```
                                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
```