;SEE IF NUMBER IS NORMALIZED
                                33640   DNORM5: JP      DNORM3          ;SHIFT FAC LEFT ONE IF IT IS NOT NORMALIZED
                                33660           MOV     A,B             ;GET THE SHIFT COUNT
                                33680           ORA     A               ;SEE IF NO SHIFTING WAS DONE
                                33700           JZ      DROUND          ;NONE WAS, PROCEED TO ROUND THE NUMBER
                                33720           LXI     H,FAC           ;GET POINTER TO EXPONENT
                                33740           ADD     M               ;UPDATE IT
                                33760           MOV     M,A             ;SAVE UPDATED EXPONENT
                                33780           JNC     ZERO            ;UNDERFLOW, THE RESULT IS ZERO
                                33800           RZ                      ;RESULT IS ALREADY ZERO, WE ARE DONE
                                33820                                   ;FALL INTO DROUND AND ROUND THE RESULT
                                33840
                                33860
                                33880           ;ROUND FAC
                                33900           ;ALTERS A,B,H,L
                                33920   DROUND: LDA     DFACLO+1        ;GET EXTRA BYTE TO SEE IF WE HAVE TO ROUND
                                33940   DROUNB: ORA     A               ;ENTRY FROM DDIV
                                33960           CM      DROUNA          ;ROUND UP IF NECESSARY
                                33980           LXI     H,FAC+1         ;GET POINTER TO UNPACKED SIGN
                                34000           MOV     A,M             ;GET SIGN
                                34020           ANI     200             ;ISOLATE SIGN BIT
                                34040           DCX     H               ;POINT TO HO
                                34060           DCX     H
                                34080           XRA     M               ;PACK SIGN AND HO
                                34100           MOV     M,A             ;PUT PACKED SIGN AND HO IN FAC
                                34120           RET                     ;WE ARE DONE
                                34140
                                34160
                                34180           ;SUBROUTINE FOR ROUND: ADD ONE TO FAC
                                34200   DROUNA: LXI     H,DFACLO        ;GET POINTER TO LO, ENTRY FROM DINT
                                34220           MVI     B,7             ;SET UP A COUNT
                                34240   DRONA1: INR     M               ;INCREMENT A BYTE
                                34260           RNZ                     ;RETURN IF THERE WAS NO CARRY
                                34280           INX     H               ;INCREMENT POINTER TO NEXT HIGHER ORDER
                                34300           DCR     B               ;HAVE WE INCREMENTED ALL BYTES
                                34320           JNZ     DRONA1          ;NO, TRY THE NEXT ONE
                                34340           INR     M               ;YES, INCREMENT THE EXPONENT
                                34360           JZ      OVERR           ;CHECK FOR OVERFLOW
                                34380           DCX     H               ;THE NUMBER OVERFLOWED ITS EXPONENT
                                34400           MVI     M,200           ;PUT 200 IN HO
                                34420           RET                     ;ALL DONE
                                34440
                                34460
                                34480           ;ADD OR SUBTRACT 2 DBL QUANTITIES
                                34500           ;ALTERS A,C,D,E,H,L
                                34520   DADDO:  LXI     H,FBUFFR+^D17   ;ENTRY FROM DDIV
                                34540           LXI     D,ARGLO         ;ADD OR SUBTRACT FBUFFR+17 AND ARG
```

```
 2097                    34560                  JMP     DADDS           ;DO THE OPERATION
 2098                    34580
 2099                    34600  DADDAA: XWD     1000,076        ;"MVI   A", ENTRY FROM DADD, DMULT
 2100                    34620          ADC     M               ;SETUP ADD INSTRUCTION FOR LOOP
 2101                    34640  DADDA:  LXI     H,ARGLO         ;GET POINTER TO ARG, ENTRY FROM DADD
 2102                    34660  DADDFO: LXI     D,DFACLO        ;GET POINTER TO FAC, ENTRY FROM FOUT
 2103                    34680  DADDS:  MVI     C,7             ;SET UP A COUNT
 2104                    34700          STA     DADDOP          ;STORE THE ADD OR SUBTRACT INSTRUCTION
 2105                    34720          XRA     A               ;CLEAR CARRY
 2106                    34740  DADDL:  LDAX    D               ;GET A BYTE FROM RESULT NUMBER
 2107                    34760  DADDOP: NOP                     ;THIS IS EITHER "ADC    M" OR "SBB      M"
 2108                    34780          STAX    D               ;SAVE THE CHANGED BYTE
 2109                    34800          INX     D               ;INCREMENT POINTERS TO NEXT HIGHER ORDER BYTE
 2110                    34820          INX     H
 2111                    34840          DCR     C               ;ARE WE DONE?
 2112                    34860          JNZ     DADDL           ;NO, DO THE NEXT HIGHER ORDER BYTE
 2113                    34880          RET                     ;ALL DONE
 2114                    34900
 2115                    34920
 2116                    34940          ;NEGATE SIGNED NUMBER IN FAC
 2117                    34960          ;THIS IS USED BY DADD, DINT
 2118                    34980          ;ALTERS A,B,C,H,L
 2119                    35000  ONEGR:  MOV     A,M             ;COMPLEMENT SIGN OF FAC
 2120                    35020          CMA                     ;USE THE UNPACKED SIGN BYTE
 2121                    35040          MOV     M,A             ;SAVE THE NEW SIGN
 2122                    35060          LXI     H,DFACLO+1      ;GET POINTER TO LO
 2123                    35080          MVI     B,10            ;SET UP A COUNT
 2124                    35100          XRA     A               ;CLEAR CARRY AND GET A ZERO
 2125                    35120          MOV     C,A             ;SAVE ZERO IN C
 2126                    35140  ONEGR1: MOV     A,C             ;GET A ZERO
 2127                    35160          SBB     M               ;NEGATE THE BYTE OF FAC
 2128                    35180          MOV     M,A             ;UPDATE FAC
 2129                    35200          INX     H               ;INCREMENT POINTER TO NEXT HIGHER ORDER BYTE
 2130                    35220          DCR     B               ;ARE WE DONE?
 2131                    35240          JNZ     ONEGR1          ;NO, NEGATE THE NEXT BYTE
 2132                    35260          RET                     ;ALL DONE
 2133                    35280
 2134                    35300
 2135                    35320          ;SHIFT DBL FAC RIGHT ONE
 2136                    35340          ;A = SHIFT COUNT
 2137                    35360          ;ALTERS A,C,D,E,H,L
 2138                    35380  DSHFTR: LXI     H,DFACLO+1      ;GET POINTER TO LO
 2139                    35400          MVI     M,0             ;PUT ZERO IN EXTRA LO ORDER BYTE
 2140                    35420  DSHFR1: SUI     10              ;SEE IF WE CAN SHIFT 8 RIGHT
 2141                    35440          JC      DSHFR3          ;WE CAN'T, CHECK IF WE ARE DONE
 2142                    35460  DSHFRM: LXI     H,FAC-1         ;ENTRY FROM DMULT, GET POINTER TO MO
 2143                    35480          MVI     E,0             ;SHIFT A ZERO INTO THE MO
 2144                    35500          MVI     D,10            ;SET UP A COUNT
 2145                    35520  DSHFR2: MOV     C,M             ;SAVE A BYTE OF FAC
 2146                    35540          MOV     M,E             ;PUT THE LAST BYTE IN ITS PLACE
 2147                    35560          MOV     E,C             ;SET UP E FOR NEXT TIME THROUGH THE LOOP
 2148                    35580          DCX     H               ;POINT TO NEXT LOWER ORDER BYTE
 2149                    35600          DCR     D               ;ARE WE DONE?
```



```
 2150                    35620          JNZ     DSHFR2          ;NO, DO THE NEXT BYTE
 2151                    35640          JMP     DSHFR1          ;YES, SEE IF WE CAN SHIFT OVER 8 MORE
 2152                    35660  DSHFR3: ADI     11              ;CORRECT SHIFT COUNT
 2153                    35680          MOV     D,A             ;SAVE SHIFT COUNT IN D
 2154                    35700  DSHFR4: XRA     A               ;CLEAR CARRY
 2155                    35720          DCR     D               ;ARE WE DONE?
 2156                    35740          RZ                      ;YES
 2157                    35760  DSHFRA: LXI     H,FAC-1         ;NO, GET POINTER TO LO, ENTRY FROM DADD, DMULT
 2158                    35780          MVI     E,10            ;SET UP A COUNT, ROTATE FAC ONE LEFT
 2159                    35800  DSHFR5: MOV     A,M             ;GET A BYTE OF THE FAC
 2160                    35820          RAR                     ;ROTATE IT LEFT
 2161                    35840          MOV     M,A             ;PUT THE UPDATED BYTE BACK
 2162                    35860          DCX     H               ;DECREMENT POINTER TO NEXT LOWER ORDER BYTE
 2163                    35880          DCR     E               ;ARE WE DONE?
 2164                    35900          JNZ     DSHFR5          ;NO, ROTATE THE NEXT LOWER ORDER BYTE
 2165                    35920          JMP     DSHFR4          ;YES, SEE IF WE ARE DONE SHIFTING
 2166                    35940
 2167                    35960
 2168                    35980          ;ROTATE FAC LEFT ONE
 2169                    36000          ;ALTERS A,C,H,L
 2170                    36020  DSHFLC: MVI     C,10            ;SET UP A COUNT
 2171                    36040  DSHFTL: MOV     A,M             ;GET A BYTE OF FAC
 2172                    36060          RAL                     ;ROTATE IT LEFT ONE
 2173                    36080          MOV     M,A             ;UPDATE BYTE IN FAC
 2174                    36100          INX     H               ;INCREMENT POINTER TO NEXT HIGHER ORDER BYTE
 2175                    36120          DCR     C               ;ARE WE DONE?
 2176                    36140          JNZ     DSHFTL          ;NO, ROTATE THE NEXT BYTE
 2177                    36160          RET                     ;ALL DONE
 2178                    36180
 2179                    36200
 2180                    36220          ;DOUBLE PRECISION MULTIPLICATION        FAC:=ARG*FAC
 2181                    36240          ;ALTERS ALL REGISTERS
 2182                    36260  DMULT:  FSIGN                   ;CHECK IF WE ARE MULTIPLYING BY ZERO
 2183                    36280          RZ                      ;YES, ALL DONE, THE FAC IS ZERO
 2184                    36300          CALL    MULDVA          ;ADD EXPONENTS AND TAKE CARE OF SIGNS
 2185                    36320          CALL    DMULDV          ;ZERO FAC AND PUT FAC IN FBUFFR
 2186                    36340          MOV     M,C             ;PUT UNPACKED MO IN ARG
 2187                    36360          LXI     D,ARGLO         ;GET POINTER TO LO OF ARG
 2188                    36380          MVI     B,7             ;SET UP A COUNT
 2189                    36400  DMULT2: LDAX    D               ;GET THE BYTE OF ARG TO MULTIPLY BY
 2190                    36420          INX     D               ;INCREMENT POINTER TO NEXT HIGHER BYTE
 2191                    36440          ORA     A               ;CHECK IF WE ARE MULTIPLYING BY ZERO
 2192                    36460          PUSH    D               ;SAVE POINTER TO ARG
 2193                    36480          JZ      DMULT3          ;WE ARE
 2194                    36500          MVI     C,10            ;SET UP A COUNT
 2195                    36520  DMULT3: PUSH    B               ;SAVE COUNTERS
 2196                    36540          RAR                     ;ROTATE MULTIPLIER RIGHT
 2197                    36560          MOV     B,A             ;SAVE IT
 2198                    36580          CC      DADDAA          ;ADD IN OLD FAC IF BIT OF MULTIPIER WAS ONE
 2199                    36600          MVI     D,1             ;ROTATE PRODUCT RIGHT ONE
 2200                    36620          CALL    DSHFRA
 2201                    36640          MOV     A,B             ;GET MULTIPLIER IN A
 2202                    36660          POP     B               ;GET COUNTERS BACK
```

```
2203  36680          DCR   C              ;ARE WE DONE WITH THIS BYTE OF ARG?
2204  36700          JNZ   DMULT3         ;NO, MULTIPLY BY THE NEXT BIT OF THE MULTIPLIER
2205
2206  36720  DMULT4: POP   D              ;YES, GET POINTER INTO ARG BACK
2207  36740          DCR   B              ;ARE WE DONE?
2208  36760          JNZ   DMULT2         ;NO, MULTIPLY BY NEXT HIGHER ORDER BY OF ARG
2209  36780          JMP   NORMAL         ;ALL DONE, NORMALIZE AND ROUND RESULT
2210  36800  DMULTS: CALL  DSHFRM         ;SHIFT PRODUCT RIGHT ONE BYTE, WE ARE
2211  36820          JMP   DMULT4         ; MULTIPLYIING BY ZERO
2212  36840
2213  36860
2214  36880          ;CONSTANT FOR DIV10, DDIV10
2215  36900  DTEN:   000     ; 1000
2216  36920          000
2217  36940          000
2218  36960          000
2219  36980  FTEN:   000     ; 10,0
2220  37000          000
2221  37020          040
2222  37040          204
2223  37060
2224  37080          ;DOUBLE PRECISION DIVIDE FAC BY 10
2225  37100          ;ALTERS ALL REGISTERS
2226  37120  DDIV10: CALL  VMOVAF         ;SAVE THE FAC IN ARG
2227  37140          LXI   M,DTEN         ;GET POINTER TO A DOUBLE PRECISION 10
2228  37160          CALL  VMOVFM         ;MOVE TEN INTO THE FAC
2229  37180                               ;FALL INTO DDIV AND DIVIDE BY TEN
2230  37200
2231  37220
2232  37240          ;DOUBLE PRECISION DIVISION      FAC:=ARG/FAC
2233  37260          ;ALTERS ALL REGISTERS
2234  37280  DDIV:   FSIGN                ;CHECK FOR DIVISION BY ZERO
2235  37300          JZ    DV0ERR         ;DON'T LET HIM DO IT
2236  37320          CALL  MULDVS         ;SUBTRACT EXPONENTS AND CHECK SIGNS
2237  37340          INR   M              ;ADD TWO TO EXPONENT TO CORRECT SCALING
2238  37360          INR   M
2239  37380          CALL  DMULDV         ;ZERO FAC AND PUT FAC IN FBUFFR
2240  37400          LXI   M,ARG          ;GET POINTER TO THE EXTRA HO BYTE WE WILL USE
2241  37420          MOV   M,C            ;ZERO IT
2242  37440          MVI   B,0            ;ZERO FLAG TO SEE WHEN WE START DIVIDING
2243  37460  DDIV1:  XWD   1000,076      ;"MVI   A", SUBTRACT FBUFFR FROM ARG
2244  37480          SBB   M              ;GET SUBTRACT INSTRUCTION
2245  37500          CALL  DADDO          ;DO THE SUBTRACTION
2246  37520          LDAX  D              ;SUBTRACT FROM EXTRA HO BYTE
2247  37540          SBB   C              ;HERE C=0
2248  37560          CMC                  ;CARRY=1 IF SUBTRACTION WAS GOOD
2249  37580          JC    DDIV2          ;WAS IT OK?
2250  37600          XWD   1000,076       ;"MVI   A"  NO, ADD FBUFFR BACK IN
2251  37620          ADC   M              ;GET ADD INSTRUCTION
2252  37640          CALL  DADDO          ;DO THE ADDITION
2253  37660          XRA   A              ;CLEAR CARRY
2254  37680          XWD   1000,332       ;"JC" OVER NEXT TWO BYTES
2255  37700  DDIV2:  STAX  D              ;STORE THE NEW HIGHEST ORDER BYTE
```



```
2256  37720          INR   B              ;INCREMENT FLAG TO SHOW WE COULD DIVIDE
2257  37740          LDA   FAC-1          ;CHECK IF WE ARE DONE DIVIDING
2258  37760          INR   A              ;SET SIGN FLAG WITHOUT AFFECTING CARRY
2259  37780          DCR   A
2260  38000          RAR                  ;PUT CARRY IN MSB FOR DROUND
2261  38020          JM    DROUNB         ;WE ARE DONE, WE HAVE 57 BITS OF ACCURACY
2262  38040          RAL                  ;GET OLD CARRY BACK WHERE IT BELONGS
2263  38060          LXI   H,QFACLO       ;GET POINTER TO LO OF FAC
2264  38080          MVI   C,7            ;SET UP A COUNT, SHIFT FAC LEFT ONE
2265  38100          CALL  DSHFTL         ;SHIFT IN THE NEXT BIT IN THE QUOTIENT
2266  38120          LXI   H,ARGLO        ;GET POINTER TO LO IN ARG
2267  38140          CALL  DSHFLC         ;SHIFT DIVIDEND ONE LEFT
2268  38160          MOV   A,B            ;IS THIS THE FIRST TIME AND WAS THE
2269  38180          ORA   A              ; SUBTRACTION NOT GOOD? (B WILL GET
2270  38200          JNZ   DDIV1          ; CHANGED ON THE FIRST OR SECOND SUBTRACTION)
2271  38220          LXI   H,FAC          ;YES, SUBTRACT ONE FROM EXPONENT TO CORRECT
2272  38240          DCR   M              ; SCALING
2273  38260          JNZ   DDIV1          ;CONTINUE DIVIDING IF NO OVERFLOW
2274  38280          JMP   OVERR          ;WE HAVE OVERFLOW!!
2275  38300
2276  38320
2277  38340          ;TRANSFER FAC TO FBUFFR FOR DMULT AND DDIV
2278  38360          ;ALTERS A,B,C,D,E,H,L
2279  38380  DMULDV: MOV   A,C            ;PUT UNPACKED HO BACK IN ARG
2280  38400          STA   ARG-1
2281  38420          DCX   H              ;POINT TO HO OF FAC
2282  38440          LXI   D,FBUFFR+^O23  ;POINT TO END OF FBUFFR
2283  38460          MVI   B,7            ;SET UP A COUNT
2284  38480          MVI   C,0            ;GET A ZERO TO FILL FAC WITH
2285  38500  DMLDV1: MOV   A,M            ;GET A BYTE FROM FAC
2286  38520          STAX  D              ;PUT IT IN FBUFFR
2287  38540          MOV   M,C            ;PUT A ZERO IN FAC
2288  38560          DCX   D              ;POINT TO NEXT BYTE IN FBUFFR
2289  38580          DCX   H              ;POINT TO NEXT LOWER ORDER BYTE IN FAC
2290  38600          DCR   B              ;ARE WE DONE?
2291  38620          JNZ   DMLDV1         ;NO, TRANSFER THE NEXT BYTE
2292  38640          RET                  ;ALL DONE
2293  38660
2294  38680
2295  38700          ;DOUBLE PRECISION MULTIPLY THE FAC BY 10
2296  38720          ;ALTERS ALL REGISTERS
2297  38740  DMUL10: CALL  VMOVAF         ;SAVE THE FAC IN ARG
2298  38760                               ;VMOVAF EXITS WITH (DE)=FAC+1
2299  39000          XCHG                 ;GET THE POINTER INTO THE FAC IN (HL)
2300  39020          DCX   H              ;POINT TO THE EXPONENT
2301  39040          MOV   A,M            ;GET THE EXPONENT
2302  39060          ADI   2              ;MULTIPLY FAC BY 4 BY ADDING 2 TO THE EXPONENT
2303  39080          JC    OVERR          ;CHECK FOR OVERFLOW
2304  39100          MOV   M,A            ;SAVE THE NEW EXPONENT
2305  39120          PUSH  H              ;SAVE POINTER TO FAC
2306  39140          CALL  DADD           ;ADD IN THE ORIGINAL FAC TO GET 5 TIMES  FAC
2307  39160          POP   H              ;GET THE POINTER TO FAC BACK
2308  39180          INR   M              ;ADD ONE TO EXPONENT TO GET 10 TIMES FAC
```

```
                                        38780           RNZ                     ;ALL DONE IF OVERFLOW DID NOT OCCUR
                                        38800           JMP     OVERR>          ;IT DID, GIVE THE APPROPRIATE MESSAGE
                                        38820   PAGE
```



```
                                        38840   SUBTTL  FLOATING POINT INPUT ROUTINE
                                        38860           ;ALTERS ALL REGISTERS
                                        38880           ;THE NUMBER IS LEFT IN FAC
                                        38900           ;AT ENTRY, (HL) POINTS TO THE FIRST CHARACTER IN A TEXT BUFFER,
                                        38920           ;THE FIRST CHARACTER IS ALSO IN A.  WE PACK THE DIGITS INTO THE FAC
                                        38940           ;AS AN INTEGER AND KEEP TRACK OF WHERE THE DECIMAL POINT IS,
                                        38960           ;C IS 377 IF WE HAVE NOT SEEN A DECIMAL POINT, 0 IF WE HAVE,
                                        38980           ;B IS THE NUMBER OF DIGITS AFTER THE DECIMAL POINT.
                                        39000           ;AT THE END, B AND THE EXPONENT (IN E) ARE USED TO DETERMINE HOW MANY
                                        39020           ;TIMES WE MULTIPLY OR DIVIDE BY TEN TO GET THE CORRECT NUMBER.
001531'                                 39040   FIN:
                                        39060           IFN     STRING,<
                                        39080           ;IF WE ARE CALLED BY VAL, THE SIGNS MAY NOT BE CRUNCHED
001531' 001000  000376                  39100           CPI     "-"             ;SEE IF NUMBER IS NEGATIVE
001532' 000000  000055
001533' 001000  000365                  39120           PUSH    PSW             ;SAVE SIGN
001534' 001000  000312                  39140           JZ      FIN1            ;IGNORE MINUS SIGN
001535' 000000  001545'
001536' 000000  001526'
001537' 001000  000376                  39160           CPI     "+"             ;IGNORE A LEADING SIGN
001540' 000000  000053
001541' 001000  000312                  39180           JZ      FIN1>
001542' 000000  001545'
001543' 000000  001535'
001544' 001000  000053                  39200           DCX     H               ;SET CHARACTER POINTER BACK ONE
001545'                                 39220   FIN1:
                                        39240           IFN     LENGTH-2,<
001545' 001000  000315                  39260           CALL    ZERO            ;CLEAR FAC
001546' 000000  000173'
001547' 000000  001542'
001550' 001000  000107                  39280           MOV     B,A             ;CLEAR FLAGS: B=DECIMAL PLACE COUNT
001551' 001000  000127                  39300           MOV     D,A             ;D=SIGN OF EXPONENT
001552' 001000  000137                  39320           MOV     E,A             ;E=EXPONENT
001553' 001000  000057                  39340           CMA
001554' 001000  000117                  39360           MOV     C,A             ;C="." FLAG
                                        39380           ;HERE TO GET THE NEXT DIGIT OF THE NUMBER, A DECIMAL POINT OR AN "E"
001555' 001000  000327                  39400   FINC:   CHRGET                  ;GET A CHARACTER
001556' 001000  000332                  39420           JC      FINDIG          ;DO WE HAVE A DIGIT?
001557' 000000  001702'
001560' 000000  001546'
001561' 001000  000376                  39440           CPI     "."             ;TEST FOR DECIMAL POINT
001562' 000000  000056
001563' 001000  000312                  39460           JZ      FINDP
001564' 000000  001635'
001565' 000000  001557'
001566' 001000  000376                  39480           CPI     "E"             ;CHECK FOR BEGINNING OF EXPONENT
001567' 000000  000105
001570' 001000  000302                  39500           JNZ     FINE            ;"NONE OF THE ABOVE" SO END OF NUMBER
001571' 000000  001641'
001572' 000000  001564'
                                        39520           ;HERE TO CHECK FOR THE SIGN OF THE EXPONENT
001573' 001000  000327                  39540           CHRGET                  ;CHECK FOR ITS SIGN
                                        39560   IFN     STRING,<
```