

```

    76          00100 SUBT1  FLOATING POINT MATH PACKAGE CONFIGURATION
    77          00120 TITLE  MATHPK FOR BASIC MCS 8080 GATES/ALLEN/DAVIDOFF
    78
    79          00160 IFNDEF LENGTH,<
    80              PRINTX_111 MUST HAVE COM 111
    81          END>
    82
    83          00240 RADIX 8           ;1111 ALERT 111
    84          00260                   ;THROUGHOUT THE MATHPACKAGE!!
    85
    86          00300 .P==0
    87
    88          00340 INTERNAL  ZERO,FLOAT,FLOATR,MUVE,FADD,FADDS,FSUB,FMULT,FDIV,FIN,FOUT
    89          00350 INTERNAL  PUSHF,ABS,INT,QINT,BSN,SGN,RND,SIN,FCOMP,SIGNC,OVERR
    90          00360 INTERNAL  INPRT,LINPRT,HOFVN,HOFVF,HOFVR,HOFVRF,MOVRF,MOVVR,NEG,INHART,INHRT
    91          00400 IFN  _EXTFNC,<
    92              00420 INTERNAL  FPWR,EXP,LOG,COS,TAN,ATN,FONE>
    93          00440 IFN  _HULDIM,<LENGTH=2>,<
    94          00460 INTERNAL  DMULT>
    95          00480 IFN  STRING,<
    96          00500 INTERNAL  SIGNS>
    97          00520 INTERNAL  LENGTH=2,<
    98          00540 INTERNAL  ADUT,FSUBT,FMULTT,FDIVT>
    99          00560 IFE  LENGTH=1,<
    100         00580 INTERNAL  FPWRTR>
    101         00600 IFE  LENGTH=2,<
    102         00620 INTERNAL  VMOVFM,VMOVVF,FRCINT,FRCNSG,FRCDBL,VNEG,PUFOUT,DCXBHT,IADD
    103         00640 INTERNAL  ISUB,IMULT,IDIV,ICOMP,INEG,DADD,DSUB,DMULT,DDIV,DCOMP,INTFNC>
    104
    105
    106         00700 EXTERNAL  FAC,FACLO,FBUFFFR,MINUTK,PLUSTK,ERROR,DY0ERR,ERRDV,FCERR,SIGN
    107         00720 EXTERNAL  SCODE
    108         00740 IFE  LENGTH=2,<
    109         00750 EXTERNAL  DFACLO,ARG,ARGLO,VALTYP,THERR,TEMP2,TEMP3>
    110
    111
    112         00820 COMMENT X
    113         00840 EXTERNAL LOCATIONS USED BY THE MATH-PACKAGE
    114             ;THE FLOATING ACCUMULATOR
    115         00860 IFE  LENGTH=2,<
    116             BLOCK 1           ;(TEMPORARY LEAST SIGNIFICANT BYTE)
    117             DFACLO: BLOCK 4> ;(FOUR LOWEST ORDERS FOR DOUBLE PRECISION)
    118         00940 FACLO: BLOCK 3           ;(LOW ORDER OF MANTISSA (LO))
    119         00960             DFACLO: BLOCK 3           ;(MIDDLE ORDER OF MANTISSA (MO))
    120         00980             DFACLO: BLOCK 3           ;(HIGH ORDER OF MANTISSA (HO))
    121         01000 FAC: BLOCK 2           ;(EXPONENT)
    122         01020             DFACLO: BLOCK 2           ;(TEMPORARY COMPLEMENT OF SIGN IN MSB)
    123
    124         01040 IFE  LENGTH=2,<
    125             ARGLO: BLOCK 7           ;(LOCATION OF SECOND ARGUMENT FOR DOUBLE
    126             ARG1: BLOCK 1>           ;PRECISION)
    127         01100 FBUFFFR: BLOCK ^D13 ;(BUFFER FOR FOUT
    128         01120 IFE  LENGTH=2,<BLOCK ^D<3@+13>> S

```

*Change to FB*

*X*

```

    129
    130         01160 THE FLOATING POINT FORMAT IS AS FOLLOWS:
    131
    132             01220 THE SIGN IS THE FIRST BIT OF THE MANTISSA
    133             01240 THE MANTISSA IS A 9-BIT LOGIC NUMBER
    134             01260 THE BINARY POINT IS TO THE LEFT OF THE MSB
    135             01280 NUMBER = MANTISSA * 2 ^ EXPONENT
    136             01300 THE MANTISSA IS POSITIVE, WITH A ONE ASSUMED TO BE WHERE THE SIGN BIT IS
    137             01320 THE SIGN OF THE EXPONENT IS THE FIRST BIT OF THE EXPONENT
    138             01340 THE EXPONENT IS STORED IN EXCESS 200 I.E., WITH A BIAS OF 200
    139             01360 SO, THE EXPONENT IS A SIGNED 8-BIT NUMBER WITH 200 ADDED TO IT
    140             01380 AN EXPONENT OF ZERO MEANS THE NUMBER IS ZERO, THE OTHER BYTES ARE IGNORED
    141             01400 TO KEEP THE SAME NUMBER IN THE FAC WHILE SHIFTING:
    142             01420 TO SHIFT RIGHT, EXP1=EXP+1
    143             01440 TO SHIFT LEFT, EXP1=EXP-1
    144
    145             01480 SO, IN MEMORY THE NUMBER LOOKS LIKE THIS:
    146                 [BITS 17-24 OF THE MANTISSA]
    147                 [BITS 9-16 OF THE MANTISSA]
    148                 [THE SIGN IN BIT 7, BITS 2-6 OF THE MANTISSA ARE IN BITS 6-0]
    149                 [THE EXPONENT AS A SIGNED NUMBER + 200]
    150             01580 (REMEMBER THAT BIT 1 OF THE MANTISSA IS ALWAYS A ONE)
    151
    152             01620 ARITHMETIC ROUTINE CALLING CONVENTIONS:
    153
    154             01660 FOR ONE ARGUMENT FUNCTIONS:
    155                 01680     THE ARGUMENT IS IN THE FAC, THE RESULT IS LEFT IN THE FAC
    156             01700 FOR TWO ARGUMENT OPERATIONS:
    157                 01720     THE FIRST ARGUMENT IS IN B,C,D,E I.E., THE "REGISTERS"
    158                 01740     THE SECOND ARGUMENT IS IN THE FAC
    159                 01760     THE RESULT IS LEFT IN THE FAC
    160
    161             01800 THE "I" ENTRY POINTS TO THE TWO ARGUMENT OPERATIONS HAVE (ML) POINTING TO
    162             THE FIRST ARGUMENT INSTEAD OF THE FIRST ARGUMENT BEING IN THE REGISTERS,
    163             MOVRM IS CALLED TO GET THE ARGUMENT IN THE REGISTERS,
    164             THE "I" ENTRY POINTS ASSUME THE FIRST ARGUMENT IS ON THE STACK,
    165             PGP IS USED TO GET THE ARGUMENT IN THE REGISTERS,
    166             NOTE THE "R" ENTRY POINTS SHOULD ALWAYS BE JUMPED TO AND NEVER CALLED
    167             BECAUSE THE RETURN ADDRESS ON THE STACK WILL BE CONFUSED WITH THE NUMBER,
    168
    169             01960 ON THE STACK, THE TWO LO'S ARE PUSHED ON FIRST AND THEN THE HI AND SIGN,
    170             THIS IS DONE SO IF A NUMBER IS STORED IN MEMORY, IT CAN BE PUSHED ON THE
    171             STACK WITH TWO PUSHES;, THE LOWER BYTE OF EACH PART IS IN THE LOWER
    172             MEMORY ADDRESS SO WHEN THE NUMBER IS POPPED INTO THE REGISTERS, THE HIGHER
    173             ORDER BYTE WILL BE IN THE HIGHER ORDER REGISTER OF THE REGISTER PAIR, I.E.,
    174             02050     THE HIGHER ORDER BYTE WILL BE POPPED INTO B, D OR H,
    175             02060     X
    176             02100 PAGE

```

MATHPK FOR BASIC MCS 8080 GATES/ALLEN/DAVIDOFF MACRO 47(\$13) 06:09 27-AUG-75 PAGE 2  
F4 MAC 23-AUG-64 06:08 FLOATING POINT ADDITION AND SUBTRACTION

```

02120 SUBTL FLOATING POINT ADDITION AND SUBTRACTION
02140 JENTRY TO FADD WITH POINTER TO ARG IN (HL)
02160 FAADD: LXI H,FHALF           JENTRY TO ADD 1/2

02180 FADDS: CALL MOVRM          JGET ARGUMENT INTO THE REGISTERS

02200 JMP FADD                 JDO THE ADDITION

02260 JSUBTRACTION FAC:=ARG+FAC
02280 IFN EXTFNC,<
02300 FSUBS1 CALL MOVRM>        JENTRY IF POINTER TO ARG IS IN (HL)

02320 IFE LENGTH=1,<
02340 XWD 1000,041>             J"LDI H" AROUND NEXT 2 BYTES
02360 IFN LENGTH=2,<
02380 FSUBT1 PUPH>              JENTRY IF ARGUMENT IS ON THE STACK

02400 FSUB: CALL NEG            JNEGATE SECOND ARGUMENT

02420 JFALL INTO FADD

02480 JADDITION FAC:=ARG+FAC
02500 JALTERS A,B,C,D,E,H,L
02520 IFN LENGTH=2,<
02540 XWD 1000,041              J"LDI H" AROUND NEXT 2 BYTES
02560 FAADD: PUPH>              JENTRY IF ARGUMENT IS ON THE STACK

02580 FADD: MUV A,B             JCHECK IF FIRST ARGUMENT IS ZERO
02600 ORA A                   JGET EXPONENT
02620 RZ                      JIT IS, RESULT IS NUMBER IN FAC
02640 LDA FAC                  JGET EXPONENT

02660 ORA A                   JSEE IF THE NUMBER IS ZERO
02680 JZ MUVR                JIT IS, ANSWER IS IN REGISTERS

02720 JWE WANT TO GET THE SMALLER NUMBER IN THE REGISTERS SO WE CAN SHIFT IT
02740 JAND ALIGN THE BINARY POINTS OF THE TWO NUMBERS, THEN WE CAN JUST ADD
02760 JSUBTRACT THEM (DEPENDING ON THEIR SIGNS) BYTewise,
02780 SUB B                   JCHECK RELATIVE SIZES
02800 JNC FAADD               JIS FAC SMALLER?

```

MATHPK FOR BASIC MCS 8080 GATES/ALLEN/DAVIDOFF MACRO 47(113) 06:09 27-AUG-75 PAGE 2-1  
F4 MAC 23-AUG-64 06:08 FLOATING POINT ADDITION AND SUBTRACTION

```

DAVIDOFF MACRO 47([15]) 08:09 27-AUG-75 PAGE 2-1
FLOATING POINT ADDITION AND SUBTRACTION

02820      CMA           YES, NEGATE SHIFT COUNT
02840      INR          A
02860      XCHG          ;SWITCH FAC AND REGISTERS, SAVE (DE)
02880      CALL          PUSHF
                           ;PUT FAC ON STACK

02900      XCHG          ;GET (DE) BACK WHERE IT BELONGS
02920      CALL          MOVFR
                           ;PUT REGISTERS IN THE FAC

02940      PUPR          ;GET THE OLD FAC IN THE REGISTERS

02960      FADD1:        IFN
02980      IFN          LENGTH,<
03000      CPI          31          FARE WE WITHIN 24 BITS?
03020      RNC>          ;NO, ALL DONE
03040      PUSH          PSW          ;SAVE SHIFT COUNT
03060      CALL          UNPACK        ;UNPACK THE NUMBERS

03080      MOV          M,A          ;SAVE SUBTRACTION FLAG
03100      POP          PSW          ;GET SHIFT COUNT BACK
03120      CALL          SHIFT         ;SHIFT REGISTERS RIGHT THE RIGHT AMOUNT

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
03200      ;IMANTISSAS ARE POSITIVE.  JUDGING BY THE EXPONENTS, THE LARGER NUMBER
03220      ;IS THE FAC, SO IF WE SUBTRACT, THE SIGN OF THE RESULT SHOULD BE THE SIGN
03240      ;OF THE 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      ;IS 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.

03380      ORA          H          ;GET SUBTRACTION FLAG
03400      LXI          M,FACD        ;SET POINTER TO LOADS

```

			LAST FINGER IS LOG 0
03420	JP	FADDS	FSUBTRACT IF THE SIGNS WERE DIFFERENT
03440	CALL	FADDA	FADD THE NUMBERS
03460	JNC	ROUND	FROUND RESULT IF THERE WAS NO OVERFLOW
03480			FTHE MOST IT CAN OVERFLOW IS ONE BIT
03500	INX	H	FTHERE WAS OVERFLOW

```

    263 000111* 001000 000054      05520 INR M           ;INCREMENT EXPONENT
    264 000112* 001000 000312      05540 JZ OVERR       ;CHECK FOR OVERFLOW
    265 000113* 000000 000267*      05560 IFE LENGTH,<
    266 000114* 000000 000106*      05580 CALL SHFTRD>   ;SHIFT RESULT RIGHT ONE, SHIFT CARRY IN
    267
    268
    269 000115* 001000 000056      05600 IFN LENGTH,<
    270 000116* 000000 000001      05620 MOV L,1         ;SHIFT RESULT RIGHT ONE, SHIFT CARRY IN
    271 000117* 001000 000315      05640 CALL SHRADD>
    272 000120* 000000 000362*      05660 JMP ROUND       ;ROUND RESULT AND WE ARE DONE
    273
    274
    275 000121* 001000 000113*      05680 JHERE TO SUBTRACT C,D,E,B FROM ((HL)+0,1,2),B
    276 000122* 001000 000305      05700 FAD03: XRA A     ;SUBTRACT NUMBERS, NEGATE UNDERFLOW BYTE
    277 000123* 000000 000000      05720 SUB B
    278 000124* 001000 000257      05740 MOV B,A         ;SAVE IT
    279 000125* 001000 000000      05760 MOV A,M         ;SUBTRACT LOW ORDERS
    280 000126* 001000 000233      05780 SBB E
    281 000127* 001000 000000      05800 MOV E,A
    282 000130* 001000 000176      05820 INX H         ;UPDATE POINTER TO NEXT BYTE
    283 000131* 001000 000253      05840 MOV H,A
    284 000132* 001000 000000      05860 SBB D
    285 000133* 001000 000000      05880 MOV D,A
    286 000134* 001000 000000      05900 INX H         ;UPDATE POINTER TO HIGH ORDERS
    287 000135* 001000 000000      05920 MOV A,M
    288 000136* 001000 000000      05940 SBB C
    289 000137* 001000 000000      05960 MOV C,A
    290 000138* 001000 000000      05980 INX H
    291 000139* 001000 000000      06000 MOV H,A
    292 000140* 001000 000176      06020 SBB D
    293 000141* 001000 000231      06040 MOV D,A
    294 000142* 001000 000017      06060 INX H
    295 000143* 001000 000334      06080 MOV A,M
    296 000144* 000000 000310*      06100 FADFLT: CC NEGR ;ENTRY FROM FLOATR, INT: NEGATE NUMBER IF IT
    297 000145* 000000 000123*      06120
    298
    299 000146* 001000 000000      06140 ;WAS NEGATIVE, FALL INTO NORMALIZE
    300
    301 000147* 001000 000143      06160
    302 000148* 001000 000000      06180
    303 000149* 001000 000000      06200
    304 000150* 001000 000000      06220
    305 000151* 001000 000107      06240
    306 000152* 001000 000171      06260
    307 000153* 001000 000267      06280
    308 000154* 001000 000302      06300
    309 000155* 000000 000210*      06320
    310 000156* 000000 000144*      06340
    311
    312 000157* 001000 000000      06360
    313 000158* 001000 000000      06380
    314 000159* 001000 000000      06400
    315 000143* 001000 000334      06420
    316 000144* 000000 000310*      06440
    317 000145* 000000 000123*      06460
    318
    319
    320
    321 000146* 001000 000000      06480 ;NORMALIZE C,D,E,B
    322 000147* 001000 000143      06500 ;ALTERS A,B,C,D,E,H,L
    323 000148* 001000 000000      06520 ;HERE WE SHIFT THE MANTISSA LEFT UNTIL THE MSB IS A ONE,
    324 000149* 001000 000000      06540 ;EXCEPT IN 4K, THE IDEA IS TO SHIFT LEFT BY 8 AS MANY TIMES AS
    325 000150* 001000 000000      06560 ;POSSIBLE.
    326 000146* 001000 000000      06580 NORMAL1: IFE LENGTH,<
    327 000147* 001000 000143      06600 MOV H,0         ;CLEAR SHIFT COUNT
    328 000148* 001000 000000      06620 MOV A,C         ;IS THE NUMBER NORMALIZED?
    329 000149* 001000 000000      06640 DRA A
    330 000150* 001000 000000      06660 JH ROUND       ;YES, WE ARE DONE
    331 000151* 001000 000000      06680 CPI 340        ;IS THE RESULT ZERO?
    332 000152* 001000 000000      06700 JZ ZERO        ;YES, ZERO THE FAC
    333 000153* 001000 000000      06720 DCR H
    334 000154* 001000 000000      06740 INO, DECREMENT SHIFT COUNT
    335 000155* 001000 000000      06760 MOV A,B
    336
    337
    338
    339
    340
    341
    342 000146* 001000 000150      06780
    343 000147* 001000 000143      06800
    344 000150* 001000 000257      06820
    345 000151* 001000 000107      06840 NORM1: XRA A     ;ZERO SHIFT COUNT
    346 000152* 001000 000171      06860 MOV B,A
    347 000153* 001000 000267      06880 ISAVE SHIFT COUNT
    348 000154* 001000 000302      06900 DRA A
    349 000155* 000000 000210*      06920 INO, DECREMENT SHIFT COUNT
    350 000156* 000000 000144*      06940
    351
    352 000157* 001000 000112      06960 ;THIS LOOP SPEEDS THINGS UP, BY SHIFTING 8 PLACES AT ONE TIME
    353 000158* 001000 000124      06980 MOV C,D
    354 000159* 001000 000145      07000 ISAVE SHIFT COUNT
    355 000160* 001000 000157      07020 MOV H,L
    356 000161* 001000 000176      07040 MOV L,A
    357 000162* 001000 000267      07060 MOV A,B
    358 000163* 000000 000326      07080 ISAVE SHIFT COUNT
    359 000164* 000000 000146*      07100 SUI 10
    360 000165* 000000 000146*      07120
    361 000166* 000000 000340      07140 CPI 340
    362 000167* 000000 000340      07160 ISAVE SHIFT COUNT
    363 000168* 001000 000302      07180 JNZ NORM1
    364 000169* 000000 000155*      07200 ;NO, TRY TO SHIFT OVER 8 MORE
    365
    366
    367 000170* 001000 000000      07220
    368 000171* 001000 000000      07240
    369 000172* 001000 000000      07260
    370 000173* 001000 000000      07280 ;BY OUR FLOATING POINT FORMAT, THE NUMBER IS ZERO IF THE EXPONENT IS
    371 000174* 001000 000000      07300 ;ZERO
    372 000175* 001000 000257      07320 ZERO1: XRA A     ;ZERO A
    373 000176* 000000 000002      07340 STA FAC
    374 000177* 000000 000031*      07360 ISAVE SHIFT COUNT
    375 000178* 000000 000002      07380
    376 000179* 001000 000000      07400
    377
    378 000180* 001000 000000      07420 RET
    379 000200* 001000 000005      07440 JALL DONE
    380 000201* 001000 000051      07460
    381 000202* 001000 000172      07480 NORM2: DCR B
    382 000203* 001000 000007      07500 DAD H
    383 000204* 001000 000127      07520 ISAVE SHIFT COUNT
    384 000205* 001000 000171      07540 RAL
    385 000206* 001000 000171      07560 MOV D,A
    386 000207* 001000 000171      07580 AUC A
    387 000208* 001000 000362*      07600 ISAVE SHIFT COUNT
    388 000209* 001000 000362*      07620 NORM3: JP NORM2
    389
    390
    391
    392
    393
    394
    395
    396
    397
    398
    399
    400
    401
    402
    403
    404
    405
    406
    407
    408
    409
    410
    411
    412
    413
    414
    415
    416
    417
    418
    419
    420
    421
    422
    423
    424
    425
    426
    427
    428
    429
    430
    431
    432
    433
    434
    435
    436
    437
    438
    439
    440
    441
    442
    443
    444
    445
    446
    447
    448
    449
    450
    451
    452
    453
    454
    455
    456
    457
    458
    459
    460
    461
    462
    463
    464
    465
    466
    467
    468
    469
    470
    471
    472
    473
    474
    475
    476
    477
    478
    479
    480
    481
    482
    483
    484
    485
    486
    487
    488
    489
    490
    491
    492
    493
    494
    495
    496
    497
    498
    499
    500
    501
    502
    503
    504
    505
    506
    507
    508
    509
    510
    511
    512
    513
    514
    515
    516
    517
    518
    519
    520
    521
    522
    523
    524
    525
    526
    527
    528
    529
    530
    531
    532
    533
    534
    535
    536
    537
    538
    539
    540
    541
    542
    543
    544
    545
    546
    547
    548
    549
    550
    551
    552
    553
    554
    555
    556
    557
    558
    559
    560
    561
    562
    563
    564
    565
    566
    567
    568
    569
    570
    571
    572
    573
    574
    575
    576
    577
    578
    579
    580
    581
    582
    583
    584
    585
    586
    587
    588
    589
    590
    591
    592
    593
    594
    595
    596
    597
    598
    599
    600
    601
    602
    603
    604
    605
    606
    607
    608
    609
    610
    611
    612
    613
    614
    615
    616
    617
    618
    619
    620
    621
    622
    623
    624
    625
    626
    627
    628
    629
    630
    631
    632
    633
    634
    635
    636
    637
    638
    639
    640
    641
    642
    643
    644
    645
    646
    647
    648
    649
    650
    651
    652
    653
    654
    655
    656
    657
    658
    659
    660
    661
    662
    663
    664
    665
    666
    667
    668
    669
    670
    671
    672
    673
    674
    675
    676
    677
    678
    679
    680
    681
    682
    683
    684
    685
    686
    687
    688
    689
    690
    691
    692
    693
    694
    695
    696
    697
    698
    699
    700
    701
    702
    703
    704
    705
    706
    707
    708
    709
    710
    711
    712
    713
    714
    715
    716
    717
    718
    719
    720
    721
    722
    723
    724
    725
    726
    727
    728
    729
    730
    731
    732
    733
    734
    735
    736
    737
    738
    739
    740
    741
    742
    743
    744
    745
    746
    747
    748
    749
    750
    751
    752
    753
    754
    755
    756
    757
    758
    759
    760
    761
    762
    763
    764
    765
    766
    767
    768
    769
    770
    771
    772
    773
    774
    775
    776
    777
    778
    779
    780
    781
    782
    783
    784
    785
    786
    787
    788
    789
    790
    791
    792
    793
    794
    795
    796
    797
    798
    799
    800
    801
    802
    803
    804
    805
    806
    807
    808
    809
    810
    811
    812
    813
    814
    815
    816
    817
    818
    819
    820
    821
    822
    823
    824
    825
    826
    827
    828
    829
    830
    831
    832
    833
    834
    835
    836
    837
    838
    839
    840
    841
    842
    843
    844
    845
    846
    847
    848
    849
    850
    851
    852
    853
    854
    855
    856
    857
    858
    859
    860
    861
    862
    863
    864
    865
    866
    867
    868
    869
    870
    871
    872
    873
    874
    875
    876
    877
    878
    879
    880
    881
    882
    883
    884
    885
    886
    887
    888
    889
    890
    891
    892
    893
    894
    895
    896
    897
    898
    899
    900
    901
    902
    903
    904
    905
    906
    907
    908
    909
    910
    911
    912
    913
    914
    915
    916
    917
    918
    919
    920
    921
    922
    923
    924
    925
    926
    927
    928
    929
    930
    931
    932
    933
    934
    935
    936
    937
    938
    939
    940
    941
    942
    943
    944
    945
    946
    947
    948
    949
    950
    951
    952
    953
    954
    955
    956
    957
    958
    959
    960
    961
    962
    963
    964
    965
    966
    967
    968
    969
    970
    971
    972
    973
    974
    975
    976
    977
    978
    979
    980
    981
    982
    983
    984
    985
    986
    987
    988
    989
    990
    991
    992
    993
    994
    995
    996
    997
    998
    999
    1000
    1001
    1002
    1003
    1004
    1005
    1006
    1007
    1008
    1009
    1010
    1011
    1012
    1013
    1014
    1015
    1016
    1017
    1018
    1019
    1020
    1021
    1022
    1023
    1024
    1025
    1026
    1027
    1028
    1029
    1030
    1031
    1032
    1033
    1034
    1035
    1036
    1037
    1038
    1039
    1040
    1041
    1042
    1043
    1044
    1045
    1046
    1047
    1048
    1049
    1050
    1051
    1052
    1053
    1054
    1055
    1056
    1057
    1058
    1059
    1060
    1061
    1062
    1063
    1064
    1065
    1066
    1067
    1068
    1069
    1070
    1071
    1072
    1073
    1074
    1075
    1076
    1077
    1078
    1079
    1080
    1081
    1082
    1083
    1084
    1085
    1086
    1087
    1088
    1089
    1090
    1091
    1092
    1093
    1094
    1095
    1096
    1097
    1098
    1099
    1100
    1101
    1102
    1103
    1104
    1105
    1106
    1107
    1108
    1109
    1110
    1111
    1112
    1113
    1114
    1115
    1116
    1117
    1118
    1119
    1120
    1121
    1122
    1123
    1124
    1125
    1126
    1127
    1128
    1129
    1130
    1131
    1132
    1133
    1134
    1135
    1136
    1137
    1138
    1139
    1140
    1141
    1142
    1143
    1144
    1145
    1146
    1147
    1148
    1149
    1150
    1151
    1152
    1153
    1154
    1155
    1156
    1157
    1158
    1159
    1160
    1161
    1162
    1163
    1164
    1165
    1166
    1167
    1168
    1169
    1170
    1171
    1172
    1173
    1174
    1175
    1176
    1177
    1178
    1179
    1180
    1181
    1182
    1183
    1184
    1185
    1186
    1187
    1188
    1189
    1190
    1191
    1192
    1193
    1194
    1195
    1196
    1197
    1198
    1199
    1200
    1201
    1202
    1203
    1204
    1205
    1206
    1207
    1208
    1209
    1210
    1211
    1212
    1213
    1214
    1215
    1216
    1217
    1218
    1219
    1220
    1221
    1222
    1223
    1224
    1225
    1226
    1227
    1228
    1229
    1230
    1231
    1232
    1233
    1234
    1235
    1236
    1237
    1238
    1239
    1240
    1241
    1242
    1243
    1244
    1245
    1246
    1247
    1248
    1249
    1250
    1251
    1252
    1253
    1254
    1255
    1256
    1257
    1258
    1259
    1260
    1261
    1262
    1263
    1264
    1265
    1266
    1267
    1268
    1269
    1270
    1271
    1272
    1273
    1274
    1275
    1276
    1277
    1278
    1279
    1280
    1281
    1282
    1283
    1284
    1285
    1286
    1287
    1288
    1289
    1290
    1291
    1292
    1293
    1294
    1295
    1296
    1297
    1298
    1299
    1300
    1301
    1302
    1303
    1304
    1305
    1306
    1307
    1308
    1309
    1310
    1311
    1312
    1313
    1314
    1315
    1316
    1317
    1318
    1319
    1320
    1321
    1322
    1323
    1324
    1325
    1326
    1327
    1328
    1329
    1330
    1331
    1332
    1333
    1334
    1335
    1336
    1337
    1338
    1339
    1340
    1341
    1342
    1343
    1344
    1345
    1346
    1347
    1348
    1349
    1350
    1351
    1352
    1353
    1354
    1355
    1356
    1357
    1358
    1359
    1360
    1361
    1362
    1363
    1364
    1365
    1366
    1367
    1368
    1369
    1370
    1371
    1372
    1373
    1374
    1375
    1376
    1377
    1378
    1379
    1380
    1381
    1382
    1383
    1384
    1385
    1386
    1387
    1388
    1389
    1390
    1391
    1392
    1393
    1394
    1395
    1396
    1397
    1398
    1399
    1400
    1401
    1402
    1403
    1404
    1405
    1406
    1407
    1408
    1409
    1410
    1411
    1412
    1413
    1414
    1415
    1416
    1417
    1418
    1419
    1420
    1421
    1422
    1423
    1424
    1425
    1426
    1427
    1428
    1429
    1430
    1431
    1432
    1433
    1434
    1435
    1436
    1437
    1438
    1439
    1440
    1441
    1442
    1443
    1444
    1445
    1446
    1447
    1448
    1449
    1450
    1451
    1452
    1453
    1454
    1455
    1456
    1457
    1458
    1459
    1460
    1461
    1462
    1463
    1464
    1465
    1466
    1467
    1468
    1469
    1470
    1471
    1472
    1473
    1474
    1475
    1476
    1477
    1478
    1479
    1480
    1481
    1482
    1483
    1484
    1485
    1486
    1487
    1488
    1489
    1490
    1491
    1492
    1493
    1494
    1495
    1496
    1497
    1498
    1499
    1500
    1501
    1502
    1503
    1504
    1505
    1506
    1507
    1508
    1509
    1510
    1511
    1512
    1513
    1514
    1515
    1516
    1517
    1518
    1519
    1520
    1521
    1522
    1523
    1524
    1525
    1526
    1527
    1528
    1529
    1530
    1531
    1532
    1533
    1534
    1535
    1536
    1537
    1538
    1539
    1540
    1541
    1542
    1543
    1544
    1545
    1546
    1547
    1548
    1549
    1550
    1551
    1552
    1553
    1554
    1555
    1556
    1557
    1558
    1559
    1560
    1561
    1562
    1563
    1564
    1565
    1566
    1567
    1568
    1569
    1570
    1571
    1572
    1573
    1574
    1575
    1576
    1577
    1578
    1579
    1580
    1581
    1582
    1583
    1584
    1585
    1586
    1587
    1588
    1589
    1590
    1591
    1592
    1593
    1594
    1595
    1596
    1597
    1598
    1599
```

```

    389  000212* 000000  000175*
    390  000213* 001000  000170      05280  MDV A,B      FALL NORMALIZED, GET SHIFT COUNT
    391  000214* 001000  000134      05300  MOV E,M      ;PUT LOTS BACK IN E,B
    392  000215* 001000  000105      05320  MOV B,L      ;CHECK IF WE DID NO SHIFTING
    393  000216* 001000  000267      05340  ORA A
    394  000217* 001000  000112      05360  JZ ROUND>
    395  000220* 000000  000253*
    396  000221* 001000  000211*   05380  LXI H,FAC      FLOOR AT FAC'S EXPONENT
    397  000222* 001000  000041
    398  000223* 001000  000175*
    399  000224* 000000  000220*
    400  000225* 001000  000206      05400  ADD M      UPDATE EXPONENT
    401  000226* 001000  000167      05420  MOV M,A
    402  000227* 001000  000322      05440  JNC ZERO      ;CHECK FOR UNDERFLOW
    403  000230* 000000  000173*
    404  000231* 000000  000223*
    405  000232* 001000  000310      05460  RZ      INNUMBER IS ZERO, ALL DONE
    406  000233* 001000  000170      05480
    407
    408
    409  000234* 001000  000041      05500  ROUND RESULT IN C,D,E,B AND PUT NUMBER IN THE FAC
    410  000235* 001000  000041      05520  FALTERS A,B,C,D,E,M,L
    411  000236* 001000  000223*      05540  ;THE ROUND C,D,E UP OR DOWN DEPENDING UPON THE MSB OF B
    412  000237* 001000  000267      05560  ROUND1 MOV A,B      ;SEE IF WE SHOULD ROUND UP
    413  000238* 001000  000041      05580  ROUNDDBI LXI H,FAC      ;ENTRY FROM FDIV, GET POINTER TO EXPONENT
    414  000239* 000000  000230*
    415  000240* 000000  000267
    416  000241* 001000  000267      05640  ORA A
    417  000242* 001000  000374      05660  CM ROUND1      DO IT IF NECESSARY
    418  000243* 000000  000235*
    419  000244* 001000  000106      05680  MOV B,M      ;PUT EXPONENT IN B
    420  000245* 001000  000106      05700  ;WHERE WE PACK THE HQ AND SIGN
    421  000246* 001000  000043      05720  INX H      ;POINT TO SIGN
    422  000247* 001000  000176      05740  MOV A,M      ;GET SIGN
    423  000248* 001000  000346      05760  ANI 208      ;GET RID OF UNWANTED BITS
    424  000249* 001000  000268
    425  000250* 001000  000251      05780  XRA C      ;PACK SIGN AND HQ
    426  000251* 001000  000251      05800  MOV C,A      ;SAVE IT IN C
    427  000252* 001000  000303      05820  JMP MUVR      ;SAVE NUMBER IN FAC
    428  000253* 000000  001229*
    429  000254* 000000  000241*
    430
    431
    432
    433  05860  IFE LENGTH,<
    434  05900  ;SHIFT C,D,E LEFT ONE
    435  05920  ;THIS IS USED BY NORMAL, FDIV
    436  05940  ;FALTERS A,C,D,E
    437  05960  SHFTL01 MOV A,E      ;GET THE LO
    438  05980  RAL                   ;SHIFT IT
    439  06000  MOV E,A      ;SAVE IT
    440  06020  MOV A,D      ;SHIFT THE NEXT HIGHER ORDER
    441  06040  RAL

```

```

    442  000260  MOV D,A      ;SHIFT THE HIGHEST ORDER
    443  000260  MOV A,C
    444  00100  ADC A      ;ROTATE A LEFT AND SET CONDITION CODES
    445  00120  MOV C,A
    446  00140  RET>      ;FALL DONE
    447
    448
    449  05620  ;SUBROUTINE FOR ROUND1: ADD ONE TO C,D,E
    450  000255* 001000  000034      056220 ROUND1 INH E      ;ADD ONE TO THE LOW ORDER, ENTRY FROM QINT
    451  000256* 001000  000306      056240 RNZ
    452  000257* 001000  000024      056260 INR D      ;ADD ONE IF IT IS NOT ZERO
    453  000258* 001000  000306      056280 RNZ
    454  000259* 001000  000014      056300 INR C      ;ADD ONE TO NEXT HIGHER ORDER
    455  000260* 001000  000000      056320 RNZ
    456  000261* 001000  000015      056340 HVI C,208      ;ADD ONE TO THE HIGHEST ORDER
    457  000262* 001000  000220
    458  000263* 001000  000664      056360 INH M      ;UPDATE EXPONENT
    459  000264* 001000  000306      056380 RNZ
    460  000265* 001000  000306      056400
    461
    462  056440  ;OVERFLOW ERROR
    463  000267* 001000  000036      056460 OVERN: HVI E,ENRV      ;SET OVERFLOW ERROR CODE
    464  000270* 001000  000000*
    465  000271* 001000  000363      056480 JMP ERROR      ;GO TO IT!!
    466  000272* 001000  000000*
    467  000273* 000000  000253*
    468
    469
    470  056500  ;FADD (HL)+2,1,0 TO C,D,E
    471  000274* 001000  000176      056520 FADD1 MOV E,M      ;THIS CODE IS USED BY FADD, FOUT
    472  000275* 001000  000043      056540 ADD B      ;GET LOWEST ORDER
    473  000276* 001000  000043      056560 ADD B      ;ADD IN OTHER LOWEST ORDER
    474  000277* 001000  000157      056580 MOV E,A      ;SAVE IT
    475  000278* 001000  000043      056600 INX H      ;UPDATE POINTER TO NEXT BYTE
    476  000279* 001000  000176      056620 MOV A,M      ;ADD MIDDLE ORDERS
    477  000280* 001000  000212      056640 ADC D
    478  000281* 001000  000127      056660 MOV D,A
    479  000282* 001000  000443      056680 INX H      ;UPDATE POINTER TO HIGH ORDER
    480  000283* 001000  000176      056700 MOV E,M      ;ADD HIGH ORDERS
    481  000284* 001000  000211      056720 ADC C
    482  000285* 001000  000117      056740 MOV C,A
    483  000286* 001000  000311      056760 RET
    484
    485
    486  056800  ;NEGATE NUMBER IN C,D,E,B
    487  000310* 001000  000041      056820 ;THIS CODE IS USED BY FADD, QINT
    488  000311* 000000  000001*
    489  000312* 000000  000072*
    490  000313* 001000  000176      056940 MOV A,M      ;GET SIGN
    491  000314* 001000  000057      056960 CMA      ;COMPLEMENT IT
    492  000315* 001000  000176      056980 MOV M,A      ;SAVE IT AGAIN
    493  000316* 001000  000167
    494  000317* 001000  000167

```

MATHPK FOR BASIC MCS 8080 GATES/ALLEN/DAVIDOFF MACRO 47(113) 06:09 27-AUG-75 PAGE 2-6  
F4 MAC 23-AUG-64 06:08 FLOATING POINT ADDITION AND SUBTRACTION

```

495 000316# 001000 000057 070200 XRA A ;ZERO A
496 000317# 001000 000157 070200 MOV L,A ;SAVE ZERO IN L
497 000320# 001000 000220 070400 SUB B ;NEGATE LOWEST ORDER
498 000321# 001000 000180 070600 MOV B,A ;SAVE IT
499 000322# 001000 000175 070700 MOV A,L ;GET A ZERO
500 000323# 001000 000235 071000 SBB E ;NEGATE NEXT HIGHEST ORDER
501 000324# 001000 000137 071200 MOV E,A ;SAVE IT
502 000325# 001000 000175 071400 MOV A,L ;GET A ZERO
503 000326# 001000 000232 071600 SBB D ;NEGATE NEXT HIGHEST ORDER
504 000327# 001000 000127 071800 MOV D,A ;SAVE IT
505 000330# 001000 000101 072000 MOV A,L ;GET ZERO BACK
506 000331# 001000 000231 072200 SBB C ;NEGATE HIGHEST ORDER
507 000332# 001000 000117 072400 MOV C,A ;SAVE IT
508 000333# 001000 000311 072600 RET ;FALL DONE
509

510
511 075200 ;SHIFFT C,D,E RIGHT
512 073400 JA = SHIFT COUNT
513 073500 ;TAKE IN A,B,C,D,E,L
514 073600 ;THE (CARRY (EXCEPT IN 4K) IS TO SHIFT RIGHT 8 PLACES AS MANY TIMES AS
515 074000 ; POSSIBLE
516 000334# 001000 000006 074200 SHFTR1: MVI B,0 ;ZERO OVERFLOW BYTE
517 000335# 001000 000000
518 074400 IFE LENGTH,<
519 074600 INR A> ;ADD ONE TO SHIFT COUNT
520 074800 IFN LENGTH,>
521 000336# 001000 000326 075000 SHFTR1: SUI 10 ;CAN WE SHIFT IT 8 RIGHT?
522 000337# 001000 000010
523 000340# 001000 000532 075200 JC SHFTR2 ;NO, SHIFT IT ONE PLACE AT A TIME
524 000341# 001000 000535# ;000311#
525 000342# 001000 000000
526 075400 ;THIS LOOP SPEEDS THINGS UP BY SHIFTING 8 PLACES AT ONE TIME
527 000343# 001000 000103 075600 MOV B,E ;SHIFT NUMBER 1 BYTE RIGHT
528 000344# 001000 000132 075800 MOV E,D
529 000345# 001000 000121 076000 MOV D,C
530 000346# 001000 00016 076200 MVI C,0 ;PUT 0 IN HO
531 000347# 001000 000008
532 000350# 001000 000303 076400 JMP SHFTR1 ;TRY TO SHIFT 8 RIGHT AGAIN
533 000351# 001000 000336# ;000311#
534 000352# 001000 000337# ;000311#
535 000354# 001000 000300 076600 SHFTR2: ADI 11> ;CORRECT SHIFT COUNT
536 000355# 001000 000811
537 000355# 001000 000157 076800 MOV L,A ;SAVE SHIFT COUNT
538 000356# 001000 000257 077000 SHFTR3: XHA A ;CLEAR CARRY
539 000357# 001000 000055 077200 DCR L ;ARE WE DONE SHIFTING?
540 000360# 001000 000310 077400 RZ ;RETURN IF WE ARE
541
542 077600 IFE LENGTH,<
543 077800 SHRADD: ADDL SHFTRD> ;SHIFT THE NUMBER RIGHT ONE
544 000361# 001000 000171 078200 MOV A,C ;GET HO
545 000362# 001000 000257 078400 SHRADD: RAR ;ENTRY FROM FADD, SHIFT IT RIGHT
546 000363# 001000 000117 078600 MOV C,A ;SAVE IT
547 000364# 001000 000172 078800 MOV A,D ;SHIFT NEXT BYTE RIGHT

```

MATHPK FOR BASIC MCS 8080 GATES/ALLEN/DAVIDOFF MACRO 47(113) 06:09 27-AUG-75 PAGE 2-7  
F4 MAC 23-AUG-64 06:08 FLOATING POINT ADDITION AND SUBTRACTION

```

548 000362# 001000 000037          07900   HAR
549 000366# 001000 000127          07920   MOV    D,A
550 000367# 001000 000173          07940   MOV    A,E      ;SHIFT LOW ORDER RIGHT
551 000370# 001000 000037          07960   HAR
552 000371# 001000 000157          07980   MOV    E,A
553 000372# 001000 000000          08000   MOV    A,B      ;SHIFT OVERFLOW BYTE RIGHT
554 000373# 001000 000037          08020   HAR
555 000374# 001000 000107          08040   MOV    B,>
556 000375# 001000 0000303         08060   JMP    SHFTR3   ;SEE IF WE ARE DONE
557 000376# 000000 000356#        08080
558 000377# 000000 000351#        08090

559

560
561           08120   IFE    LENGTH,<
562           08140   ;SHIFT C,U,E,B RIGHT ONE
563           08160   ;THIS IS USED BY SHIFTR, FMULT, FADD
564           08180   ;ALTERS A,B,C,D,E
565           08200   SHFTR1: MOV    A,C      ;GET THE MO
566           08220   SHFTR1A: MOV    A,R      ;SHIFT IT RIGHT, ENTRY FROM FMULT
567           08240   MOV    C,A
568           08260   MOV    A,D      ;SHIFT THE MO RIGHT
569           08280   HAR
570           08300   MOV    D,A
571           08320   MOV    A,E      ;SHIFT THE LO
572           08340   HAR
573           08360   MOV    E,A
574           08380   MOV    A,B      ;SHIFT THE EXTRA LO BYTE
575           08400   HAR
576           08420   MOV    B,A
577           08440   RET>   ;ALL DONE

```

```

    579          08480  SUBTL  NATURAL LOG FUNCTION
    580          08500  IFN   EXTFNC,<
    581          08520  ;CALCULATION IS BY:
    582          08540  ; LN(F*2^N)=(N*LOG2(F))*LN(2)
    583          08560  ;AN APPROXIMATION POLYNOMIAL IS USED TO CALCULATE LOG2(F)

    585          08600  ;CONSTANTS USED BY LOG
    586          08620  FONE: 000  ;1
    587          08640  000
    588          08660  000
    589          08680  201
    590          08700  LOGCN21 3  ;DEGREE+1
    591          08720  252  ;0.59878650
    592          08740  126
    593          08760  031
    594          08780  200
    595          08800  361  ;0.961470632
    596          08820  042
    597          08840  166
    598          08860  200
    599          08880  105  ;2.88539129
    600          08900  252  ;NOTE: THE REFERENCE FOR THIS CONSTANT HAS 100 NOT 105
    601          08920  070  ;IN THE LOW ORDER BYTE,
    602          08940  202

    603          08960  ;CHECK FOR A NEGATIVE OR ZERO ARGUMENT
    604          08980  LOG: FSIGN
    605          09000  JPE  FCERR  ;CHECK ONLY RETURNS 0,1 OR 377 IN A
    606          09020  ;THE PARITY WILL BE EVEN IF A HAS 0 OR 377
    607          09040  000000*  ;GET POINTER TO EXPONENT
    608          09060
    609          09080  ;FSIGN ONLY RETURNS 0,1 OR 377 IN A
    610          09100  LXI M,FAC  ;GET POINTER TO EXPONENT
    611          09120  000041
    612          09140  000235*
    613          09160  000430* 001000 000176
    614          09180  000601
    615          09200  00065  ;GET EXPONENT IN A
    616          09220  000800
    617          09240  000821
    618          09260  0008363
    619          09280  000864
    620          09300  000871* 001000 000020
    621          09320  0008365  ;SUB B
    622          09340  000840* 001000 000160  ;PUSH PSW
    623          09360  000842* 001000 000325  ;MOV M,B
    624          09380  000844* 001000 000329  ;SET EXP TO 200, RESULT IS NUM IN (.5,1)
    625          09400  000845* 001000 000315  ;SAVE SQR(.5)
    626          09420  0008445* 000000 000025*  ;CALCULATE (F=SQR(.5))/(F=SQR(.5))
    627          09440  0008426*
    628          09460  0008447* 001000 000301
    629          09480  0008450* 001000 000321
    630          09500  0008451* 001000 000004
    631          09520  0008452* 001000 000315  ;GET SQR(.5) BACK
    632          09540  0008453* 000000 000025*  ;WHERE F=NUMBER LEFT IN FAC
    633          09560  0008454* 000000 000445*
    634          09580  0008455* 001000 000001
    635          09600  0008456* 002000 000408*
    636          09620  0008457* 001000 000001
    637          09640  0008460* 001000 000315
    638          09660  0008461* 000000 000011*
    639          09680  0008462* 002000 000456*
    640          09700  0008463* 001000 000001
    641          09720  0008464* 002000 000404*
    642          09740  0008465* 002000 000461*
    643          09760  0008466* 001000 000315
    644          09780  0008467* 002000 000615*
    645          09800  0008468* 002000 000654*
    646          09820  0008471* 001000 000001
    647          09840  0008472* 002000 000200
    648          09860  0008473* 002000 000200
    649          09880  0008474* 001000 000021
    650          09900  0008475* 002000 000000
    651          09920  0008477* 001000 000000
    652          09940  0008500* 000000 000000
    653          09960  0008500* 000000 000000*
    654          09980  0008501* 002000 000000
    655          09990  0008502* 001000 000361
    656          09990  0008503* 001000 000315
    657          09990  0008504* 000000 001731*
    658          09990  0008505* 000000 000500*
    659          09990  0008506* 001000 000001
    660          09990  0008507* 001000 000000
    661          09990  0008510* 000000 000000
    662          09990  0008511* 001000 000021
    663          09990  0008512* 000000 000030
    664          09990  0008513* 000000 000162
    665          09990  0008514* 000000 000000
    666          09990  0008515* 000000 000000
    667          09990  0008516* 000000 000000
    668          09990  0008517* 000000 000000
    669          09990  0008518* 000000 000000
    670          09990  0008519* 000000 000000
    671          09990  0008520* 000000 000000
    672          09990  0008521* 000000 000000
    673          09990  0008522* 000000 000000
    674          09990  0008523* 000000 000000
    675          09990  0008524* 000000 000000
    676          09990  0008525* 000000 000000
    677          09990  0008526* 000000 000000
    678          09990  0008527* 000000 000000
    679          09990  0008528* 000000 000000
    680          09990  0008529* 000000 000000
    681          09990  0008530* 000000 000000
    682          09990  0008531* 000000 000000
    683          09990  0008532* 000000 000000
    684          09990  0008533* 000000 000000
    685          09990  0008534* 000000 000000
    686          09990  0008535* 000000 000000
    687          09990  0008536* 000000 000000
    688          09990  0008537* 000000 000000
    689          09990  0008538* 000000 000000
    690          09990  0008539* 000000 000000
    691          09990  0008540* 000000 000000
    692          09990  0008541* 000000 000000
    693          09990  0008542* 000000 000000
    694          09990  0008543* 000000 000000
    695          09990  0008544* 000000 000000
    696          09990  0008545* 000000 000000
    697          09990  0008546* 000000 000000
    698          09990  0008547* 000000 000000
    699          09990  0008548* 000000 000000
    700          09990  0008549* 000000 000000
    701          09990  0008550* 000000 000000
    702          09990  0008551* 000000 000000
    703          09990  0008552* 000000 000000
    704          09990  0008553* 000000 000000
    705          09990  0008554* 000000 000000
    706          09990  0008555* 000000 000000
    707          09990  0008556* 000000 000000
    708          09990  0008557* 000000 000000
    709          09990  0008558* 000000 000000
    710          09990  0008559* 000000 000000
    711          09990  0008560* 000000 000000
    712          09990  0008561* 000000 000000
    713          09990  0008562* 000000 000000
    714          09990  0008563* 000000 000000
    715          09990  0008564* 000000 000000
    716          09990  0008565* 000000 000000
    717          09990  0008566* 000000 000000
    718          09990  0008567* 000000 000000
    719          09990  0008568* 000000 000000
    720          09990  0008569* 000000 000000
    721          09990  0008570* 000000 000000
    722          09990  0008571* 000000 000000
    723          09990  0008572* 000000 000000
    724          09990  0008573* 000000 000000
    725          09990  0008574* 000000 000000
    726          09990  0008575* 000000 000000
    727          09990  0008576* 000000 000000
    728          09990  0008577* 000000 000000
    729          09990  0008578* 000000 000000
    730          09990  0008579* 000000 000000
    731          09990  0008580* 000000 000000
    732          09990  0008581* 000000 000000
    733          09990  0008582* 000000 000000
    734          09990  0008583* 000000 000000
    735          09990  0008584* 000000 000000
    736          09990  0008585* 000000 000000
    737          09990  0008586* 000000 000000
    738          09990  0008587* 000000 000000
    739          09990  0008588* 000000 000000
    740          09990  0008589* 000000 000000
    741          09990  0008590* 000000 000000
    742          09990  0008591* 000000 000000
    743          09990  0008592* 000000 000000
    744          09990  0008593* 000000 000000
    745          09990  0008594* 000000 000000
    746          09990  0008595* 000000 000000
    747          09990  0008596* 000000 000000
    748          09990  0008597* 000000 000000
    749          09990  0008598* 000000 000000
    750          09990  0008599* 000000 000000
    751          09990  0008600* 000000 000000
    752          09990  0008601* 000000 000000
    753          09990  0008602* 000000 000000
    754          09990  0008603* 000000 000000
    755          09990  0008604* 000000 000000
    756          09990  0008605* 000000 000000
    757          09990  0008606* 000000 000000
    758          09990  0008607* 000000 000000
    759          09990  0008608* 000000 000000
    760          09990  0008609* 000000 000000
    761          09990  0008610* 000000 000000
    762          09990  0008611* 000000 000000
    763          09990  0008612* 000000 000000
    764          09990  0008613* 000000 000000
    765          09990  0008614* 000000 000000
    766          09990  0008615* 000000 000000
    767          09990  0008616* 000000 000000
    768          09990  0008617* 000000 000000
    769          09990  0008618* 000000 000000
    770          09990  0008619* 000000 000000
    771          09990  0008620* 000000 000000
    772          09990  0008621* 000000 000000
    773          09990  0008622* 000000 000000
    774          09990  0008623* 000000 000000
    775          09990  0008624* 000000 000000
    776          09990  0008625* 000000 000000
    777          09990  0008626* 000000 000000
    778          09990  0008627* 000000 000000
    779          09990  0008628* 000000 000000
    780          09990  0008629* 000000 000000
    781          09990  0008630* 000000 000000
    782          09990  0008631* 000000 000000
    783          09990  0008632* 000000 000000
    784          09990  0008633* 000000 000000
    785          09990  0008634* 000000 000000
    786          09990  0008635* 000000 000000
    787          09990  0008636* 000000 000000
    788          09990  0008637* 000000 000000
    789          09990  0008638* 000000 000000
    790          09990  0008639* 000000 000000
    791          09990  0008640* 000000 000000
    792          09990  0008641* 000000 000000
    793          09990  0008642* 000000 000000
    794          09990  0008643* 000000 000000
    795          09990  0008644* 000000 000000
    796          09990  0008645* 000000 000000
    797          09990  0008646* 000000 000000
    798          09990  0008647* 000000 000000
    799          09990  0008648* 000000 000000
    800          09990  0008649* 000000 000000
    801          09990  0008650* 000000 000000
    802          09990  0008651* 000000 000000
    803          09990  0008652* 000000 000000
    804          09990  0008653* 000000 000000
    805          09990  0008654* 000000 000000
    806          09990  0008655* 000000 000000
    807          09990  0008656* 000000 000000
    808          09990  0008657* 000000 000000
    809          09990  0008658* 000000 000000
    810          09990  0008659* 000000 000000
    811          09990  0008660* 000000 000000
    812          09990  0008661* 000000 000000
    813          09990  0008662* 000000 000000
    814          09990  0008663* 000000 000000
    815          09990  0008664* 000000 000000
    816          09990  0008665* 000000 000000
    817          09990  0008666* 000000 000000
    818          09990  0008667* 000000 000000
    819          09990  0008668* 000000 000000
    820          09990  0008669* 000000 000000
    821          09990  0008670* 000000 000000
    822          09990  0008671* 000000 000000
    823          09990  0008672* 000000 000000
    824          09990  0008673* 000000 000000
    825          09990  0008674* 000000 000000
    826          09990  0008675* 000000 000000
    827          09990  0008676* 000000 000000
    828          09990  0008677* 000000 000000
    829          09990  0008678* 000000 000000
    830          09990  0008679* 000000 000000
    831          09990  0008680* 000000 000000
    832          09990  0008681* 000000 000000
    833          09990  0008682* 000000 000000
    834          09990  0008683* 000000 000000
    835          09990  0008684* 000000 000000
    836          09990  0008685* 000000 000000
    837          09990  0008686* 000000 000000
    838          09990  0008687* 000000 000000
    839          09990  0008688* 000000 000000
    840          09990  0008689* 000000 000000
    841          09990  0008690* 000000 000000
    842          09990  0008691* 000000 000000
    843          09990  0008692* 000000 000000
    844          09990  0008693* 000000 000000
    845          09990  0008694* 000000 000000
    846          09990  0008695* 000000 000000
    847          09990  0008696* 000000 000000
    848          09990  0008697* 000000 000000
    849          09990  0008698* 000000 000000
    850          09990  0008699* 000000 000000
    851          09990  0008700* 000000 000000
    852          09990  0008701* 000000 000000
    853          09990  0008702* 000000 000000
    854          09990  0008703* 000000 000000
    855          09990  0008704* 000000 000000
    856          09990  0008705* 000000 000000
    857          09990  0008706* 000000 000000
    858          09990  0008707* 000000 000000
    859          09990  0008708* 000000 000000
    860          09990  0008709* 000000 000000
    861          09990  0008710* 000000 000000
    862          09990  0008711* 000000 000000
    863          09990  0008712* 000000 000000
    864          09990  0008713* 000000 000000
    865          09990  0008714* 000000 000000
    866          09990  0008715* 000000 000000
    867          09990  0008716* 000000 000000
    868          09990  0008717* 000000 000000
    869          09990  0008718* 000000 000000
    870          09990  0008719* 000000 000000
    871          09990  0008720* 000000 000000
    872          09990  0008721* 000000 000000
    873          09990  0008722* 000000 000000
    874          09990  0008723* 000000 000000
    875          09990  0008724* 000000 000000
    876          09990  0008725* 000000 000000
    877          09990  0008726* 000000 000000
    878          09990  0008727* 000000 000000
    879          09990  0008728* 000000 000000
    880          09990  0008729* 000000 000000
    881          09990  0008730* 000000 000000
    882          09990  0008731* 000000 000000
    883          09990  0008732* 000000 000000
    884          09990  0008733* 000000 000000
    885          09990  0008734* 000000 000000
    886          09990  0008735* 000000 000000
    887          09990  0008736* 000000 000000
    888          09990  0008737* 000000 000000
    889          09990  0008738* 000000 000000
    890          09990  0008739* 000000 000000
    891          09990  0008740* 000000 000000
    892          09990  0008741* 000000 000000
    893          09990  0008742* 000000 000000
    894          09990  0008743* 000000 000000
    895          09990  0008744* 000000 000000
    896          09990  0008745* 000000 000000
    897          09990  0008746* 000000 000000
    898          09990  0008747* 000000 000000
    899          09990  0008748* 000000 000000
    900          09990  0008749* 000000 000000
    901          09990  0008750* 000000 000000
    902          09990  0008751* 000000 000000
    903          09990  0008752* 000000 000000
    904          09990  0008753* 000000 000000
    905          09990  0008754* 000000 000000
    906          09990  0008755* 000000 000000
    907          09990  0008756* 000000 000000
    908          09990  0008757* 000000 000000
    909          09990  0008758* 000000 000000
    910          09990  0008759* 000000 000000
    911          09990  0008760* 000000 000000
    912          09990  0008761* 000000 000000
    913          09990  0008762* 000000 000000
    914          09990  0008763* 000000 000000
    915          09990  0008764* 000000 000000
    916          09990  0008765* 000000 000000
    917          09990  0008766* 000000 000000
    918          09990  0008767* 000000 000000
    919          09990  0008768* 000000 000000
    920          09990  0008769* 000000 000000
    921          09990  0008770* 000000 000000
    922          09990  0008771* 000000 000000
    923          09990  0008772* 000000 000000
    924          09990  00087
```

```

    667          09500  SUBTLL  FLOATING MULTIPLICATION AND DIVISION
    668          09520  ;MULTIPLICATION      FAC=ARG*FAC
    669          09540  ;ALTERS A,B,C,D,E,M,L
    670          09560  IFE     EXTFCN,<
    671          09580  FMULTL CALL    HUVRM>           ;ENTRY WITH POINTER TO ARG IN (HL)
    672          09580  IFN     LENGTHH,<
    673          09620  XWD    1000,0041           ;"LXI H" AROUND NEXT 2 BYTES
    674          09640  FMULTL POPR>           ;ENTRY IF ARGUMENT IS ON THE STACK
    675          09660  FMULTL FSIGN             ;CHECK IF FAC IS ZERO
    676          09680  RZ     ,0                ;IF IT IS, RESULT IS ZERO
    677          09700  MVI    L,0              ;ADD THE TWO EXPONENTS, L IS A FLAG
    678          09720  CALL   MULDIV            ;FIX UP THE EXPONENTS
    679          09740  ;SAVE THE NUMBER IN THE REGISTERS SO WE CAN ADD IT FAST
    680          09760  MOV    A,C              ;GET HO
    681          09780  STA    FMULTB+1          ;STORE HO OF REGISTERS
    682          09800  XCHG   SHLD   FMULTB+1          ;STORE THE TWO LO'S OF THE REGISTERS
    683          09820  XCHG   SHLD   FMULTB+1          ;STORE THE TWO LO'S OF THE REGISTERS
    684          09840  LXI    B,SCODE            ;ZERO THE PRODUCT REGISTERS
    685          09860  MOV    D,B              ;GET HD
    686          09880  MOV    E,B              ;GET LD
    687          09900  LXI    H,NORMAL            ;PUT ADDRESS OF NORMAL, HERE WE FINISH UP,
    688          09920  PUSH   H                ;ON THE STACK
    689          09940  LXI    H,FMULT2            ;PUT FMULT2 ON THE STACK TWICE, SO AFTER
    690          09960  MOV    D,B              ;WE MULTIPLY BY THE LO BYTE, WE WILL
    691          09980  MOV    E,B              ;MULTIPLY BY THE HD AND LD
    692          09980  LXI    H,FACLO            ;GET ADDRESS OF LO OF FAC
    693          09980  XCHG   SHLD   FMULTB+1          ;WE MULTIPLYING BY ZERO?
    694          09980  PUSH   H                ;SAVE POINTER
    695          09980  LENGTH, <             ;SET UP A COUNT
    696          09980  IFN     LENGTHH,<
    697          09980  XCHG   SHLD   FMULTB+1          ;WE MULTIPLYING BY ZERO?
    698          09980  PUSH   H                ;SAVE POINTER
    699          09980  LENGTH, <             ;SET UP A COUNT
    700          09980  IFN     LENGTHH,<
    701          09980  XCHG   SHLD   FMULTB+1          ;WE MULTIPLYING BY ZERO?
    702          09980  PUSH   H                ;SAVE POINTER
    703          09980  LENGTH, <             ;SET UP A COUNT
    704          09980  IFN     LENGTHH,<
    705          09980  XCHG   SHLD   FMULTB+1          ;WE MULTIPLYING BY ZERO?
    706          09980  PUSH   H                ;SAVE POINTER
    707          09980  LENGTH, <             ;SET UP A COUNT
    708          09980  IFN     LENGTHH,<
    709          09980  XCHG   SHLD   FMULTB+1          ;WE MULTIPLYING BY ZERO?
    710          09980  PUSH   H                ;SAVE POINTER
    711          09980  LENGTH, <             ;SET UP A COUNT
    712          09980  IFN     LENGTHH,<
    713          09980  XCHG   SHLD   FMULTB+1          ;WE MULTIPLYING BY ZERO?
    714          09980  PUSH   H                ;SAVE POINTER
    715          09980  LENGTH, <             ;SET UP A COUNT
    716          09980  IFN     LENGTHH,<
    717          09980  XCHG   SHLD   FMULTB+1          ;WE MULTIPLYING BY ZERO?
    718          09980  PUSH   H                ;SAVE POINTER
    719          09980  LENGTH, <             ;SET UP A COUNT

```

```

    720          00056* 001000 000353  10200  XCHG   LENGTHH,<             ;GET LO'S IN (HL)
    721          00056/* 001000 000036  10220  MVI    E,10>           ;SET UP A COUNT
    722          00057* 000000 000010
    723
    724
    725          10200  ;THE PRODUCT WILL BE FORMED IN C,D,E,B, THIS WILL BE IN C,M,L,B PART OF THE
    726          10200  ;TIME IN ORDER TO USE THE "ROTATE" INSTRUCTION, AT FMULT2 WE GET THE NEXT
    727          10200  ;BYTE OF THE MANTISSA, WE THE FACT TO ROTATE IT, AND THEN POINT TO IT
    728          10200  ;THE FMULT2 SUBROUTINE PRESERVES (HL), IN 8K, IF THE BYTE IS ZERO, WE JUST
    729          10200  ;SHIFT THE PRODUCT 8 RIGHT, THIS BYTE IS THEN SHIFTED RIGHT AND SAVED IN D
    730          10200  ;IF WE DO, WE ADD IT TO C,M,L, B IS ONLY USED TO DETERMINE WHICH WAY WE
    731          10200  ;ROUND, WE THEN SHIFT C,M,L,B (C,D,E,B) IN 40 RIGHT ONE TO GET READY FOR THE
    732          10200  ;NEXT TIME THROUGH THE LOOP, NOTE THAT THE CARRY IS SHIFTED INTO THE MSB OF
    733          10200  ;D, IF D HAS A COUNT (L IN 4K) TO DETERMINE WHERE WE HAVE LOOKED AT ALL THE BITS
    734          10200  ;OF D (IN 4K)
    735          000571* 001000 000037  10240  FMULT2: RAR              ;ROTATE BYTE RIGHT
    736          10250  IFE    LENGTHH,<             ;SAVE THE COUNT
    737          10250  XCHG   LENGTHH,<             ;SAVE THE COUNT
    738          10250  IFN    LENGTHH,<             ;SAVE IT
    739          10250  XCHG   LENGTHH,<             ;SAVE IT
    740          10250  MVI    D,A>              ;SAVE IT
    741          10250  MVI    A,C              ;GET HO
    742          10250  JNC    FMULT5            ;DON'T ADD IN NUMBER IF BIT WAS ZERO
    743          000576* 000000 000053*
    744
    745          10260  IFE    LENGTHH,<
    746          10260  XCHG   LENGTHH,<             ;PUT THE LO'S IN (HL)
    747          10260  MVI    D,0              ;SAVE COUNTERS
    748          10260  FMULTB: LXI   D,SCODE            ;GET LO'S OF NUMBER TO ADD, THIS IS SET ABOVE
    749          00057* 000000 000057*
    750          00058* 001000 000051  10270  DAD    D              ;ADD THEM IN
    751          00058* 001000 000051  10270  POP    D              ;GET COUNTERS BACK
    752          00058* 001000 000051  10270  DCR    L              ;ADD IN HO, THIS IS SET UP ABOVE
    753          00058* 000000 000000
    754
    755          10270  IFE    LENGTHH,<
    756          10270  XCHG   LENGTHH,<             ;PUT THE LO'S BACK IN (DE)
    757          10270  FMULT5: CALL  SHFRDA            ;SHIFT THE RESULT RIGHT ONE
    758          10270  DCR    L              ;ARE WE DONE?
    759          10270  MVI    A,H>              ;GET NUMBER WE ARE MULTIPLYING BY
    760          00060* 001000 000037  10280  FMULT5: RAR              ;ROTATE RESULT RIGHT ONE
    761          00061* 001000 000117  10280  MVI    C,A              ;ROTATE NEXT LOWER ORDER
    762          00061* 001000 000174  10280  MVI    A,H>              ;ROTATE NEXT LOWER ORDER
    763          00061* 001000 000037  10290  RAR
    764          00061* 001000 000147  10290  MOV    H,A
    765          00061* 001000 000175  10290  MOV    A,L
    766          00061* 001000 000147  10290  RAR
    767          00061* 001000 000157  10290  MVI    L,A
    768          00061* 001000 000170  10290  MOV    A,B
    769          00062* 001000 000037  10290  RAR
    770          00062* 001000 000107  10290  MOV    B,A
    771          00062* 001000 000035  11100  DCR    E
    772          00062* 001000 000172  11100  MOV    A,D>           ;ARE WE DONE?
    773          00062* 001000 000172  11120  MVI    C,A>           ;GET NUMBER WE ARE MULTIPLYING BY

```

```

773 000624# 001000 000302      11140 JNZ FMULT4 ;MULTIPLY AGAIN IF WE ARE NOT DONE
774 000625# 000000 000571#      11160 IFN LENGTH,<
775 000626# 000000 000601#      11180 XCHG H ;GET LOFS IN (ML)
776 000627# 001000 000353      11200 POPRNT: PDR H ;GET POINTER TO NUMBER TO MULTIPLY BY
777 000628# 001000 000341      11220 RET ;TALL DONE
778 000629# 001000 000311      11240 IFN LENGTH,<
779 000630# 001000 000103      11260 FMULT3: MOV B,E ;MULTIPLY BY ZERO! SHIFT EVERYTHING 8 RIGHT
780 000631# 001000 000132      11280 MOV E,0
781 000632# 001000 000121      11300 MOV D,C
782 000633# 001000 000117      11320 MOV C,A ;SHIFT IN 8 ZEROS ON THE LEFT
783 000634# 001000 000121      11340 RET ;TALL DONE
784 000635# 001000 000117      11400 ;DIVIDE FAC BY 10
785 000636# 001000 000511      11420 TALTERS A,B,C,D,E,M,L ;ALTERS A,B,C,D,E,M,L
786 000637# 001000 000315      11440 DIV10: CALL PUSHMF ;SAVE NUMBER
787 000640# 000000 000625#      11460 IFN LENGTH=2,<
788 000641# 000000 000625#      11480 MOVR1=204,040,000,000 ;LOAD CONSTANT #10# INTO REGISTERS
789 000642# 001000 000801      11500 CALL MUVFR> ;MOVE THE CONSTANT TO THE FAC
790 000643# 000000 000040      11520 IFE LENGTH=2,<
791 000644# 002000 000234      11540 LXI H,FTEN ;GET POINTER TO THE CONSTANT #10#
792 000645# 001000 000021      11560 CALL MOVF#> ;MOVE TEN INTO THE FAC
793 000646# 001000 000200      11580 FDIVT: POPR ;GET NUMBER BACK IN REGISTERS
794 000647# 001000 000000      11600 ;TALL INTO DIVIDE AND WE ARE DONE
795 000648# 001000 000125#      11620 ;DIVISION FAC#ARG/FAC
796 000649# 000000 000040      11640 PDR H,FTEN ;GET NUMBER BACK IN REGISTERS
797 000650# 001000 000021      11660 CALL DV0ERR ;CHECK FOR DIVISION BY ZERO
798 000651# 001000 000200      11680 JZ DV0ERR ;HE IS TRYING TO GET AWAY WITH IT
799 000652# 001000 000000      11700 HVI L,377 ;SUBTRACT THE TWO EXPONENTS, L IS A FLAG
800 000653# 001000 000301      11720 CALL MULDIV ;FIX UP THE EXPONENTS AND THINGS
801 000654# 001000 000321      11740 INR M ;ADD 2 TO EXPONENT TO CORRECT SCALING
802 000655# 001000 000648#      11760 CALL MULDIV ;THERE WE SAVE THE FAC IN MEMORY SO WE CAN SUBTRACT IT FROM THE NUMBER
803 000656# 001000 000657#      11780 INR M ;IN THE REGISTERS QUICKLY.
804 000657# 001000 000658#      11800
805 000658# 001000 000659#      11820
806 000659# 001000 000660#      11840
807 000660# 001000 000661#      11860
808 000661# 001000 000662#      11880
809 000662# 001000 000663#      11900
810 000663# 001000 000664#      11920 DCX H ;POINT TO HO
811 000664# 001000 000665#      11940 MOV A,M ;GET HO
812 000665# 001000 000666#      11960 STA FUIV4+1 ;SAVE IT
813 000666# 001000 000667#      11980 DCX H ;SAVE MIDDLE ORDER
814 000667# 001000 000668#      12000 MOV A,M
815 000668# 001000 000669#      12020 STA FDIVB+1 ;PUT IT WHERE NOTHING WILL HURT IT
816 000669# 001000 000670#      12040
817 000670# 001000 000671#      12060 ;THE NUMERATOR WILL BE KEPT IN B,M,L. THE QUOTIENT WILL BE FORMED IN C,D,E.
818 000671# 001000 000672#      12080 ;TO GET A BIT OF THE QUOTIENT, WE FIRST SAVE B,M,L ON THE STACK, THEN
819 000672# 001000 000673#      12100 ;SUBTRACT THE DENOMINATOR THAT WE SAVED IN MEMORY. THE CARRY INDICATES
820 000673# 001000 000674#      12120 ;WHETHER OR NOT B,M,L WAS BIGGER THAN THE DENOMINATOR. IF B,M,L WAS BIGGER,
821 000674# 001000 000675#      12140 ;THE NEXT BIT OF THE QUOTIENT IS A ONE, TO GET THE OLD B,M,L OFF THE STACK,
822 000675# 001000 000676#      12160 ;POP THEM INTO THE PSW. IF THE DENOMINATOR WAS BIGGER, THE NEXT BIT OF
823 000676# 001000 000677#      12180 ;THE QUOTIENT IS A ZERO, AND WE GET THE OLD B,M,L BACK BY POPPING IT OFF THE
824 000677# 001000 000678#      12200 ;STACK. WE HAVE TO KEEP AN EXTRA BIT OF THE QUOTIENT IN FDIVG+1 IN CASE THE
825 000678# 001000 000679#      12220 ;DENOMINATOR WAS BIGGER, THEN B,M,L WILL GET SHIFTED LEFT. IF THE MSB OF
826 000679# 001000 000680#      12240 ;B WAS ONE, IT HAS TO BE STORED SOMEWHERE, SO WE STORE IT IN FDIVG+. THEN
827 000680# 001000 000681#      12260 ;THE NEXT TIME THROUGH THE LOOP B,M,L WILL LOOK BIGGER BECAUSE IT HAS AN
828 000681# 001000 000682#      12280 ;EXTRA HO BIT IN FDIVG+, WE ARE DONE DIVIDING WHEN THE MSB OF C IS A ONE,
829 000682# 001000 000683#      12300 ;THIS OCCURS WHEN WE HAVE CALCULATED 24 BITS OF THE QUOTIENT. WHEN WE JUMP
830 000683# 001000 000684#      12320 ;TO FDIVC+1, THE 25TH BIT OF THE QUOTIENT DETERMINES WHETHER WE ROUND OR NOT.
831 000684# 001000 000685#      12340 ;IT IS IN THE MSB OF C. IF IT IS A ONE, THE DENOMINATOR IS BIGGER THAN THE
832 000685# 001000 000686#      12360 ;NUMERATOR, THE FIRST BIT OF THE QUOTIENT WILL BE ZERO. THIS MEANS WE
833 000686# 001000 000687#      12380 ;WILL GO THROUGH THE DIVIDE LOOP 26 TIMES. SINCE IT STOPS ON THE 25TH BIT
834 000687# 001000 000688#      12400 ;AFTER THE FIRST NON-ZERO BIT OF THE EXPONENT. SO, THIS QUOTIENT WILL LOOK
835 000688# 001000 000689#      12420 ;SHIFTED LEFT ONE FROM THE QUOTIENT OF TWO NUMBERS IN WHICH THE NUMERATOR IS
836 000689# 001000 000690#      12440 ;BIGGER. THIS CAN ONLY OCCUR ON THE FIRST TIME THROUGH THE LOOP, SO C,D,E
837 000690# 001000 000691#      12460 ;ARE ALL ZERO. SO, IF WE FINISH THE LOOP AND C,D,E ARE ALL ZERO, THEN WE
838 000691# 001000 000692#      12480 ;MUST DECREMENT THE EXPONENT TO CORRECT FOR THIS,
839 000692# 001000 000693#      12500 ;B,C GET NUMBER IN B,M,L
840 000693# 001000 000694#      12520 XCHG A ;ZERO C,D,E AND HIGHEST ORDER
841 000694# 001000 000695#      12540 XRA A ;ZERO C,D,E AND HIGHEST ORDER
842 000695# 001000 000696#      12560 MOV C,A ;SUBTRACT NUMBER THAT WAS IN FAC
843 000696# 001000 000697#      12580 MOV D,A ;SUBTRACT NUMBER THAT WAS IN FAC
844 000697# 001000 000698#      12600 MOV E,A ;SUBTRACT NUMBER THAT WAS IN FAC
845 000698# 001000 000699#      12620 STA FDIVC+1 ;SUBTRACT NUMBER THAT WAS IN FAC
846 000699# 001000 000700#      12640 FDIV1: PUSH H ;SAVE LO'S OF NUMBER
847 000700# 001000 000701#      12660 PUSH B ;SAVE HO OF NUMBER
848 000701# 001000 000702#      12680 MOV A,L ;SUBTRACT NUMBER THAT WAS IN FAC
849 000702# 001000 000703#      12700 FDIVC: SUI B ;SUBTRACT LO
850 000703# 001000 000704#      12720 MOV L,A ;SAVE IT
851 000704# 001000 000705#      12740
852 000705# 001000 000706#      12760
853 000706# 001000 000707#      12780
854 000707# 001000 000708#      12800
855 000708# 001000 000709#      12820
856 000709# 001000 000710#      12840
857 000710# 001000 000711#      12860
858 000711# 001000 000712#      12880
859 000712# 001000 000713#      12900
860 000713# 001000 000714#      12920
861 000714# 001000 000715#      12940
862 000715# 001000 000716#      12960
863 000716# 001000 000717#      12980
864 000717# 001000 000718#      13000
865 000718# 001000 000719#      13020
866 000719# 001000 000720#      13040
867 000720# 001000 000721#      13060
868 000721# 001000 000722#      13080
869 000722# 001000 000723#      13100
870 000723# 001000 000724#      13120
871 000724# 001000 000725#      13140
872 000725# 001000 000726#      13160
873 000726# 001000 000727#      13180
874 000727# 001000 000728#      13200
875 000728# 001000 000729#      13220
876 000729# 001000 000730#      13240
877 000730# 001000 000731#      13260
878 000731# 001000 000732#      13280
879 000732# 001000 000733#      13300
880 000733# 001000 000734#      13320
881 000734# 001000 000735#      13340
882 000735# 001000 000736#      13360
883 000736# 001000 000737#      13380
884 000737# 001000 000738#      13400
885 000738# 001000 000739#      13420
886 000739# 001000 000740#      13440
887 000740# 001000 000741#      13460
888 000741# 001000 000742#      13480
889 000742# 001000 000743#      13500
890 000743# 001000 000744#      13520
891 000744# 001000 000745#      13540
892 000745# 001000 000746#      13560
893 000746# 001000 000747#      13580
894 000747# 001000 000748#      13600
895 000748# 001000 000749#      13620
896 000749# 001000 000750#      13640
897 000750# 001000 000751#      13660
898 000751# 001000 000752#      13680
899 000752# 001000 000753#      13700
900 000753# 001000 000754#      13720
901 000754# 001000 000755#      13740
902 000755# 001000 000756#      13760
903 000756# 001000 000757#      13780
904 000757# 001000 000758#      13800
905 000758# 001000 000759#      13820
906 000759# 001000 000760#      13840
907 000760# 001000 000761#      13860
908 000761# 001000 000762#      13880
909 000762# 001000 000763#      13900
910 000763# 001000 000764#      13920
911 000764# 001000 000765#      13940
912 000765# 001000 000766#      13960
913 000766# 001000 000767#      13980
914 000767# 001000 000768#      14000
915 000768# 001000 000769#      14020
916 000769# 001000 000770#      14040
917 000770# 001000 000771#      14060
918 000771# 001000 000772#      14080
919 000772# 001000 000773#      14100
920 000773# 001000 000774#      14120
921 000774# 001000 000775#      14140
922 000775# 001000 000776#      14160
923 000776# 001000 000777#      14180
924 000777# 001000 000778#      14200
925 000778# 001000 000779#      14220
926 000779# 001000 000780#      14240
927 000780# 001000 000781#      14260
928 000781# 001000 000782#      14280
929 000782# 001000 000783#      14300
930 000783# 001000 000784#      14320
931 000784# 001000 000785#      14340
932 000785# 001000 000786#      14360
933 000786# 001000 000787#      14380
934 000787# 001000 000788#      14400
935 000788# 001000 000789#      14420
936 000789# 001000 000790#      14440
937 000790# 001000 000791#      14460
938 000791# 001000 000792#      14480
939 000792# 001000 000793#      14500
940 000793# 001000 000794#      14520
941 000794# 001000 000795#      14540
942 000795# 001000 000796#      14560
943 000796# 001000 000797#      14580
944 000797# 001000 000798#      14600
945 000798# 001000 000799#      14620
946 000799# 001000 000800#      14640
947 000800# 001000 000801#      14660
948 000801# 001000 000802#      14680
949 000802# 001000 000803#      14700
950 000803# 001000 000804#      14720
951 000804# 001000 000805#      14740
952 000805# 001000 000806#      14760
953 000806# 001000 000807#      14780
954 000807# 001000 000808#      14800
955 000808# 001000 000809#      14820
956 000809# 001000 000810#      14840
957 000810# 001000 000811#      14860
958 000811# 001000 000812#      14880
959 000812# 001000 000813#      14900
960 000813# 001000 000814#      14920
961 000814# 001000 000815#      14940
962 000815# 001000 000816#      14960
963 000816# 001000 000817#      14980
964 000817# 001000 000818#      15000
965 000818# 001000 000819#      15020
966 000819# 001000 000820#      15040
967 000820# 001000 000821#      15060
968 000821# 001000 000822#      15080
969 000822# 001000 000823#      15100
970 000823# 001000 000824#      15120
971 000824# 001000 000825#      15140
972 000825# 001000 000826#      15160
973 000826# 001000 000827#      15180
974 000827# 001000 000828#      15200
975 000828# 001000 000829#      15220
976 000829# 001000 000830#      15240
977 000830# 001000 000831#      15260
978 000831# 001000 000832#      15280
979 000832# 001000 000833#      15300
980 000833# 001000 000834#      15320
981 000834# 001000 000835#      15340
982 000835# 001000 000836#      15360
983 000836# 001000 000837#      15380
984 000837# 001000 000838#      15400
985 000838# 001000 000839#      15420
986 000839# 001000 000840#      15440
987 000840# 001000 000841#      15460
988 000841# 001000 000842#      15480
989 000842# 001000 000843#      15500
990 000843# 001000 000844#      15520
991 000844# 001000 000845#      15540
992 000845# 001000 000846#      15560
993 000846# 001000 000847#      15580
994 000847# 001000 000848#      15600
995 000848# 001000 000849#      15620
996 000849# 001000 000850#      15640
997 000850# 001000 000851#      15660
998 000851# 001000 000852#      15680
999 000852# 001000 000853#      15700
1000 000853# 001000 000854#      15720
1001 000854# 001000 000855#      15740
1002 000855# 001000 000856#      15760
1003 000856# 001000 000857#      15780
1004 000857# 001000 000858#      15800
1005 000858# 001000 000859#      15820
1006 000859# 001000 000860#      15840
1007 000860# 001000 000861#      15860
1008 000861# 001000 000862#      15880
1009 000862# 001000 000863#      15900
1010 000863# 001000 000864#      15920
1011 000864# 001000 000865#      15940
1012 000865# 001000 000866#      15960
1013 000866# 001000 000867#      15980
1014 000867# 001000 000868#      16000
1015 000868# 001000 000869#      16020
1016 000869# 001000 000870#      16040
1017 000870# 001000 000871#      16060
1018 000871# 001000 000872#      16080
1019 000872# 001000 000873#      16100
1020 000873# 001000 000874#      16120
1021 000874# 001000 000875#      16140
1022 000875# 001000 000876#      16160
1023 000876# 001000 000877#      16180
1024 000877# 001000 000878#      16200
1025 000878# 001000 000879#      16220
1026 000879# 001000 000880#      16240
1027 000880# 001000 000881#      16260
1028 000881# 001000 000882#      16280
1029 000882# 001000 000883#      16300
1030 000883# 001000 000884#      16320
1031 000884# 001000 000885#      16340
1032 000885# 001000 000886#      16360
1033 000886# 001000 000887#      16380
1034 000887# 001000 000888#      16400
1035 000888# 001000 000889#      16420
1036 000889# 001000 000890#      16440
1037 000890# 001000 000891#      16460
1038 000891# 001000 000892#      16480
1039 000892# 001000 000893#      16500
1040 000893# 001000 000894#      16520
1041 000894# 001000 000895#      16540
1042 000895# 001000 000896#      16560
1043 000896# 001000 000897#      16580
1044 000897# 001000 000898#      16600
1045 000898# 001000 000899#      16620
1046 000899# 001000 000900#      16640
1047 000900# 001000 000901#      16660
1048 000901# 001000 000902#      16680
1049 000902# 001000 000903#      16700
1050 000903# 001000 000904#      16720
1051 000904# 001000 000905#      16740
1052 000905# 001000 000906#      16760
1053 000906# 001000 000907#      16780
1054 000907# 001000 000908#      16800
1055 000908# 001000 000909#      16820
1056 000909# 001000 000910#      16840
1057 000910# 001000 000911#      16860
1058 000911# 001000 000912#      16880
1059 000912# 001000 000913#      16900
1060 000913# 001000 000914#      16920
1061 000914# 001000 000915#      16940
1062 000915# 001000 000916#      16960
1063 000916# 001000 000917#      16980
1064 000917# 001000 000918#      17000
1065 000918# 001000 000919#      17020
1066 000919# 001000 000920#      17040
1067 000920# 001000 000921#      17060
1068 000921# 001000 000922#      17080
1069 000922# 001000 000923#      17100
1070 000923# 001000 000924#      17120
1071 000924# 001000 000925#      17140
1072 000925# 001000 000926#      17160
1073 000926# 001000 000927#      17180
1074 000927# 001000 000928#      17200
1075 000928# 001000 000929#      17220
1076 000929# 001000 000930#      17240
1077 000930# 001000 000931#      17260
1078 000931# 001000 000932#      17280
1079 000932# 001000 000933#      17300
1080 000933# 001000 000934#      17320
1081 000934# 001000 000935#      17340
1082 000935# 001000 000936#      17360
1083 000936# 001000 000937#      17380
1084 000937# 001000 000938#      17400
1085 000938# 001000 000939#      17420
1086 000939# 001000 000940#      17440
1087 000940# 001000 000941#      17460
1088 000941# 001000 000942#      17480
1089 000942# 001000 000943#      17500
1090 000943# 001000 000944#      17520
1091 000944# 001000 000945#      17540
1092 000945# 001000 000946#      17560
1093 000946# 001000 000947#      17580
1094 000947# 001000 000948#      17600
1095 000948# 001000 000949#      17620
1096 000949# 001000 000950#      17640
1097 000950# 001000 000951#      17660
1098 000951# 001000 000952#      17680
1099 000952# 001000 00095
```

MATHPK FOR BASIC MCS 8880 GATES/ALLEN/DAVIDOFF MACRO 47(113) 06109 27-AUG-75 PAGE 4-4

F4 MAC 23-AUG-64 06108  
FLOATING MULTIPLICATION AND DIVISION

879 000726\* 001000 000174 12740 MOV A,H ;SUBTRACT MIDDLE ORDER  
880 000727\* 001000 000356 12760 FDIVB: SBI 0  
881 000730\* 000000 000000 12800 MOV H,A ;SUBTRACT HD  
882 000731\* 001000 000147 12820 FDIVA: SBI 0 ;SUBTRACT HD  
883 000732\* 001000 000170 12840 MOV A,B ;GET HIGHEST ORDER  
884 000733\* 001000 000151 12860 MOV B,A ;GET OLD NUMBER BACK IF WE SUBTRACTED TOO MUCH  
885 000734\* 000000 000000 12880 FDIVG: MVI A,B ;GET MIDDLE ORDER  
886 000735\* 001000 000107 12900 CMC ;SET CARRY TO CORRESPOND TO NEXT QUOTIENT BIT  
887 000736\* 001000 000076 12920 JNC FUIV2 ;GET OLD NUMBER BACK IF WE SUBTRACTED TOO MUCH  
888 000737\* 000000 000000 12940 STA FUIVG+1 ;UPDATE HIGHEST ORDER  
889 000740\* 001000 000356 12960 SBI 0 ;SUBTRACT THE CARRY FROM IT  
890 000741\* 000000 000000 12970 PUP PSW ;THE SUBTRACTION WAS GOOD  
891 000742\* 001000 000351 12980 PUP PSW ;GET PREVIOUS NUMBER OFF STACK  
892 000743\* 001000 000351 13000 STC ;NEXT BIT IN QUOTIENT IS A ONE  
893 000744\* 000000 000755\* 13020 XHD 1000,322 ;JUMP AROUND NEXT 2 BYTES  
894 000745\* 000000 000716\* 13040 FUIV2: POP B ;THE SUBTRACTED TOO MUCH  
895 000746\* 001000 000662 13060 POP H ;GET OLD NUMBER BACK  
896 000747\* 000000 000737\* 13080 MOV A,C ;ARE WE DONE?  
897 000748\* 001000 000000 13100 INR A ;SET SIGN FLAG WITHOUT AFFECTING CARRY  
898 000749\* 001000 000352 13120 DCR A  
899 000750\* 000000 000000 13140 RAR  
900 000751\* 001000 000351 13160 JM ROUNDS ;PUT CARRY IN MSB  
901 000752\* 001000 000351 13180 HAL ;WE ARE DONE  
912 000753\* 001000 000000 13200 IFE LENGTH,< ;WE AREN'T, GET OLD CARRY BACK  
913 000754\* 001000 000000 13220 CALL SHFTL0> ;ROTATE EVERYTHING LEFT ONE  
914 000755\* 001000 000000 13240 IFN LENGTH,<  
915 000756\* 001000 000173 13260 MOV A,E ;ROTATE EVERYTHING LEFT ONE  
916 000757\* 001000 000027 13280 RAL ;ROTATE NEXT BIT OF QUOTIENT IN  
917 000758\* 001000 000137 13300 MOV E,A ;  
918 000759\* 001000 000172 13320 MOV A,D ;  
919 00075A\* 001000 000027 13340 HAL ;  
920 00075B\* 001000 000127 13360 MOV D,A ;  
921 00075C\* 001000 000027 13380 RAL ;  
922 00075D\* 001000 000027 13400 MOV C,A> ;  
923 00075E\* 001000 000051 13420 DAD H ;ROTATE A ZERO INTO RIGHT END OF NUMBER  
924 00075F\* 001000 000027 13440 MOV A,B ;THE HD BYTE, FINALLY!  
925 000760\* 001000 000170 13460 DAD H ;  
926 000761\* 001000 000027 13480 RAL ;  
927 001003\* 001000 000107 13500 MOV B,A ;  
928 001004\* 001000 000072 13520 LDA FUIVG+1 ;ROTATE THE HIGHEST ORDER  
929 001005\* 000000 000073\* 13540  
930 001006\* 000000 000076\*  
931 001007\* 001000 000027 13540 RAL

MATHPK FOR BASIC MCS 8880 GATES/ALLEN/DAVIDOFF MACRO 47(113) 06109 27-AUG-75 PAGE 4-5

F4 MAC 23-AUG-64 06108  
FLOATING MULTIPLICATION AND DIVISION

932 001010\* 001000 000062 13560 STA FUIVG+1  
933 001011\* 000000 000073\*  
934 001012\* 000000 000105\*  
935 001013\* 001000 000071 13580 MOV A,C ;ADD ONE TO EXPONENT IF THE FIRST SUBTRACTION  
936 001014\* 001000 000052 13600 ORA D ;I DID NOT WORK  
937 001015\* 001000 000263 13620 ORA E  
938 001016\* 001000 000302 13640 JNZ FUIV1 ;THIS ISN'T THE CASE  
939 001017\* 000000 000720\*  
940 001018\* 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 000042\*  
944 001024\* 000000 001017\*  
945 001025\* 001000 000065 13700 DCR H ;DECREMENT EXPONENT  
946 001026\* 001000 000341 13720 POP H ;GET NUMBER BACK  
947 001027\* 001000 000502 13740 JNZ FUIV1 ;DIVIDE MORE IF NO OVERFLOW OCCURED  
948 001030\* 000000 0000720\*  
949 001031\* 000000 001023\*  
950 001032\* 001000 000303 13760 JMP OVERR ;OVERFLOW!!  
951 001033\* 000000 000267\*  
952 001034\* 000000 001030\*  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984

13820 ;CHECK SPECIAL CASES AND ADD EXPONENTS FOR FMULT, FDIV  
13840 ;ALTERS A,B,H,L  
13860 IFE LENGTH=2,<  
13880 MULDSV: MVI A,377 ;ENTRY FROM DDIV, SUBTRACT EXPONENTS  
13900 XHD 1000,056 ;MVI L" AROUND NEXT BYTE  
13920 MULDOVA: XRA A ;ENTRY FROM DMULT, ADD EXPONENTS  
13940 XRA H ;GET NUMBER OF SIGN AND HD OF ARG  
13960 MOV C,M ;GET NUMBER OF SIGN AND HD OF ARG  
13980 INX H ;INCREMENT POINTER TO EXPONENT  
14000 MOV B,M ;GET EXPONENT FOR BELOW  
14020 MOV L,A> ;SAVE ADD OR SUBTRACT FLAG  
14040 MULDIV: MVI A,B ;IS NUMBER IN REGISTERS ZERO?  
14060 ORA A  
14080 JZ MULDV2 ;IT IS, ZERO FAC AND WE ARE DONE  
14100 MOV A,L ;GET ADD OR SUBTRACT FLAG  
14120 LXI H,FAC ;GET POINTER TO EXPONENT  
14140 XRA M ;GET EXPONENT  
14160 ADD B ;ADD TO REGISTER EXPONENT  
14180 MOV B,A ;SAVE IT  
14200 XRA B ;CHECK FOR OVERFLOW  
14220 XRA B ;OVERFLOW IF SIGN IS THE SAME AS CARRY  
14240 MOV A,B ;GET SUM  
14260 JP MULDV1 ;WE HAVE OVERFLOW!!  
14280 ADI 200 ;PUT EXPONENT IN EXCESS 200

```

    985 001060* 000000 000200
    986 001061* 001000 000167      14300  MOV M,A      JSAVE IT IN THE FAC
    987 001062* 001000 000312      14320  JZ PUPHRT   THE HAVE UNDEFLOW!! RETURN,
    988 001063* 000000 000300
    989 001064* 000000 000165*
    990 001065* 001000 000115      14340  CALL UNPACK  UNPACK THE ARGUMENTS
    991 001066* 000000 001270*
    992 001067* 000000 001963*
    993 001070* 001000 000167      14360  MOV M,A      JSAVE THE NEW SIGN
    994 001071* 001000 000053      14380  DCX H       POINT TO EXPONENT
    995 001072* 001000 0000311     14400  RET          FALL DONE, LEAVE H0 IN A
    996 001073* 001000 000057      14420  IFN EXTFCN,<
    997 001074* 001000 000057      14440  MLDVEX1 FSIGN  ENTRY FROM EXP, PICK UNDERFLOW IF NEGATIVE
    998 001075* 001000 000341      14460  CMA          PICK OVERFLOW IF POSITIVE
    999 001076* 001000 000267      14480  POP H>      DON'T SCREW UP STACK
    1000 001077* 001000 000341     14500  MULDIV1 ORA A   IS ERROR OVERFLOW OR UNDEFLOW?
    1001 001077* 001000 000341     14520  MULDIV2 POP H   GET OLD RETURN ADDRESS OFF STACK
    1002
    1003 001078* 001000 000000     14540  IFE LENGTH,<
    1004
    1005 001079* 001000 000000     14560  JM OVERR    OVERFLOW
    1006
    1007 001080* 001000 000000     14580  FUNDERFLOW FUNDERFLOW -- FALL INTO ZERO
    1008
    1009 001081* 001000 000000     14600
    1010 001082* 001000 000000     14640  ;ZERO FAC
    1011 001083* 001000 000000     14660  ;ALTERS A ONLY
    1012 001084* 001000 000000     14680  ;EXITS WITH A=0
    1013
    1014
    1015 001085* 001000 000000     14700  ZERO1 XRA A   ;ZERO A
    1016 001086* 001000 000000     14720  STA FAC    ;ZERO FAC
    1017 001087* 001000 000000     14740  RET>        FALL DONE
    1018
    1019
    1020
    1021
    1022
    1023
    1024
    1025
    1026 001106* 001000 000315     14760  ;MULTIPLY FAC BY 10
    1027 001107* 000000 001240*
    1028 001110* 000000 001104*
    1029 001111* 001000 0000170     14780  MOV A,B      ;GET EXPONENT
    1030 001112* 001000 0000157     14800  ORA A       ;RESULT IS ZERO IF ARG IS ZERO
    1031 001113* 001000 0000310     14820  RRD          ;IT IS
    1032 001114* 001000 0000306     14840  ADI 2       ;MULTIPLY BY 4 BY ADDING 2 TO EXPONENT
    1033 001115* 001000 0000022     14860  JC OVERR    ;OVERFLOW!!
    1034 001116* 001000 0000532
    1035 001117* 000000 000267*
    1036 001120* 000000 001107*
    1037 001121* 001000 000107     14880  MOV B,A      ;RESTORE EXPONENT
    1038
    1039 001122* 001000 000315     15080  CALL FADD   ;ADD IN ORIGINAL NUMBER TO GET 5 TIMES IT
    1040 001123* 000000 000025*
    1041 001124* 000000 001117*
    1042 001125* 001000 0000041     15100  LXI H,FAC   ;ADD 1 TO EXPONENT TO MULTIPLY NUMBER BY
    1043 001126* 000000 001124*
    1044 001127* 001000 0000123*
    1045 001128* 001000 000064     15120  INR M       ; 2 TO GET 10 TIMES ORIGINAL NUMBER
    1046 001129* 001000 000500     15140  RNZ          ;FALL DONE IF NO OVERFLOW
    1047 001130* 000000 000267*
    1048 001131* 001000 000000     15160  JMP OVERR    ;OVERFLOW!!
    1049
    15180 PAGE

```

```

    1049
    1050
    1051
    1052
    1053
    1054
    1055
    1056
    1057
    1058
    1059
    1060
    1061
    1062
    1063
    1064
    1065
    1066
    1067
    1068
    1069
    1070
    1071
    1072
    1073
    1074
    1075
    1076
    1077
    1078
    1079
    1080
    1081
    1082
    1083
    1084
    1085
    1086
    1087
    1088
    1089
    1090
    1091
    1092
    1093
    1094
    1095
    1096
    1097
    1098
    1099
    1100
    1101
    1102
    1103
    1104
    1105
    1106
    1107
    1108
    1109
    1110
    1111
    1112
    1113
    1114
    1115
    1116
    1117
    1118
    1119
    1120
    1121
    1122
    1123
    1124
    1125
    1126
    1127
    1128
    1129
    1130
    1131
    1132
    1133
    1134
    1135
    1136
    1137
    1138
    1139
    1140
    1141
    1142
    1143
    1144
    1145
    1146
    1147
    1148
    1149
    1150
    1151
    1152
    1153
    1154
    1155
    1156
    1157
    1158
    1159
    1160
    1161
    1162
    1163
    1164
    1165
    1166
    1167
    1168
    1169
    1170
    1171
    1172
    1173
    1174
    1175
    1176
    1177
    1178
    1179
    1180
    1181
    1182
    1183
    1184
    1185
    1186
    1187
    1188
    1189
    1190
    1191
    1192
    1193
    1194
    1195
    1196
    1197
    1198
    1199
    1200
    1201
    1202
    1203
    1204
    1205
    1206
    1207
    1208
    1209
    1210
    1211
    1212
    1213
    1214
    1215
    1216
    1217
    1218
    1219
    1220
    1221
    1222
    1223
    1224
    1225
    1226
    1227
    1228
    1229
    1230
    1231
    1232
    1233
    1234
    1235
    1236
    1237
    1238
    1239
    1240
    1241
    1242
    1243
    1244
    1245
    1246
    1247
    1248
    1249
    1250
    1251
    1252
    1253
    1254
    1255
    1256
    1257
    1258
    1259
    1260
    1261
    1262
    1263
    1264
    1265
    1266
    1267
    1268
    1269
    1270
    1271
    1272
    1273
    1274
    1275
    1276
    1277
    1278
    1279
    1280
    1281
    1282
    1283
    1284
    1285
    1286
    1287
    1288
    1289
    1290
    1291
    1292
    1293
    1294
    1295
    1296
    1297
    1298
    1299
    1300
    1301
    1302
    1303
    1304
    1305
    1306
    1307
    1308
    1309
    1310
    1311
    1312
    1313
    1314
    1315
    1316
    1317
    1318
    1319
    1320
    1321
    1322
    1323
    1324
    1325
    1326
    1327
    1328
    1329
    1330
    1331
    1332
    1333
    1334
    1335
    1336
    1337
    1338
    1339
    1340
    1341
    1342
    1343
    1344
    1345
    1346
    1347
    1348
    1349
    1350
    1351
    1352
    1353
    1354
    1355
    1356
    1357
    1358
    1359
    1360
    1361
    1362
    1363
    1364
    1365
    1366
    1367
    1368
    1369
    1370
    1371
    1372
    1373
    1374
    1375
    1376
    1377
    1378
    1379
    1380
    1381
    1382
    1383
    1384
    1385
    1386
    1387
    1388
    1389
    1390
    1391
    1392
    1393
    1394
    1395
    1396
    1397
    1398
    1399
    1400
    1401
    1402
    1403
    1404
    1405
    1406
    1407
    1408
    1409
    1410
    1411
    1412
    1413
    1414
    1415
    1416
    1417
    1418
    1419
    1420
    1421
    1422
    1423
    1424
    1425
    1426
    1427
    1428
    1429
    1430
    1431
    1432
    1433
    1434
    1435
    1436
    1437
    1438
    1439
    1440
    1441
    1442
    1443
    1444
    1445
    1446
    1447
    1448
    1449
    1450
    1451
    1452
    1453
    1454
    1455
    1456
    1457
    1458
    1459
    1460
    1461
    1462
    1463
    1464
    1465
    1466
    1467
    1468
    1469
    1470
    1471
    1472
    1473
    1474
    1475
    1476
    1477
    1478
    1479
    1480
    1481
    1482
    1483
    1484
    1485
    1486
    1487
    1488
    1489
    1490
    1491
    1492
    1493
    1494
    1495
    1496
    1497
    1498
    1499
    1500
    1501
    1502
    1503
    1504
    1505
    1506
    1507
    1508
    1509
    1510
    1511
    1512
    1513
    1514
    1515
    1516
    1517
    1518
    1519
    1520
    1521
    1522
    1523
    1524
    1525
    1526
    1527
    1528
    1529
    1530
    1531
    1532
    1533
    1534
    1535
    1536
    1537
    1538
    1539
    1540
    1541
    1542
    1543
    1544
    1545
    1546
    1547
    1548
    1549
    1550
    1551
    1552
    1553
    1554
    1555
    1556
    1557
    1558
    1559
    1560
    1561
    1562
    1563
    1564
    1565
    1566
    1567
    1568
    1569
    1570
    1571
    1572
    1573
    1574
    1575
    1576
    1577
    1578
    1579
    1580
    1581
    1582
    1583
    1584
    1585
    1586
    1587
    1588
    1589
    1590
    1591
    1592
    1593
    1594
    1595
    1596
    1597
    1598
    1599
    1600
    1601
    1602
    1603
    1604
    1605
    1606
    1607
    1608
    1609
    1610
    1611
    1612
    1613
    1614
    1615
    1616
    1617
    1618
    1619
    1620
    1621
    1622
    1623
    1624
    1625
    1626
    1627
    1628
    1629
    1630
    1631
    1632
    1633
    1634
    1635
    1636
    1637
    1638
    1639
    1640
    1641
    1642
    1643
    1644
    1645
    1646
    1647
    1648
    1649
    1650
    1651
    1652
    1653
    1654
    1655
    1656
    1657
    1658
    1659
    1660
    1661
    1662
    1663
    1664
    1665
    1666
    1667
    1668
    1669
    1670
    1671
    1672
    1673
    1674
    1675
    1676
    1677
    1678
    1679
    1680
    1681
    1682
    1683
    1684
    1685
    1686
    1687
    1688
    1689
    1690
    1691
    1692
    1693
    1694
    1695
    1696
    1697
    1698
    1699
    1700
    1701
    1702
    1703
    1704
    1705
    1706
    1707
    1708
    1709
    1710
    1711
    1712
    1713
    1714
    1715
    1716
    1717
    1718
    1719
    1720
    1721
    1722
    1723
    1724
    1725
    1726
    1727
    1728
    1729
    1730
    1731
    1732
    1733
    1734
    1735
    1736
    1737
    1738
    1739
    1740
    1741
    1742
    1743
    1744
    1745
    1746
    1747
    1748
    1749
    1750
    1751
    1752
    1753
    1754
    1755
    1756
    1757
    1758
    1759
    1760
    1761
    1762
    1763
    1764
    1765
    1766
    1767
    1768
    1769
    1770
    1771
    1772
    1773
    1774
    1775
    1776
    1777
    1778
    1779
    1780
    1781
    1782
    1783
    1784
    1785
    1786
    1787
    1788
    1789
    1790
    1791
    1792
    1793
    1794
    1795
    1796
    1797
    1798
    1799
    1800
    1801
    1802
    1803
    1804
    1805
    1806
    1807
    1808
    1809
    1810
    1811
    1812
    1813
    1814
    1815
    1816
    1817
    1818
    1819
    1820
    1821
    1822
    1823
    1824
    1825
    1826
    1827
    1828
    1829
    1830
    1831
    1832
    1833
    1834
    1835
    1836
    1837
    1838
    1839
    1840
    1841
    1842
    1843
    1844
    1845
    1846
    1847
    1848
    1849
    1850
    1851
    1852
    1853
    1854
    1855
    1856
    1857
    1858
    1859
    1860
    1861
    1862
    1863
    1864
    1865
    1866
    1867
    1868
    1869
    1870
    1871
    1872
    1873
    1874
    1875
    1876
    1877
    1878
    1879
    1880
    1881
    1882
    1883
    1884
    1885
    1886
    1887
    1888
    1889
    1890
    1891
    1892
    1893
    1894
    1895
    1896
    1897
    1898
    1899
    1900
    1901
    1902
    1903
    1904
    1905
    1906
    1907
    1908
    1909
    1910
    1911
    1912
    1913
    1914
    1915
    1916
    1917
    1918
    1919
    1920
    1921
    1922
    1923
    1924
    1925
    1926
    1927
    1928
    1929
    1930
    1931
    1932
    1933
    1934
    1935
    1936
    1937
    1938
    1939
    1940
    1941
    1942
    1943
    1944
    1945
    1946
    1947
    1948
    1949
    1950
    1951
    1952
    1953
    1954
    1955
    1956
    1957
    1958
    1959
    1960
    1961
    1962
    1963
    1964
    1965
    1966
    1967
    1968
    1969
    1970
    1971
    1972
    1973
    1974
    1975
    1976
    1977
    1978
    1979
    1980
    1981
    1982
    1983
    1984
    1985
    1986
    1987
    1988
    1989
    1990
    1991
    1992
    1993
    1994
    1995
    1996
    1997
    1998
    1999
    2000
    2001
    2002
    2003
    2004
    2005
    2006
    2007
    2008
    2009
    2010
    2011
    2012
    2013
    2014
    2015
    2016
    2017
    2018
    2019
    2020
    2021
    2022
    2023
    2024
    2025
    2026
    2027
    2028
    2029
    2030
    2031
    2032
    2033
    2034
    2035
    2036
    2037
    2038
    2039
    2040
    2041
    2042
    2043
    2044
    2045
    2046
    2047
    2048
    2049
    2050
    2051
    2052
    2053
    2054
    2055
    2056
    2057
    2058
    2059
    2060
    2061
    2062
    2063
    2064
    2065
    2066
    2067
    2068
    2069
    2070
    2071
    2072
    2073
    2074
    2075
    2076
    2077
    2078
    2079
    2080
    2081
    2082
    2083
    2084
    2085
    2086
    2087
    2088
    2089
    2090
    2091
    2092
    2093
    2094
    2095
    2096
    2097
    2098
    2099
    2100
    2101
    2102
    2103
    2104
    2105
    2106
    2107
    2108
    2109
    2110
    2111
    2112
    2113
    2114
    2115
    2116
    2117
    2118
    2119
    2120
    2121
    2122
    2123
    2124
    2125
    2126
    2127
    2128
    2129
    2130
    2131
    2132
    2133
    2134
    2135
    2136
    2137
    2138
    2139
    2140
    2141
    2142
    2143
    2144
    2145
    2146
    2147
    2148
    2149
    2150
    2151
    2152
    2153
    2154
    2155
    2156
    2157
    2158
    2159
    2160
    2161
    2162
    2163
    2164
    2165
    2166
    2167
    2168
    2169
    2170
    2171
    2172
    2173
    2174
    2175
    2176
    2177
    2178
    2179
    2180
    2181
    2182
    2183
    2184
    2185
    2186
    2187
    2188
    2189
    2190
    2191
    2192
    2193
    2194
    2195
    2196
    2197
    2198
    2199
    2200
    2201
    2202
    2203
    2204
    2205
    2206
    2207
    2208
    2209
    2210
    2211
    2212
    2213
    2214
    2215
    2216
    2217
    2218
    2219
    2220
    2221
    2222
    2223
    2224
    2225
    2226
    2227
    2228
    2229
    2230
    2231
    2232
    2233
    2234
    2235
    2236
    2237
    2238
    2239
    2240
    2241
    2242
    2243
    2244
    2245
    2246
    2247
    2248
    2249
    2250
    2251
    2252
    2253
    2254
    2255
    2256
    2257
    2258
    2259
    2260
    2261
    2262
    2263
    2264
    2265
    2266
    2267
    2268
    2269
    2270
    2271
    2272
    2273
    2274
    2275
    2276
    2277
    2278
    2279
    2280
    2281
    2282
    2283
    2284
    2285
    2286
    2287
    2288
    2289
    2290
    2291
    2292
    2293
    2294
    2295
    2296
    2297
    2298
    2299
    2300
    2301
    2302
    2303
    2304
    2305
    2306
    2307
    2308
    2309
    2310
    2311
    2312
    2313
    2314
    2315
    2316
    2317
    2318
    2319
    2320
    2321
    2322
    2323
    2324
    2325
    2326
    2327
    2328
    2329
    2330
    2331
    2332
    2333
    2334
    2335
    2336
    2337
    2338
    2339
    2340
    2341
    2342
    2343
    2344
    2345
    2346
    2347
    2348
    2349
    2350
    2351
    2352
    2353
    2354
    2355
    2356
    2357
    2358
    2359
    2360
    2361
    2362
    2363
    2364
    2365
    2366
    2367
    2368
    2369
    2370
    2371
    2372
    2373
    2374
    2375
    2376
    2377
    2378
    2379
    2380
    2381
    2382
    2383
    2384
    2385
    2386
    2387
    2388
    2389
    2390
    2391
    2392
    2393
    2394
    2395
    2396
    2397
    2398
    2399
    2400
    2401
    2402
    2403
    2404
    2405
    2406
    2407
    2408
    2409
    2410
    2411
    2412
    2413
    2414
    2415
    2416
    2417
    2418
    2419
    2420
    2421
    2422
    2423
    2424
    2425
    2426
    2427
    2428
    2429
    2430
    2431
    2432
    2433
    2434
    2435
    2436
    2437
    2438
    2439
    2440
    2441
    2442
    2443
    2444
    2445
    2446
    2447
    2448
    2449
    2450
    2451
    2452
    2453
    2454
    2455
    2456
    2457
    2458
    2459
    2460
    2461
    2462
    2463
    2464
    2465
    2466
    2467
    2468
    2469
    2470
    2471
    2472
    
```

MATHPK FOR BASIC MCS 8880 GATES/ALLEN/DAVIDOFF MACRO 47(113) 06109 27-AUG-75 PAGE 5

F4 MAC 23-AUG-64 06108 SIGN, SGN, FLOAT, NEG AND ABS

```

1050    15200 SUBTLL SIGN, SGN, FLOAT, NEG AND ABS
1051    15220 ;PUT SIGN OF FAC IN A
1052    15240 ;ALTERS A ONLY
1053    15260 ;LEAVES FAC ALONE
1054    15280 ;NOTE: TO TAKE ADVANTAGE OF THE RST INSTRUCTIONS TO SAVE BYTES,
1055    15300 ;IFSIGN IS DEFINED TO BE AN NST, "FSIGN" IS EQUIVALENT TO "CALL SIGN"
1056    15320 ;THE FIRST FEW INSTRUCTIONS SET SIGN (THE ONES BEFORE SIGNC) ARE DONE
1057    15340 ;PUT THE 8 BYTES AT THE RST LOCATION
1058    15360 REPEAT 0,< ;FSIGN IS ALWAYS AN RST
1059    15380 SIGN: LDA    FAC ;CHECK IF THE NUMBER IS ZERO
1060    15400 ORA    A
1061    15420 RZ> ;JIT IS, A IS ZERO
1062    15440 SIGNC: LUA   FAC-1 ;GET SIGN OF FAC, IT IS NON-ZERO
1063    001135* 001000 000072
1064    001135* 777777 777777*
1065    001140* 001000 000376
1066    001141* 001000 000457
1067    001142* 001000 000027
1068    001143* 001000 000237
1069    001144* 001000 000300
1070    001145* 001000 000074
1071    001146* 001000 000511
1072
1073
1074
1075
1076
1077 001147* 001000 000357
1078
1079
1080
1081
1082
1083 001150* 001000 000006
1084 001151* 000000 000210
1085 001152* 001000 000021
1086 001153* 000000 000601*
1087 001154* 000000 001136*
1088
1089
1090
1091
1092
1093 001155* 001000 000041
1094 001156* 000000 001126*
1095 001157* 000000 001153*
1096 001160* 001000 000117
1097 001161* 001000 000059
1098 001162* 001000 000006
1099 001163* 000000 000000
1100 001164* 001000 000043
1101 001165* 001000 000066
1102 001166* 000000 000200
15640 ;SGN FUNCTION
15660 ;ALTERS A,B,C,D,E,H,L
15680 IFN ;LENGTH=2,<
15700 SIGN: FSIGN> ;GET SIGN OF FAC IN A
15720 RET ;FALL INTO FLOAT
15780 ;FLOAT THE SIGNED INTEGER IN A
15800 ;ALTERS A,B,C,D,E,H,L
15820 FLOAT: MVI  B,218 ;SET EXPONENT CORRECTLY
15840 LXI D,SCODE ;ZERO D,E
15860 ;FALL INTO FLUATR
15920 ;FLOAT THE SIGNED NUMBER IN B,A,D,E
15940 ;ALTERS A,B,C,D,E,H,L
15960 FLUATR: LXI H,FAC ;GET POINTER TO FAC
15980 MOV C,A ;PUT HO IN C
16000 MOV H,B ;PUT EXPONENT IN THE FAC
16020 MVI B,B ;ZERO OVERFLOW BYTE
16040 INX H ;POINT TO SIGN
16060 MVI M,200 ;ASSUME A POSITIVE NUMBER

```

MATHPK FOR BASIC MCS 8880 GATES/ALLEN/DAVIDOFF MACRO 47(113) 06109 27-AUG-75 PAGE 5+1

F4 MAC 23-AUG-64 06108 SIGN, SGN, FLOAT, NEG AND ABS

```

1103 001167* 001000 000027
1104 001170* 001000 000303
1105 001171* 000000 000143*
1106 001172* 000000 001156*
1107
1108
1109
1110 001173* 16160 ;ABSOLUTE VALUE OF FAC
1111 001173* 16180 ;ALTERS A,,H,L
1112 16200 ABS: IFE LENGTH=2,<
1113 16240 CPI 2 ;IS THE ARGUMENT AN INTEGER?
1114 16260 JZ IABS> ;YES, USE THE INTEGER ABSOLUTE VALUE
1115 001173* 001000 000357
1116 001174* 001000 000360
1117 16320 RP ;GET THE SIGN OF FAC
1118
1119
1120
1121
1122
1123 001175* 001000 000041
1124 001175* 777777 777777*
1125 001177* 000000 001171*
1126 001200* 001000 000176
1127 001201* 001000 000356
1128 001202* 000000 000200
1129 001203* 001000 000167
1130 001204* 001000 000311
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
16380 ;NEGATE NUMBER IN THE FAC
16400 ;ALTERS A,,H,L
16420 ;NOTE: THE NUMBER MUST BE PACKED
16440 NEGI; LXI H,FAC-1 ;GET POINTER TO SIGN
16460 MOV A,M ;GET SIGN
16480 XRI 200 ;COMPLEMENT SIGN BIT
16500 MOV M,A ;SAVE IT
16520 RET ;FALL DONE
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 JHP NEG ;NEGATE SNG AND DBL THE SAME
16740
16750
16760
16780 ;SGN FUNCTION
16800 ;ALTERS A,,H,L
16820 SIGN: 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 HD
16880 SBB A
16900 MOV H,A
16920 JHP CUNISS ;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

```

MATHPK FOR BASIC MCS 8800 GATES/ALLEN/DAVIDOFF MACRO 47(113) 06109 27-AUG-75 PAGE 5\*2  
F4 MAC 23-AUG-64 06108

```
1156          17040  VSIGN: CPI 2           ;IS THE ARGUMENT AN INTEGER?  
1157          17050  JNZ SIGN           ;NO, SINGLE AND DOUBLE PREC. WORK THE SAME  
1158          17060  LHLD FACLO          ;GET THE INTEGER ARGUMENT  
1159          17100  MOV A,M             ;GET ITS SIGN  
1160          17120  ORA L               ;CHECK IF THE NUMBER IS ZERO  
1161          17140  RZ                ;IT IS, WE ARE DONE  
1162          17160  MOV A,M             ;IT ISN'T, SIGN IS THE SIGN OF H  
1163          17180  JMP ICOHPS>        ;GO SET A CORRECTLY  
1164          17200  PAGE
```

MATHPK FOR BASIC MCS 8800 GATES/ALLEN/DAVIDOFF MACRO 47(113) 06109 27-AUG-75 PAGE 6  
F4 MAC 23-AUG-64 06108

```
FLOATING POINT MOVEMENT ROUTINES  
1165          17220  SUBTTL FLOATING POINT MOVEMENT ROUTINES  
1166          17240  PUSHF FAC ON STACK  
1167          17260  FALTERS D,E  
1168  001205* 001000  000353  17280  PUSHF: XCHG             ;SAVE (HL)  
1169  001207* 001000  000052  17300  LHLD   FACLO          ;GET LO'S  
1170  001210* 000000  001176*  
1171  001210* 000000  001176*  
1172  001211* 001000  000343  17320  XTHL             ;SWITCH LO'S AND RET ADDR  
1173  001212* 001000  000345  17340  PUSH   H             ;PUT RET ADDR BACK ON STACK  
1174  001213* 001000  000052  17360  LHLD   FAC=1          ;GET HO'S  
1175  001214* 777777  777777*  
1176  001215* 000000  001207*  
1177  001215* 001000  000343  17380  XTHL             ;SWITCH HO'S AND RET ADDR  
1178  001217* 001000  000345  17400  PUSH   H             ;PUT RET ADDR BACK ON STACK  
1179  001220* 001000  000353  17420  XCHG             ;GET OLD (ML) BACK  
1180  001221* 001000  000311  17440  RET               ;ALL DONE  
1181  
1182  
1183          17500  ;MOVE NUMBER FROM MEMORY [(HL)] TO FAC  
1184          17520  ;FALTERS B,C,D,E,HL  
1185          17540  ;AT EXIT NUMBER IS IN B,C,D,E  
1186          17560  ;AT EXIT [(HL)]=(HL)+4  
1187  001222* 001000  000315  17580  MOVFM: CALL  MOVRN          ;GET NUMBER IN REGISTERS  
1188  001223* 000000  001243*  
1189  001224* 000000  001214*  
1190          17600  ;FALL INTO MOVRN AND PUT IT IN FAC  
1191  
1192  
1193          17660  ;MOVE REGISTERS (B,C,D,E) TO FAC  
1194          17680  ;FALTERS D,E  
1195  001225* 001000  000353  17700  MOVFR: XCHG             ;GET LO'S IN (HL)  
1196  001226* 001000  000042  17720  SHLD   FACLO          ;PUT THEM WHERE THEY BELONG  
1197  001227* 000000  001207*  
1198  001230* 000000  001225*  
1199  001231* 001000  000140  17740  MOV   H,B             ;GET HO'S IN (HL)  
1200  001232* 001000  000042  17760  MOV   L,C             ;PUT HO'S WHERE THEY BELONG  
1201  001233* 001000  000042  17780  SHLD   FAC=1          ;PUT HO'S WHERE THEY BELONG  
1202  001234* 777777  777777*  
1203  001235* 000000  001227*  
1204  001236* 001000  000353  17800  XCHG             ;GET OLD (ML) BACK  
1205  001237* 001000  000311  17820  RET               ;ALL DONE  
1206  
1207  
1208          17880  ;MOVE FAC TO REGISTERS (B,C,D,E)  
1209          17900  ;FALTERS B,C,D,E,HL  
1210  001240* 001000  000041  17920  MOVRF: LXI   H,FACLO          ;GET POINTER TO FAC  
1211  001241* 000000  001227*  
1212  001242* 000000  001234*  
1213          17940  ;FALL INTO MOVRN  
1214  
1215  
1216          18000  ;GET NUMBER IN REGISTERS (B,C,D,E) FROM MEMORY [(HL)]  
1217          18020  ;FALTERS B,C,D,E,HL
```

```

1218          180400    JAT EXIT (HL)=(HL)+4
1219  001243* 001000 000130 180500    MOVRM: MOV E,M      IGET LO
1220  001244* 001000 000043 180600    INX H      IPOINT TO MO
1221  001245* 001000 000125 181000    MOV D,M      IGET MO
1222  001246* 001000 000043 181200    INX H      IPOINT TO MU
1223  001247* 001000 000126 181400    MOV C,M      IGET MU
1224  001248* 001000 000043 181600    INX H      IPOINT TO EXPONENT
1225  001251* 001000 000106 181800    MOV B,M      IGET EXPONENT
1226  001252* 001000 000043 182000    INXHRT: INX H      TINC POINTER TO BEGINNING OF NEXT NUMBER
1227  001253* 001000 000311 182200    RET      FALL DONE
1228
1229
1230          182800    ?MOVE NUMBER FROM FAC TO MEMORY [(HL)]
1231          183000    ?ALTERS A,B,D,E,H,L
1232  001254* 001000 0008921 183200    MOVMF: LXI D,FACLO    IGET POINTER TO FAC
1233  001255* 000000 001241* 183400
1234  001256* 000000 001241*
1235
1236
1237
1238          184000    ?MOVE NUMBER FROM (DE) TO (HL)
1239          184200    ?ALTERS A,B,D,E,H,L
1240          184400    ?EXITS WITH (DE):=(DE)+4, (HL):=(HL)+4
1241  001257* 001000 000006 184600    MOVEI: MVI B,4      ISET COUNTER
1242  001260* 000000 000024
1243          184800    IFE LENGTH=2,<
1244          185000    XWD 1000,076    ?MVI A" OVER NEXT BYTE
1245          185200    MOVFM: XCHEP> 185400    ?MOVE NUMBERS INTO THE FAC
1246  001261* 001000 000032 185400    MOVEI: LDAX D      IGET WORD, ENTRY FROM VMOVFM
1247  001262* 001000 000033 185600    MOV M,A      IPUT IT WHERE IT BELONGS
1248  001263* 001000 000023 185800    INX D      INCREMENT POINTERS TO NEXT WORD
1249  001264* 001000 000043 186000    INX H      ISEE IF DONE
1250  001265* 001000 000005 186200    DCR B      ISEE IF DONE
1251  001266* 001000 000802 186400    JNZ MOVEI
1252  001267* 000000 001261*
1253  001270* 000000 001255*
1254  001271* 001000 000311 186600    RET
1255
1256
1257          187200    ?UNPACK THE FAC AND THE REGISTERS
1258          187400    ?ALTERS A,C,H,L
1259          187600    ?WHEN THE NUMBER IN THE FAC IS UNACKED, THE ASSUMED ONE IN THE
1260          187800    ?MANITISSA IS RESTORED, AND THE COMPLEMENT OF THE SIGN IS PLACED
1261          188000    IN FAC+1
1262  001272* 001000 000041 188200    UNPACK1: LXI H,FAC-1    IPOINT TO MO AND SIGN
1263  001273* 777777 777777* 188400    MOV A,M      IGET MO AND SIGN
1264  001274* 000000 0001267* 188600    PUSH PSW      ISAVE SIGN
1265  001275* 000000 000175 188800    ORI 200      RESTORE THE HIDDEN ONE
1266  001276* 001000 000365 189000    MOV M,A      ISAVE MO
1267  001277* 001000 000360 189200    POP PSW      IGET SIGN
1268  001300* 000000 000200
1269  001301* 001000 000167 189000    MOV M,A      ISAVE MO
1270  001302* 001000 000361 189200    POP PSW      IGET SIGN
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323

```

```

1271  001303* 001000 000256 189400    XRA M      IGET COMPLEMENT OF SIGN IN MSB
1272  001304* 001000 000043 189600    INX H      IPOINT TO TEMPORARY SIGN BYTE
1273  001305* 001000 000043 189800    INX H      IPOINT TO TEMPORARY SIGN BYTE
1274  001306* 001000 000167 190000    MOV M,A      ISAVE COMPLEMENT OF SIGN
1275  001307* 001000 000071 190200    MOV A,C      IGET MO AND SIGN OF THE REGISTERS
1276  001308* 001000 000345 190400    PUSH PSW      ISAVE SIGN
1277  001311* 001000 000366 190600    ORI 200      RESTORE THE HIDDEN ONE
1278  001312* 000000 000200
1279  001313* 001000 000211 190800    MOV C,A      ISAVE THE MO
1280  001314* 001000 000361 191000    POP PSW      IGET THE SIGN BACK
1281  001315* 001000 000256 191200    XRA M      ICOMPARE SIGN OF FAC AND SIGN OF REGISTERS
1282  001316* 001000 000311 191400    RET      IALL DONE
1283
1284
1285          192000    IFE LENGTH=2,<
1286  192200    REPEAT
1287          192400    ?PUT ANY TYPE VALUE ON THE STACK FROM THE FAC
1288          192600    ?STRINGS ARE TREATED AS INTEGERS
1289          192800    ?ALTERS A,B,C,H,L
1290          193000    VPUSHF: LDA VALTPY    IGET THE VALUE TYPE
1291          193200    CPI 4      ISET FLAGS ACCORDING TO VALTPY
1292          193400    LXI H,FACLO    IGET ADDRESS TO MO IN FAC
1293          193600    PUSH FACLO> 0000      PUSH FACLO> 0000 IN THE FAC
1294          193800    JM VPUSHM    IRETURN IF THE DATA WAS AN INTEGER OR A STRING
1295          194000    PUSHM      IPUSH FAC1,0 ON THE STACK
1296          194200    JZ VPUSHD    IRETURN IF WE HAD A SINGLE PRECISION NUMBER
1297          194400    LXI D,DFACLU    IWE HAVE A DOUBLE PRECISION NUMBER
1298          194600    PUSHM      IPUSH ITS 4 LO BYTES ON THE STACK
1299          194800    POPH      IPOP THE 4 LO BYTES FROM THE STACK
1300          195000    VPUSHD:> 195200    IALL DONE
1301
1302
1303          195600    ?MOVE ANY TYPE VALUE FROM MEMORY [(HL)] TO FAC
1304          195800    ?ALTERS A,B,D,E,H,L
1305          196000    VMOVFA: LXI H,ARGLO    JENTRY FROM FIN, DMUL10, DDIV10
1306          196200    VMOVFM: LXI D,MOVVFH    IGET ADDRESS OF LOCATION THAT DOES
1307          196400    JMP VMVFM      IAN "XCHEP" AND FALLS INTO MOVE1
1308
1309
1310          197000    ?MOVE ANY TYPE VALUE FROM FAC TO MEMORY [(HL)]
1311          197200    ?ALTERS A,B,D,E,H,L
1312          197400    VMOVAF: LXI H,ARGLO    JENTRY FROM FIN, DMUL10, DDIV10
1313          197600
1314          197800    VMOVFM: LXI D,MOVE1    IGET ADDRESS OF MOVE SUBROUTINE
1315          198000    VMVFM: PUSH D      IMOVE IT ON THE STACK
1316          198200    LXI D,DFACLU    IGET FIRST ADDRESS FOR INT, SNG
1317          198400    LD VALTPY    IGET THE VALUE TYPE
1318          198600    ANI 1      ISETTING LOCAL TYPE REALS
1319          198800    MOV B,A      ISET UP THE COUNT
1320          199000    CPI 10      JOO WE HAVE DBL?
1321          199200    RNZ      IWE DO NOT, GO DO THE MOVE
1322          199400    LXI D,DFACLO    IWE DO, GET LO ADDR OF THE DBL NUMBER
1323          199600    RET>
```

1324

19980 PAGE

1325 SUBTLL COMPARE TWO NUMBERS  
1326 ;COMPARE TWO SINGLE PRECISION NUMBERS  
1327 J#A1 IF ARG .LT. FAC  
1328 J#A0 IF ARG>FAC  
1329 J#A=1 IF ARG =GT. FAC  
1330 J#D0 IF DEPENDS UPON THE FACT THAT FCMP RETURNS WITH CARRY ON  
1331 J#C0 IF CARRY IS 377  
1332 JALTERS A,B,  
1333 001317\* 001000 000170 20060  
1334 001320\* 001000 000267 20020  
1335 001321\* 001000 000312 20180  
1336 001322\* 000000 000000\* 20200  
1337 001323\* 000000 001273\*  
1338 001324\* 000000 000441 20220  
1339 001325\* 000000 001141\*  
1340 001326\* 000000 001322\*  
1341 001327\* 001000 000345 20240  
1342 001330\* 001000 000357 20250  
1343 001331\* 001000 000171 20280  
1344 001332\* 001000 000310 20300  
1345 001333\* 001000 000041 20320  
1346 001334\* 001000 001171\*  
1347 001335\* 000000 001325\*  
1348 001336\* 001000 000256 20340  
1349 001337\* 001000 000171 20360  
1350 001340\* 001000 000376 20380  
1351 001341\* 001000 000315 20400  
1352 001342\* 000000 001347\*  
1353 001343\* 000000 001334\*  
1354 001344\* 001000 000037 20420  
1355 001345\* 001000 000251 20440  
1356 001346\* 001000 000311 20460  
1357  
1358 001347\* 001000 000043 20500  
1359 001350\* 001000 000170 20520  
1360 001351\* 001000 000276 20540  
1361 001352\* 001000 000300 20560  
1362 001353\* 001000 000053 20580  
1363 001354\* 001000 000171 20600  
1364 001355\* 001000 000276 20620  
1365 001356\* 001000 000300 20640  
1366 001357\* 001000 000053 20660  
1367 001360\* 001000 000172 20680  
1368 001361\* 001000 000276 20700  
1369 001362\* 001000 000300 20720  
1370 001363\* 001000 000053 20740  
1371 001364\* 001000 000173 20760  
1372 001365\* 001000 000276 20780  
1373 001366\* 001000 000300 20800  
1374 001367\* 001000 000341 20820  
1375 001370\* 001000 000341 20840  
1376 001371\* 001000 000511 20860  
1377  
SUBTLL COMPARE TWO NUMBERS  
J#A1 IF ARG .LT. FAC  
J#A0 IF ARG>FAC  
J#A=1 IF ARG =GT. FAC  
J#D0 IF DEPENDS UPON THE FACT THAT FCMP RETURNS WITH CARRY ON  
J#C0 IF CARRY IS 377  
JALTERS A,B,  
MOV A,B ;CHECK IF ARG IS ZERO  
ORA A  
JZ SIGN  
LXI H,FACMP ;THE JUMP TO FCOMPS WHEN WE ARE DONE  
PUSH H ;PUT THE ADDRESS ON THE STACK  
FSIGN ;CHECK IF FAC IS ZERO  
MOV A,C ;IF IT IS, RESULT IS MINUS THE SIGN OF ARG  
RZ ;IT IS  
LXI H,FAC=1 ;POINT TO SIGN OF FAC  
XRA M ;SEE IF THE SIGNS ARE THE SAME  
MOV A,C ;IF THEY ARE DIFFERENT, RESULT IS SIGN OF ARG  
RM ;THEY ARE DIFFERENT  
CALL FCOMP2 ;CHECK THE REST OF THE NUMBER  
RAR ;NUMBERS ARE DIFFERENT, CHANGE SIGN IF  
XRA C ;BOTH NUMBERS ARE NEGATIVE  
RET ;GO SET UP A  
INX H ;POINT TO EXPONENT OF ARG  
MOV A,B ;GET EXPONENT OF ARG  
CMP M ;COMPARE THE TWO  
RNZ ;NUMBERS ARE DIFFERENT  
DCX H ;POINT TO MD OF ARG  
GET NO OF ARG  
CMP M ;COMPARE WITH NO OF FAC  
RNZ ;THEY ARE DIFFERENT  
DCX H ;POINT TO NO OF ARG  
GET NO OF FAC  
CMP M ;COMPARE WITH MD OF FAC  
RNZ ;THE NUMBERS ARE DIFFERENT  
DCX H ;POINT TO LO OF FAC  
MOV A,E ;GET LO OF ARG  
SUB M ;SUBTRACT LO OF FAC  
RNZ ;NUMBERS ARE DIFFERENT  
POP H ;NUMBERS ARE THE SAME, DON'T SCREW UP STACK  
POP H  
RET ;ALL DONE

1378  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407  
1408  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1430

20920 IFE LENGTH=2,<  
20940 ;COMPARE TWO INTEGERS  
20960 JA=1 IF (UE)=LT, (HL)  
20980 JNE 0 IF ((UE)=(HL))  
21000 JA=1 IF (DE)=GT, (HL)  
21020 JALTERS A ONLY  
21040 ICOMP: MOV A,D ;ARE THE SIGNS THE SAME?  
21060 XRA H  
21080 MOV A,M ;IF NOT, ANSWER IS THE SIGN OF (HL)  
21100 JM ICOMPS ;THEY ARE DIFFERENT  
21120 CMP D ;THEY ARE THE SAME, COMPARE THE HOTS  
21140 JNZ SIGNS ;GO SET UP A  
21160 MOV A,L ;COMPARE THE LOTS  
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 JAE=0 ARG LT, FAC  
21320 JA=0 IF ARG>FAC  
21340 JA=1 IF ARG =GT, FAC  
21360 JALTERS A,B,C,D,E,H,L  
OCOMP0: LXI H,ARGLO ;ENTRY WITH POINTER TO ARG IN (UE)  
21380 MV1 B,10 ;SET UP COUNT TO MOVE DBL NUMBERS  
21400 CALL MOVE1 ;MOVE THE ARGUMENT INTO ARG  
21420 UCOMP: LXI D,ARG ;GET POINTER TO ARG  
21440 LDAX D  
21460 LDAX C,A ;SEE IF ARG#0  
21480 DCR D  
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  
FSIGN  
21560 ;SEE IF FAC#0  
21580 DCX D ;POINT TO SIGN OF ARG  
21600 LDAX D ;GET SIGN OF ARG  
21620 LDAX 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 ;JIF 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 MV1 B,10 ;SET UP A COUNT  
21800 LDAX D ;GET BYTES FROM ARG  
21820 SUB M ;COMPARE IT WITH ARG  
21840 JNZ FCOMPO ;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 OCOMP1 ;NO, COMPARE THE NEXT BYTES  
21940 POP B ;THEY ARE THE SAME, GET FCOMPS OFF STACK

1431 21960 RET>  
1432 21980 PAGE ;ALL DONE

```

1433          22000 SUBTLL CONVERSION ROUTINES BETWEEN INTEGER, SINGLE AND DOUBLE PRECISION
1434          22020 IFE LENGTH=2,<
1435          22040 ;FORCE THE FAC TO BE AN INTEGER
1436          22060 JALTERS A,B,C,D,E,M,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 RDH 0           ;WE HAVE AN INTEGER, ALL DONE
1441          22160 JM TMERR      ;WE HAVE A STRING, THAT IS A "NO-NOD"
1442          22180 CNZ GUNSD      ;IF WE HAVE A DBL, CONVERT IT TO A SNG
1443          22200 LXI H,OVERR    ;PUT OVER 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 JALTERS A,B,C,D,E,M,L
1450          22340 CONIS1: 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
1455          22440 CONIS1: PCH D      ;GET ERROR ADDRESS OFF STACK
1456          22460 JENTRY FIN, LIGHT
1457          22480 CONIS2: SHLD FACLO    ;STORE THE NUMBER IN FACLO
1458          22500 MVH A,2      ;SET VALTYP TO "INTEGER"
1459          22520 CONIS2: STA VALTYP    ;JENTRY FROM CONDS
1460          22540 RET
1461          22560 CONIS2: MOVRI 220,200,000,000 ;CHECK IF NUMBER IS -32768, ENTRY FROM FIN
1462          22580 CALL FCMP
1463          22600 RNZ           ;JERROR! IT CAN'T BE CONVERTED TO AN INTEGER
1464          22620 MOV H,C
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 JALTERS A,B,C,D,E,M,L
1471          22760 FRCNSG1: 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 CUNSI      ;WE HAVE AN INTEGER, CONVERT IT
1475          22840 JM TMERR      ;ISTRINGS!! -- ERROR!!
1476          22860 IDBL PREC -- FALL INTO CONSD
1477          22880
1478          22900
1479          22920 ;CONVERT DOUBLE PRECISION NUMBER TO A SINGLE PRECISION ONE
1480          22940 JALTERS A,B,C,D,E,M,L
1481          22960 CONSD1: CALL MOVR       ;GET THE HOTS IN THE REGISTERS
1482          22980 MVH A,4      ;SET VALTYP TO "SINGLE PRECISION"
1483          23000 STA VALTYP
1484          23020 MOV A,B
1485          23040 ORA A        ;CHECK IF THE NUMBER IS ZERO

```

```

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,H
1490          23140 JHP 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 JALTERS A,B,C,D,E,M,L
1495          23240 CUNSI: LHLD FACLO    ;GET THE INTEGER
1496          23260 CONSI: MVI A,4      ;SET VALTYP TO "SINGLE PRECISION"
1497          23280 STA VALTYP
1498          23300 MOV A,M
1499          23320 MOV D,M
1500          23340 MVI E,0
1501          23360 MVI B,220
1502          23380 JMP FLDATR    ;GO FLOAT THE NUMBER
1503          23400
1504          23420
1505          23440 ;FORCE THE FAC TO BE A DOUBLE PRECISION NUMBER
1506          23460 JALTERS A,B,C,D,E,M,L
1507          23480 FRCDBL: 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
1512          23580 CZ CUNSI      ;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 JALTERS A,H,M,L
1518          23700 CONDIS: LXI H,SCODE    ;ZERO H,M,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
1522          23780 JMP CONDIS>    ;GO TO IT
1523          23800 PAGE

```

```

1524          23620  SUBTLL GREATEST INTEGER FUNCTION
1525          23840  IQUICK GREATEST INTEGER FUNCTION
1526          23860  !LEAVES INT(FAC) IN C,D,E (SIGNED)
1527          23880  !ASSUMES FAC .C,.L, 2723 # 8388608
1528          23900  !ASSUMES THE EXPONENT OF FAC IS IN A
1529          23920  !ALTERS A,B,C,D,E
1530          23940  QINTF: MOV B,A      ;FZERO B,C,D,E IN CASE THE NUMBER IS ZERO
1531          001372* 001000 000107
1532          23960  MOV C,A
1533          001374* 001000 000117
1534          23980  MOV D,A
1535          001375* 001000 000127
1536          24000  MOV E,A
1537          001376* 001000 000267
1538          24020  ORA A      ;ISET CONDITION CODES
1539          001377* 001000 000310
1540          24040  RZ      ;JIT IS ZERO, WE ARE DONE

1541          24060  !THE HARD CASE IN QINT IS NEGATIVE NON-INTEGER, TU HANDLE THIS, IF THE
1542          24080  !NUMBER IS NEGATIVE, WE REGARD THE 3-BYTE MANTISSA AS A 3-BYTE INTEGER
1543          24100  !SUBTRACT ONE FROM THEM AND THE FRACTIONAL BITS SHIFTED LEFT BY SHIFTING THE
1544          24120  !MANTISSA RIGHT, THEN, IF THE NUMBER IS NEGATIVE, WE ADD ONE*. SO, WE
1545          24140  !HAD A NEGATIVE INTEGER, ALL THE BITS TO THE RIGHT OF THE BINARY POINT WERE
1546          24160  !ZERO, SO THE NET EFFECT IS WE HAVE THE ORIGINAL NUMBER IN C,D,E, IF THE
1547          24180  !NUMBER WAS A NEGATIVE NON-INTEGER, THERE IS AT LEAST ONE NON-ZERO BIT TO THE
1548          24200  !RIGHT OF THE BINARY POINT, SO THE NET EFFECT IS THAT WE GET THE ABSOLUTE
1549          24220  !VALUE OF INT(FAC) IN C,D,E, C,D,E IS THEN NEGATED IF THE ORIGINAL NUMBER WAS
1550          24240  !NEGATIVE SO THE RESULT WILL BE SIGNED,
1551          001408* 001000 000345
1552          24260  PUSH H      ;ISAVE (HL)
1553          001409* 001000 000315
1554          24280  CALL MOVRF  ;GET NUMBER IN THE REGISTERS
1555          001409* 001000 000345
1556          001411* 001000 000347
1557          001412* 000000 001436*
1558          001413* 000000 001425*
1559          001414* 001000 000076
1560          001415* 000000 000230
1561          001416* 001000 000029
1562          001417* 001000 000015
1563          001420* 001000 000334*
1564          001421* 000000 000334*
1565          001422* 001000 000174
1566          001423* 001000 000027
1567          001424* 001000 000334
1568          001425* 000000 000255*
1569          001426* 000000 001420*
1570          001427* 001000 000085
1571          001428* 001000 000089
1572          001431* 001000 000334
1573          001432* 000000 000310*
1574          001433* 000000 001425*
1575
1576          001434* 001000 000341
1577          24500  CALL UNPACK  ;UNPACK THE NUMBER
1578          24400  XRA M      ;GET SIGN OF NUMBER
1579          001437* 001000 000033
1580          24420  MOV M,A      ;DON'T LOSE IT
1581          001440* 001000 000243
1582          24440  CALL SHIFTR  ;SUBTRACT 1 FROM LD IF NUMBER IS NEGATIVE
1583          001441* 001000 000074
1584          24460  MVI A,230  ;SEE HOW MANY WE HAVE TO SHIFT TO CHANGE
1585          001442* 001000 000029
1586          24480  SUB B      ;NUMBER TO AN INTEGER
1587          001443* 001000 000015
1588          24500  CALL SHIFTB  ;SHIFT NUMBER TO GET RID OF FRACTIONAL BITS
1589          001444* 001000 000015
1590          24520  MVI B,0      ;FORGET THE BITS WE SHIFTED OUT
1591          001445* 001000 000089
1592          24540  CC NEGR  ;NEGATE NUMBER IF IT WAS NEGATIVE BECAUSE WE
1593          001446* 001000 000311
1594          24560  PUP H      ;WANT A SIGNED MANTISSA
1595          24580  POP H      ;GET OLD (HL) BACK

```

```

1577 001435* 001000 000311
1578
1579 001435* 001000 000033
1580 001437* 001000 000172
1581 001440* 001000 000243
1582 001441* 001000 000074
1583 001442* 001000 000300
1584 24600  IFN LENGTH=2,<   ;FALL DONE
1585 001443* 001000 000015
1586 24620  QINTA: DCX D      ;SUBTRACT ONE FROM C,D,E
1587 001447* 001000 000115*  MOV A,D      ;WE HAVE TO SUBTRACT ONE FROM C IF
1588 001448* 001000 0000172  ANA E      ;D AND E ARE BOTH ALL ONES
1589 001449* 001000 000074  INT A      ;SEE IF BOTH WERE -1
1590 001444* 001000 000300  NZR      ;THEY WERE NOT, WE ARE DONE
1591 24740  IFN LENGTH=2,<   ;THEY HERE, SUBTRACT ONE FROM C
1592 001445* 001000 000015
1593 24760  DCR C>
1594 001446* 001000 000115*  IFE LENGTH=2,<
1595 001447* 001000 000115*  24780  DCXBRT: DCX B>  ;THIS IS FOR BILL, C WILL NEVER BE ZERO
1596 001448* 001000 0000172  24800  ;(THE MSB WILL ALWAYS BE ONE) SO "DCX B"
1597 001449* 001000 0000172  24820  ;AND "DCR C" ARE FUNCTIONALLY EQUIVALENT
1598
1599 001444* 001000 000311
1600
1601 24840  RET      ;FALL DONE
1602
1603 001445* 001000 000041
1604 001446* 000000 001156*
1605 001447* 001000 000176
1606 001448* 001000 000076
1607 001449* 000000 000230
1608
1609 001445* 001000 000072
1610 001446* 000000 001255*
1611 001447* 000000 001446*
1612 001448* 001000 000520
1613
1614 001457* 001000 000176
1615 001458* 000000 000235
1616 001459* 001000 001372*
1617 001460* 000000 001454*
1618 001461* 001000 000066
1619 001462* 000000 000230
1620
1621 001463* 001000 000173
1622 001464* 001000 000365
1623 001465* 001000 000171
1624 001466* 001000 000227
1625 001467* 001000 000027
1626
1627 001468* 001000 000027
1628 001469* 001000 000027
1629 001471* 001000 000315
1630
1631 24920  SUBTLL GREATEST INTEGER FUNCTION
1632 001460  IALTERS A,B,C,D,E,M,L
1633 24940  IFE LENGTH=2,<
1634 001461  INTFC: CPI 4      ;SEE WHAT KIND OF NUMBER WE HAVE
1635 001462  RC 5
1636 001463  JNZ DINT      ;IT IS AN INTEGER, ALL DONE
1637 001464  CALL CUNIS>    ;CONVERT THE DOUBLE PRECISION NUMBER
1638 001465  25020  ;TRY TO CONVERT THE NUMBER TO AN INTEGER
1639 001466  25040  ;IF WE CAN'T, WE WILL RETURN HERE TO GIVE A
1640 001467  25060  ;SINGLE PRECISION RESULT
1641 001468  25080  ;GET EXPONENT
1642 001469  25100  INT: LXI H,FAC  ;SEE IF NUMBER HAS ANY FRACTIONAL BITS
1643 001470  25120  MOV A,M
1644 001471  25140  CPI 230
1645 001472  25160  IFN EXTFNC,<  ;THE ONLY GUY WHO NEEDS THIS DOESN'T CARE
1646 001473  LDA FACTD>    ;ABOUT THE SIGN
1647 001474  25180  ;AFTER NORMALIZATION
1648 001475  25200  RNC
1649 001476  25220  IFN EXTFNC,<
1650 001477  MOV A,M>
1651 001478  25240  PUSH PSW>  ;GET EXPONENT BACK
1652 001479  25260  CALL QINT  ;IT DOES, SHIFT THEM OUT
1653 001480  25280  RAL
1654 001481  25300  IFN EXTFNC,<
1655 001482  MOV A,E>
1656 001483  25320  PUSH PSW>  ;SAVE IT
1657 001484  25340  HLT      ;NEGATE NUMBER IF IT IS NEGATIVE
1658 001485  25360  RAL      ;PUT SIGN IN CARRY
1659 001486  25380  IFN EXTFNC,<
1660 001487  JMP FADFLT>  ;REFLOAT NUMBER
1661 001488  25400  IFN EXTFNC,<
1662 001489  CALL FADFLT  ;REFLOAT NUMBER

```

```

1630 001472' 000000 000143'  

1631 001473' 000000 001461'  

1632 001474' 001000 000361  

1633 001475' 001000 000311      25500 POPRTI: POP PSW      JGET LO BACK
1634                                         RET> JALL DONE

1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682      25500 IFE LENGTHM=2,  

1638                                         ;GREATEST INTEGER FUNCTION FOR DOUBLE PRECISION NUMBERS  

1639                                         ;ALTERS A,B,C,D,E,N,L  

1640                                         25620 DINT1: LXI H,FAC      JGET POINTER TO FAC  

1641                                         MOV A,M      JGET EXPONENT  

1642                                         CPI 220      JCAN WE CONVERT IT TO AN INTEGER?  

1643                                         JC FRCINT      JTENZ DO SO  

1644                                         JNZ DINT2      JCHECK FOR -32768  

1645                                         MOV C,A      JSAVE EXPONENT IN C  

1646                                         DCR H      JGET POINTERS TO SIGN AND HD  

1647                                         MOV A,M      JGET SIGN AND HD  

1648                                         XRI 203      JCHECK IF IT IS 200  

1649                                         MVI B,6      JSET UP A COUNT TO CHECK IF THE REST OF  

1650                                         DINT1: DCX H      ;THE NUMBER IS ZERO, POINT TO NEXT BYTE  

1651                                         ORA M      ;JIF ANY BITS ARE NON-ZERO, A WILL BE NON-ZERO  

1652                                         DCR B      JARE WE DONE?  

1653                                         JZ DINT1      JNO, CHECK THE NEXT LOWER ORDER BYTE  

1654                                         25980 LXI H,200*400+8CODE      JGET -32768 JUST IN CASE  

1655                                         JZ CUNISS      JA IS ZERO SO WE HAVE -32768  

1656                                         MOV A,C      JGET EXPONENT  

1657                                         26000 DINT2: CPI 270      JARE THERE ANY FRACTIONAL BITS?  

1658                                         RNC      JNO, THE NUMBER IS ALREADY AN INTEGER  

1659                                         26040 DINTFO: PUSH PSW      JENTRY FROM FOUT, CARRY IS ZERO IF WE COME  

1660                                         PSH PSW      JHERE FROM FOUT  

1661                                         CALL MOVRF      JGET THE LO OF NUMBER IN REGISTERS FOR UNPACKING  

1662                                         CALL UNPACK      JUNPACK IT  

1663                                         XRA M      JGET ITS SIGN BACK  

1664                                         DCX H      JSET THE EXPONENT TO NORMALIZE CORRECTLY  

1665                                         MVI M,270  

1666                                         PUSH PSW      JSAVE THE SIGN  

1667                                         CM DINTA      JSUBTRACT ONE FROM LO IF NUMBER IS NEGATIVE  

1668                                         MVI A,270      JGET HOW MANY BITS WE HAVE TO SHIFT OUT  

1669                                         SUB B  

1670                                         CALL DSHTFH      JSHIFT THEM OUT!!  

1671                                         POP PSW      JGET THE SIGN BACK  

1672                                         26300 CM DRUNA      JIF NUMBER WAS NEGATIVE, ADD ONE  

1673                                         XRA A      JPUT A ZERO IN THE EXTRA LO BYTE SO WHEN  

1674                                         STA DFACLO=1      WE NORMALIZE, WE WILL SHIFT IN ZEROS  

1675                                         POP PSW      JIF WE WERE CALLED FROM FOUT, DON'T NORMALIZE,  

1676                                         RNC      JJUST RETURN  

1677                                         JMP DRNRM      JREFLUTATE THE INTEGER  

1678
1679                                         26440 DINTA1: LXI H,DFACLO      JSUBTRACT ONE FROM FAC, GET POINTER TO LO  

1680                                         26440 DINTA1: MOV A,M      JGET A BYTE OF FAC  

1681                                         26480 DCR M      JSUBTRACT ONE FROM IT  

1682                                         26500 ORA A      JCONTINUE ONLY IF THE BYTE USED TO BE ZERO

```

```

1683
1684
1685
1686      26520 INCX H      JINCREMENT POINTER TO NEXT BYTE  

1684                                         JZ DINTA1      JCONTINUE IF NECESSARY  

1685                                         RET> JALL DONE  

1686                                         PAGE

```

```

1687          26600 SUBTL  INTEGER ARITHMETIC ROUTINES
1688          26620 IFN   MULTIM8<LENGTH=2>,<
1689          26640    ; TWO BYTE UNSIGNED INTEGER MULTIPLY
1690          26660    ; (HL)*(BC)*(DE)
1691          26680    ; A,D,E,H,L ARE CHANGED
1692          26700 DMULT:  LXI   H,SCODE  ;ZERO PRODUCT REGISTERS
1693          001475* 001000  000041
1694          001477* 000000  001153*
1695          001500* 000000  001472*
1696          001501* 001000  000170
1697          001502* 001000  000261
1698          001503* 001000  000310
1699          001504* 001000  000375
1700          001505* 000000  000020
1701          001506* 001000  000332
1702          001507* 000000  000000*
1703          001511* 000000  001477*
1704          001512* 001000  000353
1705          001513* 001000  000051
1706          001514* 001000  000353
1707          001515* 001000  000052
1708          001516* 000000  001524*
1709          001517* 002000  001516*
1710          001518* 001000  000011
1711          001519* 001000  000352
1712          001520* 000000  001516*
1713          001521* 000000  001516*
1714          001522* 001000  000275
1715          001523* 001000  000392
1716          001524* 000000  001506
1717          001527* 000000  001522*
1718          001530* 001000  000311
1719
1720
1721          27060 IFE   LENGTH=2,<
1722          27080 COMMENT X
1723          27100    ; INTEGER ARITHMETIC CONVENTIONS
1724
1725          27140 INTEGER VARIABLES ARE 2-BYTE, SIGNED NUMBERS
1726          27160 THE LO-BYTE COMES FIRST IN MEMORY
1727
1728          27200 CALLING CONVENTIONS:
1729          27220 FOR ONE ARGUMENT FUNCTIONS:
1730          27240    ; THE ARGUMENT IS IN (HL), THE RESULT IS LEFT IN (HL)
1731          27260 FOR TWO ARGUMENT FUNCTIONS:
1732          27280    ; THE FIRST ARGUMENT IS IN (DE)
1733          27300    ; THE SECOND ARGUMENT IS IN (HL)
1734          27320    ; THE RESULT IS LEFT IN (HL)
1735          27340 IF OVERFLOW OCCURS, THE ARGUMENTS ARE CONVERTED TO SINGLE PRECISION
1736          27360 WHEN INTEGERS ARE STORED IN THE FAC, THEY ARE STORED AT FACLO+0,1
1737          27380 VALTYPE(INTEGER)=2
1738          27400 X
1739          27420
    
```

```

1740          27440
1741          27460    ; INTEGER SUBTRACTION      (HL)=(DE)-(HL)
1742          27480    ; ALTERS A,B,C,D,E,H,L
1743          27500 ISUB:  MULD  A,M      ; EXTEND THE SIGN OF (HL) TO B
1744          27520 RAL   A,M      ; GET SIGN IN CARRY
1745          27540 SBB   A
1746          27560 MOV   B,A
1747          27580 CALL  INEGLH  ; NEGATE (HL)
1748          27600 MOV   A,C      ; GET A ZERO
1749          27620 SBB   B      ; NEGATE SIGN
1750          27640 JMP   IAUDS  ; GO ADD THE NUMBERS
1751
1752
1753          27660
1754          27680    ; INTEGER ADDITION        (HL)=(DE)+(HL)
1755          27700    ; ALTERS A,B,C,D,E,H,L
1756          27720 IAADD:  MULD  A,M      ; EXTEND THE SIGN OF (HL) TO B
1757          27740 RAL   A,M      ; GET SIGN IN CARRY
1758          27760 SBB   A
1759          27780 IAUDS:  MOV   B,A      ; SAVE THE SIGN
1760          27800 PUSH  M      ; SAVE THE SECOND ARGUMENT IN CASE OF OVERFLOW
1761          27820 MOV   A,D      ; EXTEND THE SIGN OF (DE) TO A
1762          27840 RAL   A,D      ; GET SIGN IN CARRY
1763          27860 SBB   A
1764          27880 DAU   D      ; ADD THE TWO LO'S
1765          27900 ADC   B      ; ADD THE EXTRA HO
1766          27920 RRC   H      ; IF THE LSB OF A IS DIFFERENT FROM THE MSB OF
1767          27940 XRA   H      ; H, THEN OVERFLOW OCCURRED
1768          27960 JP    PUPRT  ; FIND OVERFLOW, GET OLD (HL) OFF STACK AND WE
1769          28000    ; ARE DONE
1770          28020 PUSH  B      ; OVERFLOW -- SAVE EXTENDED SIGN OF (HL)
1771          28040 XCHG  H      ; GET (DE) IN (HL)
1772          28060 CALL  CONS1H  ; FLOAT IT
1773          28080 POP   PSW      ; GET SIGN OF (HL) IN A
1774          28100 POP   H      ; GET OLD (HL) BACK
1775          28120 CALL  PUSHF   ; PUT FIRST ARGUMENT ON STACK
1776          28140 XCHG  H      ; PUT SECOND ARGUMENT IN (DE) FOR FLOATR
1777          28160 CALL  INEGAD  ; FLOAT IT
1778          28180 POPH  H      ; GET FIRST ARGUMENT OFF STACK
1779          28200 FADD  H      ; ADD THE TWO NUMBERS USING SINGLE PRECISION
1780
1781          28240
1782          28260    ; INTEGER MULTIPLICATION   (HL)=(E)*(HL)
1783          28280    ; ALTERS A,B,C,D,E,H,L
1784          28300 IMULT:  PUSH  H      ; SAVE SECOND ARGUMENT IN CASE OF OVERFLOW
1785          28320 PUSH  D      ; SAVE FIRST ARGUMENT
1786          28340 CALL  IMULDV  ; FFIX UP THE SIGNS
1787          28360 PUSH  B,M      ; SAVE THE SIGN OF THE RESULT
1788          28380 MOV   C,L      ; COPY SECOND ARGUMENT INTO (BC)
1789          28400 LXI   H,SCODE  ; ZERO (HL), THAT IS WHERE THE PRODUCT GOES
1790          28420 MVII A,20      ; SET UP A COUNT
1791          28440 DAU   H      ; ROTATE PRODUCT LEFT ONE
1792          28460 IMULT1: JC    IHULTS ; CHECK FOR OVERFLOW
    
```

```

1793      28500 XCHG   A,M      JROTATE FIRST ARGUMENT LEFT ONE TO SEE IF
1794      28520 DAD   H      I WE ADD IN (BC) OR NOT
1795      28540 XCHG   B,D      JDON'T ADD IN ANYTHING
1796      28560 JNC   1MULT2
1797      28580 DAD   B      JADD IN (BC)
1798      28600 JC   1MULT5
1799      28620 IMULT2 DCR   A      JCHECK FOR OVERFLOW
1800      28640 XRA   Z      JMULT1
1801      28660 POP   B      JARE WE DONE?
1802      28680 POP   D      JNO, NOT AGAIN
1803      28700 IMULDIV1 MOV   A,M      JWE ARE DONE, SET SIGN OF RESULT
1804      28720 ORA   A      JGET ORIGINAL FIRST ARGUMENT
1805      28740 JM   1MULT3
1806      28760 POP   D      JIF IT IS, CHECK FOR SPECIAL CASE OF -32768
1807      28780 MOV   A,B      JRESULT IS OK, GET SECOND ARGUMENT OFF STACK
1808      28800 JMP   1NEGA
1809      28820 IMULT3 XRI   200      JGET THE SIGN OF RESULT IN A
1810      28840 POP   M      JNEGATE THE RESULT IF NECESSARY
1811      28860 JZ   1MULT4
1812      28880 XCHG   B,D      JNOTE! IF WE GET HERE FROM IDIV, THE RESULT
1813      28900 XWD   1000,001      JMUST BE 32768, IT CANNOT BE GREATER
1814      28920 IMULT51 POP   B      JIF IT IS, GT_32768, WE HAVE OVERFLOW
1815      28940 POP   H      JGET SIGN OF RESULT OFF STACK
1816      28960 CALL   CUNSIM
1817      28980 POP   M      JGET THE ORIGINAL FIRST ARGUMENT
1818      29000 CALL   PUSHF
1819      29020 CALL   CUNSIM
1820      29040 FMULTT1 PUPR
1821
1822      29060 JML   1MULT
1823      29080 IMULT41 HLT   A,B      JMULTPLY THE ARGUMENTS USING SINGLE PRECISION
1824      29100 ORA   A      JIS RESULT = 32768 OR -32768
1825      29120 POP   B      JDISCARD ORIGINAL SECOND ARGUMENT
1826      29140 RM
1827      29160 PUSH   D
1828      29180 CALL   CUNSIM
1829      29200 POP   D      JIF IT IS POSITIVE, SAVE REMAINDER FOR MOD
1830      29220 CALL   CUNSIM
1831      29240 POP   D      JFLOAT MOD'S REMAINDER BACK
1832      29260 JMP   NEG
1833      29280 JNEGATE THE RESULT IF NECESSARY
1834      29300 JREMAINDER IS IN (DE), QUOTIENT IN (HL)
1835      29320 JALTERS A,B,C,D,E,H,L
1836      29340 IDIV1: MOV   A,M      JCHECK FOR DIVISION BY ZERO
1837      29360 ORA   L
1838      29380 JZ   DVBRN      JWE HAVE DIVISION BY ZERO!
1839      29400 CALL   IMULDV
1840      29420 PUSH   PSW
1841      29440 XCHG   B
1842      29460 CALL   INEGML
1843      29480 MOV   B,M      JSAVE THE SIGN OF THE RESULT
1844      29500 MOV   C,L      JGET DENOMINATOR IN (HL)
1845      29520 LXI   M,SCODE
1846
1847      29540 MVI   A,21      JZERO WHERE WE DO THE SUBTRACTION
1848      29560 PUSH   PSW
1849      29580 ORA   A
1850      29600 JMP   IDIV3
1851      29620 IDIV1: PUSH   PSW
1852      29640 PUSH   M
1853      29660 JNC   IDIV2
1854      29680 JZ   IDIV2
1855      29700 XWD   1000,076
1856      29720 SIC
1857      29740 IMULDV
1858      29760 IDIV2: POP   M
1859      29780 MOV   A,E
1860      29800 HAL
1861      29820 MOV   E,A
1862      29840 MOV   A,D
1863      29860 RAL
1864      29880 MOV   D,A
1865      29900 MOV   A,L
1866      29920 RAL
1867      29940 MOV   L,A
1868      29960 MOV   A,M
1869      29980 RAL
1870      30000 MVI   M,A
1871      30020 PUSH   PSW
1872      30040 DCR   A
1873      30060 JNZ   IDIV1
1874      30080 XCHG   B
1875      30100 POP   B
1876      30120 PUSH   D
1877      30140 JMP   IMULDIV
1878      30160 JCHECK FOR SPECIAL CASE OF 32768
1879      30180
1880      30200 JGET READY TO MULTIPLY OR DIVIDE
1881      30220 JALTERS A,B,C,D,E,H,L
1882      30240 IMULDV: MOV   A,M      JGET SIGN OF RESULT
1883      30260 XRA   D
1884      30280 MOV   B,A
1885      30300 CALL   INEGH
1886      30320 XCHG   B
1887      30340 POP   D
1888      30360 JNEGATE H,L
1889      30380 JALTERS A,C,H,L
1890      30400 INEGH: MOV   A,H
1891      30420 INEGH: ORA   A
1892      30440 INEGH: MOV   A,H
1893      30460 INEGH: ORA   A
1894      30480 INEGH: XRA   A
1895      30500 MOV   C,A
1896      30520 SUB   L
1897      30540 MOV   L,A
1898      30560 MOV   A,C
1899

```

```

1846      29540 MVI   A,21      JSET UP A COUNT
1847      29560 PUSH   PSW
1848      29580 ORA   A
1849      29600 JMP   IDIV3
1850      29620 IDIV1: PUSH   PSW
1851      29640 PUSH   M
1852      29660 JNC   IDIV2
1853      29680 JZ   IDIV2
1854      29700 XWD   1000,076
1855      29720 SIC
1856      29740 IMULDV
1857      29760 IDIV2: POP   M
1858      29780 MOV   A,E
1859      29800 HAL
1860      29820 MOV   E,A
1861      29840 MOV   A,D
1862      29860 RAL
1863      29880 MOV   D,A
1864      29900 MOV   A,L
1865      29920 RAL
1866      29940 MOV   L,A
1867      29960 MOV   A,M
1868      29980 RAL
1869      30000 MVI   M,A
1870      30020 PUSH   PSW
1871      30040 DCR   A
1872      30060 JNZ   IDIV1
1873      30080 XCHG   B
1874      30100 POP   B
1875      30120 PUSH   D
1876      30140 JMP   IMULDIV
1877      30160 JCHECK FOR SPECIAL CASE OF 32768
1878      30180
1879      30200 JGET READY TO MULTIPLY OR DIVIDE
1880      30220 JALTERS A,B,C,D,E,H,L
1881      30240 IMULDV: MOV   A,M      JGET SIGN OF RESULT
1882      30260 XRA   D
1883      30280 MOV   B,A
1884      30300 CALL   INEGH
1885      30320 XCHG   B
1886      30340 POP   D
1887      30360 JNEGATE H,L
1888      30380 JALTERS A,C,H,L
1889      30400 INEGH: MOV   A,H
1890      30420 INEGH: ORA   A
1891      30440 INEGH: MOV   A,H
1892      30460 INEGH: ORA   A
1893      30480 INEGH: XRA   A
1894      30500 MOV   C,A
1895      30520 SUB   L
1896      30540 MOV   L,A
1897      30560 MOV   A,C
1898

```

MATHPK FOR BASIC MCS 8080 GATES/ALLEN/DAVIDOFF MACRO 47(113) 06:09 27-AUG-75 PAGE 10-4  
F4 MAC 23-AUG-64 06108 INTEGER ARITHMETIC ROUTINES

```

199 30600 S88 H    !NEGATE MO
1990 30620 MOV H,A   !SAVE IT
1991 30640 RET      !ALL DONE
1992
1993 30660
1994 30700 J INTEGER ABSOLUTE VALUE
1995 30720 FALTERS A,B,C,D,E,H,L
1996 30740 IABS1 LDH FACLO+1  !GET SIGN OF INTEGER IN FAC
1997 30760 DRA A      !CHECK ITS SIGN
1998 30780 RY      !IT IS POSITIVE, LEAVE IT ALONE
1999 30800
1999 30820 !FALL INTO INEG AND NEGATE IT
1999
1999 30840
1999 30860 J INTEGER NEGATION
1999 30880 FALTERS A,B,C,D,E,H,L
1999 30900 INEG1 LMD FACLO  !GET THE INTEGER
1999 30920 CALL INEGL  !NEGATE IT
1999 30940 SMLD FACLU  !STORE IT BACK IN THE FAC
1999 30960 XRI 200  !CHECK FOR SPECIAL CASE OF 32768
1999 30980 DRA L
1999 31000 RNZ
1999 31020 XCHG
1999 31040 MVI A,4  !IT DID NOT OCCUR, EVERYTHING IS FINE
1999 31060 STA VALTYP  !WE HAVE IT, FLOAT 32768
1999 31080 INEGADI MVI B,230  !CHANGE VALTYP TO "SINGLE PRECISION"
1999 31100 JMP FLOATH  !ENTRY FROM IADD, SET EXPONENT
1999 31120
1999 31140
1999 31160 JMOD OPERATOR
1999 31180 J (HL)1*(DE)-(UE)/(HL), (DE)=QUOTIENT
1999 31200 JALTRS A,B,C,D,E,H,L
1999
1999 31220 MOD1 PUSH 0  !SAVE (DE) FOR ITS SIGN
1999 31240 CALL IDIV  !DIVIDE AND GET THE REMAINDER
1999 31260 XCHG
1999 31280 MVI A,2  !PUT REMAINDER IN (DE)
1999 31300 STA VALTYP  !SET VALTYP TO "INTEGER" IN CASE RESULT OF
1999 31320 POP PSW  !THE DIVISION WAS 32768
1999 31340 JMP INEGA>  !GET THE SIGN OF (DE) BACK
1999 31360 PAGE  !NEGATE THE REMAINDER IF NECESSARY
1999

```

MATHPK FOR BASIC MCS 8080 GATES/ALLEN/DAVIDOFF MACRO 47(113) 06:09 27-AUG-75 PAGE 1  
F4 MAC 23-AUG-64 00:08 DOUBLE PRECISION ARITHMETIC ROUTINES

```

1938          SUBTIL DOUBLE PRECISION ARITHMETIC ROUTINES
1939          31400 IFE LENGTH=2,*  

1940          31420 COMMENT X  

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          FOR ONE ARGUMENT FUNCTIONS:  

1950          31600 THE FIRST ARGUMENT IS IN THE FAC, THE RESULT IS LEFT IN THE FAC  

1951          FOR TWO ARGUMENT OPERATIONS:  

1952          31640 THE FIRST ARGUMENT IS IN ARG=7,6,5,4,3,2,1,0 (NOTE: ARGLO=ARG=7)  

1953          31660 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 X  

1957          31760  

1958          31780  

1959          31800 ;DOUBLE PRECISION SUBTRACTION      FAC:=ARG=FAC  

1960          31820 ;ALTERS ALL REGISTERS  

1961          DSUB: 31840 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          DAUDI: 31960 LAD  H,ARG    ;GET POINTER TO EXPONENT OF FIRST ARGUMENT  

1968          31980 MUL  A,M    ;CHECK IF IT IS ZERO  

1969          32000 ORA   A  

1970          32020 RZ  

1971          32040 MOV   B,A    ;IT IS, RESULT IS ALREADY IN FAC  

1972          32060 DCX   H        ;SAVE EXPONENT FOR UNPACKING  

1973          32080 MOV   C,M    ;POINT TO HO AND SIGN  

1974          32100 LXI   D,FAC   ;GET HO AND SIGN FOR UNPACKING  

1975          32120 LDAX  D        ;GET POINTER TO EXPONENT OF SECOND ARGUMENT  

1976          32140 ORA   A        ;GET EXPONENT  

1977          32160 JZ    VMOVFA ;SEE IF IT IS ZERO  

1978          32180 SUB   B        ;IT IS, MOVE ARG TO FAC AND WE ARE DONE  

1979          32200 JNC   DADD2   ;SUBTRACT EXPONENTS TO GET SHIFT COUNT  

1980          32220 CMA   D        ;PUT THE SMALLER NUMBER IN FAC  

1981          32240 INR   A        ;NEGATE SHIFT COUNT  

1982          32260 PUSH  PSH    ;SAVE SHIFT COUNT  

1983          32280 PUSH  B    ;SAVE HO TO UNPACK LATER  

1984          32300 MWI   C,10   ;SWITCH FAC AND ARG, SET UP A COUNT  

1985          32320 INR   A  

1986          DADDI: 32340 LDAX  D    ;POINT TO ARG  

1987          32360 MOV   B,M    ;GET A BYTE OF THE FAC  

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

```

```

1991    32440  DCX D      POINT TO THE NEXT LO BYTE OF FAC
1992    32459  DCX H      POINT TO THE NEXT LO BYTE OF ARG
1993    32480  DCR C      ARE WE DONE?
1994    32500  JNZ DADU1   NO, DO THE NEXT LO BYTE
1995    32520  POP B      GET THE HO BACK
1996    32548  POP PSW     GET THE SHIFT COUNT BACK
1997    32560  DADU1 D,1    ARE WE WITHIN 56 BITS?
1998    32580  RNC         NO
1999    32600  PUSH PSW    SAVE SHIFT COUNT
2000    32620  CALL UNPACK  UNPACK THE NUMBERS
2001    32640  MOV B,A    ISAVE SUBTRACTION FLAG
2002    32660  MOV A,C    ISAVE THE UNPACKED HO
2003    32680  STA ANG+1   GET SHIFT COUNT
2004    32700  POP PSW    ISHIFT FAC RIGHT THE RIGHT NUMBER OF TIMES
2005    32720  CALL DSHTFR  GET SUBTRACTION FLAG AND ARG
2006    32740  DRA B      SUBTRACT NUMBERS IF THEIR SIGNS ARE DIFFERENT
2007    32760  JP DADD3   JSIGNS ARE THE SAME, ADD THE NUMBERS
2008    32780  CALL DADDAA   JROUND THE RESULT IF NO CARRY
2009    32800  JNC DROUND  JROUND THE RESULT IF CARRY
2010    32820  INR M      JWE HAVE OVERFLOW, ADD ONE TO THE EXPONENT
2011    32840  JZ OVERR   JCHECK FOR OVERFLOW
2012    32860  CALL DSHPRA  ISHIFT NUMBER RIGHT ONE, SHIFT IN CARRY
2013    32880  DADU1 D,1    ROUND THE RESULT
2014    32900  JMP DROUND  JMWVI "A", SUBTRACT THE NUMBERS
2015    32920  DADD3 X=0D 1000,076  JMWVI "B", GET THE SUBTRACT INSTRUCTION IN A
2016    32940  SBB M      ISUBTRACT THE NUMBERS
2017    32960  CALL DADUA   JPOINT TO THE UNPACKED SIGN
2018    32980  INX M      JCOMPLEMENT IT, SINCE THE FAC WAS SMALLER
2019    33000  MOV A,M    33020  CMA
2020    33020  CMA
2021    33040  MOV M,A    33060  MOV M,A
2022    33060  CC DNEGR   INEGATE THE RESULT IF IT WAS NEATIVE
2023    33080  CC DNEGR   IFALL INTO DROUND
2024
2025
2026
2027    33140  JNORMALIZE FAC
2028    33160  JALTERS A,B,C,D,H,L
2029    33180  DNORML1 XRA A      33190  CLRSHFT COUNT
2030    33200  DNORML1 M,B,A   33210  JSAVE SHIFT COUNT
2031    33220  LDA FAC+1   33230  JGET HO
2032    33240  DRA A      33250  JSEE IF WE CAN SHIFT 8 LEFT
2033    33260  JNZ DNORMS  33270  JWE CAN'T, SEE IF NUMBER IS NORMALIZED
2034    33280  LXI H,DFACLO+1 33290  JGET POINTER TO LO
2035    33300  CALL DSFLLC  33310  JSHIFT FAC LEFT
2036    33320  DNORML2 MOV D,M   33330  JDNEGR
2037    33340  MOV M,A    33350  JPUT IN BYTE FROM LAST LOCATION, THE FIRST
2038    33360  HLT                   F TIME THROUGH A IS ZERO
2039    33380  MOV A,D    33390  JINCREMENT BYTE IN A FOR NEXT TIME
2040    33400  INX H      JINCREMENT POINTER TO NEXT HIGHER ORDER
2041    33420  DCR C      JARE WE DONE?
2042    33440  JNZ DNORM2   JNO, DO THE NEXT BYTE
2043    33460  MOV A,B    JSUBTRACT 8 FROM SHIFT COUNT
        33480  SUI 10

```

```

2044    33500  CPI 300    JHAVE WE SHIFTED ALL BYTES TO ZERO?
2045    33520  JNZ DNORM1  JNO, TRY TO SHIFT 8 MORE
2046    33540  JMP ZERO   YES, THE NUMBER IS ZERO
2047    33560  DNORM31 DCR B    JDECREMENT SHIFT COUNT
2048    33580  LXI H,DFACLO+1 33590  JGET POINTER TO LO
2049    33600  CALL DSFLLC  33610  JSHIFT FAC LEFT
2050    33620  DRA A      33630  JSEE IF NUMBER IS NORMALIZED
2051    33640  DNORM51 JP DNORM3  JSHIFT FAC LEFT ONE IF IT IS NOT NORMALIZED
2052    33660  MOV A,B    33670  JGET SHIFT COUNT
2053    33680  DRA A      33690  JSEE IF NO SHIFTING HAS DONE
2054    33700  JZ DROUND  JNONE WAS, PROCEED TO ROUND THE NUMBER
2055    33720  LXI H,FAC   33730  JGET POINTER TO EXPONENT
2056    33740  ADD M      JUPDATE IT
2057    33760  MOV H,A    JSAVE UPDATED EXPONENT
2058    33780  JNC ZERO   JUNDERFLOW, THE RESULT IS ZERO
2059    33800  RZ             RESULT IS ALREADY ZERO, WE ARE DONE
2060
2061
2062    33840
2063
2064
2065    33860  JROUND FAC
2066    33880  JALTERS A,B,H,L
2067    33900  DROUN1: XRA M,DFACLO+1 33910  JGET EXTRA BYTE TO SEE IF WE HAVE TO ROUND
2068    33920  DROUN1: DRA A,DFACLO+1 33930  JENTRY FROM DDIV
2069    33940  DROUN1: CM DROUNA   JROUND UP IF NECESSARY
2070    33960  DROUN1: LXI H,FAC+1 33970  JGET POINTER TO UNPACKED SIGN
2071    34000  DROUN1: MOV A,M    33980  JGET SIGN
2072    34020  DROUN1: ANI 200   33990  JISOLATE SIGN BIT
2073    34040  DROUN1: DCX H      34000  JPOINT TO HO
2074    34060  DROUN1: DCX H
2075    34080  DROUN1: XRA M,   JPACK SIGN AND HO
2076    34100  DROUN1: MOV H,A    JPUT PACKED SIGN AND HO IN FAC
2077    34120  DROUN1: RET       THE ARE DONE
2078
2079    34160  JSUBROUTINE FOR ROUND1: ADD ONE TO FAC
2080    34200  DROUN1: LXI H,DFACLO   JGET POINTER TO LO, ENTRY FROM DINT
2081    34220  DROUN1: MOV B,7    JSET UP A COUNT
2082    34240  DROUN1: INR M      JINCREMENT BY ONE
2083    34260  DROUN1: JNZ H     JTEST IF THERE WAS NO CARRY
2084    34280  DROUN1: INX H      JINCREMENT POINTER TO NEXT HIGHER ORDER
2085    34300  DROUN1: DCR B      JHAVE WE INCREMENTED ALL BYTES
2086    34320  DROUN1: JNZ DROUN1  JNO, TRY THE NEXT ONE
2087    34340  DROUN1: INR M      YES, INCREMENT THE EXPONENT
2088    34360  DROUN1: JZ OVERR   JCHECK FOR OVERFLOW
2089    34380  DROUN1: DCX H      JTHE NUMBER OVERFLOWED ITS EXPONENT
2090    34400  DROUN1: MOV H,200  JPUT 200 IN HO
2091    34420  DROUN1: RET       FALL DONE
2092
2093    34460
2094    34480  JADD OR SUBTRACT 2 DBL QUANTITIES
2095    34500  JALTERS A,C,D,E,H,L
2096    34520  DADD3: LXI H,FBUFFR*D17  JENTRY FROM DDIV
        34540  LXI D,ARGLO  JADD OR SUBTRACT FBUFFER+17 AND ARG

```

```

2097      34560  JMP    DADDUS   ;DO THE OPERATION
2098      34580
2099      34600  DADDAA: XWD  1000,076  ;MV1 = A", ENTRY FROM DADD, DMULT
2100      34620  ADC    M      ;SETUP ADD INSTRUCTION FOR LDOP
2101      34640  DADDA: LXI  H,ANGLE  ;GET POINTER TO ARG, ENTRY FROM DADU
2102      34660  DADFOF: LXI  H,DFACLO ;GET POINTER TO FAC, ENTRY FROM FOUT
2103      34680  DADDS: LXI  H,C7    ;SET UP A COUNT
2104      34700  STA    DADDP   ;STORE ADD OR SUBTRACT INSTRUCTION
2105      34720  XRA    A      ;CLEAR CARRY
2106      34740  DADDL: LDAX  D      ;GET A BYTE FROM RESULT NUMBER
2107      34760  DADDP: 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
2114
2115
2116      34920
2117      34940  ;NEGATE SIGNED NUMBER IN FAC
2118      34960  ;THIS IS USED BY DADD, DINT
2119
2120      34980  ;ALTERS A,B,C,H,L
2121      35000  ONEGR: MOV    A,M   ;COMPLEMENT SIGN OF FAC
2122      35020  STA    A      ;JUSE THE UNCHANGED SIGN BYTE
2123      35040  MOV    M,A   ;SAVE THE NEW SIGN
2124      35060  LXI    H,DFACLO=1 ;GET POINTER TO LO
2125      35080  MVI    B,10  ;SET UP A COUNT
2126      35100  XRA    A      ;CLEAR CARRY AND GET A ZERO
2127      35120  MOV    C,A   ;SAVE ZERO IN C
2128      35140  ONEGR1: MOV   A,C   ;GET A ZERO
2129      35160  SBB   M      ;NEGATE THE BYTE OF FAC
2130      35180  MOV    M,A   ;PUT INTO FAC
2131      35200  INX    H      ;INCREMENT POINTER TO NEXT HIGHER ORDER BYTE
2132      35220  DCR    B      ;ARE WE DONE?
2133      35240  JNZ    ONEGR1 ;NO, NEGATE THE NEXT BYTE
2134      35260  RET
2135
2136      35300
2137      35320  ;SHIFT DBL FAC RIGHT ONE
2138      35340  ;ALTERS A,D,E,H,L
2139      35360  DSHFR1: LXI  H,DFACLO=1 ;ENTRY POINTER TO LO
2140      35380  MVI    H,0   ;PUT ZERO IN EXTRA LO ORDER BYTE
2141      35400  JC    DSHFR3  ;SEE IF WE CAN SHIFT 8 RIGHT
2142      35420  DSHFR11: SUI  10
2143      35440  JC    DSHFR3  ;WE CAN'T, CHECK IF WE ARE DONE
2144      35460  DSHFRM1: LXI  H,FAC=1 ;ENTRY FROM DMULT, GET POINTER TO HD
2145      35480  MVI    E,0   ;SHIFT A ZERO INTO THE HD
2146      35500  JC    DSHFR2  ;SET UP A COUNT
2147      35520  DSHFR2: MOV   C,M   ;SAVE A BYTE OF FAC
2148      35540  MOV    M,E   ;PUT THE LAST BYTE IN ITS PLACE
2149      35560  MOV    E,C   ;SET UP E FOR NEXT TIME THROUGH THE LOOP
2150      35580  DCX    H      ;POINT TO NEXT LOWER ORDER BYTE
2151      35600  DCR    D      ;ARE WE DONE?

```

```

2152      35620  JNZ    DSHFR2  ;NO, DO THE NEXT BYTE
2153      35640  JMP    DSHFR1  ;YES, SEE IF WE CAN SHIFT OVER 8 MORE
2154      35660  DSHFR3: ADI  11  ;CORRECT SHIFT COUNT
2155      35680  MOV    D,A   ;SAVE SHIFT COUNT IN D
2156      35700  DSHFR4: XRA  A      ;CLEAR CARRY
2157      35720  DCR    D      ;ARE WE DONE?
2158      35740  RAR
2159      35760  DSHFR1: LXI  H,FAC=1 ;NO, GET POINTER TO LO, ENTRY FROM DADD, DMULT
2160      35780  MVI    E,10  ;SET UP A COUNT, ROTATE FAC ONE LEFT
2161      35800  DSHFR51: MOV   A,M   ;GET A BYTE OF THE FAC
2162      35820  RAR
2163      35840  MOV    M,A   ;PUT THE UPDATED BYTE BACK
2164      35860  DCX    H      ;DECREMENT POINTER TO NEXT LOWER ORDER BYTE
2165      35880  DCR    E      ;ARE WE DONE?
2166      35900  JNZ    DSHFR5  ;NO, ROTATE THE NEXT LOWER ORDER BYTE
2167      35920  JMP    DSHFR4  ;YES, SEE IF HE ARE DONE SHIFTING
2168
2169      35940
2170      35960  ;ROTATE FAC LEFT ONE
2171      36000  ;ALTERS A,C,H,L
2172      36020  DSHFLC: MVI  C,10  ;SET UP A COUNT
2173      36040  DSHFLT1: MOV   A,M   ;GET BYTE OF FAC
2174      36060  RAR
2175      36080  MOV    M,A   ;ROTATE FAC LEFT ONE
2176      36100  INX    H      ;INCREMENT POINTER TO NEXT HIGHER ORDER BYTE
2177      36120  DCR    C      ;ARE WE DONE?
2178      36140  JNZ    DSHFLT1 ;NO, ROTATE THE NEXT BYTE
2179      36160  RET
2180
2181      36180
2182      36240  ;DOUBLE PRECISION MULTIPLICATION          FAC1=ARG*FAC
2183      36260  DMULT: FSIGN  ;CHECK IF WE ARE MULTIPLYING BY ZERO
2184      36280  RZ
2185      36300  CALL   MUL0VA  ;ADD EXPONENTS AND TAKE CARE OF SIGNS
2186      36320  CALL   DMULDV  ;ZERO FAC AND PUT FAC IN FBUFFR
2187      36340  MOV    M,C   ;PUT UNPACKED HD IN ARG
2188      36360  LXI    H,ARGLO ;GET POINTER TO LO OF ARG
2189      36380  MVI    M,7   ;SET UP A COUNT
2190      36400  DSHLT2: LDAX  D      ;GET THE HD OF ARG TO MULTIPLY BY
2191      36420  INX    D      ;INCREMENT POINTER TO NEXT HIGHER BYTE
2192      36440  DRA    A      ;CHECK IF WE ARE MULTIPLYING BY ZERO
2193      36460  PUSH   D      ;SAVE POINTER TO ARG
2194      36480  JZ    DMULTS  ;WE ARE
2195      36500  MVI    C,10  ;SET UP A COUNT
2196      36520  DMULT3: PUSH  B      ;SAVE COUNTERS
2197      36540  RAR
2198      36560  MOV    B,A   ;ROTATE MULTIPLIER RIGHT
2199      36580  CC    DADDAA  ;ADD IN OLD FAC IF BIT OF MULTIPLIER WAS ONE
2200      36600  MVI    D,1   ;ROTATE PRODUCT RIGHT ONE
2201      36620  CALL   DSHFR4
2202      36640  MOV    A,B   ;GET MULTIPLIER IN A
2203      36660  PDP    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      ;JARE WE DONE?
2208          36750      JNZ   DMULT2  ;NO, MULTIPLY BY NEXT HIGHER ORDER BY OF ARG
2209          36760      JMP   NORMAL  ;FALL DONE, NORMALIZE AND ROUND RESULT
2210          36600      DMULT5: CALL  DSHPRM  ;SHIFT PRODUCT RIGHT ONE BYTE, WE ARE
2211          36620      JMP   DMULT4  ;MULTIPLYING BY ZERO
2212
2213          36640
2214          36660      ;CONSTANT FOR DIV10, DDIV10
2215          36680      DTEN: 0000  ;1000
2216
2217          36690      0000
2218          36690      0000
2219          36690      FTEN: 0000  ;10-6
2220          37000      0000
2221          37020      0000
2222          37040      204
2223
2224          37060      ;DOUBLE PRECISION DIVIDE FAC BY 10
2225          37100      ;ALTERS ALL REGISTERS
2226          37120      DDIV10: CALL  VMOVAF  ;SAVE THE FAC IN ARG
2227          37140      LXI   H,DTEN  ;GET POINTER TO A DOUBLE PRECISION 10
2228          37160      CALL  VMOVFM  ;MOVE TEN INTO THE FAC
2229
2230          37180      ;FALL INTO DDIV AND DIVIDE BY TEN
2231
2232          37200
2233          37220      ;DOUBLE PRECISION DIVISION  FAC=ARG/FAC
2234          37240      ;ALTERS ALL REGISTERS
2235          37260      DSIGN: JZ    DVERR  ;CHECK FOR DIVISION BY ZERO
2236          37280      CALL  MULDVS  ;DON'T LET HIM DO IT
2237          37300      INR   M      ;SUBTRACT EXPONENTS AND CHECK SIGNS
2238          37320      INR   M      ;ADD TWO TO EXPONENT TO CORRECT SCALING
2239          37340
2240          37360      CALL  DMULDV  ;ZERO FAC AND PUT FAC IN FBUFFR
2241          37380      LAH   H,ARG  ;GET POINTER TO THE EXTRA MU BYTE WE WILL USE
2242          37400      MOV   M,C      ;ZERO IT
2243          37420      MWI   B,0      ;ZERO FLAG TO SEE WHEN WE START DIVIDING
2244          37440      DDIV1: XHD  1000,0876  ;MVII A, , SUBTRACT FBUFFR FROM ARG
2245          37460      SBB   M      ;GET SUBTRACT INSTRUCTION
2246          37500      CALL  DADDD  ;DO THE SUBTRACTION
2247          37520      LDAD  D      ;SUBTRACT FROM EXTRA MU BYTE
2248          37540      SBB   M      ;MVII C, , SET CARRY
2249          37560      JC    DDIV2  ;JCARRY IF SUBTRACTION WAS GOOD
2250          37600      XHD  1000,0876  ;MVII A, NO, ADD FBUFFR BACK IN
2251          37620      ADC   M      ;GET ADD INSTRUCTION
2252          37640      CALL  DADDD  ;DO THE ADDITION
2253          37660      XRA  A      ;CLEAR CARRY
2254          37680      XHD  1000,532  ;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
2260          37780      RAR
2261          37820      JM    DRUND  ;WE ARE DONE, WE HAVE 57 BITS OF ACCURACY
2262          37840      HAL
2263          37860      LXI   H,DFACLO  ;GET OLD CARRY BACK WHERE IT BELONGS
2264          37880      MVII C,7      ;SET UP A COUNT, SHIFT FAC LEFT ONE
2265          37900      CALL  DSHPRL  ;SHIFT FAC ONE LEFT IN THE QUOTIENT
2266          37920      CALL  DSHPRLC  ;GET POINTER TO LO IN ARG
2267          37940      MOV   A,B      ;SHIFT DIVIDEND ONE LEFT
2268          37960      QRA   A      ;IS THIS THE FIRST TIME AND WAS THE
2269          37980      QRA   A      ;SUBTRACTION NOT GOOD? (WILL GET
2270          38000      JNZ   DDIV1  ;CHANGED ON THE FIRST OR SECOND SUBTRACTION)
2271          38020      LXI   H,FAC  ;YES, SUBTRACT ONE EXPONENT TO CORRECT
2272          38040      DCR   M      ;SCALING
2273          38060      JNZ   DDIV1  ;CONTINUE DIVIDING IF NO OVERFLOW
2274          38080      JRP   OVERK  ;WE HAVE OVERFLOW!!
2275
2276          38120      ;TRANSFER FAC TO FBUFFR FOR DMULT AND DDIV
2277          38140      ;ALTERS A,B,C,D,E,H,L
2278          38160      DMULDV: MOV   A,C      ;PUT UNPACKED MU BACK IN ARG
2279
2280          38180      STA   ARG+1  ;POINT TO MU OF FAC
2281          38200      DCX   H      ;POINT TO END OF FBUFFR
2282          38220      LXI   D,FBUFFR-*023  ;POINT TO THE NEXT BYTE IN FBUFFR
2283          38240      INR   M      ;SET UP A COUNT
2284          38260      MVII C,8      ;GET A ZERO TO FILL FAC WITH
2285          38280      DMLUV1: MOV   A,M      ;GET A BYTE FROM FAC
2286          38300      STAX  D      ;PUT IT IN FBUFFR
2287          38340      MOV   M,C      ;PUT A ZERO IN FAC
2288          38360      DCX   D      ;POINT TO THE NEXT BYTE IN FBUFFR
2289          38380      DCX   H      ;POINT TO THE LOWER ORDER BYTE IN FAC
2290          38400      DCR   B      ;JARE WE DONE?
2291          38420      JNZ   DMLUV1  ;NO, TRANSFER THE NEXT BYTE
2292          38440      RET
2293
2294          38460
2295          38480      ;DOUBLE PRECISION MULTIPLY THE FAC BY 10
2296          38500      ;ALTERS ALL REGISTERS
2297          38520      DMUL10: CALL  VMOVAF  ;SAVE THE FAC IN ARG
2298
2299          38540      ;VMOVAF EXITS WITH (DE)=FAC+1
2300          38560      XCHG  H      ;GET THE POINTER INTO THE FAC IN (HL)
2301          38600      DCX   H      ;POINT TO THE EXPONENT
2302          38620      MOV   A,M      ;GET THE EXPONENT
2303          38640      ADI   2      ;MULTIPLY FAC BY 4 BY ADDING 2 TO THE EXPONENT
2304          38660      JC    OVERR  ;CHECK FOR OVERFLOW
2305          38680      MOV   M,A      ;SAVE THE EXPONENT
2306          38700      CALL  DADDD  ;ADD TO THE ORIGINAL FAC TO GET 5 TIMES FAC
2307          38720      POP   M      ;GET THE POINTER TO FAC BACK
2308          38740      INR   M      ;ADD ONE TO EXPONENT TO GET 10 TIMES FAC
2309          38760

```

2309	38780	RNZ	JALL DONE IF OVERFLOW DID NOT OCCUR	
2310	38800	JMP	OVERR>	IF DID, GIVE THE APPROPRIATE MESSAGE
2311	38820	PAGE		

2312	38640	SUBTL	FLOATING POINT INPUT ROUTINE	
2313	38660	HALTS ALL REGISTERS		
2314	38680	TESTS FOR A DECIMAL LEFT IN FAC		
2315	38700	JAT ENTRY, (ML) POINTS TO THE FIRST CHARACTER IN A TEXT BUFFER,		
2316	38720	THE FIRST CHARACTER IS ALSO IN A, WE PACK THE DIGITS INTO THE FAC		
2317	38740	JAS AN INTEGER AND KEEP TRACK OF WHERE THE DECIMAL POINT IS,		
2318	38760	JC IS 377 IF WE HAVE NOT SEEN A DECIMAL POINT, 0 IF WE HAVE,		
2319	38780	JD IS THE NUMBER OF DIGITS AFTER THE DECIMAL POINT,		
2320	38800	JAT THE END, B AND THE EXPONENT (IN E) ARE USED TO DETERMINE HOW MANY		
2321	38820	TIMES WE MULTIPLY OR DIVIDE BY TEN TO GET THE CURRENT NUMBER.		
2322	001531*	39040	FIN1:	
2323		39060	IFN:	
2324	001531* 001000 000376	39080	STRING,<	IF WE ARE CALLED BY VAL, THE SIGNS MAY NOT BE CRUNCHED
2325	001532* 000000 000055	39100	CPI "#"	ISSEE IF NUMBER IS NEGATIVE
2326	001533* 001000 000365	39120	PUSH PSM	ISAVE SIGN
2327	001534* 001000 000312	39140	JZ FIN1	IGNORE MINUS SIGN
2328	001535* 000000 001545*			
2329	001536* 000000 001546*			
2330	001537* 001000 000375	39160	CPI "#"	IGNORE A LEADING SIGN
2331	001540* 000000 000053	39180	JZ FIN1>	
2332	001541* 001000 000312			
2333	001542* 000000 001545*			
2334	001543* 000000 001535*			
2335	001544* 001000 000053	39200	DCX H	ISET CHARACTER POINTER BACK ONE
2336	001545* 001000 000315	39220	FIN1:	
2337	001546* 000000 000317*	39240	IFN:	
2338	001547* 000000 001542*	39260	LENGTH=2,<	
2339	001548* 000000 000315	39280	CALL ZERO	JCLEAR FAC
2340	001549* 000000 000317*			
2341	001547* 000000 001542*			
2342	001550* 001000 000107	39280	MOV B,A	ICLEAR FLAG: B=DECIMAL PLACE COUNT
2343	001551* 001000 000127	39300	MOV D,A	ID=SIGN OF EXPONENT
2344	001552* 001000 000137	39320	MOV E,A	IE=EXPONENT
2345	001553* 001000 000057	39340	CRA	
2346	001554* 001000 000017	39360	MOV C,A	ICP*," FLAG
2347	001555* 001000 000327	39380	HERE TO GET THE NEXT DIGIT OF THE NUMBER, A DECIMAL POINT OR AN "E"	
2348	001556* 001000 000332	39400	FIN1:	CHRGET
2349	001557* 001000 000332	39420	JC FINDIG	GET A CHARACTER
2350	001557* 000000 001702*			DO WE HAVE A DIGIT?
2351	001556* 000000 001546*			
2352	001561* 021000 000376	39440	CPI "#,"	TEST FOR DECIMAL POINT
2353	001562* 000000 000055	39460	JZ FINDP	
2354	001563* 001000 000112			
2355	001564* 000000 001135*			
2356	001565* 000000 001557*			
2357	001566* 001000 000376	39480	CPI "#,"	ICHECK FOR BEGINNING OF EXPONENT
2358	001567* 000000 000105	39500	JNZ FINE	INONE OF THE ABOVE? SO END OF NUMBER
2359	001570* 001000 000302			
2360	001571* 000000 001641*			
2361	001572* 000000 001564*			
2362	001573* 001000 000327	39520		HERE TO CHECK FOR THE SIGN OF THE EXPONENT
2363	001573* 001000 000327	39540	CHRGET	ICHECK FOR ITS SIGN
2364		39560	IFN:	STRING,<