```
FFFFFFFFFFFFFFF          333333333
FFFFFFFFFFFFFFF          333333333
FFFFFFFFFFFFFFF          333333333
FFF                  333         333
FFF                  333         333
FFF                  333         333
FFF                              333
FFF                              333
FFF                              333
FFFFFFFFFFFF                 333
FFFFFFFFFFFF                 333
FFFFFFFFFFFF                 333
FFF                              333
FFF                              333
FFF                              333
FFF                  333         333
FFF                  333         333
FFF                  333         333
FFF                      333333333
FFF                    333333333
FFF                    333333333
```

```
LPTSPL VERSION 6(344)  RUNNING ON LPT0
 *START* USER MITS  (6000,6000) JOB  PS SEQ, 42 DATE 10-SEP-75 08:24:32 MONITOR ALBUQUERQUE SCHOOLS 5078 *START*
REQUEST CREATED: 10-SEP-75  03:16:06
FILE: DSK0:PS(6000,6000) CREATED: 10-SEP-75 03:16:00 PRINTED: 10-SEP-75 08:52:43
QUEUE SWITCHES:  /FILE:ASCII /COPIES:2 /SPACING:1 /LIMIT:1305 /FORMS:NORMAL
FILE WILL BE DELETED AFTER PRINTING
```

```
BASIC MCS 8080  GATES/ALLEN/DAVIDOFF     MACRO 47(113)  03:12 10-SEP-75 PAGE 1
C       6-SEP-54 03:08          COMMON FILE

     1                                  00100  SEARCH  MCS808                  ;THE UNIVERSAL FILE
     2                                  00200  SUBTTL  COMMON FILE
     3                                  00300  SALL
     4                  000002          00400  LENGTH==2                       ; 0 MEANS 4K, 1 MEANS 8K, 2 MEANS 12K
     5                  000001          00500  REALIO==1
     6                  000000          00600  CASSW==0                        ;CASSETTE SWITCH (CSAVE,CLOAD)
     7                  000000          00700  PURE==0                         ;ON FOR PURE CODE
     8                  000000          00800  LPTSW==0
     9                  000000          00900  DSKFUN==0                       ;ON TO READ/WRITE
    10                  000000          01000  CONSSW==0
    11
    12                  000016          01200  CLMWID==^D14                    ;MAKE COMMA COLUMNS FOURTEEN CHARACTERS
    13                  020000          01300  RAMBOT=^O20000                  ;BOTTOM LOCATION OF RAM FOR PURE SWITCH
    14                  000001          01400  CONTRW==1                       ;ALLOW ^O
    15                                  01500  IFE     REALIO,<
    16                                  01600          LPTSW==0                ;SIMULATOR DEFAULTS
    17                                  01700          CASSW==0
    18                                  01800          CONSSW==0
    19                                  01900          DSKFUN==0
    20                                  02000          CONTRW==0>
    21
    22                                  02200  IFE     LENGTH,<
    23                                  02300          EXTFNC==0               ;ON MEANS EXTENDED FUNCTIONS
    24                                  02400          MULDIM==0               ;ON MEANS MULTIPLE DIMENSIONED ARRAYS ALLOWED
    25                                  02500          STRING==0               ;ON MEANS STRINGS ALLOWED
    26                                  02600          CASSW==0
    27                                  02700          LPTSW==0
    28                                  02800          DSFFUN==0
    29                                  02900          CONSSW==0
    30                                  03000          CONTRW==0>
    31
    32                                  03200  IFE     LENGTH-1,<
    33                                  03300          EXTFNC==1
    34                                  03400          MULDIM==1
    35                                  03500          STRING==1>
    36
    37                                  03700  IFE     LENGTH-2,<
    38                  000001          03800          EXTFNC==1
    39                  000001          03900          MULDIM==1
    40                  000001          04000          STRING==1>
    41
    42                                  04200  DEFINE  SYNCHK(A),<RST  1
    43                                  04300                      A>
    44                                  04400  DEFINE  CHRGET,<RST     2>
    45                                  04500  DEFINE  OUTCHK,<RST     3>
    46                                  04600  DEFINE  COMPAR,<RST     4>
    47                                  04700  DEFINE  FSIGN,<RST      5>
    48                                  04800  DEFINE  PUSHM,<RST      6>
    49                                  04900  DEFINE  PUSHFM,<PUSHM
    50                                  05000                  PUSHM>
    51                                  05100  DEFINE  ACRLF,<
    52                                  05200          ^D13
    53                                  05300  IFN     STRING,<^D10>>
```

```
BASIC MCS 8080  GATES/ALLEN/DAVIDOFF    MACRO 47(113) 03:12 10-SEP-75 PAGE 1-1
C       6-SEP-64 03:08          COMMON FILE

    54                                  05400   DEFINE  PUSHR,<
    55                                  05500           PUSH    D
    56                                  05600           PUSH    B>
    57                                  05700   DEFINE  POPR,<
    58                                  05800           POP     B
    59                                  05900           POP     D>
    60                                  06000   DEFINE  MOVRI(B,C,D,E),<
    61                                  06100           X=D     ^O1000,^O001    ;; "LXI B"
    62                                  06200           EXP     C
    63                                  06300           EXP     B
    64                                  06400           X=D     ^O1000,^O021    ;; "LXI D"
    65                                  06500           EXP     E
    66                                  06600           EXP     D>
    67
    68                                  06800   IF1,<
    69                                  06900   IFE     LENGTH,<PRINTX  /SMALL/>
    70                                  07000   IFE     LENGTH-1,<PRINTX        /MEDIUM/>
    71                                  07100   IFE     LENGTH-2,<PRINTX        /BIG/>
    72                                  07200   IFE     REALIO,<PRINTX  /SIMULATE/>
    73                                  07300   IFN     REALIO,<PRINTX  /ON MACHINE/>
    74                                  07400   IFN     CASSW,<PRINTX   /CASSETTE/>
    75                                  07500   IFN     PURE,<PRINTX    /PURE/>
    76                                  07600   IFN     LPTSW,<PRINTX   /LPT/>
    77                                  07700   IFN     DSKFUN,<PRINTX  /DISK/>
    78                                  07800   IFN     CONSSW,<PRINTX  /CONSOLE/>>
    79                                  07900   PAGE
```

```
BASIC MCS 8080  GATES/ALLEN/DAVIDOFF    MACRO 47(113) 03:12 10-SEP-75 PAGE 1
F3      MAC     6-SEP-64 03:11          VERSION 3,0 -- MORE FEATURES TO GO

    80                                  00020   SUBTTL  VERSION 3,0 -- MORE FEATURES TO GO
    81                                  00040   TITLE   BASIC MCS 8080  GATES/ALLEN/DAVIDOFF
    82                                  00060   IFNDEF  LENGTH,<PRINTX !!! MUST HAVE COM !!!
    83                                  00080           END>
    84  000000'                         00100   MCSSIM(START)
    85                                  00120   COMMENT *
    86
    87                                  00160   --------- ---- -- ---- ----- --- ---- -----
    88                                  00180   COPYRIGHT 1975 BY BILL GATES AND PAUL ALLEN
    89                                  00200   --------- ---- -- ---- ----- --- ---- -----
    90
    91
    92                                  00260   ORIGINALLY WRITTEN ON THE POP-10 FROM
    93                                  00280   FEBRUARY 9 TO  APRIL 9
    94
    95                                  00320   BILL GATES WROTE THE RUNTIME STUFF,
    96                                  00340   PAUL ALLEN WROTE THE NON-RUNTIME STUFF,
    97                                  00360   MONTE DAVIDOFF WROTE THE MATH PACKAGE,
    98
    99                                  00400   THINGS TO DO:
   100                                  00420   GOSUB / INPUT BUG  (BUF SMASH)
   101                                  00440   PRINT PUNCTUATION MANDATORY
   102                                  00460   MULTIPLE LET
   103                                  00480   RESTART AT 0 SHOULD  GO THROUGH  STKINI
   104                                  00500   USER DEFINED FUNCTIONS(MULTI-ARG,MULTI-LINE,STRINGS)
   105                                  00520   MAKE STACK BOUNDARY STUFF EXACT
   106                                  00540   PUNCH,RENUMBER,,,
   107                                  00560   INLINE CONSTANT CONVERSION--MAKE IT WORK
   108                                  00580   *
   109                                  00600   RADIX 10                        ;THROUGHOUT THE NON-MATH PACKAGE ROUTINES
   110                     000000       00620   .P==0                           ;FOR SIMULATOR FIXUPS
   111                     000025       00640   NUMLEV==17+LENGTH*2             ;NUMBER OF STACK LEVELS RESERVED
   112                                  00660                                   ;WITH AN EXPLICIT CALL TO GETSTK
   113                     000110       00680   LPTLEN==72                      ;WIDTH OF LINE-PRINTER
   114                     000110       00700   LINLEN==72                      ;TELETYPE LINE LENGTH
   115                     000110       00720   BUFLEN==72                      ;INPUT BUFFER SIZE
   116                     000004       00740   STRSIZ==4
   117                     000003       00760   IFE     LENGTH-2,<STRSIZ==3>
   118                     000003       00780   NUMTMP==3                       ;NUMBER OF STRING TEMPORARIES
   119                     000005       00800   IFE     LENGTH-2,<NUMTMP=5>
   120                     000015       00820   CR==13
   121                     000017       00840   CONT==15                        ;CHARACTER TO SUPRESS OUTPUT
   122                     000200       00860   ODONE==128                      ;BIT FOR OUTPUT DONE
   123                     000001       00880   IODNE==1                        ;BIT FOR INPUT DONE
   124                     000001       00900   TTOCHN==1                       ;TELETYPE OUTPUT CHANNEL
   125                     000001       00920   TTICHN==1                       ;TELETYPE INPUT CHANNEL
   126                     000000       00960   LISTEN==0                       ;ON MEANS LISTEN FOR ^C
   127                     000001       01000   FUNCTS==1                       ;ON MEANS USER FUNCTIONS ALLOWED
   128                                  01020                                   ;; CANNOT BE ON WITH LENGTH=0 (SUBFLG)
   129                                  01040   IFN     REALIO,<
   130                     000001       01060   LISTEN==1>
   131                                  01080   IFE     LENGTH,<
   132                                  01120   FUNCTS==0>
```

```
                                01200   INTERNAL        ,C1,BUF,READY,REASON,SNERR,OMERR,REPINI
                                01220   INTERNAL        STREND,CURLIN,DV0ERR,ERROV
                                01240   IFN     REALIO,<
                                01260   INTERNAL        CNLCA1,CNLCA2,CNLCA3>
                                01280   IFN     EXTFNC,<INTERNAL        ATNFIX,COSFIX,SINFIX,TANFIX
                                01300   EXTERNAL        FPWR,EXP>
                                01320   EXTERNAL        QINT,ZERO,MOVE,FOUT,FIN,FCOMP,FADD,PUSHF,INT,INIT
                                01340   EXTERNAL        MOVFR,MOVRF,MOVRM,INPRT,LINPRT
                                01360   EXTERNAL        MOVFM,MOVMF,TSTACK,FLOATRI,FADDS
                                01380   INTERNAL        ILLFUN,FAC,FACLO,TXTTAB,STROUT,SCRTCH
                                01400   EXTERNAL        INPART,NEG,FLOAT
                                01420   INTERNAL        OUTDO,STRQUI
                                01440   INTERNAL        STKTOP,ERROR,FCERR
                                01460   IFN     STRING,<
                                01462   INTERNAL          VALTYP,TEMPPT
                                01464   INTERNAL        TEMPST,STRLIT
                                01466   IFN     LENGTH-2,<
                                01468   INTERNAL        TMERR>
                                01480   INTERNAL        MEMSIZ,FRETOP
                                01500   EXTERNAL        SIGNS>
                                01520   INTERNAL        FBUFFR,MINUTK,PLUSTK,CRDO,LINGET,INTXT,GINLIN
                                01540   IFN     MULDIM,<INTERNAL        BSERR>
                                01560   IFN     MULDIM,<EXTERNAL        UMULT>
                                01580   IFE     LENGTH,<INTERNAL        RNDFIX,SQRFIX,SINFIX>
                                01600   INTERNAL        SIGN
                                01620   EXTERNAL        SIGNC,PGPHRT
                                01640   IFN     CONTRW,<
                                01660   INTERNAL        CNTWFL>
                                01680   IFN     LPTSW,<
                                01700   INTERNAL        LPTPOS,PRTFLG>
                                01720   IFE     LENGTH-2,<
                                01730   EXTERNAL        CONSIN,VMOVFA,VMOVAF,ISIGN,FPWRQ,CONIA,GETBCD,VSIGN
                                01740   EXTERNAL        VMOVMF,VMOVFM,FRCINT,FRCSNG,FRCDBL,VNEG,PUFOUT,DCXBRT,IADD
                                01760   EXTERNAL        ISUB,IMULT,IDIV,ICOMP,INEG,DADD,DSUB,DMULT,DDIV,DCOMP,VINT
                                01780   EXTERNAL        TMERR,VMOVE,VALINT,VALSNG,FRCSTR,CMKSTR,MAKINT
                                01800   INTERNAL        DFACLO,ARG,ARGLO,VALTYP,ERRTM,TEMP2,TEMP3,GETYPE>
                                01820   PAGE
```



```
                                01840   SUBTTL  SOME EXPLANATION

                                01880   COMMENT *

                                01920   ALTAIR BASIC CONFIGURES MEMORY AS FOLLOWS:

                                01960   LOW LOCATIONS

                                02000           RST     SUBROUTINES

                                02040                   0       STARTUP
                                02060                           INITIALLY A JMP TO THE INITIALIZATION CODE
                                02080                           BUT CHANGED TO A JMP TO READY.
                                02100                           RESTARTING THE MACHINE AT 0 DURING PROGRAM
                                02120                           EXECUTION CAN LEAVE THINGS MESSED UP.

                                02160                   1       SYNCHK
                                02180                           A CHECK IS MADE TO MAKE SURE THE
                                02200                           CHARACTER POINTER POINTS AT A SPECIFIC
                                02220                           CHARACTER, IF NOT THE "SYNTAX ERROR"
                                02240                           ROUTINE IS CALLED. IF SO,
                                02260                           THE CHRGET RST IS DROPPED INTO SO
                                02280                           THE CHARACTER AFTER THE MATCHED
                                02300                           ONE WILL BE PUT IN [A] AND
                                02320                           THE CONDITION CODES WILL REFLECT THIS
                                02340                           EXAMPLE: SYNCHK THENTK (THE MATCH CHARACTER IS
                                02360                           GIVEN IN THE LOCATION AFTER THE RST)
                                02380                           WOULD CHECK TO MAKE SURE (H,L) POINTED TO A THENTK
                                02400                           AND IF SO FETCH THE NEXT CHARACTER INTO [A],
                                02420                           IF NOT, A "SYNTAX ERROR" WOULD BE GIVEN.

                                02460                   2       CHRGET
                                02480                           USING (H,L) AS THE TEXT POINTER
                                02500                           THE TEXT POINTER IS INCREMENTED
                                02520                           AND THE NEXT CHARACTER IS FETCHED INTO [A]
                                02540                           IF THE CHARACTER IS A " " IT IS SKIPPED
                                02560                           OVER AND THE NEXT CHARACTER IS FETCHED.
                                02580                           THE STATEMENT TERMINATORS ":" AND 0
                                02600                           LEAVE THE ZERO FLAG SET.
                                02620                           THE NUMERICS "0" THROUGH "9" LEAVE THE CARRY
                                02640                           FLAG SET. THE CURRENT CHARACTER CAN BE
                                02660                           REFETCHED INTO [A] BY DOING A MOV A,M.
                                02680                           IF THE CONDITION CODES MUST BE SET UP AGAIN
                                02700                           DCX H,CHRGET WILL WORK, IT IS VERY DIFFICULT
                                02720                           TO REEXAMINE THE CHARACTER BEFORE THE CURRENT
                                02740                           ONE SINCE SPACES MAY BE IN-BETWEEN.
                                02760                           DCX H,DCX H,CHRGET WILL NOT ALWAYS WORK.

                                02800                   3       OUTCHR
                                02820                           THE CHARACTER IN [A] IS PRINTED ON
                                02840                           THE USER'S TERMINAL. [A] AND THE
                                02860                           CONDITION CODES ARE PRESERVED.
```

```
                                        4       COMPAR
                                                [D,E] AND [H,L] ARE COMPARED AS UNSIGNED
                                                DOUBLE-BYTE INTEGERS, CARRY IS SET IF
                                                [H,L] IS LESS THAN [D,E], ZERO IS SET IF THEY
                                                ARE EQUAL, [A] IS SMASHED, THE ONLY DEFINITE
                                                THING THAT CAN BE SAID ABOUT [A] ON RETURN
                                                IS THAT IF THE ZERO FLAG IS SET, [A] WILL
                                                EQUAL 0.

                                        5       FSIGN
                                                THE FAC (FLOATING ACCUMULATOR)
                                                WHICH IS USED TO STORE NUMERIC RESULTS
                                                IS CHECKED TO SEE WHAT SIGN ITS
                                                VALUE HAS.

                                        6       PUSHM
                                                A DOUBLE BYTE QUANTITY POINTED
                                                TO BY [H,L] IS PUSHED ONTO THE
                                                STACK, [B,C] IS SET EQUAL TO THE
                                                VALUE PUSHED, [H,L] IS INCREMENTED BY TWO.

                                        7       IN THE 4K VERSION RST 7 IS UNUSED AND THE LOCATIONS
                                                ASSOCIATED WITH IT ARE USED TO CONTINUE
                                                THE CODE FOR RST 6, IN THE 8K A JMP IS MADE
                                                AROUND THE FIRST THREE RST 7 LOCATIONS
                                                DURING RST 6 EXECUTION, RST 7 INITIALLY
                                                CONTAINS A RET, BUT THE USER CAN CHANGE IT TO
                                                A JMP TO AN INTERRUPT SERVICE ROUTINE.

                                FUNCTION DISPATCH ADDRESSES
                                        FUNDSP CONTAINS THE ADDRESSES OF THE
                                        FUNCTION ROUTINES IN THE ORDER OF THE
                                        FUNCTION NAMES IN THE CRUNCH LIST.
                                        THE FUNCTIONS THAT TAKE MORE THAN ONE ARGUMENT
                                        ARE AT THE END, SEE THE EXPLANATION AT ISFUN.

                                THE OPERATOR TABLE
                                        THE OPTAB TABLE CONTAINS AN OPERATORS PRECEDENCE
                                        FOLLOWED BY THE ADDRESS OF THE ROUTINE TO PERFORM
                                        THE OPERATION, THE INDEX INTO THE
                                        OPERATOR TABLE IS MADE BY SUBTRACTING OFF THE CRUNCH VALUE
                                        OF THE LOWEST NUMBERED OPERATOR, THE ORDER
                                        OF OPERATORS IN THE CRUNCH LIST AND IN OPTAB IS IDENTICAL.
                                        THE PRECEDENCES ARE ARBITRARY,EXCEPT FOR THEIR
                                        COMPARATIVE SIZES, NOTE THAT THE PRECEDENCE FOR
                                        UNARY OPERATORS SUCH AS NOT AND NEGATION ARE
                                        SETUP SPECIALLY WITHOUT USING A TABLE.

                                THE RESERVED WORD OR CRUNCH LIST
                                        WHEN A COMMAND OR PROGRAM LINE IS TYPED IN
                                        IT IS STORED IN BUF, AS SOON AS THE WHOLE LINE
                                        HAS BEEN TYPED IN (INLIN RETURNS) CRUNCH IS
                                        CALLED TO CONVERT ALL RESERVED WORDS TO THEIR
```



```
                                        CRUNCH VALUES, THIS REDUCES THE SIZE OF THE
                                        PROGRAM AND SPEEDS UP EXECUTION BY ALLOWING
                                        TABLE DISPATCHES TO PERFORM FUNCTIONS,STATEMENTS,
                                        AND OPERATIONS, THIS IS BECAUSE ALL THE STATEMENT
                                        NAMES ARE STORED CONSECUTITVELY IN THE CRUNCH LIST,
                                        WHEN A MATCH IS FOUND BETWEEN A STRING
                                        OF CHARACTERS AND A WORD IN THE CRUNCH LIST
                                        THE ENTIRE TEXT OF THE MATCHED WORD IS TAKEN OUT OF
                                        THE INPUT LINE AND A RESERVED WORD TOKEN IS PUT
                                        IN ITS PLACE, A RESERVED WORD TOKEN IS ALWAYS EQUAL
                                        TO OCTAL 200 PLUS THE POSITION OF THE MATCHED WORD
                                        IN THE CRUNCH LIST.

                                STATEMENT DISPATCH ADDRESSES
                                        WHEN A STATEMENT IS TO BE EXECUTED, THE FIRST
                                        CHARACTER OF THE STATEMENT IS EXAMINED
                                        TO SEE IF IT IS LESS THAN THE RESERVED
                                        WORD TOKEN FOR THE LOWEST NUMBERED STATEMENT NAME,
                                        IF SO, THE "LET" CODE IS CALLED TO
                                        TREAT THE STATEMENT AS AN ASSIGNMENT STATEMENT,
                                        OTHERWISE A CHECK IS MADE TO MAKE SURE THE
                                        RESERVED WORD NUMBER IS NOT TOO LARGE TO BE A
                                        STATEMENT TYPE NUMBER, IF NOT THE ADDRESS
                                        TO DISPATCH TO IS FETCHED FROM STMDSP (THE STATEMENT
                                        DISPATCH TABLE) USING THE RESERVED WORD
                                        NUMBER FOR THE STATEMENT TO CALCULATE AN INDEX INTO
                                        THE TABLE.

                                ERROR MESSAGES
                                        WHEN AN ERROR CONDITION IS DETECTED
                                        [E] MUST BE SET UP TO INDICATE WHICH ERROR
                                        MESSAGE IS APPROPRIATE AND A BRANCH MUST BE MADE
                                        TO ERROR, THE STACK WILL BE RESET AND ALL
                                        PROGRAM CONTEXT WILL BE LOST, VARIABLES
                                        VALUES AND THE ACTUAL PROGRAM REMAIN INTACT,
                                        ONLY THE VALUE OF [E] IS IMPORTANT WHEN
                                        THE BRANCH IS MADE TO ERROR, [E] IS USED AS AN
                                        INDEX INTO ERRTAB WHICH GIVES THE TWO
                                        CHARACTER ERROR MESSAGE THAT WILL BE PRINTED ON THE
                                        USER'S TERMINAL.

                                IMPURE STORAGE
                                        ALL TEMPORARIES,FLAGS,POINTERS, THE BUFFER AREA,
                                        THE FLOATING ACCUMULATOR,AND ANYTHING ELSE THAT
                                        IS USED TO STORE A CHANGING VALUE SHOULD BE LOCATED
                                        IN THIS AREA, CARE MUST BE MADE IN MOVING LOCATIONS
                                        IN THIS AREA SINCE THE JUXTAPOSITION OF TWO LOCATIONS
                                        IS OFTEN DEPENDED UPON.

                                TEXTUAL MESSAGES
                                        CONSTANT MESSAGES ARE STORED HERE, UNLESS
                                        THE CODE TO CHECK IF A STRING MUST BE COPIED
                                        IS CHANGED THESE STRINGS MUST BE STORED ABOVE
```

```
                                05020                   DSCTMP, OR ELSE THEY WILL BE COPIED BEFORE
                                05040                   THEY ARE PRINTED.
                                05080           FNDFOR
                                05100                   MOST SMALL ROUTINES ARE FAIRLY SIMPLE
                                05120                   AND ARE DOCUMENTED IN PLACE, FNDFOR IS
                                05140                   USED FOR FINDING "FOR" ENTRIES ON
                                05160                   THE STACK, WHENEVER A "FOR" IS EXECUTED AN
                                05180                   18 BYTE ENTRY IS PUSHED ONTO THE STACK,
                                05200                   BEFORE THIS IS DONE, HOWEVER, A CHECK
                                05220                   MUST BE MADE TO SEE IF THERE
                                05240                   ARE ANY "FOR" ENTRIES ALREADY ON THE STACK
                                05260                   FOR THE SAME LOOP VARIABLE, IF SO, THAT "FOR" ENTRY
                                05280                   AND ALL OTHER "FOR" ENTRIES THAT WERE MADE AFTER IT
                                05300                   ARE ELIMINATED FROM THE STACK, THIS IS SO A
                                05320                   PROGRAM THAT JUMPS OUT OF THE MIDDLE
                                05340                   OF A "FOR" LOOP AND THEN RESTARTS THE LOOP AGAIN
                                05360                   AND AGAIN WON'T USE UP 18 BYTES OF STACK
                                05380                   SPACE EVERY TIME, THE "NEXT" CODE ALSO
                                05400                   CALLS FNDFOR TO SEARCH FOR A "FOR" ENTRY WITH
                                05420                   THE LOOP VARIABLE IN
                                05440                   THE "NEXT", AT WHATEVER POINT A MATCH IS FOUND
                                05460                   THE STACK IS RESET, IF NO MATCH IS FOUND A
                                05480                   "NEXT WITHOUT FOR"  ERROR OCCURS, GOSUB EXECUTION
                                05500                   ALSO PUTS A 6 BYTE ENTRY ON STACK,
                                05520                   WHEN A RETURN IS EXECUTED FNDFOR IS
                                05540                   CALLED WITH A VARIABLE POINTER THAT CAN'T
                                05560                   BE MATCHED, WHEN "FNDFOR" HAS RUN
                                05580                   THROUGH ALL THE "FOR" ENTRIES ON THE STACK
                                05600                   IT RETURNS AND THE RETURN CODE MAKES
                                05620                   SURE THE ENTRY THAT WAS STOPPED
                                05640                   ON IS A GOSUB ENTRY, THIS ASSURES THAT
                                05660                   IF YOU GOSUB TO A SECTION OF CODE
                                05680                   IN WHICH A FOR LOOP IS ENTERED BUT NEVER
                                05700                   EXITED THE RETURN WILL STILL BE
                                05720                   ABLE TO FIND THE MOST RECENT
                                05740                   GOSUB ENTRY, THE "RETURN" CODE ELIMINATES THE
                                05760                   "GOSUB" ENTRY AND ALL "FOR" ENTRIES MADE AFTER
                                05780                   THE GOSUB ENTRY.
                                05820           NON-RUNTIME STUFF
                                05840                   THE CODE TO INPUT A LINE,CRUNCH IT,GIVE ERRORS,
                                05860                   FIND A SPECIFIC LINE IN THE PROGRAM,
                                05880                   PERFORM A "NEW","CLEAR", AND "LIST" ARE
                                05900                   ALL IN THIS AREA, GIVEN THE EXPLANATION OF
                                05920                   PROGRAM STORAGE GIVEN BELOW THESE ARE
                                05940                   ALL STRAIGHTFORWARD.
                                05980           NEWSTT
                                06000                   WHENEVER A STATEMENT FINISHES EXECUTION IT
                                06020                   DOES A "RET" WHICH TAKES
                                06040                   EXECUTION BACK TO NEWSTT, STATEMENTS THAT
                                06060                   CREATE OR LOOK AT SEMI-PERMANENT STACK ENTRIES
```



```
                                06080                   MUST GET RID OF THE RETURN ADDRESS OF NEWSTT AND
                                06100                   JMP TO NEWSTT WHEN DONE, NEWSTT ALWAYS
                                06120                   CHANGES THE FIRST CHARACTER AFTER THE STATEMENT
                                06140                   NAME BEFORE DISPATCHING, WHEN RETURNING
                                06160                   BACK TO NEWSTT THE ONLY THING THAT
                                06180                   MUST BE SET UP IS THE TEXT POINTER IN
                                06200                   (H,L), NEWSTT WILL CHECK TO MAKE SURE
                                06220                   (H,L) IS POINTING TO A STATEMENT TERMINATOR,
                                06240                   IF A STATEMENT SHOULDN'T BE PERFORMED UNLESS
                                06260                   IT IS PROPERLY FORMATTED (I.E. "NEW") IT CAN
                                06280                   SIMPLY DO A "RNZ" AFTER READING ALL OF
                                06300                   ITS ARGUMENTS, SINCE THE ZERO FLAG
                                06320                   BEING OFF INDICATES THERE IS NOT
                                06340                   A STATEMENT TERMINATOR NEWSTT WILL
                                06360                   DO THE JMP TO THE "SYNTAX ERROR"
                                06380                   ROUTINE, IF A STATEMENT SHOULD BE STARTED
                                06400                   OVER IT CAN DO LHLD TEMP,RET SINCE THE (H,L)
                                06420                   AT NEWSTT IS ALWAYS STORED IN TEMP, OF COURSE
                                06440                   CARE MUST BE TAKEN THAT NO ROUTINE
                                06460                   THAT SMASHES TEMP HAS BEEN CALLED,
                                06480                   THE "C CODE STORES TEMP IN OLDTXT AND CURLIN (THE
                                06500                   CURRENT LINE NUMBER) IN OLDLIN SINCE THE "C CHECK
                                06520                   IS MADE BEFORE THE STATEMENT POINTED TO IS
                                06540                   EXECUTED, "STOP" AND "END" STORE THE TEXT POINTER
                                06560                   IN (H,L) WHICH POINTS AT THEIR TERMINATING
                                06580                   CHARACTER IN OLDTXT.
                                06620           STATEMENT CODE
                                06640                   THE INDIVIDUAL STATEMENT CODE COMES
                                06660                   NEXT, THE APPROACH USED IN EXECUTING EACH
                                06680                   STATEMENT IS DOCUMENTED IN THE STATEMENT CODE
                                06700                   ITSELF.
                                06740           FRMEVL, THE FORMULA EVALUATOR
                                06760                   GIVEN AN (H,L) POINTING TO THE STARTING
                                06780                   CHARACTER OF A FORMULA FRMEVL
                                06800                   EVALUATES THE FORMULA AND LEAVES
                                06820                   THE VALUE IN THE FLOATING ACCUMULATOR (FAC),
                                06840                   (H,L) IS RETURNED POINTING TO THE FIRST CHARACTER
                                06860                   THAT COULD NOT BE INTERPRETED AS PART OF THE
                                06880                   FORMULA, THE ALGORITHM USES THE STACK
                                06900                   TO STORE TEMPORARY RESULTS:
                                06940                           0, PUT A DUMMY PRECEDENCE OF ZERO ON
                                06960                                   THE STACK.
                                06980                           1, READ LEXEME (CONSTANT,FUNCTION,
                                07000                                   VARIABLE,FORMULA IN PARENS)
                                07020                                   AND TAKE THE LAST PRECEDENCE VALUE
                                07040                                   OFF THE STACK,
                                07060                           2, SEE IF THE NEXT CHARACTER IS AN OPERATOR
                                07080                                   IF NOT,RETURN, THIS MAY CAUSE
                                07100                                   OPERATOR APPLICATION OR AN ACTUAL
                                07120                                   RETURN FROM FRMEVL.
```

```
                                        3, IF IT IS, SEE WHAT PRECEDENCE IT HAS
                                            AND COMPARE IT TO THE PRECEDENCE
                                            OF THE LAST OPERATOR ON THE STACK
                                        4, IF = OR LESS REMEMBER THE TEXT
                                            POINTER AT THE START OF THIS OPERATOR
                                            AND DO A RETURN TO CAUSE
                                            APPLICATION OF THE LAST OPERATOR,
                                            EVENTUALLY RETURN TO STEP 2
                                            BY RETURNING TO RETAOP.
                                        5, IF GREATER PUT THE LAST PRECEDENCE
                                            BACK ON, SAVE THE CURRENT
                                            TEMPORARY RESULT, OPERATOR ADDRESS
                                            AND PRECEDENCE AND RETURN TO STEP 1,

                                RELATIONAL OPERATORS ARE ALL HANDLED THROUGH
                                A COMMON ROUTINE, SPECIAL
                                CARE IS TAKEN TO DETECT TYPE MISMATCHES SUCH AS 3+"F"

                        EVAL -- THE ROUTINE TO READ A LEXEME
                                EVAL CHECKS FOR THE DIFFERENT TYPES OF
                                ENTITIES IT IS SUPPOSED TO DETECT,
                                LEADING PLUSES ARE IGNORED,
                                DIGITS AND "." CAUSE FIN (FLOATING INPUT)
                                TO BE CALLED, FUNCTION NAMES CAUSE THE
                                FORMULA INSIDE THE PARENTHESES TO BE EVALUATED
                                AND THE FUNCTION ROUTINE TO BE CALLED, VARIABLE
                                NAMES CAUSE PTRGET TO BE CALLED TO GET A POINTER
                                TO THE VALUE, AND THEN THE VALUE IS PUT INTO
                                THE FAC, AN OPEN PARENTHESIS CAUSES FRMEVL
                                TO BE CALLED (RECURSIVELY), AND THE ")" TO
                                BE CHECKED FOR, UNARY OPERATORS (NOT AND
                                NEGATION)  PUT THEIR PRECEDENCE ON THE STACK
                                AND ENTER FORMULA EVALUATION AT STEP 1, SO
                                THAT EVERYTHING UP TO AN OPERATOR GREATER THAN
                                THEIR PRECEDENCE OR THE END OF THE FORMULA
                                WILL BE EVALUATED, WHEN FRMEVL DOES A RETURN
                                BECAUSE IT SEES AN OPERATOR OF HIGHER PRECEDENCE
                                IT DOES NOT PASS THE TEXT POINTER IN (H,L), SO
                                AFTER THE UNARY OPERATION HAS BEEN PERFORMED
                                ON THE FAC THE TEXT POINTER MUST BE FETCHED FROM
                                A TEMPORARY LOCATION THAT FRMEVL USES AND
                                A RETURN BACK TO FRMEVL DONE,

                        DIMENSION AND VARIABLE SEARCHING
                                SPACE IS ALLOCATED FOR VARIABLES AS THEY ARE
                                ENCOUNTERED, THUS "DIM" STATEMENTS MUST BE
                                EXECUTED TO HAVE EFFECT, 6 BYTES ARE ALLOCATED
                                FOR EACH SIMPLE VARIABLE, WHETHER IT IS A STRING,
                                NUMBER OR USER DEFINED FUNCTION, THE FIRST TWO
                                BYTES GIVE THE NAME OF THE VARIABLE AND THE LAST FOUR
                                GIVE ITS VALUE, (VARTAB) GIVES THE FIRST LOCATION
                                WHERE A SIMPLE VARIABLE NAME IS FOUND AND (ARYTAB)
                                GIVES THE LOCATION TO STOP SEARCHING FOR SIMPLE
```



```
                                VARIABLES, A "FOR" ENTRY HAS A TEXT POINTER
                                AND A POINTER TO A VARIABLE VALUE SO NEITHER
                                THE PROGRAM OR THE SIMPLE VARIABLES CAN BE
                                MOVED WHILE THERE ARE ACTIVE "FOR" ENTRIES ON THE STACK,
                                USER DEFINED FUNCTION VALUES ALSO CONTAIN
                                POINTERS INTO SIMPLE VARIABLE SPACE SO NO USER-DEFINED
                                FUNCTION VALUES CAN BE RETAINED IF SIMPLE VARIABLES
                                ARE MOVED, ADDING A SIMPLE VARIABLE
                                ADDING SIX TO ARYTAB AND STREND, BLOCK TRANSFERING
                                THE ARRAY VARIABLES UP BY SIX AND MAKING SURE THE
                                NEW (STREND) IS NOT TO CLOSE TO THE STACK,
                                THIS MOVEMENT OF ARRAY VARIABLES MEANS
                                THAT NO POINTER TO AN ARRAY WILL STAY VALID WHEN
                                NEW SIMPLE VARIABLES CAN BE ENCOUNTERED, THIS IS
                                WHY ARRAY VARIABLES ARE NOT ALLOWED "FOR"
                                LOOP VARIABLES, SETING UP ANEW ARRAY VARIABLE
                                MERELY INVOLVES BUILDING THE DESCRIPTOR,
                                UPDATING STREND, AND MAKING SURE THERE IS
                                STILL ENOUGH ROOM BETWEEN STREND AND THE
                                STACK, WITHOUT MULTIPLE DIMENSIONS THE FORMAT
                                OF AN ARRAY VARIABLE IS SIMPLY:
                                        SECOND CHARACTER
                                        FIRST CHARACTER
                                        NUMBER OF BYTES USED BY VALUES
                                        VALUES
                                THE FORMAT WHEN MULTIPLY DIMENSIONED VARIABLES
                                ARE ALLOWED IS DESCRIBED IN THE "MULDIM" CODE,
                                PTRGET, THE ROUTINE WHICH RETURNS A POINTER
                                TO A VARIABLE VALUE, HAS TWO IMPORTANT FLAGS, ONE IS
                                "DIMFLG" WHICH INDICATED WHETHER "DIM" CALLED PTRGET
                                OR NOT, IF SO, NO PRIOR ENTRY FOR THE VARIABLE IN
                                QUESTION SHOULD BE FOUND, AND THE INDEX INDICATES
                                HOW MUCH SPACE TO SET ASIDE, SIMPLE VARIABLES CAN
                                BE "DIMENSIONED", BUT THE ONLY EFFECT WILL BE TO
                                SET ASIDE SPACE FOR THE VARIABLE IF IT HASN'T BEEN
                                ENCOUNTERED YET, THE OTHER IMPORTANT FLAG IS SUBFLG
                                WHICH INDICATES WHETHER A SUBSCRIPTED VARIABLE SHOULD BE
                                ALLOWED IN THE CURRENT CONTEXT, IF SUBFLG IS NON-ZERO
                                THE OPEN PARENTHESIS FOR A SUBSCRIPTED VARIABLE
                                WILL NOT BE SCANNED BY PTRGET, AND PTRGET WILL RETURN
                                WITH A TEXT POINTER POINTING TO THE "(", IF
                                THERE WAS ONE,
                        STRINGS
                                IN THE VARIABLE TABLE STRINGS ARE STORED JUST LIKE
                                NUMERIC VARIABLES, SIMPLE STRINGS HAVE FOUR VALUE
                                BYTES WHICH ARE INITIALIZED TO ALL ZEROS (WHICH
                                REPRESENTS THE NULL STRING), THE ONLY DIFFERENCE
                                IN HANDLING IS THAT WHEN PTRGET SEES A "$" AFTER THE
                                NAME OF A VARIABLE, PTRGET SETS VALTYP TO ONE AND TURNS
                                ON THE MSB (MOST-SIGNIFIGANT-BIT) OF THE VALUE OF
                                THE FIRST CHARACTER OF THE VARIABLE NAME,
                                MAKING THIS BIT ON IN THE NAME OF THE VARIABLE ENSURES
                                THAT THE SEARCH ROUTINE WILL NOT MATCH
```

```
                                09260                   'A' WITH 'AS' OR 'AS' WITH 'A', THE MEANING OF
                                09280                   THE FOUR VALUE BYTES ARE:
                                09300                           LOW
                                09320                                   LENGTH OF THE STRING
                                09340                                   UNUSED
                                09360                                   LOW 8 BITS
                                09380                                   HIGH 8 BITS  OF THE ADDRESS
                                09400                                           OF THE CHARACTERS IN THE
                                09420                                           STRING IF LENGTH.NE.0,
                                09440                                           MEANINGLESS OTHERWISE,
                                09460                           HIGH
                                09480                   THE VALUE OF A STRING VARIABLE (THESE 4 BYTES)
                                09500                   IS CALLED THE STRING DESCRIPTOR TO DISTINGUISH
                                09520                   IT FROM THE ACTUAL STRING DATA, WHENEVER A
                                09540                   STRING CONSTANT IS ENCOUNTERED IN A FORMULA OR AS
                                09560                   PART OF AN INPUT STRING, OR AS PART OF DATA, STRLIT
                                09580                   IS CALLED, CAUSING A DESCRIPTOR TO BE BUILT FOR
                                09600                   THE STRING, IF THE STRING CONSTANT IS IN BUF (WHICH
                                09620                   IT WILL BE IF THE STRING IS BEING "INPUT", OR THE
                                09640                   STRING IS PART OF SOME FORMULA IN A DIRECT STATEMENT)
                                09660                   THE VALUE IS COPIED INTO STRING SPACE SINCE BUF
                                09680                   IS ALWAYS CHANGING, "STRCPY" IS USED TO COPY
                                09700                   STRINGS.
                                09720
                                09740                   STRING FUNCTIONS AND THE ONE STRING OPERATOR "+"
                                09760                   ALWAYS RETURN THEIR VALUES IN STRING SPACE,
                                09780                   ASSIGNING A STRING A CONSTANT VALUE IN A PROGRAM
                                09800                   THROUGH A "READ" OR ASSIGNMENT STATEMENT
                                09820                   WILL NOT USE ANY STRING SPACE SINCE
                                09840                   THE STRING DESCRIPTOR  WILL POINT INTO THE
                                09860                   PROGRAM ITSELF, IN GENERAL, COPYING IS DONE
                                09880                   WHEN A STRING VALUE IS IN BUF, OR IT IS IN STRING
                                09900                   SPACE AND THERE IS AN ACTIVE POINTER TO IT,
                                09920                   THUS FS=GS WILL CAUSE COPYING IF GS HAS ITS
                                09940                   STRING DATA IN STRING SPACE, FS=CHR$(7)
                                09960                   WILL USE ONE BYTE OF STRING SPACE TO STORE THE
                                09980                   NEW ONE CHARACTER STRING CREATED BY "CHR$", BUT
                                10000                   THE ASSIGNMENT ITSELF WILL CAUSE NO COPYING SINCE
                                10020                   THE ONLY POINTER AT THE NEW STRING IS A
                                10040                   TEMPORARY DESCRIPTOR CREATED BY FRMEVL WHICH WILL
                                10060                   GO AWAY AS SOON AS THE ASSIGNMENT IS DONE,
                                10080                   IT IS THE NATURE OF GARBAGE COLLECTION THAT
                                10100                   DISALLOWS HAVING TWO STRING DESCRIPTORS POINT TO THE SAME
                                10120                   AREA IN STRING SPACE, STRING FUNCTIONS AND OPERATORS
                                10140                   MUST PROCEED AS FOLLOWS:
                                10160                           1) FIGURE OUT THE LENGTH OF THEIR RESULT
                                10180                           2) CALL GETSPA TO FIND SPACE FOR THEIR
                                10200                           RESULT, THE ARGUMENTS TO THE FUNCTION
                                10220                           OR OPERATOR MAY CHANGE SINCE GARBAGE COLLECTION
                                10240                           MAY BE INVOKED, THE ONLY THING THAT CAN
                                10260                           BE SAVED DURING THE CALL TO GETSPA IS A POINTER
                                10280                           TO THE DESCRIPTORS OF THE ARGUMENTS,
                                10300                           3) CONSTRUCT THE RESULT DESCRIPTOR IN DSCTMP,
```



```
                                10320                           GETSPA RETURNS THE LOCATION OF THE AVAILABLE
                                10340                           SPACE,
                                10360                           4) CREATE THE NEW VALUE BY COPYING PARTS
                                10380                           OF THE ARGUMENTS OR WHATEVER,
                                10400                           5) FREE UP THE ARGUMENTS BY CALLING FRETMP,
                                10420                           6) JUMP TO PUTNEW TO GET THE DESCRIPTOR IN
                                10440                           DSCTMP TRANSFERRED INTO A NEW STRING TEMPORARY,
                                10460
                                10480                   THE REASON FOR STRING TEMPORARIES IS THAT GARBAGE
                                10500                   COLLECTION HAS TO KNOW ABOUT ALL ACTIVE STRING DESCRIPTORS
                                10520                   SO IT KNOWS WHAT IS AND ISN'T IN USE, STRING TEMPORARIES ARE
                                10540                   USED TO STORE THE DESCRIPTORS OF STRING EXPRESSIONS,
                                10560
                                10580                   INSTEAD OF HAVING AN ACTUAL VALUE STORED IN THE
                                10600                   FAC, AND HAVING THE VALUE OF A TEMPORARY RESULT
                                10620                   BEING SAVED ON THE STACK, AS HAPPENS WITH NUMERIC
                                10640                   VARIABLES, STRINGS HAVE THE POINTER TO A STRING DESCRIPTOR
                                10660                   STORED IN THE FAC, AND IT IS THIS POINTER
                                10680                   THAT GETS SAVED ON THE STACK BY FORMULA EVALUATION,
                                10700                   STRING FUNCTIONS CANNOT FREE THEIR ARGUMENTS UP RIGHT
                                10720                   AWAY SINCE GETSPA MAY FORCE
                                10740                   GARBAGE COLLECTION AND THE ARGUMENT STRINGS
                                10760                   MAY BE OVER-WRITTEN SINCE GARBAGE COLLECTION
                                10780                   WILL NOT BE ABLE TO FIND AN ACTIVE POINTER TO
                                10800                   THEM, FUNCTION AND OPERATOR RESULTS ARE BUILT IN
                                10820                   DSCTMP SINCE STRING TEMPORARIES ARE ALLOCATED
                                10840                   (PUTNEW) AND DEALLOATED (FRETMP) IN A FIFO ORDERING
                                10860                   (I.E, A STACK) SO THE NEW TEMPORARY CANNOT
                                10880                   BE SET UP UNTIL THE OLD ONE(S) ARE FREED, TRYING
                                10900                   TO BUILD A RESULT IN A TEMPORARY AFTER
                                10920                   FREEING UP THE ARGUMENT TEMPORARIES COULD RESULT
                                10940                   IN ONE OF THE ARGUMENT TEMPORARIES BEING OVERWRITTEN
                                10960                   TOO SOON BY THE NEW RESULT,
                                11000
                                11020                   STRING SPACE IS ALLOCATED AT THE VERY TOP
                                11040                   OF MEMORY, MEMSIZ POINTS BEYOND THE LAST LOCATION OF
                                11060                   STRING SPACE, STRING ARE STORED IN HIGH LOCATIONS
                                11080                   FIRST, WHENEVER STRING SPACE IS ALLOCATED (GETSPA)
                                11100                   FRETOP, WHICH IS INITIALIZED TO (MEMSIZ), IS UPDATED
                                11120                   TO GIVE THE HIGHEST LOCATION IN STRING SPACE
                                11140                   THAT IS NOT IN USE, THE RESULT IS THAT
                                11160                   FRETOP GETS SMALLER AND SMALLER, UNTIL SOME
                                11180                   ALLOCATION WOULD MAKE (FRETOP) LESS THAN OR EQUAL TO
                                11200                   (STKTOP), THIS MEANS STRING SPACE HAS RUN INTO THE
                                11220                   STACK AND THAT GARBAGE COLLECTION MUST BE CALLED,
                                11240
                                11260                   GARBAGE COLLECTION:
                                11280                           0, MINPTR=(STKTOP) (FRETOP)=(MEMSIZ)
                                11300                           1, REMMIN=0
                                11320                           2, FOR EACH STRING DESCRIPTOR
                                11340                           (TEMPORARIES, SIMPLE STRINGS, STRING ARRAYS)
                                11360                           IF THE STRING IS NOT NULL AND ITS POINTER IS
                                                                .GT.MINPTR AND .LT.FRETOP,
```

```
                                        MINPTR=THIS STRING DESCRIPTORS POINTER
                                        REMMIN=POINTER AT THIS STRING DESCRIPTOR
                                        END
                                        3, IF REMMIN,NE,0 (WE FOUND AN UNCOLLECTED STRING)
                                        BLOCK TRANSFER THE STRING DATA POINTED
                                        TO IN THE STRING DESCRIPTOR POINTED TO BY REMMIN
                                        SO THAT THE LAST BYTE OF STRING DATA IS AT
                                        (FRETOP), UPDATE FRETOP SO THAT IT
                                        POINTS TO THE LOCATION JUST BELOW THE ONE
                                        THE STRING DATA WAS MOVED INTO, UPDATE
                                        THE POINTER IN THE DESCRIPTOR SO IT POINTS
                                        TO THE NEW LOCATION OF THE STRING DATA,
                                        GO TO STEP 1,

                                AFTER CALLING GARBAGE COLLECTION GETSPA AGAIN CHECKS
                                TO SEE IF [A] CHARACTERS ARE AVAILABLE BETWEEN
                                (STKTOP) AND (FRETOP) , IF NOT AN "OUT OF STRING"
                                ERROR IS INVOKED,

                        MATH PACKAGE
                                THE MATH PACKAGE CONTAINS FLOATING INPUT (FIN),
                                FLOATING OUTPUT (FOUT) FLOATING COMPARE (FCOMP)
                                ... AND ALL THE NUMERIC OPERATORS AND FUNCTIONS,
                                THE FORMATS,CONVENTIONS AND ENTRY POINTS ARE ALL
                                DESCRIBED IN THE MATH PACKAGE ITSELF,

                        INIT -- THE INITIALIZATION ROUTINE
                                INITIALIZATION FIRST LOOKS AT THE SWITCH REGISTER
                                TO SEE WHAT TYPE OF I/O SHOULD BE DONE,
                                ANY NON-STANDARD I/O CAUSES LOCATIONS IN BASIC
                                TO BE CHANGED, THEN THE AMOUNT OF MEMORY,
                                TERMINAL WIDTH,AND WHICH FUNCTIONS TO BE RETAINED
                                ARE ASCERTAINED FROM THE USER, A ZERO IS PUT DOWN
                                AT THE FIRST LOCATION NOT USED BY THE MATH-PACKAGE
                                AND TXTTAB IS SET UP TO POINT AT THE NEXT LOCATION,
                                THIS DETERMINES WHERE PROGRAM STORAGE WILL START, THE
                                HIGHEST MEMORY LOCATION MINUS THE AMOUNT OF DEFAULTED
                                STRING SPACE (50) GIVES THE FIRST LOCATION USED BY THE
                                STACK, SPECIAL CHECKS ARE MADE TO MAKE SURE
                                ALL QUESTIONS IN INIT ARE ANSWERED REASONABLY, SINCE
                                ONCE INIT FINISHES THE LOCATIONS IT USES ARE
                                USED FOR PROGRAM STORAGE, THE LAST THING INIT DOES IS
                                CHANGE LOCATION ZERO TO BE A JUMP TO READY INSTEAD
                                OF INIT, ONCE THIS IS DONE THERE IS NO WAY TO RESTART
                                INIT,

                        STORAGE
                                                        A ZERO,
                                (TXTTAB)                POINTER TO NEXT LINE'S POINTER
                                                        LINE # OF THIS LINE (2 BYTES)
                                                        CHARACTERS ON THIS LINE
                                                        ZERO
                                                        POINTER AT NEXT LINE'S POINTER
```



```
                                                        (POINTED TO BY THE ABOVE POINTER)
                                                ... REPEATS ...
                                LAST LINE:      POINTER AT ZERO POINTER
                                                LINE # OF THIS LINE
                                                CHARACTERS ON THIS LINE
                                                ZERO
                                                DOUBLE ZERO (POINTED TO BY THE ABOVE POINTER)
                                (VARTAB)        SIMPLE VARIABLES, 6 BYTES PER VALUE,
                                                2 BYTES GIVE THE NAME, 4 BYTES THE VALUE
                                                ... REPEATS ...
                                (ARYTAB)        ARRAY VARIABLES, 2 BYTES NAME, 2 BYTE
                                                LENGTH, VALUE (EXTRA IF MULDIM ON)
                                                ... REPEATS ...
                                (STREND)        FREE SPACE
                                                ... REPEATS ...
                                                MOST RECENT STACK ENTRY
                                                ... REPEATS ...
                                (STKTOP)        FIRST STACK ENTRY
                                                FREE STRING SPACE
                                                ... REPEATS ...
                                (FRETOP)        STRING SPACE IN USE
                                                ... REPEATS ...
                                (MEMSIZ)        HIGHEST MACHINE LOCATION
                                                UNUSED EXCEPT BY THE VAL FUNCTION,

                        HIGH LOCATIONS

                        *
                        PAGE
```

```
                                        13000  SUBTTL  RST ROUTINES
                                        13020  RELOC 0
000000' 001000  000363                  13040  START:  DI                      ;DISENABLE INTERRUPTS
000001' 001000  000303                  13060          JMP     INIT             ;INIT IS THE INTIALIZE ROUTINE
000002' 000000  000000*
000003' 000000  000000
                                        13080                                   ;IT SETS UP CERTAIN
                                        13100                                   ;LOCATIONS DELETES FUNCTIONS IF
                                        13120                                   ;DESIRED AND
                                        13140                                   ;CHANGES THIS TO JMP READY
                                        13160  IFN     LENGTH-2,<
                                        13180          ADR(DEINT)               ;STORE HERE THE ROUTINE
                                        13200                                   ;TO TURN THE FAC INTO
                                        13220                                   ;A TWO-BYTE SIGNED INTEGER
                                        13240          ADR(GIVABF)>             ;STORE HERE THE ADDRESS
                                        13260                                   ;OF THE ROUTINE TO CONVERT (A,B)
                                        13280                                   ;TO A FLOATING POINT NUMBER IN THE FAC
                                        13300  IFE     LENGTH-2,<
000004' 000000  000000*                 13320          ADR(FRCINT)              ;TURN FAC INTO AN INTEGER IN (H,L)
000005' 000000  000000'
000006' 000000  000000*                 13340          ADR(MAKINT)>             ;TURN (H,L) INTO A VALUE IN THE FAC
000007' 000000  000004'
                                        13360                                   ;SET VALTYP FOR INTEGER
000010'                                 13380  RELOC 8
                                        13400  ;
                                        13420  ; SYNCHR LOOKS AT THE CURRENT CHARACTER TO MAKE SURE IT
                                        13440  ; IS A SPECIFIC THING (CONTAINED IN THE LOCATION AFTER THE CALL)
                                        13460  ; IF NOT IT CALLS THE 'SYNTAX ERROR' ROUTINE, OTHERWISE IT GOBBLES
                                        13480  ; THE NEXT CHARACTER AND RETURNS, (BY FALLING INTO CHRGET)
                                        13500  ;
                                        13520  ; ALL REGISTERS ARE PRESERVED EXCEPT (A)=NEW CHAR
                                        13540  ; AND (H,L) ENDS UP POINTING AT THE CHARACTER AFTER THE ONE
                                        13560  ; WHICH WAS CHECKED.
                                        13580  ;
000010' 001000  000176                  13600          MOV     A,M              ;GET THE CURRENT CHARACTER
000011' 001000  000343                  13620          XTHL                     ;GET CALL ADDRESS INTO (H,L)
                                        13640                                   ;PUT TEXT POINTER ON STACK
000012' 001000  000276                  13660          CMP     M                ;SEE IF (A) =CURRENT CHARACTER
                                        13680                                   ;IS THE RIGHT THING,
000013' 001000  000043                  13700          INX     H                ;FIX RETURN ADDRESS,
000014' 001000  000343                  13720          XTHL                     ;PUT RETURN ADDRESS BACK AND RESTORE
                                        13740                                   ;THE TEXT POINTER,
000015' 001000  000302                  13760          JNZ     SNERR            ;IF THE CHARACTER WASN'T RIGHT CALL
000016' 000000  002027'
000017' 000000  000000
                                        13780                                   ;THE 'SYNTAX ERROR' ROUTINE,
                                        13800                                   ;OTHERWISE FALL THROUGH
                                        13820                                   ;AND GET ANOTHER CHARACTER.
                                        13840  ;
                                        13860  ; CHRGET, USING (H,L) AS THE CURRENT TEXT POINTER FETCHES
                                        13880  ; A NEW CHARACTER INTO (A) AFTER INCREMENTING (H,L)
                                        13900  ; AND SETS CONDITION CODES ACCORDING TO WHATS IN (A)
                                        13920  ;       C=      NUMERIC    ("0" THROUGH "9")
```



```
                                        13940  ;       Z=      ":" OR END-OF-LINE (A 0)
                                        13960  ;
                                        13980  ; ALL REGISTERS SAVED EXCEPT (A)=NEW CHAR
                                        14000  ; (H,L)=(H,L)+1
                                        14020  ;
000020'                                 14040  RELOC 16
                                        14060  IFE     LENGTH,<CHRGTR:>
000020' 001000  000043                  14080          INX     H                ;UPDATE THE TEXT POINTER
000021' 001000  000176                  14100          MOV     A,M              ;GET NEW CHARACTER
000022' 001000  000376                  14120          CPI     ":"              ;MAKE ":" HAVE ZERO ON AND
000023' 000000  000072
                                        14140                                   ;CARRY OFF
                                        14160                                   ;ALL ALPHABETICS & RESERVED
                                        14180                                   ;WORDS GET ZERO & CARRY OFF
000024' 001000  000320                  14200          RNC                      ;,GT,":" GO BACK
000025' 001000  000303                  14220          JMP     CHRCON           ;NO ROOM FOR WHOLE ROUTINE
000026' 000000  003433'
000027' 000000  000016'
                                        14240  ;
                                        14260  ; THIS RST ROUTINE OUTPUTS THE CHARACTER IN (A) USING PRTFLG (LPT OR TTY)
                                        14280  ; CNTRFL (SUPRESS OUTPUT OR NOT), TTYPOS (PRINT HEAD POSITION),
                                        14300  ; TIMING ETC,. NO REGISTERS OR CONDITION CODES ARE CHANGED.
                                        14320  ;
000030'                                 14340  RELOC 24
000030' 001000  000365                  14360  OUTDO:  PUSH    PSW
                                        14380  IFN     CONTRW,<
000031' 001000  000072                  14400          LDA     CNTWFL           ;GET SUPPRESS FLAG
000032' 000000  001541'
000033' 000000  000026'
000034' 001000  000267                  14420          ORA     A>               ;SEE IF IT IS SET
                                        14440  IFE     LENGTH!CONTRW!LPTSW,<
                                        14460          LDA     TTYPOS>          ;USE RST BYTES, (A)=TTYPOS
000035' 001000  000303                  14480          JMP     OUTCON
000036' 000000  003065'
000037' 000000  000032'
                                        14500  ;
                                        14520  ; COMPAR COMPARES (H,L) WITH (D,E) UNSIGNED
                                        14540  ;
                                        14560  ;               (H,L) LESS THAN (D,E) SET CARRY
                                        14580  ;               (H,L) = (D,E) SET ZERO
                                        14600  ;
                                        14620  ; (A) IS THE ONLY REGISTER USED
                                        14640  ;
000040'                                 14660  RELOC 32
000040' 001000  000174                  14680          MOV     A,H
000041' 001000  000222                  14700          SUB     D
000042' 001000  000300                  14720          RNZ
000043' 001000  000175                  14740          MOV     A,L
000044' 001000  000223                  14760          SUB     E
000045' 001000  000311                  14780          RET
                                        14800
000046' 000000  000001                  14820  NULCNT: 1                        ;STORE HERE THE NUMBER OF NULLS
                                        14840                                   ;TO PRINT AFTER CRLF
```

```
000047'                         14860   TTYPOS: BLOCK   1               ;STORE TERMINAL POSITION HERE
                                14880   ;
                                14900   ;THE FSIGN  RST RETURNS A=+1 IF FAC IS LESS THAN 0
                                14920   ;       A=0 IF FAC=0
                                14940   ;       A=1 IF FAC GREATER THAN ZERO
                                14960   ; THE CONDITION CODES REFLECT THE VALUE OF [A]
                                14980   ; AND NO OTHER REGISTERS ARE MODIFIED.
                                15000   ; THIS WORKS ONLY WHEN THE FAC IS A SINGLE OR DOUBLE PRECISION NUMBER
                                15020   ; THE 'VSIGN' ROUTINE IS MORE GENERAL SINCE
                                15040   ; IT WILL TAKE THE SIGN OF INTEGERS AS WELL
                                15050   ; AND GIVES "TMERR" ON STRINGS.
                                15060   ;
000050'                         15080   RELOC   40
000050' 001000  000072          15100   SIGN:   LDA     FAC
000051' 000000  001642'
000053' 000000  000036'
000053' 001000  000267          15120           ORA     A
000054' 001000  000302          15140           JNZ     SIGNC
000055' 000000  000000*
000056' 000000  000051'
000057' 001000  000311          15160           RET
                                15180   ;
                                15200   ; THIS IS THE PUSHM RST
                                15220   ; EFFECT IS:
                                15240   ;       MOV     C,M
                                15260   ;       INX     H
                                15280   ;       MOV     B,M
                                15300   ;       INX     H
                                15320   ;       PUSH    B
                                15340   ; DIFFICULTY COMES IN BECAUSE OF THE
                                15360   ; RETURN ADDRESS.
                                15380   ;
000060'                         15400   RELOC   48
000060' 001000  000343          15420           XTHL                            ;SWITCH [H,L] AND RETURN ADDRESS
000061' 001000  000042          15440           SHLD    PUSHMA+1                ;FIXUP JUMP TO PLACE TO GO
000062' 000000  000101'
000063' 000000  000055'
000064' 001000  000341          15460           POP     H                       ;REGAIN [H,L]
                                15480   IFN     LENGTH,<
000065' 001000  000303          15500           JMP     $CODE+59                ;IN 8K ALLOW USER TO HAVE RST 7
000066' 000000  000073'
000067' 000000  000662'
                                15520                                           ;FOR INTERRUPT TRAPPING
000070'                         15540   RELOC   56
000070' 001000  000311          15560           RET                             ;INITIALLY NO INTERRUPT
                                15580                                           ;ROUTINE
000071' 001000  000000          15600           NOP
000072' 001000  000000          15620           NOP>
000073' 001000  000116          15640           MOV     C,M                     ;GRAB FROM MEMORY
000074' 001000  000043          15660           INX     H
000075' 001000  000106          15680           MOV     B,M
000076' 001000  000043          15700           INX     H
000077' 001000  000305          15720           PUSH    B                       ;PUSH [B,C] ONTO THE STACK
```



```
                                15740                                           ;SINCE IT CONTAINS [M]
000100' 001000  000303          15760   PUSHMA: JMP     PUSHMA                  ;RETURN  ADDRESS STORED HERE
000101' 000000  000100'
000102' 000000  000066'
                                15800   PAGE
```

```
                                                15820   SUBTTL DISPATCH TABLES,RESERVED WORDS, ERROR TEXT,,, ALL CONSTANT
                                                15840
000103' 000000  000000*                         15860   FUNDSP: ADR(SGN)
000104' 000000  000101'                         15880   IFN     LENGTH-2,<
                                                15900           ADR(INT)>
                                                15920   IFE     LENGTH-2,<
000105' 000000  000000*                         15940           ADR(VINT)>
000106' 000000  000103'
000107' 000000  000000*                         15960           ADR(ABS)
000110' 000000  000105'
000111' 000000  010776'                         15980   USRLOC: ADR(ILLFUN)             ;INITIALLY NO USER ROUTINE
000112' 000000  000107'
000113' 000000  007337'                         16000   IFN     LENGTH,<ADR(FRE)
000114' 000000  000111'
000115' 000000  010712'                         16020           ADR(FNINP)
000116' 000000  000113'
                                                16040   IFN     LPTSW,<ADR(LPOS)>
000117' 000000  007406'                         16060           ADR(POS)>
000120' 000000  000115'
000121' 000000  000000*                         16080   SQRFIX: ADR(SQR)
000122' 000000  000117'
000123' 000000  000000*                         16100   RNDFIX: ADR(RND)
000124' 000000  000121'
                                                16120   IFN     EXTFNC,<
000125' 000000  000000*                         16140           ADR(LOG)
000126' 000000  000123'
000127' 000000  000000*                         16160           ADR(EXP)
000130' 000000  000125'
000131' 000000  000000*                         16180   COSFIX: ADR(COS)>
000132' 000000  000127'
000133' 000000  000000*                         16200   SINFIX: ADR(SIN)
000134' 000000  000131'
                                                16220   IFN     EXTFNC,<
000135' 000000  000000*                         16240   TANFIX: ADR(TAN)
000136' 000000  000133'
000137' 000000  000000*                         16260   ATNFIX: ADR(ATN)>
000140' 000000  000135'
                                                16280   IFN     LENGTH,<
000141' 000000  011314'                         16300           ADR(PEEK)>
000142' 000000  000137'
                                                16320   IFN     DSKFUN,<ADR(DSK1$)>
                                                16340   IFN     STRING,<
000143' 000000  010501'                         16360           ADR(LEN)
000144' 000000  000141'
000145' 000000  007564'                         16380           ADR(STR$)
000146' 000000  000143'
000147' 000000  011042'                         16400           ADR(VAL)
000150' 000000  000145'
000151' 000000  010515'                         16420           ADR(ASC)
000152' 000000  000147'
000153' 000000  010532'                         16440           ADR(CHR$)
000154' 000000  000151'
```



```
000155' 000000  010552'                         16460           ADR(LEFT$)
000156' 000000  000153'
000157' 000000  010631'                         16480           ADR(RIGHT$)
000160' 000000  000155'
000161' 000000  010643'                         16500           ADR(MID$)>
000162' 000000  000157'

                                                16540   DEFINE ADRP(X),<ADR(X)>
                                                16560   IFE     LENGTH-2,<
                                                16580   DEFINE ADRP(X),<>>
000163' 000000  000171                          16600   OPTAB:  121                     ;OPERATOR TABLE CONTAINS
                                                16620                                   ;PRECEDENCE FOLLOWED BY
                                                16640                                   ;THE ROUTINE ADDRESS
                                                16660           ADRP(FADDT)
000164' 000000  000171                          16680           121
                                                16700           ADRP(FSUBT)
000165' 000000  000173                          16720           123
                                                16740           ADRP(FMULTT)
000166' 000000  000173                          16760           123
                                                16780           ADRP(FDIVT)
000167' 000000  000177                          16800   IFN     EXTFNC,<127
                                                16820           ADRP(FPWRT)>
                                                16840   IFN     LENGTH,<
000170' 000000  000120                          16860           80
                                                16880           ADRP(AND)
000171' 000000  000106                          16900           70
                                                16920           ADRP(OR)>

                                                16960   ;
                                                16980   ; TOKENS FOR RESERVED WORDS ALWAYS HAVE THE MOST
                                                17000   ; SIGNIFIGANT BIT ON
                                                17020   ; THE LIST OF RESERVED WORDS
                                                17040   ;
                        000177                  17060   Q=128-1
                                                17080   DEFINE  DCI(A),<Q=Q+1
                                                17090           XLIST
                                                17100           DC(A)
                                                17110           LIST>
                        000200                  17140           ENDTK==Q
                        000201                  17160           FORTK==Q
                        000203                  17240           DATATK==Q
                        000210                  17360           GOTOTK==Q
                        000212                  17420           IFTK==Q
                        000214                  17480           GOSUTK=Q
                        000216                  17540           REMTK=Q
                                                17660   IFE     LENGTH-2,<
                        000220                  17700           ELSETK==Q
                                                17760   IFN     DSKFUN,<DCI"DSKUS">
                                                17800   IFN     LPTSW,<DCI"LPRINT">
                                                17820   IFN     LENGTH,<
                        000231                  17880           PRINTK==Q
                                                17940   IFE     REALIO,<
                                                17960           DCI"DDT">
```

```
1001                                    18000   IFN     LPTSW,<OCI"LLIST>
1002                                    18060   IFN     CASSW,<OCI"CLOAD"
1003                                    18080           OCI"CSAVE">
1004                                    18100   IFN     CONSSW,<OCI"CONSOLE">
1005                    000237          18140           SCRATK=Q
1006                                    18160   ; END OF COMMAND LIST
1007    000371' 000000  000124          18180           "T"
1008    000372' 000000  000101          18200           "A"
1009    000373' 000000  000102          18220           "B"
1010    000374' 000000  000250          18240           "("+128
1011                    000240          18260           Q=Q+1
1012                    000240          18280           TABTK=Q
1013                    000241          18320           TOTK=:Q
1014                                    18340   IFN     LENGTH,<
1015    000377' 000000  000123          18360           "S"
1016    000400' 000000  000120          18380           "P"
1017    000401' 000000  000103          18400           "C"
1018    000402' 000000  000250          18420           "("+128                 ;MACRO DOESNT LIKE ('S IN ARGUMENTS
1019                    000242          18440           Q=Q+1
1020                    000242          18460           SPCTK=:Q>
1021                    000243          18500           FNTK=:Q>
1022                    000244          18540           USINTK=:Q>
1023                    000245          18580           THENTK=Q
1024                                    18600   IFN     LENGTH,<
1025                    000246          18640           NOTTK=:Q>
1026                    000247          18680           STEPTK=Q
1027                    000250          18720           PLUSTK=Q
1028                    000251          18760           MINUTK=Q
1029                    000007          18800           LSTOPK=:Q+1-PLUSTK      ;CRUNCH # OF HIGHEST OP+1-PLUSTK
1030    000437' 000000  000276          18900           190                     ;A GREATER THAN SIGN
1031                    000257          18920           Q=Q+1
1032                    000257          18940           GREATK=Q
1033                    000260          18980           EQULTK=Q
1034    000441' 000000  000274          19000           188
1035                    000261          19020           Q=Q+1                   ;A LESS THAN SIGN
1036                    000261          19040           LESSTK=Q
1037                                    19060   ;
1038                                    19080   ; NOTE DANGER OF ONE RESERVED WORD BEING A PART
1039                                    19100   ; OF ANOTHER
1040                                    19120   ; IE , , IF 2 GREATER THAN F OR T=5 THEN,,,
1041                                    19140   ; WILL NOT WORK!!! SINCE "FOR" WILL BE CRUNCHED!!
1042                                    19160   ; IN ANY CASE MAKE SURE THE SMALLER WORD APPEARS
1043                                    19180   ; SECOND IN THE RESERVED WORD TABLE ("INP" AND "INPUT")
1044                                    19190   ; ANOTHER EXAPMLE: IF T OR Q THEN ,,, "TO" IS CRUNCHED
1045                                    19200   ;
1046                    000266          19240           ONEFUN=Q
1047                                    19360   IFN     LPTSW,<OCI"LPOS">
1048                    000271          19420           SQRTK=:Q
1049                                    19460   IFN     EXTFNC,<
1050                    000300          19600           ATNTK=:Q>
1051                                    19620   IFN     LENGTH,<
1052                                    19660   IFN     DSKFUN,<OCI"DSKIS">
1053                                    19680   IFN     STRING,<
```



```
1054                    000306          19800   LASNUM==Q                       ;NUMBER OF LAST FUNCTION
1055                                    19820                                   ;THAT TAKES ONE ARG
1056    000563' 000000  000000          19900           0                       ;MARKS END OF RESERVED WORD LIST
1057
1058    000564' 000000  003474'         19940   STMDSP: ADR(END)
1059    000565' 000000  000161'
1060    000566' 000000  003154'         19960           ADR(FOR)
1061    000567' 000000  000564'
1062    000570' 000000  005225'         19980           ADR(NEXT)
1063    000571' 000000  000566'
1064    000572' 000000  004072'         20000           ADR(DATA)
1065    000573' 000000  000570'
1066    000574' 000000  004711'         20020           ADR(INPUT)
1067    000575' 000000  000572'
1068    000576' 000000  006500'         20040           ADR(DIM)
1069    000577' 000000  000574'
1070    000600' 000000  004760'         20060           ADR(READ)
1071    000601' 000000  000576'
1072    000602' 000000  004131'         20080           ADR(LET)
1073    000603' 000000  000600'
1074    000604' 000000  004010'         20100           ADR(GOTO)
1075    000605' 000000  000602'
1076    000606' 000000  003754'         20120           ADR(RUN)
1077    000607' 000000  000604'
1078    000610' 000000  004325'         20140           ADR(IF)
1079    000611' 000000  000606'
1080    000612' 000000  003446'         20160           ADR(RESTORE)
1081    000613' 000000  000610'
1082    000614' 000000  003770'         20180           ADR(GOSUB)
1083    000615' 000000  000612'
1084    000616' 000000  004044'         20200           ADR(RETURN)
1085    000617' 000000  000614'
1086    000620' 000000  004074'         20220           ADR(REM)
1087    000621' 000000  000616'
1088    000622' 000000  003472'         20240           ADR(STOP)
1089    000623' 000000  000620'
1090                                    20260   IFE     LENGTH=2,<
1091    000624' 000000  004074'         20280           ADR(ELSE)
1092    000625' 000000  000622'
1093    000626' 000000  003604'         20300           ADR(TON)
1094    000627' 000000  000624'
1095    000630' 000000  003605'         20320           ADR(TOFF)
1096    000631' 000000  000626'
1097    000632' 000000  000000*         20340           ADR(EDIT)>
1098    000633' 000000  000630'
1099    000634' 000000  010725'         20360   IFN     LENGTH,<ADR(FNOUT)
1100    000635' 000000  000632'
1101    000636' 000000  004271'         20380           ADR(ONGOTO)
1102    000637' 000000  000634'
1103    000640' 000000  005566'         20400           ADR(NULL)
1104    000641' 000000  000636'
1105    000642' 000000  010733'         20420           ADR(FNWAIT)>
1106    000643' 000000  000640'
```

```
                                        20440   IFN     DSKFUN,<ADR(DSKOS)>
                                        20460   IFN     LPTSW,<ADR(LPRINT)>
                                        20480   IFN     LENGTH,<
000644' 000000  011323'                 20500           ADR(POKE)>
000645' 000000  000642'
000646' 000000  004411'                 20520           ADR(PRINT)
000647' 000000  000644'
000650' 000000  007411'                 20540   IFN     FUNCTS,<ADR(DEF)>
000651' 000000  000646'

000652' 000000  003542'                 20580   IFN     LENGTH,<ADR(CONT)>
000653' 000000  000650'
                                        20600   IFE     REALIO,<ADR(DDT)>
000654' 000000  011672'                 20620           ADR(LIST)
000655' 000000  000652'

                                        20640   IFN     LPTSW,<ADR(LLIST)>
000656' 000000  011245'                 20660   IFE     LENGTH-2,<ADR(DELETE)>
000657' 000000  000654'
000660' 000000  003703'                 20680           ADR(CLEAR)
000661' 000000  000656'

                                        20700   IFN     CASSW,<ADR(CLOAD)
                                        20720           ADR(CSAVE)>
                                        20740   IFN     CONSSW,<ADR(CONSOLE)>
000662' 000000  002421'                 20760           ADR(SCRATH)
000663' 000000  000660'

                                        20800   IFE     LENGTH-2,<
000664' 000000  000000*                 20820   FRCTBL: ADR(FRCDBL)
000665' 000000  000662'
000666' 000000  000000*                 20840           ADR(FRCINT)
000667' 000000  000664'
000670' 000000  000000*                 20860           ADR(FRCSNG)
000671' 000000  000666'

                                        20880   ;
                                        20900   ; THESE TABLES ARE USED AFTER THE DECISION HAS BEEN MADE
                                        20920   ; TO APPLY AN OPERATOR AND ALL THE NECESSARY CONVERSION HAS
                                        20940   ; BEEN DONE TO MATCH THE TWO ARGUMENT TYPES (APPLOP)
                                        20960   ;
000672' 000000  000000*                 20980   DBLDSP: ADR(DADD)               ;DOUBLE PRECISION ROUTINES
000673' 000000  000670'
000674' 000000  000000*                 21000           ADR(DSUB)
000675' 000000  000672'
000676' 000000  000000*                 21020           ADR(DMULT)
000677' 000000  000674'
000700' 000000  000000*                 21040           ADR(DDIV)
000701' 000000  000676'
000702' 000000  000000*                 21060           ADR(DCOMP)
000703' 000000  000700'
000704' 000000  000000*                 21080   SNGDSP: ADR(FADD)               ;SINGLE PRECISION ROUTINES
000705' 000000  000702'
000706' 000000  000000*                 21100           ADR(FSUB)
000707' 000000  000704'
000710' 000000  000000*                 21120           ADR(FMULT)
```



```
000711' 000000  000706'
000712' 000000  000000*                 21140           ADR(FDIV)
000713' 000000  000710'
000714' 000000  000000*                 21160           ADR(FCOMP)
000715' 000000  000712'
000716' 000000  000000*                 21180   INTDSP: ADR(IADD)               ;INTEGER ROUTINES
000717' 000000  000714'
000720' 000000  000000*                 21200           ADR(ISUB)
000721' 000000  000716'
000722' 000000  000000*                 21220           ADR(IMULT)
000723' 000000  000720'
000724' 000000  000000*                 21240           ADR(IDIV)
000725' 000000  000722'
000726' 000000  000000*                 21260           ADR(ICOMP)>
000727' 000000  000724'

                777777  777776          21320   Q=-2
                                        21340   DEFINE  DCL(X),<>
                                        21360   DEFINE  DCE(X),<Q=Q+2
                                        21370           XLIST
                                        21380           DC(X)
                                        21390           LIST>

000730'                                 21420   ERRTAB:
                                        21440   IFE     LENGTH-2,<
000730' 000000  000000                  21460           0
                000000                  21480           Q=0
                                        21500           DEFINE  DCE(X),<>
                                        21520           DEFINE  DCL(X),<
                                        21540           Q=Q+1
                                        21560           DC(X)
                                        21580           0>>
                                        21600           DCE"NF"
000731' 000000  000116                  21620           DCL"NEXT WITHOUT FOR"
000732' 000000  000105
000733' 000000  000130
000734' 000000  000124
000735' 000000  000040
000736' 000000  000127
000737' 000000  000111
000740' 000000  000124
000741' 000000  000110
000742' 000000  000117
000743' 000000  000125
000744' 000000  000124
000745' 000000  000040
000746' 000000  000106
000747' 000000  000117
000750' 000000  000122
000750' 000000  000322
000751' 000000  000000
                000001          21640           ERRNF==Q
```

```
1213                                        21660           DCE"SN"
1214  000752'  000000  000123              21680           DCL"SYNTAX ERROR"
1215  000753'  000000  000131
1216  000754'  000000  000116
1217  000755'  000000  000124
1218  000756'  000000  000101
1219  000757'  000000  000130
1220  000760'  000000  000040
1221  000761'  000000  000105
1222  000762'  000000  000122
1223  000763'  000000  000122
1224  000764'  000000  000117
1225  000765'  000000  000122
1226  000765'  000000  000302
1227  000766'  000000  000000
1228                    000002          21700           ERRSN==Q
1229                                    21720           DCE"RG"
1230  000767'  000000  000122          21740           DCL"RETURN WITHOUT GOSUB"
1231  000770'  000000  000105
1232  000771'  000000  000124
1233  000772'  000000  000125
1234  000773'  000000  000122
1235  000774'  000000  000116
1236  000775'  000000  000040
1237  000776'  000000  000127
1238  000777'  000000  000111
1239  001000'  000000  000124
1240  001001'  000000  000110
1241  001002'  000000  000117
1242  001003'  000000  000125
1243  001004'  000000  000124
1244  001005'  000000  000040
1245  001006'  000000  000107
1246  001007'  000000  000117
1247  001010'  000000  000123
1248  001011'  000000  000125
1249  001012'  000000  000102
1250  001013'  000000  000302
1251  001015'  000000  000000
1252                    000003          21760           ERRRG==Q
1253                                    21780           DCE"OD"
1254  001014'  000000  000117          21800           DCL"OUT OF DATA"
1255  001015'  000000  000125
1256  001016'  000000  000124
1257  001017'  000000  000040
1258  001020'  000000  000117
1259  001021'  000000  000106
1260  001022'  000000  000040
1261  001023'  000000  000104
1262  001024'  000000  000101
1263  001025'  000000  000124
1264  001026'  000000  000101
1265  001028'  000000  000301
```



```
1266  001027'  000000  000000
1267                    000004          21820           ERROD==Q
1268                                    21840           DCE"FC"
1269  001030'  000000  000111          21860           DCL"ILLEGAL FUNCTION CALL"
1270  001031'  000000  000114
1271  001032'  000000  000114
1272  001033'  000000  000105
1273  001034'  000000  000107
1274  001035'  000000  000101
1275  001036'  000000  000114
1276  001037'  000000  000040
1277  001040'  000000  000105
1278  001041'  000000  000125
1279  001042'  000000  000116
1280  001043'  000000  000103
1281  001044'  000000  000124
1282  001045'  000000  000111
1283  001046'  000000  000117
1284  001047'  000000  000116
1285  001050'  000000  000040
1286  001051'  000000  000103
1287  001052'  000000  000101
1288  001053'  000000  000114
1289  001054'  000000  000114
1290  001054'  000000  000314
1291  001055'  000000  000000
1292                    000005          21880           ERRFC==Q
1293                                    21900           DCE"OV"
1294  001056'  000000  000117          21920           DCL"OVERFLOW"
1295  001057'  000000  000126
1296  001060'  000000  000105
1297  001061'  000000  000122
1298  001062'  000000  000106
1299  001063'  000000  000114
1300  001064'  000000  000117
1301  001065'  000000  000127
1302  001065'  000000  000327
1303  001066'  000000  000000
1304                    000006          21940           ERROV==Q
1305                                    21960           DCE"OM"
1306  001067'  000000  000117          21980           DCL"OUT OF MEMORY"
1307  001070'  000000  000125
1308  001071'  000000  000124
1309  001072'  000000  000040
1310  001073'  000000  000117
1311  001074'  000000  000106
1312  001075'  000000  000040
1313  001076'  000000  000115
1314  001077'  000000  000105
1315  001100'  000000  000115
1316  001101'  000000  000117
1317  001102'  000000  000122
1318  001103'  000000  000131
```

```
1319  001103' 000000  000331
1320  001104' 000000  000000
1321                  000007          22000           ERROM==0
1322                                  22020           DCE"US"
1323  001105' 000000  000125          22040           DCL"UNDEFINED STATEMENT"
1324  001106' 000000  000116
1325  001107' 000000  000104
1326  001110' 000000  000105
1327  001111' 000000  000106
1328  001112' 000000  000111
1329  001113' 000000  000116
1330  001114' 000000  000105
1331  001115' 000000  000104
1332  001116' 000000  000040
1333  001117' 000000  000123
1334  001120' 000000  000124
1335  001121' 000000  000101
1336  001122' 000000  000124
1337  001123' 000000  000105
1338  001124' 000000  000115
1339  001125' 000000  000105
1340  001126' 000000  000116
1341  001127' 000000  000124
1342  001127' 000000  000324
1343  001130' 000000  000000
1344                  000010          22060           ERRUS==0
1345                                  22080           DCE"BS"
1346  001131' 000000  000123          22100           DCL"SUBSCRIPT OUT OF RANGE"
1347  001132' 000000  000125
1348  001133' 000000  000102
1349  001134' 000000  000123
1350  001135' 000000  000103
1351  001136' 000000  000122
1352  001137' 000000  000111
1353  001140' 000000  000120
1354  001141' 000000  000124
1355  001142' 000000  000040
1356  001143' 000000  000117
1357  001144' 000000  000125
1358  001145' 000000  000124
1359  001146' 000000  000040
1360  001147' 000000  000117
1361  001150' 000000  000106
1362  001151' 000000  000040
1363  001152' 000000  000122
1364  001153' 000000  000101
1365  001154' 000000  000116
1366  001155' 000000  000107
1367  001156' 000000  000105
1368  001156' 000000  000305
1369  001157' 000000  000000
1370                  000011          22120           ERRBS==0
1371                                  22140           DCE"DD"
```



```
1372  001160' 000000  000122          22160           DCL"REDIMENSIONED ARRAY"
1373  001161' 000000  000105
1374  001162' 000000  000104
1375  001163' 000000  000111
1376  001164' 000000  000115
1377  001165' 000000  000105
1378  001166' 000000  000116
1379  001167' 000000  000123
1380  001170' 000000  000111
1381  001171' 000000  000117
1382  001172' 000000  000116
1383  001173' 000000  000105
1384  001174' 000000  000104
1385  001175' 000000  000040
1386  001176' 000000  000101
1387  001177' 000000  000122
1388  001200' 000000  000122
1389  001201' 000000  000101
1390  001202' 000000  000131
1391  001202' 000000  000331
1392  001203' 000000  000000
1393                  000012          22180           ERRDD==0
1394                                  22200           DCE"/0"
1395  001204' 000000  000104          22220           DCL"DIVISION BY ZERO"
1396  001205' 000000  000111
1397  001206' 000000  000126
1398  001207' 000000  000111
1399  001210' 000000  000123
1400  001211' 000000  000111
1401  001212' 000000  000117
1402  001213' 000000  000116
1403  001214' 000000  000040
1404  001215' 000000  000102
1405  001216' 000000  000131
1406  001217' 000000  000040
1407  001220' 000000  000132
1408  001221' 000000  000105
1409  001222' 000000  000122
1410  001223' 000000  000117
1411  001223' 000000  000317
1412  001224' 000000  000000
1413                  000013          22240           ERRDV0==0
1414                                  22260           DCE"ID"
1415  001225' 000000  000111          22280           DCL"ILLEGAL DIRECT"
1416  001226' 000000  000114
1417  001227' 000000  000114
1418  001230' 000000  000105
1419  001231' 000000  000107
1420  001232' 000000  000101
1421  001233' 000000  000114
1422  001234' 000000  000040
1423  001235' 000000  000104
1424  001236' 000000  000111
```

```
1425  001237' 000000  000122
1426  001240' 000000  000105
1427  001241' 000000  000103
1428  001242' 000000  000124
1429  001243' 000000  000324
1430  001245' 000000  000000
1431                  000014          22300           ERRID==0
1432                                  22320   IFN     STRING,<
1433                                  22340           DCE"TM"
1434  001244' 000000  000124         22360           DCL"TYPE MISMATCH"
1435  001245' 000000  000131
1436  001246' 000000  000120
1437  001247' 000000  000105
1438  001250' 000000  000040
1439  001251' 000000  000115
1440  001252' 000000  000111
1441  001253' 000000  000123
1442  001254' 000000  000115
1443  001255' 000000  000101
1444  001256' 000000  000124
1445  001257' 000000  000103
1446  001260' 000000  000110
1447  001260' 000000  000310
1448  001261' 000000  000000
1449                  000015          22380           ERRTM==0
1450                                  22400           DCE"OS"
1451  001262' 000000  000117         22420           DCL"OUT OF STRING SPACE"
1452  001263' 000000  000125
1453  001264' 000000  000124
1454  001265' 000000  000040
1455  001266' 000000  000117
1456  001267' 000000  000106
1457  001270' 000000  000040
1458  001271' 000000  000123
1459  001272' 000000  000124
1460  001273' 000000  000122
1461  001274' 000000  000111
1462  001275' 000000  000116
1463  001276' 000000  000107
1464  001277' 000000  000040
1465  001300' 000000  000123
1466  001301' 000000  000120
1467  001302' 000000  000101
1468  001303' 000000  000103
1469  001304' 000000  000105
1470  001304' 000000  000305
1471  001305' 000000  000000
1472                  000016          22440           ERRSO==0
1473                                  22460           DCE"LS"
1474  001306' 000000  000123         22480           DCL"STRING TOO LONG"
1475  001307' 000000  000124
1476  001310' 000000  000122
1477  001311' 000000  000111
```



```
1478  001312' 000000  000116
1479  001313' 000000  000107
1480  001314' 000000  000040
1481  001315' 000000  000124
1482  001316' 000000  000117
1483  001317' 000000  000117
1484  001320' 000000  000040
1485  001321' 000000  000114
1486  001322' 000000  000117
1487  001323' 000000  000116
1488  001324' 000000  000107
1489  001324' 000000  000307
1490  001325' 000000  000000
1491                  000017          22500           ERRLS==0
1492                                  22520           DCE"ST"
1493  001326' 000000  000123         22540           DCL"STRING FORMULA TOO COMPLEX"
1494  001327' 000000  000124
1495  001330' 000000  000122
1496  001331' 000000  000111
1497  001332' 000000  000116
1498  001333' 000000  000107
1499  001334' 000000  000040
1500  001335' 000000  000106
1501  001336' 000000  000117
1502  001337' 000000  000122
1503  001340' 000000  000115
1504  001341' 000000  000125
1505  001342' 000000  000114
1506  001343' 000000  000101
1507  001344' 000000  000040
1508  001345' 000000  000124
1509  001346' 000000  000117
1510  001347' 000000  000117
1511  001350' 000000  000040
1512  001351' 000000  000103
1513  001352' 000000  000117
1514  001353' 000000  000115
1515  001354' 000000  000120
1516  001355' 000000  000114
1517  001356' 000000  000105
1518  001357' 000000  000130
1519  001357' 000000  000330
1520  001360' 000000  000000
1521                  000020          22560           ERRST==0>
1522                                  22580   IFN     LENGTH,<
1523                                  22600           DCE"CN"
1524  001361' 000000  000103         22620           DCL"CAN'T CONTINUE"
1525  001362' 000000  000101
1526  001363' 000000  000116
1527  001364' 000000  000047
1528  001365' 000000  000124
1529  001366' 000000  000040
1530  001367' 000000  000103
```

```
1531  001370' 000000  000117
1532  001371' 000000  000116
1533  001372' 000000  000124
1534  001373' 000000  000111
1535  001374' 000000  000116
1536  001375' 000000  000125
1537  001376' 000000  000105
1538  001376' 000000  000305
1539  001377' 000000  000000
1540                  000021          22640           ERRCN==.-0>
1541                                  22660   IFN     FUNCTS,<
1542                                  22680           DCE"UF"
1543  001400' 000000  000125         22700           DCL"UNDEFINED USER FUNCTION"
1544  001401' 000000  000116
1545  001402' 000000  000104
1546  001403' 000000  000105
1547  001404' 000000  000106
1548  001405' 000000  000111
1549  001406' 000000  000116
1550  001407' 000000  000105
1551  001410' 000000  000104
1552  001411' 000000  000040
1553  001412' 000000  000125
1554  001413' 000000  000123
1555  001414' 000000  000105
1556  001415' 000000  000122
1557  001416' 000000  000040
1558  001417' 000000  000106
1559  001420' 000000  000125
1560  001421' 000000  000116
1561  001422' 000000  000103
1562  001423' 000000  000124
1563  001424' 000000  000111
1564  001425' 000000  000117
1565  001426' 000000  000116
1566  001426' 000000  000316
1567  001427' 000000  000000
1568                  000022          22720           ERRUF==.-0>
1569
1570                                  22760   PAGE
```



```
1571                                  22780   SUBTTL  LOW SEGMENT -- RAM -- IE THIS STUFF IS NOT CONSTANT
1572                                  22800   ;
1573                                  22820   ; THIS IS THE "VOLATILE" STORAGE AREA AND NONE OF IT
1574                                  22840   ; CAN BE KEPT IN ROM, ANY CONSTANTS IN THIS AREA CANNOT
1575                                  22860   ; BE KEPT IN A ROM, BUT MUST BE LOADED IN BY THE
1576                                  22880   ; PROGRAM INSTRUCTIONS IN ROM.
1577                                  22900   ;
1578
1579  001430' 000000  000054         22940   BUFMIN: 44                              ;A COMMA (PRELOAD OR ROM)
1580                                  22960                                           ;USED BY INPUT STATEMENT SINCE THE
1581                                  22980                                           ;DATA POINTER ALWAYS STARTS ON A
1582                                  23000                                           ;COMMA OR TERMINATOR
1583  001431'                         23020   BUF:    BLOCK   BUFLEN                  ;TYPE IN STORED HERE
1584                                  23040                                           ;DIRECT STATEMENTS EXECUTE OUT OF
1585                                  23060                                           ;HERE, REMEMBER INPUT SMASHES BUF,
1586                                  23080                                           ;MUST BE AT A LOWER ADDRESS
1587                                  23100                                           ;THAN DSCTMP OR ASSIGMENT OF STRING
1588                                  23120                                           ;VALUES IN DIRECT STATEMENTS WON'T COPY
1589                                  23140                                           ;INTO STRING SPACE -- WHICH IT MUST
1590                                  23160   IFN     LPTSW,<
1591                                  23180   LPTPOS: BLOCK   1               ;POSITION OF LPT PRINT HEAD
1592                                  23200   PRTFLG: BLOCK   1>              ;WHETHER OUTPUT GOES TO LPT
1593                                  23220                                   ;NON-ZERO MEANS SEND OUTPUT TO LPT
1594                                  23240   IFN     CONTRW,<
1595  001541'                         23260   CNTWFL: BLOCK   1>              ;SUPRESS OUTPUT FLAG
1596  001542'                         23280   DIMFLG: BLOCK   1               ;IN GETTING A POINTER TO A VARIABLE
1597                                  23300                                   ;IT IS IMPORTANT TO REMEMBER WHETHER IT
1598                                  23320                                   ;IS BEING DONE FOR "DIM" OR NOT
1599                                  23340                                   ;DIMFLG AND VALTYP MUST BE
1600                                  23360                                   ;CONSECUTIVE LOCATIONS
1601                                  23380   IFN     STRING,<
1602  001543'                         23400   VALTYP: BLOCK   1               ;THE TYPE INDICATOR
1603                                  23420                                   ;IN THE 8K 0=NUMERIC 1=STRING
1604  001544'                         23435   OPRTYP:                         ;USED TO STORE OPERATOR NUMBER
1605                                  23436                                   ;IN THE EXTENDED MOMENTARILY BEFORE
1606                                  23437                                   ;OPERATOR APPLICATION
1607  001544'                         23440   DORES:  BLOCK   1               ;WHETHER CAN OR CAN'T CRUNCH RES'D WORDS
1608                                  23460                                   ;TURNED ON IN THE 8K WHEN "DATA"
1609                                  23480                                   ;BEING SCANNED BY CRUNCH SO UNQUOTED
1610                                  23500                                   ;STRINGS WON'T BE CRUNCHED.
1611  001545'                         23520   MEMSIZ: BLOCK   2               ;HIGHEST LOCATION IN MEMORY
1612  001547'                         23540   TEMPPT: BLOCK   2               ;POINTER AT FIRST FREE TEMP DESCRIPTOR
1613                                  23560                                   ;INITIALIZED TO POINT TO TEMPST
1614  001551'                         23580   TEMPST: BLOCK   STRSIZ*NUMTMP   ;STORAGE FOR NUMTMP TEMP DESCRIPTORS
1615  001570'                         23600   DSCTMP: BLOCK   STRSIZ          ;STRING FUNCTIONS BUILD ANSWER DESCRIPTOR HERE
1616  001573'                         23620   FRETOP: BLOCK   2>              ;TOP OF STRING FREE SPACE
1617                                  23640   IFN     LENGTH!STRING,<
1618  001575'                         23660   TEMP3:  BLOCK   2>              ;USED TO HOLD VARB # OF HIGH LOC FOUND
1619                                  23680                                   ;IN GARBAGE COLLECTION
1620                                  23700                                   ;AND USED MOMENTARILY BY FRMEVL
1621                                  23720                                   ;USED IN EXTENDED BY FOUT
1622                                  23740                                   ;ARRAY VARIABLE HANDLING TEMPORARY
1623                                  23760   IFN     LENGTH,<
```

```
1624  001577'                 23780  DATLIN: BLOCK   2               ;DATA LINE # -- REMEMBER FOR ERRORS
1625  001601'                 23800  SUBFLG: BLOCK   1>              ;FLAG WHETHER SUBSCRIPTED VARIABLE ALLOWED
1626                          23820                                  ;"FOR" AND USER-DEFINED FUNCTION
1627                          23840                                  ;POINTER FETCHING TURN
1628                          23860                                  ;THIS ON BEFORE CALLING PTRGET
1629                          23880                                  ;SO ARRAYS WON'T BE DETECTED.
1630                          23900                                  ;STKINI AND PTRGET CLEAR IT.
1631  001602'                 23920  FLGINP: BLOCK   1               ;FLAGS WHETHER WE ARE DOING INPUT
1632                          23940                                  ;OR A READ
1633  001603'                 23960  TEMP:   BLOCK   2               ;TEMPORARY FOR STATEMENT CODE
1634                          23980                                  ;NEWSTT SAVES [H,L] HERE FOR INPUT AND ^C
1635                          24000                                  ;"LET" SAVES NUMBERIC VARIABLE
1636                          24020                                  ;POINTERS HERE FOR "FOR"
1637                          24040                                  ;"NEXT" SAVES ITS TEXT POINTER HERE
1638                          24060                                  ;CLEARC SAVE [H,L] HERE
1639  001605'                 24080  TEMP2:  BLOCK   2               ;FORMULA EVALUATOR TEMP
1640                          24100                                  ;MUST BE PRESERVED BY OPERATORS
1641                          24120                                  ;USED IN EXTENDED BY FOUT
1642                          24140                                  ;ARRAY VARIABLE HANDLER TEMPORARY
1643  001607'                 24160  CURLIN: BLOCK   2               ;CURRENT LINE #
1644                          24180                                  ;SET TO 65535 WHEN DIRECT STATEMENTS EXECUTE
1645                          24200  IFN     LENGTH,<
1646  001611'                 24220  OLDLIN: BLOCK   2               ;OLD LINE NUMBER
1647  001613'                 24240  OLDTXT: BLOCK   2>              ;OLD TEXT POINTER
1648                          24260                                  ;POINTS AT STATEMENT TO BE EXECUTED NEXT
1649  001615'                 24280  STKTOP: BLOCK   2               ;TOP LOCATION TO USE FOR THE STACK
1650                          24300                                  ;INITIALLY SET UP BY INIT
1651                          24320                                  ;ACCORDING TO MEMORY SIZE
1652                          24340                                  ;TO ALLOW FOR 50 BYTES OF STRING SPACE,
1653                          24360                                  ;CHANGED BY A CLEAR COMMAND WITH
1654                          24380                                  ;AN ARGUMENT,
1655  001617'                 24400  TXTTAB: BLOCK   2               ;POINTER TO BEGINNING OF TEXT
1656                          24420                                  ;DOESN'T CHANGE AFTER BEING
1657                          24440                                  ;SETUP BY INIT,
1658  001621'                 24460  VARTAB: BLOCK   2               ;POINTER TO START OF SIMPLE
1659                          24480                                  ;VARIABLE SPACE
1660                          24500                                  ;UPDATED WHENEVER THE SIZE OF THE
1661                          24520                                  ;PROGRAM CHANGES, SET TO [TXTTAB]
1662                          24540                                  ;BY SCRATCH ("NEW"),
1663  001623'                 24560  ARYTAB: BLOCK   2               ;POINTER TO BEGINNING OF ARRAY
1664                          24580                                  ;TABLE
1665                          24600                                  ;INCREMENTED BY 6 WHENEVER
1666                          24620                                  ;A NEW SIMPLE VARIABLE IS FOUND, AND
1667                          24640                                  ;SET TO [VARTAB] BY CLEARC,
1668  001625'                 24660  STREND: BLOCK   2               ;END OF STORAGE IN USE
1669                          24680                                  ;INCREASED WHENEVER A NEW ARRAY
1670                          24700                                  ;OR SIMPLE VARIABLE IS ENCOUNTERED
1671                          24720                                  ;SET TO [VARTAB] BY CLEARC,
1672  001627'                 24740  DATPTR: BLOCK   2               ;POINTER TO DATA, INITIALIZED TO POINT
1673                          24760                                  ;AT THE ZERO IN FRONT OF [TXTTAB]
1674                          24780  IFE     LENGTH-2,<
1675  001631'                 24800  TRCFLG: BLOCK   1>              ;0 MEANS NO TRACE IN PROGRESS
1676
```



```
1677                          24840  ;THE FLOATING ACCUMULATOR
1678                          24860  IFE     LENGTH-2,<
1679  001632'                 24880          BLOCK   1               ;[TEMPORARY LEAST SIGNIFICANT BYTE]
1680  001633'                 24900  DFACLO: BLOCK   4>              ;[FOUR LOWEST ORDERS FOR DOUBLE PRECISION]
1681  001637'                 24920  FACLO:  BLOCK   3               ;[LOW ORDER OF MANTISSA]
1682                          24940                                  ;[MIDDLE ORDER OF MANTISSA]
1683                          24960                                  ;[HIGH ORDER OF MANTISSA]
1684  001642'                 24980  FAC:    BLOCK   2               ;[EXPONENT]
1685                          25000                                  ;[TEMPORARY COMPLEMENT OF SIGN IN MSB]
1686                          25020  IFE     LENGTH-2,<
1687  001644'                 25040          BLOCK   1               ;[TEMPORARY LEAST SIGNIFICANT BYTE]
1688  001645'                 25060  ARGLO:  BLOCK   7               ;[LOCATION OF SECOND ARGUMENT FOR DOUBLE
1689  001654'                 25080  ARG:    BLOCK   1>              ; PRECISION]
1690  001655'                 25100  FBUFFR: BLOCK   13              ;BUFFER FOR FOUT
1691  001672'                 25120  IFE     LENGTH-2,<BLOCK 35-13>
1692                          25140  PAGE
```

```
                                    25140   SUBTTL  TEXT CONSTANTS FOR PRINT OUT
                                    25160   ;
                                    25180   ; NEEDED FOR MESSAGES IN ALL VERSIONS
                                    25200   ; MUST BE STORED ABOVE DSCTMP OR ELSE STRLIT
                                    25220   ; WILL COPY THEM BEFORE STRPRT PRINTS THEM, THIS IS BAD, SINCE IF THE
                                    25240   ; USER IS OUT OF STRING SPACE BASIC WILL LOOP GETTING "OUT OF STRING SPACE"
                                    25260   ; ERRORS.
                                    25280   ;
                                    25300   IFN     LENGTH-2,<
                                    25320   ERR:    DC" ERROR"
                                    25340           0>
001720' 000000  000040              25360   INTXT:  DC" IN "
001721' 000000  000111
001722' 000000  000116
001723' 000000  000040
001724' 000000  000240
001725' 000000  000000              25380           0
001726' 000000  000015              25400   READY:  ACRLF
001727' 000000  000117              25420           DC"OK"
001730' 000000  000113
001731' 000000  000015              25440           ACRLF
001732' 000000  000012
001733' 000000  000000              25460           0
                                    25480   IFN     LENGTH,<
001734' 000000  000015              25500   BRKTXT: ACRLF
001735' 000000  000012
001736' 000000  000102              25520           DC"BREAK"
001737' 000000  000122
001740' 000000  000105
001741' 000000  000101
001742' 000000  000113
001743' 000000  000313
001745' 000000  000000              25540           0>
                                    25560   PAGE
```



```
                                    25600   SUBTTL  GENERAL STORAGE MANAGEMENT ROUTINES
                                    25620   ;
                                    25640   ; FIND A FOR ENTRY ON THE STACK WITH THE VARIABLE POINTER
                                    25660   ; PASSED IN [D,E].
                                    25680   ;
001744' 001000  000041              25700   FNDFOR: LXI     H,4+$CODE       ;IGNORING EVERYONES "NEWSTT"
001745' 000000  000004'
001746' 000000  000726'
                                    25720                                   ;AND THE RETURN ADDRESS OF THIS
001747' 001000  000071              25740           DAD     SP              ;SUBROUTINE, SET [H,L]=SP
001750' 001000  000176              25760   LOOPER: MOV     A,M             ;SEE WHAT TYPE OF THING IS ON THE STACK
001751' 001000  000043              25780           INX     H
001752' 001000  000376              25800           CPI     FORTK           ;IS THIS STACK ENTRY A FOR?
001753' 000000  000201
001754' 001000  000300              25820           RNZ                     ;NO SO OK
                                    25840   IFE     LENGTH,<
                                    25860           PUSHM                   ;GET VARIABLE NAME
                                    25880           XTHL>
                                    25900   IFN     LENGTH,<
001755' 001000  000116              25920           MOV     C,M
001756' 001000  000043              25940           INX     H               ;DO EQUIVALENT OF PUSHM / XTHL
001757' 001000  000106              25960           MOV     B,M
001760' 001000  000043              25980           INX     H
001761' 001000  000345              26000           PUSH    H               ;PUT H  ON
001762' 001000  000151              26020           MOV     L,C             ;PUSH B / XTHL IS SLOWER
001763' 001000  000140              26040           MOV     M,B
001764' 001000  000172              26060           MOV     A,D             ;FOR THE "NEXT" STATMENT WITHOUT AN ARGUMENT
001765' 001000  000263              26080           ORA     E               ;WE MATCH ON ANYTHING
001766' 001000  000353              26100           XCHG                    ;MAKE SURE WE RETURN [D,E]
001767' 001000  000312              26120           JZ      POPGOF          ;POINTING TO THE VARIABLE
001770' 000000  001774'
001771' 000000  001745'
001772' 001000  000353              26140           XCHG>
001773' 001000  000347              26160           COMPAR
001774' 001000  000001              26180   POPGOF: LXI     B,$CODE+13      ;TO WIPE OUT A FOR ENTRY
001775' 000000  000015'
001776' 000000  001770'
001777' 001000  000341              26200           POP     H
002000' 001000  000310              26220           RZ                      ;IF FOR MATCHES GOOD
002001' 001000  000011              26240           DAD     B
002002' 001000  000303              26260           JMP     LOOPER          ;TRY THE NEXT ONE
002003' 000000  001750'
002004' 000000  001775'
                                    26280   ;
                                    26300   ; THIS IS THE BLOCK TRANSFER ROUTINE
                                    26320   ; IT MAKES SPACE BY SHOVING EVERYTHING FORWARD
                                    26340   ;
                                    26360   ; [H,L] = DESTINATION OF HIGH ADDRESS
                                    26380   ; [D,E] = LOW ADDRESS TO BE TRANSFERRED
                                    26400   ; [B,C] = HIGH ADDRESS TO BE TRANSFERRED
                                    26420   ;
                                    26440   ; A CHECK IS MADE TO MAKE SURE A REASONABLE AMOUNT
                                    26460   ; OF SPACE REMAINS BETWEEN THE TOP OF THE STACK AND THE HIGHEST LOCATION
```

```
                                        26480   ; TRANSFERRED INTO
                                        26500   ;
                                        26520   ; ON EXIT [H,L]=[D,E]=LOW [B,C]=LOCATION LOW WAS MOVED INTO
                                        26540   ;
002005' 001000  000315                  26580   BLTU:   CALL    REASON          ;CHECK DESTINATION TO MAKE
002006' 000000  002045'
002007' 000000  002003'
                                        26600                                   ;SURE THE STACK WON'T BE OVERRUN
002010' 001000  000305                  26620   BLTUC:  PUSH    B               ;EXCHANGE [B,C] AND [H,L]
002011' 001000  000343                  26640           XTHL
002012' 001000  000301                  26660           POP     B
002013' 001000  000347                  26680   BLTLOP: COMPAR                  ;SEE IF WE ARE DONE
002014' 001000  000176                  26700           MOV     A,M             ;GET THE WORD TO TRANSFER
002015' 001000  000002                  26720           STAX    B               ;TRANSFER IT
002016' 001000  000310                  26740           RZ
002017' 001000  000013                  26760           DCX     B
002020' 001000  000053                  26780           DCX     H               ;BACKUP FOR NEXT GUY
002021' 001000  000303                  26800           JMP     BLTLOP
002022' 000000  002013'
002023' 000000  002006'
                                        26820   ;
                                        26840   ; THIS ROUTINE IS USED TO MAKE SURE A CERTAIN NUMBER
                                        26860   ; OF LOCATIONS REMAIN AVAILABLE FOR THE
                                        26880   ; STACK, THE CALL IS :
                                        26900   ;       CALL    GETSTK
                                        26920   ;       NUMBER OF 2 BYTE ENTRIES NECESSARY
                                        26940   ;
                                        26960   ; THIS ROUTINE MUST BE CALLED BY ANY ROUTINE WHICH PUTS
                                        26980   ; AN ARBITRARY AMOUNT OF STUFF ON THE STACK
                                        27000   ; (I,E, ANY RECURSIVE ROUTINE LIKE FRMEVL)
                                        27020   ; IT IS ALSO CALLED BY ROUTINES SUCH AS GOSUB AND FOR
                                        27040   ; WHICH MAKE PERMANENT ENTRIES ON THE STACK
                                        27060   ;
                                        27080   ;
                                        27100   ; ROUTINES WHICH MERELY USE AND FREE UP THE GUARANTEED
                                        27120   ; NUMLEV STACK LOCATIONS NEED NOT CALL THIS
                                        27140   ;
002024' 001000  000343                  27160   GETSTK: XTHL
002025' 001000  000116                  27180           MOV     C,M             ;GET ARGUMENT INTO [C]
                                        27200
002026' 001000  000043                  27220           INX     H
002027' 001000  000343                  27240           XTHL                    ;PUT BACK RETURN ADDRESS
002030' 001000  000345                  27260           PUSH    H               ;SAVE [H,L]
002031' 001000  000052'                 27280           LHLD    STREND
002032' 000000  001625'
002033' 000000  002022'
002034' 001000  000006                  27300           MVI     B,0
002035' 000000  000000
002036' 001000  000011                  27320           DAD     B
002037' 001000  000011                  27340           DAD     B               ;SEE IF WE CAN HAVE THIS MANY
002040' 001000  000315                  27360           CALL    REASON
002041' 000000  002045'
```



```
002042' 000000  002032'
002043' 001000  000341                  27380           POP     H               ;RESTORE [H,L]
002044' 001000  000311                  27400           RET
                                        27420
                                        27440   ;
                                        27460   ; [H,L]= SOME ADDRESS
                                        27480   ; [H,L] IS EXAMINED TO MAKE SURE AT LEAST NUMLEV
                                        27500   ; LOCATIONS REMAIN BETWEEN IT AND THE TOP OF THE STACK
                                        27520   ;
002045' 001000  000325                  27560   REASON: PUSH    D               ;SAVE [D,E]
002046' 001000  000353                  27580           XCHG                    ;PUT [H,L] IN [D,E]
002047' 001000  000041                  27600           LXI     H,$CODE+65536-2*NUMLEV  ;SETUP OFFSET OF GUARANTEED
002050' 000000  177726'
002051' 000000  002041'
                                        27620                                   ;LOCATIONS
002052' 001000  000071                  27640           DAD     SP              ;[H,L]=STACK POINTER + OFFSET
002053' 001000  000347                  27660           COMPAR                  ;SEE IF THIS IS ,GT, ENTERING [H,L]
002054' 001000  000353                  27680           XCHG                    ;RESTORE [H,L] FROM [D,E]
002055' 001000  000321                  27700           POP     D               ;GET [D,E] BACK
002056' 001000  000320                  27720           RNC                     ;WAS OK?
002057' 001000  000036                  27740   OMERR:  MVI     E,ERROM         ;"OUT OF MEMORY"
002060' 000000  000007
                                        27760   IFE     LENGTH,<
                                        27780           XWD     "01000,1>       ;"LXI B," OVER THE NEXT 2
                                        27800   IFN     LENGTH,<
002061' 001000  000303                  27820           JMP     ERROR>
002062' 000000  002102'
002063' 000000  002050'
                                        27840   PAGE
```

```
                                        27860   SUBTTL ERROR HANDLER, READY, COMPACTIFICATION, NEW, CLEAR, MAIN
                                        27880   IFN     LENGTH,<
002064' 001000  000052                  27900   DATSNE: LHLD    DATLIN          ;GET DATA LINE
002065' 000000  001577'
002066' 000000  002062'
002067' 001000  000042                  27920           SHLD    CURLIN>         ;MAKE IT CURRENT LINE
002070' 000000  001607'
002071' 000000  002665'
002072' 001000  000036                  27940   SNERR:  MVI     E,ERRSN         ;"SYNTAX ERROR"
002073' 000000  000002
002074' 001000  000001                  27960           XWD     ^O1000,1        ;"LXI B," OVER THE NEXT 2
002075' 001000  000036                  27980   DV0ERR: MVI     E,ERRDV0        ;DIVISION BY ZERO
002076' 000000  000013
                                        28000   IFN     LENGTH,<
002077' 001000  000001                  28020           XWD     ^O1000,1        ;SKIP NEXT TWO
002100' 001000  000036                  28040   NFERR:  MVI     E,ERRNF>        ;"NEXT WITHOUT FOR" ERROR
002101' 000000  000001
002102' 001000  000315                  28060   ERROR:  CALL    STKINI          ;RESET THE STACK AND FLAGS
002103' 000000  002470'
002104' 000000  002070'
                                        28080   IFN     CONTRW,<
002105' 001000  000257                  28100           XRA     A
002106' 001000  000062                  28120           STA     CNTWFL>         ;FORCE OUTPUT
002107' 000000  001541'
002110' 000000  002103'
002111' 001000  000315                  28140           CALL    CRDO            ;CRLF
002112' 000000  004523'
002113' 000000  002107'
002114' 001000  000041                  28220           LXI     H,ERRTAB        ;GET START OF ERROR TABLE
002115' 000000  000730'
002116' 000000  002112'
                                        28240   IFE     LENGTH-2,<
002117' 001000  000315                  28260   LEPSKP: CALL    REM             ;SKIP AN ERROR MESSAGE
002120' 000000  004074'
002121' 000000  002115'
002122' 001000  000035                  28280           DCR     E               ;DECREMENT ERROR COUNT
002123' 001000  000043                  28282           INX     H               ;SKIP OVER THIS ERROR MESSAGE
002124' 001000  000302                  28300           JNZ     LEPSKP>         ;SKIP SOME MORE
002125' 000000  002117'
002126' 000000  002120'
                                        28320   IFN     LENGTH-2,<
                                        28322           MOV     D,A             ;GET ZERO INTO D
                                        28324           MVI     A,"?"           ;START OF ERROR MESSAGE
                                        28326           OUTCHR                  ;TYPE IT
                                        28340           DAD     D               ;ADD IN ERROR CODE
                                        28360           MOV     A,M             ;GET FIRST ERROR CHARACTER
                                        28380           OUTCHR                  ;TYPE IT
                                        28400           CHRGET                  ;GET 2ND CHARACTER OF ERROR CODE
                                        28420           OUTCHR                  ;TYPE IT
                                        28440           LXI     H,ERR>          ;GET POINTER TO " ERROR"
002127' 001000  000315                  28460   ERRFIN: CALL    STROUT          ;TYPE IT
002130' 000000  007743'
002131' 000000  002125'
```



```
002132' 001000  000052                  28480           LHLD    CURLIN          ;CURRENT LINE #
002133' 000000  001607'
002134' 000000  002130'
002135' 001000  000174                  28500           MOV     A,H             ;SEE IF IN DIRECT MODE
002136' 001000  000245                  28520           ANA     L
002137' 001000  000074                  28540           INR     A               ;ZERO SAYS DIRECT MODE
002140' 001000  000304                  28560           CNZ     INPRT           ;PRINT LINE NUMBER IN (H,L)
002141' 000000  000000*
002142' 000000  002133'
                                        28580   IFE     LENGTH,<
                                        28600           XWD     ^O1000,1        ;"LXI B," OVER THE NEXT 2
                                        28620   END:
                                        28640   STOP:   RNZ                     ;MAKE SURE WE TERMINATED IT
                                        28660   STPEND: POP     B               ;GET RID OF "NEWSTT" RETURN ADDRESS
                                        28680   ENDCON:>
                                        28682   IFE     LENGTH-2,<
                                        28684   ;
                                        28686   ; FOR "LIST" COMMAND STOPPING
                                        28688   ;
002143' 001000  000076                  28690           XWD     ^O1000,^O76             ;SKIP THE NEXT BYTE
002144' 001000  000301                  28692   STPRDY: POP     B>
                                        28700   READY:  IFN     LPTSW,<
                                        28720           CALL    FINLPT>         ;PRINT ANY LEFT OVERS
                                        28740   IFN     CONTRW,<
002145' 001000  000257                  28760           XRA     A
002146' 001000  000062                  28780           STA     CNTWFL>         ;FORCE OUTPUT
002147' 000000  001541'
002150' 000000  002141'
002151' 001000  000041                  28800           LXI     H,$CODE+65535
002152' 000000  177777'
002153' 000000  002147'
002154' 001000  000042                  28820           SHLD    CURLIN          ;SETUP CURLIN FOR DIRECT MODE
002155' 000000  001607'
002156' 000000  002152'
002157' 001000  000041                  28840           LXI     H,REDDY         ;"READY" CRLF CRLF
002160' 000000  001725'
002161' 000000  002155'
002162' 001000  000315                  28860   REPINI: CALL    INIT            ;PRINT IT, REPLACED BY CALL STROUT
002163' 000000  000000*
002164' 000000  002160'
                                        28880                                   ;BY THE INIT CODE, THIS IS HERE SO AFTER
                                        28900                                   ;ERRORS DURING INIT, INIT IS RESTARTED
002165' 001000  000315                  28920   MAIN:   CALL    INLIN           ;GET A LINE FROM TTY
002166' 000000  002776'
002167' 000000  002163'
002170' 001000  000327                  28940           CHRGET                  ;GET THE FIRST
002171' 001000  000074                  28960           INR     A               ;SEE IF 0 SAVING THE CARRY FLAG
002172' 001000  000075                  28980           DCR     A
002173' 001000  000312                  29000           JZ      MAIN            ;IF SO, A BLANK LINE WAS INPUT
002174' 000000  002165'
002175' 000000  002166'
002176' 001000  000365                  29020           PUSH    PSW             ;SAVE STATUS INDICATOR FOR 1ST CHARACTER
002177' 001000  000315                  29040           CALL    LINGET          ;READ IN A LINE #
```

```
1972  002200' 000000  003642'
1973  002201' 000000  002174'
1974  002202' 001000  000325          29060           PUSH    D               ;SAVE LINE #
1975  002203' 001000  000315          29080           CALL    CRUNCH          ;CRUNCH THE LINE DOWN
1976  002204' 000000  002533'
1977  002205' 000000  002200'
1978  002206' 001000  000107          29100           MOV     B,A             ;A=0 AFTER CRUNCH, (B,C)=CHAR COUNT FOR NODEL
1979  002207' 001000  000321          29120           POP     D               ;RESTORE LINE #
1980  002210' 001000  000361          29140           POP     PSW             ;WAS THERE A LINE #?
1981  002211' 001000  000322          29160           JNC     GONE            ;IF NOT ITS A DIRECT STATEMENT
1982  002212' 000000  003370'
1983  002213' 000000  002204'
1984  002214' 001000  000325          29180           PUSH    D
1985  002215' 001000  000305          29200           PUSH    B               ;SAVE LINE # AND CHARACTER COUNT
1986  002216' 001000  000327          29220           CHRGET                  ;REMEMBER IF THIS LINE IS
1987  002217' 001000  000365          29240           PUSH    PSW             ;BLANK SO WE DON'T INSERT IT
1988  002220' 001000  000315          29260           CALL    FNDLIN          ;GET A POINTER TO THE LINE
1989  002221' 000000  002371'
1990  002222' 000000  002212'
1991  002223' 001000  000305          29280           PUSH    B               ;SAVE THE POINTER
1992                                  29300   IFE     LENGTH=2,<
1993  002224' 001000  000334          29320           CC      DEL>            ;DELETE THE LINE
1994  002225' 000000  011271'
1995  002226' 000000  002221'
1996                                  29340   IFN     LENGTH=2,<
1997                                  29360           JNC     NODEL           ;NO MATCH SO DON'T DELETE
1998                                  29380           XCHG                    ;(D,E) NOW HAS THE POINTER TO THE LINE
1999                                  29400                                   ;BEYOND THIS ONE
2000                                  29420           LHLD    VARTAB          ;COMPACTIFYING TO VARTAB
2001                                  29440   MLOOP:  LDAX    D
2002                                  29460           STAX    B               ;SHOVING DOWN TO ELIMINATE A LINE
2003                                  29480           INX     B
2004                                  29500           INX     D
2005                                  29520           COMPAR
2006                                  29540           JNC     MLOOP           ;DONE COMPACTIFYING?
2007                                  29560           MOV     M,B
2008                                  29580           MOV     L,C
2009                                  29600           INX     H               ;NEW VARTAB
2010                                  29620           SHLD    VARTAB>
2011  002227' 001000  000321          29640   NODEL:  POP     D               ;POP POINTER AT PLACE TO INSERT
2012  002230' 001000  000361          29660           POP     PSW             ;SEE IF THIS LINE HAD
2013                                  29680                                   ;ANYTHING ON IT
2014  002231' 001000  000312          29700           JZ      FINI            ;IF NOT DON'T INSERT
2015  002232' 000000  002276'
2016  002233' 000000  002225'
2017  002234' 001000  000052          29720           LHLD    VARTAB          ;CURRENT END
2018  002235' 000000  001621'
2019  002236' 000000  002232'
2020  002237' 001000  000343          29740           XTHL                    ;(H,L)=CHARACTER COUNT, VARTAB
2021                                  29760                                   ;ONTO THE STACK
2022  002240' 001000  000301          29780           POP     B               ;(B,C)=OLD VARTAB
2023  002241' 001000  000011          29800           DAD     B
2024  002242' 001000  000345          29820           PUSH    H               ;SAVE NEW VARTAB
```



```
2025  002243' 001000  000315          29840           CALL    BLTU
2026  002244' 000000  002605'
2027  002245' 000000  002235'
2028  002246' 001000  000341          29860           POP     H               ;POP OFF VARTAB
2029  002247' 001000  000042          29880           SHLD    VARTAB          ;UPDATE VARTAB
2030  002250' 000000  001621'
2031  002251' 000000  002244'
2032  002252' 001000  000353          29900           XCHG
2033  002253' 001000  000164          29920           MOV     M,H             ;FOOL CHEAD WITH NON-ZERO LINK
2034  002254' 001000  000043          29940           INX     H               ;SO IT DOESN'T THINK
2035                                  29960                                   ;THIS LINK IS THE
2036                                  29980                                   ;END OF THE PROGRAM
2037  002255' 001000  000043          30000           INX     H
2038  002256' 001000  000321          30020           POP     D               ;GET LINE # OFF STACK
2039  002257' 001000  000163          30040           MOV     M,E
2040  002260' 001000  000043          30060           INX     H               ;PUT DOWN LINE #
2041  002261' 001000  000162          30080           MOV     M,D
2042  002262' 001000  000043          30100           INX     H
2043
2044  002263' 001000  000021          30140           LXI     D,BUF           ;MOVE LINE FROM BUF TO PROGRAM AREA
2045  002264' 000000  001431'
2046  002265' 000000  002250'
2047  002266' 001000  000032          30160   MLOOPR: LDAX    D               ;NOW TRANSFERING LINE
2048                                  30180                                   ;IN FROM BUF
2049  002267' 001000  000167          30200           MOV     M,A
2050  002270' 001000  000043          30220           INX     H
2051  002271' 001000  000023          30240           INX     D
2052  002272' 001000  000267          30260           ORA     A               ;ZERO MARKS THE END
2053  002273' 001000  000302          30280           JNZ     MLOOPR
2054  002274' 000000  002266'
2055  002275' 000000  002264'
2056  002276' 001000  000315          30300   FINI:   CALL    RUNC            ;DO CLEAR & SET UP STACK
2057  002277' 000000  002437'
2058  002300' 000000  002274'
2059                                  30320                                   ;ALSO SETS (H,L) TO (TXTTAB)-1
2060  002301' 001000  000043          30340           INX     H
2061                                  30360   ;
2062                                  30380   ; CHEAD GOES THROUGH PROGRAM STORAGE AND FIXES
2063                                  30400   ; UP ALL THE LINKS, THE END OF EACH
2064                                  30420   ; LINE IS FOUND BY SEARCHING FOR THE ZERO AT THE END,
2065                                  30440   ; THE DOUBLE ZERO LINK IS USED TO DETECT THE END OF THE PROGRAM
2066                                  30460   ;
2067  002302' 001000  000124          30480   CHEAD:  MOV     D,H             ;(D,E)=(H,L)
2068  002303' 001000  000135          30500           MOV     E,L
2069  002304' 001000  000176          30520           MOV     A,M             ;SEE IF END OF CHAIN
2070  002305' 001000  000043          30540           INX     H               ;BUMP POINTER
2071  002306' 001000  000266          30560           ORA     M               ;2ND BYTE
2072  002307' 001000  000312          30580           JZ      MAIN            ;DONE
2073  002310' 000000  002165'
2074  002311' 000000  002277'
2075  002312' 001000  000043          30600           INX     H               ;FIX H TO START OF TEXT
2076  002313' 001000  000043          30620           INX     H
2077  002314' 001000  000043          30640           INX     H
```

```
002078' 002315' 001000  000257          30660          XRA     A               ;SEARCHING FOR A ZERO IN MEMORY
002079' 002316' 001000  000276          30680  CZLOOP: CMP     M               ;TO MARK THE END OF THIS LINE
002080' 002317' 001000  000043          30700          INX     H               ;BUMP POINTER
002081' 002320' 001000  000302          30720          JNZ     CZLOOP          ;END OF LINE
002082' 002321' 000000  002316'
002083' 002322' 000000  002310'
002084' 002323' 001000  000353          30740          XCHG                    ;SWITCH TEMP
002085' 002324' 001000  000163          30760          MOV     M,E             ;DO FIRST BYTE OF FIXUP
002086' 002325' 001000  000043          30780          INX     H               ;ADVANCE POINTER
002087' 002326' 001000  000162          30800          MOV     M,D             ;2ND BYTE OF FIXUP
002088' 002327' 001000  000353          30820          XCHG                    ;AND BACK AGAIN
002089' 002330' 001000  000303          30840          JMP     CHEAD           ;KEEP CHAINING TIL DONE
002090' 002331' 000000  002302'
002091' 002332' 000000  002321'
002092                                  30860  IFE     LENGTH-2,<
002093                                  30880  ;
002094                                  30900  ; SCNLIN SCANS A LINE RANGE OF
002095                                  30920  ; THE FORM  #-# OR # OR #- OR -# OR BLANK
002096                                  30940  ; AND THEN FINDS THE FIRST LINE IN THE RANGE
002097                                  30960  ;
002098  002333' 001000  000021          30980  SCNLIN: LXI     D,$CODE         ;ASSUME START LIST AT ZERO
002099  002334' 000000  000000'
002100  002335' 000000  002331'
002101  002336' 001000  000325          31000          PUSH    D               ;SAVE INITIAL ASSUMPTION
002102  002337' 001000  000312          31020          JZ      ALLLST          ;IF FINISHED, LIST IT ALL
002103  002340' 000000  002354'
002104  002341' 000000  002334'
002105  002342' 001000  000321          31040          POP     D               ;WE ARE GOING TO GRAB A #
002106  002343' 001000  000315          31060          CALL    LINGET          ;GET A LINE #, IF NONE, RETURNS ZERO
002107  002344' 000000  003642'
002108  002345' 000000  002348'
002109  002346' 001000  000325          31080          PUSH    D               ;SAVE FIRST
002110  002347' 001000  000312          31100          JZ      ONELIN          ;IF ONLY # THEN DONE,
002111  002350' 000000  002365'
002112  002351' 000000  002344'
002113  002352' 001000  000317          31120          SYNCHK  MINUTK          ;MUST BE A DASH,
002114  002353' 000000  000251
002115  002354' 001000  000021          31140  ALLLST: LXI     D,$CODE+*D65529 ;ASSUME MAX END OF RANGE
002116  002355' 000000  177771'
002117  002356' 000000  002350'
002118  002357' 001000  000304          31160          CNZ     LINGET          ;GET THE END OF RANGE
002119  002360' 000000  003642'
002120  002361' 000000  002355'
002121  002362' 001000  000302          31180          JNZ     SNERR           ;MUST BE TERMINATOR
002122  002363' 000000  002072'
002123  002364' 000000  002360'
002124  002365' 001000  000353          31200  ONELIN: XCHG                    ;(H,L) = FINAL
002125  002366' 001000  000321          31220          POP     D               ;GET INITIAL IN (D,E)
002126  002367' 001000  000343          31240          XTHL                    ;PUT MAX ON STACK, RETURN ADDR TO (H,L)
002127  002370' 001000  000345          31260          PUSH    H>              ;SAVE RETURN ADDRESS BACK
002128                                  31280  ;
002129                                  31300  ; FNDLIN SEARCHES THE PROGRAM TEXT FOR THE LINE
002130                                  31320  ; WHOSE LINE # IS PASSED IN (D,E), (D,E) IS PRESERVED,
```



```
002131                                  31340  ; THERE ARE THREE POSSIBLE RETURNS:
002132                                  31360  ;
002133                                  31380  ;       1) ZERO FLAG SET, CARRY NOT SET,  LINE NOT FOUND,
002134                                  31400  ;          NO LINE IN PROGRAM GREATER THAN ONE SOUGHT,
002135                                  31420  ;          (B,C) POINTS TO TWO ZERO BYTES AT END OF PROGRAM,
002136                                  31440  ;          (H,L)=(B,C)
002137                                  31460  ;
002138                                  31480  ;       2) ZERO, CARRY SET,
002139                                  31500  ;          (B,C) POINTS TO THE LINK FIELD IN THE LINE
002140                                  31520  ;          WHICH IS THE LINE SEARCHED FOR,
002141                                  31540  ;          (H,L) POINTS TO THE LINK FIELD IN THE NEXT LINE,
002142                                  31560  ;
002143                                  31580  ;       3) NON-ZERO, CARRY NOT SET,
002144                                  31600  ;          LINE NOT FOUND, (B,C)  POINTS TO LINE IN PROGRAM
002145                                  31620  ;          GREATER THAN ONE SEARCHED FOR,
002146                                  31640  ;          (H,L) POINTS TO THE LINK FIELD IN THE NEXT LINE,
002147                                  31660  ;
002148  002371' 001000  000052          31680  FNDLIN: LHLD    TXTTAB          ;GET POINTER TO START OF TEXT
002149  002372' 000000  001617'
002150  002373' 000000  002363'
002151  002374' 001000  000104          31700  LOOP:   MOV     B,H             ;IF EXITING BECAUSE OF END OF PROGRAM,
002152                                  31720                                  ;SET (B,C) TO POINT TO DOUBLE ZEROES,
002153  002375' 001000  000115          31740          MOV     C,L
002154  002376' 001000  000176          31760          MOV     A,M             ;GET WORD POINTER TO
002155  002377' 001000  000043          31780          INX     H               ;BUMP POINTER
002156  002400' 001000  000266          31800          ORA     M               ;GET 2ND BYTE
002157  002401' 001000  000053          31820          DCX     H               ;GO BACK
002158  002402' 001000  000310          31840          RZ                      ;IF ZERO THEN DONE
002159  002403' 001000  000305          31860          PUSH    B
002160  002404' 001000  000367          31880          PUSHM                   ;PUSH LINK
002161  002405' 001000  000367          31900          PUSHM                   ;PUSH BINARY LINE #
002162  002406' 001000  000341          31920          POP     H               ;POP HERE
002163  002407' 001000  000347          31940          COMPAR                  ;COMPARE (D,E) TO (H,L)
002164  002410' 001000  000341          31960          POP     H               ;GET LINK
002165  002411' 001000  000301          31980          POP     B               ;GET POINTER TO THIS LINE IN (B,C)
002166  002412' 001000  000077          32000          CMC                     ;TURN CARRY ON
002167  002413' 001000  000310          32020          RZ                      ;EQUAL RETURN
002168  002414' 001000  000077          32040          CMC                     ;MAKE CARRY ZERO
002169  002415' 001000  000320          32060          RNC                     ;NO MATCH RETURN (GREATER)
002170  002416' 001000  000303          32080          JMP     LOOP            ;KEEP LOOPING
002171  002417' 000000  002374'
002172  002420' 000000  002372'
002173                                  32100  ;
002174                                  32120  ; THE "NEW" COMMAND CLEARS THE PROGRAM TEXT AS WELL
002175                                  32140  ; AS VARIABLE SPACE
002176                                  32160  ;
002177  002421' 001000  000300          32180  SCRATH: RNZ                     ;MAKE SURE THERE IS A TERMINATOR
002178  002422' 001000  000052          32200  SCRTCH: LHLD    TXTTAB
002179  002423' 000000  001617'
002180  002424' 000000  002417'
002181                                  32220  IFN     LENGTH-2,<
002182                                  32240          XRA     A>
002183                                  32260  IFE     LENGTH-2,<
```

```
 2184  002425' 001000  000315          32280            CALL    TUFF>           ;TURN OFF TRACE, SET [A]=0,
 2185  002426' 000000  003605'
 2186  002427' 000000  002423'
 2187  002430' 001000  000167          32300            MOV     M,A             ;SAVE AT END OFF TEXT
 2188  002431' 001000  000043          32320            INX     H               ;BUMP POINTER
 2189  002432' 001000  000167          32340            MOV     M,A             ;SAVE ZERO
 2190  002433' 001000  000043          32360            INX     H               ;BUMP POINTER
 2191  002434' 001000  000042          32380            SHLD    VARTAB          ;NEW START OF VARIABLE
 2192  002435' 000000  001621'
 2193  002436' 000000  002426'
 2194                                   32400    IFE     LENGTH,<
 2195                                   32420    RUN1:   RNZ>                    ;CHECK FOR A TERMINATOR
 2196  002437' 001000  000052          32440    RUNC:   LHLD    TXTTAB          ;POINT AT THE START OF TEXT
 2197  002440' 000000  001617'
 2198  002441' 000000  002435'
 2199  002442' 001000  000053          32460            DCX     H
 2200                                   32480    ;
 2201                                   32500    ; CLEARC IS A SUBROUTINE WHICH INITIALIZES THE VARIABLE AND
 2202                                   32520    ; ARRAY SPACE BY RESETING ARYTAB [THE END OF SIMPLE VARIABLE SPACE]
 2203                                   32540    ; AND STREND [THE END OF ARRAY STORAGE], IT FALLS INTO STKINI
 2204                                   32560    ; WHICH RESETS THE STACK, [H,L] IS PRESERVED,
 2205                                   32580    ;
 2206                                   32600    IFE     STRING,<CLEAR1>
 2207  002443' 001000  000042          32620    CLEARC: SHLD    TEMP            ;SAVE [H,L] IN TEMP
 2208  002444' 000000  001603'
 2209  002445' 000000  002440'
 2210                                   32640    IFN     STRING,<
 2211  002446' 001000  000052          32660            LHLD    MEMSIZ
 2212  002447' 000000  001545'
 2213  002450' 000000  002444'
 2214  002451' 001000  000042          32680            SHLD    FRETOP>         ;FREE UP STRING SPACE
 2215  002452' 000000  001573'
 2216  002453' 000000  002447'
 2217  002454' 001000  000315          32700            CALL    RESTORE         ;RESTORE DATA
 2218  002455' 000000  003446'
 2219  002456' 000000  002452'
 2220  002457' 001000  000052          32720            LHLD    VARTAB          ;GET START OF VARIABLE SPACE
 2221  002460' 000000  001621'
 2222  002461' 000000  002455'
 2223  002462' 001000  000042          32740            SHLD    ARYTAB          ;SAVE IN START OF ARRAY SPACE
 2224  002463' 000000  001623'
 2225  002464' 000000  002460'
 2226  002465' 001000  000042          32760            SHLD    STREND          ;AND END OF VARIABLE STORAGE
 2227  002466' 000000  001625'
 2228  002467' 000000  002463'
 2229                                   32780    ;
 2230                                   32800    ; STKINI RESETS THE STACK POINTER ELIMINATING
 2231                                   32820    ; GOSUB & FOR CONTEXT,  STRING TEMPORARIES ARE FREED
 2232                                   32840    ; UP, SUBFLG IS RESET, CONTINUING IS DISALLOWED,
 2233                                   32860    ; AND A DUMMY ENTRY IS PUT ON THE STACK, THIS IS SO
 2234                                   32880    ; FNDFOR WILL ALWAYS FIND A NON-"FOR" ENTRY AT THE BOTTOM
 2235                                   32900    ; OF THE STACK, [A]=0 AND [D,E] IS PRESERVED,
 2236                                   32920    ;
```



```
 2237  002470' 001000  000301          32940    STKINI: POP     B               ;GET RETURN ADDRESS HERE
 2238  002471' 001000  000052          32960            LHLD    STKTOP          ;[H,L] POINTER TO END OF MEMORY
 2239  002472' 000000  001615'
 2240  002473' 000000  002466'
 2241  002474' 001000  000371          32980            SPHL                    ;INITIALIZE STACK
 2242                                   33000    IFN     STRING,<
 2243  002475' 001000  000041          33020            LXI     H,TEMPST
 2244  002476' 000000  001551'
 2245  002477' 000000  002472'
 2246  002500' 001000  000042          33040            SHLD    TEMPPT>         ;INITIALIZE STRING TEMPORARIES
 2247  002501' 000000  001547'
 2248  002502' 000000  002476'
 2249  002503' 001000  000041          33060            LXI     H,$CODE         ;PUT ZERO [NON NEXT, FOR, GOSUB TOKEN]
 2250  002504' 000000  000000'
 2251  002505' 000000  002501'
 2252  002506' 001000  000345          33080            PUSH    H               ;ON STACK
 2253  002507' 001000  000042          33100    IFN     LENGTH,<SHLD    OLDTXT> ;MAKE CONTINUING ILLEGAL
 2254  002510' 000000  001613'
 2255  002511' 000000  002504'
 2256                                   33120    IFN     LPTSW,<
 2257                                   33140            CALL    FINLPT>
 2258  002512' 001000  000052          33160            LHLD    TEMP            ;GET SAVED [H,L]
 2259  002513' 000000  001603'
 2260  002514' 000000  002510'
 2261                                   33180    IFN     LENGTH,<
 2262                                   33200    IFE     CONTRW,<XRA     A>
 2263  002515' 001000  000062          33220            STA     SUBFLG>         ;ALLOW SUBSCRIPTS
 2264  002516' 000000  001601'
 2265  002517' 000000  002513'
 2266  002520' 001000  000305          33240            PUSH    B               ;PUT RETURN ADDRESS BACK ON
 2267  002521' 001000  000311          33260            RET                     ;GO BACK
 2268
 2269  002522' 001000  000076          33300    QINLIN: MVI     A,"?"           ;GET A QMARK
 2270  002523' 000000  000077
 2271  002524' 001000  000337          33320            OUTCHR                  ;TYPE IT
 2272  002525' 001000  000076          33340            MVI     A," "           ;SPACE
 2273  002526' 000000  000040
 2274  002527' 001000  000337          33360            OUTCHR                  ;TYPE IT TOO
 2275                                   33380    IFE     STRING,<CALL    INLIN   ;IN THE NON-STRING VERSIONS ALL
 2276                                   33400                                    ;INPUT IS CRUNCHED
 2277                                   33420            INX     H>              ;GET A LINE OF INPUT FROM TTY
 2278  002530' 001000  000303          33440    IFN     STRING,<JMP     INLIN>  ;NO CRUNCHING IN THIS CASE
 2279  002531' 000000  002776'
 2280  002532' 000000  002516'
 2281                                   33460    ;
 2282                                   33520    ; ALL "RESERVED" WORDS ARE TRANSLATED INTO SINGLE
 2283                                   33540    ; BYTES WITH THE MSB ON, THIS SAVES SPACE AND TIME
 2284                                   33560    ; BY ALLOWING FOR TABLE DISPATCH DURING EXECUTION,
 2285                                   33580    ; THEREFORE ALL STATEMENTS APPEAR TOGETHER IN THE
 2286                                   33600    ; RESERVED WORD LIST IN THE SAME
 2287                                   33620    ; ORDER THEY APPEAR IN IN STMDSP,
 2288                                   33640    ;
 2289  002533'                          33700    CRUNCH: IFN     STRING,<
```

```
2290  002533' 001000  000257          33720           XRA     A
2291  002534' 001000  000062          33740           STA     DURES>          ;ALLOW CRUNCHING
2292  002535' 000000  001544'
2293  002536' 000000  002531'
2294  002537' 001000  000016          33760           MVI     C,5             ;COUNT OF CHARS AT LEAST 5
2295  002540' 000000  000005
2296  002541' 001000  000021          33780           LXI     D,BUF           ;SETUP DESTINATION POINTER
2297  002542' 000000  001431'
2298  002543' 000000  002535'
2299  002544' 001000  000176          33800   KLOOP:  MOV     A,M             ;GET CHARACTER FROM BUF
2300  002545' 001000  000376          33820           CPI     " "             ;IS IT A SPACE WE WANT TO SAVE
2301  002546' 000000  000040
2302  002547' 001000  000312          33840           JZ      STUFFM          ;YES, STUFF IN DESTINATION LINE,
2303  002550' 000000  002667'
2304  002551' 000000  002542'
2305  002552' 001000  000107          33860           MOV     B,A             ;GET A CHARACTER FROM THE LINE
2306                                   33880                                   ;SETUP B WITH A QUOTE IF IT IS A STRING
2307  002553' 001000  000376          33900           CPI     34              ;QUOTE SIGN?
2308  002554' 000000  000042
2309  002555' 001000  000312          33920           JZ      STRNG           ;YES, GO TO SPECIAL STRING HANDLING
2310  002556' 000000  002727'
2311  002557' 000000  002550'
2312  002560' 001000  000267          33940           ORA     A               ;END OF LINE?
2313  002561' 001000  000312          33960           JZ      CRDONE          ;YES, DONE CRUNCHING
2314  002562' 000000  002753'
2315  002563' 000000  002556'
2316                                   33980   IFN     STRING,<
2317  002564' 001000  000072          34000           LDA     DURES           ;IN DATA STATEMENT AND NO CRUNCH?
2318  002565' 000000  001544'
2319  002566' 000000  002562'
2320  002567' 001000  000267          34020           ORA     A
2321  002570' 001000  000107          34040           MOV     B,A             ;INITIALIZE RESERVED WORD COUNT
2322  002571' 001000  000176          34060           MOV     A,M             ;GET THE CHARACTER AGAIN
2323  002572' 001000  000302          34080           JNZ     STUFFM>         ;IF NO CRUNCHING JUST STORE
2324  002573' 000000  002667'
2325  002574' 000000  002565'
2326                                   34100                                   ;THE CHARACTER
2327                                   34120   IFN     LENGTH,<                ;NOT FOR 4K VERSION
2328  002575' 001000  000376          34140           CPI     "?"             ;A QMARK?
2329  002576' 000000  000077
2330  002577' 001000  000076          34160           MVI     A,PRINTK
2331  002600' 000000  000231
2332  002601' 001000  000312          34180           JZ      STUFFM          ;THEN USE A "PRINT" TOKEN
2333  002602' 000000  002667'
2334  002603' 000000  002573'
2335  002604' 001000  000176          34200           MOV     A,M
2336  002605' 001000  000376          34220           CPI     "0"             ;SKIP NUMERICS
2337  002606' 000000  000060
2338  002607' 001000  000332          34240           JC      MUSTCR          ;SINCE CRUNCHING IS SLOW
2339  002610' 000000  002617'
2340  002611' 000000  002602'
2341  002612' 001000  000376          34260           CPI     60              ;":" ALSO PUT IN QUICKLY
2342  002613' 000000  000074
```



```
2343  002614' 001000  000332          34280           JC      STUFFM
2344  002615' 000000  002667'
2345  002616' 000000  002610'
2346  002617'                          34300   MUSTCR: >
2347  002617' 001000  000325          34320           PUSH    D               ;SAVE STORE POINTER
2348                                   34340   IFE     STRING,<
2349                                   34360           MVI     B,0>            ;INIT RESERVED WORD COUNT
2350  002620' 001000  000021          34380           LXI     D,RESLST-1      ;INIT TO RESERVED WORD LIST
2351  002621' 000000  000171'
2352  002622' 000000  002615'
2353  002623' 001000  000345          34400           PUSH    H               ;SAVE IN BUF POINTER
2354  002624' 001000  000076          34420           XWD     "01000,"076     ;"MVI" AROUND CHRGET
2355  002625' 001000  000327          34440   NXTRES: CHRGET                  ;GET CHAR FROM LINE
2356  002626' 001000  000023          34460           INX     D               ;BUMP DEPOSIT POINTER
2357  002627' 001000  000032          34480   RESER:  LDAX    D               ;GET A BYTE FROM RESERVED WORD LIST
2358  002630' 001000  000346          34500           ANI     127             ;GET RID OF SIGN BIT
2359  002631' 000000  000177
2360  002632' 001000  000312          34520           JZ      TABEND          ;END OF RESERVED WORD TABLE
2361  002633' 000000  002653'
2362  002634' 000000  002621'
2363  002635' 001000  000276          34540           CMP     M               ;TWO CHARS THE SAME?
2364  002636' 001000  000302          34560           JNZ     NTHIS           ;NO, DIFFERENT
2365  002637' 000000  002736'
2366  002640' 000000  002633'
2367  002641' 001000  000032          34580           LDAX    D               ;GET RESERVED WORD BYTE
2368  002642' 001000  000267          34600           ORA     A               ;SET CONDITION CODES
2369  002643' 001000  000362          34620           JP      NXTRES          ;IF SIGN SET, RESERVED WORD FOUND
2370  002644' 000000  002626'
2371  002645' 000000  002637'
2372  002646' 001000  000361          34640   FOUND:  POP     PSW             ;TAKE OFF GARBAGE ORIG POINTER
2373  002647' 001000  000170          34660           MOV     A,B             ;GET RESERVED WORD #
2374  002650' 001000  000366          34680           ORI     128             ;SET MSB TO FLAG AS RESERVED WORD
2375  002651' 000000  000200
2376  002652' 001000  000362          34700           XWD     "01000,"0362    ;"JP" AROUND THE POP H AND MOV A,M
2377  002653' 001000  000341          34720   TABEND: POP     H               ;GET BACK ORIG POINTER
2378  002654' 001000  000176          34740           MOV     A,M             ;GET BACK ORIG CHAR
2379  002655' 001000  000321          34760           POP     D               ;GET STUFF POINTER BACK
2380                                   34780   IFE     LENGTH-2,<
2381  002656' 001000  000353          34800           XCHG                    ;[H,L]=STUFF POINTER
2382  002657' 001000  000376          34820           CPI     ELSETK          ;HAVE TO PUT A HIDDEN
2383  002660' 000000  000220
2384                                   34840                                   ;COLON IN FRONT OF "ELSE"S
2385  002661' 001000  000066          34860           MVI     M,":"           ;STORE IT
2386  002662' 000000  000072
2387  002663' 001000  000314          34880           CZ      INXHRT##        ;ADVANCE POINTER ON "ELSE"
2388  002664' 000000  000000*
2389  002665' 000000  002644'
2390                                   34900                                   ;SO ONLY ON "ELSE" THE COLON IS NOT OVERWRITTEN
2391
2392  002666' 001000  000353          34920           XCHG>                   ;[D,E]=STUFF POINTER
2393  002667' 001000  000043          34940   STUFFM: INX     H               ;ENTRY TO BUMP [H,L]
2394  002670' 001000  000022          34960           STAX    D               ;SAVE CHARACTER IN CRUNCHED LINE
2395  002671' 001000  000023          34980           INX     D               ;BUMP SAVE POINTER
```

```
2396  002672' 001000  000014        35000            INR     C               ;BUMP CRUNCHED CHARACTER COUNT
2397                                  35020    IFN     STRING,<
2398  002673' 001000  000326        35040            SUI     ":"             ;SEE IF IT IS A COLON
2399  002674' 000000  000072
2400  002675' 001000  000312        35060            JZ      COLIS           ;IF SO ALLOW CRUNCHING AGAIN
2401  002676' 000000  002705'
2402  002677' 000000  002664'
2403  002700' 001000  000376        35080            CPI     DATATK-":"
2404  002701' 000000  000111
2405  002702' 001000  000302        35100            JNZ     NODATT          ;SEE IF IT IS A DATA TOKEN
2406  002703' 000000  002716'
2407  002704' 000000  002676'
2408  002705' 001000  000062        35120    COLIS:  STA     DORES           ;SETUP FLAG
2409  002706' 000000  001544'
2410  002707' 000000  002703'
2411  002710' 001000  000326        35140    NODATT: SUI     REMTK-":">
2412  002711' 000000  000124
2413                                  35160    IFE     STRING,<SUI     REMTK>  ;WAS IT A REM STATEMENT
2414  002712' 001000  000302        35180            JNZ     KLOOP           ;KEEP LOOPING
2415  002713' 000000  002544'
2416  002714' 000000  002706'
2417  002715' 001000  000107        35200            MOV     B,A             ;REM DOESN'T STOP ON ":", ONLY ON A ZERO
2418  002716' 001000  000176        35220    STR1:   MOV     A,M             ;GET A CHAR
2419  002717' 001000  000267        35240            ORA     A               ;SET CONDITION CODES
2420  002720' 001000  000312        35260            JZ      CRDONE          ;IF END OF LINE THEN DONE
2421  002721' 000000  002753'
2422  002722' 000000  002713'
2423  002723' 001000  000270        35280            CMP     B               ;END OF GOBBLE
2424  002724' 001000  000312        35300            JZ      STUFFH          ;IF YES, DONE WITH STRING
2425  002725' 000000  002667'
2426  002726' 000000  002721'
2427  002727' 001000  000043        35320    STRNG:  INX     H               ;INCREMENT TEXT POINTER
2428  002730' 001000  000022        35340            STAX    D               ;STORE CHAR
2429  002731' 001000  000014        35360            INR     C               ;BUMP COUNT
2430  002732' 001000  000023        35380            INX     D               ;AND POINTER
2431  002733' 001000  000303        35400            JMP     STR1            ;KEEP LOOPING
2432  002734' 000000  002716'
2433  002735' 000000  002725'
2434
2435  002736' 001000  000341        35440    NTHIS:  POP     H               ;RESTORE TEXT STRING
2436  002737' 001000  000345        35460            PUSH    H               ;AND SAVE IT BACK
2437  002740' 001000  000004        35480            INR     B               ;INCREMENT RESERVED WORD #
2438  002741' 001000  000353        35500            XCHG                    ;RESLST POINTER INTO [H,L]
2439  002742' 001000  000266        35520    NTHIS1: ORA     M               ;TEST BITS IN RESERVED WORD LIST
2440  002743' 001000  000043        35540            INX     H
2441  002744' 001000  000362        35560            JP      NTHIS1          ;SKIP MORE
2442  002745' 000000  002742'
2443  002746' 000000  002734'
2444  002747' 001000  000353        35580            XCHG                    ;RESLST POINTER INTO [D,E]
2445                                  35600                                    ;TEXT POINTER INTO [H,L]
2446  002750' 001000  000303        35620            JMP     RESER           ;DONE, MOVE TO NEXT RESERVED WORD
2447  002751' 000000  002627'
2448  002752' 000000  002745'
```



```
2449  002753' 001000  000041        35640    CRDONE: LXI     H,BUFMIN        ;LEAVE WITH [H,L] POINTER TO START OF LINE
2450  002754' 000000  001430'
2451  002755' 000000  002751'
2452  002756' 001000  000022        35660            STAX    D               ;NEED THREE 0'S ON THE END
2453  002757' 001000  000023        35680            INX     D               ;ONE FOR END-OF-LINE
2454  002760' 001000  000022        35700            STAX    D               ;AND 2 FOR A ZERO LINK
2455  002761' 001000  000023        35720            INX     D               ;SINCE IF THIS IS A DIRECT STATEMENT
2456  002762' 001000  000022        35740            STAX    D               ;ITS END MUST LOOK LIKE THE END OF A PROGRAM
2457  002763' 001000  000311        35760            RET                     ;END OF CRUNCHING
2458                                  35780    ;
2459                                  35800    ; THIS IS THE LINE INPUT ROUTINE
2460                                  35820    ; IT READS CHARACTERS INTO BUF USING ^ AS THE
2461                                  35840    ; CHARACTER DELETE CHARACTER AND # AS THE LINE DELETE CHARACTER
2462                                  35860    ; IF MORE THAN BUFLEN CHARACTER ARE TYPED, NO ECHOING
2463                                  35880    ; IS DONE UNTIL A ^ # OR CARRIAGE-RETURN IS TYPED.
2464                                  35900    ; CONTROL-G WILL BE TYPED FOR EACH EXTRA CHARACTER.
2465                                  35920    ; THE ROUTINE IS ENTERED AT INLIN
2466                                  35940    ;
2467  002764' 001000  000005        35960    LINLIN: DCR     B               ;BACK ARROW SO DECREMENT COUNT
2468  002765' 001000  000053        35980            DCX     H               ;BACK UP POINTER
2469                                  36000    IFN     REALIO,<
2470  002766' 001000  000337        36020            OUTCHR>
2471  002767' 001000  000302        36040            JNZ     INLINC          ;NOT TOO MANY SO CONTINUE
2472  002770' 000000  003003'
2473  002771' 000000  002754'
2474  002772'                       36060    INLINN: IFN     REALIO,<
2475  002772' 001000  000337        36080            OUTCHR>                 ;PRINT THE #, OR A SECOND ^ IF THERE
2476                                  36100                                    ;WERE TOO MANY
2477  002773' 001000  000315        36120            CALL    CRDO            ;TYPE A CRLF
2478  002774' 000000  004523'
2479  002775' 000000  002770'
2480  002776' 001000  000041        36140    INLIN:  LXI     H,BUF
2481  002777' 000000  001431'
2482  003000' 000000  002774'
2483  003001' 001000  000006        36160            MVI     B,1             ;CHARACTER COUNT
2484  003002' 000000  000001
2485  003003' 001000  000315        36180    INLINC: CALL    INCHR           ;GET A CHARACTER
2486  003004' 000000  003126'
2487  003005' 000000  002777'
2488                                  36182    IFN     LENGTH,<
2489  003006' 001000  000376        36184            CPI     7               ;IS IT BOB ALBRECHT RINGING THE BELL
2490  003007' 000000  000007
2491  003010' 001000  000312        36186            JZ      GOODCH>         ;FOR SCHOOL KIDS?
2492  003011' 000000  003044'
2493  003012' 000000  003004'
2494  003013' 001000  000376        36200            CPI     13              ;IS IT A CARRIAGE RETURN?
2495  003014' 000000  000015
2496  003015' 001000  000312        36220            JZ      FININL          ;IF SO FINISH UP
2497  003016' 000000  004516'
2498  003017' 000000  003011'
2499  003020' 001000  000376        36240            CPI     32              ;CHECK FOR FUNNY CHARACTERS
2500  003021' 000000  000040
2501  003022' 001000  000332        36260            JC      INLINC
```

```
2502  003025' 000000  003003'
2503  003026' 000000  003016'
2504  003027' 001000  000376          36280           CPI     125
2505  003028' 000000  000175
2506  003029' 001000  003022          36300           JNC     INLINC          ;BIG ONES BAD TOO
2507  003030' 000000  003003'
2508  003031' 000000  003023'
2509  003032' 001000  000376          36320           CPI     "@"             ;LINE DELETE?
2510  003033' 000000  000100
2511  003034' 001000  000312          36340           JZ      INLINN
2512  003035' 000000  00277E'
2513  003036' 000000  003030'
2514  003037' 001000  000376          36360           CPI     "_"             ;CHARACTER DELETE?
2515  003040' 000000  000137
2516  003041' 001000  000312          36380           JZ      LINLIN
2517  003042' 000000  002764'
2518  003043' 000000  003035'
2519  003044' 001000  000117          36400   GOODCH: MOV     C,A
2520  003045' 001000  000170          36420           MOV     A,B
2521  003046' 001000  000376          36440           CPI     BUFLEN
2522  003047' 000000  000110
2523  003050' 001000  000076          36460           MVI     A,7             ;GET A BELL IN CASE LINE TOO LONG.
2524  003051' 000000  000007
2525  003052' 001000  003322          36480           JNC     OUTBEL          ;LINE TOO LONG, RING BELL.
2526  003053' 000000  003061'
2527  003054' 000000  003042'
2528  003055' 001000  000171          36500           MOV     A,C
2529  003056' 001000  000161          36520           MOV     M,C             ;STORE THIS CHARACTER
2530  003057' 001200  000043          36540           INX     H
2531  003060' 001000  000004          36560           INR     B
2532  003061'                         36580   OUTBEL:
2533                                  36600           IFN     REALIO,<
2534  003061' 001000  000337          36620           OUTCHR>
2535  003062' 001000  000303          36640           JMP     INLINC
2536  003063' 000000  003003'
2537  003064' 000000  003053'
2538  003065'                         36660   OUTCON:
2539                                  36680           IFN     CONTRW,<
2540  003065' 001000  000302          36700           JNZ     PPSWRT>         ;NO, DO OUTPUT
2541  003066' 000000  010024'
2542  003067' 000000  003063'
2543                                  36720           IFN     REALIO,<        ;MITS I/O
2544                                  36740           IFN     LPTSW,<
2545                                  36760           LDA     PRTFLG          ;SEE IF WE WANT TO TALK TO LPT
2546                                  36780           ORA     A               ;TEST BITS
2547                                  36800           JZ      TTYCHK          ;IF ZERO THEN NOT
2548                                  36820           POP     PSW             ;GET CHARACTER WE WANT TO PRINT
2549                                  36840           PUSH    PSW
2550                                  36860           CPI     13              ;IS IT CARRIAGE RETURN?
2551                                  36880           CZ      PRINTW          ;FORCE OUT A LINE
2552                                  36900           JC      PPSWRT          ;IF FUNNY CONTROL CHARACTER (LF) DO NOTHING
2553                                  36920           LDA     LPTPOS          ;WHERE ARE WE?
2554                                  36940           CPI     LPTLEN          ;AT THE END?
```



```
2555                                  36960           CNC     PRINTW          ;YES, START OVER
2556                                  36980           INR     A
2557                                  37000           STA     LPTPOS
2558                                  37020   LPTWAT: IN      2
2559                                  37040           ANI     2
2560                                  37060           JZ      LPTWAT
2561                                  37080           POP     PSW
2562                                  37100           OUT     3               ;SEND OUT CHAR
2563                                  37120           RET                     ;RETURN
2564                                  37140   PATLPT: BLOCK   20
2565                                  37160   FINLPT: XRA     A               ;RESET PRINT FLAG SO OUTPUT
2566                                  37180           STA     PRTFLG          ;GOES TO THE TERMINAL
2567                                  37200           LDA     LPTPOS          ;SEE IF ANY LEFTOVERS MUST BE
2568                                  37220           ORA     A               ;FORCED OUT
2569                                  37240           RZ                      ;BY LOOKING AT LPTPOS
2570                                  37260   PRINTW: IN      2               ;MAKE SURE LAST PRINT
2571                                  37280           ANI     2               ;FINISHED BY TESTING DONE
2572                                  37300           JZ      PRINTW          ;BIT
2573                                  37320   ; SEE IF BUFFER MUST BE EMPTIED
2574                                  37340           LDA     LPTPOS
2575                                  37360           ORA     A               ;CHARACTERS IN THE BUFFER?
2576                                  37380           JNZ     PRINTR          ;IF SO DON'T CLEAR THE BUFFER
2577                                  37400           MVI     A,4             ;OTHERWISE BUFFER MUST BE EMPTIED
2578                                  37420           OUT     2               ;CLEAR THE BUFFER
2579                                  37440                                   ;TO PRINT A BLANK LINE
2580                                  37460   PRINTR: MVI     A,1             ;TELL LPT TO PRINT
2581                                  37480           OUT     2               ;STATUS REG
2582                                  37500           DCR     A               ;[A]=0
2583                                  37520           STA     LPTPOS          ;RESET LINE PRINT POSITION
2584                                  37540           RET>
2585  003076'                         37560   TTYCHR:>
2586                                  37580           IFN     STRING,<
2587  003076' 001000  000361          37600           POP     PSW             ;GET THE CHARACTER
2588  003077' 001000  000365          37620           PUSH    PSW             ;AND SAVE IT AGAIN
2589  003078' 001000  000376          37640           CPI     32              ;IS THIS A MEANINGFUL CHARACTER?
2590  003079' 000000  000040
2591  003074' 001000  000332          37660           JC      TRYOUT>         ;IF IT'S A NON-PRINTING CHARACTER
2592  003075' 000000  003113'
2593  003076' 000000  003066'
2594                                  37680                                   ;DON'T INCLUDE IT IN TTYPOS
2595                                  37700           IFN     LENGTH!CONTRW!LPTSW,<
2596  003077' 001000  000072          37720           LDA     TTYPOS>         ;SEE IF PRINT HEAD IS AT THE END OF THE LINE
2597  003100' 000000  000047'
2598  003101' 000000  003075'
2599                                  37740           CPI     LINLEN          ;MODIFIED BY "TERMINAL WIDTH?" QUESTION IN INIT
2600  003102' 001000  000376
2601  003103' 000000  000110
2602                  003103'         37760   LINPT1==:,-1
2603  003104' 001000  000314          37780           CZ      CRDO            ;TYPE CRLF AND SET TTYPOS AND [A]=0 IF SO
2604  003105' 000000  004523'
2605  003106' 000000  003100'
2606  003107' 001000  000074          37800           INR     A               ;INCREMENT TTYPOS SINCE WE'RE
2607                                  37820                                   ;GOING TO PRINT A CHARACTER.
```

```
 2608  003110' 001000  000062          37840           STA     TTYPOS          ;STORE NEW PRINT HEAD POSITION
 2609  003111' 000000  000047'
 2610  003112' 000000  003105'
 2611  003113'                         37860   TRYOUT:
 2612                                  37880   IFN     REALIO,<
 2613  003113' 001000  000333          37900   NOPRIN: IN      0               ;GET STATUS
 2614  003114' 000000  000000
 2615                  003114'         37920   CNLCA1==.-1                     ;CONSOLE COMMAND CHANGE LOC
 2616  003115' 001000  000346          37940           ANI     ODONE           ;OK TO SEND CHAR
 2617  003116' 000000  000200
 2618  003117' 001000  000302          37960           JNZ     NOPRIN>         ;KEEP LOOPING
 2619  003120' 000000  003113'
 2620  003121' 000000  003111'
 2621  003122' 001000  000361          37980           POP     PSW             ;GET CHARACTER BACK
 2622  003123' 001000  000323          38000           OUT     TTOCHN          ;SEND OUT THE CHAR
 2623  003124' 000000  000001
 2624                  003124'         38020   CNLCB1==.-1                     ;CONSOLE COMMAND CHANGE LOC
 2625  003125' 001000  000311          38040           RET                     ;RETURN FROM OUTCHR
 2626
 2627
 2628  003126'                         38100   INCHR:  IFN     REALIO,<
 2629  003126' 001000  000333          38120   TRYIN:  IN      0               ;GET STATUS
 2630  003127' 000000  000000
 2631                  003127'         38140   CNLCA2==.-1                     ;CONSOLE COMMAND CHANGE LOC
 2632  003130' 001000  000346          38160           ANI     IDONE           ;TEST BIT
 2633  003131' 000000  000001
 2634  003132' 001000  000302          38180           JNZ     TRYIN>          ;GO BACK & DO IT AGAIN
 2635  003133' 000000  003126'
 2636  003134' 000000  003120'
 2637  003135' 001000  000333          38200           IN      TTICHN          ;GET A CHAR
 2638  003136' 000000  000001
 2639                  003136'         38220   CNLCB2==.-1                     ;CONSOLE COMMAND CHANGE LOC
 2640  003137' 001000  000346          38240           ANI     127             ;GET RID OF PARITY BIT
 2641  003140' 000000  000177
 2642                                  38260   IFN     CONTRW,<
 2643  003141' 001000  000376          38280           CPI     CONTW           ;IS IT SUPRESS OUTPUT?
 2644  003142' 000000  000017
 2645  003143' 001000  000300          38300           RNZ
 2646  003144' 001000  000072          38320           LDA     CNTWFL
 2647  003145' 000000  001541'
 2648  003146' 000000  003133'
 2649  003147' 001000  000057          38340           CMA                     ;COMPLEMENT ITS STATE
 2650  003150' 001000  000062          38360           STA     CNTWFL>         ;SAVE BACK
 2651  003151' 000000  001541'
 2652  003152' 000000  003145'
 2653  003153' 001000  000311          38380           RET
 2654                                  38400   PAGE
```



```
 2655                                  38420   SUBTTL  THE "LIST" COMMAND
 2656
 2657                                  38460   IFN     LENGTH-2,<
 2658                                  38480   IFN     LPTSW,<
 2659                                  38500   LLIST:  MVI     A,1             ;GET NON ZERO VALUE
 2660                                  38520           STA     PRTFLG>         ;SAVE IN I/O FLAG
 2661                                  38540   LIST:   CALL    LINGET          ;GET LINE NUMBER INTO [D,E]
 2662                                  38560           RNZ                     ;MUST BE A TERMINATOR OR ERROR
 2663                                  38580           POP     B               ;GET RID OF NEWSTT RETURN ADDR
 2664                                  38600           CALL    FNDLIN          ;FIND LINE GREATER THAN OR EQUAL TO [D,E]
 2665                                  38620           PUSH    B               ;SAVE START POINTER
 2666                                  38640   LIST4:  POP     H               ;GET POINTER TO LINE
 2667                                  38660           PUSHM                   ;PUSH LINK
 2668                                  38680           POP     B               ;TAKE OFF FOR A SECOND
 2669                                  38700           MOV     A,B             ;SEE IF END OF CHAIN
 2670                                  38720           ORA     C
 2671                                  38740           JZ      READY
 2672                                  38760   IFN     LISTEN,<
 2673                                  38780           CALL    ISCNTC>         ;CHECK FOR CONTROL-C
 2674                                  38800           PUSH    B               ;PUT BACK ON
 2675                                  38820           CALL    CRDO            ;DO CRLF TO START OUT
 2676                                  38840           PUSHM                   ;PUSH LINE #
 2677                                  38860           XTHL                    ;GET LINE # INTO [H,L]
 2678                                  38880                                   ;AND WE WANT [H,L] ON THE STACK
 2679                                  38900           CALL    LINPRT          ;PRINT AS INT WITOUT LEADING SPACE
 2680                                  38920           MVI     A," "
 2681                                  38940   PRIT4:  POP     H               ;RESTORE POINTER TO START OF TEXT
 2682                                  38960   PLOOP:  OUTCHR                  ;ALWAYS A SPACE AFTER THE LINE #
 2683                                  38980           MOV     A,M             ;GET A CHARACTER FROM LINE,
 2684                                  39000           ORA     A               ;IS IT A RESERVED WORD
 2685                                  39020           INX     H               ;INCREMENT POINTER INTO TEXT
 2686                                  39040           JZ      LIST4           ;ZERO, END OF LINE, GET NEXT LINE
 2687                                  39060           JP      PLOOP           ;REGULAR CHAR, JUST PRINT IT
 2688                                  39080           SUI     127             ;GET RID OF SIGN BIT AND ADD ONE
 2689                                  39100           MOV     C,A             ;GET RESERVED WORD # IN C
 2690                                  39120           PUSH    H               ;SAVE CURRRENT POSIT
 2691                                  39140           LXI     D,RESLST        ;GET RESLST POINTER,
 2692                                  39160   RESRCH: PUSH    D               ;SAVE
 2693                                  39180
 2694                                  39200   RESCR1: LDAX    D               ;GET CHARACTER FROM RESLST
 2695                                  39220           INX     D               ;BUMP RESLST POINTER
 2696                                  39240           ORA     A               ;TEST BITS
 2697                                  39260           JP      RESCR1          ;NOT AT END OF RESERVED WORD YET
 2698                                  39280           DCR     C               ;DECREMENT CHAR
 2699                                  39300           POP     H               ;POP START POINTER HERE
 2700                                  39320           JNZ     RESRCH          ;NOT AT END OF RESLST YET,
 2701                                  39340   ;HERE WHEN FOUND RIGHT RESERVED WORD
 2702                                  39360   PRIT3:  MOV     A,M             ;GET A CHARACTER FROM RESERVED WORD
 2703                                  39380           ORA     A               ;SET CONDITION CODES
 2704                                  39400           JM      PRIT4
 2705                                  39420           OUTCHR
 2706                                  39440           INX     H               ;BUMP RESLST POINTER
 2707                                  39460           JMP     PRIT3>          ;PRINT THE REST
```

```
 2708
 2709
 2710                                    39520   PAGE
```



```
 2711                                    39540   SUBTTL "FOR" STATEMENT
 2712                                    39560   ;
 2713                                    39580   ; NOTE:
 2714                                    39600   ;
 2715                                    39620   ; A FOR ENTRY ON THE STACK HAS THE FOLLOWING FORMAT:
 2716                                    39640   ;
 2717                                    39660   ;
 2718                                    39700   ; LOW ADDRESS
 2719                                    39720   ;       TOKEN (FORTK IN HIGH BYTE)  1 BYTE
 2720                                    39740   ;       A POINTER TO THE LOOP VARIABLE  2 BYTES
 2721                                    39760   ;       A BYTE REFLECTING THE SIGN OF THE INCREMENT 1 BYTE
 2722                                    39780   ;       THE STEP 4 BYTES
 2723                                    39800   ;       THE UPPER VALUE 4 BYTES
 2724                                    39820   ;       THE LINE # OF THE "FOR" STATEMENT 2 BYTES
 2725                                    39840   ;       A TEXT POINTER INTO THE "FOR" STATEMENT 2 BYTES
 2726                                    39860   ; HIGH ADDRESS
 2727                                    39880   ;
 2728                                    39900   ; TOTAL 16 BYTES
 2729                                    39920   ;
 2730
 2731  003154'                           39960   FOR:    IFN     LENGTH,<
 2732  003154' 001000  000076            39980           MVI     A,100
 2733  003155' 000000  000144
 2734  003156' 001000  000062            40000           STA     SUBFLG>         ;DONT RECOGNIZE SUBSCRIPTED VARIABLES
 2735  003157' 000000  001601'
 2736  003160' 000000  003151'
 2737  003161' 001000  000315            40020           CALL    LET             ;READ THE VARIABLE AND ASSIGN IT
 2738  003162' 000000  004131'
 2739  003163' 000000  003157'
 2740                                    40040                                   ;THE CORRECT INTIAL VALUE
 2741                                    40060                                   ;AND STORE A POINTER
 2742                                    40080                                   ;TO THE VARIABLE IN (TEMP)
 2743  003164' 001000  000343            40100           XTHL                    ;SAVE TEXT PTR ON THE STACK
 2744  003165' 001000  000315            40120           CALL    FNDFOR          ;MUST HAVE VARIABLE POINTER IN (D,E)
 2745  003166' 000000  001744'
 2746  003167' 000000  003162'
 2747  003170' 001000  000321            40140           POP     D               ;(D,E)=TEXT POINTER
 2748  003171' 001000  000302            40160           JNZ     NOTOL           ;IF NO MATCHING ENTRY, DON'T
 2749  003172' 000000  003176'
 2750  003173' 000000  003166'
 2751                                    40180                                   ;ELIMINATE ANYTHING
 2752  003174' 001000  000011            40200           DAD     B               ;IN THE CASE OF "FOR"
 2753                                    40220                                   ;WE ELIMINATE THE MATCHING ENTRY
 2754                                    40240                                   ;AS WELL AS EVERYTHING AFTER IT
 2755  003175' 001000  000371            40260           SPHL                    ;DO THE ELIMINATION
 2756                                    40280                                   ;SINCE A MATCHING ENTRY WAS FOUND
 2757  003176' 001000  000353            40300   NOTOL:  XCHG                    ;(H,L)=TEXT POINTER
 2758  003177' 001000  000315            40320           CALL    GETSTK
 2759  003200' 000000  002624'
 2760  003201' 000000  003172'
 2761  003202' 000000  000010            40340           8                       ;MAKE SURE 16 BYTES ARE AVAILABLE
 2762                                    40360                                   ;OFF OF THE STACK
 2763  003203' 001000  000345            40380           PUSH    H               ;REALLY SAVE THE TEXT POINTER
```

```
                                40400           CALL    DATA            ;GET AN [H,L] THAT POINTS
                                40420                                   ;JUST BEYOND THE TERMINATOR
003207' 001000  000343          40440           XTHL                    ;PUT [H,L] POINTER TO TERMINATOR ON THE STACK
                                40460                                   ;AND RESTORE [H,L] AS TEXT POINTER AT
                                40480                                   ;VARIABLE NAME
003210' 001000  000345          40500           PUSH    H               ;PUSH THE TEXT POINTER ONTO THE STACK
003211' 001000  000052          40520           LHLD    CURLIN          ;[H,L] GET THE CURRENT LINE #
003212' 000000  001607'
003213' 000000  003205'
003214' 001000  000343          40540           XTHL                    ;NOW THE CURRENT LINE # IS ON THE STACK AND
                                40560                                   ;[H,L] IS THE TEXT POINTER
                                40580   IFN     LENGTH=2,<
                                40600   IFN     STRING,<CALL    CHKNUM>>
003215' 001000  000317          40620           SYNCHK  TOTK            ;"TO" IS NECESSARY
003216' 000000  000241
                                40640   IFN     LENGTH=2,<
                                40660           CALL    FRMNUM>         ;READ FINAL VALUE
                                40680   IFE     LENGTH=2,<
003217' 001000  000315          40700           CALL    FRMEVL>
003220' 000000  005336'
003221' 000000  003212'
003222' 001000  000345          40720           PUSH    H               ;SAVE THE TEXT POINTER
                                40740   IFE     LENGTH=2,<
003223' 001000  000315          40760           CALL    FRCSNG>
003224' 000000  000670*
003225' 000000  003220'
003226' 001000  000315          40780           CALL    MOVRF           ;GET THE STUFF
003227' 000000  000000*
003230' 000000  003224'
003231' 001000  000341          40800           POP     H               ;REGAIN TEXT POINTER
003232' 001000  000305          40820           PUSH    B               ;OPPOSITE OF PUSHR
003233' 001000  000325          40840           PUSH    D               ;SAVE THE SIGN OF THE INCREMENT
003234' 001000  000001          40860           LXI     B,$CODE+^O201*256
003235' 000000  100400'
003236' 000000  003227'
003237' 001000  000121          40880           MOV     D,C
003240' 001000  000132          40900           MOV     E,D             ;GET 1,0 IN THE REGISTERS
003241' 001000  000176          40920           MOV     A,M             ;GET TERMINATING CHARACTER
003242' 001000  000376          40940           CPI     STEPTK          ;DO WE HAVE "STEP" ?
003243' 000000  000247
003244' 001000  000076          40960           MVI     A,1             ;SETUP DEFAULT SIGN
003245' 000000  000001
003246' 001000  000302          40980           JNZ     ONEON           ;PUSH SOME CONSTANTS ON IF NOT
003247' 000000  003265'
003250' 000000  003235'
                                41000   IFN     LENGTH=2,<
                                41020   IFN     STRING,<
                                41040           CHRGET
                                41060           CALL    FRMNUM>>        ;READ THE STEP
                                41080   IFE     <LENGTH=2>&STRING,<
003251' 001000  000315          41100           CALL    FRMCHK>         ;DON'T NEED TO CHECK THE TYPE
```



```
003252' 000000  005337'
003253' 000000  003247'
003254' 001000  000345          41120           PUSH    H
                                41140   IFE     LENGTH=2,<
003255' 001000  000315          41160           CALL    FRCSNG>
003256' 000000  003224*
003257' 000000  003252'
003260' 001000  000315          41180           CALL    MOVRF           ;SET UP THE REGISTERS
003261' 000000  003227*
003262' 000000  003256'
003263' 001000  000341          41200           POP     H
003264' 001000  000357          41220           FSIGN                   ;GET THE SIGN OF THE INCREMENT
003265' 001000  000305          41240   ONEON:  PUSH    B               ;PUT VALUE ON BACKWARDS
003266' 001000  000325          41260           PUSH    D               ;OPPOSITE OF PUSHR
003267' 001000  000365          41280   IFORON: PUSH    PSW             ;SAVE THE SIGN OF THE INCREMENT
003270' 001000  000063          41300           INX     SP              ;A ONE BYTE ENTRY ONLY
003271' 001000  000345          41320           PUSH    H
003272' 001000  000052          41340           LHLD    TEMP            ;GET THE POINTER TO THE VARIABLE BACK
003273' 000000  001623'
003274' 000000  003261'
003275' 001000  000343          41360           XTHL                    ;PUT THE POINTER TO THE VARIABLE
                                41380                                   ;ONTO THE STACK AND RESTORE THE TEXT POINTER
003276' 001000  000006          41400   NXTCON: MVI     B,FORTK         ;PUT A "FOR" TOKEN ONTO THE STACK
003277' 000000  000201
003300' 001000  000305          41420           PUSH    B
003301' 001000  000063          41440           INX     SP              ;THE "TOKEN" ONLY TAKES ONE BYTE OF
                                41460                                   ;STACK SPACE
                                41480   ;       JMP     NEWSTT          ;ALL DONE
                                41500   PAGE
```

```
                                        41520   SUBTTL NEW STATEMENT FETCHER
                                        41540   ;
                                        41560   ; BACK HERE FOR NEW STATEMENT, CHARACTER POINTED TO BY (H,L)
                                        41580   ; "I" ON END-OF-LINE, THE ADDRESS OF THIS LOCATION IS
                                        41600   ; LEFT ON THE STACK WHEN A STATEMENT IS EXECUTED SO
                                        41620   ; IT CAN MERELY DO A RETURN WHEN IT IS DONE.
                                        41640   ;
003302'                                 41660   NEWSTT:
                                        41680           IFN     LISTEN,<
                                        41700           IFN     LENGTH,<
003302' 001000  000333                  41720                   IN      0               ;CHECK FOR A CHARACTER WITHOUT
003303' 000000  000000
                                        41740                                           ;DOING A "CALL" FOR SPEED
                        003303'         41760   CNLCA4==:,=1
003304' 001000  000346                  41780                   ANI     IDONE           ;CHARACTER THERE?
003305' 000000  000001
003306' 001000  000314                  41800                   CZ      CNTCCN>         ;SEE IF IT'S CONTROL-C
003307' 000000  003465'
003310' 000000  003273'
                                        41820           IFE     LENGTH,<
                                        41840                   CALL    ISCNTC>>
                                        41860           IFN     LENGTH,<
003311' 001000  000042                  41880                   SHLD    TEMP>           ;USED BY CONTINUE AND INPUT AND CLEAR
003312' 000000  001603'
003313' 000000  003307'
                                        41900                                           ;TO REMEMBER HOW TO RESTART THIS
                                        41920                                           ;STATEMENT
                                        41940           IFN     LPTSW,<XRA      A       ;FORCE PRINT TO GO TO TTY AFTER LPRINT
                                        41960                   STA     PRTFLG>
003314' 001000  000176                  41980                   MOV     A,M             ;GET CURRENT CHARACTER
                                        42000                                           ;WHICH TERMINATED THE LAST STATEMENT
003315' 001000  000376                  42020                   CPI     ":"             ;IS IT A COLON?
003316' 000000  00007E
003317' 001000  000312                  42040                   JZ      GONE
003320' 000000  003370'
003321' 000000  003312'
003322' 001000  000267                  42060                   ORA     A
003323' 001000  000302                  42080                   JNZ     SNERR           ;MUST BE A ZERO
003324' 000000  002072'
003325' 000000  003320'
003326' 001000  000043                  42100                   INX     H
003327' 001000  000176                  42120                   MOV     A,M             ;CHECK POINTER TO SEE IF
                                        42140                                           ;IT IS ZERO, IF SO WE ARE AT THE
                                        42160                                           ;END OF THE PROGRAM
003330' 001000  000043                  42180                   INX     H
003331' 001000  000266                  42200                   ORA     M               ;OR IN HIGH PART
003332' 001000  000043                  42220                   INX     H
003333' 001000  000312                  42240                   JZ      ENDCON          ;RAN OFF THE END -- OK
003334' 000000  003501'
003335' 000000  003324'
003336' 001000  000136                  42260                   MOV     E,M
003337' 001000  000043                  42280                   INX     H
003340' 001000  000126                  42300                   MOV     D,M             ;GET LINE # IN (D,E)
```



```
003341' 001000  000353                  42320                   XCHG                    ;(H,L)=LINE #
003342' 001000  000042                  42340                   SHLD    CURLIN          ;SETUP CURLIN WITH THE CURRENT LINE #
003343' 000000  001607'
003344' 000000  003334'
                                        42360   IFE     LENGTH=2,<                      ;TRACE FEATURE
003345' 001000  000072                  42380                   LDA     TRCFLG          ;SEE IF TRACE IS ON
003346' 000000  001631'
003347' 000000  003343'
003350' 001000  000267                  42400                   ORA     A               ;NON-ZERO MEANS YES
003351' 001000  000312                  42420                   JZ      NOTTRC          ;SKIP THIS PRINTING
003352' 000000  003367'
003353' 000000  003346'
003354' 001000  000325                  42440                   PUSH    D               ;SAVE THE TEXT POINTER
003355' 001000  000076                  42460                   MVI     A,"["           ;FORMAT THE LINE NUMBER
003356' 000000  000133
003357' 001000  000337                  42480                   OUTCHR                  ;OUTPUT IT
003360' 001000  003315                  42500                   CALL    LINPRT          ;PRINT THE LINE # IN (H,L)
003361' 000000  000000*
003362' 000000  003352'
003363' 001000  000076                  42520                   MVI     A,"]"           ;SOME MORE FORMATING
003364' 000000  000135
003365' 001000  000337                  42540                   OUTCHR
003366' 001000  000321                  42560                   POP     D               ;(D,E)=TEXT POINTER
003367'                         002580  NOTTRC:>
003370' 001000  000353                  42600                   XCHG                    ;RESTORE THE TEXT POINTER
003371' 001000  000327                  42620   GONE:   CHRGET                          ;GET THE STATEMENT TYPE
003372' 001000  000021                  42640                   LXI     D,NEWSTT        ;PUSH ON A RETURN ADDRESS OF NEWSTT
003373' 000000  003302'
003374' 000000  003361'
003375' 001000  000325                  42660                   PUSH    D               ;STATEMENT
003376' 001000  000310                  42680   GONE3:  RZ                              ;IF A TERMINATOR TRY AGAIN
                                        42700                                           ;"IF" COMES HERE
003377' 001000  000326                  42720   GONE2:  SUI     ENDTK                   ;"ON ... GOTO" AND "ON ... GOSUB" COME HERE
003400' 001000  000332                  42740                   JC      LET             ;MUST BE A LET
003401' 000000  004131'
003402' 000000  003372'
                                        002040
                                        42760   NUMCMD=SCRATK-ENDTK+1
003403' 001000  000376                  42780                   CPI     NUMCMD
003404' 000000  000040
003405' 001000  000322                  42800                   JNC     SNERR           ;SOME RESERVED WORD,BUT NOT
003406' 000000  002072'
003407' 000000  003401'
                                        42820                                           ;A STATEMENT RESERVED WORD
003410' 001000  000007                  42840                   RLC                     ;MULTIPLY BY 2
003411' 001000  000117                  42860                   MOV     C,A
003412' 001000  000006                  42880                   MVI     B,0
003413' 000000  000000
003414' 001000  000353                  42900                   XCHG
003415' 001000  000041                  42920                   LXI     H,STMDSP        ;STATEMENT DISPATCH TABLE
003416' 000000  000564'
003417' 000000  003406'
003420' 001000  000011                  42940                   DAD     B               ;ADD ON OFFSET
```

```
BASIC MCS 8080  GATES/ALLEN/DAVIDOFF     MACRO 47(113) 03:12 10-SEP-75 PAGE 11-2
F3      MAC      6-SEP-64 03:11          NEW STATEMENT FETCHER

 2952  003421' 001000  000116          42960           MOV     C,M             ;PUSH THE ADDRESS TO GO TO ONTO
 2953  003422' 001000  000043          42980           INX     H               ;THE STACK
 2954  003423' 001000  000106          43000           MOV     B,M             ;PUSHM SAVES BYTES BUT NOT SPEED
 2955  003424' 001000  000305          43020           PUSH    B
 2956  003425' 001000  000353          43040           XCHG                    ;RESTORE THE TEXT POINTER
 2957                                   43060   IFE     LENGTH,<
 2958                                   43080           CHRGET                  ;EAT THE FIRST CHARACTER
 2959                                   43100           RET>                    ;GO DO THE STATEMENT
 2960                                   43120   IFN     LENGTH,<
 2961  003426' 001000  000043          43140   CHRGTR: INX     H               ;DUPLICATION OF CHRGET RST FOR SPEED
 2962  003427' 001000  000176          43160           MOV     A,M             ;SEE CHRGET RST FOR EXPLANATION
 2963  003430' 001000  000376          43180           CPI     ":"
 2964  003431' 000000  000072
 2965  003432' 001000  000320          43200           RNC>
 2966                                   43220   ;
 2967                                   43240   ; CHRCON IS THE CONTINUATION OF THE CHRGET RST
 2968                                   43260   ;
 2969  003433' 001000  000376          43280   CHRCON: CPI     " "             ;MUST SKIP SPACES
 2970  003434' 000000  000040
 2971  003435' 001000  000312          43300           JZ      CHRGTR          ;GET ANOTHER CHARACTER
 2972  003436' 000000  003420'
 2973  003437' 000000  003416'
 2974  003440' 001000  000376          43320           CPI     "0"             ;ALL CHARACTERS GREATER THAN
 2975  003441' 000000  000060
 2976                                   43340                                   ;"9" HAVE RETURNED, SO SEE IF NUMERIC
 2977  003442' 001000  000077          43360           CMC                     ;MAKE NUMERICS HAVE CARRY ON
 2978  003443' 001000  000074          43380           INR     A               ;SET ZERO IF [A]=0
 2979  003444' 001000  000075          43400           DCR     A
 2980  003445' 001000  000311          43420           RET
 2981                                   43440   PAGE
```

```
BASIC MCS 8080  GATES/ALLEN/DAVIDOFF     MACRO 47(113) 03:12 10-SEP-75 PAGE 12
F3      MAC      6-SEP-64 03:11          RESTORE,STOP,END,LINGET,CHRCON

 2982                                   43460   SUBTTL RESTORE,STOP,END,LINGET,CHRCON
 2983
 2984  003446' 001000  000353          43500   RESTOR: XCHG                    ;SAVE (H,L) IN (D,E)
 2985  003447' 001000  000052          43520           LHLD    TXTTAB
 2986  003450' 000000  001617'
 2987  003451' 000000  003436'
 2988  003452' 001000  000053          43540           DCX     H               ;INITIALIZE DATPTR TO (TXTTAB)-1
 2989  003453' 001000  000042          43560   RESFIN: SHLD    DATPTR          ;READ FINISHES COME TO RESFIN
 2990  003454' 000000  001627'
 2991  003455' 000000  003456'
 2992  003456' 001000  000353          43580           XCHG                    ;GET THE TEXT POINTER BACK
 2993  003457' 001000  000311          43600           RET
 2994
 2995                                   43640   IFN     LISTEN,<
 2996  003460' 001000  000333          43660   ISCNTC: IN      0
 2997  003461' 000000  000000
 2998                  003461'          43680   CNLCA3==,-1                     ;CONSOLE COMMAND CHANGE LOC
 2999  003462' 001000  000346          43700           ANI     IDONE
 3000  003463' 000000  000001
 3001  003464' 001000  000300          43720           RNZ                     ;IF NO CHARACTERS THEN NO ^C
 3002  003465' 001000  000315          43740   CNTCCN: CALL    INCHR
 3003  003466' 000000  003126'
 3004  003467' 000000  003454'
 3005  003470' 001000  000376          43760           CPI     3               ;STOP CHARACTER IS ^C
 3006  003471' 000000  000003
 3007                                   43780   IFE     LENGTH,<
 3008                                   43800           JMP     STOP>>
 3009                                   43820   IFN     LENGTH,<
 3010  003472' 001000  000300          43840   STOP:   RNZ                     ;RETURN IF NOT CONTROL-C AND MAKE
 3011                                   43860                                   ;SURE "STOP" STATEMENTS HAVE A TERMINATOR
 3012  003473' 001000  000366          43880           XWD     ^O1000,^O366    ;SETUP (A) AS A FLAG WHETHER
 3013                                   43900                                   ;TO TYPE THE BREAK MESSAGE
 3014  003474' 001000  000300          43920   END:    RNZ                     ;MAKE SURE "END" STATEMENTS HAVE A TERMINATOR
 3015  003475' 001000  000042          43940           SHLD    TEMP            ;SAVE FOR "CONTINUE"
 3016  003476' 000000  001605'
 3017  003477' 000000  003466'
 3018  003500' 001000  000301          43960   STPEND: POP     B
 3019  003501' 001000  000365          43980   ENDCON: PUSH    PSW             ;SAVE THE MESSAGE FLAG
 3020                                   44000                                   ;ZERO MEANS DON'T PRINT "BREAK"
 3021  003502' 001000  000052          44020           LHLD    CURLIN          ;SAVE CURLIN
 3022  003503' 000000  001607'
 3023  003504' 000000  003476'
 3024  003505' 001000  000175          44040           MOV     A,L
 3025  003506' 001000  000244          44060           ANA     H               ;SEE IF IT WAS DIRECT
 3026  003507' 001000  000074          44080           INR     A
 3027  003510' 001000  000312          44100           JZ      DIRIS           ;IF NOT SET UP FOR CONTINUE
 3028  003511' 000000  003524'
 3029  003512' 000000  003503'
 3030  003513' 001000  000042          44120           SHLD    OLDLIN          ;SAVE OLD LINE #
 3031  003514' 000000  001611'
 3032  003515' 000000  003511'
 3033  003516' 001000  000052          44140           LHLD    TEMP            ;GET POINTER TO START OF STATEMENT
 3034  003517' 000000  001603'
```

```
003520' 000000  003514'
003521' 001000  000042          44160           SHLD    OLDTXT          ;SAVE IT
003522' 000000  001613'
003523' 000000  003517'
003524'                         44180   DIRIS:
                                44200   IFN     CONTRW,<
003524' 001000  000257          44220           XRA     A
003525' 001000  000062          44240           STA     CNTWFL>         ;FORCE OUTPUT
003526' 000000  001541'
003527' 000000  003522'
003530' 001000  000361          44280           POP     PSW             ;GET BACK ^C FLAG
003531' 001000  000041          44300           LXI     H,BRKTXT        ;"BREAK"
003532' 000000  001734'
003533' 000000  003526'
003534' 001000  000302          44320           JNZ     ERRFIN          ;CALL STROUT AND FALL INTO READY
003535' 000000  002127'
003536' 000000  003532'
003537' 001000  000303          44340           JMP     READY>          ;TYPE "READY"
003540' 000000  002145'
003541' 000000  003535'
                                44360   IFE     REALIO,<
                                44380   DDT:    POP     B               ;GET RID OF NEWSTT RETURN
                                44400           HRRZ    14,JOBDDT##
                                44420           JRST    0(14)>

                                44460   IFN     LENGTH,<
003542' 001000  000300          44480   CONT:   RNZ                     ;MAKE SURE THERE IS A TERMINATOR
003543' 001000  000036          44500           MVI     E,ERRCN
003544' 000000  000021
003545' 001000  000052          44520           LHLD    OLDTXT          ;A STORED TEXT POINTER OF
003546' 000000  001613'
003547' 000000  003540'
                                44540                                   ;ZERO IS SETUP BY STKINI
                                44560                                   ;AND INDICATES THERE IS NOTHING
                                44580                                   ;TO CONTINUE
003550' 001000  000174          44600           MOV     A,H             ;"STOP","END",TYPING CRLF
003551' 001000  000265          44620           ORA     L               ;TO "INPUT" AND ^C SETUP OLDTXT
003552' 001000  000312          44640           JZ      ERROR
003553' 000000  002102'
003554' 000000  003546'
003555' 001000  000353          44660           XCHG                    ;SAVE [H,L]
003556' 001000  000052          44680           LHLD    OLDLIN
003557' 000000  001611'
003560' 000000  003553'
003561' 001000  000042          44700           SHLD    CURLIN          ;SET UP OLD LINE # AS CURRENT LINE #
003562' 000000  001607'
003563' 000000  003557'
003564' 001000  000353          44720           XCHG                    ;RESTORE [H,L]
003565' 001000  000311          44740           RET>
                                44760   IFN     LENGTH,<
003566' 001000  000315          44780   NULL:   CALL    GETBYT
003567' 000000  011020'
003570' 000000  003562'
```



```
003571' 001000  000300          44800           RNZ                     ;MAKE SURE THERE IS A TERMINATOR
003572' 001000  000074          44820           INR     A
003573' 001000  000376          44840           CPI     LINLEN          ;MAKE SURE THE NUMBER IS REASONABLE
003574' 000000  000110
                                44860                                   ;CRDO WON'T WORK IF IT ISN'T
                003574'         44880   LINPT2==.-1                     ;TERMINAL WIDTH CHANGE LOCATION
003575' 001000  000322          44900           JNC     FCERR           ;"FUNCTION CALL" ERROR
003576' 000000  010776'
003577' 000000  003567'
003600' 001000  000062          44920           STA     NULCNT          ;CHANGE NUMBER OF NULLS
003601' 000000  000046'
003602' 000000  003576'
003603' 001000  000311          44940           RET>
                                44960   IFE     LENGTH-2,<
003604' 001000  000076          44980   TON:    XWD     ^O1000,^O76     ;"MVI A," NON-ZERO QUANTITY
003605' 001000  000257          45000   TOFF:   XRA     A               ;MAKE [A]=0 FOR NO TRACE
003606' 001000  000062          45020           STA     TRCFLG          ;UPDATE THE TRACE FLAG
003607' 000000  001631'
003610' 000000  003601'
003611' 001000  000311          45040           RET>
                                45060   ;
                                45080   ;TEST FOR A LETTER / CARRY ON=NOT A LETTER
                                45100   ;                    CARRY OFF=A LETTER
                                45120   ;
003612' 001000  000176          45140   ISLET:  MOV     A,M
003613' 001000  000376          45160           CPI     "A"
003614' 000000  000101
003615' 001000  000330          45180           RC                      ;IF LESS THAN "A", RETURN EARLY
003616' 001000  000376          45200           CPI     91              ;91="Z"+1
003617' 000000  000133
003620' 001000  000077          45220           CMC
003621' 001000  000311          45240           RET
                                45260   ;
                                45280   ; INTIDX READS A FORMULA FROM THE CURRENT POSITION AND
                                45300   ; TURNS IT INTO A POSITIVE INTEGER
                                45320   ; LEAVING THE RESULT IN [D,E].  NEGATIVE ARGUMENTS
                                45340   ; ARE NOT ALLOWED. [H,L] POINTS TO THE TERMINATING
                                45360   ; CHARACTER OF THE FORMULA ON RETURN.
                                45380   ;
                                45400   IFN     LENGTH-2,<
                                45420   INTIDX: CHRGET
                                45440   INTID2: CALL    FRMNUM
                                45460   POSINT: FSIGN
                                45480           JM      FCERR           ;IF NEGATIVE BLOW HIM OUT
                                45500   DEINT:  LDA     FAC             ;SEE IF ARG GREATER THAN 32767
                                45520           CPI     144
                                45540           JC      QINT
                                45560   IFN     LENGTH,<
                                45580           MOVRI   144,128,0,0     ;REGISTERS = FLOATING -32768
                                45600           CALL    FCOMP           ;SEE IF FAC=REGISTERS
                                45620           MOV     D,C             ;SETUP D=200 E=0 FOR -32768
                                45640           RZ>                     ;WAS -32768. [D,E] IS SET UP
                                45660   ILLFUN:
```

```
                              45680   FCERR:  MVI     E,ERRFC           ;TOO BIG. FUNCTION CALL ERROR
                              45700           JMP     ERROR>
                              45720   IFE     LENGTH-2,<
003622' 001000  000327        45740   INTIDX: CHRGET
003623' 001000  000315        45760   INTID2: CALL    FRMEVL            ;EVALUATE A FORMULA
003624' 000000  005336'
003625' 000000  003607'
003626' 001000  000345        45780           PUSH    H                 ;SAVE THE TEXT POINTER
003627' 001000  000315        45800           CALL    FRCINT            ;CONVERT THE FAC TO AN INTEGER
003630' 000000  000666*
003631' 000000  003624'
003632' 001000  000174        45820           MOV     A,H               ;SEE IF THE RESULT IS NEGATIVE
003633' 001000  000267        45840           ORA     A                 ;BY LOOKING AT [H]'S MSB
003634' 001000  000372        45860           JM      FCERR             ;DON'T ALLOW NEGATIVE NUMBERS
003635' 000000  010776'
003636' 000000  003630'
003637' 001000  000353        45880           XCHG                      ;RETURN THE INTEGER IN [D,E]
003640' 001000  000341        45900           POP     H                 ;RESTORE THE TEXT POINTER
003641' 001000  000311        45920           RET>

                              45960   ;
                              45980   ; LINGET READS A LINE # FROM THE CURRENT TEXT POSITION
                              46000   ;
                              46020   ; LINE NUMBERS RANGE FROM 0 TO 65529
                              46040   ;
                              46060   ; [D,E] IS SMASHED.
                              46080   ;
                              46100   ; ANSWER RETURNED IN [D,E].
                              46120   ; [H,L] IS UPDATED TO POINT TO THE TERMINATING CHARACTER
                              46140   ; AND [A] CONTAINS THE TERMINATING CHARACTER WITH CONDITION
                              46160   ; CODES SET UP TO REFLECT ITS VALUE.
                              46180   ;
003642' 001000  000053        46200   LINGET: DCX     H
003643' 001000  000021        46220   LINGT2: LXI     D,$CODE           ;ZERO ACCUMULATED LINE #
003644' 000000  000000'
003645' 000000  003635'
003646' 001000  000327        46240   MORLIN: CHRGET
003647' 001000  000320        46260           RNC                       ;WAS IT A DIGIT
003650' 001000  000345        46280           PUSH    H
003651' 001000  000365        46300           PUSH    PSW
003652' 001000  000041        46320           LXI     H,$CODE+6552      ;SEE IF THE LINE # IS TOO BIG
003653' 000000  014630'
003654' 000000  003644'
003655' 001000  000347        46340           COMPAR
003656' 001000  000332        46360           JC      SNERR             ;YES, SYNTAX ERROR
003657' 000000  002072'
003660' 000000  003653'
003661' 001000  000142        46380           MOV     H,D               ;SAVE [D,E]
003662' 001000  000153        46400           MOV     L,E
003663' 001000  000031        46420           DAD     D
003664' 001000  000051        46440           DAD     H
003665' 001000  000031        46460           DAD     D
003666' 001000  000051        46480           DAD     H                 ;PUTTING [D,E]*10 INTO [H,L]
```



```
003667' 001000  000361        46500           POP     PSW
003670' 001000  000326        46520           SUI     "0"
003671' 000000  000060
003672' 001000  000137        46540           MOV     E,A
003673' 001000  000026        46560           MVI     D,0
003674' 000000  000000
003675' 001000  000031        46580           DAD     D                 ;ADD THE NEW DIGIT
003676' 001000  000353        46600           XCHG
003677' 001000  000341        46620           POP     H                 ;GET BACK TEXT POINTER
003700' 001000  000303        46640           JMP     MORLIN
003701' 000000  003646'
003702' 000000  003657'
                              46660   IFN     LENGTH,<
003703' 001000  000312        46680   CLEAR:  JZ      CLEARC            ;IF NO FORMULA JUST CLEAR
003704' 000000  002443'
003705' 000000  003701'
003706' 001000  000315        46700           CALL    INTID2            ;GET AN INTEGER INTO [D,E]
003707' 000000  003623'
003710' 000000  003704'
003711' 001000  000053        46720           DCX     H
003712' 001000  000327        46740           CHRGET                    ;SEE IF ITS THE END
003713' 001000  000300        46760           RNZ                       ;SHOULD FINISH THERE
003714' 001000  000345        46780           PUSH    H                 ;SAVE TXTPTR
003715' 001000  000052        46800           LHLD    MEMSIZ            ;GET HIGHEST ADDRESS
003716' 000000  001545'
003717' 000000  003707'
003720' 001000  000175        46820           MOV     A,L               ;SUBTRACT [H,L]-[D,E] INTO [D,E]
003721' 001000  000223        46840           SUB     E
003722' 001000  000137        46860           MOV     E,A
003723' 001000  000174        46880           MOV     A,H
003724' 001000  000232        46900           SBB     D
003725' 001000  000127        46920           MOV     D,A
003726' 001000  000332        46940           JC      SNERR             ;WANTED MORE THAN TOTAL!
003727' 000000  002072'
003730' 000000  003716'
003731' 001000  000052        46960           LHLD    VARTAB            ;TOP LOCATION IN USE
003732' 000000  001621'
003733' 000000  003727'
003734' 001000  000001        46980           LXI     B,$CODE+40        ;LEAVE BREATHING ROOM
003735' 000000  000050'
003736' 000000  003732'
003737' 001000  000011        47000           DAD     B
003740' 001000  000347        47020           COMPAR                    ;ROOM?
003741' 001000  000322        47040           JNC     OMERR             ;NO, DON'T EVEN CLEAR
003742' 000000  002057'
003743' 000000  003735'
003744' 001000  000353        47060           XCHG                      ;NEW STACK LOCATION [H,L]
003745' 001000  000042        47080           SHLD    STKTOP            ;SET UP NEW STACK LOCATION
003746' 000000  001615'
003747' 000000  003742'
003750' 001000  000341        47100           POP     H                 ;REGAIN THE TEXT POINTER
003751' 001000  000303        47120           JMP     CLEARC>           ;GO CLEAR
003752' 000000  002443'
```

```
 3247  003753' 000000  003746'
 3248                          47140   PAGE
```



```
 3249                          47160   SUBTTL  RUN,GOTO,GOSUB,RETURN
 3250                          47180   IFN     LENGTH,<
 3251  003754' 001000  000312  47200   RUN:    JZ      RUNC            ;NO LINE # ARGUMENT
 3252  003755' 000000  002437'
 3253  003756' 000000  003752'
 3254                          47220                                   ;CLEAN UP,SET [H,L]=[TXTTAB]+1 AND
 3255                          47240                                   ;RETURN TO NEWSTT
 3256  003757' 001000  000315  47260           CALL    CLEARC          ;CLEAN UP -- RESET THE STACK
 3257  003760' 000000  002443'
 3258  003761' 000000  003755'
 3259                          47280                                   ;DATPTR,VARIABLES ...
 3260                          47300                                   ;[H,L] IS THE ONLY THING PRESERVED
 3261  003762' 001000  000001  47320           LXI     B,NEWSTT
 3262  003763' 000000  003302'
 3263  003764' 000000  003760'
 3264  003765' 001000  000303  47340           JMP     RUNC2>          ;PUT "NEWSTT" ON AND FALL INTO "GOTO"
 3265  003766' 000000  004007'
 3266  003767' 000000  003763'
 3267                          47360   ;
 3268                          47380   ; A GOSUB ENTRY ON THE STACK HAS THE FOLLOWING FORMAT
 3269                          47400   ;
 3270                          47420   ; LOW ADDRESS
 3271                          47440   ;
 3272                          47460   ;       A TOKEN EQUAL TO GOSUTK 1 BYTE
 3273                          47480   ;       THE LINE # OF THE THE GOSUB STATEMENT 2 BYTES
 3274                          47500   ;       A POINTER INTO THE TEXT OF THE GOSUB 2 BYTES
 3275                          47520   ;
 3276                          47540   ; HIGH ADDRESS
 3277                          47560   ;
 3278                          47580   ; TOTAL 5 BYTES
 3279                          47600   ;
 3280  003770' 001000  000315  47620   GOSUB:  CALL    GETSTK          ;MAKE SURE THERE IS ROOM
 3281  003771' 000000  002624'
 3282  003772' 000000  003766'
 3283  003773' 000000  000003  47640           3
 3284  003774' 001000  000301  47660           POP     B               ;POP OFF RETURN ADDRESS OF "NEWSTT"
 3285  003775' 001000  000345  47680           PUSH    H               ;REALLY PUSH THE TEXT POINTER
 3286  003776' 001000  000345  47700           PUSH    H               ;SAVE TEXT POINTER
 3287  003777' 001000  000052  47720           LHLD    CURLIN          ;GET THE CURRENT LINE #
 3288  004000' 000000  001607'
 3289  004001' 000000  003771'
 3290  004002' 001000  000343  47740           XTHL                    ;PUT CURLIN ON THE STACK AND [H,L]=TEXT PTR
 3291  004003' 001000  000026  47760           MVI     D,GOSUTK        ;LEAVE A GOSUB TOKEN
 3292  004004' 000000  000214
 3293                          47780                                   ;ON THE STACK
 3294  004005' 001000  000325  47800           PUSH    D
 3295  004006' 001000  000063  47820           INX     SP              ;THE GOSUB TOKEN TAKES ONLY ONE BYTE
 3296  004007' 001000  000305  47840   RUNC2:  PUSH    B               ;RESTORE RETURN ADDRESS
 3297                          47860                                   ;OF "NEWSTT"
 3298                          47880   ;
 3299                          47900   ; IN THE 4K VERSION WE START AT THE BEGINNING
 3300                          47920   ; AND SEARCH, IN THE 8K WE START WHERE WE
 3301                          47940   ; ARE IF WE ARE  GOING TO A FORWARD LOCATION,
```

```
                                    47960  ;
 3303  004010' 001000  000315       47980  GOTO:   CALL    LINGET          ;PICK UP THE LINE #
 3304  004011' 000000  003642'
 3305  004012' 000000  004000'
 3306                               48000                                  ;AND PUT IT IN [D,E]
 3307                               48020  IFE      LENGTH,<RNZ             ;SHOULD END WITH A LINE
 3308                               48040                                  ;TERMINATOR -- BLOW HIM UP
 3309                               48060                                  ;IF IT DOESN'T
 3310                               48080                                  ;(ON GOTO MAKES THIS WRONG
 3311                               48100                                  ;IN OTHER VERSONS)
 3312                               48120          CALL    FNDLIN>
 3313                               48140  IFN      LENGTH,<
 3314  004013' 001000  000315       48160          CALL    REM             ;SKIP TO THE END OF THIS LINE
 3315  004014' 000000  004074'
 3316  004015' 000000  004011'
 3317  004016' 001000  000345       48180          PUSH    H               ;SAVE THE POINTER
 3318  004017' 001000  000052       48200          LHLD    CURLIN          ;GET THE CURRENT LINE #
 3319  004020' 000000  001607'
 3320  004021' 000000  004014'
 3321  004022' 001000  000347       48220          COMPAR                  ;[D,E] CONTAINS WHERE WE ARE GOING
 3322                               48240                                  ;[H,L] CONTAINS THE CURRENT LINE#
 3323                               48260                                  ;SO COMPARING THEM TELL US WHETHER TO
 3324                               48280                                  ;START SEARCHING FROM WHERE WE ARE OR
 3325                               48300                                  ;TO START SEARCHING FROM THE BEGINNING
 3326                               48320                                  ;OF TXTTAB
 3327  004023' 001000  000341       48340          POP     H               ;[H,L]=CURRENT POINTER
 3328  004024' 001000  000043       48360          INX     H               ;POINT AT THE LINK BEYOND IT
 3329  004025' 001000  000334       48380          CC      LOOP            ;SEARCH FROM THIS POINT
 3330  004026' 000000  002374'
 3331  004027' 000000  004020'
 3332  004030' 001000  000324       48400          CNC     FNDLIN>         ;SEARCH FROM THE BEGINNING -- ACTUALLY
 3333  004031' 000000  002371'
 3334  004032' 000000  004026'
 3335                               48420                                  ;SEARCH AGAIN IF ABOVE SEARCH FAILED
 3336  004033' 001000  000140       48440          MOV     H,B
 3337  004034' 001000  000151       48460          MOV     L,C
 3338  004035' 001000  000053       48480          DCX     H
 3339  004036' 001000  000330       48500          RC                      ;IF A MATCH WE ARE DONE
 3340  004037' 001000  000036       48520  USERR:  MVI     E,ERRUS
 3341  004040' 000000  000010
 3342  004041' 001000  000303       48540          JMP     ERROR           ;C=MATCH,SO IF NO MATCH WE
 3343  004042' 000000  002102'
 3344  004043' 000000  004031'
 3345                               48560                                  ;GIVE A "US" ERROR
 3346                               48580  ;
 3347                               48600  ; SEE "GOSUB" FOR THE FORMAT OF THE STACK ENTRY
 3348                               48620  ; "RETURN" RESTORES THE LINE NUMBER AND TEXT POINTER ON THE STACK
 3349                               48640  ; AFTER ELIMINATING ALL THE "FOR" ENTRIES IN FRONT OF THE "GOSUB"
 3350                               48660  ; ENTRY
 3351                               48680  ;
 3352  004044' 001000  000300       48700  RETURN: RNZ                     ;BLOW HIM UP IF THERE ISN'T A TERMINATOR
 3353  004045' 001000  000026       48720          MVI     D,255           ;MAKE SURE THIS VARIABLE POINTER
 3354  004046' 000000  000377
```



```
                                    48740                                  ;IN [D,E] NEVER GETS MATCHED
 3356  004047' 001000  000315       48760          CALL    FNDFOR          ;GO PAST ALL THE "FOR" ENTRIES
 3357  004050' 000000  001744'
 3358  004051' 000000  004042'
 3359  004052' 001000  000371       48780          SPHL                    ;UPDATE THE STACK
 3360  004053' 001000  000376       48800          CPI     GOSUTK
 3361  004054' 000000  000214
 3362  004055' 001000  000036       48820          MVI     E,ERRRG         ;ERROR ERRRG IS "RETURN WITHOUT GOSUB"
 3363  004056' 000000  000003
 3364  004057' 001000  000302       48840          JNZ     ERROR
 3365  004060' 000000  002102'
 3366  004061' 000000  004050'
 3367  004062' 001000  000341       48860          POP     H               ;GET LINE # "GOSUB" WAS FROM
 3368  004063' 001000  000042       48880          SHLD    CURLIN          ;PUT IT INTO CURLIN
 3369  004064' 000000  001607'
 3370  004065' 000000  004060'
 3371  004066' 001000  000041       48900          LXI     H,NEWSTT
 3372  004067' 000000  003302'
 3373  004070' 000000  004064'
 3374  004071' 001000  000343       48920          XTHL                    ;PUT RETURN ADDRESS OF "NEWSTT"
 3375                               48940                                  ;BACK ONTO THE STACK, GET TEXT POINTER
 3376                               48960                                  ;FROM "GOSUB"
 3377                               48980                                  ;SKIP OVER SOME CHARACTERS
 3378                               49000                                  ;SINCE WHEN "GOSUB" STUCK THE TEXT POINTER
 3379                               49020                                  ;ONTO THE STACK THE LINE # ARGUMENT HADN'T
 3380                               49040                                  ;BEEN READ IN YET,
 3381
 3382
 3383                               49100  IFN      STRING,<
 3384  004072' 001000  000001       49120  DATA:   XWD     "01000,"01      ;"LXI B," TO PICK UP ":" INTO C AND SKIP
 3385  004073' 000000  000072       49140          ":"                     ;"DATA" TERMINATES ON ":"
 3386                               49160                                  ;AND 0, ":" ONLY APPLIES IF
 3387                               49180                                  ;QUOTES HAVE MATCHED UP
 3388  004074'                      49200  IFE      LENGTH-2,<ELSE:>        ;EXECUTED "ELSE"S ARE SKIPPED
 3389                               49220  ;
 3390                               49240  ; NOTE: REM MUST PRESERVE [D,E] BECAUSE OF "GO TO" AND ERROR
 3391                               49260  ;
 3392  004074' 001000  000016       49280  REM:    XWD     "01000,"016     ;MVI C,   THE ONLY TERMINATOR IS 0
 3393  004075' 001000  000000       49300          XWD     "01000,0        ;NO-OPERATION
 3394                               49320                                  ;"DATA" ACTUALLY EXECUTES THIS 0
 3395  004076' 001000  000006       49340          MVI     B,0             ;INSIDE QUOTES THE ONLY TERMINATOR IS ZERO
 3396  004077' 000000  000000
 3397  004100' 001000  000171       49360  EXCHQT: MOV     A,C             ;WHEN A QUOTE IS SEEN THE SECOND
 3398  004101' 001000  000110       49380          MOV     C,B             ;TERMINATOR IS TRADED, SO IN "DATA"
 3399  004102' 001000  000107       49400          MOV     B,A             ;COLONS INSIDE QUOTATIONS WILL HAVE NO EFFECT
 3400  004103' 001000  000176       49420  REMER:  MOV     A,M             ;GET THE CHARACTER
 3401  004104' 001000  000267       49440          ORA     A               ;ZERO IS ALWAYS A TERMINATOR
 3402  004105' 001000  000310       49460          RZ
 3403  004106' 001000  000270       49480          CMP     B               ;TEST FOR THE OTHER TERMINATOR
 3404  004107' 001000  000310       49500          RZ
 3405  004110' 001000  000043       49520          INX     H
 3406  004111' 001000  000376       49540          CPI     34              ;IS IT A QUOTE?
 3407  004112' 000000  000042
```

```
3408  004113' 001000  000312  49560          JZ      EXCHQT          ;IF SO TIME TO TRADE
3409  004114' 000000  004100'
3410  004115' 000000  004067'
3411                           49580  IFE     LENGTH-2,<
3412                           49600  ;
3413                           49620  ; WHEN AN "IF" TAKES A FALSE BRANCH IT MUST FIND THE APPROPRIATE "ELSE"
3414                           49640  ; TO START EXECUTION AT, "DATA" COUNTS THE NUMBER OF "IF"S
3415                           49660  ; IT SEES SO THAT THE "ELSE" CODE CAN MATCH "ELSE"S WITH
3416                           49680  ; "IF"S, THE COUNT IS KEPT IN [D]
3417                           49700  ;
3418  004116' 001000  000326  49720          SUI     IFTK            ;IS IT AN "IF"
3419  004117' 000000  000212
3420  004120' 001000  000302  49740          JNZ     REMER           ;IF NOT, CONTINUE ON
3421  004121' 000000  004103'
3422  004122' 000000  004114'
3423  004123' 001000  000270  49760          CMP     B               ;SINCE "REM" CAN'T SMASH
3424                           49780                                  ;[D,E] WE HAVE TO BE CAREFUL
3425                           49800                                  ;SO ONLY IF B DOESN'T EQUAL
3426                           49820                                  ;ZERO WE INCREMENT D. (THE "IF" COUNT)
3427  004124' 001000  000212  49840          ADC     D               ;CARRY ON IF [B] NOT ZERO
3428  004125' 001000  000127  49860          MOV     D,A>            ;UPDATE [D]
3429  004126' 001000  000303  49880          JMP     REMER>
3430  004127' 000000  004103'
3431  004130' 000000  004121'
3432                           49900  ;
3433                           49902  ; WITHOUT STRINGS THERE IS NO NEED TO WATCH QUOTATIONS
3434                           49904  ; SO [B] IS SET UP AS THE SECONDARY TERMINATOR
3435                           49906  ; AND A SCAN IS MADE FOR [B] OR ZERO
3436                           49908  ;
3437                           49920  IFE     STRING,<
3438                           49940  DATA:   XWD     "01000,1        ;MAKE [C]=":" AND SKIP
3439                           49960          ":"
3440                           49980  REM:    XWD     "01000,16       ;MAKE [C]=0
3441                           50000          XWD     "01000,0        ;ZERO AND NO-OPERATION
3442                           50020  LOOPDR: MOV     A,M
3443                           50040          ORA     A               ;ALWAYS STOP ON ZERO
3444                           50060          RZ
3445                           50080          CMP     C               ;CHECK FOR ":" IN DATA
3446                           50100          RZ
3447                           50120          INX     H               ;LOOK AT NEXT CHARACTER
3448                           50140          JMP     LOOPDR>
3449                           50160  PAGE
```



```
3450                           50180  SUBTTL "LET"
3451  004131' 001000  000315  50200  LET:    CALL    PTRGET          ;GET THE POINTER TO THE VARIABLE
3452  004132' 000000  006505'
3453  004133' 000000  004127'
3454                           50220                                  ;NAMED IN TEXT AND PUT
3455                           50240                                  ;IT ON THE STACK, [H,L] REMAINS
3456                           50260                                  ;THE TEXT POINTER AND A,PSW ARE SETUP
3457                           50280                                  ;AS THE TERMINATING CHARACTER,
3458  004134' 001000  000317  50300          SYNCHK  EQULTK          ;CHECK FOR "="
3459  004135' 000000  000260
3460  004136'                  50320  REDINP:
3461                           50340  IFN     STRING,<
3462  004136' 001000  000072  50360          LDA     VALTYP
3463  004137' 000000  001543'
3464  004140' 000000  004132'
3465  004141' 001000  000365  50380          PUSH    PSW>
3466  004142' 001000  000325  50400          PUSH    D
3467  004143' 001000  000315  50420          CALL    FRMEVL          ;GET THE VALUE OF THE FORMULA
3468  004144' 000000  005336'
3469  004145' 000000  004137'
3470                           50440                                  ;INTO FAC
3471  004146' 001000  000343  50460          XTHL                    ;[H,L]=POINTER TO VARIABLE
3472                           50480                                  ;TEXT POINTER TO ON TOP OF STACK
3473  004147' 001000  000042  50500          SHLD    TEMP            ;SAVE VARIABLE POINTER FOR "FOR"
3474  004150' 000000  001603'
3475  004151' 000000  004144'
3476                           50520  IFN     STRING,<
3477  004152' 001000  000321  50540          POP     D               ;GET TEXT POINTER OFF
3478  004153' 001000  000361  50560          POP     PSW             ;GET THE VALTYP OF THE
3479                           50580                                  ;VARIABLE INTO [A]
3480  004154' 001000  000325  50581          PUSH    D               ;RESAVE THE TEXT-POINTER
3481                           50582  IFE     LENGTH-2,<
3482  004155' 001000  000376  50584          CPI     3               ;SEE IF ITS A STRING
3483  004156' 000000  000003
3484  004157' 001000  000345  50586  INPCOM: PUSH    H               ;SAVE THE POINTER AT THE VALUE POSITION
3485  004160' 001000  000302  50588          JNZ     COPNUM          ;NUMERIC, SO FORCE IT AND COPY
3486  004161' 000000  004237'
3487  004162' 000000  004150'
3488  004163' 001000  000315  50590          CALL    CHKSTR>         ;MAKE SURE THE FORMULA WAS A STRING
3489  004164' 000000  000000*
3490  004165' 000000  004161'
3491                           50600  IFN     LENGTH-2,<
3492                           50620          RAR                     ;CARRY SET FOR STRING OFF
3493                           50640                                  ;FOR NUMERIC
3494                           50660          CALL    CHKVAL          ;MAKE SURE VALTYP MATCHES CARRY
3495                           50680                                  ;AND SET THE ZERO FLAG
3496                           50700                                  ;ON A NUMERIC VALTYP
3497                           50860          JZ      COPNUM          ;IF A NUMBER  COPY
3498                           50880  INPCOM: PUSH    H>              ;SAVE POINTER AT VARIABLE
3499  004166' 001000  000052  50900          LHLD    FACLO           ;GET POINTER TO THE DESCRIPTOR OF THE RESULT
3500  004167' 000000  001637'
3501  004170' 000000  004164'
3502  004171' 001000  000345  50920          PUSH    H               ;SAVE THE POINTER AT THE DESCRIPTOR
```

```
3503  004172' 001000  000045          50940          INX     H
3504  004173' 001000  000043          50960          INX     H
3505  004174' 001000  000367          50980          PUSHM
3506  004175' 001000  000321          51000          POP     D
3507  004176' 001000  000652          51020          LHLD    STKTOP          ;SEE IF IT POINTS INTO STRING SPACE
3508  004177' 000000  001615'
3509  004200' 000000  004167'
3510  004201' 001000  000347          51040          COMPAR                  ;IF NOT DON'T COPY
3511  004202' 001000  000321          51060          POP     D               ;GET BACK THE POINTER AT THE DESCRIPTOR
3512  004203' 001000  000322          51080          JNC     DNTCPY          ;DON'T COPY LITERALS
3513  004204' 000000  004217'
3514  004205' 000000  004177'
3515  004206' 001000  000052          51100          LHLD    VARTAB          ;NOW, SEE IF ITS A VARIABLE
3516  004207' 000000  001621'
3517  004210' 000030  004204'
3518  004211' 001000  000347          51120          COMPAR                  ;BY SEEING IF THE  DESCRIPTOR
3519  004212' 001000  000153          51140          MOV     L,E
3520  004213' 001000  000142          51160          MOV     H,D
3521  004214' 001000  000334          51180          CC      STRCPY          ;IS BEYOND [VARTAB], IF SO COPY
3522  004215' 000000  007601'
3523  004216' 000000  004207'
3524  004217' 001000  000032          51200  DNTCPY: LDAX    D               ;GET THE LENGTH AND SAVE IT
3525  004220' 001000  000365          51220          PUSH    PSW             ;SINCE WE ARE GOING TO SET IT TO
3526                                   51240                                  ;Ø SO FRETMP DOESN'T UPDATE FRETOP
3527  004221' 001000  000257          51260          XRA     A               ;SET IT TO Ø -- ELIMINATING NULL
3528                                   51280                                  ;STRING IN FRETMP IS HARMLESS
3529  004222' 001000  000022          51300          STAX    D               ;PUT Ø IN THE LENGTH FIELD
3530  004223' 001000  000315          51320          CALL    FRETMP          ;FREE IT UP
3531  004224' 000000  0120460'
3532  004225' 000000  004215'
3533  004226' 001000  000361          51340          POP     PSW             ;GET LENGTH BACK
3534  004227' 001000  000167          51360          MOV     M,A             ;REPLACE IT ( FRETMP RETURNS [D,E] IN [H,L] )
3535  004230' 001000  000353          51380          XCHG                    ;PUT THE DESCRIPTOR POINTER BACK IN [D,E]
3536  004231' 001000  000341          51400          POP     H               ;GET THE PLACE OF THE NEW VARIABLE
3537                                   51420  IFN     LENGTH=2,<
3538                                   51440          CALL    MOVE>           ;COPY THE DESCRIPTOR
3539                                   51460  IFE     LENGTH=2,<
3540  004232' 001000  000315          51480          CALL    VMOVE>
3541  004233' 000000  000000*
3542  004234' 000000  004224'
3543  004235' 001000  000341          51500          POP     H               ;GET THE TEXT POINTER BACK
3544  004236' 001000  000311          51520          RET>
3545  004237'                         51540  COPNUM:
3546                                   51560  IFE     LENGTH=2,<
3547  004237' 001000  000346          51580          ANI     6               ;SETUP DISPATCH TO FORCE
3548  004240' 000000  000006
3549                                   51600                                  ;FORMULA TYPE TO CONFORM
3550                                   51620                                  ;TO THE VARIABLE ITS BEING ASSIGNED TO
3551  004241' 001000  000041          51640          LXI     H,FRCTBL        ;TABLE OF FORCE ROUTINES
3552  004242' 000000  000664'
3553  004243' 000000  004233'
3554  004244' 001000  000117          51660          MOV     C,A             ;[B,C]=TWO BYTE OFFSET
3555  004245' 001000  000006          51680          MVI     B,Ø
```



```
3556  004246' 000000  000000
3557  004247' 001000  000011          51700          DAD     B
3558  004250' 001000  000176          51720          MOV     A,M             ;[H,L]=ADDRESS TO GO TO
3559  004251' 001000  000043          51740          INX     H
3560  004252' 001000  000146          51760          MOV     H,M
3561  004253' 001000  000157          51780          MOV     L,A
3562  004254' 001000  000001          51800          LXI     B,PUTVAL        ;RETURN TO PUTVAL
3563  004255' 000000  004261'
3564  004256' 000000  004242'
3565  004257' 001000  000305          51820          PUSH    B
3566  004260' 001000  000351          51840          PCHL                    ;DISPATCH TO FORCE
3567  004261' 001000  000341          51860  PUTVAL: POP     H               ;GET THE POINTER OF WHERE TO STORE
3568                                   51880                                  ;THE VALUE
3569  004262' 001000  000345          51900          PUSH    H               ;SAVE IT BACK
3570  004263' 001000  000315          51920          CALL    VMOVMF>         ;MOVE THE VALUE IN
3571  004264' 000000  000000*
3572  004265' 000000  004255'
3573                                   51940  IFN     LENGTH=2,<
3574                                   51960          PUSH    H               ;SAVE THE VARIABLE POINTER FOR "FOR"
3575                                   51980          CALL    MOVMF>          ;TRANSFER THE VALUE
3576  004266' 001000  000321          52000          POP     D               ;"FOR" WANTS VARIABLE POINTER IN
3577                                   52020                                  ;[D,E] FOR FNDFOR
3578  004267' 001000  000341          52040          POP     H               ;GET THE TEXT POINTER
3579  004270' 001000  000311          52060          RET
3580                                   52080  PAGE
```

```
                                    52100   SUBTTL ON GOTO CODE
                                    52120   IFN     LENGTH,<
                                    52160   ONGOTO: CALL    GETBYT          ;GET VALUE INTO [E]
                                    52180           MOV     A,M             ;GET THE TERMINATOR BACK
                                    52200           MOV     B,A             ;SAVE THIS CHARACTER FOR LATER
                                    52220           CPI     GOSUTK          ;AN "ON ... GOSUB" PERHAPS?
                                    52240           JZ      ISGOSU          ;YES, SOME FEATURE USE
                                    52260           SYNCHK  GOTOTK          ;OTHERWISE MUST BE "GOTO"
                                    52280           DCX     H               ;BACK UP CHARACTER POINTER
                                    52300   ISGOSU: MOV     C,E             ;GET COUNT INTO  [C]
                                    52320   LOOPON: DCR     C               ;SEE IF ENOUGH SKIPS
                                    52340           MOV     A,B             ;PUT DISPATCH CHARACTER IN PLACE
                                    52360           JZ      GONE2           ;IF DONE, GO OFF
                                    52380           CALL    LINGT2          ;SKIP OVER A LINE #
                                    52400           CPI     44              ;A COMMA
                                    52420           RNZ                     ;IF A COMMA DOESN'T DELIMIT THE END OF
                                    52440                                   ;THE LAST LINE # MUST BE THE END OF THE LINE
                                    52460           JMP     LOOPON>         ;CONTINUE GOBBLING LINE #S
                                    52500   PAGE
```



```
                                    52520   SUBTTL IF ... THEN CODE
                                    52540   IF:     CALL    FRMEVL          ;EVALUATE A FORMULA
                                    52560   IFE     LENGTH,<
                                    52580   IFN     STRING,<
                                    52600           LDA     VALTYP          ;GET VALUE TYPE INTO [A]
                                    52620           PUSH    PSW>>           ;SAVE THE VALUE TYPE ON THE STACK
                                    52640           MOV     A,M             ;GET TERMINATING CHARACTER OF FORMULA
                                    52660   IFE     LENGTH,<
                                    52680           CALL    PUSHF           ;ONTO THE STACK
                                    52700           MVI     D,0             ;KEEPS RELATIONAL OPERATOR MEMORIES
                                    52720                                   ;LESS THAN =4
                                    52740                                   ;EQUAL     =2
                                    52760                                   ;GREATER THAN =1
                                    52780   LOOPIF: SUI     GREATK          ;CHECK FOR A RELATIONAL OPERATOR
                                    52800           JC      ENDREL          ;NOPE
                                    52820   NUMREL=LESSTK-GREATK+1          ;NUMBER OF RELATIONAL OPERATORS
                                    52840           CPI     NUMREL          ;IS THIS ONE OF THEM?
                                    52860           JNC     ENDREL          ;NO SEE WHAT WE HAVE
                                    52880           CPI     1               ;SETUP BITS BY MAPPING
                                    52900           RAL                     ;0 TO 1, 1 TO 2 AND 2 TO 4
                                    52920           ORA     D               ;OR WITH EARLIER BITS
                                    52940           MOV     D,A             ;STORE NEW BITS
                                    52960           CHRGET                  ;GET NEW CHARACTER
                                    52980           JMP     LOOPIF          ;SEE IF RELATIONAL
                                    53000   ENDREL: MOV     A,D             ;GET REALTIONAL MEMORIES
                                    53020           ORA     A               ;SEE IF THERE ARE ANY
                                    53040           JZ      SNERR           ;NO RELATIONAL OPERATORS!
                                    53060           PUSH    PSW             ;SAVE RELATIONAL MEMORIES
                                    53080           CALL    FRMEVL>         ;PICK UP FIRST NON-RELATIONAL
                                    53100                                   ;CHARACTER AGAIN AND INTERPRET FORMULA
                                    53120                                   ;ANSWER LEFT IN FAC
                                    53140   IFE     LENGTH-2,<CPI   44      ;A COMMA?
                                    53160           CZ      CHRGTR>         ;IF SO SKIP IT
                                    53180   IFN     LENGTH,<
                                    53200           CPI     GOTOTK          ;ALLOW "GOTO" AS WELL
                                    53220           JZ      OKGOTO>
                                    53240           SYNCHK  THENTK          ;MUST HAVE A THEN
                                    53260           DCX     H
                                    53280   OKGOTO:
                                    53300   IFE     LENGTH,<
                                    53320           POP     PSW
                                    53340           POPR                    ;POP OFF NUMBER
                                    53360   IFN     STRING,<
                                    53380           XTHL>                   ;COMPARE FORMULA TYPES
```

```
                                                53400   IFE     STRING,<PUSH    H>      ;SAVE THE TEXT POINTER
                                                53420           PUSH    PSW             ;RESAVE RELATIONAL MEMORIES
                                                53440   IFN     STRING,<
                                                53460           LDA     VALTYP          ;GET VALUE TYPE
                                                53480           CMP     M               ;(M) HAS OLD VALTYP ARE THEY =?
                                                53500           JNZ     TMERR           ;IF NOT ITS A TYPE ERROR
                                                53520           ORA     A               ;SEE WHAT TYPE IT WAS
                                                53540           JZ      NUMCMP          ;ZERO MEANS IT WAS NUMERIC
                                                53560           CALL    STRCMP          ;MUST BE STRING,SO STRING COMPARE
                                                53580           JMP     SKPNCM>         ;SKIP OVER NUMERIC COMPARE
                                                53600   NUMCMP: CALL    FCOMP           ;COMPARE THE 2 SIDE OF THE RELATION STATEMENT
                                                53620   SKPNCM: INR     A               ;BUILD RELATIONAL BITS
                                                53640           RAL                     ;LES$=4 EQUAL=2 GREATER=1
                                                53660                                   ;SINCE CARRY IS ON ONLY IN
                                                53680                                   ;377  CASE (FCOMP & STRCMP)
                                                53700           POP     B               ;POP OFF WHAT RELATIONAL OPERATOR WAS
                                                53720           ANA     B>              ;SEE IF WE MATCHED
                                                53740   IFN     LENGTH,<
                                                53742   IFE     LENGTH=2,<
  004346' 001000  000315                        53744           CALL    VSIGN##>
  004347' 000000  000000*
  004350' 000000  004341'
                                                53746   IFN     LENGTH=2,<
                                                53760           FSIGN>>                 ;0=FALSE ALL OTHERS=TRUE
                                                53780   IFE     LENGTH,<
                                                53800           POP     H>              ;POP OFF TEXT POINTER
                                                53820   IFE     LENGTH=2,<
  004351' 001000  000312                        53840           JZ      FALSIF>         ;HANDLE POSSIBLE "ELSE"
  004352' 000000  004363'
  004353' 000000  004347'
                                                53860   IFN     LENGTH=2,<
                                                53880           JZ      REM>            ;IF TEST FAILED -- JUST SKIP REST OF THE LINE
  004354' 001000  000327                        53900   DOCOND: CHRGET                  ;PICK UP THE FIRST LINE # CHARACTER
  004355' 001000  000332                        53920           JC      GOTO            ;LINE NUMBER MEANS "GOTO"
  004356' 000000  004010'
  004357' 000000  004352'
  004360' 001000  000303                        53940           JMP     GONE3           ;INTERPRET NEW STATEMENT
  004361' 000000  003375'
  004362' 000000  004356'
                                                53960   IFE     LENGTH=2,<
                                                53980   ;
                                                54000   ; "ELSE" HANDLER, HERE ON FALSE "IF" CONDITION
                                                54020   ;
  004363' 001000  000026                        54040   FALSIF: MVI     D,1             ;NUMBER OF "ELSE"S THAT MUST
  004364' 000000  000001                                                                ;BE SEEN, "DATA" INCREMENTS THIS
                                                54060                                   ;COUNT EVERY TIME AN "IF" IS SEEN
                                                54080                                   ;BE SEEN, "DATA" INCREMENTS THIS
  004365' 001000  000315                        54100   SKPMRF: CALL    DATA            ;SKIP A STATEMENT
  004366' 000000  004072'
  004367' 000000  004361'
                                                54120                                   ;":" IS STUCK IN FRONT OF "ELSE"S
                                                54140                                   ;SO THAT "DATA" WILL STOP BEFORE "ELSE" CLAUSES
```



```
  004370' 001000  000267                        54160           ORA     A               ;END OF LINE?
  004371' 001000  000310                        54180           RZ                      ;IF SO, NO "ELSE" CLAUSE
  004372' 001000  000327                        54200           CHRGET                  ;SEE IF WE HIT AN "ELSE"
  004373' 001000  000376                        54220           CPI     ELSETK
  004374' 000000  000220
  004375' 001000  000302                        54240           JNZ     SKPMRF          ;NO, STILL IN THE "THEN" CLAUSE
  004376' 000000  004365'
  004377' 000000  004366'
  004400' 001000  000025                        54260           DCR     D               ;DECREMENT THE NUMBER OF "ELSE"S THAT
                                                54280                                   ;MUST BE SEEN
  004401' 001000  000302                        54300           JNZ     SKPMRF          ;SKIP MORE IF HAVEN'T SEEN
  004402' 000000  004365'
  004403' 000000  004376'
                                                54320                                   ;ENOUGH
  004404' 001000  000303                        54340           JMP     DOCOND>         ;FOUND THE RIGHT "ELSE" -- GO EXECUTE
  004405' 000000  004354'
  004406' 000000  004402'

                                                54380   PAGE
```

```
                                        54400   SUBTTL PRINT CODE
                                        54420
                                        54440   IFN     LPTSW,<
                                        54460   LPRINT: MVI     A,1             ;SAY NON ZERO
                                        54480           STA     PRTFLG>         ;SAVE AWAY
  3745  004407' 001000  000053          54500   NEWCHR: DCX     H
  3746  004410' 001000  000327          54520   MORPR:  CHRGET                  ;GET ANOTHER CHARACTER
  3747  004411' 001000  000312          54540   PRINT:  JZ      CRDO            ;IF WE SEE A TERMINATOR
  3748  004412' 000000  004523'
  3749  004413' 000000  004485'
                                        54560                                   ;GO TYPE A CRLF
  3751  004414' 001000  000310          54580   PRINTC: RZ                      ;HERE AFTER SEEING TAB(X) OR , OR ;
                                        54600                                   ;IN WHICH CASE A TERMINATOR DOES NOT
                                        54620                                   ;MEAN WE SHOULD TYPE A CRLF
                                        54640                                   ;BUT JUST RETURN
                                        54660   IFE     STRING,<
                                        54680           CPI     34              ;A TERMINATING QUOTE?
                                        54700           CZ      STROU1
                                        54720           JZ      NEWCHR>         ;A QUOTATION -- JUST PRINT IT
  3759  004415' 001000  000376          54740           CPI     TABTK
  3760  004416' 000000  000240
  3761  004417' 001000  000312          54760           JZ      TABER           ;THE TAB FUNCTION?
  3762  004420' 000000  004577'
  3763  004421' 000000  004412'
                                        54780   IFN     LENGTH,<
  3765  004422' 001000  000376          54800           CPI     SPCTK
  3766  004423' 000000  000242
  3767  004424' 001000  000312          54820           JZ      TABER>          ;THE SPC FUNCTION?
  3768  004425' 000000  004577'
  3769  004426' 000000  004420'
  3770  004427' 001000  000345          54840           PUSH    H               ;SAVE THE TEXT POINTER
  3771  004430' 001000  000376          54860           CPI     44
  3772  004431' 000000  000054
  3773  004432' 001000  000312          54880           JZ      COMPRT          ;IS IT A COMMA?
  3774  004433' 000000  004553'
  3775  004434' 000000  004425'
  3776  004435' 001000  000376          54900           CPI     59              ;IS IT A ";"
  3777  004436' 000000  000073
  3778  004437' 001000  000312          54920           JZ      NOTABR
  3779  004440' 000000  004637'
  3780  004441' 000000  004433'
  3781  004442' 001000  000301          54940           POP     B               ;GET RID OF OLD TEXT POINTER
  3782  004443' 001000  000315          54960           CALL    FRMEVL          ;BACK UP ONE CHARACTER AND READ THAT ONE
  3783  004444' 000000  005336'
  3784  004445' 000000  004440'
                                        54980                                   ;AGAIN SO THE CONDITION CODES ARE RIGHT
                                        55000                                   ;EVALUATE THE FORMULA
  3787  004446' 001000  000053          55020           DCX     H               ;BACKUP FROM TERMINATOR
  3788  004447' 001000  000345          55040           PUSH    H               ;SAVE TEXT POINTER
                                        55060   IFN     STRING,<
                                        55062   IFE     LENGTH=2,<
  3791  004450' 001000  000315          55064           CALL    GETYPE          ;SEE IF WE HAVE A STRING
  3792  004451' 000000  006307'
```



```
  3793  004452' 000000  004444'
  3794  004453' 001000  000312          55066           JZ      STRDON>         ;IF SO, PRINT SPECIALY
  3795  004454' 000000  004507'
  3796  004455' 000000  004451'
                                        55068   IFN     LENGTH=2,<
                                        55080           LDA     VALTYP
                                        55100           ORA     A
                                        55120           JNZ     STRDON>>        ;WAS IT A STRING FORMULA?
  3801  004456' 001000  000315          55140           CALL    FOUT            ;MAKE A NUMBER INTO A STRING
  3802  004457' 000000  000000*
  3803  004460' 000000  004454'
                                        55160   IFE     STRING,<CALL    STROUT> ;IF STRINGS OFF JUST OUTPUT IT
                                        55180   IFN     STRING,<
  3806  004461' 001000  000315          55200           CALL    STRLIT          ;MAKE IT  A STRING
  3807  004462' 000000  007637'
  3808  004463' 000000  004457'
  3809  004464' 001000  000052          55220           LHLD    FACLO           ;GET THE POINTER
  3810  004465' 000000  001637'
  3811  004466' 000000  004462'
                                        55240   IFN     LPTSW,<
                                        55260           LDA     PRTFLG
                                        55280           ORA     A
                                        55300           JZ      ISTTY           ;LPT OR TTY?
                                        55320           LDA     LPTPOS
                                        55340           ADD     M
                                        55360           CPI     LPTLEN          ;WILL THIS NUMBER OVERLAP?
                                        55380           JMP     LINCHK
                                        55400   ISTTY:>
  3821  004467' 001000  000072          55420           LDA     TTYPOS          ;SEE WHERE WE ARE
  3822  004470' 000000  000047'
  3823  004471' 000000  004465'
  3824  004472' 001000  000206          55440           ADD     M               ;ADD THIS LENGTH
  3825  004473' 001000  000376          55460           CPI     LINLEN          ;SEE IF GREATER THAN THE LINE LENGTH
  3826  004474' 000000  001110
  3827                  004474'         55480   LINPT3==:1,=1
  3828  004475' 001000  000324          55500   LINCHK: CNC     CRDO            ;IF SO CRLF
  3829  004476' 000000  004523'
  3830  004477' 000000  004470'
  3831  004500' 001000  000315          55520           CALL    STRPRT>         ;PRINT THE NUMBER
  3832  004501' 000000  007746'
  3833  004502' 000000  004476'
  3834  004503' 001000  000076          55540           MVI     A," "           ;ALWAYS END WITH A SPACE
  3835  004504' 000000  000040
  3836  004505' 001000  000337          55560           OUTCHR
                                        55580   IFN     STRING,<
                                        55590   IFE     LENGTH=2,<
  3839  004506' 001000  000267          55595           ORA     A               ;TURN OFF THE ZERO FLAG
  3840  004507' 001000  000314          55597   STRDON: CZ      STRPRT>
  3841  004510' 000000  007746'
  3842  004511' 000000  004501'
                                        55598   IFN     LENGTH=2,<
                                        55599           XRA     A
                                        55600   STRDON: CNZ     STRPRT>>        ;JUST PRINT STRINGS
```

```
3846  004512' 001000  000341          55620            POP     H
3847  004513' 001000  000303          55640            JMP     MORPR           ;PRINT SOME MORE
3848  004514' 000000  004410'
3849  004515' 000000  004510'
3850  004516' 001000  000066          55660    FININL: MVI     M,0             ;PUT A ZERO AT THE END OF BUF
3851  004517' 000000  000000
3852  004520' 001000  000041          55680            LXI     H,BUFMIN        ;SETUP POINTER
3853  004521' 000000  001430'
3854  004522' 000000  004514'
3855                                   55700    CRDO:   IFN     LPTSW,<
3856                                   55720            LDA     PRTFLG
3857                                   55740            ORA     A
3858                                   55760            JNZ     PRINTW>
3859  004523' 001000  000076          55780            MVI     A,13
3860  004524' 000000  000015
3861  004525' 001000  000062          55800            STA     TTYPOS          ;MAKE TTYPOS LESS THAN LINE LENGTH
3862  004526' 000000  000047'
3863  004527' 000000  004521'
3864                                   55820                                    ;SO CRDO DOESN'T GET CALLED AGAIN
3865  004530' 001000  000337          55840            OUTCHR
3866  004531' 001000  000076          55860            MVI     A,10
3867  004532' 000000  000012
3868  004533' 001000  000337          55880            OUTCHR
3869  004534' 001000  000072          55900    CRFIN:  LDA     NULCNT          ;GET NUMBER OF NULLS
3870  004535' 000000  000046'
3871  004536' 000000  004526'
3872  004537' 001000  000075          55920    PRTNUL: DCR     A
3873  004540' 001000  000062          55940            STA     TTYPOS          ;EVENTUALLY SETUP TTYPOS=0
3874  004541' 000000  000047'
3875  004542' 000000  004535'
3876  004543' 001000  000310          55960            RZ                      ;ALL NULLS DONE (A)=0
3877                                   55980                                    ;SOME ROUTINES DEPEND ON CRDO
3878                                   56000                                    ;AND CRFINS RETURN (A)=0 AND Z  TRUE
3879  004544' 001000  000365          56020            PUSH    PSW             ;SAVE THE COUNT
3880  004545' 001000  000257          56040            XRA     A               ;(A)= A NULL
3881  004546' 001000  000337          56060            OUTCHR                  ;SEND IT OUT
3882  004547' 001000  000361          56080            POP     PSW             ;RESTORE THE COUNT
3883  004550' 001000  000303          56100            JMP     PRTNUL          ;LOOP PRINTING NULLS
3884  004551' 000000  004537'
3885  004552' 000000  004541'
3886                                   56120    IFE     STRING,<
3887                                   56140    STROUI: INX     H
3888                                   56160    STROUT: MOV     A,M
3889                                   56180            ORA     A               ;CHECK FOR END OF LINE
3890                                   56200            RZ                      ;LET IT END THAT WAY
3891                                   56220            INX     H
3892                                   56240            CPI     34              ;A TERMINATING QUOTE?
3893                                   56260            RZ                      ;DONE IF SO
3894                                   56280            OUTCHR                  ;PRINT THE CHARACTER IN (A)
3895                                   56300            CPI     CR              ;IF IT'S A CARRIAGE RETURN
3896                                   56320            CZ      CRDO            ;TYPE LINE-FEED AND SET (TTYPOS)=0
3897                                   56340            JMP     STROUT>         ;PRINT MORE CHARACTERS
3898
```



```
3899                                   56380    COMPRT: IFN     LPTSW,<
3900                                   56400            LDA     PRTFLG          ;OUTPUT TO THE LINE PRINTER?
3901                                   56420            ORA     A               ;NON-ZERO MEANS YES
3902                                   56440            JZ      ISCTTY          ;NO, DO TELETYPE COMMA
3903                                   56460            LDA     LPTPOS          ;GET LINE PRINTER POSITION
3904                                   56480    NLPPOS==<<<LPTLEN/CLMWID>-1>*CLMWID>;POSITION BEYOND WHICH THERE ARE
3905                                   56500                                    ;NO MORE COMMA FIELDS, SO
3906                                   56520            CPI     NLPPOS          ;COMMA JUST DOES A "CRDO"
3907                                   56540            JMP     CHKCOM          ;USE TELETYPE CHECK
3908                                   56560    ISCTTY:>
3909  004553' 001000  000072          56580            LDA     TTYPOS          ;GET TELETYPE POSITION
3910  004554' 000000  000047'
3911  004555' 000000  004551'
3912                   000070          56600    NUTPOS==<<<LINLEN/CLMWID>-1>*CLMWID>;POSITION BEYOND WHICH THERE ARE
3913                                   56620                                    ;NO MORE COMMA FIELDS
3914  004556' 001000  000376          56640            CPI     NUTPOS          ;SO ALL COMMA DOES IS A "CRDO"
3915  004557' 000000  000070
3916                   004557'         56660    LINPT4==:,-1                    ;FIXED UP BY "TERMINAL WIDTH" QUESTION
3917  004560' 001000  000324          56680    CHKCOM: CNC     CRDO            ;TYPE CRLF
3918  004561' 000000  004523'
3919  004562' 000000  004554'
3920  004563' 001000  000322          56700            JNC     NOTABR          ;AND QUIT IF BEYOND THE LAST COMMA FIELD
3921  004564' 000000  004637'
3922  004565' 000000  004561'
3923  004566' 001000  000326          56720    MORCOM: SUI     CLMWID          ;GET (A) MODULUS CLMWID
3924  004567' 000000  000016
3925  004570' 001000  000322          56740            JNC     MORCOM
3926  004571' 000000  004566'
3927  004572' 000000  004564'
3928  004573' 001000  000057          56760            CMA                     ;WE WANT TO  FILL
3929                                   56780                                    ;THE PRINT POSITION OUT
3930                                   56800                                    ;TO AN EVEN CLMWID, SO
3931                                   56820                                    ;WE PRINT CLMWID-(A) MOD CLMWID SPACES
3932  004574' 001000  000303          56840            JMP     ASPA2           ;GO PRINT (A)+1 SPACES
3933  004575' 000000  004626'
3934  004576' 000000  004571'
3935
3936  004577'                          56880    TABER:
3937  004577' 001000  000365          56900    IFN     LENGTH,<PUSH    PSW>    ;REMEMBER IF (A)=SPCTK OR TABTK
3938                                   56920    IFE     LENGTH,<
3939                                   56940            CALL    INTIDX>         ;INTEGERIZE A FORMULA INTO (D,E)
3940                                   56960    IFN     LENGTH,<
3941  004600' 001000  000315          56980            CALL    GTBYTC>         ;GET VALUE INTO (E)
3942  004601' 000000  011017'
3943  004602' 000000  004575'
3944  004603' 001000  000317          57000            SYNCHK  ")"
3945  004604' 000000  000051
3946  004605' 001000  000053          57020            DCX     H
3947                                   57040    IFN     LENGTH,<
3948  004606' 001000  000361          57060            POP     PSW             ;GET BACK SPCTK OR TABTK
3949  004607' 001000  000376          57080            CPI     SPCTK>          ;WAS IT SPCTK?
3950  004610' 000000  000242
3951  004611' 001000  000345          57100            PUSH    H               ;SAVE THE TEXT POINTER
```

```
                                        57120   IFN     LENGTH,<
 3953  004612' 001000  000173          57140           MOV     A,E             ;FOR "SPC" PUT THE FORMULA
 3954  004613' 001000  000512          57160           JZ      ASPAC>          ;VALUE IN [A]
 3955  004614' 000000  004627'
 3956  004615' 000000  004601'
 3957                                   57180   IFN     LPTSW,<
 3958                                   57200           LDA     PRTFLG          ;LINE PRINTER OR TTY?
 3959                                   57220           ORA     A               ;NON-ZERO MEANS LPT
 3960                                   57240           JZ      TTYIST
 3961                                   57260           LDA     LPTPOS          ;GET LINE PRINTER POSITION
 3962                                   57280           JMP     DOSIZT>
 3963  004616' 001000  000072          57300   TTYIST: LDA     TTYPOS          ;GET TELETYPE PRINT POSITION
 3964  004617' 000000  000047'
 3965  004620' 000000  004614'
 3966  004621' 001000  000057          57320   DOSIZT: CMA                     ;PRINT [E]-[A] SPACES
 3967  004622' 001000  000203          57340           ADD     E
 3968  004623' 001000  000322          57360           JNC     NOTABK          ;IF NEGATIVE, DON'T PRINT ANY
 3969  004624' 000000  004637'
 3970  004625' 000000  004617'
 3971                                   57380                                   ;SPACES
 3972  004626' 001000  000074          57400   ASPA2:  INR     A
 3973  004627' 001000  000107          57420   ASPAC:  MOV     B,A             ;[B]=NUMBER OF SPACES TO PRINT
 3974  004630' 001000  000076          57440           MVI     A," "           ;[A]=SPACE
 3975  004631' 000000  000040
 3976  004632' 001000  000337          57460   REPOUT: OUTCHR                  ;PRINT [A]
 3977  004633' 001000  000005          57480           DCR     B               ;DECREMENT THE COUNT
 3978  004634' 001000  000302          57500           JNZ     REPOUT
 3979  004635' 000000  00463E'
 3980  004636' 000000  004624'
 3981  004637' 001000  000341          57520   NOTABK: POP     H               ;PICK UP TEXT POINTER
 3982  004640' 001000  000327          57540           CHRGET                  ;AND THE NEXT CHARACTER
 3983  004641' 001000  000303          57560           JMP     PRINTC          ;AND SINCE WE JUST PRINTED
 3984  004642' 000000  004414'
 3985  004643' 000000  004635'
 3986                                   57580                                   ;SPACES, DON'T CALL CRDO
 3987                                   57600                                   ;IF IT'S THE END OF THE LINE
 3988                                   57620   PAGE
```



```
                                        57640   SUBTTL INPUT AND READ CODE
                                        57660   IFN     LENGTH,<
 3991  004644' 000000  000077          57680   TRYAGN: DC"?REDO FROM START"
 3992  004645' 000000  000122
 3993  004646' 000000  000105
 3994  004647' 000000  000104
 3995  004650' 000000  000117
 3996  004651' 000000  000040
 3997  004652' 000000  000106
 3998  004653' 000000  000122
 3999  004654' 000000  000117
 4000  004655' 000000  000115
 4001  004656' 000000  000040
 4002  004657' 000000  000123
 4003  004660' 000000  000124
 4004  004661' 000000  000101
 4005  004662' 000000  000122
 4006  004663' 000000  000124
 4007  004663' 000000  000324
 4008  004664' 000000  000015          57700           ACRLF
 4009  004665' 000000  000012
 4010  004666' 000000  000000          57720           0
 4011                                   57740   ;
 4012                                   57760   ; HERE WHEN THE DATA THAT WAS TYPED IN OR IN "DATA" STATEMENTS
 4013                                   57780   ; IS IMPROPERLY FORMATTED, FOR "INPUT" WE START AGAIN,
 4014                                   57800   ; FOR "DATA" WE GIVE A SYNTAX ERROR AT THE DATA LINE
 4015                                   57820   ;
 4016  004667' 001000  000072          57840   TRMNOK: LDA     FLGINP          ;WAS IT READ OR INPUT?
 4017  004670' 000000  001602'
 4018  004671' 000000  004642'
 4019  004672' 001000  000267          57860           ORA     A               ;ZERO=INPUT
 4020  004673' 001000  000302          57880           JNZ     DATSNE          ;GIVE ERROR AT DATA LINE
 4021  004674' 000000  002664'
 4022  004675' 000000  004670'
 4023                                   57900           POP     B               ;GET RID OF THE POINTER INTO THE VARIABLE LIST
 4024  004676' 001000  000301
 4025  004677' 001000  000041          57920           LXI     H,TRYAGN
 4026  004700' 000000  004644'
 4027  004701' 000000  004674'
 4028  004702' 001000  000315          57940           CALL    STROUT          ;PRINT "?REDO FROM START"
 4029  004703' 000000  007743'
 4030  004704' 000000  004700'
 4031  004705' 001000  000052          57960           LHLD    TEMP            ;START ALL OVER BY GOING BACK
 4032  004706' 000000  001603'
 4033  004707' 000000  004703'
 4034  004710' 001000  000311          57980           RET>                    ;TO NEWSTT POINTING AT THE START OF
 4035                                   57982                                   ;OF THE "INPUT" STATEMENT
 4036  004711'                         58000   INPUT:
 4037                                   58020   IFN     LENGTH,<
 4038  004711' 001000  000376          58040           CPI     34              ;IS IT A QUOTE?
 4039  004712' 000000  000042
 4040                                   58060   IFN     CONTRW,<
 4041  004713' 001000  000076          58080           MVI     A,0             ;BE TALKATIVE
```

```
004714' 000000  000000
004715' 001000  000062      58100           STA     CNTWFL>         ;FORCE OUTPUT
004716' 000000  001541'
004717' 000000  004706'
004720' 001000  000302      58120           JNZ     NOTQTI          ;IF NOT NO MESSAGE
004721' 000000  004735'
004722' 000000  004716'
                            58140   IFN     STRING,<
004723' 001000  000315      58160           CALL    STRLTI          ;MAKE THE MESSAGE A STRING
004724' 000000  007640'
004725' 000000  004721'
004726' 001000  000317      58180           SYNCHK  59              ;MUST END WITH SEMI-COLON
004727' 000000  000073
004730' 001000  000345      58200           PUSH    H               ;REMEMBER WHERE IT ENDED
004731' 001000  000315      58220           CALL    STRPRT          ;PRINT IT OUT
004732' 000000  007746'
004733' 000000  004724'
004734' 001000  000341      58240           POP     H>
                            58260   IFE     STRING,<
                            58280           CALL    STROUI
                            58300           SYNCHK  59>>            ;ENDS WITH SEMI-COLON
004735' 001000  000345      58320   NOTQTI: PUSH    H
                            58340   IFE     FUNCTS,<
                            58360           LHLD    CURLIN          ;IS IT DIRECT?
                            58380           MVI     E,ERRID         ;IF SO "ILLEGAL DIRECT" ERROR
                            58400           INX     H               ;DIRECT MEANS THAT
                            58420           MOV     A,L             ;CURLIN=65535
                            58440           ORA     H               ;ADDED ONE AND GOT 0?
                            58460           JZ      ERROR>          ;IF SO,GO COMPLAIN
                            58480   IFN     FUNCTS,<
004736' 001000  000315      58500           CALL    ERRDIR>         ;USE COMMON ROUTINE SINCE DEF
004737' 000000  007532'
004740' 000000  004732'
                            58520                                   ;DIRECT IS ALSO ILLEGAL
004741' 001000  000315      58540   GETAGN: CALL    QINLIN          ;TYPE '?' AND INPUT A LINE OF TEXT
004742' 000000  002522'
004743' 000000  004737'
004744' 001000  000043      58560   IFN     LENGTH,<INX     H       ;IF NO INPUT WE QUIT
004745' 001000  000176      58580           MOV     A,M
004746' 001000  000267      58600           ORA     A
004747' 001000  000053      58620           DCX     H
004750' 001000  000301      58640           POP     B               ;TAKE OFF SINCE MAYBE LEAVING
004751' 001000  000312      58660           JZ      STPEND          ;IF EMPTY LEAVE
004752' 000000  003500'
004753' 000000  004742'
004754' 001000  000305      58680           PUSH    B>              ;PUT BACK  SINCE DIDN'T LEAVE
004755' 001000  000303      58700           JMP     INPCON
004756' 000000  004765'
004757' 000000  004752'
004760' 001000  000345      58720   READ:   PUSH    H               ;SAVE THE TEXT POINTER
004761' 001000  000052      58740           LHLD    DATPTR          ;GET LAST DATA LOCATION
004762' 000000  001627'
004763' 000000  004756'
```



```
004764' 001000  000366      58760           XWD     ^O1000,^O366    ;"ORI" TO SET [A] NON-ZERO
004765' 001000  000257      58780   INPCON: XRA     A               ;SET FLAG THAT THIS IS AN INPUT
004766' 001000  000062      58800           STA     FLGINP          ;STORE THE FLAG
004767' 000000  001602'
004770' 000000  004762'
                            58820   ;
                            58840   ; IN THE PROCESSING OF DATA AND READ STATEMENTS:
                            58860   ; ONE POINTER POINTS TO THE DATA (IE THE NUMBERS BEING FETCHED)
                            58880   ; AND ANOTHER POINTS TO THE LIST OF VARIABLES
                            58900   ;
                            58920   ; THE POINTER INTO THE DATA ALWAYS STARTS POINTING TO A
                            58940   ; TERMINATOR -- A , : OR END-OF-LINE
                            58960   ;
004771' 001000  000343      58980           XTHL                    ;[H,L]=VARIABLE LIST POINTER
                            58982                                   ;DATA POINTER GOES ON THE STACK
004772' 001000  000001      59000           XWD     ^O1000,1        ;"LXI B," OVER THIS CHECK
004773' 001000  000317      59020   LOPDT2: SYNCHK  44              ;MAKE SURE THERE IS A ","
004774' 000000  000054
004775' 001000  000315      59040           CALL    PTRGET          ;READ THE VARIABLE LIST
004776' 000000  006505'
004777' 000000  004767'
                            59060                                   ;AND GET THE POINTER TO A VARIABLE INTO [D,E]
005000' 001000  000343      59080           XTHL                    ;PUT THE VARIABLE LIST POINTER ONTO THE
                            59100                                   ;STACK AND TAKE THE
                            59120                                   ;DATA LIST POINTER OFF
                            59140   ;
                            59160   ; NOTE AT THIS POINT WE HAVE A VARIABLE WHICH WANTS DATA
                            59180   ; AND SO WE MUST GET DATA OR COMPLAIN
                            59200   ;
005001' 001000  000325      59220           PUSH    D               ;SAVE THE POINTER TO THE VARIABLE WE
                            59240                                   ;ARE ABOUT TO SET UP WITH A VALUE
005002' 001000  000176      59260           MOV     A,M             ;SINCE THE DATA LIST POINTER ALWAYS POINTS
                            59280                                   ;AT A TERMINATOR LETS READ THE
                            59300                                   ;TERMINATOR INTO [A] AND SEE WHAT
                            59320                                   ;IT IS
005003' 001000  000376      59340           CPI     44
005004' 000000  000054
005005' 001000  000312      59360           JZ      DATBK           ;A COMMA SO A VALUE MUST FOLLOW
005006' 000000  005025'
005007' 000000  004776'
                            59380   IFE     LENGTH,<
                            59400           ORA     A               ;IN THE 4K VERSION
                            59420           JNZ     SNERR>          ;DATA MUST BE ALONE ON A LINE
005010' 001000  000072      59440           LDA     FLGINP          ;SEE WHAT TYPE OF STATEMENT THIS WAS
005011' 000000  001602'
005012' 000000  005006'
005013' 001000  000267      59460           ORA     A
                            59480   IFE     LENGTH,<
                            59500           INX     H               ;POINT AT POINTER TO NEXT LINE
                            59520           JNZ     DATFND>         ;IF IT IS A READ GO
                            59540                                   ;SEARCH FOR ANOTHER DATA STATEMENT
005014' 001000  000302      59560   IFN     LENGTH,<JNZ     DATLOP>
005015' 000000  005163'
```

```
4148  005016' 000000  005011'
4149  005017' 001000  000076          59580          MVI     A,"?"
4150  005020' 000000  000077
4151  005021' 001000  000337          59600          OUTCHR                  ;DOUBLE PROMPT WHEN WE NEED MORE INPUT
4152  005022' 001000  000315          59620          CALL    QINLIN          ;GET A WHOLE LINE AFTER TYPING "?"
4153  005023' 000000  002522'
4154  005024' 000000  005015'
4155                                   59640                                  ;THE DATA NOW STARTS AT THE BEGINNING
4156                                   59660                                  ;OF THE BUFFER
4157                                   59680                                  ;AND QINLIN LEAVES [H,L]=BUF
4158  005025'                          59700  DATBK:
4159                                   59720  IFE     LENGTH,<POP     D       ;POP OFF POINTER TO THE VARIABLE
4160                                   59740          INX     H               ;FUDGE CHARACTER POINTER
4161                                   59760          CALL    REDINP>         ;READ A VALUE USING "LET" CODE
4162                                   59780  IFN     LENGTH,<
4163                                   59800  IFN     STRING,<
4164  005025' 001000  000072          59820          LDA     VALTYP          ;SEE IF ITS NUMERIC OR STRING
4165  005026' 000000  001543'
4166  005027' 000000  005023'
4167                                   59840  IFE     LENGTH=2,<
4168  005030' 001000  000376          59860          CPI     3               ;IS IT A STRING ?
4169  005031' 000000  000003
4170  005032' 001000  000365          59880          PUSH    PSW             ;SAVE THE TYPE INFORMATION
4171  005033' 001000  000302          59900          JNZ     NUMINS>         ;IF NUMERIC,USE FIN TO GET IT
4172  005034' 000000  005070'
4173  005035' 000000  005026'
4174                                   59920  IFN     LENGTH=2,<
4175                                   59940          ORA     A
4176                                   59960          JZ      NUMINS>         ;INPUT A NUMBER IF NUMERIC
4177                                   59980                                  ;ONLY THE VARAIBLE TYPE IS
4178                                   60000                                  ;CHECKED SO AN UNQUOTED STRING
4179                                   60020                                  ;CAN BE ALL DIGITS
4180  005036' 001000  000327          60040          CHRGET
4181  005037' 001000  000127          60060          MOV     D,A             ;ASSUME QUOTED STRING
4182  005040' 001000  000107          60080          MOV     B,A             ;SETUP TERMINATORS
4183  005041' 001000  000376          60100          CPI     34              ;QUOTE ?
4184  005042' 000000  000042
4185  005043' 001000  000312          60120          JZ      NOWGET          ;TERMINATORS OK
4186  005044' 000000  005053'
4187  005045' 000000  005034'
4188  005046' 001000  000026          60140          MVI     D,":"           ;UNQUOTED STRING TERMINATORS
4189  005047' 000000  000072
4190  005050' 001000  000006          60160          MVI     B,44            ;ARE COLON AND COMMA
4191  005051' 000000  000054
4192                                   60180          DCX     H               ;BACKUP SINCE START CHARACTER MUST BE INCLUDED
4193  005052' 001000  000053
4194                                   60200                                  ;IN THE QUOTED STRING CASE WE DON'T WANT TO
4195                                   60220                                  ;INCLUDE THE STARTING OR ENDING QUOTE
4196  005053' 001000  000315          60240  NOWGET: CALL    STRLT2          ;MAKE A STRING DESCRIPTOR FOR THE VALUE
4197  005054' 000000  007643'
4198  005055' 000000  005044'
4199                                   60242                                  ;AND COPY IF NECESSARY
4200                                   60260  IFE     LENGTH=2,<
```



```
4201  005056' 001000  000361          60280  DOASIG: POP     PSW>            ;POP OFF THE TYPE INFORMATION
4202  005057' 001000  000353          60320          XCHG                    ;[D,E]=TEXT POINTER
4203  005060' 001000  000041          60340          LXI     H,STRDN2        ;RETURN LOC
4204  005061' 000000  005077'
4205  005062' 000000  005054'
4206  005063' 001000  000343          60360          XTHL                    ;[H,L]=PLACE TO STORE VARIABLE VALUE
4207  005064' 001000  000325          60380          PUSH    D               ;TEXT POINTER GOES ON
4208  005065' 001000  000303          60400          JMP     INPCOM>         ;DO ASSIGNMENT
4209  005066' 000000  004157'
4210  005067' 000000  005061'
4211  005070' 001000  000327          60420  NUMINS: CHRGET
4212  005071' 001000  000315          60440          CALL    FIN
4213  005072' 000000  000000*
4214  005073' 000000  005066'
4215                                   60460  IFE     LENGTH=2,<
4216  005074' 001000  000303          60480          JMP     DOASIG>         ;ASSIGNMENT IS COMPLICATED
4217  005075' 000000  005056'
4218  005076' 000000  005072'
4219                                   60500                                  ;EVEN FOR NUMERICS SO USE THE "LET" CODE
4220                                   60520  IFN     LENGTH=2,<
4221                                   60540          XTHL                    ;[H,L] GET POINTER AT VARIABLE
4222                                   60560          CALL    MOVMF           ;DO THE ASSIGNMENT
4223                                   60580          POP     H>>             ;GET BACK THE TEXT POINTER
4224  005077'                          60600  STRDN2:
4225                                   60620  IFN     LENGTH,<
4226  005077' 001000  000053          60640          DCX     H
4227  005100' 001000  000327          60660          CHRGET
4228  005101' 001000  000312          60680          JZ      TRMOK
4229  005102' 000000  005111'
4230  005103' 000000  005075'
4231  005104' 001000  000376          60700          CPI     44
4232  005105' 000000  000054
4233  005106' 001000  000302          60720          JNZ     TRMNOK>         ;ENDED PROPERLY?
4234  005107' 000000  004667'
4235  005110' 000000  005102'
4236  005111' 001000  000343          60740  TRMOK:  XTHL
4237  005112' 001000  000053          60760          DCX     H               ;LOOK AT TERMINATOR
4238  005113' 001000  000327          60780          CHRGET                  ;AND SET UP CONDITION CODES
4239  005114' 001000  000302          60800          JNZ     LOPDT2          ;NOT ENDING, CHECK FOR COMMA
4240  005115' 000000  004773'
4241  005116' 000000  005107'
4242                                   60820                                  ;AND GET ANOTHER VARIABLE
4243                                   60840                                  ;TO FILL WITH DATA
4244
4245  005117' 001000  000321          60880          POP     D               ;POP OFF THE POINTER INTO DATA
4246  005120' 001000  000072          60900          LDA     FLGINP          ;FETCH THE STATEMENT TYPE FLAG
4247  005121' 000000  001602'
4248  005122' 000000  005115'
4249  005123' 001000  000267          60920          ORA     A
4250                                   60940  IFE     LENGTH,<RZ>             ;DON'T UPDATE DATPTR IF IT WAS AN
4251                                   60960                                  ;INPUT STATEMENT
4252  005124' 001000  000353          60980          XCHG
4253  005125' 001000  000302          61000          JNZ     RESFIN          ;UPDATE DATPTR
```

```
 4254  005126' 000000  003453'
 4255  005127' 000000  005121'
 4256                          61020   IFN     LENGTH,<
 4257  005130' 001000  000266  61040           ORA     M                       ;COULD HAVE ENDED WITH COMMA OR
 4258                          61060                                           ;COLON, BUT SHOULD BE A ZERO
 4259  005131' 001000  000041  61080           LXI     H,EXIGNT                ;TEXT FOR "EXTRA"
 4260  005132' 000000  005142'
 4261  005133' 000000  005126'
 4262  005134' 001000  000325  61100           PUSH    D                       ;SAVE THE TEXT POINTER
 4263  005135' 001000  000304  61120           CNZ     $TROUT                  ;IF WASN'T REAL END SAY SOMETHING
 4264  005136' 000000  007743'
 4265  005137' 000000  005132'
 4266  005140' 001000  000341  61140           POP     H                       ;GET BACK THE TEXT POINTER
 4267  005141' 001000  000311  61160           RET
 4268  005142' 000000  000077  61180   EXIGNT: DC"?EXTRA IGNORED"
 4269  005143' 000000  000105
 4270  005144' 000000  000130
 4271  005145' 000000  000124
 4272  005146' 000000  000122
 4273  005147' 000000  000101
 4274  005150' 000000  000040
 4275  005151' 000000  000111
 4276  005152' 000000  000107
 4277  005153' 000000  000116
 4278  005154' 000000  000117
 4279  005155' 000000  000122
 4280  005156' 000000  000105
 4281  005157' 000000  000104
 4282  005157' 000000  000304
 4283  005160' 000000  000015  61200           ACRLF
 4284  005161' 000000  000012
 4285  005162' 000000  000000  61220           0>
 4286                          61240   ;
 4287                          61260   ; SUBROUTINE TO FIND DATA
 4288                          61280   ; IN THE 4K "DATA" MUST BE AT THE START OF THE LINE
 4289                          61300   ; SO THE SEARCH IS MADE USING THE LINKS AT THE START OF EACH LINE.
 4290                          61320   ;
 4291                          61340   ; IN THE 8K AND EXTENDED THE SEARCH IS MADE BY USING THE EXECUTION CODE
 4292                          61360   ; FOR DATA TO SKIP OVER STATEMENTS, THE START WORD OF EACH STATEMENT
 4293                          61380   ; IS COMPARED WITH DATATK, EACH NEW LINE NUMBER
 4294                          61400   ; IS STORED IN DATLIN SO THAT IF AN ERROR OCCURS WHILE READING
 4295                          61420   ; DATA THE ERROR MESSAGE WILL GIVE THE LINE NUMBER OF THE
 4296                          61440   ; ILL-FORMATTED DATA
 4297                          61460   ;
 4298  005163' 001000  000315  61480   DATLOP: IFN     LENGTH,<CALL    DATA>
 4299  005164' 000000  004072'
 4300  005165' 000000  005136'
 4301                          61500           IFE     LENGTH,<POP     H>
 4302  005166'                 61520   DATFND: IFN     LENGTH,<
 4303  005166' 001000  000267  61540           ORA     A
 4304  005167' 001000  000302  61560           JNZ     NOWLIN>
 4305  005170' 000000  005214'
 4306  005171' 000000  005164'
```



```
 4307                          61580   IFN     LENGTH,<
 4308  005172' 001000  000043  61600           INX     H>
 4309  005173' 001000  000367  61620           PUSHM                           ;SAVE POINTER TO THE NEXT STATEMENT
 4310  005174' 001000  000171  61640           MOV     A,C                     ;SEE IF WE ARE AT THE END
 4311  005175' 001000  000260  61660           ORA     B
 4312  005176' 001000  000036  61680           MVI     E,ERROD                 ;NO DATA IS ERROR ERROD
 4313  005177' 000000  000004
 4314  005200' 001000  000312  61700           JZ      ERROR                   ;IF SO COMPLAIN
 4315  005201' 000000  002102'
 4316  005202' 000000  005170'
 4317                          61720   IFE     LENGTH,<INX     H>      ;SKIP PAST LINE #
 4318  005203' 001000  000301  61740   IFN     LENGTH,<POP     B
 4319  005204' 001000  000136  61760           MOV     E,M                     ;GET DATA LINE #
 4320  005205' 001000  000043  61780           INX     H
 4321  005206' 001000  000126  61800           MOV     D,M
 4322  005207' 001000  000353  61820           XCHG
 4323  005210' 001000  000042  61840           SHLD    DATLIN
 4324  005211' 000000  001577'
 4325  005212' 000000  005201'
 4326  005213' 001000  000353  61860           XCHG>                           ;RESTORE TEXT POINTER
 4327  005214' 001000  000327  61880   NOWLIN: CHRGET                          ;GET THE STATEMENT TYPE
 4328  005215' 001000  000376  61900           CPI     DATATK                  ;IS IS "DATA"?
 4329  005216' 000000  000203
 4330  005217' 001000  000302  61920           JNZ     DATLOP                  ;NOT DATA SO LOOK SOME MORE
 4331  005220' 000000  005163'
 4332  005221' 000000  005211'
 4333                          61940   IFE     LENGTH,<POP     B>
 4334  005222' 001000  000303  61960           JMP     DATBK                   ;CONTINUE READING
 4335  005223' 000000  005025'
 4336  005224' 000000  005220'
 4337
 4338
 4339
 4340                          62040   PAGE
```

```
                                        62060   SUBTTL NEXT CODE
                                        62080   ;
                                        62100   ; NOTE:
                                        62120   ;
                                        62140   ;
                                        62160   ; A FOR ENTRY ON THE STACK HAS THE FOLLOWING FORMAT:
                                        62180   ;
                                        62200   ;
                                        62220   ; LOW ADDRESS
                                        62240   ;       TOKEN (FORTK IN HIGH BYTE)  1 BYTES
                                        62260   ;       A POINTER TO THE LOOP VARIABLE  2 BYTES
                                        62280   ;       A BYTE REFLECTING THE SIGN OF THE INCREMENT 1 BYTE
                                        62300   ;       THE STEP 4 BYTES
                                        62320   ;       THE UPPER VALUE 4 BYTES
                                        62340   ;       THE LINE # OF THE "FOR" STATEMENT 2 BYTES
                                        62360   ;       A TEXT POINTER INTO THE "FOR" STATEMENT 2 BYTES
                                        62380   ; HIGH ADDRESS
                                        62400   ;
                                        62420   ; TOTAL 16 BYTES
                                        62440   ;
005225'                                 62460   NEXT:
005225' 001000  000021                  62480   IFN     LENGTH,<LXI     D,$CODE>;FOR THE "NEXT"
005226' 000000  000000'
005227' 000000  005223'
                                        62500                                   ;STATEMENT WITHOUT ANY ARGS
                                        62520                                   ;WE CALL FNDFOR WITH (D,E)=0
005230'                                 62540   NEXTC:
                                        62560   IFE     LENGTH,<
                                        62580           CALL    PTRGET>         ;MUST HAVE A VARIABLE
                                        62600   IFN     LENGTH,<
005230' 001000  000304                  62620           CNZ     PTRGET>         ;GET A POINTER TO THE
005231' 000000  006505'
005232' 000000  005226'
                                        62640                                   ;LOOP VARIABLE INTO (D,E)
005233' 001000  000042                  62660           SHLD    TEMP            ;PUT THE TEXT POINTER
005234' 000000  001605'
005235' 000000  005231'
                                        62680                                   ;IN A TEMP LOCATION
                                        62700                                   ;IN CASE THE LOOP TERMINATES
005236' 001000  000315                  62720           CALL    FNDFOR          ;TRY TO FIND A FOR ENTRY
005237' 000000  001744'
005240' 000000  005234'
                                        62740                                   ;ON THE STACK WHOSE VARIABLE NAME
                                        62760                                   ;MATCHES THIS ONES
                                        62780   IFN     LENGTH,<
005241' 001000  000302                  62800           JNZ     NFERR>          ;"NEXT WITHOUT FOR"
005242' 000000  002100'
005243' 000000  005237'
005244' 001000  000371                  62820           SPHL                    ;SETUP STACK POINTER BY CHOPPING
                                        62840                                   ;AT THIS POINT
005245' 001000  000325                  62860           PUSH    D               ;PUT THE VARIABLE PTR BACK ON
005246' 001000  000176                  62880           MOV     A,M             ;STEP ONTO THE STACK
005247' 001000  000043                  62900           INX     H
```



```
005250' 001000  000365                  62920           PUSH    PSW
005251' 001000  000325                  62940           PUSH    D               ;PUT THE POINTER TO THE LOOP
                                        62960                                   ;VARIABLE ONTO THE STACK
                                        62980   IFE     LENGTH,<
                                        63000           MVI     E,ERRNF
                                        63020           JNZ     ERROR>
005252' 001000  000315                  63040           CALL    MOVFM           ;STEP VALUE INTO THE FAC
005253' 000000  000000*
005254' 000000  005242'
005255' 001000  000343                  63060           XTHL                    ;PUT THE POINTER INTO THE
                                        63080                                   ;FOR ENTRY ONTO THE STACK
005256' 001000  000345                  63100           PUSH    H               ;PUT THE POINTER TO THE LOOP
                                        63120                                   ;VARIABLE BACK ONTO THE STACK
005257' 001000  000315                  63140           CALL    FADDS           ;ADD THE STEP AND LOOP VARIABLE
005260' 000000  000000*
005261' 000000  005253'
005262' 001000  000341                  63160           POP     H               ;POP OFF THE POINTER TO
                                        63180                                   ;THE LOOP VARIABLE
005263' 001000  000315                  63200           CALL    MOVMF           ;MOV FAC INTO LOOP VARIABLE
005264' 000000  000000*
005265' 000000  005260'
005266' 001000  000341                  63220           POP     H               ;GET THE ENTRY POINTER
005267' 001000  000315                  63240           CALL    MOVRM           ;GET THE FINAL INTO THE REGISTERS
005270' 000000  000000*
005271' 000000  005264'
005272' 001000  000345                  63260           PUSH    H               ;SAVE THE ENTRY POINTER
005273' 001000  000315                  63280           CALL    FCOMP           ;COMPARE THE NUMBERS
005274' 000000  000714*
005275' 000000  005270'
005276' 001000  000341                  63300           POP     H               ;STILL POINTING TO THE FINAL VALUE
005277' 001000  000301                  63320           POP     B               ;GET THE SIGN  OF THE INCREMENT
005300' 001000  000220                  63340           SUB     B               ;SUBTRACT THE INCREMENTS SIGN FROM THAT
                                        63360                                   ;OF (CURRENT VALUE-FINAL VALUE)
005301' 001000  000315                  63380           CALL    MOVRM           ;GET LINE # OF "FOR" INTO (D,E)
005302' 000000  005270*
005303' 000000  005274'
                                        63400                                   ;GET TEXT POINTER OF "FOR" INTO (B,C)
005304' 001000  000312                  63420           JZ      LOOPDN          ;IF SIGN(FINAL-CURRENT)+SIGN(STEP)=0
005305' 000000  005320'
005306' 000000  005302'
                                        63440                                   ;THEN THE LOOP IS FINISHED
005307' 001000  000353                  63460           XCHG
005310' 001000  000042                  63480           SHLD    CURLIN          ;STORE THE LINE #
005311' 000000  001607'
005312' 000000  005305'
005313' 001000  000151                  63500           MOV     L,C             ;SETUP THE TEXT POINTER
005314' 001000  000140                  63520           MOV     H,B
005315' 001000  000303                  63540           JMP     NXTCON
005316' 000000  003276'
005317' 000000  005311'
                                        63560
005320' 001000  000371                  63580   LOOPDN: SPHL                    ;ELIMINATE THE FOR ENTRY
                                        63600                                   ;SINCE (H,L) MOVED ALL
```

```
                                                63620                                   ;THE WAY DOWN THE ENTRY
 4448  005321' 001000  000052                   63640           LHLD    TEMP            ;RESTORE THE TEXT POINTER
 4449  005322' 000000  001603'
 4450  005323' 000000  005316'
 4451                                           63660   IFE     LENGTH,<JMP     NEWSTT>
 4452                                           63680   IFN     LENGTH,<
 4453  005324' 001000  000176                   63700           MOV     A,M             ;IS THERE A COMMA AT THE END
 4454  005325' 001000  000376                   63720           CPI     44              ;IF SO LOOK AT ANOTHER
 4455  005326' 000000  000054
 4456  005327' 001000  000302                   63740           JNZ     NEWSTT          ;VARIABLE NAME TO "NEXT"
 4457  005330' 000000  003302'
 4458  005331' 000000  005322'
 4459  005332' 001000  000327                   63760           CHRGET                  ;READ FIRST CHARCTER
 4460  005335' 001000  000315                   63780           CALL    NEXTC>          ;DO NEXT, BUT DON'T ALLOW
 4461  005334' 000000  005230'
 4462  005335' 000000  005330'
 4463                                           63800                                   ;BLANK VARIABLE NAME (D,E)=STK PTR
 4464                                           63820                                   ;AND WILL NEVER MATCH ANY VARPTR
 4465                                           63840                                   ;USE CALL TO PUT DUMMY "NEWSTT" ENTRY ON
 4466
 4467                                           63880   PAGE
```



```
 4468
 4469
```

```
                                00020   SUBTTL FORMULA EVALUATION CODE
                                00040
                                00060   IFN     LENGTH-2,<
                                00080   IFN     STRING,<
                                00100   ;
                                00120   ; THESE ROUTINES CHECK FOR A CERTAIN VALTYP
                                00140   ; (A) IS NOT PRESERVED
                                00160   ;
                                00180   FRMNUM: CALL    FRMEVL          ;EVALUATE A FORMULA
                                00200   CHKNUM: XWD     "01000,"0366    ;TURN CARRY OFF WITH ORI
                                00220   CHKSTR: STC                     ;SET CARRY
                                00240   CHKVAL: LDA     VALTYP          ;0 MEANS NUMERIC 1 MEANS STRING
                                00260           ADC     A               ;RESULT SHOULD BE 0 OR 3
                                00280                                   ;BAD RESULTS ARE 2 AND 1
                                00300           RPE                     ;RETURN IF CORRECT RESULT
                                00320   TMERR:  MVI     E,ERRTM         ;"TYPE MISMATCH ERROR"
                                00340           JMP     ERROR>>
                                00360   ;
                                00380   ; THE FORMULA EVALUATOR STARTS WITH
                                00400   ; (H,L) POINTING TO THE FIRST CHARACTER OF THE FORMULA,
                                00420   ; AT THE END (H,L) POINTS TO THE TERMINATOR,
                                00440   ; THE RESULT IS LEFT IN THE FAC,
                                00460   ;
                                00480   ; THE FORMULA EVALUATOR USES THE OPERATOR TABLE (OPTAB)
                                00500   ; TO DETERMINE PRECEDENCE AND DISPATCH ADDRESSES FOR
                                00520   ; EACH OPERATOR,
                                00540   ; A TEMPORARY RESULT ON THE STACK HAS THE FOLLOWING FORMAT
                                00560   ;
                                00580   ; THE ADDRESS OF 'RETAOP' -- THE PLACE TO RETURN ON COMPLETION
                                00600   ; OF OPERATOR APPLICATION
                                00620   ;
                                00640   ; THE FLOATING POINT TEMPORARY RESULT
                                00660   ;
                                00680   ; THE ADDRESS OF THE OPERATOR ROUNTINE
                                00700   ;
                                00720   ; THE PRECEDENCE OF THE OPERATOR
                                00740   ;
                                00760   ; TOTAL 10 BYTES
                                00780   ;
                                00800   IFE     STRING,<FRMNUM:>
005336' 001000  000053          00820   FRMEVL: DCX     H               ;BACK UP CHARACTER POINTER
005337' 001000  000026          00840   FRMCHK: MVI     D,0             ;INITIAL DUMMY PRECEDENCE IS 0
005340' 000000  000000
005341' 001000  000325          00860   LPOPER: PUSH    D               ;SAVE PRECEDENCE
005342' 001000  000315          00880           CALL    GETSTK          ;MAKE SURE THERE IS ROOM FOR RECURSIVE CALLS
005343' 000000  002024'
005344' 000000  005334'
005345' 000000  000001          00900           1
005346' 001000  000315          00920           CALL    EVAL            ;EVALUATE SOMETHING
005347' 000000  006061'
005350' 000000  005343'
005351' 001000  000042          00940           SHLD    TEMP2           ;SAVE TEXT POINTER
005352' 000000  001605'
```



```
005353' 000000  005347'
005354' 001000  000052          00960   RETAOP: LHLD    TEMP2           ;RESTORE TEXT PTR
005355' 000000  001605'
005356' 000000  005352'
005357' 001000  000301          00980   TSTOP:  POP     B               ;POP OFF THE PRECEDENCE OF OLDOP
                                01000           IFN     LENGTH-2,<
                                01020           IFN     STRING,<
                                01040           MOV     A,B             ;LOOK AT PRECEDENCE
                                01060           CPI     120             ;IF ITS SOME ARITHMETIC
                                01080                                   ;TYPE THING WE SHOULDN'T
                                01100                                   ;SEE STRINGS
                                01120           CNC     CHKNUM>>        ;TYPE MISMATCH ERROR IF NOT NUMERIC
005360' 001000  000176          01140   NOTSTV: MOV     A,M             ;GET NEXT CHARACTER
                                01160           IFN     LENGTH,<
005361' 001000  000026          01180           MVI     D,0             ;ASSUME NO RELATION OPS
005362' 000000  000000
005363' 001000  000326          01200   LOPREL: SUI     GREATK          ;IS THIS ONE RELATION?
005364' 000000  000257
005365' 001000  000332          01220           JC      ENDREL          ;RELATIONS ALL THROUGH
005366' 000000  005415'
005367' 000000  005355'
                                000003          NMREL==LESSTK-GREATK+1
005370' 001000  000376          01260           CPI     NMREL           ;IS IT REALLY RELATIONAL?
005371' 000000  000003
005372' 001000  000322          01280           JNC     ENDREL          ;NO JUST BIG
005373' 000000  005415'
005374' 000000  005366'
005375' 001000  000376          01300           CPI     1               ;SET UP BITS BY MAPPING
005376' 000000  000001
005377' 001000  000027          01320           RAL                     ;0 TO 1 1 TO 2 AND 2 TO 4
005400' 001000  000252          01340           XRA     D               ;BRING IN THE OLD BITS
005401' 001000  000272          01360           CMP     D               ;MAKE SURE RESULT IS BIGGER
005402' 001000  000127          01380           MOV     D,A             ;SAVE THE MASK
005403' 001000  000332          01400           JC      SNERR           ;DON'T ALLOW TWO OF THE SAME
005404' 000000  002072'
005405' 000000  005373'
005406' 001000  000042          01420           SHLD    TEMP3           ;SAVE CHARACTER POINTER
005407' 000000  001575'
005410' 000000  005404'
005411' 001000  000327          01440           CHRGET                  ;GET THE NEXT CANDIDATE
005412' 001000  000303          01460           JMP     LOPREL
005413' 000000  005363'
005414' 000000  005407'
005415' 001000  000172          01480   ENDREL: MOV     A,D             ;GET THE MASK
005416' 001000  000267          01500           ORA     A               ;WERE THERE ANY?
005417' 001000  000302          01520           JNZ     FINREL          ;IF SO, HANDLE AS SPECIAL OP
005420' 000000  005605'
005421' 000000  005413'
005422' 001000  000176          01540           MOV     A,M             ;GET THE CHARACTER AGAIN
005423' 001000  000042          01560           SHLD    TEMP3>          ;SAVE UPDATED CHARACTER POINTER
005424' 000000  001575'
005425' 000000  005420'
005426' 001000  000326          01580           SUI     PLUSTK          ;AN OPERATOR?
```

```
4576  005427' 000000  000250
4577  005430' 001000  000330          01600           RC                      ;RETURN IF NOT
4578                                   01620                                   ;THIS CAN RESULT IN OPERATOR
4579                                   01640                                   ;APPLICATION OR ACTUAL RETURN
4580  005431' 001000  000376          01660           CPI     LSTOPK          ;HIGHER THAN THE LAST OP?
4581  005432' 000000  000007
4582  005433' 001000  000320          01680           RNC
4583  005434' 001000  000137          01700           MOV     E,A             ;MUST MUTIPLY BY 3 SINCE
4584                                   01720                                   ;OPTAB ENTRIES ARE 3 LONG
4585                                   01740   IFN     STRING,<
4586  005435' 001000  000072          01760           LDA     VALTYP          ;SEE IF LEFT PART IS STRING
4587  005436' 000000  001543'
4588  005437' 000000  005424'
4589                                   01780   IFE     LENGTH=2,<
4590  005440' 001000  000376          01800           CPI     3>              ;SEE IF ITS A STRING
4591  005441' 000000  000003
4592                                   01820   IFN     LENGTH=2,<
4593                                   01840           DCR     A>
4594  005442' 001000  000263          01860           ORA     E               ;SET CONDITION CODES
4595                                   01880   IFN     LENGTH=2,<
4596                                   01900           MOV     A,E>            ;REFETCH OP-VALUE
4597  005443' 001000  000312          01920           JZ      CAT>            ;MUST BE CAT
4598  005444' 000000  010320'
4599  005445' 000000  005436'
4600                                   01940   IFN     LENGTH=2,<
4601                                   01960           RLC                     ;A= ORIGINAL A*2
4602                                   01980           ADD     E               ;ADD IN ORIGINAL A
4603                                   02000           MOV     E,A>            ;CREATE TWO BYTE VALUE
4604                                   02020   IFE     LENGTH,<
4605                                   02040           MVI     D,0>            ;HIGH ORDER =0
4606  005446' 001000  000041          02060           LXI     H,OPTAB         ;CREATE INDEX INTO OPTAB
4607  005447' 000000  000163'
4608  005450' 000000  005444'
4609  005451' 001000  000031          02080           DAD     D               ;ADD IN CALCULATED OFFSET
4610  005452' 001000  000170          02100           MOV     A,B             ;[A] GETS OLD PRECEDENCE
4611  005453' 001000  000126          02120           MOV     D,M             ;REMEMBER NEW PRECEDENCE
4612  005454' 001000  000272          02140           CMP     D               ;OLD=NEW
4613  005455' 001000  000320          02160           RNC                     ;MUST APPLY OLD OP
4614                                   02180                                   ;IF HAS GREATER OR = PRECEDENCE
4615                                   02200   IFN     LENGTH=2,<
4616                                   02220           INX     H               ;NOW POINTING AT ROUTINE ADDRESS
4617                                   02240   IFN     STRING,<CALL    CHKNUM> ;CAN'T BE STRING HERE
4618                                   02260                                   ;SINCE THE ONLY STRING OPERATOR
4619                                   02280                                   ;IS PLUS AND RELATIONALS
4620                                   02300                                   ;DON'T COME THROUGH HERE
4621                                   02320   DOPREC: PUSH    B               ;SAVE OLD PRECEDENCE
4622                                   02340           LXI     B,RETAOP        ;OPERATOR RETURN ADDRESS
4623                                   02360           PUSH    B               ;FIRST PART OF "TEMP" ENTRY
4624                                   02380   IFN     LENGTH,<
4625                                   02400           MOV     B,E>            ;SAVE SECOND BYTE OF PRECEDENCE
4626                                   02420                                   ;SINCE FOR RELATIONAL OPERATORS
4627                                   02440                                   ;IT GIVES THE VALUE TYPE OF
4628                                   02460                                   ;THE LEFT SIDE AND IT TELLS
```



```
4629                                   02480                                   ;WHICH RELATIONAL-OPERATOR
4630                                   02500                                   ;IT WAS
4631                                   02520           MOV     C,D
4632                                   02540           CALL    PUSHF
4633                                   02560   IFN     LENGTH,<
4634                                   02580           MOV     E,B>            ;GET SECOND BYTE OF PRECEDENCE AGAIN
4635                                   02600           MOV     D,C             ;[D] GETS PRECEDENCE
4636                                   02620           PUSHM                   ;PUT ROUTINE ADDRESS ON THE STACK
4637                                   02640   IFN     LENGTH,<
4638                                   02660           LHLD    TEMP3>
4639                                   02680   IFE     LENGTH,<
4640                                   02700           LHLD    TEMP2>          ;IF WE DONT HAVE "LENGTH"
4641                                   02720                                   ;ON OPERATORS CAN ONLY
4642                                   02740                                   ;BE ONE CHARACTER SO TO "REREAD"
4643                                   02760                                   ;AN OPERATOR THAT WE LOOKED
4644                                   02780                                   ;AT BEFORE AND DECIDED NOT TO
4645                                   02800                                   ;APPLY WE JUST "DCX H"
4646                                   02820                                   ;IF LENGTH IS ON WE HAVE TO
4647                                   02840                                   ;REMEMBER THE TEXT POINTER BEFORE
4648                                   02860                                   ;THE OPERATOR AND AFTER SO WE CAN
4649                                   02880                                   ;EITHER RESCAN THE OPERATOR
4650                                   02900                                   ;LATER IF IT DOESN'T GET APPLIED
4651                                   02920                                   ;OR GO BEYOND IT WHEN IT DOES
4652                                   02940           JMP     LPOPER>         ;PUT ON PRECEDENCE AND LOOK AT A
4653                                   02960                                   ;NEW OPERATOR
4654
4655                                   03000   IFE     LENGTH=2,<
4656  005456' 001000  000305          03020           PUSH    B               ;SAVE THE OLD PRECEDENCE
4657  005457' 001000  000001          03040           LXI     B,RETAOP        ;PUT ON THE ADDRESS OF THE
4658  005460' 000000  005354'
4659  005461' 000000  005447'
4660                                   03060           PUSH    B               ;PLACE TO RETURN TO AFTER OPERATOR APPLICATION
4661  005462' 001000  000305
4662  005463' 001000  000172          03080           MOV     A,D             ;SEE IF THE OPERATOR IS EXPONENTIATION
4663  005464' 001000  000376          03100           CPI     127             ;WHICH HAS PRECEDENCE 127
4664  005465' 000000  000177
4665                                   03120           JZ      EXPSTK          ;IF SO, "FRCSNG" AND MAKE A SPECIAL STACK ENTRY
4666  005466' 001000  000312
4667  005467' 000000  005553'
4668  005470' 000000  005460'
4669  005471' 001000  000376          03140           CPI     81              ;SEE IF THE OPERATOR IS "AND" OR "OR"
4670  005472' 000000  000121
4671  005473' 001000  000332          03160           JC      ANDORD          ;AND IF SO "FRCINT" AND
4672  005474' 000000  005571'
4673  005475' 000000  005467'
4674                                   03162                                   ;MAKE A SPECIAL STACK ENTRY
4675                                   03200   ;
4676                                   03220   ; THIS CODE PUSHES THE CURRENT VALUE IN THE FAC
4677                                   03240   ; ONTO THE STACK, EXCEPT IN THE CASE OF STRINGS IN WHICH IT CALLS
4678                                   03260   ; TYPE MISMATCH ERROR, [D] AND [E] ARE PRESERVED.
4679                                   03280   ;
4680  005476' 001000  000072          03300   NUMREL: LDA     VALTYP          ;GET THE VALUE TYPE
4681  005477' 000000  001543'
```

```
 4682  005500' 000000  005474'
 4683  005501' 001000  000376          03302           CPI     3               ;AND SET THE CONDITION CODES BASED ON IT
 4684  005502' 000000  000003
 4685  005503' 001000  000312          03320           JZ      TMERR           ;BLOW UP ON STRINGS
 4686  005504' 000000  000000*
 4687  005505' 000000  005477'
 4688  005506' 001000  000041          03340           LXI     H,FACLO         ;GET POINTER TO LO IN FAC
 4689  005507' 000000  001637'
 4690  005510' 000000  005504'
 4691  005511' 001000  000116          03342           MOV     C,M
 4692  005512' 001000  000043          03344           INX     H
 4693  005513' 001000  000106          03345           MOV     B,M
 4694  005514' 001000  000043          03348           INX     H
 4695  005515' 001000  000305          03360           PUSH    B               ;PUSH FACLO+0,1 ON THE STACK
 4696  005516' 001000  000372          03380           JM      VPUSHD          ;ALL DONE IF THE DATA WAS AN INTEGER
 4697  005517' 000000  005536'
 4698  005520' 000000  005507'
 4699  005521' 001000  000116          03382           MOV     C,M
 4700  005522' 001000  000043          03384           INX     H
 4701  005523' 001000  000106          03386           MOV     B,M
 4702  005524' 001000  000043          03388           INX     H
 4703  005525' 001000  000305          03400           PUSH    B               ;PUSH FAC-1,0 ON THE STACK
 4704  005526' 001000  000342          03420           JPO     VPUSHD          ;ALL DONE IF WE HAD A SNG
 4705  005527' 000000  005536'
 4706  005530' 000000  005517'
 4707  005531' 001000  000041          03440           LXI     H,DFACLO        ;WE HAVE A DOUBLE PRECISON NUMBER
 4708  005532' 000000  001633'
 4709  005533' 000000  005527'
 4710  005534' 001000  000367          03460           PUSHM                   ;PUSH ITS 4 LO BYTES ON THE STACK
 4711  005535' 001000  000367          03480           PUSHM
 4712  005536' 001000  000113          03520   VPUSHD: MOV     C,E             ;(C)=OPERATOR NUMBER
 4713  005537' 001000  000107          03560           MOV     B,A             ;(B)=TYPE OF VALUE ON THE STACK
 4714  005540' 001000  000305          03580           PUSH    B               ;SAVE THESE THINGS FOR APPLOP
 4715  005541' 001000  000001          03600           LXI     B,APPLOP        ;GENERAL OPERATOR APPLICATION
 4716  005542' 000000  005642'
 4717  005543' 000000  005532'
 4718                                   03620                                   ;ROUTINE -- DOES TYPE CONVERSIONS
 4719  005544' 001000  000305          03640   FINTMP: PUSH    B               ;SAVE PLACE TO GO
 4720  005545' 001000  000052          03660           LHLD    TEMP3           ;REGET THE TEXT POINTER
 4721  005546' 000000  001575'
 4722  005547' 000000  005542'
 4723  005550' 001000  000303          03680           JMP     LPOPER          ;PUSH ON THE PRECEDENCE AND READ MORE
 4724  005551' 000000  005341'
 4725  005552' 000000  005546'
 4726                                   03700                                   ;FORMULA
 4727                                   03720   ;
 4728                                   03740   ; FOR EXPONENTIATION WE WANT TO FORCE THE CURRENT VALUE IN THE FAC
 4729                                   03760   ; TO BE SINGLE PRECISION, WHEN APPLICATION TIME COMES WE FORCE
 4730                                   03780   ; THE RIGHT HAND OPERAND TO SINGLE PRECISION AS WELL
 4731                                   03800   ;
 4732  005553' 001000  000315          03820   EXPSTK: CALL    FRCSNG          ;COERCE LEFT HAND OPERAND
 4733  005554' 000000  003e56*
 4734  005555' 000000  005551'
```



```
 4735  005556' 001000  000315          03840           CALL    PUSHF           ;PUT IT ON THE STACK
 4736  005557' 000000  000000*
 4737  005560' 000000  005554'
 4738  005561' 001000  000001          03860           LXI     B,FPWRQ##       ;PLACE TO COERCE RIGHT HAND
 4739  005562' 000000  000000*
 4740  005563' 000000  005557'
 4741                                   03880                                   ;OPERAND AND DO EXPONENTIATION
 4742  005564' 001000  000026          03900           MVI     D,127           ;RESTORE THE PRECEDENCE
 4743  005565' 000000  000177
 4744  005566' 001000  000303          03920           JMP     FINTMP          ;FINISH ENTRY AND EVALUATE MORE FORMULA
 4745  005567' 000000  005544'
 4746  005570' 000000  005562'
 4747                                   03940   ;
 4748                                   03960   ; FOR "AND" AND "OR" WE WANT TO FORCE THE CURRENT VALUE IN THE
 4749                                   03980   ; FAC TO BE AN INTEGER, AND AT APPLICATION TIME FORCE THE RIGHT
 4750                                   04000   ; HAND OPERAND TO BE AN INTEGER
 4751                                   04020   ;
 4752  005571' 001000  000325          04040   ANDORD: PUSH    D               ;SAVE THE PRECEDENCE (70 OR 80)
 4753  005572' 001000  000315          04060           CALL    FRCINT
 4754  005573' 000000  003630*
 4755  005574' 000000  005567'
 4756  005575' 001000  000321          04080           POP     D               ;(D)=PRECEDENCE
 4757  005576' 001000  000345          04100           PUSH    H               ;PUSH THE LEFT HAND OPERAND
 4758  005577' 001000  000001          04120           LXI     B,DANDOR        ;"AND" AND "OR" DOER
 4759  005600' 000000  006437'
 4760  005601' 000000  005573'
 4761  005602' 001000  000303          04140           JMP     FINTMP          ;PUSH ON THIS ADDRESS,PRECEDENCE
 4762  005603' 000000  005544'
 4763  005604' 000000  005600'
 4764                                   04160                                   ;AND CONTINUE EVALUATION
 4765                                   04180   ;
 4766                                   04200   ; HERE TO BUILD AN ENTRY FOR A RELATIONAL OPERATOR
 4767                                   04220   ; STRINGS ARE TREATED SPECIALLY, NUMERIC COMPARES ARE DIFFERENT
 4768                                   04240   ; FROM MOST OPERATOR ENTRIES ONLY IN THE FACT THAT AT THE
 4769                                   04260   ; BOTTOM INSTEAD OF HAVING RETAUP, DOCMP AND THE RELATIONAL
 4770                                   04280   ; BITS ARE STORED, STRINGS HAVE STRCMP,THE POINTER AT THE STRING DESCRIPTOR,
 4771                                   04300   ; DOCMP AND THE RELATIONAL BITS.
 4772                                   04320   ;
 4773  005605' 001000  000170          04340   FINREL: MOV     A,B             ;(A)=OLD PRECEDENCE
 4774  005606' 001000  000376          04360           CPI     100             ;RELATIONALS HAVE PRECEDENCE 100
 4775  005607' 000000  000144
 4776  005610' 001000  000320          04380           RNC                     ;APPLY EARLIER OPERATOR IF IT HAS
 4777                                   04400                                   ;HIGHER PRECEDENCE
 4778  005611' 001000  000305          04420           PUSH    B               ;SAVE THE OLD PRECEDENCE
 4779  005612' 001000  000325          04440           PUSH    D               ;SAVE (D)=RELATIONAL BITS
 4780  005613' 001000  000021          04460           LXI     D,$CODE+25604   ;(D)=PRECEDENCE=100
 4781  005614' 000000  262004'
 4782  005615' 000000  005603'
 4783                                   04480                                   ;(E)=DISPATCH OFFSET FOR
 4784                                   04500                                   ;COMPARES IN APPLOP=4
 4785                                   04520                                   ;IN CASE THIS IS A NUMERIC COMPARE
 4786  005616' 001000  000041          04540           LXI     H,DOCMP         ;ROUTINE TO TAKE COMPARE ROUTINE RESULT
 4787  005617' 000000  006375'
```

```
 4788  005620' 000000  005614'
 4789                          04560                           ;AND RELATIONAL BITS AND RETURN THE ANSWER
 4790  005621' 001000  000345 04580          PUSH    H        ;DOES A JMP TO RETAOP WHEN DONE
 4791  005622' 001000  000315 04600          CALL    GETYPE   ;SEE IF WE HAVE A NUMERIC COMPARE
 4792  005623' 000000  006307'
 4793  005624' 000000  005617'
 4794  005625' 001000  000302 04620          JNZ     NUMREL   ;YES, BUILD AN APPLOP ENTRY
 4795  005626' 000000  005476'
 4796  005627' 000000  005623'
 4797  005630' 001000  00005E 04640          LHLD    FACLO    ;GET THE POINTER AT THE STRING DESCRIPTOR
 4798  005631' 000000  001637'
 4799  005632' 000000  005626'
 4800  005635' 001000  000345 04660          PUSH    H        ;SAVE IT FOR STRCMP
 4801  005634' 001000  000001 04680          LXI     B,STRCMP ;STRING COMPARE ROUTINE
 4802  005635' 000000  006320'
 4803  005636' 000000  005631'
 4804  005637' 001000  000303 04700          JMP     FINTMP   ;PUSH THE ADDRESS, REGET THE TEXT POINTER
 4805  005640' 000000  005544'
 4806  005641' 000000  005635'
 4807                          04720                           ;SAVE THE PRECEDENCE AND SCAN
 4808                          04722                           ;MORE OF THE FORMULA
 4809                          04740  ;
 4810                          04760  ; APPLOP IS RETURNED TO WHEN IT IS TIME TO APPLY AN ARITHMETIC
 4811                          04762  ; OR NUMERIC COMPARISON OPERATION,
 4812                          04764  ; THE STACK HAS A DOUBLE BYTE ENTRY WITH THE OPERATOR
 4813                          04766  ; NUMBER AND THE VALTYP OF THE VALUE ON THE STACK,
 4814                          04768  ; APPLOP DECIDES WHAT VALUE LEVEL THE OPERATION
 4815                          04770  ; WILL OCCUR AT, AND CONVERTS THE ARGUMENTS, APPLOP
 4816                          04772  ; USES DIFFERENT CALLING CONVENTIONS FOR EACH VALUE TYPE,
 4817                          04776  ; INTEGERS: LEFT IN [D,E] RIGHT IN [H,L]
 4818                          04778  ; SINGLES:  LEFT IN [B,C,D,E] RIGHT IN THE FAC
 4819                          04780  ; DOUBLES:  LEFT IN FAC   RIGHT IN ARG
 4820                          04798  ;
 4821  005642' 001000  000301 04800  APPLOP: POP     B        ;[B]=STACK OPERAND VALUE TYPE
 4822                          04820                           ;[C]=OPERATOR OFFSET
 4823  005643' 001000  000171 04840          MOV     A,C      ;SAVE IN MEMORY SINCE THE STACK WILL BE BUSY
 4824  005644' 001000  000062 04860          STA     OPRTYP   ;A RAM LOCATION
 4825  005645' 000000  001544'
 4826  005646' 000000  005640'
 4827  005647' 001000  000170 04880          MOV     A,B      ;CHECK FOR DOUBLE
 4828  005650' 001000  000376 04900          CPI     8        ;PRECISION ENTRY ON THE STACK
 4829  005651' 000000  000010
 4830  005652' 001000  000312 04920          JZ      STKDBL   ;FORCE FAC TO DOUBLE
 4831  005653' 000000  005725'
 4832  005654' 000000  005645'
 4833  005655' 001000  000072 04940          LDA     VALTYP   ;SEE IF THE FAC IS DOUBLE PRECISION
 4834  005656' 000000  001543'
 4835  005657' 000000  005653'
 4836  005660' 001000  000376 04960          CPI     8        ;AND IF SO, CONVERT THE STACK OPERAND
 4837  005661' 000000  000010
 4838  005662' 001000  000312 04980          JZ      FACDBL   ;TO DOUBLE PRECISION
 4839  005663' 000000  005775'
 4840  005664' 000000  005656'
```



```
 4841  005665' 001000  000127 05000          MOV     D,A      ;SAVE THE VALUE TYPE OF THE FAC
 4842  005666' 001000  000170 05020          MOV     A,B      ;SEE IF THE STACK ENTRY IS SINGLE
 4843  005667' 001000  000376 05040          CPI     4        ;PRECISION AND IF SO, CONVERT
 4844  005670' 000000  000004
 4845  005671' 001000  000312 05060          JZ      STKSNG   ;THE FAC TO SINGLE PRECISION
 4846  005672' 000000  006021'
 4847  005673' 000000  005663'
 4848  005674' 001000  000172 05080          MOV     A,D      ;SEE IF THE FAC IS SINGLE PRECISION
 4849  005675' 001000  000376 05100          CPI     3        ;AND IF SO CONVERT THE STACK TO SINGLE
 4850  005676' 000000  000003
 4851  005677' 001000  000322 05120          JNC     FACSNG   ;PRECISION
 4852  005700' 000000  006034'
 4853  005701' 000000  005672'
 4854                          05140                           ;NOTE: THE STACK MUST BE INTEGER AT THIS POINT
 4855  005702' 001000  000312 05160          JZ      TMERR    ;BLOW UP ON RIGHT HAND STRING OPERAND
 4856  005703' 000000  005504*
 4857  005704' 000000  005700'
 4858  005705' 001000  000041 05180          LXI     H,INTDSP ;INTEGER INTEGER CASE
 4859  005706' 000000  000716'
 4860  005707' 000000  005703'
 4861  005710' 001000  000006 05200          MVI     B,0      ;SPECIAL DISPATCH FOR SPEED
 4862  005711' 000000  000000
 4863  005712' 001000  000011 05220          DAD     B        ;[H,L] POINTS TO THE ADDRESS TO GO TO
 4864  005713' 001000  000011 05240          DAD     B
 4865  005714' 001000  000116 05260          MOV     C,M      ;[B,C]=ROUTINE ADDRESS
 4866  005715' 001000  000043 05280          INX     H
 4867  005716' 001000  000106 05300          MOV     B,M
 4868  005717' 001000  000321 05320          POP     D        ;[D,E]=LEFT HAND OPERAND
 4869  005720' 001000  000052 05340          LHLD    FACLO    ;[H,L]=RIGHT HAND OPERAND
 4870  005721' 000000  001637'
 4871  005722' 000000  005706'
 4872  005723' 001000  000305 05360          PUSH    B        ;DISPATCH
 4873  005724' 001000  000311 05380          RET
 4874                          05400  ;
 4875                          05420  ; THE STACK OPERAND IS DOUBLE PRECISION, SO
 4876                          05440  ; THE FAC MUST BE FORCED TO DOUBLE PRECISION, MOVED INTO ARG
 4877                          05460  ; AND THE STACK VALUE POPED INTO THE FAC
 4878                          05480  ;
 4879  005725' 001000  000315 05500  STKDBL: CALL    FRCDBL   ;MAKE THE FAC DOUBLE PRECISION
 4880  005726' 000000  000664*
 4881  005727' 000000  005721'
 4882  005730' 001000  000315 05520          CALL    VMOVAF   ;MOVE THE FAC INTO ARG
 4883  005731' 000000  000000*
 4884  005732' 000000  005726'
 4885  005735' 001000  000341 05540          POP     H        ;POP OFF THE STACK OPERAND INTO THE FAC
 4886  005734' 001000  000042 05560          SHLD    DFACLO+2
 4887  005735' 000000  001635'
 4888  005736' 000000  005731'
 4889  005737' 001000  000341 05580          POP     H
 4890  005740' 001000  000042 05600          SHLD    DFACLO   ;STORE LOW BYTES AWAY
 4891  005741' 000000  001633'
 4892  005742' 000000  005735'
 4893  005743' 001000  000301 05620  SNGDBL: POPR             ;POP OFF A FOUR BYTE VALUE
```

```
 4894  005744' 001000  000321
 4895  005745' 001000  000315          05640            CALL    MOVFR           ;INTO THE FAC
 4896  005746' 000000  000000*
 4897  005747' 000000  005741'
 4898  005750' 001000  000315          05660   SETDBL:  CALL    FRCDBL          ;MAKE SURE THE LEFT OPERAND IS
 4899  005751' 000000  005726*
 4900  005752' 000000  005746'
 4901                                   05680                                    ;DOUBLE PRECISION
 4902  005753' 001000  000041          05700            LXI     H,DBLDSP        ;DISPATCH TO A DOUBLE PRECISION ROUTINE
 4903  005754' 000000  000672'
 4904  005755' 000000  005751'
 4905  005756' 001000  000072          05720   DODSP:   LDA     OPRTYP          ;RECALL WHICH OPERAND IT WAS
 4906  005757' 000000  001544'
 4907  005760' 000000  005754'
 4908  005761' 001000  000007          05740            RLC                     ;CREATE A DISPATCH OFFSET, SINCE
 4909                                   05760                                    ;TABLE ADDRESSES ARE TWO BYTES
 4910  005762' 001000  000305          05780            PUSH    B               ;SAVE (B,C) FOR SINGLE PRECISION
 4911  005763' 001000  000117          05800            MOV     C,A             ;DOUBLE BYTE OFFSET
 4912  005764' 001000  000000          05820            MVI     B,0             ;INTO (B,C)
 4913  005765' 000000  000000
 4914  005766' 001000  000011          05840            DAD     B               ;CALCULATE LOCATION OF ROUTINE TO GO TO
 4915  005767' 001000  000301          05860            POP     B               ;GET BACK (B,C) FOR SINGLE PRECISION
 4916  005770' 001000  000176          05880            MOV     A,M             ;GET THE ADDRESS
 4917  005771' 001000  000043          05900            INX     H
 4918  005772' 001000  000146          05920            MOV     H,M
 4919  005773' 001000  000157          05940            MOV     L,A
 4920  005774' 001000  000351          05960            PCHL                    ;AND PERFORM THE OPERATION, RETURNING
 4921                                   05980                                    ;TO RETAOP, EXCEPT FOR COMPARES WHICH
 4922                                   06000                                    ;RETURN TO DOCMP
 4923                                   06020   ;
 4924                                   06040   ; THE FAC IS DOUBLE PRECISION AND THE STACK IS EITHER
 4925                                   06060   ; INTEGER OR SINGLE PRECISION AND MUST BE CONVERTED
 4926                                   06080   ;
 4927  005775' 001000  000305          06100   FACDBL:  PUSH    B               ;SAVE THE STACK VALUE TYPE
 4928  005776' 001000  000315          06120            CALL    VMOVAF          ;MOVE THE FAC INTO ARG
 4929  005777' 000000  005731*
 4930  006000' 000000  005757'
 4931  006001' 001000  000361          06140            POP     PSW             ;POP THE STACK VALUE TYPE INTO (A)
 4932  006002' 001000  000062          06160            STA     VALTYP          ;PUT IT IN VALTYP FOR THE FORCE
 4933  006003' 000000  001543'
 4934  006004' 000000  005777'
 4935                                   06180                                    ;ROUTINE
 4936  006005' 001000  000376          06200            CPI     4               ;SEE IF ITS SINGLE, SO WE KNOW
 4937  006006' 000000  000004
 4938                                   06220                                    ;HOW TO POP THE VALUE OFF
 4939  006007' 001000  000312          06240            JZ      SNGDBL          ;IT'S SINGLE PRECISION
 4940  006010' 000000  005743'
 4941  006011' 000000  006003'
 4942                                   06260                                    ;SO DO A POPR / CALL MOVFR
 4943  006012' 001000  000341          06280            POP     H               ;POP OFF THE INTEGER VALUE
 4944  006013' 001000  000042          06300            SHLD    FACLO           ;SAVE IT FOR CONVERSION
 4945  006014' 000000  001637'
 4946  006015' 000000  006010'
```



```
 4947  006016' 001000  000303          06320            JMP     SETDBL          ;SET IT UP
 4948  006017' 000000  005750'
 4949  006020' 000000  006014'
 4950                                   06340   ;
 4951                                   06360   ; THIS IS THE CASE WHERE THE STACK IS SINGLE PRECISION
 4952                                   06380   ; AND THE FAC IS EITHER SINGLE PRECISION OR INTEGER
 4953                                   06400   ;
 4954  006021' 001000  000315          06420   STKSNG:  CALL    FRCSNG          ;CONVERT THE FAC IF NECESSARY
 4955  006022' 000000  005554*
 4956  006023' 000000  006017'
 4957  006024' 001000  000301          06440            POPR                    ;PUT THE LEFT HAND OPERAND IN THE REGISTERS
 4958  006025' 001000  000321
 4959  006026' 001000  000041          06460   SNGDO:   LXI     H,SNGDSP        ;SETUP THE DISPATCH ADDRESS
 4960  006027' 000000  000704'
 4961  006030' 000000  006022'
 4962                                   06480                                    ;FOR THE SINGLE PRECISION OPERATOR ROUTINES
 4963  006031' 001000  000303          06500            JMP     DODSP           ;DISPATCH
 4964  006032' 000000  005756'
 4965  006033' 000000  006027'
 4966                                   06520   ;
 4967                                   06540   ; THIS IS THE CASE WHERE THE FAC IS SINGLE PRECISION AND THE STACK
 4968                                   06560   ; IS AN INTEGER.
 4969                                   06580   ;
 4970  006034' 001000  000341          06600   FACSNG:  POP     H               ;POP OFF THE INTEGER ON THE STACK
 4971  006035' 001000  000315          06620            CALL    PUSHF           ;SAVE THE FAC ON THE STACK
 4972  006036' 000000  005557*
 4973  006037' 000000  006032'
 4974  006040' 001000  000315          06640            CALL    CONSIH          ;CONVERT (H,L) TO A SINGLE PRECISION
 4975  006041' 000000  000000*
 4976  006042' 000000  006036'
 4977                                   06660                                    ;NUMBER IN THE FAC
 4978  006043' 001000  000315          06680            CALL    MOVRF           ;PUT THE LEFT HAND OPERAND IN THE REGISTERS
 4979  006044' 000000  003261*
 4980  006045' 000000  006041'
 4981  006046' 001000  000341          06700            POP     H               ;RESTORE THE FAC
 4982  006047' 001000  000042          06720            SHLD    FAC=1           ;FROM THE STACK
 4983  006050' 000000  001641'
 4984  006051' 000000  006044'
 4985  006052' 001000  000341          06740            POP     H
 4986  006053' 001000  000042          06760            SHLD    FACLO
 4987  006054' 000000  001637'
 4988  006055' 000000  006050'
 4989  006056' 001000  000303          06780            JMP     SNGDO>          ;PERFORM THE OPERATION
 4990  006057' 000000  006026'
 4991  006060' 000000  006054'
 4992
 4993  006061'                         06820   EVAL:
 4994                                   06840   IFN      LENGTH-2,<
 4995                                   06860   IFN      STRING,<
 4996                                   06880            XRA     A
 4997                                   06900            STA     VALTYP>>        ;ASSUME THE VALUE WILL BE NUMERIC
 4998  006061' 001000  000327          06920            CHRGET
 4999  006062' 001000  000332          06940            JC      FIN             ;IF NUMERIC, INTERPRET CONSTANT
```

```
006063' 000000  005072*
006064' 000000  006057'
006065' 001000  000315          06960             CALL    ISLET          ;VARIABLE NAME?
006066' 000000  003612'
006067' 000000  006063'
006070' 001000  000322          06980             JNC     ISVAR          ;AN ALPHABETIC CHARACTER MEANS YES
006071' 000000  006164'
006072' 000000  006066'
006073' 001000  000376          07000             CPI     PLUSTK         ;IGNORE "+"
006074' 000000  000250
006075' 001000  000312          07020             JZ      EVAL
006076' 000000  006061'
006077' 000000  006071'
006100' 001000  000376          07040             CPI     "."            ;"." AS LEADING CHARACTER OF A
006101' 000000  000056
                                07060                                    ;CONSTANT?
006102' 001000  000312          07080             JZ      FIN
006103' 000000  006063*
006104' 000000  006076'
006105' 001000  000376          07100             CPI     MINUTK         ;NEGATION?
006106' 000000  000251
006107' 001000  000312          07120             JZ      DOMIN
006110' 000000  006146'
006111' 000000  006103'
                                07140     IFN     STRING,<
006112' 001000  000376          07160             CPI     34             ;STRING CONSTANT?
006113' 000000  000042
006114' 001000  000312          07180             JZ      STRLTI>        ;IF SO BUILD A DESCRIPTOR IN A TEMPORARY
006115' 000000  007640'
006116' 000000  006110'
                                07200                                    ;DESCRIPTOR LOCATION AND PUT A POINTER TO THE
                                07220                                    ;DESCRIPTOR IN FACLO.
                                07240     IFN     LENGTH,<
006117' 001000  000376          07260             CPI     NOTTK          ;CHECK FOR "NOT" OPERATOR
006120' 000000  000246
006121' 001000  000312          07280             JZ      NOTER>
006122' 000000  006412'
006123' 000000  006115'
                                07300     IFN     FUNCTS,<
006124' 001000  000376          07320             CPI     FNTK           ;USER-DEFINED FUNCTION?
006125' 000000  000243
006126' 001000  000312          07340             JZ      FNDOER>
006127' 000000  007443'
006130' 000000  006122'
006131' 001000  000326          07360             SUI     ONEFUN         ;IS IT A FUNCTION NAME?
006132' 000000  000262
006133' 001000  000322          07380             JNC     ISFUN          ;FUNCTIONS ARE THE HIGHEST
006134' 000000  006204'
006135' 000000  006127'
                                07400                                    ;NUMBERED CHARACTERS ALLOWED
                                07420                                    ;SO THERE IS NO NEED TO CHECK
                                07440                                    ;THE UPPER BOUND
006136' 001000  000317          07460     PARCHK: SYNCHK  "("            ;ONLY POSSIBILITY LEFT
```



```
006137' 000000  000050
                                07480                                    ;IS A FORMULA IN PARENTHESES
006140' 001000  000315          07500             CALL    FRMEVL         ;RECURSIVELY EVALUATE THE FORMULA
006141' 000000  005336'
006142' 000000  006134'
006143' 001000  000317          07520             SYNCHK  ")"
006144' 000000  000051
006145' 001000  000311          07540             RET
006146'                         07560     DOMIN:
                                07580     IFE     EXTFNC,<
                                07600             CALL    EVAL>          ;NO ^ OPERATOR IN THIS CASE
                                07620     IFN     EXTFNC,<
006146' 001000  000026          07640             MVI     D,125          ;A PRECEDENCE BELOW ^
006147' 000000  000175
                                07660                                    ;BUT ABOVE ALL ELSE
006150' 001000  000315          07680             CALL    LPOPER         ;SO ^ GREATER THAN UNARY MINUS
006151' 000000  005341'
006152' 000000  006141'
006153' 001000  000052          07700             LHLD    TEMP2>         ;GET TEXT POINTER
006154' 000000  001605'
006155' 000000  006151'
006156' 001000  000345          07720             PUSH    H
                                07740     IFE     LENGTH-2,<
006157' 001000  000315          07760             CALL    VNEG>
006160' 000000  000000*
006161' 000000  006154'
                                07780     IFN     LENGTH-2,<
                                07800             CALL    NEG>           ;NEGATE THE FAC.
006162'                         07820     LABBCK:                        ;FUNCTIONS THAT DON'T RETURN
                                07840                                    ;STRING VALUES COME BACK HERE
                                07860     IFN     LENGTH-2,<
                                07880     IFN     STRING,<
                                07900             CALL    CHKNUM>>
006162' 001000  000341          07920             POP     H
006163' 001000  000311          07940             RET
006164' 001000  000315          07960     ISVAR:  CALL    PTRGET         ;GET A POINTER TO THE
006165' 000000  006505'
006166' 000000  006160'
                                07980                                    ;VARIABLE IN [D,E]
006167' 001000  000345          08000             PUSH    H              ;SAVE THE TEXT POINTER
                                08020     IFE     STRING,<
                                08040             XCHG                   ;TRANSFER THE POINTER AT THE VALUE
                                08060                                    ;INTO [H,L]
                                08080             CALL    MOVFM>         ;SETUP FAC WITH VARIABLE VALUE
                                08100     IFN     STRING,<
006170' 001000  000353          08120             XCHG                   ;PUT THE POINTER TO THE VARIABLE VALUE
                                08140                                    ;INTO [H,L]. IN THE CASE OF A STRING
                                08160                                    ;THIS IS A POINTER TO A DESCRIPTOR AND NOT
                                08180                                    ;AN ACTUAL VALUE
006171' 001000  000042          08200             SHLD    FACLO          ;IN CASE IT'S STRING STORE THE POINTER
006172' 000000  001637'
006173' 000000  006165'
                                08220                                    ;TO THE DESCRIPTOR IN FACLO.
```

```
                                    08240   IFE     LENGTH-2,<
 5107  006174' 001000  000315      08260           CALL    GETYPE          ;FOR STRINGS WE JUST LEAVE
 5108  006175' 000000  006307'
 5109  006176' 000000  006172'
 5110  006177' 001000  000304      08280           CNZ     VMOVFM>         ;A POINTER IN THE FAC
 5111  006200' 000000  000000*
 5112  006201' 000000  006175'
 5113                                08300   IFN     LENGTH-2,<
 5114                                08320           LDA     VALTYP
 5115                                08340           ORA     A
 5116                                08360           CZ      MOVFM>>         ;IF NOT,ACTUALLY TRANSFER THE VALUE INTO
 5117                                08380                                   ;THE FAC USING (H,L) AS THE POINTER,
 5118  006202' 001000  000341      08400           POP     H               ;RESTORE THE TEXT POINTER
 5119  006203' 001000  000311      08420           RET
 5120  006204' 001000  000006      08440   ISFUN:  MVI     B,0
 5121  006205' 000000  000000
 5122  006206' 001000  000007      08460           RLC                     ;MULTIPLY BY 2
 5123  006207' 001000  000117      08480           MOV     C,A
 5124  006210' 001000  000305      08500           PUSH    B               ;SAVE THE FUNCTION # ON THE STACK
 5125  006211' 001000  000327      08520           CHRGET
 5126                                08540   IFN     STRING,<
 5127  006212' 001000  000171      08560           MOV     A,C             ;LOOK AT FUNCTION #
 5128                  000051      08580   NUMGFN==2*LASNUM-2*ONEFUN+1
 5129  006213' 001000  000376      08600           CPI     NUMGFN          ;IS IT PAST LASNUM?
 5130  006214' 000000  000051
 5131  006215' 001000  000332      08620           JC      OKNORM          ;NO,MUST BE NORMAL FUNCTION
 5132  006216' 000000  006251'
 5133  006217' 000000  006200'
 5134                                08640   ;
 5135                                08660   ; MOST FUNCTIONS TAKE A SINGLE ARGUMENT,
 5136                                08680   ; THE RETURN ADDRESS OF THESE FUNCTIONS IS A SMALL ROUTINE
 5137                                08700   ; THAT CHECKS TO MAKE SURE VALTYP IS 0 (NUMERIC) AND POPS OFF
 5138                                08720   ; THE TEXT POINTER, SO NORMAL FUNCTIONS THAT RETURN STRING RESULTS (I,E, CHR$)
 5139                                08740   ; MUST POP OFF THE RETURN ADDRESS OF LABBCK, AND POP OFF THE
 5140                                08760   ; TEXT POINTER AND THEN RETURN TO FRMEVL.
 5141                                08780   ;
 5142                                08800   ; THE SO CALLED "FUNNY" FUNCTIONS CAN TAKE MORE THAN ONE ARGUMENT,
 5143                                08820   ; THE FIRST OF WHICH MUST BE STRING AND THE SECOND OF WHICH
 5144                                08840   ; MUST BE A NUMBER BETWEEN 0 AND 256, THE TEXT POINTER IS
 5145                                08860   ; PASSED TO THESE FUNCTIONS SO ADDITIONAL ARGUMENTS
 5146                                08880   ; CAN BE READ, THE TEXT POINTER IS PASSED IN (D,E),
 5147                                08900   ; THE CLOSE PARENTHESIS MUST BE CHECKED AND RETURN IS DIRECTLY
 5148                                08920   ; TO FRMEVL WITH (H,L) SETUP AS THE TEXT POINTER POINTING BEYOND THE ")",
 5149                                08940   ; THE POINTER TO THE DESCRIPTOR OF THE STRING ARGUMENT
 5150                                08960   ; IS STORED ON THE STACK UNDERNEATH THE VALUE OF THE INTEGER
 5151                                08980   ; ARGUMENT (2 BYTES)
 5152                                09000   ;
 5153  006220' 001000  000317      09020           SYNCHK  "("             ;FIRST ARGUMENT ALWAYS
 5154  006221' 000000  000050
 5155                                09040                                   ;STRING -- SECOND INTEGER
 5156  006222' 001000  000315      09060           CALL    FRMEVL          ;EAT OPEN PAREN AND FIRST ARG
 5157  006223' 000000  005336'
 5158  006224' 000000  006216'
```



```
 5159  006225' 001000  000317      09080           SYNCHK  44              ;TWO ARGS SO COMMA MUST DELIMIT
 5160  006226' 000000  000054
 5161  006227' 001000  000315      09100           CALL    CHKSTR          ;MAKE SURE THE FIRST ONE WAS STRING
 5162  006230' 000000  004164*
 5163  006231' 000000  006223'
 5164  006232' 001000  000353      09120           XCHG                    ;(D,E)=TXTPTR
 5165  006233' 001000  000052      09140           LHLD    FACLO           ;GET PTR AT STRING DESCRIPTOR
 5166  006234' 000000  001637'
 5167  006235' 000000  006230'
 5168  006236' 001000  000343      09160           XTHL                    ;GET FUNCTION #
 5169                                09180                                   ;SAVE THE STRING PTR
 5170  006237' 001000  000345      09200           PUSH    H               ;PUT THE FUNCTION # ON
 5171  006240' 001000  000353      09220           XCHG                    ;(H,L)=TXTPTR
 5172  006241' 001000  000315      09240           CALL    GETBYT          ;(E)=VALUE OF FORMULA
 5173  006242' 000000  011020'
 5174  006243' 000000  006234'
 5175  006244' 001000  000353      09260           XCHG                    ;TEXT POINTER INTO (D,E)
 5176                                09280                                   ;(H,L)=INT VALUE OF SECOND ARGUMENT
 5177  006245' 001000  000343      09300           XTHL                    ;SAVE INT VALUE OF SECOND ARG
 5178                                09320                                   ;(H,L)=FUNCTION NUMBER
 5179  006246' 001000  000303      09340           JMP     FINGO>          ;DISPATCH TO FUNCTION
 5180  006247' 000000  006276'
 5181  006250' 000000  006242'
 5182  006251' 001000  000315      09360   OKNORM: CALL    PARCHK          ;MAKE SURE ITS THERE,
 5183  006252' 000000  006136'
 5184  006253' 000000  006247'
 5185                                09380                                   ;READ THE FORMULA INSIDE
 5186                                09400                                   ;AND MAKE SURE ITS FOLLOWED BY ")"
 5187  006254' 001000  000343      09420           XTHL                    ;(H,L)=FUNCTION # AND SAVE TEXT POINTER
 5188                                09440   IFE     LENGTH-2,<
 5189                                09460   ;
 5190                                09480   ; CHECK IF SPECIAL COERCION MUST BE DONE FOR ONE OF THE TRANSCENDENTAL
 5191                                09500   ; FUNCTIONS (RND, SQR, COS, SIN, TAN, ATN, LOG, AND EXP)
 5192                                09520   ;
 5193  006255' 001000  000175      09540           MOV     A,L             ;(A)=FUNCTION NUMBER
 5194                  000016      09550   BOTCON==<SQRTK-ONEFUN>*2
 5195  006256' 001000  000376      09560           CPI     BOTCON          ;LESS THAN SQUARE ROOT?
 5196  006257' 000000  000016
 5197  006260' 001000  000332      09580           JC      NOTFRF          ;DON'T FORCE THE ARGUMENT
 5198  006261' 000000  006272'
 5199  006262' 000000  006252'
 5200                  000035      09590   TOPCON==<ATNTK-ONEFUN>*2+1
 5201  006263' 001000  000376      09600           CPI     TOPCON          ;BIGGER THAN ARC-TANGENT?
 5202  006264' 000000  000035
 5203  006265' 001000  000345      09620           PUSH    H               ;SAVE THE FUNCTION NUMBER
 5204  006266' 001000  000334      09640           CC      FRCSNG          ;IF NOT, FORCE FAC TO SINGLE-PRECISION
 5205  006267' 000000  006022*
 5206  006270' 000000  006261'
 5207  006271' 001000  000341      09660           POP     H               ;RESTORE THE FUNCTION NUMBER
 5208  006272'                     09680   NOTFRF: >
 5209  006272' 001000  000021      09700           LXI     D,LABBCK        ;RETURN ADDRESS
 5210  006273' 000000  006162'
 5211  006274' 000000  006267'
```

```
                                                    5212  006275' 001000  000325  09720          PUSH    D               ;MAKE THEM REALLY COME BACK
                                                    5213  006276' 001000  000001  09740  FINGO:  LXI     B,FUNDSP        ;FUNCTION DISPATCH TABLE
                                                    5214  006277' 000000  000103'
                                                    5215  006300' 000000  006275'
                                                    5216  006301' 001000  000011  09760          DAD     B               ;ADD ON THE OFFSET
                                                    5217                           09780  IFE     LENGTH,<
                                                    5218                           09800          PUSHM
                                                    5219                           09820          RET>            ;GO TO THE ADDRESS POINTED TO BY (H,L)
                                                    5220                           09840  IFN     LENGTH,<
                                                    5221  006302' 001000  000116  09860          MOV     C,M             ;FASTER THAN PUSHM
                                                    5222  006303' 001000  000043  09880          INX     H
                                                    5223  006304' 001000  000146  09900          MOV     M,M
                                                    5224  006305' 001000  000151  09920          MOV     L,C
                                                    5225  006306' 001000  000351  09940          PCHL>           ;GO PERFORM THE FUNCTION
                                                    5226                           09960  IFE     LENGTH=2,<
                                                    5227                           09980  ;
                                                    5228                           10000  ; GET THE VALTYP AND SET CONDITION CODES AS FOLLOWS:
                                                    5229                           10020  ; CONDITION CODE        TRUE SET        FALSE SET
                                                    5230                           10030  ;
                                                    5231                           10040  ; SIGN                  INT=2           STR,SNG,DBL
                                                    5232                           10060  ; ZERO                  STR=3           INT,SNG,DBL
                                                    5233                           10080  ; ODD PARITY            SNG=4           INT,STR,DBL
                                                    5234                           10100  ; NO CARRY              DBL=10          INT,STR,SNG
                                                    5235                           10120  ;
                                                    5236  006307' 001000  000072  10140  GETYPE: LDA     VALTYP          ;GET THE VALTYP
                                                    5237  006310' 000000  001543'
                                                    5238  006311' 000000  006277'
                                                    5239  006312' 001000  000376  10160          CPI     8               ;SET CARRY CORRECTLY
                                                    5240  006313' 000000  000010
                                                    5241  006314' 001000  000075  10180          DCR     A               ;SET THE OTHER CONDITION CODES CORRECTLY
                                                    5242  006315' 001000  000075  10200          DCR     A               ; WITHOUT AFFECTING CARRY
                                                    5243  006316' 001000  000075  10220          DCR     A
                                                    5244  006317' 001000  000311  10240          RET>
                                                    5245
                                                    5246                           10280  IFN     LENGTH=2,<
                                                    5247                           10300  IFN     LENGTH,<
                                                    5248                           10320  OR:     XWD     "Q1000,"Q366    ;"ORI" , FLAG AS "OR"
                                                    5249                           10340                                  ;AND USE COMMON "AND" CODE
                                                    5250                           10360  AND:    XRA     A               ;FLAG AS "AND"
                                                    5251                           10380  ANDCON:
                                                    5252                           10400          PUSH    PSW
                                                    5253                           10420  IFN     STRING,<CALL    CHKNUM>
                                                    5254                           10440          CALL    DEINT           ;GET (D,E)=INT VALUE AND CHECK SIZE
                                                    5255                           10460          POP     PSW
                                                    5256                           10480          XCHG                    ;(H,L)=INT VALUE
                                                    5257                           10500          POP     B               ;GET HIGH ORDER OFF
                                                    5258                           10520          XTHL                    ;PUT INT VALUE ON
                                                    5259                           10540                                  ;GET LOW OF SECOND ARG OFF
                                                    5260                           10560          XCHG                    ;(D,E)=LOW OF SECOND ARG
                                                    5261                           10580          CALL    MOVFR
                                                    5262                           10600          PUSH    PSW
                                                    5263                           10620          CALL    DEINT           ;GET (D,E)=INT VALUE
                                                    5264                           10640          POP     PSW
```



```
                                                    5265                           10660                                  ;OF FIRST ARG AND CHECK SIZE
                                                    5266                           10680          POP     B               ;(B,C)=INT VALUE OF SECOND ARG
                                                    5267                           10700          MOV     A,C             ;(A)=LOW OF SECOND ARG
                                                    5268                           10720          LXI     H,GIVACF        ;SETUP JUMP ADDRESS
                                                    5269                           10740          JNZ     ORFIN           ;IT WAS "OR" SO FINISH UP
                                                    5270                           10760          ANA     E               ;AND TWO LOW ORDERS
                                                    5271                           10780          MOV     C,A             ;SAVE ANSWER IN (C)
                                                    5272                           10800          MOV     A,B             ;(A)=HIGH ORDER SECOND ARG
                                                    5273                           10820          ANA     D               ;(A)=HIGH ORDER OF ANSWER
                                                    5274                           10840          PCHL                    ;FLOAT (A,C) AS ANSWER
                                                    5275                           10860  ORFIN:  ORA     E               ;OR TWO LOW ORDERS
                                                    5276                           10880          MOV     C,A             ;SAVE ANSWER LOW ORDER IN (C)
                                                    5277                           10900          MOV     A,B             ;(A)=HIGH ORDER SECOND ARG
                                                    5278                           10920          ORA     D               ;OR TWO HIGH ORDERS
                                                    5279                           10940                                  ;(A)=HIGH ORDER OF ANSWER
                                                    5280                           10960          PCHL                    ;FLOAT (A,C) AS ANSWER
                                                    5281                           10980  FINREL: LXI     H,PTOORL        ;MAKE (H,L) POINT AT OPERATOR ADDRESS
                                                    5282                           11000  IFN     STRING,<
                                                    5283                           11020          LDA     VALTYP          ;STORE VALUE TYPE AS LOW
                                                    5284                           11040          RAR                     ;ORDER BIT OF (E)
                                                    5285                           11060          MOV     A,D             ;GET RELATIONAL MEMORIES IN (A)
                                                    5286                           11080          RAL                     ;MOVE CARRY BIT IN
                                                    5287                           11100          MOV     E,A>            ;KEEP THIS BYTE IN (E)
                                                    5288                           11120  IFE     STRING,<MOV     E,D>
                                                    5289                           11140          MVI     D,100           ;PRECEDENCE OF ALL RELATIONAL
                                                    5290                           11160                                  ;OPERATORS IS 100
                                                    5291                           11180          MOV     A,B             ;(A)=PRECEDENCE OF OLD OPERATOR
                                                    5292                           11200          CMP     D               ;SEE WHETHER TO APPLY OLD OPERATOR
                                                    5293                           11220          RNC                     ;IF OLD OPERATOR HAS EQUAL OR GREATER
                                                    5294                           11240                                  ;PRECEDENCE THAN IT MUST BE APPLIED
                                                    5295                           11260          JMP     DOPREC          ;SEE IF TIME TO APPLY
                                                    5296                           11280                                  ;AND IF NOT SAVE INFO ON THE STACK
                                                    5297                           11300
                                                    5298                           11320  PTOORL: ADR(DOREL)              ;ADDRESS OF RELATIONAL
                                                    5299                           11340                                  ;OPERATOR APPLICATION
                                                    5300                           11360                                  ;ROUTINE
                                                    5301                           11380  ;
                                                    5302                           11400  ; TIME TO PERFORM A RELATIONAL OPERATOR
                                                    5303                           11420  ; (C) CONTAINS THE BITS AS TO WHICH RELATIONAL
                                                    5304                           11440  ; OPERATOR IT WAS (IF STRINGS ON
                                                    5305                           11460  ; LOW ORDER BIT SAYS WHETHER IT WAS STRING OR NOT)
                                                    5306                           11480  ;
                                                    5307                           11500  DOREL:  MOV     A,C             ;GET MEMORIES INTO (A)
                                                    5308                           11520  IFN     STRING,<
                                                    5309                           11540          ORA     A
                                                    5310                           11560          RAR>                    ;CARRY=WHETHER WAS STRING OR NOT
                                                    5311                           11580          POPR                    ;POP OFF LEFT RESULT
                                                    5312                           11600          PUSH    PSW             ;SAVE WHICH OPERATOR IT WAS
                                                    5313                           11620  IFE     STRING,<
                                                    5314                           11640          CALL    FCOMP>          ;DO A NUMERIC COMPARE
                                                    5315                           11660  IFN     STRING,<
                                                    5316                           11680          CALL    CHKVAL          ;SEE IF VALTYP MATCHES
                                                    5317                           11700                                  ;CARRY AND SET ZERO IN THE
```

```
                                                11720
                                                11740           LXI     H,DOCMP         ;COME BACK TO COMPARE
                                                11760           PUSH    H
                                                11780           JZ      FCOMP           ;DO NUMERIC COMPARE
                                                11800           XRA     A               ;SET VALUE TYPE AS NUMERIC
                                                11820           STA     VALTYP>>>
                                                11840   IFN     STRING,<
                                                11860   ;
                                                11880   ; THE FOLLOWING ROUTINE COMPARES TWO STRINGS
                                                11900   ; ONE WITH DESC IN [D,E] OTHER WITH DESC. IN [FACLO, FACLO+1]
                                                11920   ; A=0 IF STRINGS EQUAL
                                                11940   ; A=377 IF B,C,D,E POINTER FACLO
                                                11960   ; A=1 IF B,C,D,E .LT. FACLO
                                                11980   ;
                                                12000   IFN     LENGTH=2,<
                                                12020   STRCMP: PUSH    D               ;SAVE DESC, POINTER TO FIRST STR,
                                                12040           CALL    FREFAC          ;FREE THE FACLO STR
                                                12060           POP     D               ;RESTORE 1ST DESC, POINTER
                                                12080           PUSHM                   ;SAVE LENGTH
                                                12100           PUSHM                   ;SAVE POINTER
                                                12120           CALL    FRETMP          ;FREES 1ST DESC, POINTER
                                                12140           CALL    MOVRM           ;[B,C] POINT AT FIRST CHAR
                                                12160                                   ;[E] HAS THE LENGTH
                                                12180           POP     H               ;GET 2ND CHARACTER POINTER IN H
                                                12200           XTHL                    ;GET 2ND CHARACTER COUNTER IN L
                                                12220           MOV     D,L>            ;SAVE IN D
                                                12240   IFE     LENGTH=2,<
006320' 001000  000315          12260   STRCMP: CALL    FRESTR          ;FREE UP THE FAC STRING, AND GET THE
006321' 000000  010431'
006322' 000000  006310'
                                                12280                                   ;POINTER TO THE FAC DESCRIPTOR IN [H,L]
006323' 001000  000176          12300           MOV     A,M             ;SAVE THE LENGTH OF THE FAC STRING IN [A]
006324' 001000  000043          12320           INX     H
006325' 001000  000116          12340           MOV     C,M             ;SAVE THE POINTER AT THE FAC STRING
                                12342                                   ;DATA IN [B,C]
006326' 001000  000043          12360           INX     H
006327' 001000  000106          12380           MOV     B,M
006330' 001000  000321          12400           POP     D               ;GET THE STACK STRING POINTER
006331' 001000  000305          12420           PUSH    B               ;SAVE THE POINTER AT THE FAC STRING DATA
006332' 001000  000365          12440           PUSH    PSW             ;SAVE THE FAC STRING LENGTH
006333' 001000  000315          12460           CALL    FRETMP          ;FREE UP THE STACK STRING AND RETURN
006334' 000000  010440'
006335' 000000  006321'
                                12480                                   ;THE POINTER TO THE STACK STRING DESCRIPTOR
                                12500                                   ;IN [H,L]
006336' 001000  000321          12520           POP     D               ;[D]=LENGTH OF FAC STRING
006337' 001000  000136          12540           MOV     E,M             ;[E]=LENGTH OF STACK STRING
006340' 001000  000043          12560           INX     H
006341' 001000  000116          12580           MOV     C,M             ;[B,C]=POINTER AT STACK STRING
006342' 001000  000043          12600           INX     H
006343' 001000  000106          12620           MOV     B,M>
006344' 001000  000341          12640           POP     H               ;GET BACK 2ND CHARACTER POINTER
006345' 001000  000173          12660   CSLOOP: MOV     A,E             ;BOTH STRINGS ENDED
```



```
006346' 001000  000262          12680           ORA     D               ;TEST BY OR'ING THE LENGTHS TOGETHER
006347' 001000  000310          12700           RZ                      ;IF SO, RETURN WITH A ZERO
006350' 001000  000172          12720           MOV     A,D             ;GET FACLO STRING LENGTH
006351' 001000  000267          12740           ORA     A               ;IF IT ENDED, OTHER MUST NOT HAVE
006352' 001000  000057          12760           CMA                     ;MAKE =1
006353' 001000  000310          12780           RZ                      ;TEST
006354' 001000  000257          12800           XRA     A               ;MUST NOT HAVE BEEN ZERO, TEST CASE
006355' 001000  000273          12820           CMP     E               ;OF B,C,D,E STRING HAVING ENDED FIRST
006356' 001000  000074          12840           INR     A               ;RETURN WITH A=1
006357' 001000  000320          12860           RNC                     ;TEST THE CONDITION
                                12880   ;HERE WHEN NEITHER STRING ENDED
006360' 001000  000025          12900           DCR     D               ;DECREMENT BOTH CHARACTER COUNTS
006361' 001000  000035          12920           DCR     E
006362' 001000  000012          12940           LDAX    B               ;GET CHARACTER FROM B,C,D,E STRING
006363' 001000  000276          12960           CMP     M               ;COMPARE WITH FACLO STRING
006364' 001000  000043          12980           INX     H               ;BUMP POINTERS (INX DOESNT CLOBBER CC'S)
006365' 001000  000023          13000           INX     B
006366' 001000  000312          13020           JZ      CSLOOP          ;IF BOTH THE SAME, MUST BE MORE TO STRINGS
006367' 000000  006345'
006370' 000000  006334'
006371' 001000  000077          13040           CMC                     ;HERE WHEN STRINGS DIFFER
006372' 001000  000303          13060           JMP     SIGNS>          ;SET [A] ACCORDING TO CARRY
006373' 000000  000000*
006374' 000000  006367'
                                13080   IFN     LENGTH,<
006375' 001000  000074          13100   DOCMP:  INR     A               ;SETUP BITS
006376' 001000  000217          13120           ADC     A               ;A=LESS 2=EQUAL 1=GREATER
006377' 001000  000301          13140           POP     B               ;WHAT DID HE WANT?
006400' 001000  000240          13160           ANA     B               ;ANY BITS MATCH?
006401' 001000  000306          13180           ADI     255             ;MAP 0 TO 0
006402' 000000  000377
006403' 001000  000237          13200           SBB     A               ;AND ALL OTHERS TO 377
                                13220   IFE     LENGTH=2,<
006404' 001000  000315          13240           CALL    CONIA##         ;CONVERT [A] TO AN INTEGER SIGNED
006405' 000000  000000*
006406' 000000  006373'
006407' 001000  000303          13260           JMP     RETAOP>         ;RETURN FROM OPERATOR APPLICATION
006410' 000000  005554'
006411' 000000  006405'
                                13280   IFN     LENGTH=2,<
                                13300           JMP     FLOAT           ;MAKE FAC=[A] SIGNED
                                13320                                   ;COULD FALL INTO FLOAT BUT MESSY (SAVES
                                13340                                   ;TWO BYTES)
                                13360
                                13380   NOTER:  MVI     D,90            ;"NOT" HAS PRECEDENCE 90
                                13400           CALL    LPOPER          ;GO PERFORM
                                13420   IFN     STRING,<CALL    CHKNUM> ;MAKE SURE ITS INT
                                13440           CALL    DEINT           ;GET VALUE IN [D,E]
                                13460           MOV     A,E
                                13480           CMA                     ;COMPLEMENT
                                13500           MOV     C,A             ;[C] LOW ORDER OF ANSWER
                                13520           MOV     A,D
                                13540           CMA                     ;COMPLEMENT HIGH ORDER TOO
```

```
 5424                                   13560           CALL    GIVACF          ;FLOAT [A,C] AS RESULT
 5425                                   13580           POP     B               ;TAKE RETURN ADDRESS OF FRMEVL
 5426                                   13600           JMP     RETAOP>>        ;OFF AND RETURN TO THE RIGHT
 5427                                   13620                                   ;PLACE SO THE TEXT POINTER
 5428                                   13640                                   ;WILL GET SET UP TO WHAT IT WAS
 5429                                   13660                                   ;WHEN LPOPER RETURNED.
 5430                                   13680   IFE     LENGTH-2,<
 5431  006412' 001000  000026           13700   NOTER:  MVI     D,90            ;"NOT" HAS PRECEDENCE 90, SO
 5432  006413' 000000  000132
 5433  006414' 001000  000315           13720           CALL    LPOPER          ;FORMULA EVALUATION IS ENTERED WITH A DUMMY
 5434  006415' 000000  005341'
 5435  006416' 000000  006410'
 5436                                   13730                                   ;ENTRY OF 90 ON THE STACK
 5437  006417' 001000  000315           13740           CALL    FRCINT          ;COERCE THE ARGUMENT TO INTEGER
 5438  006420' 000000  005573*
 5439  006421' 000000  006415'
 5440  006422' 001000  000175           13760           MOV     A,L             ;COMPLEMENT [H,L]
 5441  006423' 001000  000057           13780           CMA
 5442  006424' 001000  000157           13800           MOV     L,A
 5443  006425' 001000  000174           13820           MOV     A,H
 5444  006426' 001000  000057           13840           CMA
 5445  006427' 001000  000147           13860           MOV     H,A
 5446  006430' 001000  000042           13880           SHLD    FACLO           ;UPDATE THE FAC
 5447  006431' 000000  001637'
 5448  006432' 000000  006420'
 5449  006433' 001000  000301           13900           POP     B               ;FRMEVL, AFTER SEEING THE PRECEDENCE
 5450                                   13910                                   ;OF 90 THINKS IT IS APPLYING AN OPERATOR
 5451                                   13915                                   ;SO IT HAS THE TEXT POINTER IN TEMP2 SO
 5452
 5453  006434' 001000  000303           13920           JMP     RETAOP          ;RETURN TO REFETCH IT
 5454  006435' 000000  005354'
 5455  006436' 000000  006431'
 5456                                   14042   ;
 5457                                   14044   ; DANDOR APPLIES THE "AND" AND "OR" OPERATORS
 5458                                   14046   ; AND SHOULD BE USED TO IMPLEMENT ALL LOGICAL OPERATORS,
 5459                                   14048   ; WHENEVER AN OPERATOR IS APPLIED, ITS PRECEDENCE IS IN [B],
 5460                                   14050   ; THIS FACT IS USED TO DISTINGUISH BETWEEN "AND" AND "OR".
 5461                                   14052   ; THE RIGHT HAND ARGUMENT IS COERCED TO INTEGER, JUST AS
 5462                                   14054   ; THE LEFT HAND ONE WAS WHEN IT WAS PUSHED ON THE STACK.
 5463                                   14056   ;
 5464  006437' 001000  000305           14060   DANDOR: PUSH    B               ;SAVE THE PRECEDENCE "OR"=70
 5465  006440' 001000  000315           14080           CALL    FRCINT          ;COERCE RIGHT HAND ARGUMENT TO INTEGER
 5466  006441' 000000  006420*
 5467  006442' 000000  006435'
 5468  006443' 001000  000361           14100           POP     PSW             ;GET BACK THE PRECEDENCE TO DISTINGUISH
 5469                                   14120                                   ;"AND" AND "OR"
 5470  006444' 001000  000321           14140           POP     D               ;POP OFF THE LEFT HAND ARGUMENT
 5471  006445' 001000  000376           14160           CPI     70              ;SET ZERO FOR "OR"
 5472  006446' 000000  000106
 5473  006447' 001000  000173           14180           MOV     A,E             ;SETUP LOW IN [A]
 5474  006450' 001000  000312           14200           JZ      ORFIN           ;DO "OR" IF PRECEDENCE WAS 70
 5475  006451' 000000  006465'
 5476  006452' 000000  006441'
```



```
 5477  006453' 001000  000245           14220           ANA     L
 5478  006454' 001000  000157           14240           MOV     L,A
 5479  006455' 001000  000174           14260           MOV     A,H
 5480  006456' 001000  000242           14280           ANA     D
 5481  006457' 001000  000147           14300           MOV     H,A
 5482  006460' 001000  000303           14320           JMP     MAKINT          ;RETURN THE INTEGER [H,L]
 5483  006461' 000000  000000*
 5484  006462' 000000  006451'
 5485                                   14322                                   ; AS THE "AND"ED RESULT
 5486  006463' 001000  000265           14324   ORFIN:  ORA     L
 5487  006464' 001000  000157           14326           MOV     L,A
 5488  006465' 001000  000174           14328           MOV     A,H
 5489  006466' 001000  000262           14330           ORA     D
 5490  006467' 001000  000147           14332           MOV     H,A
 5491  006470' 001000  000303           14334           JMP     MAKINT>         ;RETURN THE INTEGER [H,L]
 5492  006471' 000000  006461*
 5493  006472' 000000  006461'
 5494                                   14336                                   ; AS THE "OR"ED RESULT
 5495                                   14340   PAGE
```

```
                                14360   SUBTTL  DIMENSION & VARIABLE SEARCHING
 5497  006473' 001000  000053   14380   DIMCON: DCX     H               ;SEE IF COMMA ENDED THIS VARIABLE
 5498  006474' 001000  000327   14400           CHRGET
 5499  006475' 001000  000310   14420           RZ                      ;IF TERMINATOR, GOOD BYE
 5500  006476' 001000  000317   14440           SYNCHK  44              ;MUST BE COMMA
 5501  006477' 000000  000054
 5502                           14442   ;
 5503                           14444   ; THE "DIM" CODE SETS DIMFLG AND THEN FALLS INTO THE VARIABLE
 5504                           14446   ; SEARCH ROUTINE. THE VARIABLE SEARCH ROUTINE LOOKS AT
 5505                           14448   ; DIMFLG AT THREE DIFFERENT POINTS:
 5506                           14450   ;
 5507                           14452   ;       1) IF AN ENTRY IS FOUND, DIMFLG BEING ON INDICATES
 5508                           14454   ;               A "DOUBLY DIMENSIONED" VARIABLE
 5509                           14456   ;       2) WHEN A NEW ENTRY IS BEING BUILT DIMFLG'S BEING ON
 5510                           14458   ;               INDICATES THE INDICES SHOULD BE USED FOR
 5511                           14460   ;               THE SIZE OF EACH INDICE, OTHERWISE THE DEFAULT
 5512                           14462   ;               OF TEN IS USED.
 5513                           14464   ;       3) WHEN THE BUILD ENTRY CODE FINISHES, ONLY IF DIMFLG IS
 5514                           14466   ;               OFF WILL INDEXING BE DONE
 5515                           14468   ;
 5516  006500' 001000  000001   14470   DIM:    LXI     B,DIMCON        ;PLACE TO COME BACK TO
 5517  006501' 000000  006473'
 5518  006502' 000000  006471'
 5519  006503' 001000  000305   14480           PUSH    B
 5520  006504' 001000  000366   14490           XWD     "01000,"0366    ;"ORI" NON ZERO THING
 5521                           14500   ;
 5522                           14540   ; ROUTINE TO READ THE VARIABLE NAME AT THE CURRENT TEXT POSITION
 5523                           14560   ; AND PUT A POINTER TO ITS VALUE IN [D,E]. [H,L] IS UPDATED
 5524                           14580   ; TO POINT TO THE CHARACTER AFTER THE VARIABLE NAME.
 5525                           14600   ; VALTYP IS SETUP, NOTE THAT EVALUATING SUBSCRIPTS IN
 5526                           14620   ; A VARIABLE NAME CAN CAUSE RECURSIVE CALLS TO PTRGET SO AT
 5527                           14640   ; THAT POINT ALL VALUES MUST BE STORED ON THE STACK.
 5528                           14660   ;
 5529  006505' 001000  000257   14680   PTRGET: XRA     A               ;MAKE [A]=0
 5530  006506' 001000  000062   14700           STA     DIMFLG          ;FLAG IT AS SUCH
 5531  006507' 000000  001542'
 5532  006510' 000000  006501'
 5533  006511' 001000  000106   14720           MOV     B,M             ;GET FIRST CHARACTER IN [B]
 5534  006512' 001000  000315   14740   PTRGT2: CALL    ISLET           ;CHECK FOR LETTER
 5535  006513' 000000  003612'
 5536  006514' 000000  006507'
 5537  006515' 001000  000332   14760           JC      SNERR           ;MUST HAVE A LETTER
 5538  006516' 000000  002672'
 5539  006517' 000000  006513'
 5540  006520' 001000  000257   14780           XRA     A
 5541  006521' 001000  000117   14800           MOV     C,A             ;ASSUME NO SECOND CHARACTER
 5542                           14820   IFN     LENGTH=2,<
 5543                           14840   IFN     STRING,<
 5544                           14860           STA     VALTYP>>        ;DEFAULT IS ZERO (NUMERIC)
 5545  006522' 001000  000327   14880           CHRGET                  ;GET THE FOLLOWING CHARACTER
 5546                           14900   IFE     LENGTH,<
 5547                           14920           JNC     NOSEC>          ;ONLY NUMBERS ALLOWED
 5548                           14940   IFN     LENGTH,<
```



```
 5549  006523' 001000  000332   14960           JC      ISSEC           ;CARRY SET BY CHRGET IF CHARACTER IS
 5550  006524' 000000  006534'
 5551  006525' 000000  006516'
 5552                           14980                                   ;NUMERIC
 5553  006526' 001000  000315   15000           CALL    ISLET           ;SET CARRY IF NOT ALPHABETIC
 5554  006527' 000000  003612'
 5555  006530' 000000  006524'
 5556  006531' 001000  000332   15020           JC      NOSEC>          ;ALLOW ALPHABETICS
 5557  006532' 000000  006547'
 5558  006533' 000000  006527'
 5559  006534' 001000  000117   15040   ISSEC:  MOV     C,A             ;IT IS A NUMBER--SAVE IN C
 5560  006535' 001000  000327   15060   EATEM:  CHRGET                  ;LOOK AT NEXT CHARACTER
 5561                           15080   IFN LENGTH,<
 5562  006536' 001000  000332   15100           JC      EATEM           ;SKIP NUMERICS
 5563  006537' 000000  006535'
 5564  006540' 000000  006536'
 5565  006541' 001000  000315   15120           CALL    ISLET
 5566  006542' 000000  003612'
 5567  006543' 000000  006537'
 5568  006544' 001000  000322   15140           JNC     EATEM>          ;SKIP ALPHABETICS
 5569  006545' 000000  006535'
 5570  006546' 000000  006542'
 5571  006547'                  15160   NOSEC:
 5572                           15180   IFE     LENGTH-2,<
 5573  006547' 001000  000021   15200           LXI     D,MAVTYP        ;SAVE JUMPS BY USING RETURN ADDRESS
 5574  006550' 000000  006574'
 5575  006551' 000000  006545'
 5576  006552' 001000  000325   15220           PUSH    D
 5577  006553' 001000  000026   15240           MVI     D,8             ;ASSUME ITS DOUBLE PRECISION
 5578  006554' 000000  000010
 5579  006555' 001000  000376   15260           CPI     "#"             ;CHECK THE CHARACTER
 5580  006556' 000000  000043
 5581  006557' 001000  000310   15280           RZ                      ;WHEN WE MATCH, SETUP VALTYP
 5582  006560' 001000  000026   15300           MVI     D,2             ;CHECK FOR INTEGER
 5583  006561' 000000  000002
 5584  006562' 001000  000376   15320           CPI     "%"
 5585  006563' 000000  000045
 5586  006564' 001000  000310   15340           RZ
 5587  006565' 001000  000024   15360           INR     D               ;CHECK FOR STRING
 5588  006566' 001000  000376   15380           CPI     "$"
 5589  006567' 000000  000044
 5590  006570' 001000  000310   15400           RZ
 5591  006571' 001000  000024   15420           INR     D               ;SINGLE PRECISION IS THE DEFAULT
 5592  006572' 001000  000053   15440           DCX     H               ;NO MARKING CHARACTER
 5593  006573' 001000  000311   15460           RET                     ;GET RID OF RETURN ADDRESS
 5594  006574' 001000  000172   15480   MAVTYP: MOV     A,D             ;SETUP VALTYP
 5595  006575' 001000  000062   15500           STA     VALTYP
 5596  006576' 000000  001543'
 5597  006577' 000000  006550'
 5598  006600' 001000  000327   15520           CHRGET>                 ;READ PAST TYPE MARKER
 5599                           15540   IFN     LENGTH-2,<
 5600                           15560   IFN     STRING,<
 5601                           15580           SUI     "$"             ;IS IT A STRING?
```

```
                                15600           JNZ     NOTSTR          ;IF NOT VALTYP ALREADY=0
                                15620           INR     A               ;[A]=1
                                15640           STA     VALTYP          ;FLAG THIS AS A STRING
                                15660           RRC                     ;MAKE [A]=128
                                15680           ADD     C               ;MAKE [A]=SECOND CHARACTER
                                15700           MOV     C,A             ;BACK INTO [C] WITH STRING BIT ON
                                15720           CHRGET                  ;GET CHARACTER AFTER "$"
                                15740   NOTSTR: >>
                                15760   IFN     LENGTH,<
006621' 001000  000072          15780           LDA     SUBFLG          ;GET FLAG WHETHER TO ALLOW ARRAYS
006622' 000000  001601'
006623' 000000  006576'
006624' 001000  000028          15800           ADD     M>              ;ADD ONTO CHARACTER
                                15820           CPI     "("             ;ARRAY PERHAPS (IF SUBFLG SET NEVER WILL MATCH)
006625' 001000  000376
006626' 000000  000050
006627' 001000  000312          15840           JZ      ISARY           ;IT IS!
006630' 000000  006745'
006631' 000000  006602'
                                15860   IFN     LENGTH,<
006632' 001000  000257          15880           XRA     A               ;ALLOW PARENS AGAIN
006633' 001000  000062          15900           STA     SUBFLG>         ;SAVE IN FLAG LOCATION
006634' 000000  001601'
006635' 000000  006610'
006636' 001000  000345          15920           PUSH    H               ;SAVE THE TEXT POINTER
                                15940   IFE     LENGTH-2,<
006637' 001000  000172          15960           MOV     A,D             ;VALUE TYPE INTO [A]
006638' 001000  000052          15980           LHLD    VARTAB          ;[H,L]=PLACE TO START THE SEARCH
006639' 000000  001621'
006640' 000000  006614'
006641' 001000  000365          16000   LOPFND: PUSH    PSW             ;SAVE THE VALUE TYPE
006642' 001000  000353          16020           XCHG                    ;[D,E]=POINTER INTO SIMPLE VARIABLES
006643' 001000  000052          16040           LHLD    ARYTAB          ;[H,L]=END OF SIMPLE VARIABLES
006644' 000000  001623'
006645' 000000  006621'
006646' 001000  000347          16060           COMPAR                  ;SEE IF THE END HAS BEEN REACHED
006647' 001000  000341          16080           POP     H               ;[H]=VALTYP
006650' 001000  000312          16100           JZ      NOTFNS          ;COULDN'T FIND IT, SO MAKE A NEW ENTRY
006651' 000000  006671'
006652' 000000  006626'
006653' 001000  000032          16120           LDAX    D               ;GET THE VALTYP OF THIS SIMPLE VARIABLE
006654' 001000  000157          16140           MOV     L,A             ;SAVE SO WE KNOW HOW MUCH TO SKIP
006655' 001000  000274          16160           CMP     H               ;COMPARE WITH OUR VALTYP
006656' 001000  000023          16180           INX     D
006657' 001000  000302          16200           JNZ     NOTIT1          ;NOT RIGHT KIND -- SKIP IT
006660' 000000  006660'
006661' 001000  000032          16220           LDAX    D               ;[A]=FIRST CHARACTER OF THIS VARIABLE
006662' 001000  000271          16240           CMP     C               ;SEE IF OUR VARIABLE MATCHES
006663' 001000  000302          16260           JNZ     NOTIT1
006664' 000000  006642'
006665' 000000  006642'
006666' 001000  000023          16280           INX     D
```



```
006667' 001000  000032          16300           LDAX    D               ;SEE IF SECOND CHACRACTER MATCHES
006670' 001000  000270          16320           CMP     B
006671' 001000  000312          16340           JZ      FINPTR          ;THAT WAS IT, ALL DONE
006672' 000000  006742'
006673' 000000  006647'
006674' 001000  000076          16360           XWD     "\01000,"\076   ;"MVI A," AROUND THIS INX SINCE THE POINTER
                                16380                                   ;IS ALREADY INCREMENTED
006675' 001000  000023          16400   NOTIT1: INX     D
006676' 001000  000023          16420           INX     D
006677' 001000  000174          16440           MOV     A,M             ;SKIP OVER THE
                                16460                                   ;CURRENT VARIABLE SINCE WE DIDN'T MATCH
                                16480                                   ;SAVE THE VALTYP IN [A]
006700' 001000  000046          16500           MVI     M,0             ;[M,L]=NUMBER OF BYTES TO SKIP
006701' 000000  000000
006702' 001000  000031          16520           DAD     D               ;ADD ON THE POINTER
006703' 001000  000303          16540           JMP     LOPFND          ;AND SEARCH SOME MORE
006704' 000000  006655'
006705' 001000  000305          16560   NOTFNS: PUSH    B               ;SAVE THE LOOKS
006706' 001000  000114          16580           MOV     C,M             ;[B,C]=LENGTH OF THIS VARIABLE
006707' 001000  000107          16600           MOV     B,A             ;[B]=0
006710' 001000  000305          16620           PUSH    B               ;SAVE THE VALTYP ON THE STACK
006711' 001000  000003          16640           INX     B               ;MAKE THE LENGTH INCLUDE
                                16660                                   ;THE LOOKS TOO
006712' 001000  000003          16680           INX     B
006713' 001000  000003          16700           INX     B>
                                16720   IFN     LENGTH-2,<
                                16730           LHLD    ARYTAB          ;PLACE TO STOP SEARCHING
                                16740           XCHG
                                16760           LHLD    VARTAB          ;GET THE PLACE TO START
                                16780   LOPFND: COMPAR                  ;SEE IF WE ARE THERE
                                16800           JZ      NOTFNS          ;COULDN'T FIND THIS VARIABLE
                                16820                                   ;SO MAKE ROOM FOR IT
                                16840           MOV     A,C
                                16860           SUB     M               ;IS THIS VARIABLE THE ONE?
                                16880           INX     H
                                16900           JNZ     NOTIT           ;NOPE
                                16920           MOV     A,B
                                16940           SUB     M               ;TRY SECOND CHARACTER MATCHING
                                16960   NOTIT:  INX     H
                                16980           JZ      FINPTR          ;THAT WAS IT!
                                17000           INX     H               ;SKIP OVER THAT ONE--NOT IT
                                17020           INX     H
                                17040           INX     H
                                17060           INX     H
                                17080           JMP     LOPFND          ;TRY AGAIN
                                17100   NOTFNS: PUSH    B               ;REMEMBER WHAT THIS
                                17120                                   ;VARIABLE LOOKS LIKE
                                17140           LXI     B,6+$CODE>      ;THE AMOUNT TO SHOVE
                                17160                                   ;EVERYTHING UP BY
006714' 001000  000052          17180           LHLD    STREND          ;THE CURRENT END OF STORAGE
006715' 000000  001625'
006716' 000000  006677'
```

```
 5708  006703' 001000  000345          17200          PUSH    H               ;SAVE THIS #
 5709  006704' 001000  000011          17220          DAD     B               ;ADD ON THE AMOUNT OF SPACE
 5710                                  17240                                  ;EXTRA NOW BEING USED
 5711  006705' 001000  000301          17260          POP     B               ;POP OFF HIGH ADDRESS TO MOVE
 5712  006706' 001000  000345          17280          PUSH    H               ;SAVE NEW CANDIDATE FOR STREND
 5713  006707' 001000  000315          17300          CALL    BLTU            ;BLOCK TRANSFER AND MAKE SURE
 5714  006710' 000000  002005'
 5715  006711' 000000  006701'
 5716                                  17320                                  ;WE ARE NOT OVERFLOWING THE
 5717                                  17340                                  ;STACK SPACE
 5718  006712' 001000  000341          17360          POP     H               ;[H,L]=NEW STREND
 5719  006713' 001000  00004E          17380          SHLD    STREND          ;STORE SINCE WAS OK
 5720  006714' 000000  001625'
 5721  006715' 000000  006710'
 5722                                  17400                                  ;THERE WAS ROOM, AND BLOCK TRANSFER
 5723                                  17420                                  ;WAS DONE, SO UPDATE POINTERS
 5724  006716' 001000  000140          17440          MOV     H,B             ;GET BACK [H,L] POINTING AT THE END
 5725  006717' 001000  000151          17460          MOV     L,C             ;OF THE NEW VARIABLE
 5726  006720' 001000  00004E          17480          SHLD    ARYTAB          ;UPDATE THE ARRAY TABLE POINTER
 5727  006721' 000000  001623'
 5728  006722' 000000  006714'
 5729  006723' 001000  000053          17500  ZEROEK: DCX     H               ;[H,L] IS RETURNED POINTING TO THE
 5730  006724' 001000  000066          17520          MVI     M,0             ;END OF THE VARIABLE SO WE
 5731  006725' 000000  000000
 5732  006726' 001000  000347          17540          COMPAR                  ;ZERO BACKWARDS TO [D,E] WHICH
 5733  006727' 001000  000302          17560          JNZ     ZEROEK          ;POINTS TO THE START OF THE VARIABLE
 5734  006730' 000000  006723'
 5735  006731' 000000  006721'
 5736                                  17580  IFE     LENGTH-2,<
 5737  006732' 001000  000321          17600          POP     D               ;[E]=VALTYP
 5738  006733' 001000  000163          17620          MOV     M,E             ;STORE AS PART OF THE LOOKS
 5739  006734' 001000  000043          17640          INX     H>
 5740  006735' 001000  000321          17660          POP     D
 5741  006736' 001000  000163          17680          MOV     M,E             ;PUT DESCRIPTION
 5742  006737' 001000  000043          17700          INX     H
 5743  006740' 001000  000162          17720          MOV     M,D             ;OF THIS VARIABLE
 5744                                  17740                                  ;INTO MEMORY
 5745                                  17760  IFE     LENGTH-2,<
 5746  006741' 001000  000353          17780          XCHG                    ;POINTER AT VARIABLE INTO [D,E]
 5747  006742' 001000  000023          17800  FINPTR: INX     D>              ;POINT AT THE VALUE
 5748                                  17820  IFN     LENGTH-2,<
 5749                                  17840          INX     H
 5750                                  17860  FINPTR: XCHG>                   ;VARIABLE POINTER INTO [D,E]
 5751  006743' 001000  000341          17880          POP     H               ;RESTORE THE TEXT POINTER
 5752  006744' 001000  000311          17900          RET
 5753
 5754                                  17940  IFE     MULDIM,<
 5755                                  17960  ISARY:  PUSH    B               ;REMEMBER WHAT VARIABLE LOOKS
 5756                                  17980                                  ;LIKE
 5757                                  18000  IFN     STRING,<
 5758                                  18020          PUSH    H               ;SAVE THE TXTPTR
 5759                                  18040          LHLD    DIMFLG          ;[L]=DIMFLG [H]=VALTYP
 5760                                  18060          XTHL>                   ;PUT VALTYP AND DIMFLG ON THE STACK
```



```
 5761                                  18080                                  ;AND RESTORE THE TEXT POINTER
 5762                                  18100  IFE     STRING,<
 5763                                  18120          LDA     DIMFLG          ;SINCE THIS CALL IS RECURSIVE
 5764                                  18140          PUSH    PSW>            ;DIMFLG MUST BE SAVED ON THE STACK
 5765                                  18160          CALL    INTIDX          ;EVALUATE THE INDEX INTO [D,E]
 5766                                  18180          SYNCHK  ")"             ;MAKE SURE HE CLOSED IT
 5767                                  18200  IFN     STRING,<
 5768                                  18220          XTHL                    ;[L]=DIMFLG [H]=VALTYP
 5769                                  18240                                  ;TEXT POINTER ONTO THE STACK
 5770                                  18260          SHLD    DIMFLG          ;SAVE BOTH VALUES BACK
 5771                                  18280          POP     H>              ;RESTORE THE TEXT POINTER
 5772                                  18300  IFE     STRING,<
 5773                                  18320          POP     PSW             ;GET DIMFLG OFF THE STACK
 5774                                  18340          STA     DIMFLG>         ;RESTORE IT
 5775                                  18360          XTHL                    ;[H,L] GET VARIABLE DESCRIPTOR
 5776                                  18380                                  ;TEXT POINTER IS PUT ONTO
 5777                                  18400                                  ;THE STACK
 5778                                  18420          XCHG                    ;[D,E]=DESCRIPTOR
 5779                                  18440                                  ;[H,L]=INDEX
 5780                                  18460          DAD     H               ;MULTIPLY BY 4 TO GET
 5781                                  18480          DAD     H               ;BYTE OFFSET
 5782                                  18500          PUSH    H               ;SAVE THE INDEX
 5783                                  18520          LHLD    ARYTAB          ;PLACE TO START SEARCH
 5784                                  18540          XWD     "D1000,1        ;"LXI B," OVER THE NEXT 2
 5785                                  18560  LOPFD2: POP     B               ;[B,C]=LENGTH OF LAST VARIABLE
 5786                                  18580          DAD     B               ;SKIP OVER LAST VARIABLE BY ADDING ITS
 5787                                  18600                                  ;LENGTH ONTO [H,L]
 5788                                  18620          XCHG                    ;[D,E] GET CURRENT SEARCH POINT
 5789                                  18640          PUSH    H               ;SAVE THE VARIABLE LOOK
 5790                                  18660          LHLD    STREND          ;GET PLACE TO STOP
 5791                                  18680          COMPAR                  ;SEE IF WE ARE THERE
 5792                                  18700          XCHG                    ;[H,L] GETS SEARCH POINT
 5793                                  18720          POP     D               ;POP OFF VARIABLE LOOKS
 5794                                  18740          JZ      NOTFDD          ;COULDN'T FIND IT
 5795                                  18760          PUSHM                   ;PUT ON LOOKS OF VARIABLE
 5796                                  18780                                  ;WE ARE EXAMINING
 5797                                  18800          XTHL                    ;PUT [H,L] ON THE STACK AND
 5798                                  18820                                  ;LOOKS OF VARIABLE WE ARE
 5799                                  18840                                  ;EXAMINING INTO [H,L]
 5800                                  18860          COMPAR                  ;IS THIS THE VARIABLE
 5801                                  18880          POP     H               ;POP OFF SEARCH POINTER
 5802                                  18900          PUSHM                   ;PUSH LENGTH OF VARIABLE
 5803                                  18920                                  ;BEING EXAMINED ONTO THE STACK
 5804                                  18940          JNZ     LOPFD2          ;IF NO MATCH,GO LOOK SOMEMORE
 5805                                  18960          LDA     DIMFLG          ;IS THIS VARIABLE TRYING TO BE
 5806                                  18980                                  ;DIMENSIONED AND ALREADY
 5807                                  19000          ORA     A               ;EXISTS?
 5808                                  19020          MVI     E,ERRDD         ;THATS ERROR ERRDD
 5809                                  19040          JNZ     ERROR
 5810                                  19060  MAKOFN: POP     D               ;POP OFF LENGTH OF THIS VARIABLE
 5811                                  19080          DCX     D               ;DECREMENT LENGTH SO WE CAN
 5812                                  19100                                  ;JUST LOOK AT 'CARRY' AFTER
 5813                                  19120                                  ;CALLING COMPAR
```

```
                                        19140           XTHL                    ;TRADE POINTER AT VARIABLE WITH
                                        19160                                   ;INDEX INTO THE VARIABLE
                                        19180           COMPAR                  ;SEE IF INDEX IS TOO BIG
                                        19200           MVI     E,ERRBS         ;THATS ERROR ERRBS
                                        19220           JNC     ERROR           ;SINCE LENGTH REALLY HAS AN
                                        19240                                   ;EXTRA ONE ADDED TO IT
                                        19260                                   ;IF INDEX=LENGTH DOESN'T CARRY
                                        19280                                   ;HE IS IN TROUBLE
                                        19300           POP     D               ;POP OFF POINTER AT VARIABLE
                                        19320           DAD     D               ;ADD IT TO THE INDEX
                                        19340           POP     D               ;POP OFF TEXT POINTER
                                        19360           XCHG                    ;TEXT POINTER INTO (H,L)
                                        19380           RET                     ;VARIABLE POINTER INTO (D,E)
                                        19400
                                        19420   NOTFDD: MOV     M,E             ;PUT LOOKS DOWN
                                        19440           INX     H
                                        19460           MOV     M,D
                                        19480           INX     H
                                        19500           LXI     D,$CODE+44      ;DEFAULT SIZE IS 10
                                        19520           LDA     DIMFLG          ;ARE WE DIMENSIONING
                                        19540           ORA     A
                                        19560           JZ      NOTDIM
                                        19580           POP     D               ;POP OFF INDEX
                                        19600           PUSH    D               ;PUT INDEX BACK ON
                                        19620           INX     D
                                        19640           INX     D
                                        19660           INX     D
                                        19680           INX     D
                                        19700   NOTDIM: PUSH    D
                                        19720           MOV     M,E             ;PUT LENGTH DOWN
                                        19740           INX     H
                                        19760           MOV     M,D
                                        19780           INX     H
                                        19800           PUSH    H
                                        19820           DAD     D
                                        19840           CALL    REASON          ;MAKE SURE WE'RE NOT RUNNING
                                        19860                                   ;INTO THE STACK
                                        19880           SHLD    STREND          ;SETUP NEW STORAGE END
                                        19900           POP     D
                                        19920   ZERIT2: DCX     H
                                        19940           MVI     M,0
                                        19960           COMPAR
                                        19980           JNZ     ZERIT2
                                        20000           JMP     MAKOFN>         ;FINISH UP
                                        20020
                                        20040   PAGE
```



```
                                        20060   SUBTTL  MULTIPLE DIMENSION CODE
                                        20080
                                        20100   IFN     MULDIM,<
                                        20120   ;
                                        20140   ; FORMAT OF ARRAYS IN CORE
                                        20160   ;
                                        20180   ; DESCRIPTOR
                                        20200   ;       LOW BYTE = SECOND CHARCTER (200 BIT IS STRING FLAG)
                                        20220   ;       HIGH BYTE = FIRST CHARACTER
                                        20240   ; LENGTH OF ARRAY IN CORE IN BYTES (DOES NOT INCLUDE DESCRIPTOR)
                                        20260   ; NUMBER OF DIMENSIONS 1 BYTE
                                        20280   ; FOR EACH DIMENSION STARTING WITH THE FIRST A LIST
                                        20300   ; (2 BYTES EACH) OF THE MAX INDICE+1
                                        20320   ; THE VALUES
                                        20340   ;
006745' 001000  000345  20360   ISARY:  PUSH    H               ;SAVE DIMFLG AND VALTYP FOR RECURSION
006746' 001000  000052  20380           LHLD    DIMFLG
006747' 000000  001542'
006750' 000000  006730'
006751' 001000  000343  20400           XTHL                    ;TEXT POINTER BACK INTO (H,L)
006752' 001000  000026  20420           MVI     D,0             ;SET # DIMENSIONS =0
006753' 000000  000000
006754' 001000  000325  20440   INDLOP: PUSH    D               ;SAVE NUMBER OF DIMENSIONS
006755' 001000  000305  20460           PUSH    B               ;SAVE LOOKS
006756' 001000  003115  20480           CALL    INTIOX          ;EVALUATE INDICE INTO (D,E)
006757' 000000  003622'
006760' 000000  006747'
006761' 001000  000301  20500           POP     B               ;POP OFF THE LOOKS
006762' 001000  000361  20520           POP     PSW             ;(A) = NUMBER OF DIMENSIONS SO FAR
006763' 001000  000353  20540           XCHG                    ;(D,E)=TEXT POINTER
                                        20560                   ;(H,L)=INDICE
006764' 001000  000343  20580           XTHL                    ;PUT THE INDICE ON THE STACK
                                        20600                   ;(H,L)=VALTYP & DIMFLG
006765' 001000  000345  20620           PUSH    H               ;RESAVE VALTYP AND DIMFLG
006766' 001000  000353  20640           XCHG                    ;(H,L)=TEXT POINTER
006767' 001000  000074  20660           INR     A               ;INCREMENT # OF DIMENSIONS
006770' 001000  000127  20680           MOV     D,A             ;(D)=NUMBER OF DIMENSIONS
006771' 001000  000176  20700           MOV     A,M             ;GET TERMINATING CHARACTER
006772' 001000  000376  20720           CPI     44              ;A COMMA SO MORE INDICES FOLLOW?
006773' 000000  000054
006774' 001000  000312  20740           JZ      INDLOP          ;IF SO, READ MORE
006775' 000000  006754'
006776' 000000  006757'
006777' 001000  000317  20760           SYNCHK  ")"             ;MAKE SURE IT ENDED PROPERLY
007000' 000000  000051
007001' 001000  000042  20780           SHLD    TEMP2           ;SAVE THE TEXT POINTER
007002' 000000  001605'
007003' 000000  006775'
007004' 001000  000341  20800           POP     H               ;(H,L)= VALTYP & DIMFLG
007005' 001000  000042  20820           SHLD    DIMFLG          ;SAVE VALTYP AND DIMFLG
007006' 000000  001542'
007007' 000000  007002'
007010' 001000  000325  20840           PUSH    D               ;SAVE NUMBER OF DIMENSIONS
```

```
                                    20860   ;
                                    20880   ; AT THIS POINT [B,C]=LOOKS, THE TEXT POINTER IS IN TEMP2,
                                    20900   ; THE INDICES ARE ALL ON THE STACK, FOLLOWED BY THE NUMBER OF DIMENSIONS,
                                    20920   ;
007011' 001000  000052              20940           LHLD    ARYTAB          ;[H,L]=PLACE TO START THE SEARCH
007012' 000000  001623'
007013' 000000  007006'
007014' 001000  000076              20960           XWD     "01000,"076     ;"MVI A," AROUND THE NEXT BYTE
007015' 001000  000031              20980   LOPFDA: DAD     D               ;SKIP OVER THIS ARRAY SINCE IT'S
                                    21000                                   ;NOT THE ONE
007016' 001000  000353              21020           XCHG                    ;[D,E]=CURRENT SEARCH POINT
007017' 001000  000052              21040           LHLD    STREND          ;GET THE PLACE TO STOP INTO [H,L]
007020' 000000  001625'
007021' 000000  007012'
007022' 001000  000353              21050           XCHG                    ;[H,L]=SEARCH POINT
007023' 001000  000347              21060           COMPAR                  ;STOPPING TIME?
                                    21080   IFE     LENGTH-2,<
007024' 001000  000072              21100           LDA     VALTYP>
007025' 000000  001543'
007026' 000000  007020'
007027' 001000  000312              21120           JZ      NOTFDD          ;YES,COULDN'T FIND THIS ARRAY
007030' 000000  007104'
007031' 000000  007025'
                                    21140   IFE     LENGTH-2,<
007032' 001000  000276              21160           CMP     M               ;SEE IF THE VALTYPS ARE THE SAME
007033' 001000  000043              21180           INX     H
007034' 001000  000302              21200           JNZ     NMARY2>
007035' 000000  007050'
007036' 000000  007030'
007037' 001000  000176              21240           MOV     A,M             ;GET FIRST CHARACTER
007040' 001000  000271              21260           CMP     C               ;SEE IF IT MATCHES
007041' 001000  000043              21280           INX     H
007042' 001000  000302              21300           JNZ     NMARY1          ;NOT THIS ONE
007043' 000000  007051'
007044' 000000  007035'
007045' 001000  000176              21320           MOV     A,M             ;GET SECOND CHARACTER
007046' 001000  000270              21340           CMP     B               ;ANOTHER MATCH?
                                    21360   IFE     LENGTH-2,<
007047' 001000  000076              21380           XWD     "01000,"076     ;SKIP THIS INCREMENT WITH "MVI A,"
007050' 001000  000043              21400   NMARY2: INX     H>
007051' 001000  000043              21420   NMARY1: INX     H               ;POINT TO SIZE ENTRY
007052' 001000  000136              21440           MOV     E,M             ;[D,E]=LENGTH
007053' 001000  000043              21460           INX     H               ;OF THE ARRAY BEING LOOKED AT
007054' 001000  000126              21480           MOV     D,M
007055' 001000  000043              21500           INX     H
007056' 001000  000302              21520           JNZ     LOPFDA          ;IF NO MATCH, SKIP THIS ONE
007057' 000000  007015'
007060' 000000  007043'
                                    21540                                   ;AND TRY AGAIN
007061' 001000  000072              21560           LDA     DIMFLG          ;SEE IF CALLED BY "DIM"
007062' 000000  001542'
007063' 000000  007057'
007064' 001000  000267              21580           ORA     A               ;ZERO MEANS NO
```



```
007065' 001000  000036              21600           MVI     E,ERRDD         ;"DOUBLY DIMENSIONED" ERROR
007066' 000000  000012
007067' 001000  000302              21620           JNZ     ERROR
007070' 000000  002102'
007071' 000000  007062'
                                    21640   ;
                                    21660   ; TEMP2=THE TEXT POINTER
                                    21680   ; WE HAVE LOCATED THE VARIABLE WE WERE LOOKING FOR
                                    21700   ; AT THIS POINT [H,L] POINTS BEYOND THE SIZE TO THE NUMBER OF DIMENSIONS
                                    21720   ; THE INDICES ARE ON THE STACK FOLLOWED BY THE NUMBER OF DIMENSIONS
                                    21740   ;
007072' 001000  000361              21760           POP     PSW             ;[A]=NUMBER OF DIMENSIONS
007073' 001000  000276              21780           CMP     M               ;MAKE SURE THE NUMBER GIVEN NOW AND
                                    21800                                   ;AND WHEN THE ARRAY WAS SET UP ARE THE
                                    21820                                   ;SAME
007074' 001000  000312              21840           JZ      GETDEF          ;JUMP OFF AND READ
007075' 000000  007245'
007076' 000000  007070'
                                    21860                                   ;THE INDICES,...
007077' 001000  000036              21880   BSERR:  MVI     E,ERRBS         ;"SUBSCRIPT OUT OF RANGE"
007100' 000000  000011
007101' 001000  000303              21900           JMP     ERROR
007102' 000000  002102'
007103' 000000  007075'
                                    21920   ;
                                    21940   ; HERE WHEN VARIABLE IS NOT FOUND IN THE ARRAY TABLE
                                    21960   ;
                                    21980   ; BUILDING AN ENTRY:
                                    22000   ;
                                    22020   ;       PUT DOWN THE DESCRIPTOR
                                    22040   ;       SETUP NUMER OF DIMENSIONS
                                    22060   ;       MAKE SURE THERE IS ROOM FOR THE NEW ENTRY
                                    22080   ;       REMEMBER VARPTR
                                    22100   ;       TALLY=0 (VALTYP FOR THE EXTENDED)
                                    22120   ;       SKIP 2 LOCS FOR LATER FILL IN -- THE SIZE
                                    22140   ; LOOP: GET AN INDICE
                                    22160   ;       PUT NUMBER +1 DOWN AT VARPTR AND INCREMENT VARPTR
                                    22180   ;       TALLY= TALLY * NUMBER+1
                                    22200   ;       DECREMENT NUMBER=DIMS
                                    22220   ;       JNZ     LOOP
                                    22240   ;       CALL REASON WITH [H,L] REFLECTING LAST LOC OF VARIABLE
                                    22260   ;       UPDATE STREND
                                    22280   ;       ZERO BACKWARDS
                                    22300   ;       MAKE TALLY INCLUDE MAXDIMS
                                    22320   ;       PUT DOWN TALLY
                                    22340   ;       IF CALLED BY DIMENSION, RETURN
                                    22360   ;       OTHERWISE INDEX INTO THE VARIABLE AS IF IT
                                    22380   ;       WERE FOUND ON THE INITIAL SEARCH
                                    22400   ;
007104'                             22420   NOTFDD:
                                    22440   IFE     LENGTH-2,<
007104' 001000  000167              22460           MOV     M,A             ;PUT DOWN THE VARIABLE TYPE
007105' 001000  000043              22480           INX     H
```

```
 6019  007106' 001000  000137          22500          MOV     E,A
 6020  007107' 001000  000026          22520          MVI     D,0>            ;[D,E]=SIZE OF ONE VALUE (VALTYP)
 6021  007110' 000000  000000
 6022                                  22540  IFN     LENGTH-2,<
 6023                                  22560          LXI     D,$CODE+4>      ;INITIALIZE TALLY TO FOUR
 6024  007111' 001000  000161          22580          MOV     M,C             ;PUT DOWN THE DESCRIPTOR
 6025  007112' 001000  000043          22600          INX     H
 6026  007113' 001000  000160          22620          MOV     M,B
 6027  007114' 001000  000043          22640          INX     H
 6028  007115' 001000  000361          22660          POP     PSW             ;[A]=NUMBER OF DIMENSIONS
 6029  007116' 001000  000062          22680          STA     TEMP6           ;SETUP GETSTK CALL
 6030  007117' 000000  007124'
 6031  007120' 000000  007102'
 6032  007121' 001000  000315          22700          CALL    GETSTK          ;GET SPACE FOR DIMENSION ENTRIES
 6033  007122' 000000  002024'
 6034  007123' 000000  007117'
 6035  007124' 001000  000351          22720  TEMP6:  PCHL                    ;PLACE TO STORE NUMBER OF DIMENSIONS
 6036                                  22740                                  ;FOR GETSTK AND LATER RECALL
 6037                                  22760                                  ;!!IMPURE!! PCHL TO CONFUSE DISASSEMBLY
 6038  007125' 001000  000042          22780          SHLD    TEMP3           ;SAVE THE LOCATION TO PUT THE SIZE
 6039  007126' 000000  001575'
 6040  007127' 000000  007122'
 6041                                  22800                                  ;IN
 6042  007130' 001000  000043          22820          INX     H               ;SKIP OVER THE SIZE LOCATIONS
 6043  007131' 001000  000043          22840          INX     H
 6044  007132' 001000  000101          22860          MOV     B,C             ;[B]=NUMBER OF DIMENSIONS
 6045                                  22880                                  ;THIS DEPENDS ON THE FACT THAT GETSTK
 6046                                  22900                                  ;RETURNS ITS ARGUMENT IN (C)
 6047  007133' 001000  000160          22920          MOV     M,B             ;STORE THE NUMBER OF DIMENSIONS
 6048  007134' 001000  000043          22940          INX     H
 6049  007135' 001000  000072          22960  LOPPTA: LDA     DIMFLG          ;CALLED BY DIMENSION?
 6050  007136' 000000  001542'
 6051  007137' 000000  007126'
 6052  007140' 001000  000267          22980          ORA     A
 6053  007141' 001000  000170          23000          MOV     A,B             ;[A]=NUMBER OF DIMENSIONS
 6054  007142' 001000  000001          23020          LXI     B,$CODE+11      ;ASSUME ITS NOT "DIM"
 6055  007143' 000000  000013'
 6056  007144' 000000  007136'
 6057  007145' 001000  000312          23040          JZ      NOTDIM          ;DEFAULT DIMENSIONS TO TEN
 6058  007146' 000000  007152'
 6059  007147' 000000  007143'
 6060  007150' 001000  000301          23060          POP     B               ;POP OFF AN INDICE INTO [B,C]
 6061  007151' 001000  000003          23080          INX     B               ;ADD ONE TO IT FOR THE ZERO ENTRY
 6062  007152' 001000  000161          23100  NOTDIM: MOV     M,C             ;PUT THE MAXIMUM DOWN
 6063  007153' 001000  000043          23120          INX     H
 6064  007154' 001000  000160          23140          MOV     M,B
 6065  007155' 001000  000043          23160          INX     H
 6066  007156' 001000  000365          23180          PUSH    PSW             ;SAVE THE NUMBER OF DIMENSIONS
 6067  007157' 001000  000345          23200          PUSH    H               ;SAVE THE POINTER INTO THE NEW ENTRY
 6068  007160' 001000  000315          23220          CALL    UMULT           ;MULTIPLY [B,C]=NEWMAX BY CURTOL=[D,E]
 6069  007161' 000000  000000*
 6070  007162' 000000  007146'
 6071  007163' 001000  000353          23240          XCHG                    ;[D,E]=NEW CURTOL
```



```
 6072  007164' 001000  000341          23260          POP     H               ;GET THE POINTER INTO THE ENTRY BACK
 6073  007165' 001000  000301          23280          POP     B               ;GET THE NUMBER OF DIMENSIONS BACK
 6074  007166' 001000  000005          23300          DCR     B               ;DECREMENT THE NUMBER OF DIMENSIONS LEFT
 6075  007167' 001000  000302          23320          JNZ     LOPPTA          ;HANDLE THE OTHER INDICES
 6076  007170' 000000  007135'
 6077  007171' 000000  007161'
 6078  007172' 001000  000102          23340          MOV     B,D             ;[B,C]=SIZE
 6079  007173' 001000  000113          23360          MOV     C,E
 6080  007174' 001000  000353          23380          XCHG                    ;[D,E]=START OF VALUES
 6081  007175' 001000  000031          23400          DAD     D               ;[H,L]=END OF VALUES
 6082  007176' 001000  000332          23420          JC      BSERR           ;OUT OF MEMORY POINTER BEING GENERATED?
 6083  007177' 000000  007077'
 6084  007200' 000000  007170'
 6085  007201' 001000  000315          23440          CALL    REASON          ;SEE IF THERE IS ROOM FOR THE VALUES
 6086  007202' 000000  002045'
 6087  007203' 000000  007177'
 6088  007204' 001000  000042          23460          SHLD    STREND          ;UPDATE THE END OF STORAGE
 6089  007205' 000000  001625'
 6090  007206' 000000  007202'
 6091  007207' 001000  000053          23480  ZERITA: DCX     H               ;ZERO THE NEW ARRAY
 6092  007210' 001000  000066          23500          MVI     M,0
 6093  007211' 000000  000000
 6094  007212' 001000  000347          23520          COMPAR                  ;BACK AT THE BEGINNING?
 6095  007213' 001000  000302          23540          JNZ     ZERITA          ;NO,ZERO MORE
 6096  007214' 000000  007207'
 6097  007215' 000000  007205'
 6098  007216' 001000  000003          23560          INX     B               ;ADD ONE TO THE SIZE TO INCLUDE
 6099                                  23580                                  ;THE BYTE FOR THE NUMBER OF DIMENSIONS
 6100  007217' 001000  000147          23600          MOV     M,A             ;[M]=ZERO
 6101  007220' 001000  000072          23620          LDA     DIMFLG
 6102  007221' 000000  001542'
 6103  007222' 000000  007214'
 6104  007223' 001000  000267          23640          ORA     A               ;ARE WE DIMENSIONING ?
 6105  007224' 001000  000072          23660          LDA     TEMP6           ;GET THE NUMBER OF DIMENSIONS
 6106  007225' 000000  007124'
 6107  007226' 000000  007221'
 6108  007227' 001000  000157          23680          MOV     L,A             ;[L]=NUMBER OF DIMENSIONS
 6109  007230' 001000  000051          23700          DAD     H               ;[H,L]=NUMBER OF DIMENSIONS TIMES TWO
 6110  007231' 001000  000011          23720          DAD     B               ;ADD ON THE SIZE
 6111                                  23740                                  ;TO GET THE TOTAL NUMBER OF BYTES USED
 6112  007232' 001000  000353          23760          XCHG                    ;[D,E]=TOTAL SIZE
 6113  007233' 001000  000052          23780          LHLD    TEMP3           ;PLACE TO STORE SIZE
 6114  007234' 000000  001575'
 6115  007235' 000000  007225'
 6116  007236' 001000  000163          23800          MOV     M,E             ;PUT DOWN THE SIZE
 6117  007237' 001000  000043          23820          INX     H
 6118  007240' 001000  000162          23840          MOV     M,D
 6119  007241' 001000  000043          23860          INX     H
 6120  007242' 001000  000302          23880          JNZ     FINNOW
 6121  007245' 000000  007331'
 6122  007244' 000000  007234'
 6123                                  23900  ;
 6124                                  23920  ; AT THIS POINT [M,L] POINTS BEYOND THE SIZE TO THE NUMBER OF DIMENSIONS
```

```
                                        23940   ; STRATEGY:
                                        23960   ;       NUMDIM=NUMBER OF DIMENSIONS
                                        23980   ;       CURTOL=0
                                        24000   ; INLPNM:GET A NEW INDICE
                                        24020   ;       POP NEW MAX INTO CURMAX
                                        24040   ;       MAKE SURE INDICE IS NOT TOO BIG
                                        24060   ;       MULTIPLY CURTOL BY CURMAX
                                        24080   ;       ADD INDICE TO CURTOL
                                        24100   ;       NUMDIM=NUMDIM-1
                                        24120   ;       JNZ     INLPNM
                                        24140   ;       USE CURTOL*4 (VALTYP FOR EXTENDED) AS OFFSET
                                        24160   ;
007245' 001000  000043          24180   GETDEF: INX     H               ;POINT PAST THE NUMBER OF DIMENSIONS
007246' 001000  000001          24200           LXI     B,$CODE         ;CURTOL=ZERO
007247' 000000  000000'
007250' 000000  007243'
007251' 001000  000026          24220           XWD     ^O1000,^O26     ;"MVI D," AROUND THE NEXT BYTE
007252' 001000  000341          24240   INLPNM: POP     H               ;(H,L)= POINTER INTO VARIABLE ENTRY
007253' 001000  000136          24260           MOV     E,M             ;(D,E)=MAXIMUM FOR THE CURRENT INDICE
007254' 001000  000043          24280           INX     H
007255' 001000  000126          24300           MOV     D,M
007256' 001000  000043          24320           INX     H
007257' 001000  000343          24340           XTHL                    ;(H,L)=CURRENT INDICE
                                        24360                           ;POINTER INTO THE VARIABLE GOES ON THE STACK
007260' 001000  000365          24380           PUSH    PSW             ;SAVE THE NUMBER OF DIMENSIONS
007261' 001000  000347          24400           COMPAR                  ;SEE IF THE CURRENT INDICE IS TOO BIG
007262' 001000  000322          24420           JNC     BSERR           ;IF SO "BAD SUBSCRIPT" ERROR
007263' 000000  007077'
007264' 000000  007247'
007265' 001000  000345          24440           PUSH    H               ;SAVE THE CURRENT INDICE
007266' 001000  000315          24460           CALL    UMULT           ;CURTOL=CURTOL*CURRENT MAXIMUM
007267' 000000  007161*
007270' 000000  007263'
007271' 001000  000321          24480           POP     D               ;INDICE INTO (D,E)
007272' 001000  000031          24500           DAD     D               ;ADD THE INDICETO CURTOL
007273' 001000  000361          24520           POP     PSW             ;GET THE NUMBER OF DIMENSIONS IN (A)
007274' 001000  000075          24540           DCR     A               ;SEE IF ALL THE INDICES HAVE BEEN PROCESSED
007275' 001000  000104          24560           MOV     B,H             ;(B,C)=CURTOL IN CASE WE LOOP BACK
007276' 001000  000115          24580           MOV     C,L
007277' 001000  000302          24600           JNZ     INLPNM          ;PROCESS THE REST OF THE INDICES
007300' 000000  007252'
007301' 000000  007267'
                                        24620   IFE     LENGTH-2,<
007302' 001000  000072          24640           LDA     VALTYP          ;SEE HOW BIG THE VALUES ARE
007303' 000000  001543'
007304' 000000  007300'
                                        24660                           ;AND MULTIPLY BY THAT SIZE
007305' 001000  000104          24680           MOV     B,H             ;SAVE THE ORIGINAL VALUE FOR MULTIPLYING
007306' 001000  000115          24700           MOV     C,L             ;BY THREE
007307' 001000  000051          24720           DAD     H               ;MULTIPLY BY TWO AT LEAST
007310' 001000  000326          24740           SUI     4               ;FOR INTEGERS AND STRINGS
007311' 000000  000004
                                        24760                           ;NO MORE MULTIPLYING BY TWO
```



```
007312' 001000  000332          24780           JC      SMLVAL
007313' 000000  007322'
007314' 000000  007303'
007315' 001000  000051          24800           DAD     H               ;NOW MULTIPLIED BY FOUR
007316' 001000  000312          24820           JZ      DONMUL          ;IF SINGLE ALL DONE
007317' 000000  007326'
007320' 000000  007313'
007321' 001000  000051          24840           DAD     H               ;BY EIGHT FOR DOUBLES
007322' 001000  000342          24860   SMLVAL: JPO     DONMUL          ;FOR STRINGS
007325' 000000  007326'
007324' 000000  007317'
007325' 001000  000011          24880           DAD     B               ;ADD IN THE ORIGINAL
007326'                         24900   DONMUL:>
                                        24920   IFN     LENGTH-2,<
                                        24940           DAD     H       ;MULTIPLY CURTOL BY FOUR
                                        24960           DAD     H>
007326' 001000  000301          24980           POP     B               ;POP OFF THE ADDRESS OF WHERE THE VALUES
                                        25000                           ;BEGIN
007327' 001000  000011          25020           DAD     B               ;ADD IT ONTO CURTOL TO GET THE
                                        25040                           ;PLACE THE VALUE IS STORED
007330' 001000  000353          25060           XCHG                    ;RETURN THE POINTER IN (D,E)
007331' 001000  000052          25080   FINNOW: LHLD    TEMP2           ;REGET THE TEXT POINTER
007332' 000000  001605'
007333' 000000  007323'
007334' 001000  000053          25100           DCX     H               ;REREAD THE TERMINATING CHARACTER
007335' 001000  000327          25120           CHRGET
007336' 001000  000311          25140           RET>
                                        25160
                                        25180   PAGE
```

```
                                        25200   SUBTTL FRE  FUNCTION AND INTEGER TO FLOATING  ROUTINES
                                        25220   IFN     LENGTH,<
6209  007337' 001000  000052           25240   FRE:    LHLD    STREND          ;GET END OF VARIABLE AND TEXT SPACE
6210  007340' 000000  001625'
6211  007341' 000000  007332'
6212  007342' 001000  000353           25260           XCHG                    ;PUT IT IN [D,E] FOR SUBTRACTION
6213  007343' 001000  000041           25280           LXI     H,$CODE         ;ZERO [H,L]
6214  007344' 000000  000000'
6215  007345' 000000  007340'
6216  007346' 001000  000071           25300           DAD     SP              ;PUT THE STACK POINTER IN [H,L]
6217                                    25320   IFN     STRING,<
6218                                    25340   IFE     LENGTH=2,<
6219  007347' 001000  000315           25360           CALL    GETYPE
6220  007350' 000000  006307'
6221  007351' 000000  007344'
6222  007352' 001000  000302           25370           JNZ     GIVDBL>
6223  007353' 000000  00737E'
6224  007354' 000000  007350'
6225                                    25380   IFN     LENGTH=2,<
6226                                    25400           LDA     VALTYP          ;WAS THE ARGUMENT A STRING?
6227                                    25420           ORA     A
6228                                    25440           JZ      GIVDBL>         ;NO, GIVE FREE VARIABLE SPACE
6229  007355' 001000  000315           25460           CALL    FREFAC          ;FREE UP ARGUMENT AND SETUP
6230  007356' 000000  010434'
6231  007357' 000000  007353'
6232                                    25480                                   ;TO GIVE FREE STRING SPACE
6233  007360' 001000  000315           25500           CALL    GARBA2          ;DO GARBAGE COLLECTION
6234  007361' 000000  010042'
6235  007362' 000000  007356'
6236  007363' 001000  000052           25520           LHLD    STKTOP          ;BOTTOM OF FREE AREA
6237  007364' 000000  001615'
6238  007365' 000000  007361'
6239  007366' 001000  000353           25540           XCHG
6240  007367' 001000  000052           25560           LHLD    FRETOP>>        ;TOP OF FREE AREA
6241  007370' 000000  001573'
6242  007371' 000000  007364'
6243                                    25580   ;
6244                                    25600   ; THIS ROUTINE SUBTRACTS [D,E] FROM [H,L]
6245                                    25620   ; AND FLOATS THE RESULT LEAVING IT IN FAC.
6246                                    25640   ;
6247                                    25660   IFE     LENGTH=1,<
6248                                    25680   GIVDBL: MOV     A,L             ;DO THE SUBTRACTION
6249                                    25700           SUB     E
6250                                    25720           MOV     C,A
6251                                    25740           MOV     A,H
6252                                    25760           SBB     D
6253                                    25780   GIVACF: MOV     B,C>
6254                                    25790   IFN     LENGTH=2,<
6255                                    25800   GIVABF: MOV     D,B
6256                                    25820           MVI     E,0             ;GET ZERO IN LOW
6257                                    25840   IFN     STRING,<
6258                                    25860           LXI     H,VALTYP        ;FLAG VALUE TYPE AS NUMERIC
6259                                    25880           MOV     M,E>
```



```
                                        25900           MVI     B,144           ;SETUP TO FLOAT [B,C]
                                        25920           JMP     FLOATR>
                                        26120   IFE     LENGTH=2,<
6263  007372' 001000  000175           26140   GIVDBL: MOV     A,L             ;[H,L]=[H,L]-[D,E]
6264  007373' 001000  000223           26160           SUB     E
6265  007374' 001000  000157           26180           MOV     L,A
6266  007375' 001000  000174           26200           MOV     A,H
6267  007376' 001000  000232           26220           SBB     D
6268  007377' 001000  000021           26240           XWD     ^O1000,^O21     ;SKIP THE NEXT TWO BYTES WITH "LXI D,"
6269  007400' 001000  000157           26260   SNGFLT: MOV     L,A             ;MAKE [A] AN UNSIGNED INTEGER
6270  007401' 001000  000257           26280           XRA     A
6271  007402' 001000  000147           26300   GIVINT: MOV     H,A
6272  007403' 001000  000303           26320           JMP     MAKINT>
6273  007404' 000000  006471*
6274  007405' 000000  007370'
6275                                    26322   IFN     LENGTH,<
6276                                    26324   IFN     LPTSW,<
6277                                    26326   LPOS:   LDA     LPTPOS
6278                                    26328           JMP     SNGFLT>
6279  007406' 001000  000072           26330   POS:    LDA     TTYPOS          ;GET TELETYPE POSITION
6280  007407' 000000  000047'
6281  007410' 000000  007404'
6282                                    26332   IFN     LENGTH=2,<
6283                                    26334   SNGFLT: MOV     B,A             ;RETURN FLOATING 1 BYTE
6284                                    26336           XRA     A               ;UNSIGNED FROM A
6285                                    26338           JMP     GIVABF>>        ;GIVING 0-255
6286
6287                                    26360   PAGE
```

```
                                        26380   SUBTTL  SIMPLE-USER-DEFINED-FUNCTION CODE
                                        26390   IFN     FUNCTS,<
                                        26400   ;
                                        26420   ; NOTE ONLY SINGLE ARGUMENTS ARE ALLOWED TO FUNCTIONS
                                        26440   ; AND FUNCTIONS MUST BE OF THE SINGLE LINE FORM:
                                        26460   ; DEF FNA(X)=X^2+X+2
                                        26480   ; NO STRINGS CAN BE INVOLVED WITH THESE FUNCTIONS
                                        26520   ; IDEA: CREATE A FUNNY SIMPLE VARIABLE ENTRY
                                        26540   ; WHOSE FIRST CHARACTER (SECOND WORD IN MEMORY)
                                        26560   ; HAS THE 200 BIT SET,
                                        26580   ; THE VALUE WILL BE:
                                        26600   ;
                                        26620   ;       A TXTPTR TO THE FORMULA
                                        26640   ;       A PTR TO THE ARGUMENT VARIABLE
                                        26660   ;
                                        26680   ; FUNCTION NAMES CAN BE LIKE "FNA4"
                                        26700   ;
007411' 001000  000315          26780   DEF:    CALL    GETFNM          ;GET A POINTER TO THE
007412' 000000  007550'
007413' 000000  007407'
                                        26800                           ;FUNCTION VARAIBLE
007414' 001000  000001          26820           LXI     B,DATA          ;EVENTUALLY RETURN TO "DATA"
007415' 000000  004072'
007416' 000000  007412'
007417' 001000  000305          26840           PUSH    B               ;AND SKIP THE FORMULA
007420' 001000  000325          26860           PUSH    D               ;SAVE A POINTER TO IT
007421' 001000  000315          26880           CALL    ERRDIR          ;DEF IS "ILLEGAL DIRECT"
007422' 000000  007532'
007423' 000000  007415'
007424' 001000  000317          26900           SYNCHK  "("             ;MUST HAVE "("
007425' 000000  000050
                                        26920                           ;SINCE WE STORE A TEXT POINTER
007426' 001000  000315          26940           CALL    PTRGET          ;GET POINTER TO ARGUMENT
007427' 000000  006505'
007430' 000000  007422'
                                        26960   IFN     LENGTH=2,<
                                        26980   IFN     STRING,<CALL    CHKNUM>>;STRINGS ILLEGAL
007431' 001000  000317          27000           SYNCHK  ")"             ;MUST CLOSE IT WITH ")"
007432' 000000  000051
007433' 001000  000317          27020           SYNCHK  EQULTK          ;MUST HAVE EQUAL
007434' 000000  000260
007435' 001000  000104          27040           MOV     B,H
007436' 001000  000115          27060           MOV     C,L
007437' 001000  000343          27080           XTHL                    ;PUT THE TXTPTR ON THE STACK
                                        27100                           ;(H,L)=PTR TO FUNCTION VARIABLE
                                        27120                           ;(B,C)=TXTPTR
007440' 001000  000303          27140           JMP     DEFFIN          ;PUT DOWN THE TEXT-POINTER
007441' 000000  007521'
007442' 000000  007427'
                                        27160                           ;AND ARGUMENT POINTER IN
                                        27180                           ;MEMORY, RESTORE THE TXTPTR
```



```
                                        27200                           ;AND GO TO "DATA" SKIPPING THE
                                        27220                           ;REST OF THE FORMULA
007443' 001000  000315          27260   FNDOER: CALL    GETFNM          ;GET A POINTER TO
007444' 000000  007550'
007445' 000000  007441'
                                        27280                           ;THE FUNCTON DEFINITION IN (D,E)
007446' 001000  000325          27300           PUSH    D               ;SAVE THE POINTER
007447' 001000  000315          27320           CALL    PARCHK          ;EVALUATE THE VALUE TO BE PASSED
007450' 000000  006136'
007451' 000000  007444'
                                        27340   IFN     LENGTH=2,<
                                        27360   IFN     STRING,<CALL    CHKNUM>> ;ARG CANNOT BE STRING
007452' 001000  000343          27380           XTHL                    ;(H,L)=POINTER TO FUNCTION DEF
                                        27400                           ;TEXT POINTER GOES ON THE STACK
007453' 001000  000367          27420           PUSHM                   ;PUSH THE POINTER AT THE FORMULA
                                        27440                           ;ONTO THE STACK
007454' 001000  000321          27460           POP     D               ;(D,E)=PTR TO FORMULA
007455' 001000  000367          27480           PUSHM                   ;PUT A POINTER TO THE
                                        27500                           ;ARGUMENT ON THE STACK
007456' 001000  000341          27520           POP     H               ;(H,L)=POINTER TO ARG
007457' 001000  000367          27540           PUSHFM                  ;SAVE ARGS OLD VALUE ON THE STACK
007460' 001000  000367
007461' 001000  000053          27560           DCX     H
007462' 001000  000053          27580           DCX     H
007463' 001000  000053          27600           DCX     H               ;POINT TO FRONT OF ARG AGAIN
007464' 001000  000053          27620           DCX     H
007465' 001000  000345          27640           PUSH    H               ;SAVE IT
007466' 001000  000347          27660           COMPAR                  ;SHOULDN'T BE EQUAL UNLESS
                                        27680                           ;FUNCTION WAS NEVER DEFINED
007467' 001000  000325          27700           PUSH    D               ;SAVE FORMULA TEXT POINTER
007470' 001000  000036          27720           MVI     E,ERRUF         ;NOW (D,E) FREE SO CHECK IF (ZERO) SET
007471' 000000  000022
007472' 001000  000312          27740           JZ      ERROR
007473' 000000  002102'
007474' 000000  007450'
007475' 001000  000315          27760           CALL    MOVMF           ;PUT CURRENT FAC INTO OUR ARG VARIABLE
007476' 000000  005264*
007477' 000000  007473'
                                        27780                           ;OUT OF FAC INTO (H,L) LOCATION
007500' 001000  000341          27800           POP     H               ;POP OFF FORMULA TXTPTR
                                        27810   IFN     LENGTH=2,<
                                        27820           CALL    FRMNUM>         ;EVALUATE IT AND MUST SURE ITS NUMERIC
                                        27822   IFE     LENGTH=2,<
007501' 001000  000315          27824           CALL    FRMEVL
007502' 000000  005336'
007503' 000000  007476'
007504' 001000  000345          27826           PUSH    H
007505' 001000  000315          27828           CALL    FRCSNG
007506' 000000  006667*
007507' 000000  007502'
007510' 001000  000341          27830           POP     H>
007511' 001000  000053          27840           DCX     H
```

```
 6394  007512' 001000  000327          27860          CHRGET                  ;SEE IF TERMINATED
 6395  007513' 001000  000302          27880          JNZ     SNERR           ;IF NOT SYNTAX ERROR
 6396  007514' 000000  002072'
 6397  007515' 000000  007506'
 6398                                  27900                                  ;TO BE NICE SHOULD HAVE NEW CURLIN
 6399                                  27920                                  ;BUT VERY MESSY
 6400  007516' 001000  000341          27940          POP     H               ;POP OFF POINTER AT ARG VARIABLE
 6401  007517' 001000  000321          27960          POP     D
 6402  007520' 001000  000301          27980          POP     B>              ;POP OFF OLD VALUE
 6403                                  28000  IFN     MULDIMISTRINGIFUNCTS,<
 6404  007521' 001000  000161          28020  DEFFIN: MOV     M,C
 6405  007522' 001000  000043          28040          INX     H               ;STORE THE OLD VALUE
 6406  007523' 001000  000160          28060          MOV     M,B
 6407  007524' 001000  000043          28080  PUTDEI: INX     H
 6408  007525' 001000  000163          28100          MOV     M,E
 6409  007526' 001000  000043          28120          INX     H
 6410  007527' 001000  000162          28140          MOV     M,D
 6411  007530' 001000  000341          28160          POP     H               ;POP OFF OLD TXTPTR
 6412  007531' 001000  000311          28180          RET>                    ;VALUE IS IN FAC -- ALL DONE
 6413                                  28200  IFN     FUNCTS,<
 6414                                  28220  ;
 6415                                  28240  ; SUBROUTINE TO SEE IF WE ARE IN DIRECT MODE AND
 6416                                  28260  ; COMPLAIN IF SO
 6417                                  28280  ;
 6418  007532' 001000  000345          28300  ERRDIR: PUSH    H               ;SAVE THEIR [H,L]
 6419  007533' 001000  000052          28320          LHLD    CURLIN          ;SEE WHAT THE CURRENT LINE IS
 6420  007534' 000000  001607'
 6421  007535' 000000  007514'
 6422  007536' 001000  000043          28340          INX     H               ;DIRECT IS 65,535 SO NOW 0
 6423  007537' 001000  000174          28360          MOV     A,H
 6424  007540' 001000  000265          28380          ORA     L               ;IS IT ZERO NOW?
 6425  007541' 001000  000341          28400          POP     H
 6426  007542' 001000  000300          28420          RNZ                     ;RETURN IF NOT
 6427  007543' 001000  000036          28440          MVI     E,ERRID         ;"ILLEGAL DIRECT" ERROR
 6428  007544' 000000  000014
 6429  007545' 001000  000303          28460          JMP     ERROR
 6430  007546' 000000  002102'
 6431  007547' 000000  007534'
 6432                                  28480  ;
 6433                                  28500  ; SUBROUTINE TO GET A POINTER TO A FUNCTION NAME
 6434                                  28520  ;
 6435  007550' 001000  000317          28540  GETFNM: SYNCHK  FNTK            ;MUST START WITH "FN"
 6436  007551' 000000  000243
 6437  007552' 001000  000076          28560          MVI     A,128           ;DONT ALLOW AN ARRAY
 6438  007553' 000000  000200
 6439  007554' 001000  000062          28580          STA     SUBFLG          ;DON'T RECOGNIZE THE "(" AS
 6440  007555' 000000  001601'
 6441  007556' 000000  007546'
 6442                                  28600                                  ;THE START OF AN ARRAY REFEREENCE
 6443  007557' 001000  000266          28620          ORA     M               ;PUT FUNCTION BIT ON
 6444  007560' 001000  000107          28640          MOV     B,A             ;GET FIRST CHARACTER INTO [B]
 6445                                  28660  IFN     LENGTH-2,<
 6446                                  28680  IFN     STRING,<CALL    PTRGT2  ;REALLY GET THE POINTER
```



```
 6447                                  28700          JMP     CHKNUM>>        ;MAKE SURE ITS NOT A STRING NAME
 6448  007561' 001000  000303          28720  IFE     STRING&<LENGTH-2>,<JMP  PTRGT2>>
 6449  007562' 000000  006512'
 6450  007563' 000000  007555'
 6451                                  28740  PAGE
```

```
                                        28760   SUBTTL  STRING FUNCTIONS
                                        28780   IFN     STRING,<                ;STRING HANDLING SUBROUTINES
                                        28800   ;
                                        28820   ; THE STR$ FUNCTION TAKES A NUMBER AND GIVES
                                        28840   ; A STRING WITH THE CHARACTERS THE OUTPUT OF THE NUMBER
                                        28860   ; WOULD HAVE GIVEN
                                        28880   ;
007564'                                 28900   STR$:
                                        28920   IFN     LENGTH-2,<
                                        28940           CALL    CHKNUM>         ;MAKE SURE THE ARGUMENT
                                        28960                                   ;IS A NUMERIC
007564' 001000  000315                  28980           CALL    FOUT            ;DO ITS OUTPUT
007565' 010000  004457*
007566' 000000  007562'
007567' 001000  000315                  29000           CALL    STRLIT          ;SCAN IT AND TURN IT INTO A STRING
007570' 000000  007637'
007571' 000000  007565'
007572' 001000  000315                  29020           CALL    FREFAC          ;FREE UP THE TEMP
007573' 000000  010434'
007574' 000000  007570'
007575' 001000  000001                  29040           LXI     B,FINBCK
007576' 000000  010546'
007577' 000000  007573'
007600' 001000  000305                  29060           PUSH    B               ;SET UP ANSWER IN NEW TEMP
                                        29080   ;
                                        29100   ; STRCPY CREATES A COPY OF THE STRING
                                        29120   ; WHOSE DESCRIPTOR IS POINTED TO BY (H,L),
                                        29140   ; ON RETURN (D,E) POINTS TO DSCTMP
                                        29160   ; WHICH HAS THE STRING INFO (LENGTH,WHERE COPIED TO)
                                        29180   ;
007601' 001000  000176                  29200   STRCPY: MOV     A,M             ;GET LENGTH
007602' 001000  000043                  29220           INX     H               ;MOVE UP TO THE POINTER
                                        29240   IFN     LENGTH-2,<
                                        29260           INX     H>
007603' 001000  000345                  29280           PUSH    H               ;GET POINTER TO POINTER OF ARG
007604' 001000  000315                  29300           CALL    GETSPA          ;GET THE SPACE
007605' 000000  007772'
007606' 000000  007576'
007607' 001000  000341                  29320           POP     H               ;FIND OUT WHERE STRING TO COPY
007610' 001000  000367                  29340           PUSHM                   ;REALLY IS IN (B,C)
007611' 001000  000301                  29360           POP     B
007612' 001000  000315                  29380           CALL    STRAD2          ;SETUP DSCTMP
007613' 000000  007627'
007614' 000000  007605'
007615' 001000  000345                  29400           PUSH    H               ;SAVE POINTER TO DSCTMP
007616' 001000  000157                  29420           MOV     L,A             ;GET CHARACTER COUNT INTO (L)
007617' 001000  000315                  29440           CALL    MOVSTR          ;MOVE THE CHARS IN
007620' 000000  010417'
007621' 000000  007613'
007622' 001000  000321                  29460           POP     D               ;RESTORE POINTER TO DSCTMP
007623' 001000  000311                  29480           RET                     ;RETURN
                                        29500
007624' 001000  000315                  29520   STRINI: CALL    GETSPA          ;GET SOME STRING SPACE ((A) CHARS)
```



```
007625' 000000  007772'
007626' 000000  007620'
007627' 001000  000041                  29540   STRAD2: LXI     H,DSCTMP        ;GET DESC. TEMP
007630' 000000  001570'
007631' 000000  007625'
007632' 001000  000345                  29560   STRAD1: PUSH    H               ;SAVE DESC. POINTER
007633' 001000  000167                  29580           MOV     M,A             ;SAVE CHARACTER COUNT
                                        29600   IFN     LENGTH-2,<
                                        29620           INX     H>              ;MOVE TO ADDRESS FIELD
007634' 001000  000303                  29640           JMP     PUTDEI          ;USE COMMON CODE TO
007635' 000000  007524'
007636' 000000  007630'
                                        29660                                   ;STORE (D,E)=POINTER TO FREE SPACE
                                        29680                                   ;AND RESTORE (H,L) AS THE DESCRIPTOR POINTER
                                        29682   ;
                                        29684   ; STRLT2 TAKES THE STRING LITERAL WHOSE FIRST CHARACTER
                                        29686   ; IS POINTED BY (H,L)+1 AND BUILDS A DESCRIPTOR FOR IT,
                                        29688   ; THE DESCRIPTOR IS INITIALLY BUILT IN DSCTMP, BUT PUTNEW
                                        29690   ; TRANSFERS IT INTO A TEMPORARY AND LEAVES A POINTER
                                        29692   ; AT THE TEMPORARY IN FACLO, THE CHARACTERS OTHER THAN
                                        29694   ; ZERO THAT TERMINATE THE STRING SHOULD BE SET UP IN (B)
                                        29696   ; AND (D), IT THE TERMINATOR IS A QUOTE, THE QUOTE IS SKIPPED
                                        29697   ; OVER, LEADING QUOTES SHOULD BE SKIPPED BEFORE CALL, ON RETURN
                                        29698   ; THE CHARACTER AFTER THE STRING LITERAL IS POINTED TO
                                        29700   ; BY (H,L) AND IS IN (A), BUT THE CONDITION CODES ARE
                                        29702   ; NOT SET UP.
                                        29704   ;
007637' 001000  000053                  29720   STRLIT: DCX     H
007640' 001000  000006                  29740   STRLTI: MVI     B,34            ;ASSUME STR ENDS ON QUOTE
007641' 000000  000042
007642' 001000  000120                  29760   STRLT3: MOV     D,B
007643' 001000  000345                  29780   STRLT2: PUSH    H               ;SAVE POINTER TO START OF LITERAL
007644' 001000  000016                  29800           MVI     C,255           ;INITIALIZE CHARACTER COUNT
007645' 000000  000377
007646' 001000  000043                  29820   STRGET: INX     H
007647' 001000  000176                  29840           MOV     A,M             ;GET CHAR
007650' 001000  000014                  29860           INR     C               ;BUMP CHARACTER COUNT
007651' 001000  000267                  29880           ORA     A               ;IF 0, (END OF LINE) DONE
007652' 001000  000312                  29900           JZ      STRFIN          ;TEST
007653' 000000  007665'
007654' 000000  007635'
007655' 001000  000272                  29920           CMP     D
007656' 001000  000312                  29940           JZ      STRFIN
007657' 000000  007665'
007660' 000000  007653'
007661' 001000  000270                  29960           CMP     B               ;CLOSING QUOTE
007662' 001000  000302                  29980           JNZ     STRGET          ;NO, GO BACK FOR MORE
007663' 000000  007646'
007664' 000000  007657'
007665' 001000  000376                  30000   STRFIN: CPI     34              ;IF QUOTE TERMINATES THE STRING
                                        30020           CZ      CHRGTR          ;SKIP OVER THE QUOTE
007666' 000000  000042
007667' 001000  000314
007670' 000000  003426'
```

```
6558  007671' 000000  007663'
6559  007672' 001000  000343          30040           XTHL
6560  007673' 001000  000043          30060           INX     H
6561  007674' 001000  000353          30080           XCHG                    ;GET POINTER TO TEMP
6562  007675' 001000  000171          30100           MOV     A,C             ;GET CHARACTER COUNT IN A
6563  007676' 001000  000315          30120           CALL    STRAD2          ;SAVE STR INFO
6564  007677' 000000  007627'
6565  007700' 000000  007670'
6566                                   30140                                   ;COMPARE DESC. POINTER & STR POINTER
6567  007701' 001000  000347          30160           COMPAR                  ;TO SEE IF STR IN BUF
6568  007702' 001000  000324          30180           CNC     STRCPY          ;IF IN BUF, MUST COPY IT.
6569  007703' 000000  007601'
6570  007704' 000000  007677'
6571                                   30200                                   ;OTHERWISE STR IN PROGRAM.
6572                                   30220   ;
6573                                   30240   ; SOME STRING FUNCTION IS RETURNING A RESULT IN DSCTMP
6574                                   30260   ; WE WANT TO SETUP A TEMP DESCRIPTOR WITH DCSTMP IN IT
6575                                   30280   ; PUT A POINTER TO THE DESCRIPTOR IN FACLO AND FLAG THE
6576                                   30300   ; RESULT AS TYPE STRING
6577                                   30320   ;
6578  007705' 001000  000021          30340   PUTNEW: LXI     D,DSCTMP        ;[D,E] POINT AT RESULT DESCRIPTOR
6579  007706' 000000  001570'
6580  007707' 000000  007703'
6581  007710' 001000  000052          30360           LHLD    TEMPPT          ;[H,L]=POINTER TO FIRST FREE TEMP
6582  007711' 000000  001547'
6583  007712' 000000  007706'
6584  007713' 001000  00004E          30380           SHLD    FACLO           ;POINTER AT WHERE RESULT DESCRIPTOR WILL BE
6585  007714' 000000  001637'
6586  007715' 000000  007711'
6587                                   30400   IFE     LENGTH-2,<
6588  007716' 001000  000076          30420           MVI     A,3
6589  007717' 000000  000003
6590  007720' 001000  000062          30440           STA     VALTYP          ;FLAG THIS AS A STRING
6591  007721' 000000  001543'
6592  007722' 000000  007714'
6593  007723' 001000  000315          30460           CALL    VMOVE>          ;AND MOVE THE VALUE INTO A TEMPORARY
6594  007724' 000000  004233*
6595  007725' 000000  007721'
6596                                   30480   IFN     LENGTH-2,<
6597                                   30500           MVI     A,1             ;FLAG RESULT AS STRING
6598                                   30520           STA     VALTYP          ;STORE IN "VALUE-TYPE" LOCATION
6599                                   30540           CALL    MOVE>           ;TRANSFER THE DESCRIPTOR IN
6600  007726' 001000  000347          30560           COMPAR                  ;DSCTMP IS JUST BEYOND THE TEMPS
6601                                   30580                                   ;AND IF TEMPPT POINTS AT IT THERE
6602                                   30600                                   ;ARE NO FREE TEMPS
6603  007727' 001000  000036          30620           MVI     E,ERRST         ;"STRING TEMPORARY" ERROR
6604  007730' 000000  000020
6605  007731' 001000  000312          30640           JZ      ERROR           ;GO TELL HIM
6606  007732' 000000  002102'
6607  007735' 000000  007724'
6608  007734' 001000  00004E          30660           SHLD    TEMPPT          ;SAVE NEW TEMPORARY POINTER
6609  007735' 000000  001547'
6610  007736' 000000  007732'
```



```
6611  007737' 001000  000341          30680           POP     H               ;GET THE TEXT POINTER
6612  007740' 001000  000176          30700           MOV     A,M             ;GET CURRENT CHARACTER INTO [A]
6613  007741' 001000  000311          30720           RET
6614                                   30740   ;
6615                                   30760   ; PRINT THE STRING POINTED TO BY [H,L] WHICH ENDS WITH A ZERO
6616                                   30780   ; IF THE STRING IS BELOW DSCTMP IT WILL BE COPIED INTO STRING SPACE
6617                                   30800   ;
6618  007742' 001000  000043          30820   STROUI: INX     H               ;POINT AT NEXT CHARACTER
6619  007745' 001000  000315          30840   STROUT: CALL    STRLIT          ;GET A STRING LITERAL
6620  007744' 000000  007637'
6621  007745' 000000  007735'
6622                                   30860   ;
6623                                   30880   ; PRINT THE STRING WHOSE DESCRIPTOR IS POINTED TO BY FACLO.
6624                                   30900   ;
6625  007746' 001000  000315          30920   STRPRT: CALL    FREFAC          ;RETURN TEMP POINTER BY FACLO
6626  007747' 000000  010434'
6627  007750' 000000  007744'
6628                                   30940   IFE     LENGTH-2,<
6629  007751' 001000  000315          30960           CALL    GETBCD          ;[D]=LENGTH [B,C]=POINTER AT DATA
6630  007752' 000000  000000*
6631  007753' 000000  007747'
6632  007754' 001000  000024          30970           INR     D               ;INCREMENT AND DECREMENT EARLY
6633                                   30980                                   ;TO CHECK FOR NULL STRING
6634  007755' 001000  000025          30990   STRPR2: DCR     D>              ;DECREMENT THE LENGTH
6635                                   31060   IFN     LENGTH-2,<
6636                                   31080           CALL    MOVRM           ;GET LENGTH IN [E]
6637                                   31100           INR     E               ;AND POINTER TO CHARACTERS IN [B,C]
6638                                   31120   STRPR2: DCR     E>
6639  007756' 001000  000310          31140           RZ                      ;ALL DONE
6640  007757' 001000  000012          31160           LDAX    B               ;GET CHARACTER TO PRINT
6641  007760' 001000  000337          31180           OUTCHR
6642  007761' 001000  000376          31200           CPI     CR
6643  007762' 000000  000015
6644  007763' 001000  000314          31220           CZ      CRFIN
6645  007764' 000000  004534'
6646  007765' 000000  007752'
6647  007766' 001000  000003          31240           INX     B               ;POINT TO THE NEXT CHARACTER
6648  007767' 001000  000303          31260           JMP     STRPR2          ;AND PRINT IT...
6649  007770' 000000  007755'
6650  007771' 000000  007764'
6651                                   31280   ;
6652                                   31300   ; GETSPA = GET SPACE FOR CHARACTER STRING
6653                                   31320   ; MAY FORCE GARBAGE COLLECTION.
6654                                   31340   ;
6655                                   31360   ; # OF CHARS (BYTES) IN [A]
6656                                   31380   ; RETURNS WITH POINTER IN [D,E] OTHERWISE IF CANT GET SPACE
6657                                   31400   ; BLOWS OFF TO "OUT OF STRING SPACE" TYPE ERROR.
6658                                   31420   ;
6659  007772' 001000  000267          31440   GETSPA: ORA     A               ;MUST BE NON ZERO, SIGNAL NO GARBAG YET
6660  007773' 001000  000016          31460           XRD     "01000,"0016    ;"MVI C" AROUND THE NEXT
6661  007774' 001000  000361          31480   TRYGI2: POP     PSW             ;IN CASE COLLECTED WHAT WAS LENGTH?
6662  007775' 001000  000365          31500           PUSH    PSW             ;SAVE IT BACK
6663  007776' 001000  000052          31520           LHLD    STKTOP          ;GET BOTTOM OF STRING SPACE
```

```
 6664  007777' 000000  001615'
 6665  010000' 000000  007770'
 6666  010001' 001000  000353          31540            XCHG                    ;IN (D,E)
 6667  010002' 001000  000052          31560            LHLD    FRETOP          ;GET TOP OF FREE SPACE IN (H,L)
 6668  010005' 000000  001573'
 6669  010004' 000000  007777'
 6670  010005' 001000  000057          31580            CMA                     ;-# OF CHARS
 6671  010006' 001000  000117          31600            MOV     C,A             ;IN (B,C)
 6672  010007' 001000  000006          31620            MVI     B,255
 6673  010010' 000000  000377
 6674  010011' 001000  000011          31640            DAD     B               ;SUBTRACT FROM TOP OF FREE
 6675  010012' 001000  000043          31660            INX     H
 6676  010013' 001000  000347          31680            COMPAR                  ;COMPARE THE TWO
 6677  010014' 001000  000332          31700            JC      GARBAG          ;NOT ENOUGH ROOM FOR STRING, OFFAL TIME
 6678  010015' 000000  010026'
 6679  010016' 000000  010003'
 6680  010017' 001000  000042          31720            SHLD    FRETOP          ;SAVE NEW BOTTOM OF MEMORY
 6681  010020' 000000  001573'
 6682  010021' 000000  010015'
 6683  010022' 001000  000043          31740            INX     H               ;MOVE BACK TO POINT TO STRING
 6684  010023' 001000  000353          31760            XCHG                    ;RETURN WITH POINTER IN (D,E)
 6685  010024' 001000  000361          31780  PPSWRT:   POP     PSW             ;GET CHARACTER COUNT
 6686  010025' 001000  000311          31800            RET                     ;RETURN FROM GETSPA
 6687
 6688  010026' 001000  000361          31840  GARBAG:   POP     PSW             ;HAVE WE COLLECTED BEFORE?
 6689  010027' 001000  000036          31860            MVI     E,ERRSO         ;GET READY FOR OUT OF STRING SPACE ERROR
 6690  010030' 000000  000016
 6691  010031' 001000  000312          31880            JZ      ERROR           ;GO TELL USER HE LOST
 6692  010032' 000000  002102'
 6693  010033' 000000  010020'
 6694  010034' 001000  000277          31900            CMP     A               ;SET ZERO FLAG TO SAY WEVE GARBAGED
 6695  010035' 001000  000365          31920            PUSH    PSW             ;SAVE FLAG BACK ON STACK
 6696  010036' 001000  000001          31940            LXI     B,TRYG12        ;PLACE FOR GARBAG TO RETURN TO,
 6697  010037' 000000  007774'
 6698  010040' 000000  010032'
 6699  010041' 001000  000305          31960            PUSH    B               ;SAVE ON STACK
 6700  010042' 001000  000052          31980  GARBA2:   LHLD    MEMSIZ          ;START FROM TOP DOWN
 6701  010043' 000000  001545'
 6702  010044' 000000  010037'
 6703                                   32000  IFE       REALIO,<
 6704                                   32020            MVI     A,7             ;RING THE BELL ON GARBAGE COLLECTION
 6705                                   32040            OUTCHR>
 6706  010045' 001000  000042          32060  FNDVAR:   SHLD    FRETOP          ;LIKE SO
 6707  010046' 000000  001573'
 6708  010047' 000000  010043'
 6709  010050' 001000  000041          32080            LXI     H,$CODE         ;GET DOUBLE ZERO
 6710  010051' 000000  000000'
 6711  010052' 000000  010046'
 6712  010053' 001000  000345          32100            PUSH    H               ;SAY DIDNT SEE VARS THIS PASS
 6713  010054' 001000  000052          32120            LHLD    STKTOP          ;FORCE DVARS TO IGNORE STRINGS
 6714  010055' 000000  001615'
 6715  010056' 000000  010051'
 6716                                   32140                                    ;IN THE PROGRAM TEXT (LITERALS, DATA)
```



```
 6717  010057' 001000  000345          32160            PUSH    H               ;FORCE FIND HIGH ADDRESS
 6718  010060' 001000  000041          32180            LXI     H,TEMPST        ;GET START OF STRING TEMPS
 6719  010061' 000000  001551'
 6720  010062' 000000  010055'
 6721  010063' 001000  000353          32200  TVAR:     XCHG                    ;SAVE IN (D,E)
 6722  010064' 001000  000052          32220            LHLD    TEMPPT          ;SEE IF DONE
 6723  010065' 000000  001547'
 6724  010066' 000000  010061'
 6725  010067' 001000  000353          32240            XCHG                    ;FLIP
 6726  010070' 001000  000347          32260            COMPAR                  ;TEST
 6727  010071' 001000  000001          32280            LXI     B,TVAR          ;FORCE JUMP TO TVAR
 6728  010072' 000000  010063'
 6729  010073' 000000  010065'
 6730  010074' 001000  000302          32300            JNZ     DVAR2           ;DO TEMP VAR GARBAGE COLLECT
 6731  010075' 000000  010213'
 6732  010076' 000000  010072'
 6733
 6734  010077' 001000  000052          32340  SVARS:    LHLD    VARTAB          ;GET START OF SIMPLE VARIABLES
 6735  010100' 000000  001621'
 6736  010101' 000000  010075'
 6737  010102' 001000  000353          32360  SVAR:     XCHG                    ;GET IN (D,E)
 6738  010103' 001000  000052          32380            LHLD    ARYTAB          ;GET END OF SIMPS
 6739  010104' 000000  001623'
 6740  010105' 000000  010100'
 6741  010106' 001000  000353          32400            XCHG                    ;FLIP
 6742  010107' 001000  000347          32420            COMPAR                  ;SEE IF AT END OF SIMPS
 6743  010110' 001000  000312          32440            JZ      ARYVAR          ;IF YES, DO ARRAY TYPE STRINGS
 6744  010111' 000000  010140'
 6745  010112' 000000  010104'
 6746  010113' 001000  000176          32460            MOV     A,M             ;GET 2ND CHARACTER OF VARIABLE
 6747  010114' 001000  000043          32480            INX     H               ;BUMP POINTER TWICE
 6748  010115' 001000  000043          32500            INX     H               ;
 6749                                   32520  IFE       LENGTH-2,<
 6750  010116' 001000  000043          32540            INX     H               ;POINT AT THE VALUE
 6751  010117' 001000  000376          32560            CPI     3               ;SEE IF ITS A STRING
 6752  010120' 000000  000003
 6753  010121' 001000  000302          32580            JNZ     SKPVAR          ;IF NOT, JUST SKIP AROUND IT
 6754  010122' 000000  010130'
 6755  010123' 000000  010111'
 6756  010124' 001000  000315          32600            CALL    DVARS           ;COLLECT IT
 6757  010125' 000000  010214'
 6758  010126' 000000  010122'
 6759  010127' 001000  000257          32620            XRA     A               ;AND DON'T SKIP ANYTHING MORE
 6760  010130' 001000  000137          32640  SKPVAR:   MOV     E,A
 6761  010131' 001000  000026          32660            MVI     D,0             ;(D,E)=AMOUNT TO SKIP
 6762  010132' 000000  000000
 6763  010133' 001000  000031          32680            DAD     D>
 6764                                   32700  IFN       LENGTH-2,<
 6765                                   32720            ORA     A               ;SET CC'S
 6766                                   32740            CALL    DVARS>          ;CALL THE VARIABLE GARB ROUT,
 6767  010134' 001000  000303          32760            JMP     SVAR            ;GET NEXT ONE
 6768  010135' 000000  010102'
 6769  010136' 000000  010125'
```

```
                                                33000   IFN     LENGTH=2,<
                                                33020           MOV     A,E>            ;GET 2ND CHAR OF VAR NAME IN A
                                                33040           PUSH    H               ;SAVE POINTER TO DIMS
                                                33060           DAD     B               ;ADD TO CURRENT POINTER POSITION
                                                33080   IFE     LENGTH=2,<
                                                33100           CPI     $               ;SEE IF ITS A STRING
                                                33120           JNZ     ARYVA2>         ;IF NOT JUST SKIP IT
                                                33140   IFN     LENGTH=2,<
                                                33160           ORA     A               ;SEE IF STRING VAR
                                                33180           JP      ARYVA2>         ;NO, KEEP ON TRUCKIN
                                                33200           SHLD    TEMP3           ;SAVE END OF ARRAY
                                                33220           POP     H               ;GET BACK CURRENT POSITION
                                                33240           MOV     C,M             ;PICK UP NUMBER OF DIMS
                                                33260           MVI     B,0             ;MAKE DOUBLE WITH HIGH ZERO
                                                33280           DAD     B               ;GO PAST DIMS
                                                33300           DAD     B               ;BY ADDING ON TWICE #DIMS (2 BYTE GUYS)
                                                33320           INX     H               ;ONE MORE TO ACCOUNT FOR #DIMS,
                                                33340   ARYSTR: XCHG                    ;SAVE CURRENT POSIT IN (D,E)
                                                33360           LHLD    TEMP3           ;GET END OF ARRAY
                                                33380           XCHG                    ;FIX (H,L) BACK TO CURRENT
                                                33400           COMPAR                  ;SEE IF AT END OF ARRAY
                                                33420           JZ      ARYVAR          ;END OF ARRAY, TRY NEXT ARRAY
                                                33440           LXI     B,ARYSTR        ;ADDR OF WHERE TO RETURN TO
                                                33460   DVAR2:  PUSH    B               ;GOES ON STACK
```



```
                                                33480   IFE     LENGTH=2,<
                                                33500   DVAR:
                                                33520   DVARS:  XRA     A
                                                33540           ORA     M               ;SEE IF ITS THE NULL STRING
                                                33560           INX     H
                                                33580           MOV     E,M
                                                33600           INX     H
                                                33620           MOV     D,M
                                                33640           INX     H>              ;(D,E)=POINTER AT THE VALUE
                                                33660   IFN     LENGTH=2,<
                                                33680   DVAR:   ORI     128             ;FORCE DVAR TO CALL GRBVAR
                                                33700   DVARS:  PUSHM                   ;SAVE LENGTH
                                                33720           PUSHM                   ;SKIP NEXT TWO BYTES
                                                33740           POP     D               ;GET POINTER IN (D,E)
                                                33760           POP     B               ;POP OF STRING LENGTH
                                                33780           RP                      ;IF WASNT A STR, RETURN
                                                33800           MOV     A,C             ;GET LENGTH OF STRING
                                                33820           ORA     A>              ;SET CONDITION CODES
                                                33840           RZ                      ;NULL STRING, RETURN
                                                33860           MOV     B,H             ;MOVE (H,L) TO (B,C)
                                                33880           MOV     C,L
                                                33900           LHLD    FRETOP          ;GET POINTER TO TOP OF STRING FREE SPACE
                                                33920           COMPAR                  ;IS THIS STRINGS POINTER ,LT, FRETOP
                                                33940           MOV     H,B             ;MOVE (B,C) BACK TO (H,L)
                                                33960           MOV     L,C
                                                33980           RC                      ;IF NOT, NO NEED TO MESS WITH IT FURTHUR
                                                34000           POP     H               ;GET RETURN ADDRESS OFF STACK
                                                34020           XTHL                    ;GET MAX SEEN SO FAR & SAVE RETURN ADDRESS
                                                34040           COMPAR                  ;LETS SEE
                                                34060           XTHL                    ;SAVE MAX SEEN & GET RETURN ADDRESS OFF STACK
                                                34080           PUSH    H               ;SAVE RETURN ADDRESS BACK
                                                34100           MOV     H,B             ;MOVE (B,C) BACK TO (H,L)
                                                34120           MOV     L,C
                                                34140           RNC                     ;IF NOT, LETS LOOK AT NEXT VAR
                                                34160           POP     B               ;GET RETURN ADDR OFF STACK
                                                34180           POP     PSW             ;POP OFF MAX SEEN
                                                34200           POP     PSW             ;AND VARIABLE POINTER
                                                34220           PUSH    H               ;SAVE NEW VARIABLE POINTER
                                                34240           PUSH    D               ;AND NEW MAX POINTER
                                                34260           PUSH    B               ;SAVE RETURN ADDRESS BACK
                                                34280           RET                     ;AND RETURN
                                                34300   ;
                                                34320   ; HERE WHEN MADE ONE COMPLETE PASS THRU STRING VARS
                                                34340   ;
                                                34360   GRBPAS: POP     D               ;POP OFF MAX POINTER
                                                34380           POP     H               ;AND GET VARIABLE PUINTER
                                                34400           MOV     A,L             ;GET LOW IN
                                                34420           ORA     M               ;SEE IF ZERO POINTER
                                                34440           RZ                      ;IF END OF COLLECTION,
                                                34460                                   ;THEN MAYBE RETURN TO GETSPA
                                                34480           DCX     H               ;CURRENTLY JUST PAST THE DESCRIPTOR
```

```
6876  010262' 001000  000106          34500          MOV     B,M             ;[B]=HIGH BYTE OF DATA POINTER
6877  010263' 001000  000053          34520          DCX     H
6878  010264' 001000  000116          34540          MOV     C,M             ;[B,C]=POINTER AT STRING DATA
6879  010265' 001000  000345          34550          PUSH    H               ;SAVE THIS LOCATION SO THE POINTER
6880                                   34552                                  ;CAN BE UPDATED AFTER THE STRING IS
6881                                   34554                                  ;MOVED
6882  010266' 001000  000053          34560          DCX     H
6883                                   34580  IFN     LENGTH=2,<
6884                                   34600          DCX     H>
6885  010267' 001000  000156          34620          MOV     L,M             ;[L]=STRING LENGTH
6886  010270' 001000  000046          34640          MVI     H,0             ;[H,L] GET CHARACTER COUNT
6887  010271' 000000  000000
6888  010272' 001000  000011          34660          DAD     B               ;[H,L]=POINTER BEYOND STRING
6889  010273' 001000  000120          34680          MOV     D,B
6890  010274' 001000  000131          34700          MOV     E,C             ;[D,E]=ORIGINAL POINTER
6891  010275' 001000  000053          34720          DCX     H               ;DON'T MOVE ONE BEYOND STRING
6892  010276' 001000  000104          34740          MOV     B,H             ;GET TOP OF STRING IN [B,C]
6893  010277' 001000  000115          34760          MOV     C,L
6894  010300' 001000  000052          34780          LHLD    FRETOP          ;GET TOP OF FREE SPACE
6895  010301' 000000  001575'
6896  010302' 000000  010227'
6897  010303' 001000  000315          34800          CALL    BLTUC           ;MOVE STRING
6898  010304' 000000  002010'
6899  010305' 000000  010301'
6900  010306' 001000  000341          34820          POP     H               ;GET BACK POINTER TO DESC,
6901  010307' 001000  000161          34840          MOV     M,C             ;SAVE FIXED ADDR
6902  010310' 001000  000043          34860          INX     H               ;MOVE POINTER
6903  010311' 001000  000160          34880          MOV     M,B             ;HIGH PART
6904  010312' 001000  000151          34900          MOV     L,C
6905  010313' 001000  000140          34920          MOV     H,B             ;[H,L]=NEW POINTER
6906  010314' 001000  000053          34940          DCX     H               ;FIX UP FRETOP
6907  010315' 001000  000303          34960          JMP     FNOVAR          ;AND TRY TO FIND HIGH AGAIN
6908  010316' 000000  010045'
6909  010317' 000000  010304'
6910
6911                                   35000  ;
6912                                   35020  ; THE FOLLOWING ROUTINE CONCATENATES TWO STRINGS
6913                                   35040  ; THE FACLO CONTAINS THE FIRST ONE AT THIS POINT,
6914                                   35060  ; [H,L] POINTS BEYOND THE + SIGN AFTER IT
6915                                   35080  ;
6916  010320' 001000  000305          35100  CAT:    PUSH    B               ;PUT OLD PRECEDENCE BACK ON
6917  010321' 001000  000345          35120          PUSH    H               ;SAVE TEXT POINTER
6918  010322' 001000  000052          35140          LHLD    FACLO           ;GET POINTER TO STRING DESC,
6919  010323' 000000  001637'
6920  010324' 000000  010316'
6921  010325' 001000  000343          35160          XTHL                    ;SAVE ON STACK & GET TEXT POINTER BACK
6922  010326' 001000  000315          35180          CALL    EVAL            ;EVALUATE REST OF FORMULA
6923  010327' 000000  006661'
6924  010330' 000000  010323'
6925  010331' 001000  000343          35200          XTHL                    ;SAVE TEXT POINTER, GET BACK DESC,
6926  010332' 001000  000315          35220          CALL    CHKSTR
6927  010333' 000000  006630*
6928  010334' 000000  010327'
```



```
6929  010335' 001000  000176          35240          MOV     A,M
6930  010336' 001000  000345          35260          PUSH    H               ;SAVE DESC, POINTER,
6931  010337' 001000  000052          35280          LHLD    FACLO           ;GET POINTER TO 2ND DESC,
6932  010340' 000000  001637'
6933  010341' 000000  010333'
6934  010342' 001000  000345          35300          PUSH    H               ;SAVE IT
6935  010343' 001000  000206          35320          ADD     M               ;ADD TWO LENGTHS TOGETHER
6936  010344' 001000  000036          35340          MVI     E,ERRLS         ;SEE IF RESULT ,LT, 256
6937  010345' 000000  000017
6938  010346' 001000  000332          35360          JC      ERROR           ;ERROR "LONG STRING"
6939  010347' 000000  002102'
6940  010350' 000000  010340'
6941  010351' 001000  000315          35380          CALL    STRINI          ;GET INITIAL STRING
6942  010352' 000000  007624'
6943  010353' 000000  010347'
6944  010354' 001000  000321          35400          POP     D               ;GET 2ND DESC,
6945  010355' 001000  000315          35420          CALL    FRETMP
6946  010356' 000000  010440'
6947  010357' 000000  010352'
6948  010360' 001000  000343          35440          XTHL                    ;SAVE POINTER TO IT
6949  010361' 001000  000315          35460          CALL    FRETM2          ;FREE UP 1ST TEMP
6950  010362' 000000  010437'
6951  010363' 000000  010356'
6952  010364' 001000  000345          35480          PUSH    H               ;SAVE DESC, POINTER (FIRST)
6953  010365' 001000  000052          35500          LHLD    DSCTMP+2        ;GET POINTER TO FIRST
6954  010366' 000000  001572'
6955  010367' 000000  010362'
6956  010370' 001000  000353          35520          XCHG                    ;IN [D,E]
6957  010371' 001000  000315          35540          CALL    MOVINS          ;MOVE IN THE FIRST STRING
6958  010372' 000000  010407'
6959  010373' 000000  010366'
6960  010374' 001000  000315          35560          CALL    MOVINS          ;AND THE SECOND
6961  010375' 000000  010407'
6962  010376' 000000  010372'
6963  010377' 001000  000041          35580          LXI     H,TSTOP         ;CAT REENTERS FORMULA EVALUATION AT TSTOP
6964  010400' 000000  005357'
6965  010401' 000000  010375'
6966  010402' 001000  000343          35600          XTHL
6967  010403' 001000  000345          35620          PUSH    H               ;TEXT POINTER OFF FIRST
6968  010404' 001000  000303          35640          JMP     PUTNEW          ;THEN RETURN ADDRESS OF TSTOP
6969  010405' 000000  007705'
6970  010406' 000000  010400'
6971
6972
6973  010407' 001000  000341          35700  MOVINS: POP     H               ;GET RETURN ADDR
6974  010410' 001000  000343          35720          XTHL                    ;PUT BACK, BUT GET DESC,
6975                                   35740  IFE     LENGTH=2,<
6976  010411' 001000  000176          35760          MOV     A,M             ;[A]=STRING LENGTH
6977  010412' 001000  000043          35780          INX     H
6978  010413' 001000  000116          35800          MOV     C,M             ;[B,C]=POINTER AT STRING DATA
6979  010414' 001000  000043          35820          INX     H
6980  010415' 001000  000106          35840          MOV     B,M
6981  010416' 001000  000157          35860          MOV     L,A>            ;[L]=STRING LENGTH
```

```
                                        35880   IFN     LENGTH=2,<
                                        35900           PUSHM                   ;GET LENGTH ON STACK
                                        35920           PUSHM                   ;AND POINTER
                                        35940           POP     B               ;TEXT POINTER HERE
                                        35960           POP     H>              ;CHARACTER COUNT HERE
010417' 001000  000054                  35980   MOVSTR: INR     L
010420' 001000  000055                  36000   MOVLP:  DCR     L               ;SET CC'S
010421' 001000  000310                  36020           RZ                      ;0, NO BYTE TO MOVE
010422' 001000  000012                  36040           LDAX    B               ;GET CHAR
010423' 001000  000022                  36060           STAX    D               ;SAVE IT
010424' 001000  000003                  36080           INX     B               ;MOVE POINTERS
010425' 001000  000023                  36100           INX     D
010426' 001000  000303                  36120           JMP     MOVLP           ;KEEP DOING IT
010427' 000000  010420'
010430' 000000  010405'
                                        36140   ;
                                        36160   ; FRETMP IS PASSED A POINTER TO A STRING DESCRIPTOR IN (D,E)
                                        36180   ; THIS VALUE IS RETURNED IN (H,L), ALL THE OTHER REGISTERS ARE MODIFIED,
                                        36200   ; A CHECK TO IS MADE TO SEE IF THE STRING DESCRIPTOR (D,E) POINTS
                                        36220   ; TO IS THE LAST TEMPORARY DESCRIPTOR ALLOCATED BY PUTNEW,
                                        36240   ; IF SO, THE TEMPORARY IS FREED UP BY THE UPDATING OF TEMPPT,
                                        36260   ; IF A TEMPORARY IS FREED UP, A FURTHER CHECK IS MADE TO SEE IF THE
                                        36280   ; STRING DATA THAT THAT STRING TEMPORARY POINTED TO IS THE
                                        36300   ; THE LOWEST PART OF STRING SPACE IN USE,
                                        36320   ; IF SO, FRETMP IS UPDATED TO REFLECT THE FACT THAT THAT SPACE IS NO
                                        36340   ; LONGER IS USE, THIS CAUSES DIFFICULTY FOR ASSIGNMENT ("LET") BECAUSE
                                        36360   ; THOUGH A TEMPORARY IS BEING FREED UP, NAMELY THE VALUE TO THE RIGHT
                                        36380   ; OF THE EQUAL SIGN IN THE "LET", THE ACTUAL DATA
                                        36400   ; IS STILL ACTIVE DATA SINCE A VARIABLE IS BEING SET UP TO POINT
                                        36420   ; AT IT, "LET" FOOLS FRETMP BY SETTING THE LENGTH OF THE
                                        36440   ; TEMPORARY TO ZERO TEMPORARILY,
                                        36460   ;
010431' 001000  000315                  36480   FRESTR: CALL    CHKSTR          ;MAKE SURE ITS A STRING
010432' 000000  010333*
010433' 000000  010427'
010434' 001000  000052                  36500   FREFAC: LHLD    FACLO
010435' 000000  001637'
010436' 000000  010432'
010437' 001000  000353                  36520   FRETM2: XCHG                    ;FREE UP THE TEMP IN THE FACLO
010440' 001000  000052                  36540   FRETMP: LHLD    TEMPPT          ;GET TEMP POINTER
010441' 000000  001547'
010442' 000000  010435'
010443' 001000  000053                  36560           DCX     H               ;LOOK AT WHAT IS IN THE LAST TEMP
010444' 001000  000106                  36580           MOV     B,M             ;(B,C)=POINTER AT STRING
010445' 001000  000053                  36600           DCX     H               ;DECREMENT TEMPPT BY STRSIZ
010446' 001000  000116                  36620           MOV     C,M
010447' 001000  000053                  36640           DCX     H
                                        36660   IFN     LENGTH=2,<
                                        36680           DCX     H>
010450' 001000  000347                  36700           COMPAR                  ;SEE IF (D,E) POINT AT THE LAST
010451' 001000  000353                  36720           XCHG                    ;RETURN WITH (H,L)
                                        36740                                   ;POINTING AT CURRENT DESCRIPTOR
010452' 001000  000300                  36760           RNZ                     ;RETURN NOW IF NOW FREEING DONE
```



```
010453' 001000  000042                  36780           SHLD    TEMPPT          ;UPDATE THE TEMP POINTER SINCE
010454' 000000  001547'
010455' 000000  010441'
                                        36800                                   ;ITS BEEN DECREMENTED BY 4
010456' 001000  000325                  36820           PUSH    D               ;SAVE (D,E) TO RETURN IN (H,L)
010457' 001000  000120                  36840           MOV     D,B             ;(D,E)=POINTER AT STRING
010460' 001000  000131                  36860           MOV     E,C
010461' 001000  000033                  36880           DCX     D               ;SUBTRACT ONE
010462' 001000  000116                  36900           MOV     C,M             ;(C)=LENGTH OF THE STRING FREED UP
010463' 001000  000052                  36920           LHLD    FRETOP          ;SEE IF ITS THE FIRST
010464' 000000  001573'
010465' 000000  010454'
                                        36940                                   ;ONE IN STRING SPACE
010466' 001000  000347                  36960           COMPAR
010467' 001000  000302                  36980           JNZ     NOTLST          ;NO SO DON'T ADD
010470' 000000  010477'
010471' 000000  010464'
010472' 001000  000107                  37000           MOV     B,A             ;MAKE (B)=0
010473' 001000  000011                  37020           DAD     B               ;ADD
010474' 001000  000042                  37040           SHLD    FRETOP          ;AND UPDATE FRETOP
010475' 000000  001573'
010476' 000000  010470'
010477' 001000  000341                  37060   NOTLST: POP     H               ;GET POINTER AT CURRENT DESCRIPTOR
010500' 001000  000311                  37080           RET
                                        37100   ;
                                        37120   ; THE FUNCTION LEN($) RETURNS THE LENGTH OF THE
                                        37140   ; STRING PASSED AS AN ARGUMENT
                                        37160   ;
010501' 001000  000001                  37180   LEN:    LXI     B,SNGFLT        ;CALL SNGFLT WHEN DONE
010502' 000000  007400'
010503' 000000  010475'
010504' 001000  000305                  37200           PUSH    B               ;LIKE SO
010505' 001000  000315                  37220   LEN1:   CALL    FRESTR          ;FREE UP TEMP POINTED TO BY FACLO
010506' 000000  010431'
010507' 000000  010502'
010510' 001000  000257                  37240           XRA     A               ;FORCE NUMERIC FLAG
010511' 001000  000127                  37260           MOV     D,A             ;SET HIGH OF (D,E) TO ZERO FOR VAL
                                        37280   IFN     LENGTH=2,<
                                        37300           STA     VALTYP>
010512' 001000  000176                  37320           MOV     A,M
010513' 001000  000267                  37340           ORA     A               ;SET CONDITION CODES ON LENGTH
010514' 001000  000311                  37360           RET                     ;RETURN
                                        37380   ;
                                        37400   ; THE FOLLOWING IS THE ASC($) FUNCTION, IT RETURNS AN INTEGER
                                        37420   ; WHICH IS THE DECIMAL ASCII EQUIVALENT
                                        37440   ;
010515' 001000  000315                  37460   ASC:    CALL    LEN1            ;SET UP ORIGINAL STR
010516' 000000  010585'
010517' 000000  010526'
010520' 001000  000312                  37480           JZ      FCERR           ;NULL STR, BAD ARG,
010521' 000000  010776'
010522' 000000  010516'
010523' 001000  000043                  37500           INX     H               ;BUMP POINTER
```

```
                                        37520  IFN     LENGTH=2,<
                                        37540          INX     H>              ;BUMP POINTER
7090  010524' 001000  000367            37560          PUSHM                   ;GET ADDRESS
7091  010525' 001000  000341            37580          POP     H               ;GET ADDR IN (H,L)
7092  010526' 001000  000176            37600          MOV     A,M             ;GET FIRST CHAR
7093  010527' 001000  000303            37620          JMP     SNGFLT          ;SNGFLT IT
7094  010530' 000000  007400'
7095  010531' 000000  010521'
7096                                    37640  ;
7097                                    37660  ; CHRS(#) CREATES A STRING WHICH CONTAINS AS ITS ONLY
7098                                    37680  ; CHARACTER THE ASCII EQUIVALENT OF THE INTEGER ARG (#)
7099                                    37700  ; WHICH MUST BE ,LT, 255.
7100                                    37720  ;
7101  010532' 001000  000076            37740  CHR$:   MVI     A,1             ;ONE CHARACTER STR
7102  010533' 000000  000001
7103  010534' 001000  000315            37760          CALL    STRIN1          ;GET STRING IN DSCTMP
7104  010535' 000000  007624'
7105  010536' 000000  010530'
7106  010537' 001000  000315            37780          CALL    CONINT          ;GET INTEGER IN RANGE
7107  010540' 000000  011023'
7108  010541' 000000  010535'
7109  010542' 001000  000052            37800          LHLD    DSCTMP+2        ;GET ADDR OF STR
7110  010543' 000000  001572'
7111  010544' 000000  010540'
7112  010545' 001000  000163            37820          MOV     M,E             ;SAVE ASCII BYTE
7113  010546' 001000  000301            37840  FINBCK: POP     B               ;RETURN TO HIGHER LEVEL &
7114                                    37860                                  ;SKIP THE CHKNUM CALL,
7115  010547' 001000  000303            37880          JMP     PUTNEW          ;GO CALL PUTNEW
7116  010550' 000000  007705'
7117  010551' 000000  010543'
7118                                    37900  ;
7119                                    37920  ; THE FOLLOWING IS THE LEFT$(S,#) FUNCTION,
7120                                    37940  ; IT TAKES THE LEFTMOST # CHARS OF THE STR,
7121                                    37960  ; IF # IS ,GT, THAN THE LEN OF THE STR, IT RETURNS THE WHOLE STR,
7122                                    37980  ;
7123  010552' 001000  000315            38000  LEFT$:  CALL    PREAM           ;TEST THE PARAMETERS
7124  010553' 000000  010764'
7125  010554' 000000  010550'
7126  010555' 001000  000257            38020          XRA     A               ;LEFT NEVER CHANGES STRING POINTER
7127  010556' 001000  000343            38040  LEFT3:  XTHL                    ;SAVE TEXT POINTER
7128  010557' 001000  000117            38060          MOV     C,A             ;OFFSET NOW IN (C)
7129  010560' 001000  000345            38080  LEFT2:  PUSH    H               ;SAVE DESC, FOR  FRETMP
7130  010561' 001000  000176            38100          MOV     A,M             ;GET STRING LENGTH
7131  010562' 001000  000270            38120          CMP     B               ;ENTIRE STRING WANTED?
7132  010563' 001000  000332            38140          JC      ALLSTR          ;IF #CHARS ASKED FOR,GE,LENGTH,YES
7133  010564' 000000  010570'
7134  010565' 000000  010553'
7135  010566' 001000  000170            38160          MOV     A,B             ;GET TRUNCATED LENGTH OF STRING
7136  010567' 001000  000021            38180          XWD     ^O1000,^O21     ;SKIP OVER MVI USING "LXI D,"
7137  010570' 001000  000016            38200  ALLSTR: MVI     C,0             ;MAKE OFFSET ZERO
7138  010571' 000000  000000
7139  010572' 001000  000305            38220          PUSH    B               ;SAVE OFFSET ON STACK
7140  010573' 001000  000315            38240          CALL    GETSPA          ;GET SPACE FOR NEW STRING
```



```
7141  010574' 000000  007772'
7142  010575' 000000  010564'
7143  010576' 001000  000301            38260          POP     B               ;GET BACK OFFSET
7144  010577' 001000  000341            38280          POP     H               ;GET BACK DESC POINTER,
7145  010600' 001000  000345            38300          PUSH    H               ;BUT KEEP ON STACK
7146  010601' 001000  000043            38320          INX     H               ;MOVE TO STRING POINTER FIELD
7147                                    38340  IFN     LENGTH=2,<
7148                                    38360          INX     H>
7149  010602' 001000  000106            38380          MOV     B,M             ;GET POINTER LOW
7150  010603' 001000  000043            38400          INX     H               ;
7151  010604' 001000  000146            38420          MOV     H,M             ;POINTER HIGH
7152  010605' 001000  000150            38440          MOV     L,B             ;GET LOW IN  L
7153  010606' 001000  000006            38460          MVI     B,0             ;GET READY TO ADD OFFSET TO POINTER
7154  010607' 000000  000000
7155  010610' 001000  000011            38480          DAD     B               ;ADD  IT
7156  010611' 001000  000104            38500          MOV     B,H             ;GET OFFSET POINTER IN (B,C)
7157  010612' 001000  000115            38520          MOV     C,L
7158  010613' 001000  000315            38540          CALL    STRAD2          ;SAVE INFO IN DSCTMP
7159  010614' 000000  007627'
7160  010615' 000000  010574'
7161  010616' 001000  000157            38560          MOV     L,A             ;GET#  OF CHARS TO  MOVE IN L
7162  010617' 001000  000315            38580          CALL    MOVSTR          ;MOVE THEM IN
7163  010620' 000000  010417'
7164  010621' 000000  010614'
7165  010622' 001000  000321            38600          POP     D               ;GET BACK DESC, POINTER
7166  010623' 001000  000315            38620          CALL    FRETMP          ;FREE IT UP,
7167  010624' 000000  010440'
7168  010625' 000000  010620'
7169  010626' 001000  000303            38640          JMP     PUTNEW          ;PUT TEMP IN TEMP LIST
7170  010627' 000000  007705'
7171  010630' 000000  010624'
7172
7173  010631' 001000  000315            38660  RIGHT$: CALL    PREAM           ;CHECK ARG
7174  010632' 000000  010764'
7175  010633' 000000  010627'
7176  010634' 001000  000321            38700          POP     D               ;GET DESC, POINTER
7177  010635' 001000  000325            38720          PUSH    D               ;SAVE BACK FOR LEFT
7178  010636' 001000  000052            38740          LDAX    D               ;GET PRESENT LEN OF STR
7179  010637' 001000  000220            38760          SUB     B               ;SUBTRACT 2ND PARM
7180  010640' 001000  000303            38780          JMP     LEFT3           ;CONTINUE WITH LEFT CODE
7181  010641' 000000  010556'
7182  010642' 000000  010632'
7183                                    38800  ;
7184                                    38820  ; MID (S,#) RETURNS STR WITH CHARS FROM # POSITION
7185                                    38840  ; ONWARD, IF # IS GT LEN(S) THEN RETURN NULL STRING,
7186                                    38860  ; MID (S,#,#) RETURNS STR WITH CHARS FROM # POSITION
7187                                    38880  ; FOR #2 CHARS, IF #2 GOES PAST END OF STRING, RETURN
7188                                    38900  ; AS MUCH AS POSSIBLE,
7189                                    38920  ;
7190  010643' 001000  000353            38940  MID$:   XCHG                    ;PUT THE TEXT POINTER IN (H,L)
7191  010644' 001000  000176            38960          MOV     A,M             ;GET THE FIRST CHARACTER
7192  010645' 001000  000315            38980          CALL    PREAM2          ;GET OFFSET OFF STACK AND MAKE
7193  010646' 000000  010767'
```

```
7194  010647' 000000  010641'
7195                                  39000                                   ;SURE DOES NOT = 0,
7196  010650' 001000  000305         39020           PUSH    B               ;PUT OFFSET ON TO THE STACK
7197  010651' 001000  000036         39040           MVI     E,255           ;IF TWO ARG GUY, TRUNCATE,
7198  010652' 000000  000377
7199  010653' 001000  000376         39060           CPI     ")"
7200  010654' 000000  000051
7201  010655' 001000  000312         39080           JZ      MID2            ;(E) SAYS USE ALL CHARS
7202  010656' 000000  010665'
7203  010657' 000000  010646'
7204                                  39100                                   ;IF ONE ARGUMENT THIS IS CORRECT
7205  010660' 001000  000317         39120           SYNCHK  44              ;COMMA? MUST DELINEATE 3RD ARG,
7206  010661' 000000  000054
7207  010662' 001000  000315         39140           CALL    GETBYT          ;GET ARGUMENT  IN  (E)
7208  010663' 000000  011020'
7209  010664' 000000  010656'
7210  010665' 001000  000317         39160   MID2:   SYNCHK  ")"             ;MUST BE FOLLOWED BY )
7211  010666' 000000  000051
7212  010667' 001000  000361         39180           POP     PSW             ;GET OFFSET BACK IN A
7213  010670' 001000  000343         39200           XTHL                    ;SAVE TEXT POINTER, GET DESC,
7214  010671' 001000  000001         39220           LXI     B,LEFT2         ;WHERE TO RETURN TO,
7215  010672' 000000  010560'
7216  010673' 000000  010663'
7217  010674' 001000  000305         39240           PUSH    B               ;GOES ON STACK
7218  010675' 001000  000075         39260           DCR     A               ;SUB ONE FROM OFFSET
7219  010676' 001000  000276         39280           CMP     M               ;POINTER PAST END OF STR?
7220  010677' 001000  000006         39300           MVI     B,0             ;ASSUME NULL LENGTH STR
7221  010700' 000000  000000
7222  010701' 001000  000320         39320           RNC                     ;YES, JUST USE NULL STR
7223  010702' 001000  000117         39340           MOV     C,A             ;SAVE OFFSET OF CHARACTER POINTER
7224  010703' 001000  000176         39360           MOV     A,M             ;GET PRESENT LEN OF STR
7225  010704' 001000  000221         39380           SUB     C               ;SUBTRACT INDEX (2ND ARG)
7226  010705' 001000  000273         39400           CMP     E               ;IS IT TRUNCATION
7227  010706' 001000  000107         39420           MOV     B,A             ;GET CALCED LENGTH IN B
7228  010707' 001000  000330         39440           RC                      ;IF NOT USE PARTIAL STR
7229  010710' 001000  000103         39460           MOV     B,E             ;USE TRUNCATED LENGTH
7230  010711' 001000  000311         39480           RET>                    ;RETURN TO LEFT2
7231                                  39500   IFN LENGTH,<
7232                                  39520   ;
7233                                  39540   ; THE FOLLOWING FUNCTIONS ALLOW THE
7234                                  39560   ; USER FULL ACCESS TO THE ALTAIR I/O PORTS
7235                                  39580   ; INP(CHANNEL#) RETURNS AN INTEGER WHICH IS THE STATUS
7236                                  39600   ; OF THE CHANNEL, OUT CHANNEL#,VALUE PUTS OUT THE INTEGER
7237                                  39620   ; VALUE ON CHANNEL #, IT IS A STATEMENT, NOT A FUNCTION,
7238                                  39640   ;
7239  010712' 001000  000315         39660   FNINP:  CALL    CONINT          ;GET INTEGER CHANNEL #
7240  010713' 000000  011023'
7241  010714' 000000  010672'
7242  010715' 001000  000062         39680           STA     INPWRD+1        ;GEN INP INSTR
7243  010716' 000000  010721'
7244  010717' 000000  010713'
7245  010720' 001000  000333         39700   INPWRD: IN      0               ;THE INP INSTR
7246  010721' 000000  000000
```



```
7247  010722' 001000  000303         39720           JMP     SNGFLT          ;SNGFLT RESULT
7248  010723' 000000  007400'
7249  010724' 000000  010716'
7250
7251  010725' 001000  000315         39760   FNOUT:  CALL    SETIO           ;GET READY
7252  010726' 000000  011003'
7253  010727' 000000  010723'
7254  010730' 001000  000323         39780   OUTWRD: OUT     0               ;DO IT
7255  010731' 000000  000000
7256  010732' 001000  000311         39800           RET                     ;AND THATS ALL
7257                                  39820   ;
7258                                  39840   ; THE WAIT CHANNEL#,MASK,MASK2 WAITS UNTIL THE STATUS
7259                                  39860   ; RETURNED BY CHANNEL# IS NON ZERO WHEN XORED WITH MASK2
7260                                  39880   ; AND THEN ANDED WITH MASK, IF MASK2 IS NOT PRESENT IT IS ASSUMED
7261                                  39900   ; TO BE ZERO,
7262                                  39920   ;
7263  010733' 001000  000315         39940   FNWAIT: CALL    SETIO           ;SET UP FOR WAIT
7264  010734' 000000  011003'
7265  010735' 000000  010726'
7266  010736' 001000  000365         39960           PUSH    PSW             ;SAVE THE MASK
7267  010737' 001000  000036         39980           MVI     E,0             ;DEFAULT MASK2 TO ZERO
7268  010740' 000000  000000
7269  010741' 001000  000053         40000           DCX     H
7270  010742' 001000  000327         40020           CHRGET                  ;SEE IF THE STATEMENT ENDED
7271  010743' 001000  000312         40040           JZ      NOTTHR          ;IF NO THIRD ARGUMENT SKIP THIS
7272  010744' 000000  010753'
7273  010745' 000000  010734'
7274  010746' 001000  000317         40060           SYNCHK  44              ;MAKE SURE THERE IS A ","
7275  010747' 000000  000054
7276  010750' 001000  000315         40080           CALL    GETBYT
7277  010751' 000000  011020'
7278  010752' 000000  010744'
7279  010753' 001000  000301         40100   NOTTHR: POP     B               ;REGET THE "AND" MASK
7280  010754' 001000  000333         40120   STAINP: IN      0               ;THE INPUT INSTR
7281  010755' 000000  000000
7282  010756' 001000  000253         40140           XRA     E               ;XOR WITH MASK2
7283  010757' 001000  000240         40160           ANA     B               ;AND WITH MASK
7284  010760' 001000  000312         40180           JZ      STAINP          ;LOOP UNTIL RESULT IS NON-ZERO
7285  010761' 000000  010754'
7286  010762' 000000  010751'
7287                                  40200                                   ;NOTE: THIS LOOP CANNOT BE CONTROL-C'ED
7288                                  40220                                   ;UNLESS THE WAIT IS BEING DONE ON CHANNEL
7289                                  40240                                   ;ZERO, HOWEVER A RESTART AT 0 IS OK,
7290  010763' 001000  000311         40260           RET>
7291                                  40280   IFN     STRING,<
7292                                  40300   ;USED BY RIGHT$ AND LEFT$ FOR PARAMETER CHECKING AND SETUP
7293  010764' 001000  000353         40320   PREAM:  XCHG                    ;PUT THE TEXT POINTER IN (H,L)
7294  010765' 001000  000317         40340           SYNCHK  ")"             ;PARAM LIST SHOULD END
7295  010766' 000000  000051
7296                                  40360   ;USED BY MID$ FOR PARAMETER CHECKING AND SETUP
7297  010767' 001000  000301         40380   PREAM2: POP     B               ;GET RETURN ADDR OFF STACK
7298  010770' 001000  000321         40400           POP     D               ;GET LENGTH OF ARG OFF STACK
7299  010771' 001000  000305         40420           PUSH    B               ;SAVE RETURN ADDR BACK ON
```

```
 7300  010772' 001000  000103          40440                   MOV     B,E             ;SAVE INIT LENGTH
 7301  010773' 001000  000004          40460                   INR     B
 7302  010774' 001000  000005          40480                   DCR     B               ;SEE IF EQUAL TO ZERO
 7303                                   40500           IFE     LENGTH-2,<
 7304  010775' 001000  000300          40520                   RNZ
 7305  010776'                        40536           ILLFUN:
 7306  010776' 001000  000036          40540           FCERR:  MVI     E,ERRFC
 7307  010777' 000000  000005
 7308  011000' 001000  000303          40560                   JMP     ERROR>
 7309  011001' 000000  002102'
 7310  011002' 000000  010761'
 7311                                   40580           IFN     LENGTH-2,<
 7312                                   40600                   JZ      FCERR           ;IT MUST NOT BE 0
 7313                                   40620                   RET>
 7314                                   40640           IFN     LENGTH,<
 7315  011003' 001000  000315          40660           SETIO:  CALL    GETBYT          ;GET INTEGER CHANNEL NUMBER IN [A]
 7316  011004' 000000  011020'
 7317  011005' 000000  011001'
 7318  011006' 001000  000066          40680                   STA     STAINP+1        ;SETUP "WAIT"
 7319  011007' 000000  010755'
 7320  011010' 000000  011004'
 7321  011011' 001000  000062          40700                   STA     OUTWRD+1        ;SETUP "OUT"
 7322  011012' 000000  010731'
 7323  011013' 000000  011007'
 7324  011014' 001000  000317          40720                   SYNCHK  44              ;MAKE SURE THERE IS A COMMA
 7325  011015' 000000  000054
 7326  011016' 001000  000006          40740                   XWD     "0100B,6>       ;"MVI B," AROUND THE CHRGET
 7327                                   40760           IFN     STRING!LENGTH,<
 7328  011017' 001000  000327          40780           GTBYTC: CHRGET
 7329                                   40800           IFE     LENGTH-2,<
 7330  011020' 001000  000315          40820           GETBYT: CALL    FRMEVL
 7331  011021' 000000  005336'
 7332  011022' 000000  011012'
 7333  011023' 001000  000345          40840           CONINT: PUSH    H
 7334  011024' 001000  000315          40860                   CALL    FRCINT
 7335  011025' 000000  006441*
 7336  011026' 000000  011021'
 7337  011027' 001000  000353          40880                   XCHG
 7338  011030' 001000  000341          40900                   POP     H>
 7339                                   40920           IFN     LENGTH-2,<
 7340                                   40940           GETBYT: CALL    FRMNUM          ;READ FORMULA INTO THE FAC,
 7341                                   40960           CONINT: CALL    POSINT>         ;CONVERT THE FAC TO A SINGLE BYTE INTEGER
 7342  011031' 001000  000172          40980                   MOV     A,D             ;SHOULD BE .LT. 255
 7343  011032' 001000  000267          41000                   ORA     A               ;SET CC'S
 7344  011033' 001000  000302          41020                   JNZ     FCERR           ;WASN'T ERROR
 7345  011034' 000000  010776'
 7346  011035' 000000  011025'
 7347  011036' 001000  000053          41040                   DCX     H               ;ACTUALLY FUNCTIONS CAN GET HERE
 7348                                   41060                                           ;WITH BAD [H,L] BUT NOT SERIOUS
 7349                                   41080                                           ;SET CONDITION CODES ON TERMINATOR
 7350  011037' 001000  000327          41100                   CHRGET
 7351  011040' 001000  000173          41120                   MOV     A,E             ;RETURN THE RESULT IN [A] AND [E]
 7352  011041' 001000  000311          41140                   RET>
```



```
 7353
 7354                                   41160   IFN     STRING,<
 7355                                   41200   ;
 7356                                   41202   ; THE VAL FUNCTION TAKES A STRING AND TURN IT INTO
 7357                                   41204   ; A NUMBER BY INTERPRETING THE ASCII DIGITS, ETC.,
 7358                                   41206   ; EXCEPT FOR THE PROBLEM THAT A TERMINATOR MUST BE SUPPLIED
 7359                                   41208   ; BY REPLACING THE CHARACTER BEYOND THE STRING, VAL
 7360                                   41210   ; IS MERELY A CALL TO FLOATING INPUT (FIN),
 7361                                   41212   ;
 7362  011042' 001000  000315          41220   VAL:    CALL    LEN1            ;DO SETUP, SET RESULT=REAL
 7363  011043' 000000  010505'
 7364  011044' 000000  011034'
 7365  011045' 001000  000312          41240           JZ      ZERO            ;RETURN ZERO IF NULL
 7366  011046' 000000  000000*
 7367  011047' 000000  011045'
 7368  011050' 001000  000137          41260           MOV     E,A             ;GET LENGTH OF STR
 7369                                   41270   IFN     LENGTH-2,<
 7370                                   41280           INX     H>              ;THIS IS ALL A KLUDGE
 7371  011051' 001000  000043          41300           INX     H               ;TO HANDLE THE FACT THE IF
 7372  011052' 001000  000367          41320           PUSHM                   ;TWO STRINGS "1" AND "2"
 7373  011053' 001000  000140          41340           MOV     H,B             ;ARE STORED NEXT TO EACH OTHER
 7374  011054' 001000  000151          41360           MOV     L,C             ;AND FIN IS CALLED POINTING TO
 7375  011055' 001000  000031          41380           DAD     D               ;THE FIRST TWELVE WILL BE RETURNED
 7376  011056' 001000  000106          41400           MOV     B,M             ;THE IDEA IS TO STORE 0 IN THE
 7377  011057' 001000  000162          41420           MOV     M,D             ;STRING BEYOND THE ONE VAL
 7378  011060' 001000  000343          41440           XTHL                    ;IS BEING CALLED ON
 7379  011061' 001000  000305          41460           PUSH    B               ;THE FIRST CHARACTER OF THE NEXT STRING
 7380  011062' 001000  000176          41480           MOV     A,M             ;GET FIRST CHARACTER OF ARGUMENT
 7381  011063' 001000  000315          41500           CALL    FIN             ;TURN IT INTO A NUMBER IN THE FAC
 7382  011064' 000000  006103*
 7383  011065' 000000  011046'
 7384  011066' 001000  000301          41520           POP     B               ;GET THE MODIFIED CHARACTER OF THE NEXT
 7385                                   41540                                   ;STRING INTO [B]
 7386  011067' 001000  000341          41560           POP     H               ;GET THE POINTER TO THE MODIFIED CHARACTER
 7387  011070' 001000  000160          41580           MOV     M,B             ;RESTORE THE CHARACTER
 7388                                   41600                                   ;IF STRING IS HIGHEST IN STRING SPACE
 7389                                   41620                                   ;WE ARE MODIFYING [MEMSIZ] AND
 7390                                   41640                                   ;THIS IS WHY [MEMSIZ] CAN'T BE USED TO STORE
 7391                                   41660                                   ;STRING DATA BECAUSE WHAT IF THE
 7392                                   41680                                   ;USER TOOK VAL OFF THAT HIGH STRING
 7393  011071' 001000  000311          41700           RET>
 7394
 7395                                   41740   PAGE
```

```
                                                41760   SUBTTL  FANCY LIST,DELETE,EDIT,LLIST
                                                41780   IFE     LENGTH-2,<
                                                41800   IFN     LPTSW,<
                                                41820   LLIST:  MVI     A,1             ;GET NON ZERO VALUE
                                                41840           STA     PHTFLG>         ;SAVE IN I/O FLAG
7401  011072' 001000  000301                   41860   LIST:   POP     B               ;GET RID OF NEWSTT RETURN ADDR
7402  011073' 001000  000315                   41880           CALL    SCNLIN          ;SCAN LINE RANGE
7403  011074' 000000  002333'
7404  011075' 000000  011064'
7405  011076' 001000  000305                   41900           PUSH    B               ;SAVE POINTER TO 1ST LINE
7406  011077' 001000  000341                   41920   LIST4:  POP     H               ;GET POINTER TO LINE
7407  011100' 001000  000321                   41940           POP     D               ;GET MAX LINE # OFF STACK
7408  011101' 001000  000367                   41960           PUSHM                   ;PUSH LINK
7409  011102' 001000  000170                   41980           MOV     A,B             ;SEE IF END OF CHAIN
7410  011103' 001000  000261                   42000           ORA     C
7411  011104' 001000  000301                   42020           POP     B               ;GET LINK OFF STACK FOR ISCNTC
7412  011105' 001000  000312                   42040           JZ      READY           ;LAST LINE, STOP.
7413  011106' 000000  002145'
7414  011107' 000000  011674'
                                                42060   IFN     LISTEN,<
7416  011110' 001000  000315                   42080           CALL    ISCNTC>         ;CHECK FOR CONTROL-C
7417  011111' 000000  003460'
7418  011112' 000000  011106'
7419  011113' 001000  000305                   42100           PUSH    B               ;SAVE LINK BACK ON
7420  011114' 001000  000367                   42120           PUSHM                   ;PUSH LINE #
7421  011115' 001000  000343                   42140           XTHL                    ;GET LINE # INTO (H,L)
7422  011116' 001000  000353                   42160           XCHG                    ;GET MAX LINE IN (H,L)
7423  011117' 001000  000347                   42180           COMPAR                  ;PAST LAST LINE IN RANGE?
7424  011120' 001000  000301                   42200           POP     B               ;TEXT POINTER TO (B,C)
7425  011121' 001000  000332                   42220           JC      STPRDY          ;IF PAST, THEN DONE LISTING.
7426  011122' 000000  002144'
7427  011123' 000000  011111'
7428  011124' 001000  000343                   42280           XTHL                    ;SAVE MAX ON BOTTOM OF STACK
7429  011125' 001000  000345                   42300           PUSH    H               ;SAVE LINK ON TOP
7430  011126' 001000  000305                   42320           PUSH    B               ;SAVE TEXT POINTER BACK
7431  011127' 001000  000353                   42340           XCHG                    ;GET LINE # IN (H,L)
7432  011130' 001000  000315                   42360           CALL    CRDO            ;DO CRLF TO START OUT
7433  011131' 000000  004523'
7434  011132' 000000  011122'
7435                                            42380                                   ;AND WE WANT (H,L) ON THE STACK
7436  011133' 001000  000315                   42400           CALL    LINPRT          ;PRINT AS INT WITHOUT LEADING SPACE
7437  011134' 000000  003361*
7438  011135' 000000  011131'
7439  011136' 001000  000076                   42420           MVI     A," "
7440  011137' 000000  000040
7441  011140' 001000  000341                   42440           POP     H
7442  011141' 001000  000337                   42460           OUTCHR                  ;PRINT A SPACE AFTER THE LINE #
7443  011142' 001000  000315                   42480           CALL    BUFLIN          ;UNPACK THE LINE INTO BUF
7444  011143' 000000  011163'
7445  011144' 000000  011134'
7446  011145' 001000  000041                   42500           LXI     H,BUF-1         ;POINT AT THE START OF THE UNPACKED CHARACTERS
7447  011146' 000000  001430'
7448  011147' 000000  011143'
```



```
7449  011150' 001000  000006                   42520           MVI     B,0             ;STOP ON ZERO ONLY
7450  011151' 000000  000000
7451  011152' 001000  000315                   42540           CALL    STRLTS          ;LITERALIZE THE LINE STRING
7452  011153' 000000  007642'
7453  011154' 000000  011146'
7454  011155' 001000  000315                   42560           CALL    STRPRT          ;PRINT OUT THE CHARACTERS
7455  011156' 000000  007746'
7456  011157' 000000  011153'
7457  011160' 001000  000305                   42580           JMP     LIST4           ;PRINT ANOTHER LINE
7458  011161' 000000  011077'
7459  011162' 000000  011156'
7460  011163' 001000  000001                   42600   BUFLIN: LXI     B,BUF-1
7461  011164' 000000  001430'
7462  011165' 000000  011161'
7463  011166' 001000  000026                   42620           XWD     ^O1000,^O26     ;"MVI   D," AROUND THE NEXT BYTE
7464  011167' 001000  000341                   42640   PRIT4:  POP     H               ;RESTORE POINTER TO START OF TEXT
7465  011170' 001000  000176                   42660   PLOOP:  MOV     A,M             ;GET A CHARACTER FROM LINE,
7466  011171' 001000  000003                   42680           INX     B               ;ADVANCE STUFF COUNT
7467  011172' 001000  000267                   42700           ORA     A               ;IS IT A RESERVED WORD
7468  011173' 001000  000043                   42720           INX     H               ;INCREMENT POINTER INTO TEXT
7469  011174' 001000  000002                   42740           STAX    B               ;STORE A ZERO IF THE END
7470  011175' 001000  000310                   42760           RZ                      ;ZERO, END OF LINE,
7471  011176' 001000  000362                   42780           JP      PLOOP           ;REGULAR CHAR, JUST PRINT IT
7472  011177' 000000  011170'
7473  011200' 000000  011164'
7474  011201' 001000  000376                   42800           CPI     ELSETK          ;IF ITS "ELSE" DON'T PRINT THE COLON
7475  011202' 000000  000220
7476                                            42820                                   ;IN FRONT OF IT
7477  011203' 001000  000314                   42840           CZ      DCXBRT##        ;BACKUP STUFF COUNT TO ELIMINATE
7478  011204' 000000  000000*
7479  011205' 000000  011177'
7480  011206' 001000  000326                   42860           SUI     127             ;GET RID OF SIGN BIT AND ADD ONE
7481  011207' 000000  000177
7482  011210' 001000  000345                   42880           PUSH    H               ;SAVE CURRENT POSIT
7483  011211' 001000  000021                   42900           LXI     D,RESLST        ;GET RESLST POINTER,
7484  011212' 000000  000172'
7485  011213' 000000  011204'
7486  011214' 001000  000325                   42920   RESRCH: PUSH    D
7487  011215' 001000  000365                   42940           PUSH    PSW             ;SAVE THE RESERVED WORD NUMBER
7488  011216' 001000  000063                   42960   RESCR1: LDAX    D               ;GET CHARACTER FROM RESLST
7489  011217' 001000  000023                   42980           INX     D               ;BUMP RESLST POINTER
7490  011220' 001000  000267                   43000           ORA     A               ;TEST BITS
7491  011221' 001000  000362                   43020           JP      RESCR1          ;NOT AT END OF RESERVED WORD YET
7492  011222' 000000  011216'
7493  011223' 000000  011212'
7494  011224' 001000  000361                   43040           POP     PSW
7495  011225' 001000  000075                   43060           DCR     A               ;DECREMENT CHAR
7496  011226' 001000  000341                   43080           POP     H               ;POP START POINTER HERE
7497  011227' 001000  000362                   43100           JNZ     RESRCH          ;NOT AT END OF RESLST YET,
7498  011230' 000000  011214'
7499  011231' 000000  011222'
7500                                            43120   ;HERE WHEN FOUND RIGHT RESERVED WORD
7501  011232' 001000  000176                   43140   PRIT3:  MOV     A,M             ;GET A CHARACTER FROM RESERVED WORD
```

```
 7502  011233' 001000  000267          43160            ORA     A               ;SET CONDITION CODES
 7503  011234' 001000  000002          43180            STAX    B
 7504  011235' 001000  000372          43200            JM      PRIT4
 7505  011236' 000000  011167'
 7506  011237' 000000  011230'
 7507  011240' 001000  000003          43220            INX     B
 7508  011241' 001000  000043          43240            INX     H               ;BUMP RESLST POINTER
 7509  011242' 001000  000303          43260            JMP     PRIT3           ;PRINT THE REST
 7510  011243' 000000  011232'
 7511  011244' 000000  011236'
 7512                                   43280    ;
 7513                                   43300    ; THE FOLLOWING CODE IS FOR THE DELETE RANGE
 7514                                   43320    ; COMMAND, BEFORE THE LINES ARE DELETED, 'OK'
 7515                                   43340    ; IS TYPED,
 7516                                   43360    ;
 7517  011245' 001000  000315          43380    DELETE: CALL    SCNLIN          ;SCAN LINE RANGE
 7518  011246' 000000  002333'
 7519  011247' 000000  011243'
 7520  011250' 001000  000321          43400            POP     D               ;POP MAX LINE OFF STACK
 7521  011251' 001000  000305          43420            PUSH    B               ;SAVE POINTER TO START OF 1ST LINE
 7522  011252' 001000  000315          43440            CALL    FNDLIN          ;FIND THE LAST LINE
 7523  011253' 000000  002371'
 7524  011254' 000000  011246'
 7525  011255' 001000  000301          43460            POP     B               ;GET POINTER TO FIRST IN (B,C)
 7526  011256' 001000  000345          43480            PUSH    H               ;SAVE THE POINTER TO THE NEXT LINE
 7527  011257' 001000  000041          43500            LXI     H,REDDY         ;PRINT "OK" PREMATURELY
 7528  011260' 000000  001725'
 7529  011261' 000000  011253'
 7530  011262' 001000  000315          43520            CALL    STROUT
 7531  011263' 000000  007743'
 7532  011264' 000000  011260'
 7533  011265' 001000  000041          43540            LXI     H,FINI          ;GO BACK TO FINI WHEN DONE
 7534  011266' 000000  002276'
 7535  011267' 000000  011263'
 7536  011270' 001000  000343          43560            XTHL                    ;(H,L)=POINTER TO THE NEXT LINE
 7537  011271' 001000  000353          43580    DEL:    XCHG                    ;(D,E) NOW HAVE THE POINTER TO THE LINE
 7538                                   43600                                    ;BEYOND THIS ONE
 7539  011272' 001000  00005E          43620            LHLD    VARTAB          ;COMPACTIFYING TO VARTAB
 7540  011273' 000000  001621'
 7541  011274' 000000  011266'
 7542  011275' 001000  00003E          43640    MLOOP:  LDAX    D
 7543  011276' 001000  000002          43660            STAX    B               ;SHOVING DOWN TO ELIMINATE A LINE
 7544  011277' 001000  000003          43680            INX     B
 7545  011300' 001000  000023          43700            INX     D
 7546  011301' 001000  000347          43720            COMPAR
 7547  011302' 001000  000322          43740            JNC     MLOOP           ;DONE COMPACTIFYING?
 7548  011303' 000000  011275'
 7549  011304' 000000  011273'
 7550  011305' 001000  000140          43760            MOV     H,B
 7551  011306' 001000  000151          43780            MOV     L,C
 7552  011307' 001000  000043          43800            INX     H               ;NEW VARTAB
 7553  011310' 001000  00004E          43820            SHLD    VARTAB
 7554  011311' 000000  001621'
```



```
 7555  011312' 000000  011303'
 7556  011313' 001000  000311          43840            RET>
 7557                                   43860    PAGE
```

```
                                43880   SUBTTL  DISK CODE
                                43900   IFN     DSKFUN,<
                                43920   ;
                                43940   ; THE STATEMENT DSKO$ STRING,SECTOR WRITES
                                43960   ; THE STRING (UP TO 132 DECIMAL CHARS
                                43980   ; ON THE SECTOR SPECIFIED.
                                44000   ; DSKI$ (SECTOR) IS A STRING FUNCTION THAT
                                44020   ; RETURNS THE 133 BYTE STRING STORED ON SECTOR.
                                44040   ;
                                44060   DSKO$:  CALL    FRMEVL          ;EVALUATE FORMULA
                                44080           SYNCHK  44              ;FOLLOWED BY COMMA
                                44100           PUSH    H               ;SAVE TEXT POINTER
                                44120           CALL    FRESTR          ;FREE UP THE FACLO
                                44140           XTHL                    ;(H,L)=TXTPTR SAVE POINTER AT
                                44160                                   ;STRING DESCRIPTOR ON THE STACK
                                44180           CALL    GETBYT          ;EVALUATE 2ND ARG(SECTOR) IN (E)
                                44200           XTHL                    ;SAVE TEXT POINTER, GET DESC.
                                44220           PUSHM                   ;(C)=LENGTH (H,L)=POINTER
                                44240           PUSHM
                                44260           POP     H               ;(H,L) GET STRING POINTER
                                44280           POP     B
                                44300           MOV     B,A             ;SECTOR NUMBER INTO (B)
                                44320           MVI     A,^O137
                                44340           SUB     C
                                44360           JC      FCERR           ;STRING TOO LONG
                                44380           INR     A
                                44400           MOV     E,A             ;NUMBER OF ZEROS+1
                                44420           MVI     D,64            ;SETUP A MASK
                                44440           INR     C
                                44460           MVI     A,4             ;LOAD THE HEAD
                                44480           OUT     9               ;TO DISK STATUS
                                44500   SECLP:  IN      ^O11            ;GET SECTOR STATUS
                                44520           RAR                     ;TEST FOR START OF SECTOR
                                44540           JNC     SECLP           ;KEEP WAITING
                                44560           ANI     63              ;START OF SECTOR, RIGHT ONE
                                44580           CMP     B               ;COMPARE TO FIND OUT
                                44600           JNZ     SECLP           ;IF NOT
                                44620           MVI     A,128           ;WRITE ENABLE DISK
                                44640           OUT     9
                                44660           MVI     B,255           ;ALL ONE'S ALWAYS WRITTEN FIRST
                                44680   WRITOK: IN      8               ;GET STATUS
                                44700           ANA     D               ;WRITE OK
                                44720           JZ      WRITOK          ;NO, MORE LOOPING,
                                44740           MOV     A,B             ;GET CHARACTER TO WRITE
                                44760           OUT     10              ;SEND IT OUT
                                44780           DCR     C               ;TEST FOR NULL
                                44800           JZ      ZRLOP
                                44820   NOTYTO: IN      8               ;POLL
                                44840           ANA     D               ;MASK TEST
                                44860           JZ      NOTYTO          ;WAITING
                                44880           MOV     A,M             ;GET CHARACTER
                                44900           OUT     10
                                44920           DCR     C               ;DECREMENT CHARACTER COUNT
```



```
                                44940           INX     H
                                44960           JNZ     NOTYTO
                                44980   ZRLOP:  IN      8
                                45000           ANA     D
                                45020           JZ      ZRLOP
                                45040           XRA     A               ;PUT OUT A ZERO
                                45060           OUT     10
                                45080           DCR     E
                                45100           JNZ     ZRLOP
                                45120   TRUFIN: MVI     A,8             ;UNLOAD THE HEAD
                                45140           OUT     9
                                45160           POP     H
                                45180           RET                     ;DONE
                                45200
                                45220   DSKI$:  MVI     A,137           ;A LOT OF CHARACTERS ARE COMING
                                45240           CALL    STRIN1          ;MAKE ROOM!
                                45260           CALL    CONINT          ;WHERE ARE THEY?
                                45280                                   ;SECTOR NOW IN (E)
                                45300           LHLD    DSCTMP+2        ;PLACE TO STORE THEM
                                45320           MVI     A,4             ;LOAD THE HEAD
                                45340           OUT     9
                                45360   SECLP2: IN      9               ;GET SECTOR INFO
                                45380           ORA     A               ;SEE IF BEGINNING OF SECTOR(READ)
                                45400           JP      SECLP2          ;IF NOT, KEEP WAITING
                                45420           RAR                     ;FIX UP SECTOR #
                                45440           ANI     63              ;GET SECTOR #
                                45460           CMP     E               ;IS IT THE ONE WE WANTED
                                45480           JNZ     SECLP2          ;TRY TO FIND IT
                                45500           MVI     C,137           ;GET # OF CHARS TO READ
                                45520   READOK: IN      8               ;GET DISK STATUS
                                45540           ORA     A               ;READY TO READ BYTE
                                45560           JP      READOK
                                45580           IN      10              ;READ THE STUFF
                                45600           MOV     M,A             ;SAVE IN STR
                                45620           INX     H               ;BUMP DEST POINTER
                                45640           DCR     C               ;LESS CHARS
                                45660           JNZ     READOK
                                45680           MVI     A,8             ;UNLOAD HEAD
                                45700           OUT     9
                                45720           JMP     FINBCK          ;USE CHR$ TO FINISH UP
                                45740
                                45760   PATCH:  BLOCK   20>
                                45780   PAGE
```

```
                                45800   SUBTTL  CLOAD,CSAVE,CONSOLE
                                45820   ;
                                45840   ; THE CONSOLE COMMAND ALLOWS THE USER TO CHANGE THE I/O CHANNEL
                                45860   ; THAT THE USER TERMINAL IS ON, BY GIVING THE COMMAND CONSOLE X
                                45880   ; WHERE X IS SOME INTEGER THE TERMINAL DEVICE WILL BE POLLED FROM
                                45900   ; CHANNELS X AND X+1. RESTARTING AT LOCATION ZERO FORCES THE TERMINAL
                                45920   ; TO BE ON CHANNEL ZERO AGAIN.
                                45940   ;
                                45960   IFN     CONSSW,<
                                45980   INTERNAL        CONSOO
                                46000   CONSOO: XRA     A               ;FORCE A CHANNEL ZERO CONSOLE
                                46020           CALL    CONS2           ;ON RESTART AT ZERO
                                46040           JMP     READY          ;TYPE "OK" AND ACCEPT INPUT
                                46060   CONSOL: CALL    GETBYT          ;FETCH AN INTEGER INTO [A]
                                46080           RNZ                     ;CHECK FOR A TERMINATOR
                                46100   CONS2:
                                46120   IFN     REALIO,<
                                46140           STA     CNLCA1          ;CHANGE ALL THE FLAG INPUT CHANNEL REFERENCES
                                46160           STA     CNLCA2
                                46180           STA     CNLCA3>
                                46200   IFN     LENGTH,<
                                46220           STA     CNLCA4>
                                46240           INR     A               ;[A]=DATA INPUT CHANNEL
                                46260           STA     CNLCB1          ;CHANGE ALL THE DATA INPUT CHANNEL REFERENCES
                                46280           STA     CNLCB2
                                46300           RET>
                                46320   IFN     CASSW,<
                                46340   ;
                                46360   ; CASIN READS A CHARACTER FROM THE CASSETTE
                                46380   ; INTO [A] WITHOUT MODIFYING ANYTHING BUT [A] AND THE CONDITION
                                46400   ; CODES
                                46420   ;
                                46440   CASIN:  IN      6               ;ROUTINE TO READ A CHARACTER
                                46460           ANI     IDONE           ;FROM THE CASSETTE INTO [A]
                                46480           JNZ     CASIN
                                46500           IN      7               ;READ THE DATA
                                46520           RET
                                46540   ;
                                46560   ; CASOUT OUTPUTS THE CHARACTER IN [A] TO THE CASSETTE
                                46580   ; WITHOUT MODIFYING ANYTHING
                                46600   ;
                                46620   TWOCSO: CALL    CASOUT          ;DOUBLE OUT OF [A]
                                46640   CASOUT: PUSH    PSW             ;ROUTINE TO WRITE A CHARACTER IN [A]
                                46660   CASLK:  IN      6               ;ONTO THE CASSETTE
                                46680           ANI     ODONE
                                46700           JNZ     CASLK           ;WAIT TILL CASSETTE IS READY
                                46720           POP     PSW             ;GET THE CHARACTER BACK
                                46740           OUT     7               ;OUTPUT THE CHARACTER
                                46760           RET
                                46780   ;
                                46800   ; THE CSAVE COMMAND WRITES A PROGRAM ONTO CASSETTE BY DUMPING
                                46820   ; BASICS CORE. THE HEADER IS THREE 211'S FOLLOWED BY A ONE
                                46840   ; CHARACTER FILE NAME. THE END IS THREE ZEROS IN A ROW.
```



```
                                46860   ;
                                46880   CSAVE:  PUSH    H
                                46900           MVI     A,211
                                46920           CALL    CASOUT          ;PUT OUT THE START BYTES
                                46940           CALL    TWOCSO          ;TWO MORE TIMES
                                46960           MOV     A,M             ;GET FILENAME
                                46980           CALL    CASOUT          ;STORE AFTER 211'S
                                47000           LHLD    TXTTAB          ;START OF PROGRAM
                                47020           XCHG
                                47040           LHLD    VARTAB          ;END OF PROGRAM
                                47060   LOPCSO: LDAX    D               ;GET A BYTE FROM THE PROGRAM
                                47080           INX     D
                                47100           CALL    CASOUT          ;SEND IT OUT TO THE CASSETTE
                                47120           COMPAR                  ;THE END?
                                47140           JNZ     LOPCSO          ;IF NOT,OUTPUT MORE
                                47160           CALL    TWOCSO          ;TWO MORE 0'S TO MARK THE END
                                47180           POP     H               ;RESTORE THE TEXT POINTER
                                47200           CHRGET                  ;GO PAST THE FILE NAME
                                47220           RET
                                47240   ;
                                47260   ; THE CLOAD COMMAND CLEARS CORE AND THEN READS A PROGRAM
                                47280   ; FROM CASSETTE, SINCE THE LINKS OF THE FILE ON CASSETTE
                                47300   ; WILL BE WRONG IF THE FILE WAS SAVED WITH A DIFFERENT VERSION OF
                                47320   ; BASIC FINI IS JUMPED TO, A SCRATCH IS DONE AT THE START SO RESTARTS
                                47340   ; AT 0 WON'T LEAVE THINGS IN A GARBAGE STATE.
                                47360   ;
                                47380   CLOAD:  STA     FACLO           ;SAVE THE FILENAME
                                47400           CALL    SCRTCH          ;RESET EVERYTHING
                                47420   LOPCLK: MVI     B,3             ;NUMBER OF START CHARACTERS
                                47440   LOPCL2: CALL    CASIN           ;GET A CHARACTER
                                47460           CPI     211             ;START CHARACTER?
                                47480           JNZ     LOPCLK          ;NO, RESET COUNT AND LOOK SOME MORE
                                47500           DCR     B               ;DECREMENT THE COUNT
                                47520           JNZ     LOPCL2          ;SEEN THREE YET?
                                47540           LXI     H,FACLO         ;POINT AT THE FILENAME
                                47560           CALL    CASIN           ;READ THIS FILENAME
                                47580           CMP     M               ;THE RIGHT FILE?
                                47600           JNZ     LOPCLK          ;IF NOT,START COMPLETELY OVER
                                47620           LHLD    TXTTAB          ;PLACE TO STORE THE PROGRAM
                                47640   DOCKS:  MVI     B,4             ;NUMBER OF ZEROS TO GET
                                47660   DOCSMR: CALL    CASIN           ;BEFORE STOPPING
                                47680           MOV     M,A             ;GET A CHARACTER
                                47700           CALL    REASON          ;STORE IT
                                47720           MOV     A,M             ;MAKE SURE THERE IS ROOM
                                47740           ORA     A               ;REGET THE CHARACTER
                                47760           INX     H               ;A ZERO?
                                47780           JNZ     DOCKS           ;RESET # OF ZEROS SEEN
                                47800           DCR     B               ;DECREMENT NUMBER OF ZEROS
                                47820           JNZ     DOCSMR          ;SEEN FOUR?
                                47840           SHLD    VARTAB          ;SETUP END OF PROGRAM
                                47860           LXI     H,READY         ;TYPE "OK" PREMATURELY
                                47880           CALL    STROUT
                                47900
```

```
BASIC MCS 8080  GATES/ALLEN/DAVIDOFF    MACRO 47(113) 03:12 10-SEP-75 PAGE 29-2
F3      MAC     6-SEP-64 03:11          CLOAD,CSAVE,CONSOLE

  7760                                  47920           JMP     FIN1>           ;FIX UP THE LINKS AND GO BACK TO MAIN
  7761                                  47940   PAGE
```

```
BASIC MCS 8080  GATES/ALLEN/DAVIDOFF    MACRO 47(113) 03:12 10-SEP-75 PAGE 30
F3      MAC     6-SEP-64 03:11          PEEK AND POKE

  7762                                  47960   SUBTTL PEEK AND POKE
  7763                                  47980   IFN     LENGTH,<
  7764                                  48000   IFE     LENGTH-2,<
  7765  011314' 001000  000315         48020   PEEK:   CALL    FRCINT          ;GET AN INTEGER IN [H,L]
  7766  011315' 000000  011025*
  7767  011316' 000000  011311'
  7768  011317' 001000  000176         48040           MOV     A,M>            ;GET THE VALUE TO RETURN
  7769                                  48060   IFN     LENGTH-2,<
  7770                                  48080   PEEK:   CALL    POSINT          ;GET THE VALUE OF FACLO INTO [D,E]
  7771                                  48100           LDAX    D>              ;READ THE VALUE
  7772  011320' 001000  000303         48120           JMP     SNGFLT          ;AND FLOAT IT
  7773  011321' 000000  007400'
  7774  011322' 000000  011315'
  7775                                  48140   IFE     LENGTH-2,<
  7776  011323' 001000  000315         48160   POKE:   CALL    FRMEVL
  7777  011324' 000000  005336'
  7778  011325' 000000  011321'
  7779  011326' 001000  000345         48180           PUSH    H               ;SAVE THE TEXT POINTER
  7780  011327' 001000  000315         48200           CALL    FRCINT          ;GET INTEGER VALUE OF FAC IN [H,L]
  7781  011330' 000000  011315*
  7782  011331' 000000  011324'
  7783  011332' 001000  000343         48220           XTHL>                   ;GET BACK THE TEXT POINTER
  7784                                  48240   IFN     LENGTH-2,<
  7785                                  48260   POKE:   CALL    INT102          ;READ LOCATION TO POKE
  7786                                  48280           PUSH    D>              ;SAVE THE LOCATION
  7787  011333' 001000  000317         48300           SYNCHK  44              ;CHECK FOR A COMMA
  7788  011334' 000000  000054
  7789  011335' 001000  000315         48320           CALL    GETBYT
  7790  011336' 000000  011020'
  7791  011337' 000000  011330'
  7792  011340' 001000  000321         48340           POP     D               ;GET THE ADDRESS BACK
  7793  011341' 001000  000022         48360           STAX    D               ;STORE IT AWAY
  7794  011342' 001000  000311         48380           RET>                    ;SCANNED EVERYTHING
  7795                                  48400   ;
  7796                                  48420   ; NOTE: IN THE 8K PEEK ONLY ACCEPTS POSITIVE NUMBERS UP TO 32767
  7797                                  48440   ; POKE WILL ONLY TAKE AN ADDRESS UP TO 32767 , NO
  7798                                  48460   ; FUDGING ALLOWED, THE VALUE IS UNSIGNED.
  7799                                  48480   ;
  7800                  011336'         48500   .C1=.P
  7801                                  48520   END

NO ERRORS DETECTED

PROGRAM BREAK IS 011343

10K CORE USED
```

A	000007'		CNLCA1	003114'	SIN	DCOMP	000702'	EXT
ABS	000107'	EXT	CNLCA2	003127'	SIN	DCX8RT	011204'	EXT
AFF	000104		CNLCA3	003461'	SIN	DDIV	000700'	EXT
ALLLST	002354'		CNLCA4	003303'	SIN	DEF	007411'	
ALLSTR	010570'		CNLCB1	003124'	SPD	DEFFIN	007521'	
ANDORD	005571'		CNLCB2	003136'	SPD	DEL	011271'	
APPLOP	005642'		CNTCON	003465'		DELETE	011245'	
ARG	001654'	INT	CNTWFL	001541'	INT	DFACLO	001633'	INT
ARGLO	001645'	INT	COLIS	002705'		DIM	006500'	
ARYSTR	010177'		COMPRT	004553'		DIMCON	006473'	
ARYTAB	001623'		CONIA	006405'	EXT	DIMFLG	001542'	
ARYVA2	010137'		CONINT	011023'		DIRIS	003526'	
ARYVAR	010140'		CONSIM	006041'	EXT	DMULT	000676'	EXT
ASC	010515'		CONSSW	000000	SPD	DNTCPY	004217'	
ASPA2	004626'		CONT	003542'		DOASIG	005056'	
ASPAC	004627'		CONTRW	000001	SPD	DOCMP	006375'	
ATN	000137'	EXT	CONTW	000017		DOCOND	004354'	
ATNFIX	000137'	INT	COPNUM	004237'		DODSP	005756'	
ATNTK	000300	SPD	COS	000151'	EXT	DOMIN	006146'	
B	000000		COSFIX	000131'	INT	DONMUL	007326'	
BLTLOP	002013'		CR	000015	SPD	DORES	001544'	
BLTU	002005'		CROD	004523'	INT	DOSIZT	004621'	
BLTUC	002010'		CRDONE	002753'		DSCTMP	001570'	
BOTCON	000016	SPD	CRFIN	004534'		DSKFUN	000000	SPD
BRKTXT	001734'		CRUNCH	002533'		DSUB	000674'	EXT
BSERR	007677'	INT	CSLOOP	006345'		DVBERR	002075'	INT
BUF	001431'	INT	CURLIN	001607'	INT	DVAR	010214'	
BUFLEN	000110	SPD	CZLOOP	002316'		DVAR2	010213'	
BUFLIN	011163'		D	000002		DVARS	010214'	
BUFMIN	001430'		DADD	000672'	EXT	E	000003	
C	000001		DANDOR	006437'		EATEM	006535'	
CASSW	000000	SPD	DATA	004072'		EDIT	000632'	EXT
CAT	010320'		DATATK	000203	SPD	ELSE	004074'	
CHEAD	002302'		DATBK	005025'		ELSETK	000220	SPD
CHKCOM	004560'		DATFND	005166'		END	003474'	
CHKSTR	010432'	EXT	DATLIN	001577'		ENDCON	003501'	
CHRS	010532'		DATLOP	005163'		ENDREL	005415'	
CHRCON	003433'		DATPTR	001627'		ENDTK	000200	SPD
CHRGTR	003426'		DATSNE	002064'		EQULTK	000260	
CLEAR	003703'		DBLDSP	000672'		ERR8S	000011	SPD
CLEARC	002443'					ERRCN	000021	SPD
CLMWID	000016	SPD				ERRDD	000012	SPD
						ERRDIM	007532'	
						ERRDV0	000013	SPD
						ERRFC	000005	SPD
						ERRFIN	002127'	
						ERRID	000014	SPD
						ERRLS	000017	SPD
						ERRNF	000001	SPD
						ERROD	000004	SPD

ERROM	000007	SPD	FPWR	000000	EXT	INCHR	003126'	
ERROR	002102'	INT	FPWRQ	005562'	EXT	INDLOP	006754'	
ERROV	000006	SIN	FRCDBL	005751'	EXT	INEG	000000	EXT
ERRRG	000003	SPD	FRCINT	011330'	EXT	INIT	002163'	EXT
ERRSN	000002	SPD	FRCSNG	007506'	EXT	INLIN	002776'	
ERRSO	000016	SPD	FRCSTR	000000	EXT	INLINC	003003'	
ERRST	000020	SPD	FRCTBL	000664'		INLINN	002772'	
ERRTAB	000730'		FRE	007337'		INLPNM	007252'	
ERRTM	000015	SIN	FREFAC	010434'		INPCON	004157'	
ERRUF	000022	SPD	FRESTR	010431'		INPCON	004765'	
ERRUS	000010	SPD	FRETM2	010437'		INPRT	002141'	EXT
EVAL	006061'		FRETMP	010440'		INPUT	004711'	
EXCHGT	004100'		FRETOP	001573'	INT	INPWRD	010720'	
EXIGNT	005142'		FRMCHK	005337'		INRART	000000	EXT
EXP	000127'	EXT	FRMEVL	005336'		INT	000000	EXT
EXPSTK	005553'		FSUB	000706'	EXT	INTDSP	000716'	
EXTFNC	000001	SPD	FUNCTS	000001	SPD	INTID2	003623'	
FAC	001642'	INT	FUNDSP	000103'		INTIDX	003622'	
FACDBL	005775'		GARBA2	010042'		INTXT	001720'	INT
FACLO	001637'	INT	GARBAG	010026'		INXHRT	002664'	EXT
FACSNG	006034'		GETAGN	004741'		ISARY	006745'	
FADD	000704'	EXT	GETBCD	007752'	EXT	ISCNTC	003460'	
FADDS	005260'	EXT	GETBYT	011020'		ISFUN	006204'	
FALSIF	004363'		GETDEF	007245'		ISGOSU	004306'	
FBUFFR	001655'	INT	GETFNM	007550'		ISIUN	000000	EXT
FCERR	010776'	INT	GETSPA	007772'		ISLET	003612'	
FCOMP	005274'	EXT	GETSTK	002024'		ISSEC	006534'	
FDIV	000712'	EXT	GETYPE	006307'	INT	ISUB	000720'	EXT
FIN	011064'	EXT	GIVDBL	007372'		ISVAR	006164'	
FINBCK	010546'		GIVINT	007402'		KLOOP	002544'	
FINGO	006276'		GONE	003370'		L	000005	
FINI	002276'		GONE2	003376'		LABBCK	006162'	
FININL	004516'		GONE3	003375'		LASNUM	000306	SPD
FINNOW	007331'		GOODCH	003044'		LEFT$	010552'	
FINPTR	006742'		GOSUB	003770'		LEFT2	010560'	
FINREL	005605'		GOSUTK	000214		LEFT3	010556'	
FINTMP	005544'		GOTO	004010'		LEN	010501'	
FLGINP	001602'		GOTOTK	000210	SPD	LEN1	010505'	
FLOAT	000000	EXT	GRBPAS	010254'		LENGTH	000002	SPD
FLOATR	000000	EXT	GREATK	000257		LEPSKP	002117'	
FMULT	000710'	EXT	GTBYTC	011017'				
FNDFOR	001744'		H	000004				
FNDLIN	002371'		HAVTYP	006574'				
FNDUER	007443'		IADD	000716'	EXT			
FNDVAR	010045'		ICOMP	000726'	EXT			
FNINP	010712'		IDIV	000724'	EXT			
FNOUT	010725'		IDONE	000001	SPD			
FNTK	000243	SPD	IF	004325'				
FNWAIT	010733'		IFORDN	003267'				
FOR	003154'		IFTK	000212	SPD			
FORTK	000201	SPD	ILLFUN	010776'	INT			
FOUND	002646'		IMULT	000722'	EXT			
FOUT	007565'	EXT						

```
LESSTK          000261'
LET             004131'
LINCHK          004475'
LINGET          003642' INT
LINGT2          003643'
LINLEN          000110  SPD
LINLIN          002764'
LINPRT          011134' EXT
LINPT1          003103' SIN
LINPT2          003574' SIN
LINPT3          004474' SIN
LINPT4          004557' SIN
LIST            011072'
LIST4           011077'
LISTEN          000001  SPD
LOG             000125' EXT
LOOP            002374'
LOOPDN          005320'
LOOPER          001750'
LOOPON          004307'
LOPDT2          004773'
LOPFDA          007015'
LOPFND          006623'
LOPPTA          007135'
LOPREL          005363'
LPOPER          005341'
LPTLEN          000110  SPD
LPTSW           000000  SPD
LSTOPK          000007  SPD
M               000006  SPD
MAIN            002165'
MAKINT          007404' EXT
MEMSIZ          001545' INT
MID$            010643'
MID2            010665'
MINUTK          000251  INT
MLOOP           011275'
MLOOPR          002266'
MORCOM          004566'
MORLIN          003646'
MORPR           004410'
MOVE            000000  EXT
MOVFR           005253' EXT
MOVFR           005746' EXT
MOVINS          010407'
MOVLP           010420'
MOVMF           007476' EXT
MOVRF           006044' EXT
MOVRM           010154' EXT
MOVSTR          010417'
MULDIM          000001  SPD
MUSTCR          002617'
NEG             000000  EXT
NEWCHR          004407'
NEWSTT          003302'
NEXT            005225'
NEXTC           005230'
NFERR           002100'
NMARY1          007051'
NMARY2          007050'
NMREL           000003  SPD
NODATT          002710'
NODEL           002227'
NOPRIN          003113'
NOSEC           006547'
NOTABR          004637'
NOTDIM          007152'
NOTER           006412'
NOTFDD          007104'
NOTFNS          006671'
NOTFRF          006272'
NOTIT1          006660'
NOTLST          010477'
NOTOL           003176'
NOTQT1          004735'
NOTSTV          005360'
NOTTHN          010753'
NOTTK           000246  SPD
NOTTRC          003367'
NUMGET          005053'
NUMLIN          005214'
NTHIS           002736'
NTHIS1          002742'
NULCNT          000046'
NULL            003566'
NUMCMD          000040
NUMGFN          000051  SPD
NUMINS          005070'
NUMLEV          000025  SPD
NUMREL          005476'
NUMTMP          000005  SPD
NUTPOS          000070  SPD
NXTCON          003276'
NXTRES          002625'
ODONE           000200  SPD
OKGOTO          004346'
OKNORM          006251'
OLDLIN          001611'
OLDTXT          001613'
OMERR           002057' INT
ONEFUN          000262
ONELIN          002365'
ONEON           003265'
ONGOTO          004271'
OPRTYP          001544'
OPTAB           000163'
ORFIN           006463'
OUTBEL          003061'
OUTCON          003065'
OUTDO           000030' INT
OUTWRD          010730'
PARCHK          006136'
PEEK            011314'
PLOOP           011170'
PLUSTK          000250  INT
POKE            011323'
POPGOF          001774'
POPHRT          000000  EXT
POS             007406'
PRSWRT          010024'
PREAM           010764'
PREAM2          010767'
PRINT           004411'
PRINTC          004414'
PRINTK          000231  SPD
PRIT3           011232'
PRIT4           011167'
PRTNUL          004577'
PSW             000006  SPD
PTRGET          006505'
PTRGT2          006512'
PUFOUT          000000  EXT
PURE            000000  SPD
PUSHF           006036' EXT
PUSHMA          0001000'
PUTDEI          007524'
PUTNEW          007705'
PUTVAL          004261'
Q               000022
QINLIN          002522' INT
QINT            000000  EXT
```



```
RAMBOT          020000
READ            004760'
READY           002145' INT
REALIO          000001  SPD
REASON          002045' INT
REDDY           001725'
REDINP          004136'
REM             004074'
REMER           004103'
REMTK           000216
REPINI          002162' INT
REPOUT          004632'
RESCR1          011216'
RESER           002627'
RESFIN          003453'
RESLST          000172'
RESRCH          011214'
RESTOR          003446'
RETADP          005554'
RETURN          004044'
RIGHT$          010631'
RND             000123' EXT
RNDFIX          000123'
RUN             003754'
RUNC            002437'
RUNC2           004007'
SCNLIN          002333'
SCRATH          002421'
SCRATK          000237
SCRTCH          002422' INT
SETDBL          005750'
SETIO           011003'
SGN             000103' EXT
SIGN            000050' INT
SIGNC           000055' EXT
SIGNS           006373' EXT
SIN             000133' EXT
SINFIX          000133' INT
SKPMRF          004365'
SKPVAR          010130'
SMLVAL          007322'
SNERR           002072' INT
SNGDBL          005743'
SNGOO           006026'
SNGUSP          002704'
SNGFLT          007400'
SP              000006
SPCTK           000242  SPD
SQR             000121' EXT
SQRFIX          000121'
SQRTK           000271  SPD
STAINP          010754'
START           000000'
STEPTK          000247
STKOBL          005725'
STKINI          002470'
STKSNG          006021'
STKTOP          001615' INT
STMOSP          000564'
STOP            003472'
STPEND          003500'
STPRDY          002144'
STR$            007564'
STR1            002716'
STRAD1          007632'
STRAD2          007627'
STRCMP          006320'
STRCPY          007601'
STRUN2          005077'
STRUON          004507'
STRENO          001625' INT
STRFIN          007665'
STRGET          007646'
STRING          000001  SPD
STRIN1          007624'
STRLIT          007637' INT
STRLT2          007643'
STRLT3          007642'
STRLTI          007640'
STRNG           002727'
STROUI          007742' INT
STROUT          007743' INT
STRPR2          007755'
STRPRT          007746'
STRSIZ          000003  SPD
STUFFH          002667'
SUBFLG          001681'
SVAR            010102'
SVARS           010077'
TABEND          002653'
TABER           004577'
TABTK           000240
TAN             000135' EXT
TANFIX          000135' INT
TEMP            001603'
TEMP2           001605' INT
TEMP3           001575' INT
TEMP6           007124'
TEMPPT          001547' INT
TEMPST          001551' INT
THENTK          000245
TMERR           005703' EXT
TOFF            003605'
TON             003604'
TOPCON          000035  SPD
TOTK            000241  SPD
TRCFLG          001631'
TRMNOK          004667'
TRMOK           005111'
TRYAGN          004644'
TRYGI2          007774'
TRYIN           003126'
TRYOUT          003113'
TSTACK          000000  EXT
TSTOP           005357'
TTICHN          000001  SPD
TTOCHN          000001  SPD
TTYCHK          003070'
TTYIST          004616'
TTYPOS          000047'
TVAR            010063'
TXTTAB          001617' INT
UMULT           007267' EXT
USERR           004037'
USINTK          000244  SPD
USRLOC          000111'
VAL             011042'
VALINT          000000  EXT
VALSNG          000000  EXT
VALTYP          001543' INT
VARTAB          001621'
VINT            000105' EXT
VMOVAF          005777' EXT
VMOVE           007724' EXT
VMOVFA          000000  EXT
VMOVFM          006200' EXT
VMOVMF          004264' EXT
VNEG            006160' EXT
VPUSHD          005536'
VSIGN           004347' EXT
```

```
ZERITA          007207'
ZERO            011046' EXT
ZEROER          006723'
$CODE           000000' INT
```

```
A       764   791   812   827   830   852  1740  1756  1796  1887  1922  1924  1943  1965
       1966  1978  2049  2052  2069  2078  2154  2187  2189  2269  2272  2290  2299  2305
       2312  2320  2321  2322  2330  2335  2368  2373  2378  2417  2418  2419  2519  2520
       2523  2528  2606  2732  2803  2806  2875  2882  2887  2907  2912  2918  2944  2962
       2976  2979  3024  3026  3041  3070  3089  3103  3112  3152  3153  3197  3220  3222
       3223  3225  3397  3399  3400  3401  3428  3527  3534  3554  3558  3561  3587  3588
       3599  3623  3721  3834  3839  3859  3866  3872  3880  3953  3972  3973  3974  4019
       4041  4080  4081  4096  4126  4141  4149  4181  4182  4249  4303  4310  4392  4453
       4535  4555  4566  4567  4571  4583  4610  4662  4713  4773  4823  4827  4841  4842
       4848  4911  4916  4919  5123  5127  5193  5241  5242  5243  5349  5370  5373  5374
       5377  5379  5396  5397  5402  5440  5442  5443  5445  5473  5478  5479  5481  5487
       5488  5490  5529  5540  5541  5559  5594  5622  5628  5643  5664  5675  5895  5896
       5897  5942  5948  5965  6017  6019  6052  6053  6100  6104  6108  6161  6263  6265
       6266  6269  6270  6271  6423  6437  6444  6482  6497  6511  6540  6542  6562  6588
       6612  6659  6671  6694  6746  6759  6760  6782  6825  6871  6929  6976  6981  7052
       7070  7071  7074  7075  7092  7101  7126  7128  7130  7135  7161  7191  7218  7223
       7224  7227  7342  7343  7351  7368  7380  7409  7439  7465  7467  7490  7495  7501
       7502  7768
ABS     904   904#
AFFF    986#   986   987#   987   988#   988   989#   989   990#   990   991#   991   992#   992
        993#   993   994#   994   995#   995   998#   998   999#   999  1001#  1001  1002#  1002
       1005#  1005  1013#  1013  1021#  1021  1022#  1022  1023#  1023  1025#  1025  1026#  1026
       1027#  1027  1028#  1028  1029#  1029  1033#  1033  1046#  1046  1047#  1047  1048#  1048
       1049#  1049  1050#  1050  1052#  1052  1054#  1054  1056#  1056  1195#  1196#  1197#  1198#
       1199#  1200#  1201#  1202#  1203#  1204#  1205#  1206#  1207#  1208#  1209#  1210#  1210   1215#
       1216#  1217#  1218#  1219#  1220#  1221#  1222#  1223#  1224#  1225#  1226#  1226   1231#  1232#
       1233#  1234#  1235#  1236#  1237#  1238#  1239#  1240#  1241#  1242#  1243#  1244#  1245#  1246#
       1247#  1248#  1249#  1250#  1250   1255#  1256#  1257#  1258#  1259#  1260#  1261#  1262#  1263#
       1264#  1265#  1265   1276#  1271#  1272#  1273#  1274#  1275#  1276#  1277#  1278#  1279#  1280#
       1281#  1282#  1283#  1284#  1285#  1286#  1287#  1288#  1289#  1290#  1290   1295#  1296#  1297#
       1298#  1299#  1300#  1301#  1302#  1302   1307#  1308#  1309#  1310#  1311#  1312#  1313#  1314#
       1315#  1316#  1317#  1318#  1319#  1319   1324#  1325#  1326#  1327#  1328#  1329#  1330#  1331#
       1332#  1333#  1334#  1335#  1336#  1337#  1338#  1339#  1340#  1341#  1342#  1342   1347#  1348#
       1349#  1350#  1351#  1352#  1353#  1354#  1355#  1356#  1357#  1358#  1359#  1360#  1361#  1362#
       1363#  1364#  1365#  1366#  1367#  1368#  1368   1373#  1374#  1375#  1376#  1377#  1378#  1379#
       1380#  1381#  1382#  1383#  1384#  1385#  1386#  1387#  1388#  1389#  1390#  1391#  1391   1396#
       1397#  1398#  1399#  1400#  1401#  1402#  1403#  1404#  1405#  1406#  1407#  1408#  1409#  1410#
       1411   1411   1416#  1417#  1418#  1419#  1420#  1421#  1422#  1423#  1424#  1425#  1426#  1427#
       1428#  1429#  1429   1435#  1436#  1437#  1438#  1439#  1440#  1441#  1442#  1443#  1444#  1445#
       1446#  1447#  1447   1452#  1453#  1454#  1455#  1456#  1457#  1458#  1459#  1460#  1461#  1462#
       1463#  1464#  1465#  1466#  1467#  1468#  1469#  1470#  1470   1475#  1476#  1477#  1478#  1479#
       1480#  1481#  1482#  1483#  1484#  1485#  1486#  1487#  1488#  1489#  1489   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#  1519   1525#  1526#  1527#  1528#
       1529#  1530#  1531#  1532#  1533#  1534#  1535#  1536#  1537#  1538#  1538   1544#  1545#  1546#
       1547#  1548#  1549#  1550#  1551#  1552#  1553#  1554#  1555#  1556#  1557#  1558#  1559#  1560#
       1561#  1562#  1563#  1564#  1565#  1566#  1566   1705#  1706#  1707#  1708#  1708   1713#  1714#
       1714   1722#  1723#  1724#  1725#  1726#  1726   3992#  3993#  3994#  3995#  3996#  3997#  3998#
       3999#  4000#  4001#  4002#  4003#  4004#  4005#  4006#  4007#  4007   4269#  4270#  4271#  4272#
       4273#  4274#  4275#  4276#  4277#  4278#  4279#  4280#  4281#  4282#  4282
ALLLST  2103  2115#
ALLSTR  7133  7137#
ANDORD  4672  4752#
```

DATLIN	1624#	1869	4324											
DATLOP	4147	4298#	4331											
DATPTR	1672#	2990	4093											
DATSNE	1866#	4021												
DBLDSP	1145#	4903												
DCOMP	168#	1153												
DCXBRT	167#	7478#	7478											
DDIV	166#	1151												
DEF	1114	6307#												
DEFFIN	6337	6404#												
DEL	1994	7537#												
DELETE	1123	7517#												
DFACLO	170	1600#	4708	4887	4891									
DIM	1068	5516#												
DIMCON	5497#	5517												
DIMFLG	1598#	5531	5877	5910	5963	6050	6102							
DIRIS	3028	3039#												
DMULT	168#	1149												
DNTCPY	3513	3524#												
DOASIG	4201#	4217												
DOCMP	4787	5396#												
DOCOND	3700#	3736												
DODSP	4905#	4964												
DOMIN	5022	5061#												
DONMUL	6183	6187	6190#											
DORES	1607#	2292	2318	2409										
DOSIZT	3966#													
DSCMP	1615#	6508	6579	6954	7110									
DSKFUN	98	936	995	1052	1107	7559								
DSUB	166#	1147												
DVBERR	136	1877#												
DVAR	6824#													
DVAR2	6731	6822#												
DVARS	6757	6825#												
E	831	1757	1857	1874	1877	1881	1901	2039	2068	2085	2802	2896	3062	3189
	3197	3221	3222	3340	3362	3519	3597	3953	3967	4312	4319	4583	4594	4712
	5364	5370	5376	5383	5473	5738	5741	5954	5966	5985	6019	6079	6116	6143
	6264	6372	6488	6427	6603	6689	6760	6828	6890	6936	7041	7112	7197	7226
	7229	7267	7282	7300	7306	7351	7368							
EATEM	5560#	5563	5569											
EDIT	1097	1097#												
ELSE	1091	3388#												
ELSETK	994#	2383	3725	7475										
END	1058	3014#												
ENDCON	2894	3019#												
ENDREL	4542	4548	4566#											
ENDTK	986#	2932	2936											
EQULTK	1033#	3459	6330											
ERRBS	1370#	5986												
ERRCN	1540#	3063												
ERRDD	1393#	5967												
ERRDIR	4073	6317	6418#											
ERRDV0	1413#	1878												

ERRFC	1292#	7307												
ERRFIN	1916#	3050												
ERRID	1431#	6428												
ERRLS	1491#	6937												
ERRNF	1212#	1682												
ERROD	1267#	4313												
ERROM	1321#	1858												
ERROR	147	1863	1883#	3073	3343	3365	4315	5969	5988	6375	6430	6606	6692	6939
	7309													
ERROV	136	1304#												
ERRRG	1252#	3363												
ERRSN	1220#	1675												
ERRSO	1472#	6690												
ERRST	1521#	6604												
ERRTAB	1184#	1895												
ERRTM	170	1449#												
ERRUF	1568#	6373												
ERRUS	1344#	3341												
EVAL	4519	4993#	5011	6923										
EXCMGT	3397#	3409												
EXIGNT	4260	4268#												
EXP	140#	922												
EXPSTK	4667	4732#												
EXTFNC	38#	139	919	928	968	1029	1049	1050	5062	5064				
FAC	144	850	1684#	4983										
FACDBL	4839	4927#												
FACLO	144	1681#	3500	3810	4689	4798	4870	4945	4987	5103	5166	5447	6585	6919
	6932	7018												
FACSNG	4852	4970#												
FADD	141#	1155												
FADDS	143#	4408												
FALSIF	3696	3711#												
FBUFFR	155	1690#												
FCERR	147	3095	3155	7085	7306#	7345								
FCOMP	141#	1163	4421											
FDIV	1161	1161#												
FIN	141#	4213	5000	5017	7382									
FINBCK	6473	7113#												
FINGO	5180	5213#												
FINI	2015	2056#	7534											
FININL	2497	3850#												
FINNOW	6121	6199#												
FINPTR	5656	5747#												
FINREL	4569	4773#												
FINTMP	4719#	4745	4762	4805										
FLGINP	1631#	4017	4098	4139	4247									
FLOAT	145#													
FLOATR	143#													
FMULT	1159	1159#												
FNDFOR	1735#	2745	3357	4381										
FNDLIN	1989	2148#	3333	7523										
FNDUER	5042	6344#												
FNDVAR	6706#	6908												

FNINP	910	7239#													
FNOUT	1099	7251#													
FNTK	1021#	5040	6436												
FNWAIT	1105	7263#													
FOR	1060	2731#													
FORTK	987#	1743	2640												
FOUND	2372#														
FOUT	141#	3802	6464												
FPWR	140#														
FPWRQ	166#	4739#	4739												
FRCDBL	167#	1134	4880	4899											
FRCINT	167#	748	1136	3150	4754	5438	5466	7335	7766	7781					
FRCSNG	167#	1138	2790	2822	4733	4955	5205	6390							
FRCSTR	169#														
FRCTBL	1134#	3552													
FRE	920	6209#													
FREFAC	6230	6470	6626	7017#											
FRESTR	5346	7014#	7068												
FRETM2	6950	7020#													
FRETMP	3531	5359	6946	7021#	7167										
FRETOP	153	1616#	2215	6241	6668	6681	6707	6845	6895	7045	7055				
FRMCHK	2817	4511#													
FRMEVL	2783	3146	3468	3617	3783	4510#	5056	5157	6386	7331	7777				
FSUB	1157	1157#													
FUNCTS	127#	999	1021	1114	1541	4064	4071	5038	6289	6403	6413				
FUNDSP	897#	5214													
GARBA2	6234	6700#													
GARBAG	6678	6688#													
GETAGN	4076#														
GETBCD	166#	6630													
GETBYT	3086	3585	5173	7208	7277	7316	7330#	7790							
GETDEF	5982	6137#													
GETFNM	6308	6345	6435#												
GETSPA	6488	6505	6659#	7141											
GETSTK	1821#	2759	3281	4515	6033										
GETYPE	170	3792	4792	5108	5236#	6220									
GIVDBL	6223	6263#													
GIVINT	6271#														
GONE	1982	2880	2924#												
GONE2	2931#	3601													
GONE3	2929#	3705													
GOODCH	2492	2519#													
GOSUB	1082	3280#													
GOSUTK	991#	3292	3361	3590											
GOTO	1074	3303#	3702												
GOTOTK	989#	3595	3655												
GRBPAS	6779	6869#													
GREATK	1032#	4540	4544												
GTBYTC	3942	7328#													
H	769	790	827	873	885	887	1735	1741	1750	1752	1753	1755	1767	1800	
	1824	1826	1837	1848	1894	1902	1922	1947	1953	2024	2028	2033	2034	2037	
	2040	2042	2050	2060	2067	2070	2075	2076	2077	2080	2086	2127	2151	2155	
	2157	2162	2164	2188	2190	2199	2243	2249	2252	2353	2377	2393	2427	2435	

	2436	2440	2449	2468	2480	2530	2763	2771	2787	2795	2819	2827	2833	2886	
	2890	2892	2897	2948	2953	2961	2988	3025	3046	3070	3148	3152	3158	3173	
	3179	3181	3188	3191	3193	3202	3213	3216	3223	3244	3285	3286	3317	3327	
	3328	3336	3338	3367	3371	3405	3484	3502	3503	3504	3520	3536	3543	3551	
	3559	3560	3567	3569	3578	3596	3661	3745	3770	3787	3788	3846	3852	3946	
	3951	3981	4025	4055	4059	4063	4079	4082	4091	4192	4203	4226	4237	4259	
	4266	4308	4320	4393	4405	4410	4415	4419	4423	4440	4510	4606	4688	4692	
	4694	4700	4702	4707	4757	4786	4790	4800	4858	4866	4885	4889	4902	4917	
	4918	4943	4959	4970	4981	4985	5074	5086	5092	5118	5170	5203	5207	5222	
	5223	5350	5353	5365	5367	5369	5386	5443	5445	5479	5481	5488	5490	5497	
	5592	5626	5638	5644	5664	5667	5674	5708	5712	5718	5724	5729	5739	5742	
	5751	5875	5893	5908	5938	5944	5952	5953	5955	5957	6018	6025	6027	6042	
	6043	6048	6063	6065	6067	6072	6091	6100	6109	6117	6119	6137	6142	6144	
	6146	6154	6162	6172	6174	6181	6185	6202	6213	6266	6271	6331	6361	6364	
	6365	6366	6367	6368	6381	6388	6392	6393	6400	6405	6407	6409	6411	6418	
	6422	6423	6425	6483	6486	6490	6496	6507	6510	6532	6536	6539	6560	6611	
	6616	6675	6683	6709	6712	6717	6718	6747	6748	6750	6783	6789	6803	6809	
	6827	6829	6831	6842	6848	6851	6855	6856	6862	6870	6872	6875	6877	6879	
	6882	6886	6891	6892	6900	6902	6905	6906	6917	6930	6934	6952	6963	6967	
	6973	6977	6979	7024	7026	7028	7057	7087	7091	7129	7144	7145	7146	7150	
	7151	7156	7269	7333	7338	7347	7371	7373	7386	7406	7429	7441	7446	7464	
	7468	7482	7496	7508	7526	7527	7533	7550	7552	7779					
HAVTYP	5574	5594#													
IADD	167#	1165													
ICOMP	166#	1173													
IDIV	168#	1171													
IDONE	123#	2633	2861	3000											
IF	1078	3616#													
IFORDN	2831#														
IFTK	990#	3419													
ILLFUN	144	906	7305#												
IMULT	168#	1169													
INCHK	2486	2628#	3003												
INDLOP	5882#	5901													
INEG	166#														
INIT	1414	734	1957												
INLIN	1962	2279	2480#												
INLINC	2472	2485#	2502	2507	2536										
INLINN	2474#	2512													
INLPNM	6142#	6165													
INPCOM	3484#	4209													
INPCON	4089	4096#													
INPKT	142#	1926													
INPUT	1060	4036#													
INPWRD	7243	7245#													
INRART	145#														
INT	141#														
INTDSP	1165#	4859													
INTID2	3145#	3211													
INTIDX	3144#	5885													
INTXT	155	1704#													
INXHRT	2386#	2388													
ISARY	5619	5875#													

ISCNTC	2996#	7417												
ISFUN	5047	5120#												
ISGOSU	3592	3597#												
ISIGN	166#													
ISLET	3112#	5003	5535	5554	5566									
ISSEC	5550	5559#												
ISUB	160#	1167												
ISVAR	5006	5088#												
KLOUP	2299#	2415												
L	830	1754	1923	2068	2153	3024	3071	3189	3220	3337	3519	3561	4439	4919
	5193	5224	5440	5442	5477	5478	5486	5487	5643	5725	6108	6163	6173	6263
	6265	6269	6332	6424	6497	6843	6849	6857	6871	6885	6893	6904	6981	6987
	6988	7152	7157	7161	7374	7551								
LABBCK	5081#	5210												
LASNUM	1054#	5128												
LEFT3	940	7123#												
LEFT2	7129#	7215												
LEFT3	7127#	7181												
LEN	938	7063#												
LEN1	7067#	7082	7363											
LENGTH	4#	22	32	37	82	111	117	119	131	151	158	165	740	747
	789	813	874	899	901	908	933	956	970	993	995	997	999	1002
	1014	1022	1024	1029	1047	1051	1090	1099	1109	1117	1123	1133	1185	1522
	1617	1623	1645	1674	1678	1686	1691	1701	1718	1745	1748	1859	1861	1867
	1879	1897	1906	1928	1934	1992	1996	2092	2181	2183	2194	2253	2261	2327
	2380	2488	2595	2657	2731	2777	2781	2783	2788	2811	2815	2820	2855	2865
	2867	2903	2957	2960	3007	3009	3060	3084	3101	3127	3143	3206	3250	3307
	3313	3388	3411	3481	3491	3537	3539	3546	3573	3582	3619	3624	3648	3653
	3663	3685	3686	3690	3692	3694	3698	3707	3764	3790	3797	3838	3843	3937
	3938	3940	3947	3952	3990	4037	4079	4135	4142	4146	4159	4162	4167	4174
	4200	4215	4220	4225	4250	4256	4298	4301	4302	4307	4317	4318	4333	4362
	4368	4370	4385	4397	4451	4452	4472	4528	4536	4589	4592	4595	4600	4604
	4615	4655	4994	5032	5075	5079	5083	5106	5113	5188	5217	5220	5226	5246
	5332	5344	5395	5403	5410	5430	5542	5546	5548	5561	5572	5599	5610	5621
	5627	5681	5736	5745	5748	5929	5936	5950	6016	6022	6167	6191	6208	6218
	6225	6247	6254	6262	6275	6282	6325	6352	6382	6384	6445	6448	6460	6484
	6512	6587	6596	6628	6635	6749	6764	6781	6787	6791	6797	6823	6832	6883
	6975	6982	7029	7072	7088	7147	7231	7303	7311	7314	7327	7329	7339	7369
	7397	7763	7764	7769	7775	7784								
LEPSKP	1896#	1904												
LESSTK	1036#	4544												
LET	1072	2738	2934	3451#										
LINCHK	3828#													
LINGET	155	1972	2107	2119	3173#	3304								
LINGT2	3174#	3624												
LINLEN	114#	2601	3091	3826	3912									
LINLIN	2067#	2517												
LINPRT	142#	2916	7437											
LINPT1	2602#													
LINPT2	3093#													
LINPT3	3827#													
LINPT4	3916#													
LIST	1120	7401#												

LIST4	7406#	7456												
LISTEN	126#	130#	2854	2995	7415									
LOG	920	920#												
LOOP	2151#	2171	3330											
LOOPDN	4432	4445#												
LOOPER	1740#	1771												
LOOPON	3598#	3611												
LOPOT2	4111#	4240												
LOPFDA	5921#	5959												
LOPFND	5632#	5671												
LOPPTA	6049#	6076												
LOPREL	4539#	4564												
LPOPER	4513#	4724	5069	5434										
LPTLEN	113#													
LPTSW	8#	163	813	912	996	1001	1047	1108	1122	1590	1940	2256	2544	2595
	2873	3742	3812	3855	3899	3957	6276	7398						
LSTOPK	1029#	4581												
M	764	767	791	884	886	1740	1749	1751	1796	1822	2033	2039	2041	2049
	2069	2071	2079	2085	2087	2154	2156	2187	2189	2299	2322	2335	2363	2378
	2385	2418	2439	2529	2803	2875	2887	2891	2896	2898	2952	2954	2962	3112
	3400	3534	3556	3560	3587	3623	3824	3850	4080	4126	4257	4319	4321	4392
	4453	4535	4571	4611	4691	4693	4699	4701	4865	4867	4916	4918	5221	5223
	5349	5351	5354	5364	5366	5368	5385	5533	5614	5730	5738	5741	5743	5897
	5937	5942	5948	5954	5956	5978	6017	6024	6026	6047	6062	6064	6092	6116
	6118	6143	6145	6404	6406	6408	6410	6443	6482	6511	6540	6612	6746	6782
	6804	6826	6828	6830	6876	6878	6885	6901	6903	6929	6935	6976	6978	6980
	7025	7027	7043	7074	7092	7112	7130	7149	7151	7191	7219	7224	7376	7377
	7380	7387	7465	7501	7768									
MAIN	1961#	1968	2073											
MAKINT	169#	750	5483	5492	6273									
MEMSIZ	153	1611#	2212	3218	6701									
MID$	952	7190#												
MID2	7202	7210#												
MINUTK	155	1028#	2114	5020										
MLOOP	7542#	7548												
MLOOPR	2047#	2054												
MORCOM	3923#	3926												
MORLIN	3177#	3204												
MORPR	3746#	3848												
MOVE	141#													
MOVFM	143#	4401												
MOVFR	142#	4696												
MOVINS	6956	6961	6973#											
MOVLF	6980#	6995												
MOVMF	143#	4413	6378											
MOVRF	142#	2793	2825	4979										
MOVRM	142#	4417	4428	6785										
MOVSTR	6499	6987#	7163											
MULDIM	39#	156	157	5754	5862	6403								
MUSTCH	2339	2346#												
NEG	145#													
NEWCHR	3745#													
NEWSTT	2853#	2926	3262	3372	4457									

NEXT	1062	4361#													
NEXTC	4367#	4461													
NFERR	1881#	4387													
NMARY1	5946	5953#													
NMARY2	5940	5952#													
NMREL	4544#	4546													
NODATT	2406	2411#													
NODEL	2011#														
NOPRIN	2613#	2619													
NOSEC	5557	5571#													
NOTABR	3779	3921	3969	3981#											
NOTDIM	6058	6062#													
NOTER	5036	5431#													
NOTFDD	5934	6015#													
NOTFNS	5640	5673#													
NOTFRF	5198	5208#													
NOTITL	5647	5652	5662#												
NOTLST	7050	7057#													
NOTOL	2749	2757#													
NOTQTI	4047	4063#													
NOTSTV	4535#														
NOTTHR	7272	7279#													
NOTTK	1025#	5034													
NOTTRC	2909	2922#													
NOWGET	4186	4196#													
NOWLIN	4305	4327#													
NTHIS	2365	2435#													
NTHIS1	2439#	2442													
NULCNT	834#	3098	3870												
NULL	1103	3085#													
NUMCMD	2936#	2938													
NUMGFN	5128#	5130													
NUMINS	4172	4211#													
NUMLEV	111#	1449													
NUMREL	4680#	4795													
NUMTMP	116#	119#	1614												
NUTPOS	3912#	3915													
NXTCON	2839#	4442													
NXTRES	2355#	2370													
ODONE	122#	2617													
OKGOTO	3657	3662#													
OKNORM	5132	5182#													
OLDLIN	1646#	3031	3077												
OLDTXT	1647#	2254	3037	3065											
OMERR	135	1857#	3238												
ONEFUN	1046#	5045	5128	5194	5200										
ONELIN	2111	2124#													
ONEUN	2809	2829#													
ONGOTO	1101	3584#													
OPRTYP	1604#	4825	4906												
OPTAB	956#	4607													
ORFIN	5475	5486#													
OUTBEL	2526	2532#													
OUTCON	816	2538#													
OUTOO	146	807#													
OUTWRD	7254#	7322													
PARCHK	5052#	5183	6350												
PEEK	934	7765#													
PLOOP	7465#	7472													
PLUSTK	155	1027#	1029	4576	5009										
POKE	1110	7776#													
POPGOF	1760	1764#													
POPHRT	160#														
POS	913	6279#													
PPSWRT	2541	6685#													
PREAM	7124	7174	7293#												
PREAM2	7193	7297#													
PRINT	1112	3747#													
PRINTC	3751#	3984													
PRINTK	998#	2331													
PRIT3	7501#	7510													
PRIT4	7464#	7505													
PRTNUL	3872#	3884													
PSW	807	1970	1980	1987	2012	2372	2587	2588	2621	2831	3019	3045	3180	3194	
	3465	3478	3525	3533	3879	3882	3937	3948	4170	4201	4394	4931	5357	5468	
	5632	5888	5977	6028	6066	6149	6160	6661	6662	6685	6688	6695	6860	6861	
	7212	7266	7487	7494											
PTRGET	3452	4114	4372	5089	5529#	6323									
PTRGT2	5534#	6449													
PUFOUT	167#														
PURE	7#														
PUSHF	141#	4736	4972												
PUSHMA	871	890#	891												
PUTDEI	6407#	6515													
PUTNEW	6578#	6969	7116	7170											
PUTVAL	3563	3567#													
Q	981#	986	986#	987	987#	988	988#	989	989#	990	990#	991	991#	992	
	992#	993	993#	994	994#	995	995#	998	998#	999	999#	1001	1001#	1002	
	1002#	1005	1005#	1011	1011#	1012	1013	1013#	1019	1019#	1020	1021	1021#	1022	
	1022#	1023	1023#	1025	1025#	1026	1026#	1027	1027#	1028	1028#	1029	1029#	1031	
	1031#	1032	1033	1033#	1035	1035#	1036	1046	1046#	1047	1047#	1048	1048#	1049	
	1049#	1050	1050#	1052	1052#	1054	1054#	1056	1056#	1177#	1187#	1194	1194#	1212	
	1214	1214#	1228	1230	1230#	1252	1254	1254#	1267	1269	1269#	1292	1294	1294#	
	1304	1306	1306#	1321	1323	1323#	1344	1346	1346#	1370	1372	1372#	1393	1395	
	1395#	1413	1415	1415#	1431	1434	1434#	1449	1451	1451#	1472	1474	1474#	1491	
	1493	1493#	1521	1524	1524#	1540	1543	1543#	1568						
QINLIN	155	2269#	4077	4153											
QINT	141#														
RAMBOT	13#														
READ	1070	4091#													
READY	135	1940#	3053	7413											
REALIO	5#	15	129	137	999	1119	2469	2474	2533	2543	2612	2628	3055	6703	
REASON	135	1789	1835	1846#	6086										
REDDY	1710#	1954	7528												
REDINP	3460#														
REM	1086	1899	3315	3392#											

```
REMER    3400#   3421    3430
REMTK     992#   2412
REPIN1    135    1956#
REPOUT   3976#   3979
RESCR1   7486#   7492
RESER    2357#   2447
RESFIN   2989#   4254
RESLST    986#   2351    7484
RESKCH   7486#   7498
RESTOR   1080    2218    2984#
RETADP   4524#   4658    5408    5454
RETURN   1084    3352#
RIGMTS    950    7173#
RND       917     917#
RNDFIX    917#
RUN      1076    3251#
RUNC     2057    2196#   3252
RUNC2    3265    3296#
SCNLIN   2098#   7403    7518
SCRATH   1130    2177#
SCRATK   1005#   2936
SCRTCH    144    2178#
SETDBL   4898#   4948
SETIO    7252    7264    7315#
SGN       897     897#
SIGN      159     849#
SIGNC     160#    854
SIGNS     154#   5393
SIN       926     926#
SINFIX    139     926#
SKPMRF   3715#   3727    3732
SKPVAR   6754    6760#
SMLVAL   6179    6186#
SNERR     135     773    1674#   2122    2884    2940    3186    3227    4557    5538    6396
SNGDBL   4893#   4940
SNGDO    4959#   4990
SNGDSP   1159#   4960
SNGFLT   6269#   7064    7094    7248    7773
SP       1734    1852    2832    2842    3295    6216
SPCTK    1020#   3766    3950
SQR       915     915#
SQRFIX    915#
SQRTK    1048#   5194
STAINP   7280#   7285    7319
START      84     732#
STEPTK   1026#   2805
STKDBL   4831   4879#
STKINI   1884   2237#
STKSNG   4846   4954#
STKTOP    147   1649#   2239    3242    3508    6237    6664    6714
STMDSP   1058#   2949
STOP     1088   3010#
STPEND   3018#   4085
```

```
STPRDY   1939#   7426
STRS      940   6459#
STR1     2418#   2432
STRAD1   6510#
STRAD2   6494    6507#   6564    7159
STRCMP   4802    5345#
STRCPY   3522    6482#   6569
STRDN2   4204    4224#
STRDON   3795    3840#
STREND    136   1668#   1828    2227    5706    5720    5925    6089    6210    6774
STRFIN   6544    6548    6554#
STRGET   6539#   6552
STRING     40#    148     937    1053    1432    1601    1617    1711    1716    1720    2206    2210    2242    2275
          2278   2289    2316    2348    2397    2413    2586    2815    3383    3437    3461    3476    3755    3789
          3804   3805    3837    3886    4009    4049    4060    4163    4284    4509    4585    5024    5093    5097
          5126   5324    6217    6403    6448    6453    7291    7327    7354
STRIN1   6504#   6942    7104
STRLIT    150   3807    6467    6532#   6620
STRLT2   4197   6536#
STRLT3   6535#   7452
STRLTI   4051   5028    6533#
STRNG    2310   2427#
STROUI    146   6618#
STROUT    144   1917    4029    4264    6619#   7531
STRPR2   6634#   6649
STRPRT   3832   3841    4057    6625#   7455
STRSIZ    110#    117#   1614    1615
STUFFH   2303   2324    2333    2344    2393#   2425
SUBFLG   1625#   2264    2735    5612    5624    6440
SVAR     6737#   6768
SVARS    6734#
TABEND   2361   2377#
TABER    3762   3768    3936#
TABTK    1012#   3760
TAN       929     929#
TANFIX    139     929#
TEMP     1633#   2208    2259    2835    2869    3016    3034    3474    4032    4376    4449
TEMP2     170   1639#   4522    4525    5072    5906    6200
TEMP3     170   1618#   4560    4573    4721    6039    6114    6801    6812
TEMP6    6030   6035#   6106
TEMPPT    149   1612#   2247    6582    6609    6723    7022    7036
TEMPST    150   1614#   2244    6719
THENTK   1023#   3660
THERR     169#   4686    4856
TOFF     1095   2185    3103#
TON      1093   3102#
TOPCON   5200#   5202
TOTK     1013#   2780
TRCFLG   1675#   2905    3105
TRMNOK   4016#   4234
TRMOK    4229   4236#
TRYAGN   3991#   4026
TRYG12   6661#   6697
```

TRYIN	2629#	2635												
TRYOUT	2592	2611#												
TSTACK	143#													
TSTOP	4527#	6964												
TTICHN	123#	2638												
TTOCHN	124#	2623												
TTYCHR	2585#													
TTYIST	3963#													
TTYPOS	836#	2597	2609	3822	3862	3874	3910	3964	6280					
TVAR	6721#	6728												
TXTTAB	144	1655#	2149	2179	2197	2986								
UMULT	157#	6069	6156											
USERR	3340#													
USINTK	1022#													
USRLOC	906#													
VAL	942	7362#												
VALINT	169#													
VALSNG	169#													
VALTYP	149	170	1602#	3463	4165	4587	4681	4834	4933	5237	5596	5931	6169	6591
VARTAB	1650#	2018	2030	2192	2221	3230	3516	5630	6735	7540	7554			
VINT	168#	902												
VMOVAF	166#	4883	4929											
VMOVE	169#	3541	6594											
VMOVFA	166#													
VMOVFM	167#	5111												
VMOVMF	167#	3571												
VNEG	167#	5077												
VPUSHD	4697	4705	4712#											
VSIGN	166#	3688#	3688											
ZERITA	6091#	6096												
ZERO	141#	7366												
ZERUER	5729#	5734												
$CODE	84	84#	676	1736	1765	1849	1948	2099	2116	2250	2799	3175	3182	3233
	4363	4781	6055	6139	6214	6710								
$START	84#	84												
.C1	135	7600#												
.P	110#	735	736#	749	750#	751	752#	774	775#	800	801#	811	812#	817
	818#	851	852#	855	856#	872	873#	877	878#	892	893#	898	899#	903
	904#	905	906#	907	908#	909	910#	911	912#	914	915#	916	917#	918
	919#	921	922#	923	924#	925	926#	927	928#	930	931#	932	933#	935
	936#	939	940#	941	942#	943	944#	945	946#	947	948#	949	950#	951
	952#	953	954#	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#	1092	1093#	1094	1095#	1096	1097#	1098
	1099#	1100	1101#	1102	1103#	1104	1105#	1106	1107#	1111	1112#	1113	1114#	1115
	1116#	1118	1119#	1121	1122#	1124	1125#	1126	1127#	1131	1132#	1135	1136#	1137
	1138#	1139	1140#	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#	1737	1738#	1761	1762#	1766	1767#	1772	1773#	1790
	1791#	1803	1804#	1829	1830#	1836	1837#	1850	1851#	1864	1865#	1870	1871#	1873
	1874#	1885	1886#	1890	1891#	1893	1894#	1896	1897#	1900	1901#	1905	1906#	1918
	1919#	1921	1922#	1927	1928#	1946	1947#	1949	1950#	1952	1953#	1955	1956#	1958
	1959#	1963	1964#	1969	1970#	1973	1974#	1977	1978#	1983	1984#	1990	1991#	1995

	1996#	2016	2017#	2019	2020#	2027	2028#	2031	2032#	2046	2047#	2055	2056#	2058
	2059#	2074	2075#	2083	2084#	2091	2092#	2100	2101#	2104	2105#	2108	2109#	2112
	2113#	2117	2118#	2120	2121#	2123	2124#	2150	2151#	2172	2173#	2180	2181#	2186
	2187#	2193	2194#	2198	2199#	2209	2210#	2213	2214#	2216	2217#	2219	2220#	2222
	2223#	2225	2226#	2228	2229#	2240	2241#	2245	2246#	2248	2249#	2251	2252#	2255
	2256#	2260	2261#	2265	2266#	2280	2281#	2293	2294#	2298	2299#	2304	2305#	2311
	2312#	2315	2316#	2319	2320#	2325	2326#	2334	2335#	2340	2341#	2345	2346#	2352
	2353#	2362	2363#	2366	2367#	2371	2372#	2389	2390#	2402	2403#	2407	2408#	2410
	2411#	2416	2417#	2422	2423#	2426	2427#	2433	2434#	2443	2444#	2448	2449#	2451
	2452#	2473	2474#	2479	2480#	2482	2483#	2487	2488#	2493	2494#	2498	2499#	2503
	2504#	2508	2509#	2513	2514#	2518	2519#	2527	2528#	2537	2538#	2542	2543#	2593
	2594#	2598	2599#	2605	2606#	2610	2611#	2620	2621#	2636	2637#	2648	2649#	2652
	2653#	2736	2737#	2739	2740#	2746	2747#	2750	2751#	2760	2761#	2766	2767#	2774
	2775#	2786	2787#	2791	2792#	2794	2795#	2800	2801#	2810	2811#	2818	2819#	2823
	2824#	2826	2827#	2836	2837#	2864	2865#	2870	2871#	2881	2882#	2885	2886#	2895
	2896#	2902	2903#	2906	2907#	2910	2911#	2917	2918#	2927	2928#	2935	2936#	2941
	2942#	2950	2951#	2973	2974#	2987	2988#	2991	2992#	3004	3005#	3017	3018#	3023
	3024#	3029	3030#	3032	3033#	3035	3036#	3038	3039#	3044	3045#	3048	3049#	3051
	3052#	3054	3055#	3066	3067#	3074	3075#	3078	3079#	3081	3082#	3087	3088#	3096
	3097#	3099	3100#	3106	3107#	3147	3148#	3151	3152#	3156	3157#	3176	3177#	3183
	3184#	3187	3188#	3205	3206#	3209	3210#	3212	3213#	3219	3220#	3228	3229#	3231
	3232#	3234	3235#	3239	3240#	3243	3244#	3247	3248#	3253	3254#	3258	3259#	3263
	3264#	3266	3267#	3282	3283#	3289	3290#	3305	3306#	3316	3317#	3320	3321#	3331
	3332#	3334	3335#	3344	3345#	3358	3359#	3366	3367#	3370	3371#	3373	3374#	3410
	3411#	3422	3423#	3431	3432#	3453	3454#	3464	3465#	3469	3470#	3475	3476#	3487
	3488#	3490	3491#	3501	3502#	3509	3510#	3514	3515#	3517	3518#	3523	3524#	3532
	3533#	3542	3543#	3553	3554#	3564	3565#	3572	3573#	3586	3587#	3593	3594#	3602
	3603#	3605	3606#	3612	3613#	3618	3619#	3632	3653#	3658	3659#	3689	3690#	3697
	3698#	3703	3704#	3706	3707#	3717	3718#	3728	3729#	3733	3734#	3737	3738#	3749
	3750#	3763	3764#	3769	3770#	3775	3776#	3780	3781#	3784	3785#	3793	3794#	3796
	3797#	3803	3804#	3808	3809#	3811	3812#	3823	3824#	3830	3831#	3833	3834#	3842
	3843#	3849	3850#	3854	3855#	3863	3864#	3871	3872#	3875	3876#	3885	3886#	3911
	3912#	3919	3920#	3922	3923#	3927	3928#	3934	3935#	3943	3944#	3956	3957#	3965
	3966#	3970	3971#	3980	3981#	3985	3986#	4018	4019#	4022	4023#	4027	4028#	4030
	4031#	4033	4034#	4045	4046#	4048	4049#	4052	4053#	4058	4059#	4074	4075#	4078
	4079#	4086	4087#	4090	4091#	4094	4095#	4099	4100#	4115	4116#	4134	4135#	4140
	4141#	4148	4149#	4154	4155#	4166	4167#	4173	4174#	4187	4188#	4198	4199#	4205
	4206#	4210	4211#	4214	4215#	4218	4219#	4230	4231#	4235	4236#	4241	4242#	4248
	4249#	4255	4256#	4261	4262#	4265	4266#	4300	4301#	4306	4307#	4316	4317#	4325
	4326#	4332	4333#	4336	4337#	4364	4365#	4373	4374#	4377	4378#	4382	4383#	4388
	4389#	4402	4403#	4409	4410#	4414	4415#	4418	4419#	4422	4423#	4429	4430#	4433
	4434#	4438	4439#	4443	4444#	4450	4451#	4456	4459#	4462	4463#	4516	4517#	4520
	4521#	4523	4524#	4526	4527#	4543	4544#	4549	4550#	4558	4559#	4561	4562#	4565
	4566#	4570	4571#	4574	4575#	4580	4581#	4599	4600#	4608	4609#	4659	4660#	4668
	4669#	4673	4674#	4682	4683#	4687	4688#	4690	4691#	4698	4699#	4706	4707#	4709
	4710#	4717	4718#	4722	4723#	4725	4726#	4734	4735#	4737	4738#	4740	4741#	4746
	4747#	4755	4756#	4760	4761#	4763	4764#	4782	4783#	4788	4789#	4793	4794#	4796
	4797#	4799	4800#	4803	4804#	4806	4807#	4826	4827#	4832	4833#	4835	4836#	4840
	4841#	4847	4848#	4853	4854#	4857	4858#	4860	4861#	4871	4872#	4881	4882#	4884
	4885#	4888	4889#	4892	4893#	4897	4898#	4900	4901#	4904	4905#	4907	4908#	4930
	4931#	4934	4935#	4941	4942#	4946	4947#	4949	4950#	4956	4957#	4961	4962#	4965
	4966#	4973	4974#	4976	4977#	4980	4981#	4984	4985#	4988	4989#	4991	4992#	5001
	5002#	5004	5005#	5007	5008#	5012	5013#	5018	5019#	5023	5024#	5029	5030#	5037

	5038#	5043	5044#	5048	5049#	5057	5058#	5070	5071#	5073	5074#	5078	5079#	5090	
	5091#	5104	5105#	5109	5110#	5112	5113#	5133	5134#	5158	5159#	5163	5164#	5167	
	5168#	5174	5175#	5181	5182#	5184	5185#	5199	5200#	5206	5207#	5211	5212#	5215	
	5216#	5238	5239#	5347	5340#	5360	5361#	5390	5391#	5394	5395#	5406	5407#	5409	
	5410#	5435	5436#	5439	5440#	5448	5449#	5455	5456#	5467	5468#	5476	5477#	5484	
	5485#	5493	5494#	5518	5519#	5532	5533#	5536	5537#	5539	5540#	5551	5552#	5555	
	5556#	5558	5559#	5564	5565#	5567	5568#	5570	5571#	5575	5576#	5597	5598#	5613	
	5614#	5620	5621#	5625	5626#	5631	5632#	5636	5637#	5641	5642#	5648	5649#	5653	
	5654#	5659	5660#	5672	5673#	5707	5708#	5715	5716#	5721	5722#	5728	5729#	5735	
	5736#	5878	5879#	5886	5887#	5902	5903#	5907	5908#	5911	5912#	5919	5920#	5926	
	5927#	5932	5933#	5935	5936#	5941	5942#	5947	5948#	5960	5961#	5964	5965#	5970	
	5971#	5983	5984#	5989	5990#	6031	6032#	6034	6035#	6040	6041#	6051	6052#	6056	
	6057#	6059	6060#	6070	6071#	6077	6078#	6084	6085#	6087	6088#	6090	6091#	6097	
	6098#	6103	6104#	6107	6108#	6115	6116#	6122	6123#	6140	6141#	6153	6154#	6157	
	6158#	6166	6167#	6170	6171#	6180	6181#	6184	6185#	6188	6189#	6201	6202#	6211	
	6212#	6215	6216#	6221	6222#	6224	6225#	6231	6232#	6235	6236#	6238	6239#	6242	
	6243#	6274	6275#	6281	6282#	6309	6310#	6313	6314#	6316	6319#	6324	6325#	6338	
	6339#	6346	6347#	6351	6352#	6376	6377#	6379	6380#	6387	6388#	6391	6392#	6397	
	6398#	6421	6422#	6431	6432#	6441	6442#	6456	6451#	6465	6466#	6468	6469#	6471	
	6472#	6474	6475#	6489	6490#	6495	6496#	6500	6501#	6506	6507#	6509	6510#	6516	
	6517#	6545	6546#	6549	6550#	6553	6554#	6558	6559#	6565	6566#	6570	6571#	6580	
	6581#	6583	6584#	6586	6587#	6592	6593#	6595	6596#	6607	6608#	6610	6611#	6621	
	6622#	6627	6628#	6631	6632#	6646	6647#	6650	6651#	6665	6666#	6669	6670#	6679	
	6680#	6682	6683#	6693	6694#	6698	6699#	6702	6703#	6708	6709#	6711	6712#	6715	
	6716#	6720	6721#	6724	6725#	6729	6730#	6732	6733#	6736	6737#	6740	6741#	6745	
	6746#	6755	6756#	6758	6759#	6769	6770#	6775	6776#	6780	6781#	6786	6787#	6796	
	6797#	6802	6803#	6813	6814#	6818	6819#	6821	6822#	6846	6847#	6896	6897#	6899	
	6900#	6909	6910#	6920	6921#	6924	6925#	6928	6929#	6933	6934#	6940	6941#	6943	
	6944#	6947	6948#	6951	6952#	6955	6956#	6959	6960#	6962	6963#	6965	6966#	6970	
	6971#	6996	6997#	7016	7017#	7019	7020#	7023	7024#	7037	7038#	7046	7047#	7051	
	7052#	7056	7057#	7065	7066#	7069	7070#	7083	7084#	7086	7087#	7095	7096#	7105	
	7106#	7108	7109#	7111	7112#	7117	7118#	7125	7126#	7134	7135#	7142	7143#	7160	
	7161#	7164	7165#	7168	7169#	7171	7172#	7175	7176#	7182	7183#	7194	7195#	7203	
	7204#	7209	7210#	7216	7217#	7241	7242#	7244	7245#	7249	7250#	7253	7254#	7265	
	7266#	7273	7274#	7278	7279#	7286	7287#	7310	7311#	7317	7318#	7320	7321#	7323	
	7324#	7332	7333#	7336	7337#	7346	7347#	7364	7365#	7367	7368#	7383	7384#	7404	
	7405#	7414	7415#	7418	7419#	7427	7428#	7434	7435#	7438	7439#	7445	7446#	7448	
	7449#	7453	7454#	7456	7457#	7459	7460#	7462	7463#	7473	7474#	7479	7480#	7485	
	7486#	7493	7494#	7499	7500#	7506	7507#	7511	7512#	7519	7520#	7524	7525#	7529	
	7530#	7532	7533#	7535	7536#	7541	7542#	7549	7550#	7555	7556#	7767	7768#	7774	
	7775#	7778	7779#	7782	7783#	7791	7792#	7800							

ACRLF	51#	1710	1715	1719	4008	4283									
ADC	3427	5397													
ADD	3824	3967	5614	6935											
ADI	5400														
ADR	748	750	897	902	904	906	908	910	913	915	917	920	922	924	
	926	929	931	934	938	940	942	944	946	948	950	952	1058	1060	
	1062	1064	1066	1068	1070	1072	1074	1076	1078	1080	1082	1084	1086	1088	
	1091	1093	1095	1097	1099	1101	1103	1105	1110	1112	1114	1117	1120	1123	
	1125	1130	1134	1136	1138	1145	1147	1149	1151	1153	1155	1157	1159	1161	
	1163	1165	1167	1169	1171	1173									
ADRP	955#	957#	961	963	965	967	969	972	974						
ANA	1923	3025	5399	5477	5480	7283									
ANI	2358	2616	2632	2640	2660	2999	3547								
CALL	1788	1834	1883	1891	1898	1916	1956	1961	1971	1975	1988	2025	2056	2106	
	2184	2217	2477	2485	2737	2744	2758	2764	2784	2789	2792	2816	2821	2824	
	2915	3002	3085	3145	3149	3210	3256	3280	3303	3314	3356	3451	3467	3488	
	3530	3540	3570	3584	3603	3616	3687	3715	3782	3791	3801	3826	3831	3941	
	4028	4050	4056	4072	4076	4113	4152	4196	4212	4298	4380	4400	4407	4412	
	4416	4420	4427	4460	4514	4518	4732	4735	4753	4791	4879	4882	4895	4898	
	4926	4954	4971	4974	4978	5002	5055	5060	5076	5088	5107	5156	5161	5172	
	5182	5345	5356	5404	5435	5437	5465	5534	5553	5565	5713	5884	6032	6068	
	6085	6155	6219	6229	6233	6307	6316	6322	6344	6349	6377	6385	6389	6463	
	6466	6469	6487	6493	6498	6504	6563	6593	6619	6625	6629	6756	6784	6897	
	6922	6926	6941	6945	6949	6957	6960	7014	7067	7081	7103	7106	7123	7140	
	7158	7162	7166	7173	7192	7207	7239	7251	7263	7276	7315	7330	7334	7362	
	7381	7402	7416	7432	7436	7443	7451	7454	7517	7522	7530	7765	7776	7788	
	7789														
CC	1993	3329	3521	5204											
CHRGET	44#	1964	1986	2355	2924	3144	3177	3214	3700	3723	3746	3982	4180	4211	
	4227	4238	4327	4459	4562	4998	5125	5498	5545	5560	5598	6203	6394	7270	
	7328	7350													
CMA	2649	3928	3966	5375	5441	5444	6670								
CMC	2166	2168	2977	3118	5391										
CMP	767	2079	2363	2423	3403	3423	4554	4612	5378	5385	5644	5650	5656	5937	
	5943	5949	5978	6546	6550	6694	7131	7219	7226						
CNC	3332	3628	3917	6560											
CNZ	1925	2118	4263	4371	5110										
COMPAR	46#	1763	1795	1853	2163	3184	3236	3321	3510	3518	5637	5732	5928	6094	
	6150	6369	6567	6600	6676	6726	6742	6777	6815	6847	6853	7031	7048	7423	
	7546														
CPI	792	1742	2300	2307	2328	2336	2341	2382	2403	2489	2494	2499	2504	2509	
	2514	2521	2589	2599	2643	2804	2877	2937	2963	2969	2974	3005	3090	3113	
	3116	3360	3406	3482	3589	3606	3648	3654	3724	3759	3765	3771	3776	3825	
	3914	3949	4038	4150	4168	4183	4231	4328	4454	4545	4550	4580	4590	4663	
	4669	4683	4774	4828	4836	4843	4849	4936	5008	5013	5019	5025	5033	5039	
	5129	5195	5201	5239	5471	5579	5584	5588	5615	5698	6554	6642	6751	6792	
	7199	7474													
CZ	2387	2603	2862	3650	3840	6556	6644	7477							
DAD	1739	1769	1832	1833	1852	2023	2752	2951	3190	3191	3192	3193	3200	3235	
	3557	4609	4863	4864	4914	5216	5669	5709	5921	6081	6109	6110	6159	6174	
	6181	6185	6189	6196	6216	6674	6763	6790	6807	6808	6888	7053	7155	7375	
DC	986	987	988	989	990	991	992	993	994	995	998	999	1001	1002	
	1005	1013	1021	1022	1023	1025	1026	1027	1028	1029	1033	1046	1047	1048	

	1049	1050	1052	1054	1056	1194	1214	1230	1254	1269	1294	1306	1323	1346
	1372	1395	1415	1434	1451	1474	1493	1524	1543	1704	1712	1721	3991	4268
DCE	1179#	1188#	1193	1213	1229	1253	1268	1293	1305	1322	1345	1371	1394	1414
	1433	1450	1473	1492	1523	1542								
DCI	982#	986	987	988	989	990	991	992	993	994	995	998	999	1001
	1002	1005	1013	1021	1022	1023	1025	1026	1027	1028	1029	1033	1046	1047
	1048	1049	1050	1052	1054	1056								
DCL	1178#	1189#	1194	1214	1230	1254	1269	1294	1306	1323	1346	1372	1395	1415
	1434	1451	1474	1493	1524	1543								
DCR	1901	1966	2467	2979	3598	3729	3872	3977	5241	5242	5243	5382	5383	6074
	6161	6634	6988	7218	7302	7495								
DCX	1799	1800	2157	2199	2468	2988	3173	3213	3338	3596	3661	3745	3787	3946
	4082	4192	4226	4237	4510	5497	5592	5729	6091	6202	6364	6365	6366	6367
	6393	6532	6875	6877	6882	6891	6906	7024	7026	7028	7042	7269	7347	
DI	732													
FSIGN	47#	2828												
IN	2613	2629	2637	2856	2996	7245	7280							
INR	1924	1965	2396	2429	2437	2531	2606	2978	3026	3089	3972	5379	5396	5587
	5591	5895	6541	6632	6987	7301								
INX	769	790	885	887	1741	1750	1752	1824	1902	2034	2037	2040	2042	2050
	2051	2060	2070	2075	2076	2077	2080	2086	2155	2188	2190	2356	2393	2395
	2427	2430	2440	2453	2455	2530	2832	2842	2886	2890	2892	2897	2953	2961
	3295	3328	3405	3503	3504	3559	4079	4308	4320	4393	4692	4694	4700	4702
	4866	4917	5222	5350	5353	5365	5367	5386	5387	5645	5654	5662	5663	5677
	5679	5680	5739	5742	5747	5938	5944	5952	5953	5955	5957	6018	6025	6027
	6042	6043	6048	6061	6063	6065	6098	6117	6119	6137	6144	6146	6405	6407
	6409	6422	6483	6539	6560	6618	6647	6675	6683	6747	6748	6750	6783	6809
	6827	6829	6831	6902	6977	6979	6992	6993	7087	7146	7150	7371	7466	7468
	7489	7507	7508	7544	7545	7552								
JC	2338	2343	2501	2591	2933	3185	3226	3701	4541	4556	4671	4999	5131	5197
	5537	5549	5556	5562	6082	6178	6677	6938	7132	7425				
JM	3154	4696	7504											
JMP	733	798	815	875	890	1770	1801	1862	2089	2170	2278	2431	2446	2535
	3052	3203	3245	3264	3342	3429	3610	3704	3735	3847	3883	3932	3983	4088
	4206	4216	4334	4441	4563	4723	4744	4761	4804	4947	4963	4989	5179	5392
	5407	5453	5482	5491	5670	5987	6272	6336	6429	6448	6514	6648	6767	6907
	6968	6994	7093	7115	7169	7180	7247	7308	7457	7509	7772			
JNC	1981	2506	2525	2939	3094	3237	3512	3920	3925	3968	4547	4851	5005	5046
	5568	6151	7547											
JNZ	772	853	1903	2053	2081	2121	2323	2364	2405	2414	2471	2540	2618	2634
	2748	2808	2883	3049	3364	3420	3485	3726	3731	3978	4020	4046	4146	4171
	4233	4239	4253	4304	4330	4386	4456	4568	4794	5646	5651	5733	5939	5945
	5958	5968	6075	6095	6120	6164	6222	6395	6551	6730	6753	6794	7049	7344
	7497													
JP	2369	2441	7471	7491										
JPO	4704	6186												
JZ	1759	1967	2014	2072	2102	2110	2302	2309	2313	2332	2360	2400	2420	2424
	2491	2496	2511	2516	2879	2893	2908	2971	3027	3072	3207	3251	3408	3591
	3600	3656	3695	3747	3761	3767	3773	3778	3794	3954	4084	4132	4185	4228
	4314	4431	4597	4665	4685	4830	4838	4845	4855	4939	5010	5016	5021	5027
	5035	5041	5388	5474	5618	5639	5657	5900	5933	5981	6057	6182	6374	6543
	6547	6605	6691	6743	6778	6816	7084	7201	7271	7284	7365	7412		
LDA	809	849	2317	2596	2646	2904	3462	3821	3869	3909	3963	4016	4138	4164
	4246	4586	4680	4833	4905	5236	5611	5930	5962	6049	6101	6105	6168	6279
LDAX	2047	2357	2367	3524	5384	5642	5649	5655	6640	6990	7178	7488	7542	
LHLD	1827	1868	1919	2017	2148	2178	2196	2211	2220	2238	2258	2772	2834	2985
	3021	3033	3064	3076	3217	3229	3287	3318	3499	3507	3515	3809	4031	4092
	4448	4524	4720	4797	4869	5071	5165	5629	5634	5705	5876	5917	5924	6113
	6199	6209	6236	6240	6419	6581	6663	6667	6700	6713	6722	6734	6738	6773
	6811	6844	6894	6918	6931	6953	7017	7021	7044	7109	7539			
LXI	1735	1764	1848	1894	1947	1953	2044	2098	2115	2243	2249	2296	2350	2449
	2480	2798	2925	2948	3046	3174	3181	3232	3261	3371	3551	3562	3852	4025
	4203	4259	4362	4606	4657	4688	4707	4715	4738	4758	4780	4786	4801	4858
	4902	4959	5209	5213	5516	5573	6054	6138	6213	6311	6472	6507	6578	6696
	6709	6718	6727	6819	6963	7063	7214	7446	7460	7483	7527	7533		
MCSSIM	84													
MOV	764	791	827	830	884	886	1740	1749	1751	1754	1755	1756	1796	1822
	1922	1978	2033	2039	2041	2049	2067	2068	2069	2085	2087	2151	2153	2154
	2187	2189	2299	2305	2321	2322	2335	2373	2378	2417	2418	2519	2520	2528
	2529	2801	2802	2803	2875	2887	2896	2898	2944	2952	2954	2962	3024	3070
	3112	3152	3188	3189	3197	3220	3222	3223	3225	3336	3337	3397	3398	3399
	3400	3428	3519	3520	3534	3554	3558	3560	3561	3587	3588	3597	3599	3623
	3953	3973	4080	4126	4181	4182	4310	4319	4321	4392	4439	4440	4453	4535
	4555	4566	4571	4583	4610	4611	4662	4691	4693	4699	4701	4712	4713	4773
	4823	4827	4841	4842	4848	4865	4867	4911	4916	4918	4919	5123	5127	5193
	5221	5223	5224	5349	5351	5354	5364	5366	5368	5370	5373	5440	5442	5443
	5445	5473	5478	5479	5481	5487	5488	5490	5533	5541	5559	5594	5628	5643
	5664	5674	5675	5724	5725	5738	5741	5743	5896	5897	5942	5948	5954	5956
	6017	6019	6024	6026	6044	6047	6053	6062	6064	6078	6079	6100	6108	6116
	6118	6143	6145	6162	6163	6172	6173	6263	6265	6266	6269	6271	6331	6332
	6404	6406	6408	6410	6423	6444	6482	6497	6511	6535	6540	6562	6612	6671
	6746	6760	6782	6804	6828	6830	6842	6843	6848	6849	6856	6857	6871	6876
	6878	6885	6889	6890	6892	6893	6901	6903	6904	6905	6929	6976	6978	6980
	6981	7025	7027	7040	7041	7043	7052	7071	7074	7092	7112	7128	7130	7135
	7149	7151	7152	7156	7157	7161	7191	7223	7224	7227	7229	7300	7342	7351
	7368	7373	7374	7376	7377	7380	7387	7409	7465	7501	7550	7551	7768	
MOVRI	60#													
MVI	1830	1857	1874	1877	1881	2269	2272	2294	2330	2385	2483	2523	2732	2806
	2839	2912	2918	2945	3062	3198	3291	3340	3353	3362	3395	3555	3711	3834
	3850	3859	3866	3974	4041	4149	4188	4190	4312	4511	4537	4742	4861	4912
	5065	5120	5431	5577	5582	5667	5730	5880	5966	5985	6020	6092	6372	6427
	6437	6533	6537	6588	6603	6672	6689	6761	6805	6886	6936	7101	7137	7153
	7197	7220	7267	7306	7439	7449								
NOP	882	883												
ORA	812	852	1757	2052	2071	2156	2312	2320	2368	2419	2439	2882	2891	2907
	3071	3153	3401	3721	3839	4019	4081	4141	4249	4257	4303	4311	4567	4594
	5371	5374	5486	5489	5965	6052	6104	6424	6443	6542	6659	6826	6872	7075
	7343	7410	7467	7490	7502									
ORI	2374													
OUT	2622	7254												
OUTCHR	45#	2271	2274	2470	2475	2534	2914	2920	3836	3865	3868	3881	3976	4151
	6641	7442												
PCHL	3566	4920	5225	6035										
POP	873	1767	1794	1837	1855	1939	1979	1980	2011	2012	2022	2028	2038	2105
	2125	2162	2164	2165	2237	2372	2377	2379	2435	2587	2621	2747	2795	2827
	2921	3018	3045	3158	3194	3202	3244	3284	3327	3367	3477	3478	3506	3511

	3533	3536	3543	3567	3576	3578	3781	3846	3882	3948	3981	4023	4059	4083
	4201	4245	4266	4316	4410	4415	4423	4424	4527	4756	4821	4868	4885	4889
	4893	4894	4915	4931	4943	4957	4958	4970	4981	4985	5086	5118	5207	5355
	5363	5369	5398	5449	5468	5470	5638	5711	5718	5737	5740	5751	5887	5888
	5908	5977	6028	6060	6072	6073	6142	6158	6160	6194	6358	6361	6381	6392
	6400	6401	6402	6411	6425	6490	6492	6501	6611	6661	6685	6688	6771	6803
	6851	6859	6860	6861	6869	6870	6900	6944	6973	7057	7091	7113	7143	7144
	7163	7176	7212	7279	7297	7298	7338	7384	7386	7401	7406	7407	7411	7424
	7441	7464	7494	7496	7520	7525	7792							
POPR	57#	4893	4957											
PUSH	807	888	1753	1792	1826	1846	1970	1974	1984	1985	1987	1991	2024	2101
	2109	2127	2159	2252	2266	2347	2353	2436	2588	2763	2771	2787	2796	2797
	2819	2829	2830	2831	2833	2841	2911	2928						

Formula Evaluation

2955	3019	3148	3179	3180	3216										
	3285	3286	3294	3296	3317	3465	3466	3480	3484	3502	3525	3565	3569	3770	
	3788	3879	3937	3951	4055	4063	4087	4091	4124	4170	4207	4262	4391	4394	
	4395	4405	4419	4513	4656	4660	4695	4703	4714	4719	4752	4757	4778	4779	
	4790	4800	4872	4910	4927	5074	5092	5124	5170	5203	5212	5356	5357	5464	
	5519	5576	5626	5632	5673	5676	5708	5712	5875	5882	5883	5893	5912	6066	
	6067	6149	6154	6314	6315	6348	6368	6371	6388	6418	6475	6486	6496	6510	
	6536	6662	6695	6699	6712	6717	6789	6822	6855	6862	6863	6864	6879	6916	
	6917	6930	6934	6952	6967	7039	7066	7129	7139	7145	7177	7196	7217	7266	
	7299	7333	7379	7405	7419	7429	7430	7482	7486	7487	7521	7526	7779		
PUSHFM	49#	6362													
PUSHM	48#	2160	2161	3505	4309	4710	4711	6356	6359	6362	6363	6491	7090	7372	
	7406	7420													
PUSHR	54#														
RAL	4552														
RC	3115	3339	4577	6850	7228										
RET	832	856	880	1038	2267	2457	2625	2653	2980	2993	3083	3100	3107	3119	
	3159	3544	3579	4034	4267	4873	5060	5087	5119	5244	5593	5752	6204	6412	
	6506	6613	6686	6865	7058	7076	7230	7256	7290	7352	7393	7556	7794		
RLC	2943	4908	5122												
RNC	797	1856	2169	2965	3178	4582	4613	4776	5380	6858	7222				
RNZ	829	1744	2177	2645	3001	3010	3014	3061	3088	3215	3352	3608	6426	7034	
	7304														
RST	1763	1795	1853	1964	1986	2113	2160	2161	2163	2271	2274	2355	2470	2475	
	2534	2779	2826	2914	2920	2924	3144	3177	3184	3214	3236	3321	3458	3505	
	3510	3518	3594	3659	3700	3723	3746	3836	3865	3868	3881	3944	3976	3982	
	4053	4111	4151	4180	4211	4227	4238	4309	4327	4459	4562	4710	4711	4998	
	5052	5058	5125	5153	5159	5498	5500	5545	5560	5598	5637	5732	5903	5928	
	6094	6150	6203	6319	6327	6329	6356	6359	6362	6363	6369	6394	6435	6491	
	6567	6600	6641	6676	6726	6742	6777	6815	6847	6853	7031	7048	7090	7205	
	7210	7270	7274	7294	7324	7328	7350	7372	7408	7420	7423	7442	7546	7787	
RZ	1768	1798	2158	2167	2929	3402	3404	3722	3751	3876	5372	5376	5499	5581	
	5586	5590	6639	6841	6873	6989	7470								
SBB	3224	5402	6667												
SHLD	870	1871	1950	2029	2191	2207	2214	2223	2226	2246	2253	2868	2900	2989	
	3015	3030	3036	3079	3241	3368	3473	4323	4375	4436	4521	4559	4572	4886	
	4890	4944	4982	4986	5102	5446	5719	5726	5905	5909	6038	6088	6584	6608	
	6680	6706	6800	7035	7054	7553									
SPHL	2241	2755	3359	4389	4445										
STA	1886	1944	2263	2291	2408	2608	2650	2734	3042	3097	3104	3861	3873	4043	
	4097	4624	4932	5530	5595	5623	6029	6439	6590	7242	7318	7321			

STAX	1797	2394	2428	2452	2454	2456	3529	6991	7469	7503	7543	7793			
SUB	826	831	3221	4425	6264	7179	7225								
SUI	2398	2411	2931	3195	3418	3923	4539	4575	5044	6175	7480				
SYNCHK	42#	2113	2779	3458	3594	3659	3944	4053	4111	5052	5058	5153	5159	5500	
	5903	6319	6327	6329	6435	7205	7210	7274	7294	7324	7787				
XCHG	1758	1762	1847	1854	2032	2084	2088	2124	2381	2392	2438	2444	2757	2899	
	2923	2947	2956	2984	2992	3075	3082	3157	3201	3240	3535	4202	4252	4322	
	4326	4435	5098	5164	5171	5175	5633	5746	5889	5894	5923	5927	6071	6080	
	6112	6198	6212	6239	6561	6666	6684	6721	6725	6737	6741	6772	6776	6810	
	6814	6956	7020	7032	7190	7293	7337	7422	7431	7537					
XRA	1887	1943	2078	2290	3041	3103	3527	3880	4096	4553	5377	5529	5540	5622	
	6270	6759	6825	7070	7126	7282									
XTHL	765	770	869	1793	1821	1825	2020	2126	2743	2768	2775	2837	3290	3374	
	3471	4108	4117	4220	4236	4403	5168	5177	5187	5879	5891	6147	6333	6354	
	6559	6852	6854	6921	6925	6948	6966	6974	7127	7213	7378	7421	7428	7536	
	7783														

```
SPOOLER RUNTIME 48 SECONDS, 310 KCS, 1637 DISK READS, 123 DISK WRITES, 386 PAGES
 **END** USER MITS  [6000,6000] JOB  F3 SEQ. 42 DATE 10-SEP-75 09:22:31 MONITOR ALBUQUERQUE SCHOOLS 507B **END**

 **END** USER MITS  [6000,6000] JOB  F3 SEQ. 42 DATE 10-SEP-75 09:22:31 MONITOR ALBUQUERQUE SCHOOLS 507B **END**

 **END** USER MITS  [6000,6000] JOB  F3 SEQ. 42 DATE 10-SEP-75 09:22:31 MONITOR ALBUQUERQUE SCHOOLS 507B **END**

 **END** USER MITS  [6000,6000] JOB  F3 SEQ. 42 DATE 10-SEP-75 09:22:31 MONITOR ALBUQUERQUE SCHOOLS 507B **END**

 **END** USER MITS  [6000,6000] JOB  F3 SEQ. 42 DATE 10-SEP-75 09:22:31 MONITOR ALBUQUERQUE SCHOOLS 507B **END**

 **END** USER MITS  [6000,6000] JOB  F3 SEQ. 42 DATE 10-SEP-75 09:22:31 MONITOR ALBUQUERQUE SCHOOLS 507B **END**

 **END** USER MITS  [6000,6000] JOB  F3 SEQ. 42 DATE 10-SEP-75 09:22:31 MONITOR ALBUQUERQUE SCHOOLS 507B **END**

 **END** USER MITS  [6000,6000] JOB  F3 SEQ. 42 DATE 10-SEP-75 09:22:31 MONITOR ALBUQUERQUE SCHOOLS 507B **END**

 **END** USER MITS  [6000,6000] JOB  F3 SEQ. 42 DATE 10-SEP-75 09:22:31 MONITOR ALBUQUERQUE SCHOOLS 507B **END**

 **END** USER MITS  [6000,6000] JOB  F3 SEQ. 42 DATE 10-SEP-75 09:22:31 MONITOR ALBUQUERQUE SCHOOLS 507B **END**

 **END** USER MITS  [6000,6000] JOB  F3 SEQ. 42 DATE 10-SEP-75 09:22:31 MONITOR ALBUQUERQUE SCHOOLS 507B **END**

 **END** USER MITS  [6000,6000] JOB  F3 SEQ. 42 DATE 10-SEP-75 09:22:31 MONITOR ALBUQUERQUE SCHOOLS 507B **END**

 **END** USER MITS  [6000,6000] JOB  F3 SEQ. 42 DATE 10-SEP-75 09:22:31 MONITOR ALBUQUERQUE SCHOOLS 507B **END**

 **END** USER MITS  [6000,6000] JOB  F3 SEQ. 42 DATE 10-SEP-75 09:22:31 MONITOR ALBUQUERQUE SCHOOLS 507B **END**

 **END** USER MITS  [6000,6000] JOB  F3 SEQ. 42 DATE 10-SEP-75 09:22:31 MONITOR ALBUQUERQUE SCHOOLS 507B **END**


000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111

FFFFFFFFFFFFFFF      444      444
FFFFFFFFFFFFFFF      444      444
FFFFFFFFFFFFFFF      444      444
FFF                  444      444
FFF                  444      444
FFF                  444      444
FFF                  444      444
FFF                  444      444
FFF                  444      444
FFFFFFFFFFFF         444444444444444
FFFFFFFFFFFF         444444444444444
FFFFFFFFFFFF         444444444444444
FFF                           444
FFF                           444
FFF                           444
FFF                           444
FFF                           444
FFF                           444
FFF                           444
FFF                           444
FFF                           444
```

Confirm plane reservation on Tue Sept. 2 or Wed Sept. 3
Make sure recieved payment in good order

(FCONPM?)

check if more than 1 person (FOUT) calls VSIGN after doing a LDA VALTYP

See if a lot of people would save if INEG preserved B,C,D,E
or just B,C

sub that does (GETTYP)

```
LDA VALTYP
CPI 4
RET
```

? (definitely Yes)

Look at [illegible] BASIC- see if so much stuff

a lot of Call FINDIV
3N → loops?

LPTSPL VERSION 6(344) RUNNING ON LPT0
 START USER MITS [6000,6000] JOB F4 SEQ. 205 DATE 27-AUG-75 07:23:23 MONITOR ALBUQUERQUE SCHOOLS 507A *START*
REQUEST CREATED: 27-AUG-75 06:11:25
FILE: DSK80:F4[6000,6000] CREATED: 27-AUG-75 06:10:00 PRINTED: 27-AUG-75 07:23:30
QUEUE SWITCHES: /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:468 /FORMS:NORMAL
FILE WILL BE DELETED AFTER PRINTING

/MEDIUM/
/ON MACHINE/
/CASSETTE/

2955	3019	3148	3179	3180	3216										
	3285	3286	3294	3296	3317	3465	3466	3480	3484	3502	3525	3565	3569	3770	
	3788	3879	3937	3951	4055	4063	4087	4091	4124	4170	4207	4262	4391	4394	
	4395	4405	4419	4513	4656	4660	4695	4703	4714	4719	4752	4757	4778	4779	
	4790	4800	4872	4910	4927	5074	5092	5124	5170	5203	5212	5356	5357	5464	
	5519	5576	5626	5632	5673	5676	5708	5712	5875	5882	5883	5893	5912	6066	
	6067	6149	6154	6314	6315	6348	6368	6371	6388	6418	6475	6486	6496	6510	
	6530	6662	6693	6699	6712	6717	6789	6822	6855	6862	6863	6864	6879	6916	
	6917	6930	6934	6952	6967	7039	7066	7129	7139	7145	7177	7196	7217	7266	
	7299	7333	7379	7405	7419	7429	7430	7482	7486	7487	7521	7526	7779		
PUSHPM	49#	6362													
PUSHM	48#	2160	2161	3505	4309	4710	4711	6356	6359	6362	6363	6491	7090	7372	
	7406	7420													
PUSHR	54#														
RAL	4552														
RC	3115	3339	4577	6850	7228										
RET	832	856	880	1838	2267	2457	2625	2653	2980	2993	3083	3100	3107	3119	
	3159	3544	3579	4034	4267	4873	5060	5087	5119	5244	5593	5752	6204	6412	
	6502	6613	6686	6865	7058	7076	7230	7256	7290	7352	7393	7556	7794		
RLC	2943	4908	5122												
RNC	797	1856	2169	2965	3178	4582	4613	4776	5380	6858	7222				
RNZ	829	1744	2177	2645	3001	3010	3014	3061	3088	3215	3352	3608	6426	7034	
	7304														
RST	1763	1795	1853	1964	1986	2113	2160	2161	2163	2271	2274	2355	2470	2475	
	2534	2779	2828	2914	2920	2924	3144	3177	3184	3214	3236	3321	3458	3505	
	3510	3518	3594	3659	3700	3723	3746	3836	3865	3868	3881	3944	3976	3982	
	4053	4111	4151	4180	4211	4227	4238	4309	4327	4459	4562	4710	4711	4998	
	5052	5058	5125	5153	5159	5498	5500	5545	5560	5598	5637	5732	5903	5928	
	6094	6150	6203	6319	6327	6329	6356	6359	6362	6363	6369	6394	6435	6491	
	6567	6600	6641	6676	6726	6742	6777	6815	6847	6853	7031	7048	7090	7205	
	7210	7270	7274	7294	7324	7328	7350	7372	7408	7420	7423	7442	7546	7787	
RZ	1768	1798	2158	2167	2929	3402	3404	3722	3751	3876	5372	5376	5499	5581	
	5586	5590	6639	6841	6873	6989	7470								
SBB	3224	5402	6267												
SHLD	870	1671	1950	2029	2191	2207	2214	2223	2226	2246	2253	2868	2900	2989	
	3015	3030	3036	3079	3241	3368	3473	4323	4375	4436	4521	4559	4572	4886	
	4890	4944	4982	4986	5102	5446	5719	5726	5905	5909	6038	6088	6584	6608	
	6680	6706	6800	7035	7054	7553									
SPHL	2241	2755	3359	4389	4445										
STA	1888	1944	2263	2291	2408	2608	2650	2734	3042	3097	3104	3861	3873	4043	
	4097	4624	4932	5530	5595	5623	6029	6439	6590	7242	7318	7321			

STAX	1797	2394	2428	2452	2454	2456	3529	6991	7469	7503	7543	7793			
SUB	826	831	3221	4425	6264	7179	7225								
SUI	2398	2411	2931	3195	3418	3923	4539	4575	5044	6175	7480				
SYNCHK	42#	2113	2174	3458	3594	3659	3944	4053	4111	5052	5058	5153	5159	5500	
	5903	6319	6327	6329	6435	7205	7210	7274	7294	7324	7787				
XCHG	1758	1762	1847	1854	2032	2084	2088	2124	2381	2392	2438	2444	2757	2899	
	2923	2947	2956	2984	2992	3075	3082	3157	3201	3240	3535	4202	4252	4322	
	4326	4435	5098	5164	5171	5175	5633	5746	5889	5894	5923	5927	6071	6080	
	6112	6198	6212	6239	6561	6666	6684	6721	6725	6737	6741	6772	6776	6810	
	6814	6956	7020	7032	7190	7293	7337	7422	7431	7537					
XRA	1887	1943	2078	2290	3041	3103	3527	3880	4096	4553	5377	5529	5540	5622	
	6270	6759	6825	7070	7126	7282									
XTHL	765	770	869	1793	1821	1825	2020	2126	2743	2768	2775	2837	3290	3374	
	3471	4108	4117	4206	4236	4403	5168	5177	5187	5879	5891	6147	6333	6354	
	6559	6852	6854	6921	6925	6948	6966	6974	7127	7213	7378	7421	7428	7536	
	7783														

```
                                00100   SEARCH  MCS808                 ;THE UNIVERSAL FILE
                                00200   SUBTTL  COMMON FILE
                                00300   SALL
                    000001      00400   LENGTH==1                       ; 0 MEANS 4K, 1 MEANS 8K, 2 MEANS 12K
                    000001      00500   REALIO==1
                    000001      00600   CASSW==1                        ;CASSETTE SWITCH (CSAVE,CLOAD)
                    000000      00700   LPTSW==0
                    000000      00800   DSKFUN==0                       ;ON TO READ/WRITE
                    000000      00900   CONSSW==0
                                01000
                    000001      01100   CONTRW==1                       ;ALLOW ^O
                                01200   IFE     REALIO,<
                                01300           LPTSW==0                ;SIMULATOR DEFAULTS
                                01400           CASSW==0
                                01500           CONSSW==0
                                01600           DSKFUN==0
                                01700           CONTRW==0>
                                01800
                                01900   IFE     LENGTH,<
                                02000           EXTFNC==0               ; ON MEANS EXTENDED FUNCTIONS
                                02100           MULDIM==0               ; ON MEANS MULTIPLE DIMENSIONED ARRAYS ALLOWED
                                02200           STRING==0               ; ON MEANS STRINGS ALLOWED
                                02300           CASSW==0
                                02400           LPTSW==0
                                02500           DSFFUN==0
                                02600           CONSSW==0
                                02700           CONTRW==0>
                                02800
                                02900   IFE     LENGTH-1,<
                    000001      03000           EXTFNC==1
                    000001      03100           MULDIM==1
                    000001      03200           STRING==1>
                                03300
                                03400   IFE     LENGTH-2,<
                                03500           EXTFNC==1
                                03600           MULDIM==1
                                03700           STRING==1>
                                03800
                                03900   DEFINE  SYNCHK(A),<RST  1
                                04000                       A>
                                04100   DEFINE  CHRGET,<RST     2>
                                04200   DEFINE  OUTCHR,<RST     3>
                                04300   DEFINE  COMPAR,<RST     4>
                                04400   DEFINE  FSIGN,<RST      5>
                                04500   DEFINE  PUSHM,<RST      6>
                                04600   DEFINE  PUSHFN,<PUSHM
                                04700                   PUSHM>
                                04800   DEFINE  ACRLF,<
                                04900           ^D13
                                05000   IFN     STRING,<^D10>>
                                05100   DEFINE  PUSHR,<
                                05200           PUSH    D
                                05300           PUSH    B>
```



```
                                05400   DEFINE  POPR,<
                                05500           POP     B
                                05600           POP     D>
                                05700   DEFINE  MOVRI(B,C,D,E),<
                                05800           XWD     ^O1000,^O001    ;; "LXI B"
                                05900           EXP     C
                                06000           EXP     B
                                06100           XWD     ^O1000,^O021    ;; "LXI D"
                                06200           EXP     E
                                06300           EXP     D>
                                06400
                                06500   IF1,<
                                06600   IFE     LENGTH,<PRINTX  /SMALL/>
                                06700   IFE     LENGTH-1,<PRINTX        /MEDIUM/>
                                06800   IFE     LENGTH-2,<PRINTX        /BIG/>
                                06900   IFE     REALIO,<PRINTX  /SIMULATE/>
                                07000   IFN     REALIO,<PRINTX  /ON MACHINE/>
                                07100   IFN     CASSW,<PRINTX   /CASSETTE/>
                                07200   IFN     LPTSW,<PRINTX   /LPT/>
                                07300   IFN     DSKFUN,<PRINTX  /DISK/>
                                07400   IFN     CONSSW,<PRINTX  /CONSOLE/>>
                                07500   PAGE
```

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

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

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

                    000000      00300           .P==0

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


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


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



```

                                01180   THE FLOATING POINT FORMAT IS AS FOLLOWS:

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

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

                                01620   ARITHMETIC ROUTINE CALLING CONVENTIONS:

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

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

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

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

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

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

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



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

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

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

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



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

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

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

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

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


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



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


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

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


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


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

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

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



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

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

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



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

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



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

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

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

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



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

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

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



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

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



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

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



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

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



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

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


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


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


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

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


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


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


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



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


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


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


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

```
 1324                                   19980   PAGE
```



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

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



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

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



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

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



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

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



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

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

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



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

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



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

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



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

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



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

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



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

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



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

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



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

```
2365  001574' 001000  000345          39580          PUSH    H               ;SAVE TEXT POINTER
2366  001575' 001000  000041          39600          LXI     H,FINEC         ;PUT FINEC ON STACK SO WE CAN JUMP
2367  001576' 000000  001621'
2368  001577' 000000  001571'
2369  001600' 001000  000343          39620          XTHL>                   ; TO IT IN LESS BYTES
2370  001601' 001000  000025          39640          DCR     D               ;SET SIGN OF EXPONENT FLAG
2371  001602' 001000  000376          39660          CPI     MINUTK          ;NEGATIVE EXPONENT?
2372  001603' 000000  000000*
2373                                   39680  IFE     STRING,<
2374                                   39700          JZ      FINEC>
2375                                   39720  IFN     STRING,<
2376  001604' 001000  000310          39740          RZ
2377  001605' 001000  000376          39760          CPI     "-"
2378  001606' 000000  000055
2379  001607' 001000  000310          39780          RZ>
2380  001610' 001000  000024          39800          INR     D               ;NO, RESET FLAG
2381                                   39820  IFN     STRING,<
2382  001611' 001000  000376          39840          CPI     "+"
2383  001612' 000000  000053
2384  001613' 001000  000310          39860          RZ>
2385  001614' 001000  000376          39880          CPI     PLUSTK          ;IGNORE "+"
2386  001615' 000000  000000*
2387                                   39900  IFE     STRING,<
2388                                   39920          JZ      FINEC>
2389                                   39940  IFN     STRING,<
2390  001616' 001000  000310          39960          RZ
2391  001617' 001000  000361          39980          POP     PSW>            ;GET FINEC OFF STACK
2392  001620' 001000  000053          40000          DCX     H               ;CHECK IF LAST CHARACTER WAS A DIGIT
2393                                   40020          ;HERE TO GET THE NEXT DIGIT OF THE EXPONENT
2394  001621' 001000  000327          40040  FINEC:  CHRGET                  ;GET NEXT CHARACTER
2395  001622' 001000  000332          40060          JC      FINEDG          ;IS IT A DIGIT?
2396  001623' 000000  001742'
2397  001624' 000000  001576'
2398  001625' 001000  000024          40080          INR     D               ;NO, EXPONENT ALL IN
2399  001626' 001000  000302          40100          JNZ     FINE            ;SET ITS SIGN
2400  001627' 000000  001641'
2401  001630' 000000  001623'
2402  001631' 001000  000257          40120          XRA     A
2403  001632' 001000  000223          40140          SUB     E
2404  001633' 001000  000137          40160          MOV     E,A
2405  001634' 001000  000014          40180          INR     C               ;MAKE SURE C IS NOT 377
2406                                   40200          ;HERE TO CHECK IF WE HAVE SEEN 2 DECIMAL POINTS AND SET THE DECIMAL
2407                                   40220          ; POINT FLAG
2408  001635' 001000  000014          40240  FINDP:  INR     C               ;DECIMAL POINTS!! -- SET FLAG
2409  001636' 001000  000312          40260          JZ      FINC            ;CONTINUE SCANNING CHARACTERS
2410  001637' 000000  001555'
2411  001640' 000000  001627'
2412                                   40280                                  ;WE DON'T WANT TWO SO END OF NUMBER
2413                                   40300          ;HERE TO MULTIPLY OR DIVIDE BY 10 THE CORRECT NUMBER OF TIMES,
2414                                   40320          ;WE HAVE ALREADY READ IN ALL THE DIGITS,
2415  001641' 001000  000345          40340  FINE:   PUSH    H               ;SAVE POINTER FOR LATER
2416  001642' 001000  000173          40360          MOV     A,E             ;EXPONENT=EXPONENT-# OF DECIMAL PLACES
2417  001643' 001000  000220          40380          SUB     B
```



```
2418  001644' 001000  000364          40400  FINE2:  CP      FINMUL          ;MULTIPLY BY TEN IF EXPONENT IS POSITIVE
2419  001645' 000000  001672'
2420  001646' 000000  001637'
2421  001647' 001000  000362          40420          JP      FINE3           ;DIVIDE BY TEN IF EXPONENT IS NEGATIVE
2422  001650' 000000  001660'
2423  001651' 000000  001645'
2424  001652' 001000  000365          40440          PUSH    PSW             ;SAVE EXPONENT
2425  001653' 001000  000315          40460          CALL    DIV10           ;DIVIDE NUMBER BY TEN
2426  001654' 000000  000637'
2427  001655' 000000  001650'
2428  001656' 001000  000361          40480          POP     PSW             ;GET EXPONENT
2429  001657' 001000  000074          40500          INR     A               ;INCREMENT IT
2430  001660' 001000  000302          40520  FINE3:  JNZ     FINE2           ;DO AGAIN IF WE ARE NOT DONE
2431  001661' 000000  001644'
2432  001662' 000000  001654'
2433                                   40540  IFE     STRING,<
2434                                   40560          POP     H>              ;GET CHARACTER POINTER
2435                                   40580  IFN     STRING,<
2436  001663' 001000  000321          40600          POP     D               ;GET CHARACTER POINTER
2437  001664' 001000  000361          40620          POP     PSW             ;GET SIGN
2438  001665' 001000  000314          40640          CZ      NEG             ;NEGATE IF NECESSARY
2439  001666' 000000  001175'
2440  001667' 000000  001661'
2441  001670' 001000  000353          40660          XCHG>                   ;GET CHARACTER POINTER IN (HL)
2442  001671' 001000  000311          40680          RET>                    ;ALL DONE
2443                                   40700  IFE     LENGTH=2,<
2444                                   40720          XCHG                    ;SAVE THE TEXT POINTER IN (DE)
2445                                   40740          LXI     B,377+$CODE     ;CLEAR FLAGS:  B=DECIMAL PLACE COUNT
2446                                   40760                                  ;C="," FLAG
2447                                   40780          MOV     H,C             ;ZERO (HL)
2448                                   40800          MOV     L,C
2449                                   40820          CALL    CONISS          ;ZERO FAC, SET VALTYP TO "INTEGER"
2450                                   40840          XCHG                    ;GET THE TEXT POINTER BACK IN (HL)
2451                                   40860          ;HERE TO CHECK FOR A DIGIT, A DECIMAL POINT, "E" OR "D"
2452                                   40880  FINC:   CHRGET                  ;GET THE NEXT CHARACTER OF THE NUMBER
2453                                   40900          JC      FINDIG          ;WE HAVE A DIGIT
2454                                   40920          CPI     "."             ;CHECK FOR A DECIMAL POINT
2455                                   40940          JZ      FINDP           ;WE HAVE ONE, I GUESS
2456                                   40960          CPI     "E"             ;CHECK FOR A SINGLE PRECISION EXPONENT
2457                                   40980          JZ      FINEX           ;WE HAVE A SINGLE PRECISION NUMBER
2458                                   41000          CPI     "D"             ;CHECK FOR A DOUBLE PRECISION EXPONENT
2459                                   41020          JNZ     FINE            ;WE DON'T HAVE ONE, THE NUMBER IS FINISHED
2460                                   41040          ORA     A               ;DOUBLE PRECISION NUMBER -- TURN OFF ZERO FLAG
2461                                   41060  FINEX:  CALL    FINFRC          ;FORCE THE FAC TO BE SNG OR DBL
2462                                   41080          PUSH    H               ;SAVE THE TEXT POINTER
2463                                   41100          LXI     H,FINEC         ;GET ADDRESS TO JUMP TO, THIS IS TO SAVE BYTES
2464                                   41120          XTHL                    ;PUT IT ON STACK AND GET TEXT POINTER
2465                                   41140          ;HERE TO CHECK FOR THE SIGN OF THE EXPONENT
2466                                   41160          CHRGET                  ;GET THE FIRST CHARACTER OF THE EXPONENT
2467                                   41180          DCR     D               ;SET SIGN OF EXPONENT TO MINUS
2468                                   41200          CPI     MINUTK          ;CHECK IF THE EXPONENT IS NEGATIVE
2469                                   41220          RZ                      ;IT IS
2470                                   41240          CPI     "-"             ;THIS IS IN CASE WE ARE CALLED BY VAL
```

```
                                        41260           RZ
                                        41280           INR     D               ;EXPONENT IS STILL POSITIVE, RESET FLAG
                                        41300           CPI     PLUSTK          ;IGNORE A LEADING PLUS SIGN
                                        41320           RZ
                                        41340           CPI     "+"
                                        41360           RZ
                                        41380           DCX     H               ;THE FIRST CHARACTER WAS NOT A SIGN, GO BACK
                                        41400                                   ; AND CHECK FOR A DIGIT
                                        41420           POP     PSW             ;POP FINEC OFF THE STACK, WE NO LONGER NEED IT
                                        41440           ;HERE TO GET THE NEXT DIGIT OF THE EXPONENT
                                        41460   FINEC:  CHRGET                  ;GET THE NEXT CHARATER
                                        41480           JC      FINEDG          ;PACK THE NEXT DIGIT INTO THE EXPONENT
                                        41500           INR     D               ;IT WAS NOT A DIGIT, PUT THE CORRECT SIGN ON
                                        41520           JNZ     FINE            ; THE EXPONENT, IT IS POSITIVE
                                        41540           XRA     A               ;THE EXPONENT IS NEGATIVE
                                        41560           SUB     E               ;NEGATE IT
                                        41580           MOV     E,A             ;SAVE IT AGAIN
                                        41600           ;HERE TO FINISH UP THE NUMBER
                                        41620   FINE:   LDA     VALTYP          ;FINISH UP -- WHAT KIND OF A NUMBER IS IT?
                                        41640           CPI     2
                                        41660           JNZ     FINEF           ;IT IS A FLOATING POINT ONE
                                        41680           ;HERE TO FINISH UP AN INTEGER
                                        41700           POP     PSW             ;IT IS AN INTEGER, GET ITS SIGN
                                        41720           XCHG                    ;SAVE THE TEXT POINTER IN (DE)
                                        41740           CZ      INEG            ;NEGATE IT IF NECESSARY
                                        41760           XCHG                    ;GET THE TEXT POINTER BACK IN (HL)
                                        41780           RET                     ;ALL DONE
                                        41800           ;HERE TO FINISH UP A FLOATING POINT NUMBER
                                        41820   FINEF:  PUSH    H               ;SAVE THE TEXT POINTER
                                        41840           MOV     A,E             ;FIND OUT HOW MANY TIMES WE HAVE TO MULTIPLY
                                        41860           SUB     B               ; OR DIVIDE BY TEN
                                        41880           ;HERE TO MULTIPLY OR DIVIDE BY TEN THE CORRECT NUMBER OF TIMES
                                        41900   FINEF2: CP      FINMUL          ;MULTIPLY IF WE HAVE TO
                                        41920           CM      FINDIV          ;DIVIDE IF WE HAVE TO
                                        41940           JNZ     FINEF2          ;MULTIPLY OR DIVIDE AGAIN IF WE ARE NOT DONE
                                        41960           ;HERE TO PUT THE CORRECT SIGN ON THE NUMBER
                                        41980           POP     D               ;GET THE TEXT POINTER
                                        42000           POP     PSW             ;GET THE SIGN
                                        42020           CZ      NEG             ;NEGATE IF NECESSARY
                                        42040           XCHG                    ;GET THE TEXT POINTER IN (HL)
                                        42060           LDA     VALTYP          ;WE WANT -32768 TO BE AN INT, BUT UNTIL NOW
                                        42080           CPI     4               ; IT WOULD BE A SNG
                                        42100           RNZ                     ;IT IS NOT SNG, SO IT IS NOT -32768
                                        42120           PUSH    H               ;WE HAVE A SNG, SAVE TEXT POINTER
                                        42140           LXI     H,POPHRT        ;GET ADDRESS THAT POP'S H OFF STACK BECAUSE
                                        42160           PUSH    H               ; CONIS2 DOES FUNNY THINGS WITH THE STACK
                                        42180           CALL    CONIS2          ;CHECK IF WE HAVE -32768
                                        42200           RET                     ;WE DON'T, POPHRT IS STILL ON THE STACK SO
                                        42220                                   ; WE CAN JUST RETURN
                                        42240
                                        42260           ;HERE TO CHECK IF WE HAVE SEEN 2 DECIMAL POINTS AND SET THE DECIMAL
                                        42280           ; POINT FLAG
                                        42300   FINDP:  INR     C               ;SET THE FLAG
```



```
                                        42320           JNZ     FINEF           ;WE HAD 2 DECIMAL POINTS, NOW WE ARE DONE
                                        42340           CALL    FINFRC          ;THIS IS THE FIRST ONE, CONVERT FAC TO SNG
                                        42360           JMP     FINC            ;CONTINUE LOOKING FOR DIGITS
                                        42380
                                        42400           ;FORCE THE FAC TO BE SNG OR DBL
                                        42420           ;IF THE ZERO FLAG IS ON, THEN FORCE THE FAC TO BE SNG
                                        42440           ;IF THE ZERO FLAG IS OFF, FORCE THE FAC TO BE DBL
                                        42460   FINFRC: PUSH    H               ;SAVE TEXT POINTER
                                        42480           PUSH    D               ;SAVE EXPONENT INFORMATION
                                        42500           PUSH    B               ;SAVE DECIMAL POINT INFORMATION
                                        42520           PUSH    PSW             ;SAVE WHAT WE WANT THE FAC TO BE
                                        42540           CZ      FRCSNG          ;CONVERT TO SNG IF WE HAVE TO
                                        42560           POP     PSW             ;GET TYPE FLAG BACK
                                        42580           CNZ     FRCDBL          ;CONVERT TO DBL IF WE HAVE TO
                                        42600           POP     B               ;GET DECIMAL POINT INFORMATION BACK
                                        42620           POP     D               ;GET EXPONENT INFORMATION BACK
                                        42640           POP     H               ;GET TEXT POINTER BACK
                                        42660           RET>                    ;ALL DONE
                                        42680
                                        42700           ;THIS SUBROUTINE MULIPLIES BY TEN ONCE,
                                        42720           ;IT IS A SUBROUTINE BECAUSE IT SAVES BYTES WHEN WE CHECK IF A IS ZERO
001672' 001000  000310  42740   FINMUL: RZ                      ;RETURN IF EXPONENT IS ZERO, ENTRY FROM FOUT
001673' 001000  000365  42760   FINMLT: PUSH    PSW             ;SAVE EXPONENT, ENTRY FROM FOUT
                        42780   IFN     LENGTH=2,<
001674' 001000  000315  42800           CALL    MUL10>          ;MULTIPLY BY TEN
001675' 000000  001106'
001676' 000000  001666'
                        42820   IFE     LENGTH=2,<
                                        42840           LDA     VALTYP          ;SEE WHAT KIND OF NUMBER WE HAVE
                                        42860           CPI     4
                                        42880           PUSH    PSW             ;SAVE THE TYPE
                                        42900           CZ      MUL10           ;WE HAVE A SNG, MULTIPLY BY 10.0
                                        42920           POP     PSW             ;GET THE TYPE BACK
                                        42940           CNZ     DMUL10>         ;WE HAVE A DBL, MULTIPLY BY 1000
001677' 001000  000361  42960           POP     PSW             ;GET EXPONENT
001700' 001000  000075  42980   DCRART: DCR     A               ;DECREASE IT
001701' 001000  000311  43000           RET                     ;ALL DONE
                                        43020
                        43040   IFE     LENGTH=2,<
                                        43060   FINDIV: PUSH    PSW             ;WE HAVE TO DIVIDE -- SAVE COUNT
                                        43080           LDA     VALTYP          ;SEE WHAT KIND OF NUMBER WE HAVE
                                        43100           CPI     4
                                        43120           PUSH    PSW             ;SAVE THE TYPE
                                        43140           CZ      DIV10           ;WE HAVE A SNG NUMBER
                                        43160           POP     PSW             ;GET THE TYPE BACK
                                        43180           CNZ     DDIV10          ;WE HAVE A DBL NUMBER
                                        43200           POP     PSW             ;GET COUNT BACK
                                        43220           INR     A               ;UPDATE IT
                                        43240           RET>
                                        43260
                                        43280           ;HERE TO PACK THE NEXT DIGIT OF THE NUMBER INTO THE FAC
                                        43300           ;WE MULTIPLY THE FAC BY TEN AND ADD IN THE NEXT DIGIT
001702'                 43320   FINDIG:
```

THIS SUBROUTINE MULTIPLIES BY TEN ONCE. IT IS USED BY PINPOINT

```
                                                43340   IFN     LENGTH=2,<
001702' 001000  000325                          43360           PUSH    D               ;DIGITS: SAVE EXPONENT INFORMATION
001703' 001000  000127                          43380           MOV     D,A             ;PROTECT DIGIT FROM BELOW
001704' 001000  000170                          43400           MOV     A,B             ;INCREMENT DECIMAL PLACE COUNT
001705' 001000  000211                          43420           ADC     C               ; IF PAST THE DECIMAL POINT
001706' 001000  000107                          43440           MOV     B,A
001707' 001000  000305                          43460           PUSH    B               ;SAVE NECESSARY DATA
001710' 001000  000345                          43480           PUSH    H
001711' 001000  000325                          43500           PUSH    D               ;SAVE DIGIT
001712' 001000  000315                          43520           CALL    MUL10           ;MULTIPLY OLD NUMBER BY 10
001713' 000000  001106'
001714' 000000  001675'
001715' 001000  000361                          43540           POP     PSW             ;GET NEXT DIGIT
001716' 001000  000326                          43560           SUI     "0"             ;SUBTRACT OFF ASCII CODE
001717' 000000  000060
                                                43580   IFE     EXTFNC,<
                                                43600           CALL    PUSHF           ;PUT NUMBER ON STACK
                                                43620           CALL    FLOAT           ;CONVERT TO FLOATING POINT NUMBER
                                                43640           POPR
                                                43660           CALL    FADD>           ;ADD IN NEXT DIGIT
                                                43680   IFN     EXTFNC,<
001720' 001000  000315                          43700           CALL    FINLOG>
001721' 000000  001731'
001722' 000000  001713'
001723' 001000  000341                          43720           POP     H               ;RECALL DATA
001724' 001000  000301                          43740           POP     B
001725' 001000  000321                          43760           POP     D
001726' 001000  000303                          43780           JMP     FINC>           ;GET NEXT CHARACTER
001727' 000000  001555'
001730' 000000  001721'
                                                43800   IFE     LENGTH=2,<
                                                43820           PUSH    D               ;SAVE EXPONENT INFORMATION
                                                43840           MOV     A,B             ;INCREMENT DECIMAL PLACE COUNT IF WE ARE
                                                43860           ADC     C               ; PAST THE DECIMAL POINT
                                                43880           MOV     B,A
                                                43900           PUSH    B               ;SAVE DECIMAL POINT INFORMATION
                                                43920           PUSH    H               ;SAVE TEXT POINTER
                                                43940           MOV     A,M             ;GET THE DIGIT
                                                43960           SUI     "0"             ;CONVERT IT TO ASCII
                                                43980           PUSH    PSW             ;SAVE THE DIGIT
                                                44000           LDA     VALTYP          ;SEE WHAT KIND OF A NUMBER WE HAVE
                                                44020           CPI     4
                                                44040           JNC     FINDGV          ;WE DO NOT HAVE AN INTEGER
                                                44060           ;HERE TO PACK THE NEXT DIGIT OF AN INTEGER
                                                44080           LHLD    FACLO           ;WE HAVE AN INTEGER, GET IT IN (HL)
                                                44100           LXI     D,-03277+$CODE  ;SEE IF WE WILL OVERFLOW
                                                44120           COMPAR                  ;COMPAR RETURNS WITH CARRY ON IF
                                                44140           JNC     FINDG2          ; (HL) .LT. (DE), SO THE NUMBER IS TOO BIG
                                                44160           MOV     D,H             ;COPY (HL) INTO (DE)
                                                44180           MOV     E,L
                                                44200           DAD     H               ;MULTIPLY (HL) BY 2
                                                44220           DAD     H               ;MULTIPLY (HL) BY 2, (HL) NOW IS 4*(DE)
                                                44240           DAD     D               ;ADD IN OLD (HL) TO GET 5*(DE)
```



```
                                                44260           DAD     H               ;MULTIPLY BY 2 TO GET TEN TIMES THE OLD (HL)
                                                44280           POP     PSW             ;GET THE DIGIT
                                                44300           MOV     C,A             ;SAVE IT SO WE CAN USE DAD, B IS ALREADY ZERO
                                                44320           DAD     B               ;ADD IN THE NEXT DIGIT
                                                44340           MOV     A,H             ;CHECK FOR OVERFLOW
                                                44360           ORA     A               ;OVERFLOW OCCURED IF THE MSB IS ON
                                                44380           JM      FINDG1          ;WE HAVE OVERFLOW!!
                                                44400           SHLD    FACLO           ;EVERYTHING IS FINE, STORE THE NEW NUMBER
                                                44420   FINDGE: POP     H               ;ALL DONE, GET TEXT POINTER BACK
                                                44440           POP     B               ;GET DECIMAL POINT INFORMATION BACK
                                                44460           POP     D               ;GET EXPONENT INFORMATION BACK
                                                44480           JMP     FINC            ;GET THE NEXT CHARACTER
                                                44500           ;HERE TO HANDLE 32768, 32769
                                                44520   FINDG1: MOV     A,C             ;GET THE DIGIT
                                                44540           PUSH    PSW             ;PUT IT BACK ON THE STACK
                                                44560           ;HERE TO CONVERT THE INTEGER DIGITS TO SINGLE PRECISION DIGITS
                                                44580   FINDG2: CALL    CONSI           ;CONVERT THE INTEGER TO SINGLE PRECISION
                                                44600           XRA     A               ;DO NOT TAKE THE FOLLOWING JUMP
                                                44620           ;HERE TO DECIDE IF WE HAVE A SINGLE OR DOUBLE PRECISION NUMBER
                                                44640   FINDGV: JNZ     FINDGD          ;FALL THROUGH IF VALTYP WAS 4 I,E, SNG PREC
                                                44660           MOVRI   224,164,044,000 ;GET 1000000, DO WE HAVE 7 DIGITS ALREADY?
                                                44680           CALL    FCOMP           ;IF SO, FAC .GE. 1000000
                                                44700           JP      FINDG3          ;WE DO, CONVERT TO DOUBLE PRECISION
                                                44720           POP     PSW             ;GET THE NEXT DIGIT
                                                44740           CALL    FINLOG          ;PACK IT INTO THE FAC
                                                44760           JMP     FINDGE          ;GET FLAGS OFF STACK AND WE ARE DONE
                                                44780           ;HERE TO CONVERT A 7 DIGIT SINGLE PRECISION NUMBER TO DOUBLE PRECISION
                                                44800   FINDG3: CALL    CONDS           ;CONVERT SINGLE TO DOUBLE PRECISION
                                                44820           ;HERE TO PACK IN THE NEXT DIGIT OF A DOUBLE PRECISION NUMBER
                                                44840   FINDGD: CALL    DMUL10          ;MULTIPLY THE FAC BY 10
                                                44860           CALL    VMOVAF          ;SAVE THE FAC IN ARG
                                                44880           POP     PSW             ;GET THE NEXT DIGIT
                                                44900           CALL    FLOAT           ;CONVERT THE DIGIT TO SINGLE PRECISION
                                                44920           CALL    CONDS           ;NOW, CONVERT THE DIGIT TO DOUBLE PRECISION
                                                44940           CALL    DADD            ;ADD IN THE DIGIT
                                                44960           JMP     FINDGE>         ;GET THE FLAGS OFF THE STACK AND WE ARE DONE
                                                45000   IFN     EXTFNC,<
                                                45020           ;SUBROUTINE FOR FIN, LOG
001731' 001000  000315                          45040   FINLOG: CALL    PUSHF           ;SAVE FAC ON STACK
001732' 000000  001205'
001733' 000000  001727'
001734' 001000  000315                          45060           CALL    FLOAT           ;CONVERT A TO A FLOATING POINT NUMBER
001735' 000000  001150'
001736' 000000  001732'
                                                45080   IFN     LENGTH=2,<
001737' 001000  000303                          45100           JMP     FADDT>          ;ADD IT IN
001740' 000000  000023'
001741' 000000  001735'
                                                45120   IFE     LENGTH=2,<
                                                45140           POPR                    ;GET PREVIOUS NUMBER OFF STACK
                                                45160           JMP     FADD>>          ;ADD IT IN
```

```
                                        45200           ;HERE WE PACK IN THE NEXT DIGIT OF THE EXPONENT
                                        45220           ;WE MUTIPLY THE OLD EXPONENT BY TEN AND ADD IN THE NEXT DIGIT
                                        45240           ;NOTE: EXPONENT OVERFLOW IS NOT CHECKED FOR
001742' 001000  000173                  45260   FINEDG: MOV     A,E             ;EXPONENT DIGIT -- MULTIPLY EXPONENT BY 10
001743' 001000  000007                  45280           RLC                     ;FIRST BY 4
001744' 001000  000007                  45300           RLC
001745' 001000  000203                  45320           ADD     E               ;ADD 1 TO MAKE 5
001746' 001000  000007                  45340           RLC                     ;NOW DOUBLE TO GET 10
001747' 001000  000206                  45360           ADD     M               ;ADD IT IN
001750' 001000  000326                  45380           SUI     "0"             ;SUBTRACT OFF ASCII CODE
001751' 000000  000060
001752' 001000  000137                  45400           MOV     E,A             ;STORE EXPONENT
001753' 001000  000303                  45420           JMP     FINEC           ;CONTINUE
001754' 000000  001621'
001755' 000000  001740'
                                        45440   PAGE
```



```
                                        45460   SUBTTL  FLOATING POINT OUTPUT ROUTINE
                                        45480           ;ENTRY TO LINPRT
001756' 001000  000345                  45500   INPRT:  PUSH    H               ;SAVE LINE NUMBER
001757' 001000  000041                  45520           LXI     H,INTXT##       ;PRINT MESSAGE
001760' 000000  000000*
001761' 000000  001754'
001762' 001000  000315                  45540           CALL    STROUT
001763' 000000  000000*
001764' 000000  001760'
001765' 001000  000341                  45560           POP     H               ;FALL INTO LINPRT

                                        45620           ;PRINT THE 2 BYTE NUMBER IN H,L
                                        45640           ;ALTERS ALL REGISTERS
001766'                                 45660   LINPRT:
                                        45680   IFN     LENGTH-2,<
001766' 001000  000353                  45700           XCHG                    ;SET UP REGISTERS FOR FLOATR
001767' 001000  000257                  45720           XRA     A
001770' 001000  000006                  45740           MVI     B,230
001771' 000000  000230
001772' 001000  000315                  45760           CALL    FLOATR#         ;CONVERT TO FLOATING POINT
001773' 000000  001155'
001774' 000000  001763'
                                        45780   IFE     LENGTH-2,<
                                        45800           CALL    CONISS          ;PUT THE LINE NUMBER IN THE FAC AS AN INTEGER
                                        45820           XRA     A               ;SET FORMAT TO FREE FORMAT
                                        45840           CALL    FOUINI#         ;SET UP THE SIGN
001775' 001000  000041                  45860           LXI     H,STROUI##      ;PUT PRINT STRING ADDRESS ON STACK SO WE WILL
001776' 000000  000000*
001777' 000000  001773'
002000' 001000  000345                  45880           PUSH    H               ; RETURN TO IT AND DO AN "INX   H"
                                        45900           ;THIS GETS RID OF THE SPACE FOR THE SIGN AT
                                        45920           ; THE BEGINNING OF A LINE NUMBER
                                        45940           ;FALL INTO FOUT
                                        45960   IFE     LENGTH-2,<
                                        45980           PUSH    B               ;PUT DUMMY FIELD LENGTHS ON STACK
                                        46000           JMP     FOUT2           ;PRINT THE NUMBER

                                        46060           ;FLOATING OUTPUT OF FAC
                                        46080           ;ALTERS ALL REGISTERS
                                        46100           ;THE ORIGINAL CONTENTS OF THE FAC IS LOST
                                        46120   IFN     LENGTH-2,<
002001' 001000  000041                  46140   FOUT:   LXI     H,FBUFFR        ;GET BEGINING OF CHARACTER BUFFER
002002' 000000  000000*
002003' 000000  001776'
002004' 001000  000345                  46160           PUSH    H               ;SAVE IT FOR WHEN WE RETURN
                                        46180           ;PUT THE SIGN OF THE NUMBER IN THE BUFFER AND MAKE IT POSITIVE
002005' 001000  000357                  46200           FSIGN                   ;GET SIGN OF NUMBER
002006' 001000  000066                  46220           MVI     M," "           ;PRINT SPACE IF POSITIVE
002007' 000000  000040
002010' 001000  000362                  46240           JP      FOUT1
002011' 000000  002015'
```

```
002012' 000000  002002'
002013' 001000  000066        46260              MVI     M,"-"              ;PRINT A MINUS SIGN IF NEGATIVE
002014' 000000  000055
002015' 001000  000043        46280     FOUT1:   INX     H                  ;INCREMENT POINTER TO NEXT CHRACTER POSITION
002016' 001000  000066        46300              MVI     M,"0"              ;PUT A ZERO IN BUFFER IN CASE NUMBER=0
002017' 000000  000060
002020' 001000  000312        46320              JZ      FOUT19             ;DO IT IF THE NUMBER IS ZERO
002021' 000000  002266'
002022' 000000  002011'
002023' 001000  000345        46340              PUSH    H                  ;SAVE BUFFER POINTER
002024' 001000  000374        46360              CM      NEG                ;NEGATE NUMBER IF NEGATIVE
002025' 000000  001175'
002026' 000000  002021'

                              46400     ;HERE WE GET THE FAC IN THE RANGE 100000 .LE. FAC .LE. 999999 AND ROUND IT TO
                              46420     ;AN INTEGER.  WE KEEP A COUNT OF HOW MANY TIMES WE MULTIPLY OR DIVIDE BY TEN
                              46440     ;SO WE KNOW WHAT THE EXPONENT WILL BE.  THE FAC IS THEN CONVERTED TO AN
                              46460     ;INTEGER IN C,D,E.  WE USE A TABLE OF POWERS OF TEN TO CALCULATE EACH DIGIT.
                              46480     ;THIS ALGORITHM IS USED FOR SPEED.
002027' 001000  000257        46500              XRA     A                  ;PUT TEN'S EXPONENT COUNT ON STACK
002030' 001000  000365        46520              PUSH    PSW
002031' 001000  000315        46540              CALL    FOUTCB             ;SEE IF NUMBER IS TOO BIG OR TOO SMALL
002032' 000000  002274'
002033' 000000  002025'
002034' 001000  000001        46560     FOUT3:   MOVRI   221,103,117,370 ;IS NUMBER .LE. 99999.9499?  IT IS TOO SMALL
002035' 000000  000103
002036' 000000  000221
002037' 001000  000021
002040' 000000  000370
002041' 000000  000117
002042' 001000  000315        46580              CALL    FCOMP              ;FCOMP RETURNS 377, 0 OR 1 IN A, SO THE
002043' 000000  001317'
002044' 000000  002032'
                              46600                                         ; PARITY WILL BE ODD IFF 1 IS RETURNED
002045' 001000  000342        46620              JPO     FOUT5              ;NO, NUMBER IS IN RANGE
002046' 000000  002071'
002047' 000000  002043'
002050' 001000  000361        46640              POP     PSW
002051' 001000  000315        46660              CALL    FINMLT             ;YES, MULTIPLY IT BY TEN TO GET
002052' 000000  001673'
002053' 000000  002046'
002054' 001000  000365        46680              PUSH    PSW                ; IT IN RANGE
002055' 001000  000303        46700              JMP     FOUT3              ;SEE IF NUMBER IS NOW IN RANGE
002056' 000000  002034'
002057' 000000  002052'
002060' 001000  000315        46720     FOUT9:   CALL    DIV10              ;NO, DIVIDE NUMBER BY TEN, IT IS TOO BIG
002061' 000000  000637'
002062' 000000  002056'
002063' 001000  000361        46740              POP     PSW                ;ADD ONE TO EXPONENT
002064' 001000  000074        46760              INR     A
002065' 001000  000365        46780              PUSH    PSW
002066' 001000  000315        46800              CALL    FOUTCB             ;IS NUMBER .LE. 999999.499?
002067' 000000  002274'
```



```
002070' 000000  002061'
                              46820                                         ;YES, NUMBER IS IN PRINTING RANGE, I.E.
                              46840                                         ; ALL DIGITS TO BE PRINTED ARE THE INTEGER PART

002071' 001000  000315        46860     FOUT5:   CALL    FADDH              ;ROUND NUMBER TO NEAREST INTEGER
002072' 000000  000000'
002073' 000000  002067'
002074' 001000  000074        46880              INR     A                  ;MAKE A NON-ZERO, SINCE NUMBER IS POSITIVE
                              46900                                         ; AND NON-ZERO, ROUND WILL EXIT WITH THE HO
                              46920                                         ; IN A, SO THE MSB WILL ALWAYS BE ZERO AND
                              46940                                         ; ADDING ONE WILL NEVER CAUSE A TO BE ZERO
002075' 001000  000315        46960              CALL    QINT               ;GET INTEGER PART IN C,D,E
002076' 000000  001372'
002077' 000000  002072'
002100' 001000  000315        46980              CALL    MOVFR              ;SAVE NUMBER IN FAC
002101' 000000  001225'
002102' 000000  002076'
                              47000     ;DECIDE IF THE NUMBER SHOULD BE PRINTED IN FIXED OR FLOATING NOTATION
002103' 001000  000001        47020              LXI     B,2*400+6+$CODE ;SET DECIMAL POINT COUNT FOR E NOTATION
002104' 000000  001006*
002105' 000000  002101'
                              47040                                         ;C = DIGIT COUNT
002106' 001000  000361        47060              POP     PSW                ;GET EXPONENT
                              47080              ADD     C                  ;~~SEE IF~~ NUMBER ~~SHOULD~~ BE PRINTED IN E NOTATION?
                                                                              SHOULD The
002107' 001000  000201
002110' 001000  000372        47100              JM      FOUT6              ;YES, IT IS .LT. .1
002111' 000000  002124'
002112' 000000  002104'
002113' 001000  000376        47120              CPI     7
002114' 000000  000007
002115' 001000  000322        47140              JNC     FOUT6              ;YES,  IT IS .GT. 999999
002116' 000000  002124'
002117' 000000  002111'
002120' 001000  000074        47160              INR     A
002121' 001000  000107        47180              MOV     B,A                ;B = DECIMAL POINT COUNT
002122' 001000  000076        47200              MVI     A,1                ;SET FIXED POINT FLAG, THE EXPONENT IS ZERO
002123' 000000  000001
                              47220                                         ; IF WE ARE USING FIXED POINT NOTATION
002124' 001000  000075        47240     FOUT6:   DCR     A                  ;E NOTATION: ADD 5 TO ORIGINAL EXPONENT
002125' 001000  000341        47260              POP     H                  ;GET BUFFER POINTER FROM STACK
002126' 001000  000365        47280              PUSH    PSW                ;SAVE EXPONENT FOR LATER
                              47300     ;CALCULATE THE DIGITS OF THE NUMBER
002127' 001000  000021        47320              LXI     D,FOUTBL           ;STORE LOC OF LARGEST POWER OF TEN
002130' 000000  002316'
002131' 000000  002116'
002132' 001000  000005        47340     FOUT8:   DCR     B                  ;SEE IF IT IS TIME TO PRINT A DECIMAL POINT
002133' 001000  000066        47360              MVI     M,"."              ;PUT A DECIMAL POINT IN THE BUFFER
002134' 000000  000056
002135' 001000  000314        47380              CZ      INXHRT             ;INCREMENT THE BUFFER POINTER IF IT IS TIME
002136' 000000  001252'
002137' 000000  002130'
002140' 001000  000305        47400              PUSH    B                  ;SAVE FLAGS
002141' 001000  000345        47420              PUSH    H                  ;SAVE CHARACTER POINTER
```

```
2858  002142' 001000  000325          47440          PUSH    D               ;SAVE POWER OF TEN POINTER
2859  002143' 001000  000315          47460          CALL    MOVRF           ;GET NUMBER IN C,D,E
2860  002144' 000000  001240'
2861  002145' 000000  002136'
2862  002146' 001000  000341          47480          POP     H               ;GET POWER OF TEN POINTER
2863  002147' 001000  000006          47500          MVI     B,"0"-1         ;B = NEXT DIGIT TO BE PRINTED
2864  002150' 000000  000057
2865  002151' 001000  000004          47520  FOUT10: INR     B               ;ADD ONE TO DIGIT
2866  002152' 001000  000173          47540          MOV     A,E             ;SUBTRACT LO
2867  002153' 001000  000226          47560          SUB     M
2868  002154' 001000  000137          47580          MOV     E,A
2869  002155' 001000  000043          47600          INX     H               ;POINT TO NEXT BYTE OF POWER OF TEN
2870  002156' 001000  000172          47620          MOV     A,D             ;SUBTRACT MD
2871  002157' 001000  000236          47640          SBB     M
2872  002160' 001000  000127          47660          MOV     D,A
2873  002161' 001000  000043          47680          INX     H
2874  002162' 001000  000171          47700          MOV     A,C             ;SUBTRACT MO
2875  002163' 001000  000236          47720          SBB     M
2876  002164' 001000  000117          47740          MOV     C,A
2877  002165' 001000  000053          47760          DCX     H               ;POINT TO BEGINNING OF POWER OF TEN
2878  002166' 001000  000053          47780          DCX     H
2879  002167' 001000  000322          47800          JNC     FOUT10          ;SUBTRACT AGAIN IF RESULT WAS POSITIVE
2880  002170' 000000  002151'
2881  002171' 000000  002144'
2882  002172' 001000  000315          47820          CALL    FADDA           ;IT WASN'T, ADD POWER OF TEN BACK IN
2883  002173' 000000  000274'
2884  002174' 000000  002170'
2885  002175' 001000  000043          47840          INX     H               ;INCREMENT POINTER TO NEXT POWER OF TEN
2886  002176' 001000  000315          47860          CALL    MOVFR           ;SAVE C,D,E IN FAC
2887  002177' 000000  001225'
2888  002200' 000000  002173'
2889  002201' 001000  000353          47880          XCHG                    ;GET POWER OF TEN POINTER IN (DE)
2890  002202' 001000  000341          47900          POP     H               ;GET BUFFER POINTER
2891  002203' 001000  000160          47920          MOV     M,B             ;PUT CHARACTER IN BUFFER
2892  002204' 001000  000043          47940          INX     H               ;INCREMENT BUFFER POINTER
2893  002205' 001000  000301          47960          POP     B               ;GET COUNTERS OFF STACK
2894  002206' 001000  000015          47980          DCR     C               ;WAS THAT THE LAST DIGIT?
2895  002207' 001000  000302          48000          JNZ     FOUT8           ;DO MORE IF NOT
2896  002210' 000000  002132'
2897  002211' 000000  002177'
2898  002212' 001000  000005          48020          DCR     B               ;SEE IF DECIMAL POINT GOES AFTER LAST DIGIT
2899  002213' 001000  000312          48040          JZ      FOUT12          ;IT DOES, WE HAVE NO ZEROS TO SUPPRESS
2900  002214' 000000  002232'
2901  002215' 000000  002210'
2902                                   48060          ;SUPPRESS THE TRAILING ZEROS
2903  002216' 001000  000053          48080  FOUT11: DCX     H               ;GO BACK TO LAST CHARACTER
2904  002217' 001000  000176          48100          MOV     A,M             ;GET IT
2905  002220' 001000  000376          48120          CPI     "0"             ;IGNORE TRAILING ZEROS
2906  002221' 000000  000060
2907  002222' 001000  000312          48140          JZ      FOUT11
2908  002223' 000000  002216'
2909  002224' 000000  002214'
2910                                   48160          ;SUPPRESS DECIMAL POINT IF WE HAVE AN INTEGER
```



```
2911  002225' 001000  000376          48180          CPI     "."             ;IGNORE DECIMAL POINT BEFORE TRAILING ZEROS
2912  002226' 000000  000056
2913  002227' 001000  000304          48200          CNZ     INXHRT          ;IF NO DP, MOVE POINTER TO NEXT POSITION
2914  002230' 000000  001252'
2915  002231' 000000  002223'
2916  002232' 001000  000361          48220  FOUT12: POP     PSW             ;GET DECIMAL EXPONENT
2917  002233' 001000  000312          48240          JZ      FOUT17          ;RETURN IF NUMBER WAS IN FIXED POINT FORMAT
2918  002234' 000000  002271'
2919  002235' 000000  002230'
2920                                   48260          ;FLOATING POINT NOTATION -- PUT AN "E" IN THE BUFFER
2921  002236' 001000  000066          48280          MVI     M,"E"           ;PUT AN "E" IN THE BUFFER
2922  002237' 000000  000105
2923  002240' 001000  000043          48300          INX     H               ;PUT SIGN OF EXPONENT IN BUFFER
2924                                   48320          ;PUT IN THE SIGN OF THE EXPONENT
2925  002241' 001000  000066          48340          MVI     M,"+"           ;A PLUS IF POSITIVE
2926  002242' 000000  000053
2927  002243' 001000  000362          48360          JP      FOUT14
2928  002244' 000000  002252'
2929  002245' 000000  002234'
2930  002246' 001000  000066          48380          MVI     M,"-"           ;A MINUS IF NEGATIVE
2931  002247' 000000  000055
2932  002250' 001000  000057          48400          CMA                     ;NEGATE EXPONENT
2933  002251' 001000  000074          48420          INR     A
2934                                   48440          ;CALCULATE THE TWO DIGIT EXPONENT
2935  002252' 001000  000006          48460  FOUT14: MVI     B,"0"-1         ;INITIALIZE TEN'S DIGIT COUNT
2936  002253' 000000  000057
2937  002254' 001000  000004          48480  FOUT15: INR     B               ;INCREMENT DIGIT
2938  002255' 001000  000326          48500          SUI     12              ;SUBTRACT TEN
2939  002256' 000000  000012
2940  002257' 001000  000322          48520          JNC     FOUT15          ;DO IT AGAIN IF RESULT WAS POSITIVE
2941  002260' 000000  002254'
2942  002261' 000000  002244'
2943  002262' 001000  000306          48540          ADI     "0"+12          ;ADD BACK IN TEN AND CONVERT TO ASCII
2944  002263' 000000  000072
2945                                   48560          ;PUT THE EXPONENT IN THE BUFFER
2946  002264' 001000  000043          48580          INX     H
2947  002265' 001000  000160          48600          MOV     M,B             ;PUT TEN'S DIGIT OF EXPONENT IN BUFFER
2948  002266' 001000  000043          48620  FOUT19: INX     H               ;WHEN WE JUMP TO HERE, A IS ZERO
2949  002267' 001000  000167          48640          MOV     M,A             ;PUT ONE'S DIGIT IN BUFFER
2950  002270' 001000  000043          48660          INX     H               ;INCREMENT POINTER
2951  002271' 001000  000161          48680  FOUT17: MOV     M,C             ;PUT ZERO AT END OF BUFFER
2952  002272' 001000  000341          48700          POP     H               ;EXIT WITH (HL) POINTING TO STRING
2953  002273' 001000  000311          48720          RET                     ;ALL DONE
2954
2955                                   48760          ;SEE IF FAC ,LE, 999999,499
2956  002274' 001000  000001          48780  FOUTC8: MOVRI   224,164,043,367 ;COMPARE NUMBER WITH CONSTANT
2957  002275' 000000  000164
2958  002276' 000000  000224
2959  002277' 001000  000021
2960  002300' 000000  000367
2961  002301' 000000  000043
2962  002302' 001000  000315          48800          CALL    FCOMP
2963  002303' 000000  001317'
```

```
 2964  002304' 000000  002260'
 2965  002305' 001000  000341          48820          POP     H               ;GET RETURN ADDRESS OFF STACK
 2966  002306' 001000  000342          48840          JPO     FOUT9           ;NUMBER TOO BIG, DIVIDE BY TEN
 2967  002307' 000000  002260'
 2968  002310' 000000  002303'
 2969  002311' 001000  000351          48860          PCHL                    ;NUMBER OK, RETURN
 2970
 2971                                   48900          ;CONSTANTS FOR FOUT
 2972  002312' 000000  000000          48920  FHALF:  000     ; 1/2
 2973  002313' 000000  000000          48940          000     ;THIS CONSTANT IS ALSO USED BY SQR, SIN, COS
 2974  002314' 000000  000000          48960          000
 2975  002315' 000000  000200          48980          200
 2976                                   49000          ;POWER OF TEN TABLE
 2977  002316' 000000  000240          49020  FOUT8L: 240     ; 100000
 2978  002317' 000000  000206          49040          206
 2979  002320' 000000  000001          49060          001
 2980  002321' 000000  000020          49080          020     ; 10000
 2981  002322' 000000  000047          49100          047
 2982  002323' 000000  000000          49120          000
 2983  002324' 000000  000350          49140          350     ; 1000
 2984  002325' 000000  000003          49160          003
 2985  002326' 000000  000000          49180          000
 2986  002327' 000000  000144          49200          144     ; 100
 2987  002330' 000000  000000          49220          000
 2988  002331' 000000  000000          49240          000
 2989  002332' 000000  000012          49260          012     ; 10
 2990  002333' 000000  000000          49280          000
 2991  002334' 000000  000000          49300          000
 2992  002335' 000000  000001          49320          001     ; 1
 2993  002336' 000000  000000          49340          000
 2994  002337' 000000  000000          49360          000>
 2995                                   49380  IFE     LENGTH=2,<
 2996                                   49400          ;OUTPUT THE VALUE IN THE FAC ACCORDING TO THE FORMAT SPECIFICATIONS
 2997                                   49420          ; IN A,B,C
 2998                                   49440          ;ALL REGISTERS ARE ALTERED
 2999                                   49460          ;THE ORIGINAL CONTENTS OF THE FAC IS LOST
 3000                                   49480
 3001                                   49500          ;THE FORMAT IS SPECIFIED IN A, B AND C AS FOLLOWS:
 3002                                   49520          ;THE BITS OF A MEAN THE FOLLOWING:
 3003                                   49540  ;BIT 7  0 MEANS FREE FORMAT OUTPUT, I.E, THE OTHER BITS OF A MUST BE ZERO,
 3004                                   49560  ;       TRAILING ZEROS ARE SUPPRESSED, A NUMBER IS PRINTED IN FIXED OR FLOATING
 3005
 3006                                   49580  ;       POINT NOTATION ACCORDING TO ITS MAGNITUDE, THE NUMBER IS LEFT
 3007                                   49600  ;       JUSTIFIED IN ITS FIELD, B AND C ARE IGNORED,
 3008                                   49620  ;       1 MEANS FIXED FORMAT OUTPUT, I.E, THE OTHER BITS OF A ARE CHECKED FOR
 3009                                   49640  ;       FORMATTING INFORMATION, THE NUMBER IS RIGHT JUSTIFIED IN ITS FIELD,
 3010                                   49660  ;       TRAILING ZEROS ARE NOT SUPPRESSED,  THIS IS USED FOR PRINT USING,
 3011                                   49680  ;BIT 6  1 MEANS GROUP THE DIGITS IN THE INTEGER PART OF THE NUMBER INTO GROUPS
 3012                                   49700  ;       OF THREE AND SEPARATE THE GROUPS BY COMMAS
 3013                                   49720  ;       0 MEANS DON'T PRINT THE NUMBER WITH COMMAS
 3014                                   49740  ;BIT 5  1 MEANS FILL THE LEADING SPACES IN THE FIELD WITH ASTERISKS ("*")
 3015                                   49760  ;BIT 4  1 MEANS OUTPUT THE NUMBER WITH A FLOATING DOLLAR SIGN ("$")
 3016                                   49780  ;BIT 3  1 MEANS PRINT THE SIGN OF A POSITIVE NUMBER AS A PLUS SIGN ("+")
```



```
 3017                                   49800  ;       INSTEAD OF A SPACE
 3018                                   49820  ;BIT 2  1 MEANS PRINT THE SIGN OF THE NUMBER AFTER THE NUMBER
 3019                                   49840  ;BIT 1  UNUSED
 3020                                   49860  ;BIT 0  1 MEANS PRINT THE NUMBER IN FLOATING POINT NOTATION I,E, "E NOTATION"
 3021                                   49880  ;       IF THIS BIT IS ON, THE COMMA SPECIFICATION (BIT 6) IS IGNORED,
 3022                                   49900  ;       0 MEANS PRINT THE NUMBER IN FIXED POINT NOTATION,  NUMBER ,GE, 1E16
 3023                                   49920  ;       CANNOT BE PRINTED IN FIXED POINT NOTATION,
 3024                                   49940
 3025                                   49960  ;B AND C TELL HOW BIG THE FIELD IS:
 3026                                   49980  ;B  =   THE NUMBER OF PLACES IN THE FIELD TO THE LEFT OF THE DECIMAL POINT
 3027                                   50000  ;       (B DOES NOT INCLUDE THE DECIMAL POINT)
 3028                                   50020  ;C  =   THE NUMBER OF PLACES IN THE FIELD TO THE RIGHT OF THE DECIMAL POINT
 3029                                   50040  ;       (C INCLUDES THE DECIMAL POINT)
 3030                                   50060  ;       B AND C DONOT INCLUDE THE 4 POSITIONS FOR THE EXPONENT IF BIT 0 IS ON
 3031                                   50080          ;FOUT ASSUMES B+C ,LE, 24 (DECIMAL)
 3032                                   50100
 3033                                   50120          ;ENTRY TO PRINT THE FAC IN FREE FORMAT
 3034  002340' 257     000000  FOUT:   50140          XRA     A               ;SET FORMAT FLAGS TO FREE FORMATED OUTPUT
 3035                                   50160          ;ENTRY TO PRINT THE FAC USING THE FORMAT SPECIFICATIONS IN A, B AND C
 3036  002341' 315     000000  PUFOUT: 50180          CALL    FOUINI          ;SAVE THE FORMAT SPECIFICATION IN A AND PUT
 3037                                   50200                                  ;A SPACE FOR POSITIVE NUMBERS IN THE BUFFER
 3038  002344' 305     000000          50220          PUSH    B               ;SAVE THE FIELD LENGTH SPECIFICATIONS
 3039  002345' 346     000             50240          ANI     10              ;CHECK IF POSITIVE NUMBERS GET A PLUS SIGN
 3040  002347' 312     000000          50260          JZ      FOUT1           ;THEY DON'T
 3041  002352' 076     053             50280          MVI     M,"+"           ;THEY DO, PUT IN A PLUS SIGN
 3042  002354' 072     000000  FOUT1:  50300          LDA     VALTYP          ;SEE WHAT KIND OF A VALUE WE HAVE
 3043  002357' 107                     50320          MOV     B,A             ;SAVE IT
 3044  002360' 353                     50340          XCHG                    ;SAVE BUFFER POINTER
 3045  002361' 315     000000          50360          CALL    VSIGN           ;GET THE SIGN OF THE FAC
 3046  002364' 353                     50380          XCHG                    ;PUT THE BUFFER POINTER BACK IN (HL)
 3047  002365' 170                     50400          MOV     A,B             ;GET THE VALTYP BACK
 3048  002366' 362     000000          50420          JP      FOUT2           ;IF WE HAVE A NEGATIVE NUMBER, NEGATE IT
 3049  002371' 076     055             50440          MVI     M,"-"           ; AND PUT A MINUS SIGN IN THE BUFFER
 3050  002373' 345                     50460          PUSH    H               ;SAVE THE BUFFER POINTER
 3051  002374' 315     000000          50480          CALL    VNEG            ;NEGATE THE NUMBER
 3052  002377' 341                     50500          POP     H               ;GET THE BUFFER POINTER BACK
 3053  002400' 043             FOUT2:  50520          INX     H               ;POINT TO WHERE THE NEXT CHARACTER GOES
 3054  002401' 072     000000          50540          LDA     TEMP3           ;GET THE FORMAT SPECIFICATION
 3055  002404' 127                     50560          MOV     D,A             ;SAVE IT FOR LATER
 3056  002405' 027                     50580          RAL                     ;PUT THE FREE FORMAT OR NOT BIT IN THE CARRY
 3057  002406' 072     000000          50600          LDA     VALTYP          ;GET THE VALTYP, VNEG COULD HAVE CHANGED THIS
 3058                                   50620                                  ; SINCE -32768 IS INT AND 32768 IS SNG,
 3059                                   50640                                  ; SO B IS NOT ACCURATE
 3060  002411' 332     000000          50660          JC      FOUTFX          ;THE MAN WANTS FIXED FORMATED OUTPUT
 3061                                   50680          ;HERE TO PRINT NUMBERS IN FREE FORMAT
 3062  002414' 341                     50700          POP     B               ;WE CAN IGNORE THE OLD B AND C
 3063  002415' 076     060             50720          MVI     M,"0"           ;PUT A ZERO IN THE BUFFER IN CASE THE NUMBER
 3064  002417' 312     000000          50740          JZ      FOUTZR          ; IS ZERO,  IT IS, FINISH IT UP
 3065  002422' 376     004             50760          CPI     4               ;DECIDE WHAT KIND OF A VALUE WE HAVE
 3066  002424' 322     000000          50780          JNC     FOUFRY          ;WE HAVE A SNG OR DBL
 3067                                   50800          ;HERE TO PRINT AN INTEGER IN FREE FORMAT
 3068  002427' 041     000000          50820          LXI     B,$CODE         ;SET THE DECIMAL POINT COUNT AND COMMA COUNT
 3069                                   50840                                  ; TO ZERO
```

```
 3070                           50860          CALL    FOUTCI          ;CONVERT THE INTEGER TO DECIMAL
 3071                           50880                                  ;FALL INTO FOUTZS AND ZERO SUPPRESS THE THING
 3072                           50900
 3073                           50920          ;ZERO SUPPRESS THE DIGITS IN FBUFFR
 3074                           50940          ;ASTERISK FILL AND ZERO SUPPRESS IF NECESSARY
 3075                           50960          ;SET UP B AND CONDITION CODES IF WE HAVE A TRAILING SIGN
 3076                           50980  FOUTZS: LXI     H,FBUFFR+1      ;GET POINTER TO THE SIGN
 3077                           51000          MOV     B,M             ;SAVE THE SIGN IN B
 3078                           51020          MVI     C," "           ;DEFAULT FILL CHARACTER TO A SPACE
 3079                           51040          LDA     TEMP3           ;GET FORMAT SPECS TO SEE IF WE HAVE TO
 3080                           51060          MOV     E,A             ; ASTERISK FILL,  SAVE IT
 3081                           51080          ANI     40
 3082                           51100          JZ      FOTZS1          ;WE DON'T
 3083                           51120          MOV     A,B             ;WE DO, SEE IF THE SIGN WAS A SPACE
 3084                           51140          CMP     C               ;ZERO FLAG IS SET IF IT WAS
 3085                           51160          MVI     C,"*"           ;SET FILL CHARACTER TO AN ASTERISK
 3086                           51180          JNZ     FOTZS1          ;SET THE SIGN TO AN ASTERISK IF IT WAS A SPACE
 3087                           51200          MOV     B,C             ;B HAS THE SIGN, C THE FILL CHARACTER
 3088                           51220  FOTZS1: MOV     M,C             ;FILL IN THE ZERO OR THE SIGN
 3089                           51240          CHRGET                  ;GET THE NEXT CHARACTER IN THE BUFFER
 3090                           51260                                  ;SINCE THERE ARE NO SPACES, "CHRGET" IS
 3091                           51280                                  ; EQUIVALENT TO "INX     H"/"MOV A,M"
 3092                           51300          CPI     "0"             ;DO WE HAVE A ZERO?
 3093                           51320          JZ      FOTZS1          ;YES, SUPPRESS IT
 3094                           51340          CPI     54              ;54="."  DO WE HAVE A COMMA?
 3095                           51360          JZ      FOTZS1          ;YES, SUPPRESS IT
 3096                           51380          CPI     "."             ;ARE WE AT THE DECIMAL POINT?
 3097                           51400          JNZ     FOTZS2          ;NO, I GUESS NOT
 3098                           51420          DCX     H               ;YES, BACK UP AND PUT A ZERO BEFORE IT
 3099                           51440          MVI     M,"0"
 3100                           51460  FOTZS2: MOV     A,E             ;GET THE FORMAT SPECS TO CHECK FOR A FLOATING
 3101                           51480          ANI     20              ; DOLLAR SIGN
 3102                           51500          JZ      FOTZS3          ;WE DON'T HAVE ONE
 3103                           51520          DCX     H               ;WE HAVE ONE, BACK UP AND PUT IN THE DOLLAR
 3104                           51540          MVI     M,"$"           ; SIGN
 3105                           51560  FOTZS3: MOV     A,E             ;DO WE HAVE A TRAILING SIGN?
 3106                           51580          ANI     4
 3107                           51600          RNZ                     ;YES, RETURN; NOTE THE NON-ZERO FLAG IS SET
 3108                           51620  FOFXIS: DCX     H               ;NO, BACK UP ONE AND PUT THE SIGN BACK IN
 3109                           51640                                  ;PEOPLE JUMP HERE WHO WANT A "DCX       H" AND
 3110                           51660                                  ; DON'T CARE ABOUT M
 3111                           51680          MOV     M,B             ;PUT IN THE SIGN
 3112                           51700          RET                     ;ALL DONE
 3113                           51720
 3114                           51740          ;HERE TO INITIALLY SET UP THE FORMAT SPECS AND PUT IN A SPACE FOR THE
 3115                           51760          ;SIGN OF A POSITIVE NUMBER
 3116                           51780  FOUINI: STA     TEMP3           ;SAVE THE FORMAT SPECIFICATION
 3117                           51800          LXI     H,FBUFFR+1      ;GET A POINTER INTO FBUFFR
 3118                           51820          MVI     M," "           ;PUT IN A SPACE
 3119                           51840          RET                     ;ALL DONE
 3120                           51860
 3121                           51880          ;HERE TO PRINT A SNG OR DBL IN FREE FORMAT
 3122                           51900  FOUFRV: PUSH    H               ;SAVE THE BUFFER POINTER
```



```
 3123                           51920          JZ      FOUFRS          ;WE HAVE A SNG
 3124                           51940          ;HERE TO SET UP THE FLAG TO PRINT A DBL IN FREE FORMAT
 3125                           51960          MVI     D,20            ;WE HAVE A DBL, SET THE DIGIT COUNT
 3126                           51980          XWD     1000,001        ;"LXI   B" OVER THE NEXT TWO BYTES
 3127                           52000          ;HERE TO SET UP THE FLAG TO PRINT A SNG IN FREE FORMAT
 3128                           52020  FOUFRS: MVI     D,6             ;SET THE DIGIT COUNT
 3129                           52040          CALL    FOUTNV          ;NORMALIZE THE FAC SO ALL SIGNIFICANT DIGITS
 3130                           52060                                  ; ARE IN THE INTEGER PART
 3131                           52080          LXI     B,2*400+$CODE   ;B = DECIMAL POINT COUNT
 3132                           52100                                  ;C = COMMA COUNT
 3133                           52120                                  ;SET COMMA COUNT TO ZERO AND DECIMAL POINT
 3134                           52140                                  ; COUNT FOR E NOTATION
 3135                           52160          ADD     D               ;SEE IF NUMBER SHOULD BE PRINTED IN E NOTATION
 3136                           52180          JM      FOFRS1          ;IT SHOULD, IT IS .LT. .1
 3137                           52200          INR     D               ;CHECK IF IT IS TOO BG
 3138                           52220          CMP     D
 3139                           52240          JNC     FOFRS1          ;IT IS TOO BIG, IT IS .GT. 10^D-1
 3140                           52260          INR     A               ;IT IS OK FOR FIXED POINT NOTATION
 3141                           52280          MOV     B,A             ;SET DECIMAL POINT COUNT
 3142                           52300          MVI     A,1             ;SET FIXED POINT FLAG, THE EXPONENT IS ZERO
 3143                           52320                                  ; IF WE ARE USING FIXED POINT NOTATION
 3144                           52340  FOFRS1: DCR     A               ;E NOTATION: ADD D-1 TO ORIGINAL EXPONENT
 3145                           52360          POP     H               ;GET THE BUFFER POINTER BACK
 3146                           52380          PUSH    PSW             ;SAVE THE EXPONENT FOR LATER
 3147                           52400          CALL    FOUTCV          ;CONVERT THE NUMBER TO DECIMAL DIGITS
 3148                           52420          ;HERE TO SUPPRESS THE TRAILING ZEROS
 3149                           52440  FOFRS2: DCX     H               ;MOVE BACK TO THE LAST CHARACTER
 3150                           52460          MOV     A,M             ;GET IT AND SEE IF IT WAS ZERO
 3151                           52480          CPI     "0"
 3152                           52500          JZ      FOFRS2          ;IT WAS, CONTINUE SUPPRESSING
 3153                           52520          CPI     ".";HAVE WE SUPPRESSED ALL THE FRACTIONAL DIGITS?
 3154                           52540          CNZ     INXHRT          ;YES, IGNORE THE DECIMAL POINT ALSO
 3155                           52560          POP     PSW             ;GET THE EXPONENT BACK
 3156                           52580          JZ      FOUTON          ;WE ARE DONE IF WE ARE IN FIXED POINT NOTATION
 3157                           52600                                  ;FALL IN AND PUT THE EXPONENT IN THE BUFFER
 3158                           52620
 3159                           52640          ;HERE TO PUT THE EXPONENT AND "E" OR "D" IN THE BUFFER
 3160                           52660          ;THE EXPONENT IS IN A
 3161                           52680  FOFLDN: MOV     B,A             ;SAVE THE EXPONENT
 3162                           52700          LDA     VALTYP          ;GET THE VALTYP TO DEIDE IF WE PRINT AN "E"
 3163                           52720          CPI     4               ; OR A "D"
 3164                           52740          MOV     A,B             ;GET THE EXPONENT BACK
 3165                           52760          JZ      FOUCEE          ;WE HAVE TO PRINT AN "E"
 3166                           52780          MVI     M,"D"           ;GET THE "D"
 3167                           52800          XWD     1000,001        ;"LXI   B" OVER THE NEXT TWO BYTES
 3168                           52820  FOUCEE: MVI     M,"E"           ;GET AN "E"
 3169                           52840          INX     H               ;PUT SIGN OF EXPONENT IN BUFFER
 3170                           52860          ;PUT IN THE SIGN OF THE EXPONENT
 3171                           52880          MVI     M,"+"           ;A PLUS IF POSITIVE
 3172                           52900          JP      FOUCE1
 3173                           52920          MVI     M,"-"           ;A MINUS IF NEGATIVE
 3174                           52940          CMA                     ;NEGATE EXPONENT
 3175                           52960          INR     A
```

```
3176                                    52980           ;CALCULATE THE TWO DIGIT EXPONENT
3177                                    53000   FOUCE1: MVI     B,"0"-1         ;INITIALIZE TEN'S DIGIT COUNT
3178                                    53020   FOUCE2: INR     B               ;INCREMENT DIGIT
3179                                    53040           SUI     12              ;SUBTRACT TEN
3180                                    53060           JNC     FOUCE2          ;DO IT AGAIN IF RESULT WAS POSITIVE
3181                                    53080           ADI     "0"+12          ;ADD BACK IN TEN AND CONVERT TO ASCII
3182                                    53100           ;PUT THE EXPONENT IN THE BUFFER
3183                                    53120           INX     H
3184                                    53140           MOV     M,B             ;PUT TEN'S DIGIT OF EXPONENT IN BUFFER
3185                                    53160           INX     H               ;WHEN WE JUMP TO HERE, A IS ZERO
3186                                    53180           MOV     M,A             ;PUT ONE'S DIGIT IN BUFFER
3187                                    53200   FOUTZR: INX     H               ;INCREMENT POINTER, HERE TO FINISH UP PRINTING
3188                                    53220                                   ; A FREE FORMAT ZERO
3189                                    53240   FOUTON: MVI     M,0             ;PUT A ZERO AT THE END OF THE NUMBER
3190                                    53260           XCHG                    ;SAVE THE POINTER TO THE END OF THE NUMBER
3191                                    53280                                   ; IN (DE) FOR FFXFLV
3192                                    53300           LXI     H,FBUFFR+1      ;GET A POINTER TO THE BEGINNING
3193                                    53320           RET                     ;ALL DONE
3194                                    53340
3195                                    53360           ;HERE TO PUT A POSSIBLE COMMA COUNT IN C, AND ZERO C IF WE ARE NOT
3196                                    53380           ;USING THE COMMA SPECIFICATION
3197                                    53400   FOUTCC: MOV     C,A             ;SAVE A POSSIBLE COMMA COUNT
3198                                    53420   FOUICC: LDA     TEMP3           ;GET THE FORMAT SPECS
3199                                    53440           ANI     100             ;LOOK AT THE COMMA BIT
3200                                    53460           RNZ                     ;WE ARE USING COMMAS, JUST RETURN
3201                                    53480           MOV     C,A             ;WE AREN'T, ZERO THE COMMA COUNT
3202                                    53500           RET                     ;ALL DONE
3203                                    53520
3204                                    53540           ;HERE TO PRINT A NUMBER IN FIXED FORMAT
3205                                    53560   FOUTFX: CPI     4               ;CHECK WHAT KIND OF VALUE WE HAVE
3206                                    53580           MOV     A,D             ;GET THE FORMAT SPECS
3207                                    53600           JNC     FOUFXV          ;WE HAVE A SNG OR A DBL
3208                                    53620           ;HERE TO PRINT AN INTEGER IN FIXED FORMAT
3209                                    53640           RAR                     ;CHECK IF WE HAVE TO PRINT IT IN FLOATING
3210                                    53660           JC      FFXIFL          ; POINT NOTATION
3211                                    53680           ;HERE TO PRINT AN INTEGER IN FIXED FORMAT=FIXED POINT NOTATION
3212                                    53700           LXI     B,6*400+3+$CODE ;SET DECIMAL POINT COUNT TO 6 AND
3213                                    53720                                   ; COMMA COUNT TO 3
3214                                    53740           CALL    FOICC           ;CHECK IF WE DON'T HAVE TO USE THE COMMAS
3215                                    53760           POP     D               ;GET THE FIELD LENGTHS
3216                                    53780           MOV     A,D             ;SEE IF WE HAVE TO PRINT EXTRA SPACES BECAUSE
3217                                    53800           SUI     5               ; THE FIELD IS TOO BIG
3218                                    53820           CP      FOTZER          ;WE DO, PUT IN ZEROS, THEY WILL LATER BE
3219                                    53840                                   ; CONVERTED TO SPACES OR ASTERISKS BY FOUTZS
3220                                    53860           PUSH    D               ;SAVE THE FIELD LENGTHS AGAIN
3221                                    53880           CALL    FOUTCI          ;CONVERT THE NUMBER TO DECIMAL DIGITS
3222                                    53900           POP     D               ;GET THE FIELD LENGTHS BACK
3223                                    53920           ORA     E               ;DO WE NEED A DECIMAL POINT?  HERE A=0
3224                                    53940           CZ      FOFXIS          ;WE DON'T, BACKSPACE OVER IT,  AT FOFXIS,
3225                                    53960                                   ; WE DON'T CARE ABOUT B, WE ONLY WANT THE
3226                                    53980                                   ; "DCX  H",  LATER ON WE WILL DO A "MVI M,0"
3227                                    54000                                   ; AND COVER UP WHAT WAS IN B,
3228                                    54020           CNZ     FOTZER          ;FOFXIS PRESERVES THE CONDITION CODES
```



```
3229                                    54040                                   ;IF WE DO HAVE DECIMAL PLACES, FILL THEM UP
3230                                    54060                                   ; WITH ZEROS
3231                                    54080                                   ;FALL IN AND FINISH UP THE NUMBER
3232                                    54100
3233                                    54120           ;HERE TO FINISH UP A FIXED FORMAT NUMBER
3234                                    54140   FOUTTS: PUSH    H               ;SAVE BUFFER POINTER
3235                                    54160           CALL    FOUTZS          ;ZERO SUPPRESS THE NUMBER
3236                                    54180           POP     H               ;GET THE BUFFER POINTER BACK
3237                                    54200           JZ      FFXIX1          ;CHECK IF WE HAVE A TRAILING SIGN
3238                                    54220           MOV     M,B             ;WE DO, PUT THE SIGN IN THE BUFFER
3239                                    54240           INX     H               ;INCREMENT THE BUFFER POINTER
3240                                    54260   FFXIX1: MVI     M,0             ;PUT A ZERO AT THE END OF THE NUMBER
3241                                    54280
3242                                    54300           ;HERE TO CHECK IF A FIXED FORMAT=FIXED POINT NUMBER OVERFLOWED ITS
3243                                    54320           ;FIELD LENGTH
3244                                    54340           ;D = THE B IN THE FORMAT SPECIFICATION
3245                                    54360           ;THIS ASSUMES THE LOCATION OF THE DECIMAL POINT IS IN TEMP2
3246                                    54380           LXI     H,FBUFFR        ;GET A POINTER TO THE BEGINNING
3247                                    54400   FOUBE1: INX     H               ;INCREMENT POINTER TO THE NEXT CHARACTER
3248                                    54420   FOUBE5: LDA     TEMP2           ;GET THE LOCATION OF THE DECIMAL POINT
3249                                    54440           SUB     L               ;FIGURE OUT HOW MUCH SPACE WE ARE TAKING
3250                                    54460           SUB     D               ;IS THIS THE RIGHT AMOUNT OF SPACE TO TAKE?
3251                                    54480           RZ                      ;YES, WE ARE DONE, RETURN FROM FOUT
3252                                    54500           MOV     A,M             ;NO, WE MUST HAVE TOO MUCH SINCE WE STARTED
3253                                    54520                                   ; CHECKING FROM THE BEGINNING OF THE BUFFER
3254                                    54540                                   ; AND THE FIELD MUST BE SMALL ENOUGH TO FIT IN
3255                                    54560                                   ; THE BUFFER,  GET THE NEXT CHARACTER IN
3256                                    54580                                   ; THE BUFFER
3257                                    54600           CPI     " "             ;IF IT IS A SPACE OR AN ASTERISK, WE CAN
3258                                    54620           JZ      FOUBE1          ; IGNORE IT AND MAKE THE FIELD SHORTER WITH
3259                                    54640           CPI     "*"             ; NO ILL EFFECTS
3260                                    54660           JZ      FOUBE1
3261                                    54680           DCX     H               ;MOVE THE POINTER BACK ONE TO READ THE
3262                                    54700                                   ; CHARACTER WITH CHMGET
3263                                    54720           PUSH    H               ;SAVE THE POINTER
3264                                    54740
3265                                    54760           ;HERE WE SEE IF WE CAN IGNORE THE LEADING ZERO BEFORE A DECIMAL POINT,
3266                                    54780           ;THIS OCCURS IF WE SEE THE FOLLOWING: (IN ORDER)
3267                                    54800           ;       A SIGN (EITHER "-" OR "+")              [OPTIONAL]
3268                                    54820           ;       A DOLLAR SIGN                           [OPTIONAL]
3269                                    54840           ;       A ZERO                                  [MANDATORY]
3270                                    54860           ;       A DECIMAL POINT                         [MANDATORY]
3271                                    54880           ;       ANOTHER DIGIT                           [MANDATORY]
3272                                    54900           ;IF YOU SEE A LEADING ZERO, IT MUST BE THE ONE BEFORE A DECIMAL POINT
3273                                    54920           ;OR ELSE FOUTZS WOULD HAVE SUPPRESSED IT, SO WE CAN JUST "INX   H"
3274                                    54940           ;OVER THE CHARACTER FOLLOWING THE ZERO, AND NOT CHECK FOR THE
3275                                    54960           ;DECIMAL POINT EXPLICITLY,
3276                                    54980   FOUBE2: PUSH    PSW             ;PUT THE LAST CHARACTER ON THE STACK,  THE
3277                                    55000                                   ; ZERO FLAG IS SET,  THE FIRST TIME THE ZERO
3278                                    55020                                   ; ZERO FLAG IS NOT SET,
3279                                    55040           LXI     B,FOUBE2        ;GET THE ADDRESS WE GO TO IF WE SEE A CHARACTER
3280                                    55060
3281                                    55080           PUSH    B               ; WE ARE LOOKING FOR
```

```
3282                    55080           CHRGET                  ;GET THE NEXT CHARACTER
3283                    55100           CPI     "-"             ;SAVE IT AND GET THE NEXT CHARACTER IF IT IS
3284                    55120           RZ                      ; A MINUS SIGN, A PLUS SIGN OR A DOLLAR SIGN
3285                    55140           CPI     "+"
3286                    55160           RZ
3287                    55180           CPI     "$"
3288                    55200           RZ
3289                    55220           POP     B               ;IT ISN'T, GET THE ADDRESS OFF THE STACK
3290                    55240           CPI     "0"             ;IS IT A ZERO?
3291                    55260           JNZ     FOUBE4          ;NO, WE CAN NOT GET RID OF ANOTHER CHARACTER
3292                    55280           INX     H               ;SKIP OVER THE DECIMAL POINT
3293                    55300           CHRGET                  ;GET THE NEXT CHARACTER
3294                    55320           JNC     FOUBE4          ;IT IS NOT A DIGIT, WE CAN'T SHORTEN THE FIELD
3295                    55340           DCX     H               ;WE CAN;;;  POINT TO THE DECIMAL POINT
3296                    55360           XCHD    1000,001        ;"LXI   B" OVER THE NEXT 2 BYTES
3297                    55380   FOUBE3: DCX     H               ;POINT BACK ONE CHARACTER
3298                    55400           MOV     M,A             ;PUT THE CHARACTER BACK
3299                    55420
3300                    55440           ;IF WE CAN GET RID OF THE ZERO, WE PUT THE CHARACTERS ON THE STACK
3301                    55460           ;BACK INTO THE BUFFER ONE POSITION IN FRONT OF WHERE THEY ORIGINALLY
3302                    55480           ;WERE,  NOTE THAT THE MAXIMUM NUMBER OF STACK LEVELS THIS USES IS
3303                    55500           ;THREE -- ONE FOR THE LAST ENTRY FLAG, ONE FOR A POSSIBLE SIGN,
3304                    55520           ;AND ONE FOR A POSSIBLE DOLLAR SIGN,  WE DON'T HAVE TO WORRY ABOUT
3305                    55540           ;THE FIRST CHARACTER BEING IN THE BUFFER TWICE BECAUSE THE POINTER
3306                    55560           ;WHEN FOUT EXITS WILL BE POINTING TO THE SECOND OCCURANCE,
3307                    55580           POP     PSW             ;GET THE CHARACTER OFF THE STACK
3308                    55600           JZ      FOUBE3          ;PUT IT BACK IN THE BUFFER IF IT IS NOT THE
3309                    55620                                   ; LAST ONE
3310                    55640           POP     B               ;GET THE BUFFER POINTER OFF THE STACK
3311                    55660           JMP     FOUBE3          ;SEE IF THE FIELD IS NOW SMALL ENOUGH
3312                    55680           ;HERE IF THE NUMBER IS TOO BIG FOR THE FIELD
3313                    55700   FOUBE4: POP     PSW             ;GET THE CHARACTERS OFF THE STACK
3314                    55720           JZ      FOUBE4          ;LEAVE THE NUMBER IN THE BUFFER ALONE
3315                    55740           POP     H               ;GET THE POINTER TO THE BEGINNING OF THE
3316                    55760                                   ; NUMBER MINUS 1
3317                    55780           MVI     M,"%"           ;PUT IN A PERCENT SIGN TO INDICATE THE NUMBER
3318                    55800                                   ; WAS TOO LARGE FOR THE FIELD
3319                    55820           RET                     ;ALL DONE -- RETURN FROM FOUT
3320                    55840
3321                    55860           ;HERE TO PRINT A SNG OR DBL IN FIXED FORMAT
3322                    55880   FOUFXV: PUSH    H               ;SAVE THE BUFFER POINTER
3323                    55900           RAR                     ;GET FIXED OR FLOATING NOTATION FLAG IN CARRY
3324                    55920           JC      FFXFLV          ;PRINT THE NUMBER IN E-NOTATION
3325                    55940           JZ      FFXSFX          ;WE HAVE A SNG
3326                    55960           ;HERE TO PRINT A DBL IN FIXED FORMAT--FIXED POINT NOTATION
3327                    55980           LXI     D,FFXOXM        ;GET POINTER TO 1D16
3328                    56000           CALL    DCOMPD          ;WE CAN'T PRINT A NUMBER .GE. 10^16 IN FIXED
3329                    56020                                   ; POINT NOTATION
3330                    56040           LXI     D,16*400+$CODE  ;SET D = NUMBER OF DIGITS TO PRINT FOR A DBL
3331                    56060                                   ;C = 0 FOR DBL (THIS IS FOR COMMAS)
3332                    56080           JM      FFXSDO          ;IF THE FAC WAS SMALL ENOUGH, GO PRINT IT
3333                    56100           ;HERE TO PRINT IN FREE FORMAT WITH A PERCENT SIGN A NUMBER .GE. 10^16
3334                    56120   FFXSDO: XRA     A               ;SET THE FORMAT SPECS TO FREE FORMAT
```



```
3335                    56140           STA     TEMP3
3336                    56160           POP     H               ;GET THE BUFFER POINTER
3337                    56180           DCX     H               ;SET IT UP TO JUMP BACK INTO FOUT
3338                    56200                                   ;NOW (HL)=FBUFFR+1
3339                    56220           CALL    FOUT1           ;PRINT THE NUMBER IN FREE FORMAT, THE SIGN
3340                    56240                                   ; IS ALREADY IN THE BUFFER
3341                    56260           DCX     H               ;POINT IN FRONT OF THE NUMBER
3342                    56280           MVI     M,"%"           ;PUT IN THE PERCENT SIGN
3343                    56300           RET                     ;ALL DONE--RETURN FROM FOUT
3344                    56320           ;HERE TO PRINT A SNG IN FIXED FORMAT--FIXED POINT NOTATION
3345                    56340   FFXSFX: MOVRI   226,016,033,312 ;GET 1E16, CHECK IF THE NUMBER IS TOO BIG
3346                    56360           CALL    FCOMP
3347                    56380           JP      FFXSDO          ;IT IS, PRINT IT IN FREE FORMAT WITH A % SIGN
3348                    56400           LXI     D,6*400+2+$CODE ;SET UP FLAGS FOR SNG --
3349                    56420                                   ;D = NUMBER OF DIGITS TO PRINT IN A SNG
3350                    56440                                   ;C = 2 (THIS IS FOR THE COMMAS)
3351                    56460           ;HERE TO ACTUALLY PRINT A SNG OR DBL IN E NOTATION
3352                    56480   FFXSDO: FSIGN                   ;SEE IF WE HAVE ZERO
3353                    56500           CNZ     FOUTNV          ;IF NOT, NORMALIZE THE NUMBER SO ALL DIGITS TO
3354                    56520                                   ; BE PRINTED ARE IN THE INTEGER PART
3355                    56540           POP     H               ;GET THE BUFFER POINTER
3356                    56560           POP     B               ;GET THE FIELD LENGTH SPECS
3357                    56580           JM      FFXXVS          ;DO DIFFERENT STUFF IF [redacted]EXPONENT IS NEGATIVE
3358
3359                    56600           ;HERE TO PRINT A NUMBER WITH NO FRACTIONAL DIGITS
3360                    56620           PUSH    B               ;SAVE THE FIELD LENGTH SPECS AGAIN
3361                    56640           MOV     C,A             ;SAVE THE EXPONENT
3362                    56660           MOV     A,B             ;WE HAVE TO PRINT LEADING ZEROS IF THE FIELD
3363                    56680           SUB     D               ; HAS MORE CHARACTERS THAN THERE ARE DIGITS IN
3364                    56700           SUB     C               ; THE NUMBER
3365                    56720           CP      FOUTZER         ;FOUTZS WILL LATER SUPPRESS THEM
3366                    56740           MOV     A,C             ;SET UP THE COMMA COUNT
3367                    56760           ADD     E
3368                    56780           MOV     E,C             ;WE NO LONGER NEED THE NUMBER WE SETUP IN E
3369                    56800   FFXXV1: SUI     3               ;REDUCE A MOD 3
3370                    56820           JNC     FFXXV1
3371                    56840           ADI     5               ;ADD 3 BACK AND ADD IN 2 MORE FOR SCALING
3372                    56860           CALL    FOUTCC          ;CHECK IF WE HAVE TO USE COMMAS AT ALL
3373                    56880           MOV     A,E             ;SETUP THE DECIMAL POINT COUNT
3374                    56900           ADD     D
3375                    56920           INR     A
3376                    56940           MOV     B,A             ;IT GOES IN B
3377                    56960           PUSH    D               ;SAVE SNG OR DBL AND EXPONENT INFORMATION
3378                    56980           CALL    FOUTCV          ;CONVERT THE NUMBER TO DECIMAL DIGITS
3379                    57000           POP     D               ;GET THE SNG OR DBL AND EXPONENT INFO BACK
3380                    57020           ORA     E               ;PUT IN DIGITS AFTER THE NUMBER IF IT
3381                    57040                                   ; IS BIG ENOUGH, HERE A=0
3382                    57060           CNZ     FOUTZRC         ;THERE CAN BE COMMAS IN THESE ZEROS
3383                    57080           POP     D               ;GET THE FIELD LENGTH SPECS
3384                    57100           ORA     E               ;HERE A=0
3385                    57120           CNZ     FOUTDP          ;PRINT A DECIMAL POINT IF NECESSARY
3386                    57140           DCR     A               ;SEE IF WE SHOULD PRINT SOME ZEROS TO FILL IN
3387                    57160           CP      FOUTZER         ; THE DECIMAL PLACES
```

```
 3388                                   57100          JMP     FOUTTS          ;GO CHECK THE SIZE, ZERO SUPPRESS, ETC, AND
 3389                                   57200                                  ; FINISH THE NUMBER
 3390                                   57220          ;HERE TO PRINT A SNG OR DBL THAT HAS FRACTIONAL DIGITS
 3391                                   57240  FFXXV5: MOV     E,A             ;SAVE THE EXPONENT, WE DON'T NEED WHAT IS IN E
 3392                                   57260          MOV     A,C             ;DIVIDE BY TEN THE RIGHT NUMBER OF TIMES SO
 3393                                   57280          ORA     A               ; THE RESULT WILL BE ROUNDED CORRECTLY AND
 3394                                   57300          CNZ     DCRART          ; HAVE THE CORRECT NUMBER OF SIGNIFICANT
 3395                                   57320          ADD     E               ; DIGITS
 3396                                   57340          PUSH    PSW             ;SAVE THIS NUMBER FOR LATER
 3397                                   57360  FFXXV2: CM      FINDIV          ;THIS IS THE DIVIDE LOOP
 3398                                   57380          JM      FFXXV2
 3399                                   57400          MOV     A,E             ;WE HAVE TWO CASES DEPENDING ON WHETHER THE
 3400                                   57420          ADD     D               ; THE NUMBER IS ,LT, ,1 OR NOT
 3401                                   57440          MOV     A,B
 3402                                   57460          JM      FFXXV3
 3403                                   57480          ;HERE TO PRINT NUMBERS ,GE, ,1
 3404                                   57500          SUB     D               ;PRINT SOME LEADING ZEROS IF THE FIELD IS
 3405                                   57520          SUB     E               ; BIGGER THAN THE NUMBER OF DIGITS WE WILL
 3406                                   57540          CP      FOTZER          ; PRINT
 3407                                   57560          POP     PSW             ;WE DON'T NEED THE NUMBER WE SAVED BEFORE
 3408                                   57580          MOV     B,E             ;GET ALL THE PERTINENT INFO IN B,C
 3409                                   57600          PUSH    B               ;SAVE THE EXPONENT AND "C" IN FIELD SPEC
 3410                                   57620          MOV     A,E             ;SET UP THE DECIMAL POINT COUNT
 3411                                   57640          ADD     D
 3412                                   57660          INR     A
 3413                                   57680          MOV     B,A
 3414                                   57700          MOV     A,D             ;SET UP THE COMMA COUNT
 3415                                   57720          ANI     2               ;THESE 2 INSTRUCTIONS MAP 6 TO 4
 3416                                   57740          ADI     2               ; AND 16 TO 2
 3417                                   57760          ADD     E
 3418                                   57780          CALL    FOUTCC          ;CHECK IF WE HAVE TO DO THE COMMA THING
 3419                                   57800          JMP     FFXXV6          ;CONVERT THE DIGITS AND DO THE TRIMMING UP
 3420                                   57820          ;HERE TO PRINT A NUMBER ,LT, ,1
 3421                                   57840  FFXXV3: CALL    FOTZER          ;PUT ALL ZEROS BEFORE THE DECIMAL POINT
 3422                                   57860          MOV     A,C             ;SAVE C
 3423                                   57880          CALL    FOUTDP          ;PUT IN A DECIMAL POINT
 3424                                   57900          MOV     C,A             ;RESTORE C
 3425                                   57920          POP     PSW             ;GET THE NUMBER WE SAVED
 3426                                   57940          JM      FFXXV4          ;DECIDE HOW MANY ZEROS TO PRINT BETWEEN THE
 3427                                   57960          XRA     A               ; DECIMAL POINT AND THE FIRST DIGIT WE WILL
 3428                                   57980          SUB     E               ; PRINT,  HERE THE FIELD IS BIG ENOUGH TO
 3429                                   58000          SUB     D               ; HOLD ALL THE DIGITS
 3430                                   58020          JMP     FFXXV5          ;GO PRINT THEM
 3431                                   58040  FFXXV4: MOV     A,C             ;HERE WE HAD TO DIVIDE BY TEN SO THE FIELD
 3432                                   58060          SUB     D               ; IS SMALLER THAN ALL SIGNIFICANT DIGITS IN
 3433                                   58080          DCR     A               ; THE NUMBER
 3434                                   58100  FFXXV5: CALL    FOTZER          ;PRINT THE ZEROS
 3435                                   58120          MOV     B,E             ;SAVE THE EXPONENT IN B
 3436                                   58140          PUSH    B               ;SAVE EXPONENT AND THE "C" IN THE FIELD SPEC
 3437                                   58160          MOV     B,A             ;ZERO THE DECIMAL PLACE COUNT
 3438                                   58180          MOV     C,A             ;ZERO THE COMMA COUNT
 3439                                   58200  FFXXV6: CALL    FOUTCV          ;CONVERT THE NUMBER TO DECIMAL DIGITS
 3440                                   58220          POP     D               ;GET THE EXPONENT AND FIELD SPEC BACK
```



```
 3441                                   58240          ORA     E               ;CHECK IF WE HAVE TO PRINT ANY ZEROS AFTER
 3442                                   58260                                  ; THE LAST DIGIT
 3443                                   58280          JZ      FFXXV7          ;CHECK IF THERE WERE ANY DECIMAL PLACES AT ALL
 3444                                   58300          ADD     D               ;PRINT SOME MORE TRAILING ZEROS
 3445                                   58320          DCR     A
 3446                                   58340          CP      FOTZER
 3447                                   58360          JMP     FOUTTS          ;FINISH UP THE NUMBER
 3448                                   58380          ;THERE WERE NO DECIMAL PLACES, IGNORE ALL DIGITS AFTER THE DECIMAL
 3449                                   58400          ;POINT
 3450                                   58420  FFXXV7: LHLD    TEMP2           ;THE END OF THE NUMBER IS WHERE THE DP IS
 3451                                   58440          JMP     FOUTTS          ;FINISH UP THE NUMBER
 3452                                   58460
 3453                                   58480          ;HERE TO PRINT AN INTEGER IN FIXED FORMAT--FLOATING POING NOTATION
 3454                                   58500  FFXIFL: PUSH    H               ;SAVE THE BUFFER POINTER
 3455                                   58520          PUSH    D               ;SAVE THE FORMAT SPECS
 3456                                   58540          CALL    CONSI           ;CONVERT THE INTEGER TO A SNG
 3457                                   58560          POP     D               ;GET THE FORMAT SPECS BACK
 3458                                   58580          POP     H               ;GET THE BUFFER POINTER BACK
 3459                                   58600          XRA     A               ;SET FLAGS TO PRINT THE NUMBER AS A SNG
 3460                                   58620                                  ;FALL INTO FFXFLV
 3461                                   58640
 3462                                   58660          ;HERE TO PRINT A SNG OR DBL IN FIXED FORMAT=FLOTATING POINT NOTATION
 3463                                   58680  FFXFLV: JZ      FFXSFL          ;IF WE HAVE A SNG, SET THE RIGHT FLAGS
 3464                                   58700          MVI     E,20            ;WE HAVE A DBL, GET HOW MANY DIGITS WE HAVE
 3465                                   58720          X*D     1000,001        ;"LXI   B" OVER THE NEXT TWO BYTES
 3466                                   58740  FFXSFL: MVI     E,6             ;WE HAVE A DBL, GET HOW MANY DIGITS WE PRINT
 3467                                   58760          FSIGN                   ;SEE IF WE HAVE ZERO
 3468                                   58780          CNZ     FOUTNV          ;IF NOT, NORMALIZE THE NUMBER SO ALL DIGITS TO
 3469                                   58800                                  ; BE PRINTED ARE IN THE INTEGER PART
 3470                                   58820          POP     H               ;GET THE BUFFER POINTER BACK
 3471                                   58840          POP     B               ;GET THE FIELD LENGTH SPECS
 3472                                   58860          PUSH    PSW             ;SAVE THE EXPONENT
 3473                                   58880          MOV     A,C             ;CALCULATE HOW MANY SIGNIFICANT DIGITS WE MUST
 3474                                   58900          ORA     A               ; PRINT
 3475                                   58920          PUSH    PSW             ;SAVE THE "C" FIELD SPEC FOR LATER
 3476                                   58940          CNZ     DCRART
 3477                                   58960          ADD     B
 3478                                   58980          MOV     C,A
 3479                                   59000          MOV     A,D             ;GET THE FIELD SPEC
 3480                                   59020          ANI     4               ;SEE IF THE SIGN IS A TRAILING SIGN
 3481                                   59040          CPI     1               ;SET CARRY IF A IS ZERO
 3482                                   59060          SBB     A               ;SET D=0 IF WE HAVE A TRAILING SIGN,
 3483                                   59080          MOV     D,A             ; D=377 IF WE DO NOT
 3484                                   59100          ADD     C
 3485                                   59120          MOV     C,A             ;SET C=NUMBER OF SIGNIFICANT DIGITS TO PRINT
 3486                                   59140          SUB     E               ;IF WE HAVE LESS THAN E, THEN WE MUST GET RID
 3487                                   59160  FFXLV1: CM      FINDIV          ; OF SOME BY DIVIDING BY TEN AND ROUNDING
 3488                                   59180          JM      FFXLV1
 3489                                   59200          PUSH    B               ;SAVE THE "B" FIELD SPEC AND # OF SIG DIGITS
 3490                                   59220          MOV     A,B             ;SET THE DECIMAL PLACE COUNT
 3491                                   59240          INR     A
 3492                                   59260          SUB     D               ;TAKE INTO ACCOUNT IF THE SIGN IS TRAILING
 3493                                   59280          MOV     B,A
```

```
                        59300           MVI     C,0             ;SET COMMA COUNT TO ZERO, THE COMMA SPEC IS
                        59320           PUSH    D               ; IGNORED,  SAVE TRAILING SIGN INFO
                        59340           CALL    FOUTCV          ;CONVERT THE NUMBER TO DECIMAL DIGITS
                        59360           POP     D               ;GET THE TRAILING SIGN INFO BACK
                        59380           POP     B               ;GET # OF SIG DIGITS AND "B" FIELD SPEC BACK
                        59400           MOV     A,C             ;PRINT TRAILING ZEROS IF THE FIELD LENGTH IS
                        59420           SUB     E               ; LONGER THAN THE NUMBER OF DIGITS WE CAN PRINT
                        59440           CP      FOTZRC          ;THE DECIMAL POINT COULD COME OUT IN HERE
                        59460           POP     PSW             ;GET THE "C" FIELD SPEC BACK
                        59480           CZ      FOFXIS          ;IF C#0, THE LAST THING WAS A DECIMAL POINT,
                        59500                                   ; SO IGNORE IT,  ALL WE CARE ABOUT IS THE
                        59520                                   ; "DCX  H" AND NOT THE "MOV     M,B" AT FOFXIS
                        59540           POP     PSW             ;GET THE EXPONENT BACK
                        59560           ADD     E               ;SCALE IT CORRECTLY
                        59580           SUB     B
                        59600           SUB     D
                        59620           PUSH    B               ;SAVE THE "B" FIELD SPEC
                        59640           CALL    FOFLDN          ;PUT THE EXPONENT IN THE BUFFER
                        59660           XCHG                    ;GET THE POINTER TO THE END IN (HL)
                        59680                                   ; IN CASE WE HAVE A TRAILING SIGN
                        59700           POP     D               ;GET THE "B" FIELD SPEC IN D, PUT ON A POSSIBLE
                        59720           JMP     FOUTTS          ; TRAILING SIGN AND WE ARE DONE
                        59740
                        59760           ;NORMALIZE THE NUMBER IN THE FAC SO ALL THE DIGITS ARE IN THE INTEGER
                        59780           ;PART,  RETURN THE BASE 10 EXPONENT IN A
                        59800           ;D,E ARE LEFT UNALTERED
FOUTNV: 59820           PUSH    D               ;SAVE (DE)
                        59840           LDA     VALTYP          ;GET WHAT KIND OF VALUE WE HAVE
                        59860           CPI     4
                        59880           JNZ     FOUTND          ;WE HAVE A DBL
                        59900           ;NORMALIZE A SNG
                        59920           XRA     A               ;ZERO THE EXPONENT
                        59940           PUSH    PSW             ;SAVE IT
                        59960           CALL    FOUNSC          ;IS THE FAC TOO BIG OR TOO SMALL?
FOUNS1: 59980           MVIRI   221,103,117,370 ;GET 99999,9499 TO SEE IF THE FAC IS BIG
                        60000           CALL    FCOMP           ; ENOUGH YET
                        60020           JPO     FOUNS3          ;IT IS, WE ARE DONE
                        60040           POP     PSW             ;IT ISN'T, MULTIPLY BY TEN
                        60060           CALL    FINMLT
                        60080           PUSH    PSW             ;SAVE THE EXPONENT AGAIN
                        60100           JMP     FOUNS1          ;NOW SEE IF IT IS BIG ENOUGH
FOUNS2: 60120           POP     PSW             ;THE FAC IS TOO BIG, GET THE EXPONENT
                        60140           CALL    FINDIV          ;DIVIDE IT BY TEN
                        60160           PUSH    PSW             ;SAVE THE EXPONENT AGAIN
                        60180           CALL    FOUNSC          ;SEE IF THE FAC IS SMALL ENOUGH
FOUNS3: 60200           POP     PSW             ;WE ARE DONE, GET THE EXPONENT BACK
                        60220           POP     D               ;GET (DE) BACK
                        60240           RET                     ;ALL DONE
                        60260           ;HERE TO SEE IF THE FAC IS SMALL ENOUGH YET
FOUNSC: 60280           MVIRI   224,164,043,367 ;GET 999999,499 TO SEE IF THE FAC IS TOO BIG
                        60300           CALL    FCOMP
```



```
                        60320           POP     H               ;GET THE RETURN ADRESS OFF THE STACK
                        60340           JPO     FOUNS2          ;IT IS TOO BIG, MAKE IT SMALLER
                        60360           PCHL                    ;IT IS SMALL ENOUGH, RETURN
                        60380           ;HERE TO NORMALIZE A DBL NUMBER
                        60400           >
                        60420   PAGE
```

```
                                        60440   SUBTTL  EXPONENTIATION AND THE SQUARE ROOT FUNCTION
                                        60460   IFE     EXTFNC,<
                                        60480           ;SQUARE ROOT FUNCTION  ---  X=SQR(A)
                                        60500           ;WE FIRST SCALE THE ARGUMENT TO BETWEEN .5 AND 2 BY LOOKING AT THE
                                        60520           ;EXPONENT AND USING SQR(M*2^(2*N))=2^N*SQR(M),  THEN NEWTON'S METHOD
                                        60540           ;IS USED TO COMPUTE SQR(M),  THE EXPONENT IS SAVED TO SCALE THE
                                        60560           ;RESULT AT THE END.
                                        60580           ;NEWTON'S METHOD FOR SQUARE ROOT:
                                        60600           ;  X(0)=A
                                        60620           ;  X(N+1)=(X(N)+A/X(N))/2
                                        60640   SQR:    FSIGN                   ;CHECK FOR ERROR CONDITION
                                        60660           JM      FCERR           ;CAN'T TAKE SQR OF NEGATIVE NUMBER
                                        60680           RZ                      ;  0=SQR(0)
                                        60700           LXI     H,FAC           ;SCALE ARGUMENT TO BETWEEN .5 AND 2
                                        60720           MOV     A,M             ;GET EXPONENT
                                        60740           RAR                     ;GET EXPONENT OF SCALE FACTOR
                                        60760                                   ;USE  SQR(M*2^(2*N))=2^N*SQR(M)
                                        60780           PUSH    PSW             ;SAVE IT
                                        60800           PUSH    H               ;SAVE POINTER TO EXPONENT
                                        60820           MVI     A,100           ;SET EXPONENT OF SCALED DOWN NUMBER
                                        60840           RAL
                                        60860           MOV     M,A             ;REPLACE IT
                                        60880           LXI     H,FBUFFR        ;SAVE A
                                        60900           CALL    MOVMF
                                        60920           MVI     A,4             ;SET ITERATION COUNT
                                        60940   SQR1:   PUSH    PSW             ;SAVE COUNT
                                        60960           CALL    PUSHF           ;SAVE X(N)
                                        60980           LXI     H,FBUFFR        ;COMPUTE A/X(N)
                                        61000           CALL    MOVRM           ;GET A IN THE REGISTERS
                                        61020           CALL    FDIV
                                        61040           POPR
                                        61060           CALL    FADD            ;ADD IN X(N)
                                        61080           LXI     H,FHALF         ;DIVIDE BY 2
                                        61100           CALL    FMULTS
                                        61120           POP     PSW             ;GET COUNT
                                        61140           DCR     A               ;ARE WE DONE?
                                        61160           JNZ     SQR1            ;NO, DO MORE ITERATIONS
                                        61180           POP     H               ;YES, SET EXPONENT OF ANSWER
                                        61200           POP     PSW             ;GET SCALE FACTOR
                                        61220           ADI     300             ;CONVERT TO AN EXPONENT
                                        61240           ADD     M               ;ADD EXPONENT IN
                                        61260           MOV     M,A             ;REPLACE EXPONENT
                                        61280           RET>                    ;ALL DONE

                                        61340   IFN     EXTFNC,<
                                        61360           ;SUBROUTINE FOR FPWR, ATN
002340' 001000  000041                  61380   PSHNEG: LXI     H,NEG           ;GET THE ADDRESS OF NEG
002341' 000000  001175'
002342' 000000  002307'
002343' 001000  000343                  61400           XTHL                    ;SWITCH RET ADDR AND ADDR OF NEG
002344' 001000  000351                  61420           PCHL                    ;RETURN, THE ADDRESS OF NEG IS ON THE STACK
```



```
                                        61480           ;SQUARE ROOT FUNCTION
                                        61500           ;WE USE SQR(X)=X^.5
002345' 001000  000315                  61520   SQR:    CALL    PUSHF           ;SAVE ARG
002346' 000000  001205'
002347' 000000  002341'
002350' 001000  000041                  61540           LXI     H,FHALF         ;GET 1/2
002351' 000000  002312'
002352' 000000  002346'
002353' 001000  000315                  61560           CALL    MOVFM           ;  SQR(X)=X^.5
002354' 000000  001222'
002355' 000000  002351'
002356' 001000  000301                  61580   FPWRT1: POPR                    ;GET ARG IN REGISTERS, ENTRY TO FPWR IF
002357' 001000  000321                  61600                                   ; ARGUMENT IS ON STACK,  FALL INTO FPWR

                                        61660           ;EXPONENTIATION     ---     X^Y
                                        61680           ;N.B.  0^0=1
                                        61700           ;FIRST WE CHECK IF Y=0, IF SO, THE RESULT IS 1.
                                        61720           ;NEXT, WE CHECK IF X=0, IF SO, THE RESULT IS 0.
                                        61740           ;THEN WE CHECK IF X IS POSITIVE, IF NOT, WE CHECK THAT Y IS A
                                        61760           ;NEGATIVE INTEGER, AND WHETHER IT IS EVEN OR ODD.  IF Y IS A NEGATIVE
                                        61780           ;INTEGER, WE NEGATE X,  IF NOT, LOG WILL GIVE AN FC ERROR WHEN WE CALL
                                        61800           ;IT,  IF X IS NEGATIVE AND Y IS ODD, WE PUSH THE ADDRESS OF NEG ON THE
                                        61820           ;STACK SO WE WILL RETURN TO IT AND GET A NEGATIVE RESULT,  TO COMPUTE
                                        61840           ;THE RESULT WE USE X^Y=EXP(Y*LOG(X))
002360' 001000  000357                  61860   FPWR:   FSIGN                   ;SEE IF Y IS ZERO
002361' 001000  000312                  61880           JZ      EXP             ;IT IS, RESULT IS ONE
002362' 000000  002456'
002363' 000000  002354'
002364' 001000  000170                  61900           MOV     A,B             ;SEE IF X IS ZERO
002365' 001000  000267                  61920           ORA     A
002366' 001000  000312                  61940           JZ      ZERO0           ;IT IS, RESULT IS ZERO
002367' 000000  000174'
002370' 000000  002366'
002371' 001000  000325                  61960           PUSHR                   ;SAVE X ON STACK
002372' 001000  000305
002373' 001000  000171                  61980           MOV     A,C             ;CHECK THE SIGN OF X
002374' 001000  000366                  62000           ORI     177             ;TURN THE ZERO FLAG OFF
002375' 000000  000177
002376' 001000  000315                  62020           CALL    MOVRF           ;GET Y IN THE REGISTERS
002377' 000000  001240'
002400' 000000  002367'
002401' 001000  000362                  62040           JP      FPWR1           ;NO PROBLEMS IF X IS POSITIVE
002402' 000000  002422'
002403' 000000  002377'
002404' 001000  000325                  62060           PUSHR                   ;SAVE Y
002405' 001000  000305
002406' 001000  000315                  62080           CALL    INT             ;SEE IF Y IS AN INTEGER
002407' 000000  001445'
002410' 000000  002402'
002411' 001000  000301                  62100           POPR                    ;GET Y BACK
```

```
3659  002412' 001000  000321
3660  002413' 001000  000365          62120           PUSH    PSW             ;SAVE LO OF INT FOR EVEN AND ODD INFORMATION
3661  002414' 001000  000315          62140           CALL    FCOMP           ;SEE IF WE HAVE AN INTEGER
3662  002415' 000000  001317'
3663  002416' 000000  002407'
3664  002417' 001000  000341          62160           POP     H               ;GET EVEN-ODD INFORMATION
3665  002420' 001000  000174          62180           MOV     A,H             ;PUT EVEN-ODD FLAG IN CARRY
3666  002421' 001000  000037          62200           RAR
3667  002422' 001000  000341          62220   FPWR1:  POP     H               ;GET X BACK IN FAC
3668  002423' 001000  000042          62240           SHLD    FAC-1           ;STORE HO'S
3669  002424' 777777  777777*
3670  002425' 000000  002415'
3671  002426' 001000  000341          62260           POP     H               ;GET LO'S OFF STACK
3672  002427' 001000  000042          62280           SHLD    FACLO           ;STORE THEM IN FAC
3673  002430' 000000  001454*
3674  002431' 000000  002424'
3675  002432' 001000  000334          62300           CC      PSHNEG          ;NEGATE NUMBER AT END IF Y WAS ODD
3676  002433' 000000  002340'
3677  002434' 000000  002430'
3678  002435' 001000  000314          62320           CZ      NEG             ;NEGATE THE NEGATIVE NUMBER
3679  002436' 000000  001175'
3680  002437' 000000  002433'
3681  002440' 001000  000323          62340   FPWR2:  PUSHR                   ;SAVE Y AGAIN
3682  002441' 001000  000305
3683  002442' 001000  000315          62360           CALL    LOG             ;COMPUTE  EXP(Y*LOG(X))
3684  002443' 000000  000421'
3685  002444' 000000  002436'
3686  002445' 001000  000301          62380           POPR                    ;IF X WAS NEGATIVE AND Y NOT AN INTEGER THEN
3687  002446' 001000  000321
3688  002447' 001000  000315          62400           CALL    FMULT>          ; LOG WILL BLOW HIM OUT OF THE WATER
3689  002450' 000000  000517'
3690  002451' 000000  002443'
3691                                   62420   ;       JMP     EXP
3692                                   62440   PAGE
```



```
3693                                   62460   SUBTTL  EXPONENTIAL FUNCTON
3694                                   62480   IFN     EXTFNC,<
3695                                   62500           ;WE FIRST SAVE THE ORIGINAL ARGUMENT AND MULTIPLY THE FAC BY LOG2(E)
3696                                   62520           ;THE RESULT IS USED TO DETERMINE IF WE WILL GET OVERFLOW SINCE
3697                                   62540           ;EXP(X)=2^(X*LOG2(E)) WHERE LOG2(E)=LOG(E) BASE 2, THEN WE SAVE THE
3698                                   62560           ;INTEGER PART OF THIS TO SCALE THE ANSWER AT THE END, SINCE
3699                                   62580           ;2^Y*2^INT(Y)*2^(Y-INT(Y)) AND 2^INT(Y) IS EASY TO COMPUTE,  SO WE
3700                                   62600           ;NOW COMPUTE 2^(X*LOG2(E)-INT(X*LOG2(E))) BY
3701                                   62620           ;P(LN(2)*(INT(X*LOG2(E))+1)-X) WHERE P IS AN APPROXIMATION
3702                                   62640           ;POLYNOMIAL,  THE RESULT IS THEN SCALED BY THE POWER OF 2 WE
3703                                   62660           ;PREVIOUSLY SAVED,
3704  002452' 001000  000315          62680   EXP:    CALL    PUSHF           ;SAVE ARGUMENT
3705  002453' 000000  001205'
3706  002454' 000000  002450'
3707  002455' 001000  000201          62700           MOVRI   201,070,252,073 ;GET LOG(E) BASE 2,   CACULATE:
3708  002456' 000000  000070
3709  002457' 000000  000201
3710  002460' 001000  000021
3711  002461' 000000  000073
3712  002462' 000000  000252
3713  002463' 001000  000315          62720           CALL    FMULT           ;   INT(ARG/LN(2)) = INT(ARG*LOG2(E))
3714  002464' 000000  000517'
3715  002465' 000000  002453'
3716  002466' 001000  000072          62740           LDA     FAC             ;CARRY=0 IF FAC IS TOO BIG
3717  002467' 000000  001446*
3718  002470' 000000  002464'
3719  002471' 001000  000376          62760           CPI     210             ; I.E, IF ABS(FAC) ,GE, 128
3720  002472' 000000  000210
3721  002473' 001000  000332          62780           JNC     MLOVEX          ;IT IS TOO BIG
3722  002474' 000000  001073'
3723  002475' 000000  002467'
3724  002476' 001000  000315          62800           CALL    INT             ;IS ARGUMENT TOO BIG?
3725  002477' 000000  001445'
3726  002500' 000000  002474'
3727  002501' 001000  000306          62820           ADI     200             ;CHECK FOR OVERFLOW
3728  002502' 000000  000200
3729  002503' 001000  000306          62840           ADI     2
3730  002504' 000000  000002
3731  002505' 001000  000332          62860           JC      MLOVEX          ;WE HAVE OVERFLOW!!
3732  002506' 000000  001073'
3733  002507' 000000  002477'
3734  002510' 001000  000365          62880           PUSH    PSW             ;SAVE SCALE FACTOR
3735  002511' 001000  000041          62900           LXI     H,FONE          ;ADD ONE TO THE NUMBER
3736  002512' 000000  000400'
3737  002513' 000000  002506'
3738  002514' 001000  000315          62920           CALL    FADDS
3739  002515' 000000  000003'
3740  002516' 000000  002512'
3741  002517' 001000  000315          62940           CALL    MULLN2          ;MULTIPLY BY LN(2)
3742  002520' 000000  002506'
3743  002521' 000000  002515'
3744  002522' 001000  000361          62960           POP     PSW             ;GET SCALE FACTOR OFF STACK
3745  002523' 001000  000301          62980           POPR                    ;GET ARGUMENT
```

```
3746  002524' 001000  000321
3747  002525' 001000  000365          63000           PUSH    PSW             ;PUT SCALE FACTOR BACK ON STACK
3748  002526' 001000  000315          63020           CALL    FSUB            ;SUBTRACT ORIGINAL ARG
3749  002527' 000000  000017'
3750  002530' 000000  002520'
3751  002531' 001000  000315          63040           CALL    NEG
3752  002532' 000000  001175'
3753  002533' 000000  002527'
3754  002534' 001000  000041          63060           LXI     H,EXPCON        ;EVALUATE THE APPROXIMATION POLYNOMIAL
3755  002535' 000000  002552'
3756  002536' 000000  002532'
3757  002537' 001000  000315          63080           CALL    POLY
3758  002540' 000000  002632'
3759  002541' 000000  002535'
3760  002542' 001000  000021          63100           LXI     D,SCODE         ;MULTIPLY BY 2 ** (B-1) INSTEAD OF JUST
3761  002543' 000000  001477*
3762  002544' 000000  002540'
3763  002545' 001000  000301          63120           POP     B               ; ADDING IT TO THE EXPONENT SO FMULT
3764  002546' 001000  000112          63140           MOV     C,D             ; WILL CHECK FOR EXPONENT OVERFLOW
3765  002547' 001000  000303          63160           JMP     FMULT
3766  002550' 000000  000517'
3767  002551' 000000  002543'
3768
3769                                   63200           ;CONSTANTS FOR EXP
3770  002552' 000000  000010          63220   EXPCON: 10      ;DEGREE
3771  002553' 000000  000100          63240           100     ; =.0001413161
3772  002554' 000000  000056          63260           056
3773  002555' 000000  000224          63280           224
3774  002556' 000000  000164          63300           164
3775  002557' 000000  000160          63320           160     ; .001329882
3776  002560' 000000  000117          63340           117
3777  002561' 000000  000056          63360           056
3778  002562' 000000  000167          63380           167
3779  002563' 000000  000156          63400           156     ; =.00830136
3780  002564' 000000  000002          63420           002
3781  002565' 000000  000210          63440           210
3782  002566' 000000  000172          63460           172
3783  002567' 000000  000346          63480           346     ; .04165735
3784  002570' 000000  000240          63500           240
3785  002571' 000000  000052          63520           052
3786  002572' 000000  000174          63540           174
3787  002573' 000000  000120          63560           120     ; =.1666653
3788  002574' 000000  000252          63580           252
3789  002575' 000000  000252          63600           252
3790  002576' 000000  000176          63620           176
3791  002577' 000000  000377          63640           377     ; .4999999
3792  002600' 000000  000377          63660           377
3793  002601' 000000  000177          63680           177
3794  002602' 000000  000177          63700           177
3795  002603' 000000  000000          63720           000     ; =1.0
3796  002604' 000000  000000          63740           000
3797  002605' 000000  000200          63760           200
3798  002606' 000000  000201          63780           201
```



```
3799  002607' 000000  000000          63800           000     ; 1.0
3800  002610' 000000  000000          63820           000
3801  002611' 000000  000000          63840           000
3802  002612' 000000  000201          63860           201>
3803                                   63880   PAGE
```

```
                                        63900   SUBTTL  POLYNOMIAL EVALUATOR AND THE RANDOM NUMBER GENERATOR
                                        63920   IFN     EXTFNC,<
                                        63940           ;EVALUATE P(X**2)*X
                                        63960           ;POINTER TO DEGREE+1 IS IN (HL)
                                        63980           ;THE CONSTANTS FOLLOW THE DEGREE
                                        64000           ;CONSTANTS SHOULD BE STORED IN REVERSE ORDER, FAC HAS X
                                        64020           ;WE COMPUTE:
                                        64040           ; C0*X+C1*X**3+C2*X**5+C3*X**7+...+C(N)*X**(2*N+1)
002615' 001000  000315                  64060   POLYX:  CALL    PUSHF           ;SAVE X
002614' 000000  001205'
002615' 000000  002550'
002616' 001000  000021                  64080           LXI     D,FMULTT        ;PUT ADDRESS OF FMULTT ON STACK SO WHEN WE
002617' 000000  000515'
002620' 000000  002614'
002621' 001000  000325                  64100           PUSH    D               ; RETURN WE WILL MULTIPLY BY X
002622' 001000  000345                  64120           PUSH    H               ;SAVE CONSTANT POINTER
002623' 001000  000315                  64140           CALL    MOVRF           ;SQUARE X
002624' 000000  001240'
002625' 000000  002617'
002626' 001000  000315                  64160           CALL    FMULT
002627' 000000  000517'
002630' 000000  002624'
002631' 001000  000341                  64180           POP     H               ;GET CONSTANT POINTER
                                        64200                                   ;FALL INTO POLY


                                        64260           ;POLYNOMIAL EVALUATOR
                                        64280           ;POINTER TO DEGREE+1 IS IN (HL), IT IS UPDATED
                                        64300           ;THE CONSTANTS FOLLOW THE DEGREE
                                        64320           ;CONSTANTS SHOULD BE STORED IN REVERSE ORDER, FAC HAS X
                                        64340           ;WE COMPUTE:
                                        64360           ; C0+C1*X+C2*X**2+C3*X**3+...+C(N-1)*X**(N-1)+C(N)*X**N
002632' 001000  000315                  64380   POLY:   CALL    PUSHF           ;SAVE X
002633' 000000  001205'
002634' 000000  002627'
002635' 001000  000176                  64400           MOV     A,M             ;GET DEGREE
002636' 001000  000043                  64420           INX     H               ;INCREMENT POINTER TO FIRST CONSTANT
002637' 001000  000315                  64440           CALL    MOVFM           ;MOVE FIRST CONSTANT TO FAC
002640' 000000  001222'
002641' 000000  002633'
002642' 001000  000006                  64460           XWD     1000,006        ;"MVI   B" OVER NEXT BYTE
002643' 001000  000361                  64480   POLY1:  POP     PSW             ;GET DEGREE
002644' 001000  000301                  64500           POPR                    ;GET X
002645' 001000  000321
002646' 001000  000075                  64520           DCR     A               ;ARE WE DONE?
002647' 001000  000310                  64540           RZ                      ;YES, RETURN
002650' 001000  000325                  64560           PUSHR                   ;NO, SAVE X
002651' 001000  000305
002652' 001000  000365                  64580           PUSH    PSW             ;SAVE DEGREE
002653' 001000  000345                  64600           PUSH    H               ;SAVE CONSTANT POINTER
002654' 001000  000315                  64620           CALL    FMULT           ;EVALUATE THE POLY, MULTIPLY BY X
002655' 000000  000517'
002656' 000000  002640'
```



```
002657' 001000  000341                  64640           POP     H               ;GET LOCATION OF CONSTANTS
002660' 001000  000315                  64660           CALL    MOVRM           ;GET CONSTANT
002661' 000000  001243'
002662' 000000  002655'
                                        64680           PUSH    H               ;STORE LOCATION OF CONSTANTS SO FADD AND FMULT
002663' 001000  000345
002664' 001000  000315                  64700           CALL    FADD            ; WILL NOT SCREW THEM UP, ADD IN CONSTANT
002665' 000000  000025'
002666' 000000  002661'
002667' 001000  000341                  64720           POP     H               ;MOVE CONSTANT POINTER TO NEXT CONSTANT
002670' 001000  000303                  64740           JMP     POLY1>          ;SEE IF DONE
002671' 000000  002643'
002672' 000000  002665'


                                        64800           ;PSUEDO-RANDOM NUMBER GENERATOR
                                        64820           ;IF ARG=0, THE LAST RANDOM NUMBER GENERATED IS RETURNED
                                        64840           ;IF ARG .LT. 0, A NEW SEQUENCE OF RANDOM NUMBERS IS STARTED
                                        64860           ; USING THE ARGUMENT
                                        64880           ;TO FORM THE NEXT RANDOM NUMBER IN THE SEQUENCE, WE MULTIPLY THE
                                        64900           ;PREVIOUS RANDOM NUMBER BY A RANDOM CONSTANT, AND ADD IN ANOTHER
                                        64920           ;RANDOM CONSTANT.  THEN THE MO AND LO BYTES ARE SWITCHED, THE
                                        64940           ;EXPONENT IS PUT WHERE IT WILL BE SHIFTED IN BY NORMAL, AND THE
                                        64960           ;EXPONENT IN THE FAC SET TO 200 SO THE RESULT WILL BE LESS THAN 1.
                                        64980           ;THIS IS THEN NORMALIZED AND SAVED FOR THE NEXT TIME.
                                        65000           ;THE MO AND LO BYTES WERE SWITCHED SO WE HAVE A RANDOM CHANCE OF
                                        65020           ;GETTING A NUMBER LESS THAN OR GREATER THAN .5
002673'                                 65040   RND:
                                        65060   IFN     LENGTH-2,<
002673' 001000  000357                  65080           FSIGN>          ;GET SIGN OF ARG
                                        65100   IFE     LENGTH-2,<
                                        65120           CALL    VSIGN           ;GET THE SIGN OF THE ARG
                                        65140           PUSH    PSW             ;SAVE THE SIGN
                                        65160           CM      FNCSNG          ;IF IT IS NEGATIVE, FORCE IT TO BE A SNG
                                        65180           MVI     A,4             ; SINCE WE WILL USE IT
                                        65200           STA     VALTYP          ;MAKE SURE THE RESULT IS "SINGLE PRECISION"
                                        65220           POP     PSW>            ;GET THE SIGN BACK
002674' 001000  000372                  65240           JM      RND1            ;START NEW SEQUENCE IF NEGATIVE
002675' 000000  002725'
002676' 000000  002671'
002677' 001000  000041                  65260           LXI     H,RNDX          ;GET LAST NUMBER GENERATED
002700' 000000  002752'
002701' 000000  002675'
002702' 001000  000315                  65280           CALL    MOVFM
002703' 000000  001222'
002704' 000000  002700'
002705' 001000  000310                  65300           RZ                      ;RETURN LAST NUMBER GENERATED IF ZERO
                                        65320   IFE     EXTFNC,<
                                        65340           CALL    FMULTS>         ;MULTIPLY BY CONSTANT A
                                        65360   IFN     EXTFNC,<
002706' 001000  000315                  65380           CALL    MOVRM
002707' 000000  001243'
002710' 000000  002703'
```

```
                                                65400           CALL    FMULT>
                                                65420           MOVRI   150,050,261,106 ;ADD IN CONSTANT OF ORDER 2^(-24)
                                                65440           CALL    FADD
                                                65460   RND1:   CALL    MOVRF           ;SWITCH MO AND LO BYTES,
                                                65480           MOV     A,E             ;GET LO
                                                65500           MOV     E,C             ;PUT MO IN LO BYTE
                                                65520           MOV     C,A             ;PUT LO IN MO BYTE
                                                65540           MVI     M,200           ;MAKE RESULT POSITIVE
                                                65560           DCX     H               ;GET POINTER TO EXPONENT
                                                65580           MOV     B,M             ;PUT EXPONENT IN OVERFLOW POSITION
                                                65600           MVI     M,200           ;SET EXP SO RESULT WILL BE BETWEEN 0 AND 1
                                                65620           CALL    NORMAL          ;NORMALIZE THE RESULT
                                                65640           LXI     H,RNDX          ;SAVE RANDOM NUMBER GENERATED FOR NEXT
                                                65660           JMP     MOVMF           ; TIME
                                                65700           ;CONSTANTS AND STORAGE FOR RND
                                                65720   RNDX:   122     ;LAST RANDOM NUMBER GENERATED, BETWEEN 0 AND 1
                                                65740           307
                                                65760           117
                                                65780           200
                                                65800           172     ;RANDOM NUMBER OF ORDER 2^24
                                                65820           104
                                                65840           065
                                                65860           230
                                                65880   PAGE
```



```
                                                65900   SUBTTL  SINE, COSINE AND TANGENT FUNCTIONS
                                                65920   IFN     EXTFNC,<
                                                65940           ;COSINE FUNCTION
                                                65960           ;IDEA:  USE COS(X)=SIN(X+PI/2)
                                                65980   COS:    LXI     H,PI2           ;ADD PI/2 TO FAC
                                                66000           CALL    FADDS>
                                                66020                                   ;FALL INTO SIN
                                                66080           ;SINE FUNCTION
                                                66100           ;IDEA:  USE IDENTITIES TO GET FAC IN QUADRANTS I OR IV
                                                66120           ;THE FAC IS DIVIDED BY 2*PI AND THE INTEGER PART IS IGNORED BECAUSE
                                                66140           ;SIN(X+2*PI)=SIN(X),  THEN THE ARGUMENT CAN BE COMPARED WITH PI/2 BY
                                                66160           ;COMPARING THE RESULT OF THE DIVISION WITH PI/2/(2*PI)=1/4.
                                                66180           ;IDENTITIES ARE THEN USED TO GET THE RESULT IN QUADRANTS I OR IV.
                                                66200           ;AN APPROXIMATION POLYNOMIAL IS THEN USED TO COMPUTE SIN(X).
                                                66220   SIN:    CALL    PUSHF           ;DIVIDE FAC BY 2*PI
                                                66240           MOVRI   203,111,017,333 ;AFTER DIVIDING BY 2*PI, RESULT IS
                                                66260           CALL    MOVFR           ; BETWEEN 0 AND 1
                                                66280           POPR
                                                66300           CALL    FDIV
                                                66320           CALL    PUSHF           ;DISREGARD INTEGER PART SINCE SIN
                                                66340           CALL    INT             ; IS PERIODIC WITH PERIOD 2*PI
                                                66360           POPR
                                                66380           CALL    FSUB
                                                66400   IFE     EXTFNC,<
                                                66420           LXI     B,177*400+$CODE ;GET 1/4
                                                66440           MOV     D,C
                                                66460           MOV     E,C
                                                66480           CALL    FSUB>
```

```
                                66500   IFN     EXTFNC,<
003024' 001000  000041          66520           LXI     H,FR4           ;SEE WHAT QUADRANT WE ARE IN
003025' 000000  003074'
003026' 000000  003022'
003027' 001000  000315          66540           CALL    FSUBS>
003030' 000000  000011'
003031' 000000  003025'
003032' 001000  000357          66560           FSIGN
003033' 001000  000067          66580           STC                     ;SET QUADRANT I FLAG
003034' 001000  000362          66600           JP      SIN1            ;FIRST QUADRANT, GET BACK ORIGINAL X
003035' 000000  003044'
003036' 000000  003030'
003037' 001000  000315          66620           CALL    FADDH           ;ADD 1/2
003040' 000000  000000'
003041' 000000  003035'
003042' 001000  000357          66640           FSIGN
003043' 001000  000267          66660           ORA     A               ;CLEAR CARRY
003044' 001000  000365          66680   SIN1:   PUSH    PSW             ;SAVE QUADRANT FLAG
003045' 001000  000364          66700           CP      NEG             ;NEGATE IF IN QUADRANTS I, II OR III
003046' 000000  001175'
003047' 000000  003040'
                                66720   IFE     EXTFNC,<
                                66740           LXI     B,177*400+$CODE ;GET 1/4
                                66760           MOV     D,C
                                66780           MOV     E,C
                                66800           CALL    FADD>
                                66820   IFN     EXTFNC,<
003050' 001000  000041          66840           LXI     H,FR4           ;ADD 1/4, IN QUADRANTS II, III
003051' 000000  003074'
003052' 000000  003046'
                                66860                                   ; USE THE IDENTITY: SIN(PI-X)=SIN(X)
                                66880                                   ;IN QUADRANT IV, USE THE IDENTITY:
003053' 001000  000315          66900           CALL    FADDS>          ;   SIN(X-2*PI)=SIN(X)
003054' 000000  000003'
003055' 000000  003051'
003056' 001000  000361          66920           POP     PSW             ;GET QUADRANT FLAG
003057' 001000  000324          66940           CNC     NEG             ;NEGATE IF IN QUADRANTS II, III OR IV
003060' 000000  001175'
003061' 000000  003054'
                                66960   IFE     EXTFNC,<                ;EVALUATE APPROXIMATION POLYNOMIAL
                                66980           CALL    PUSHF           ;SAVE X
                                67000           CALL    MOVRF           ;SQUARE X
                                67020           CALL    FMULT
                                67040           CALL    PUSHF           ;SAVE X^2
                                67060           LXI     H,SINCON
                                67080           CALL    MOVFM           ;MOVE FIRST CONSTANT INTO FAC
                                67100           POPR                    ;GET X^2
                                67120           MVI     A,4             ;GET DEGREE
                                67140   POLY1:  PUSH    PSW             ;SAVE DEGREE
                                67160           PUSHR                   ;SAVE X^2
                                67180           PUSH    H               ;SAVE CONSTANT POINTER
                                67200           CALL    FMULT           ;EVALUATE THE POLY, MULTIPLY BY X^2
                                67220           POP     H               ;GET POINTER TO CONSTANTS
```



```
                                67240           CALL    MOVRM           ;GET CONSTANT
                                67260           PUSH    H               ;SAVE POINTER
                                67280           CALL    FADD            ;ADD IN CONSTANT
                                67300           POP     H               ;MOVE POINTER TO NEXT CONSTANT
                                67320           POPR                    ;GET X^2
                                67340           POP     PSW             ;GET DEGREE
                                67360           DCR     A               ;SEE IF DONE
                                67380           JNZ     POLY1           ;NO, DO NEXT TERM
                                67400           JMP     FMULTT>         ;MULTIPLY BY X AND WE ARE DONE
                                67420   IFN     EXTFNC,<
003062' 001000  000041          67440           LXI     H,SINCON        ;CALCULATE THE SIN BY EVALUATING
003063' 000000  003100'
003064' 000000  003060'
003065' 001000  000303          67460           JMP     POLYX>          ;  THE APPROXIMATION POLYNOMIAL
003066' 000000  002613'
003067' 000000  003063'
                                67480           ;CONSTANTS FOR SIN, COS
                                67500   IFN     EXTFNC,<
003070' 000000  000333          67520   PI2:    333     ; PI/2
003071' 000000  000017          67540           017
003072' 000000  000111          67560           111
003073' 000000  000201          67580           201
003074' 000000  000000          67600   FR4:    000     ; 1/4
003075' 000000  000000          67620           000
003076' 000000  000000          67640           000
003077' 000000  000177          67660           177>
003100'                         67680   SINCON:
                                67700   IFN     EXTFNC,<
003100' 000000  000005          67720           5>      ;DEGREE
003101' 000000  000272          67740           272     ; 39.701067
003102' 000000  000327          67760           327
003103' 000000  000036          67780           036
003104' 000000  000206          67800           206
003105' 000000  000144          67820           144     ; -76.57498
003106' 000000  000046          67840           046
003107' 000000  000231          67860           231
003110' 000000  000207          67880           207
003111' 000000  000130          67900           130     ; 81.60223
003112' 000000  000064          67920           064
003113' 000000  000043          67940           043
003114' 000000  000207          67960           207
003115' 000000  000340          67980           340     ; -41.34168
003116' 000000  000135          68000           135
003117' 000000  000245          68020           245
003120' 000000  000206          68040           206
003121' 000000  000332          68060           332     ; 6.283185
003122' 000000  000017          68080           017
003123' 000000  000111          68100           111
003124' 000000  000203          68120           203


                                68180   IFN     EXTFNC,<
                                68200           ;TANGENT FUNCTION
```

```
                                        68220           ;TAN(X)=SIN(X)/COS(X)
4114  003125' 001000  000315           68240   TAN:    CALL    PUSHF           ;SAVE ARG
4115  003126' 000000  001205'
4116  003127' 000000  003066'
4117  003130' 001000  000315           68260           CALL    SIN             ;   TAN(X)=SIN(X)/COS(X)
4118  003131' 000000  002770'
4119  003132' 000000  003126'
4120  003133' 001000  000301           68280           POP     B               ;GET X OFF STACK
4121  003134' 001000  000341           68300           POP     H               ;PUSHF SMASHES (DE)
4122  003135' 001000  000315           68320           CALL    PUSHF
4123  003136' 000000  001205'
4124  003137' 000000  003131'
4125  003140' 001000  000353           68340           XCHG                    ;GET LO'S WHERE THEY BELONG
4126  003141' 001000  000315           68360           CALL    MOVFR
4127  003142' 000000  001225'
4128  003143' 000000  003136'
4129  003144' 001000  000315           68380           CALL    COS
4130  003145' 000000  002762'
4131  003146' 000000  003142'
4132  003147' 001000  000303           68400           JMP     FDIVT>
4133  003150' 000000  000653'
4134  003151' 000000  003145'
4135                                    68420   PAGE
```



```
4136                                    68440   SUBTTL  ARCTANGENT FUNCTION
4137                                    68460   IFN     EXTFNC,<
4138                                    68480           ;IDEA: USE IDENTITIES TO GET ARG BETWEEN 0 AND 1 AND THEN USE AN
4139                                    68500           ;APPROXIMATION POLYNOMIAL TO COMPUTE ARCTAN(X)
4140  003152' 001000  000357           68520   ATN:    FSIGN                   ;SEE IF ARG IS NEGATIVE
4141  003153' 001000  000374           68540           CM      PSHNEG          ;IF ARG IS NEGATIVE, USE:
4142  003154' 000000  002340'
4143  003155' 000000  003150'
4144  003156' 001000  000374           68560           CM      NEG             ;   ARCTAN(X)=-ARCTAN(-X)
4145  003157' 000000  001175'
4146  003160' 000000  003154'
4147  003161' 001000  000072           68580           LDA     FAC             ;SEE IF FAC .GT. 1
4148  003162' 000000  002467*
4149  003163' 000000  003157'
4150  003164' 001000  000376           68600           CPI     201
4151  003165' 000000  000201
4152  003166' 001000  000332           68620           JC      ATN2
4153  003167' 000000  003205'
4154  003170' 000000  003162'
4155  003171' 001000  000001           68640           LXI     B,201*400+$CODE ;GET THE CONSTANT 1
4156  003172' 000000  100400*
4157  003173' 000000  003167'
4158  003174' 001000  000121           68660           MOV     D,C
4159  003175' 001000  000131           68680           MOV     E,C             ;COMPUTE RECIPROCAL TO USE THE IDENTITY:
4160  003176' 001000  000315           68700           CALL    FDIV            ;   ARCTAN(X)=PI/2-ARCTAN(1/X)
4161  003177' 000000  000655'
4162  003200' 000000  003172'
4163  003201' 001000  000041           68720           LXI     H,FSUBS         ;PUT FSUBS ON THE STACK SO WE WILL RETURN
4164  003202' 000000  000011'
4165  003203' 000000  003177'
4166  003204' 001000  000345           68740           PUSH    H               ; TO IT AND SUBTRACT THE REULT FROM PI/2
4167  003205' 001000  000041           68760   ATN2:   LXI     H,ATNCON        ;EVALUATE APPROXIMATION POLYNOMIAL
4168  003206' 000000  003217'
4169  003207' 000000  003202'
4170  003210' 001000  000315           68780           CALL    POLYX
4171  003211' 000000  002613'
4172  003212' 000000  003206'
4173  003213' 001000  000041           68800           LXI     H,PI2           ;GET POINTER TO PI/2 IN CASE WE HAVE TO
4174  003214' 000000  003070'
4175  003215' 000000  003E11'
4176  003216' 001000  000311           68820           RET                     ; SUBTRACT THE RESULT FROM PI/2
4177
4178                                    68860           ;CONSTANTS FOR ATN
4179  003217' 000000  000011           68880   ATNCON: 11      ;DEGREE
4180  003220' 000000  000112           68900           112     ; ,002866226
4181  003221' 000000  000327           68920           327
4182  003222' 000000  000073           68940           073
4183  003223' 000000  000170           68960           170
4184  003224' 000000  000002           68980           002     ; =-,01616574
4185  003225' 000000  000156           69000           156
4186  003226' 000000  000204           69020           204
4187  003227' 000000  000173           69040           173
4188  003230' 000000  000376           69060           376     ; ,04290961
```

```
 4189  003231' 000000  000301          69080          301
 4190  003232' 000000  000057          69100          057
 4191  003233' 000000  000174          69120          174
 4192  003234' 000000  000164          69140          164     ; -.07528964
 4193  003235' 000000  000061          69160          061
 4194  003236' 000000  000232          69180          232
 4195  003237' 000000  000175          69200          175
 4196  003240' 000000  000204          69220          204     ; .1065626
 4197  003241' 000000  000075          69240          075
 4198  003242' 000000  000132          69260          132
 4199  003243' 000000  000175          69280          175
 4200  003244' 000000  000310          69300          310     ; -.142089
 4201  003245' 000000  000177          69320          177
 4202  003246' 000000  000221          69340          221
 4203  003247' 000000  000176          69360          176
 4204  003250' 000000  000344          69380          344     ; .1999355
 4205  003251' 000000  000273          69400          273
 4206  003252' 000000  000114          69420          114
 4207  003253' 000000  000176          69440          176
 4208  003254' 000000  000154          69460          154     ; -.3333315
 4209  003255' 000000  000252          69480          252
 4210  003256' 000000  000252          69500          252
 4211  003257' 000000  000177          69520          177
 4212  003260' 000000  000000          69540          000     ; 1.0
 4213  003261' 000000  000000          69560          000
 4214  003262' 000000  000000          69580          000
 4215  003263' 000000  000201          69600          201>
 4216                                  69620  PAGE
```



```
 4217                                  69640  SUBTTL  SYSTEM INITIALIZATION CODE
 4218                                  69660  RADIX   10              ;IN ALL NON-MATH PACKAGE CODE
 4219                                  69680  ;THIS IS THE SYSTEM INITIALIZATION CODE
 4220                                  69700  ;IT SHOULD BE LOADED AT THE END OF THE BASIC
 4221                                  69720  ;INTERPRETER
 4222
 4223                                  69760  INTERNAL        INIT
 4224
 4225                                  69800  EXTERNAL        CRDO,LINGET,QINLIN,READY,SCRTCH,STROUT,REASON,BUF
 4226                                  69820  EXTERNAL        SNERR,OMERR,ILLFUN
 4227
 4228                  145040          69860  FUNIO==<^D256*^D312>+^D48+$CODE
 4229  003264'                         69880  INITSA: BLOCK   10
 4230  003276' 001000  000041          69900  INITAT: LXI     H,AUTTXT
 4231  003277' 000000  004035'
 4232  003300' 000000  003214'
 4233  003301' 001000  000315          69920          CALL    STROUT
 4234  003302' 000000  001763*
 4235  003303' 000000  003277'
 4236  003304'                         69940  INIT:
 4237                                  69960  IFN     REALIO,<
 4238  003304' 001000  000333          69980          IN      1               ;IGNORE GARBAGE CHARACTER IN INTERFACE
 4239  003305' 000000  000001
 4240  003306' 001000  000333          70000          IN      ^D255           ;SEE WHAT KIND OF I/O HE IS
 4241  003307' 000000  000377
 4242  003310' 001000  000346          70020          ANI     ^D100           ;IS HE FUNNY TTY?
 4243  003311' 000000  000100
 4244  003312' 001000  003312          70040          JZ      NOTSIO
 4245  003313' 000000  003343'
 4246  003314' 000000  003302'
 4247  003315' 001000  000041          70060          LXI     H,FUNIO
 4248  003316' 000000  145040*
 4249  003317' 000000  003313'
 4250  003320' 001000  000042          70080          SHLD    CNLCA2##+2
 4251  003321' 000000  000002*
 4252  003322' 000000  003316'
 4253  003323' 001000  000046          70100          MVI     M,^D310
 4254  003324' 000000  000310
 4255  003325' 001000  000042          70120          SHLD    CNLCA3##+2
 4256  003326' 000000  000002*
 4257  003327' 000000  003321'
 4258                                  70140  IFN     LENGTH,<
 4259  003330' 001000  000046          70160          MVI     M,^D304         ;SUBSTITUE "CNZ"
 4260  003331' 000000  000304
 4261  003332' 001000  000042          70180          SHLD    CNLCA4##+2>
 4262  003333' 000000  000002*
 4263  003334' 000000  003326'
 4264                  145002          70200  FUNIO==SCODE+<^D256*^D312>+2
 4265  003335' 001000  000041          70220          LXI     H,FUNIO
 4266  003336' 000000  145002*
 4267  003337' 000000  003335'
 4268  003340' 001000  000042          70240          SHLD    CNLCA1##+2
 4269  003341' 000000  000002*
```

```
 4270  003342' 000000  003336'
 4271  003343' 001000  000333          70260  NOTSIO: IN      ^O255
 4272  003344' 000000  000377
 4273  003345' 001000  000346          70280          ANI     ^O40
 4274  003346' 000000  000040
 4275  003347' 001000  000312          70300          JZ      NOTPIO
 4276  003350' 000000  003400'
 4277  003351' 000000  003341'
 4278                  145002          70320  FUNIO==$CODE+<^O256*^O312>+2
 4279  003352' 001000  000041          70340          LXI     H,FUNIO
 4280  003353' 000000  145002*
 4281  003354' 000000  003350'
 4282  003355' 001000  000042          70360          SHLD    CNLCA2+2
 4283  003356' 000000  000002*
 4284  003357' 000000  003353'
 4285  003360' 001000  000046          70380          MVI     H,^O310
 4286  003361' 000000  000310
 4287  003362' 001000  000042          70400          SHLD    CNLCA3+2
 4288  003363' 000000  000002*
 4289  003364' 000000  003356'
 4290                                   70420  IFN     LENGTH,<
 4291  003365' 001000  000046          70440          MVI     H,^O304
 4292  003366' 000000  000304
 4293  003367' 001000  000042          70460          SHLD    CNLCA4+2>
 4294  003370' 000000  000002*
 4295  003371' 000000  003363'
 4296                  145001          70480  FUNIO==$CODE+<^O256*^O312>+1
 4297  003372' 001000  000041          70500          LXI     H,FUNIO
 4298  003373' 000000  145001*
 4299  003374' 000000  003370'
 4300  003375' 001000  000042          70520          SHLD    CNLCA1+2
 4301  003376' 000000  000002*
 4302  003377' 000000  003373'
 4303  003400'                         70540  NOTPIO:>
 4304  003400' 001000  000041          70560          LXI     H,$CODE+^O65535
 4305  003401' 000000  177777*
 4306  003402' 000000  003376'
 4307  003403' 001000  000042          70580          SHLD    CURLIN##        ;IN CASE OF ERROR MESSAGE
 4308  003404' 000000  000000*
 4309  003405' 000000  003401'
 4310  003406' 001000  000041          70600          LXI     H,TSTACK        ;SET UP TEMP STACK
 4311  003407' 000000  004315'
 4312  003410' 000000  003404'
 4313  003411' 001000  000371          70620          SPHL
 4314  003412' 001000  000042          70640          SHLD    STKTOP##
 4315  003413' 000000  000000*
 4316  003414' 000000  003407'
 4317                                   70660  IFN     CONTRW,<
 4318  003415' 001000  000257          70680          XRA     A
 4319  003416' 001000  000062          70700          STA     CNTWFL##>
 4320  003417' 000000  000000*
 4321  003420' 000000  003413'
 4322  003421' 001000  000315          70720          CALL    CRDO            ;TYPE A CR
```



```
 4323  003422' 000000  000000*
 4324  003423' 000000  003417'
 4325                                   70740  IFN     STRING,<
 4326  003424' 001000  000041          70760          LXI     H,TEMPST##
 4327  003425' 000000  000000*
 4328  003426' 000000  003422'
 4329  003427' 001000  000042          70780          SHLD    TEMPPT##>
 4330  003430' 000000  000000*
 4331  003431' 000000  003425'
 4332                                   70800  IFN     REALIO,<
 4333  003432' 001000  000041          70820          LXI     H,MEMORY        ;ASK HOW MUCH MEMORY AVAILABLE
 4334  003433' 000000  004243'
 4335  003434' 000000  003430'
 4336  003435' 001000  000315          70840          CALL    STROUT          ;CALL THE STRING PRINTER
 4337  003436' 000000  003302*
 4338  003437' 000000  003433'
 4339  003440' 001000  000315          70860          CALL    QINLIN          ;GET A LINE OF INPUT
 4340  003441' 000000  000000*
 4341  003442' 000000  003436'
 4342  003443' 001000  000327          70880          CHRGET                  ;GET A CHAR
 4343  003444' 001000  000376          70900          CPI     "A"
 4344  003445' 000000  000101
 4345  003446' 001000  000312          70920          JZ      INITAT
 4346  003447' 000000  003276'
 4347  003450' 000000  003441'
 4348  003451' 001000  000267          70940          ORA     A
 4349  003452' 001000  000302          70960          JNZ     USEDE9          ;NON ZERO, DONT USE DEFAULT
 4350  003453' 000000  003501'
 4351  003454' 000000  003447'
 4352  003455' 001000  000041          70980          LXI     H,LASTMM
 4353  003456' 000000  004257'
 4354  003457' 000000  003453'
 4355  003460' 001000  000043          71000  LOOPMM: INX     H
 4356  003461' 001000  000076          71020          MVI     A,311
 4357  003462' 000000  000467
 4358  003463' 001000  000167          71040          MOV     M,A
 4359  003464' 001000  000276          71060          CMP     M
 4360  003465' 001000  000302          71080          JNZ     USEDEF
 4361  003466' 000000  003515'
 4362  003467' 000000  003456'
 4363  003470' 001000  000075          71100          DCR     A
 4364  003471' 001000  000167          71120          MOV     M,A
 4365  003472' 001000  000276          71140          CMP     M
 4366  003473' 001000  000312          71160          JZ      LOOPMM
 4367  003474' 000000  003460'
 4368  003475' 000000  003466'
 4369  003476' 001000  000303          71180          JMP     USEDEF
 4370  003477' 000000  003515'
 4371  003500' 000000  003474'
 4372  003501' 001000  000041          71200  USEDE9: LXI     H,BUF
 4373  003502' 000000  000000*
 4374  003503' 000000  003477'
 4375  003504' 001000  000315          71220          CALL    LINGET          ;GET DECIMAL AMOUNT OF MEMORY IN (D,E)
```

```
4376  003505'  000000  000000*
4377  003506'  000000  003502'
4378  003507'  001000  000267          71240           ORA     A
4379  003510'  001000  000302          71260           JNZ     SNERR           ;MAKE SURE HE HAS A TERMINATOR
4380  003511'  000000  000000*
4381  003512'  000000  003505'
4382  003513'  001000  000353          71280           XCHG
4383  003514'  001000  000053          71300           DCX     H
4384  003515'  001000  000053          71320   USEDEF: DCX     H>
4385                                    71340   IFE     REALIO,<
4386                                    71360           LXI     H,$CODE+16190>
4387  003516'  001000  000345          71380           PUSH    H               ;ALSO SAVE FOR LATER
4388  003517'  001000  000041          71400   TTYW:   LXI     H,TTYWID
4389  003520'  000000  004127'
4390  003521'  000000  003511'
4391  003522'  001000  000315          71420           CALL    STROUT
4392  003523'  000000  003436*
4393  003524'  000000  003520'
4394  003525'  001000  000315          71440           CALL    GINLIN
4395  003526'  000000  003441*
4396  003527'  000000  003523'
4397  003530'  001000  000327          71460           CHRGET
4398  003531'  001000  000267          71480           ORA     A
4399  003532'  001000  000312          71500           JZ      DFLENT
4400  003535'  000000  003604'
4401  003534'  000000  003526'
4402  003535'  001000  000041          71520           LXI     H,BUF
4403  003536'  000000  003502*
4404  003537'  000000  003533'
4405  003540'  001000  000315          71540           CALL    LINGET
4406  003541'  000000  003505*
4407  003542'  000000  003536'
4408  003543'  001000  000172          71560           MOV     A,D
4409  003544'  001000  000267          71580           ORA     A
4410  003545'  001000  000302          71600           JNZ     TTYW
4411  003546'  000000  003517'
4412  003547'  000000  003541'
4413  003550'  001000  000173          71620           MOV     A,E
4414  003551'  001000  000376          71640           CPI     16
4415  003552'  000000  000020
4416  003553'  001000  000332          71660           JC      TTYW
4417  003554'  000000  003517'
4418  003555'  000000  003546'
4419  003556'  001000  000062          71680           STA     LINPT1##        ;DECLARE LINPT1 EXTERNAL
4420  003557'  000000  000000*
4421  003560'  000000  003554'
4422                                    71700   IFN     LENGTH,<
4423  003561'  001000  000062          71720           STA     LINPT2##>
4424  003562'  000000  000000*
4425  003563'  000000  003557'
4426                                    71740   IFN     STRING,<
4427  003564'  001000  000062          71760           STA     LINPT3##>
4428  003565'  000000  000000*
```



```
4429  003566'  000000  003562'
4430  003567'  001000  000326          71780   MORCPS: SUI     14
4431  003570'  000000  000014
4432  003571'  001000  000322          71800           JNC     MORCPS
4433  003572'  000000  003567'
4434  003573'  000000  003565'
4435  003574'  001000  000306          71820           ADI     28
4436  003575'  000000  000034
4437  003576'  001000  000057          71840           CMA
4438  003577'  001000  000074          71860           INR     A
4439  003600'  001000  000203          71880           ADD     E
4440  003601'  001000  000062          71900           STA     LINPT4##
4441  003602'  000000  000000*
4442  003603'  000000  003572'
4443  003604'                          71920   DFLENT:
4444                                    71940   IFN     STRING,<                ;SET UP DEFAULT STRING SPACE
4445  003604'  001000  000021          71960           LXI     D,$CODE+"065536-"050+1
4446  003605'  000000  177717*
4447  003606'  000000  003602'
4448  003607'  001000  000341          71980           POP     H
4449  003610'  001000  000042          72000           SHLD    MEMSIZ##        ;SAVE IN REAL MEMORY SIZE
4450  003611'  000000  000000*
4451  003612'  000000  003605'
4452  003613'  001000  000042          72020           SHLD    FRETOP##        ;STRINGS START FROM HERE DOWN
4453  003614'  000000  000000*
4454  003615'  000000  003611'
4455  003616'  001000  000031          72040           DAD     D               ;CALC STRBOT BY SUBTRACTING 200 FROM STKTOP
4456  003617'  001000  000322          72060           JNC     OMERR           ;MUST BE POSITIVE
4457  003620'  000000  000000*
4458  003621'  000000  003614'
4459  003622'  001000  000053          72080           DCX     H               ;ONE LOWER IS STKTOP
4460  003623'  001000  000345          72100           PUSH    H>              ;SAVE IT ON STACK
4461                                    72120   IFE     EXTFNC,<
4462                                    72140
4463                                    72160   ; FUNCTION DELETION ROUTINE FOR 4K MACHINE
4464                                    72180   ; TO ADD A NEW FUNCTION JUST UPDATE TBLDO
4465                                    72200
4466                                    72220   ASKAGN: LXI     H,TBLDO         ;START OF FUNCTION TABLE
4467                                    72240   LOPASK: PUSHM                   ;PUT ON CANDIDATE FOR START OF FREE MEMORY
4468                                    72260           LXI     D,TBDASK        ;DONE ASKING?
4469                                    72280           COMPAR
4470                                    72300           JZ      FINFUN          ;YES, GET FIRST FREE MEM LOC
4471                                    72320                                   ;INTO [D,E] AND QUIT
4472                                    72340           PUSHM                   ;PUT ON MESSAGE LOCATION
4473                                    72360           XTHL                    ;[H,L]=MESSAGE POINTER
4474                                    72380                                   ;POINTER INTO TBLDO GOES ON THE STACK
4475                                    72400           CALL    STROUT          ;PRINT THE MESSAGE
4476                                    72420           CALL    GINLIN          ;SEE WHAT HE HAS TO SAY
4477                                    72440           CHRGET                  ;SEE WHAT THE FIRST CHARACTER OF INPUT WAS
4478                                    72460           POP     H               ;POP OFF POINTER INTO TBLDO
4479                                    72480           CPI     "Y"
4480                                    72500   FINFUN: POP     D               ;POP OFF CANDIDATE FOR START OF
4481                                    72520                                   ;FREE MEMORY
```

```
                                                72540           JZ      HAVFNS          ;HE WANTS IT SO WE ARE DONE
                                                72560           CPI     "N"             ;IF A BAD ANSWER
                                                72580           JNZ     ASKAGN          ;MAKE HIM START OVER
                                                72600           PUSHM                   ;PUSH ON FUNCTION CALL
                                                72620                                   ;LOCATION THAT WE FIX UP
                                                72640           XTHL                    ;[H,L]=FIXUP PLACE
                                                72660                                   ;POINTER INTO TBLDO GOES ON THE STACK
                                                72680           LXI     D,ILLFUN        ;THIS IS WHAT WE STORE
                                                72700           MOV     M,E
                                                72720           INX     H
                                                72740           MOV     M,D
                                                72760           POP     H               ;GET TBLDO POINTER
                                                72780           JMP     LOPASK>         ;GO ASK AGAIN FOR ANOTHER FUNCTION
                                                72800   IFN     EXTFNC,<
003624' 001000  000041                          72820   ASKAGN: LXI     H,FNS           ;ASK IF WANTS SIN, COS, ATN,
003625' 000000  004010'
003626' 000000  003620'
003627' 001000  000315          72840           CALL    STROUT          ;THE STRING
003630' 000000  003523*
003631' 000000  003625'
003632' 001000  000315          72860           CALL    QINLIN
003633' 000000  003532*
003634' 000000  003630'
003635' 001000  000327          72880           CHRGET
003636' 001000  000376          72900           CPI     "Y"
003637' 000000  000131
003640' 001000  000021          72920           LXI     D,INITSA        ;ASSUME NOT DELETE ANY FNS
003641' 000000  003264'
003642' 000000  003633'
003643' 001000  000312          72940           JZ      HAVFNS          ;YUP,
003644' 000000  003712'
003645' 000000  003641'
003646' 001000  000376          72960           CPI     "A"
003647' 000000  000101
003650' 001000  000312          72980           JZ      OKCHAR
003651' 000000  003660'
003652' 000000  003644'
003653' 001000  000376          73000           CPI     "N"
003654' 000000  000116
003655' 001000  000302          73020           JNZ     ASKAGN
003656' 000000  003624'
003657' 000000  003651'
003660' 001000  000041          73040   OKCHAR: LXI     H,ILLFUN        ;MAKE SURE BOMBS IF TRIES TO CALL THEM
003661' 000000  000000*
003662' 000000  003656'
003663' 001000  000021          73060           LXI     D,ATN
003664' 000000  003152'
003665' 000000  003661'
003666' 001000  000042          73080           SHLD    ATNFIX##
003667' 000000  000000*
003670' 000000  003664'
003671' 001000  000376          73100           CPI     "A"             ;DELETE ATN BUT NOT SIN, COS?
003672' 000000  000101
```



```
003673' 001000  000312          73120           JZ      HAVFNS          ;TEST
003674' 000000  003712'
003675' 000000  003667'
003676' 001000  000042          73140           SHLD    COSFIX##
003677' 000000  000000*
003700' 000000  003674'
003701' 001000  000042          73160           SHLD    TANFIX##
003702' 000000  000000*
003703' 000000  003677'
003704' 001000  000042          73180           SHLD    SINFIX##
003705' 000000  000000*
003706' 000000  003702'
003707' 001000  000021          73200           LXI     D,COS>
003710' 000000  002762'
003711' 000000  003705'
003712'                         73220   HAVFNS:                         ;TEXT ALWAYS PRECEDED BY ZERO
003712' 001000  000353          73240           XCHG                    ;PUT BOTTOM OF MEMORY IN (H,L)
003713' 001000  000066          73260           MVI     M,0             ;STORE IT
003714' 000000  000000
003715' 001000  000043          73280           INX     H               ;INCREMENT POINTER
003716' 001000  000042          73300           SHLD    TXTTAB##        ;SAVE BOTTOM OF MEMORY
003717' 000000  000000*
003720' 000000  003710'
003721' 001000  000343          73320           XTHL
003722' 001000  000021          73340           LXI     D,TSTACK
003723' 000000  004315'
003724' 000000  003717'
003725' 001000  000347          73360           COMPAR
003726' 001000  000332          73380           JC      OMERR
003727' 000000  003620*
003730' 000000  003725'
003731' 001000  000321          73400           POP     D
003732' 001000  000371          73420           SPHL
003733' 001000  000042          73440           SHLD    STKTOP
003734' 000000  003413*
003735' 000000  003727'
003736' 001000  000353          73460           XCHG
003737' 001000  000315          73480           CALL    REASON
003740' 000000  000000*
003741' 000000  003734'
003742' 001000  000173          73500           MOV     A,E             ;SUBRACT VARTAB FROM STKTOP
003743' 001000  000225          73520           SUB     L               ;LOW PART
003744' 001000  000157          73540           MOV     L,A             ;
003745' 001000  000172          73560           MOV     A,D             ;HIGH PART
003746' 001000  000234          73580           SBB     H               ;
003747' 001000  000147          73600           MOV     M,A             ;LEAVE IN [B,C]
003750' 001000  000001          73620           LXI     B,SCODE+65520
003751' 000000  177760*
003752' 000000  003740'
003753' 001000  000011          73640           DAD     B
003754' 001000  000315          73660           CALL    CRDO            ;TYPE CRLF
003755' 000000  003422*
003756' 000000  003751'
```

```
4588  003757' 001000  000315          73680           CALL    LINPRT          ;PRINT # OF BYTES FREE
4589  003760' 000000  001766'
4590  003761' 000000  003755'
4591  003762' 001000  000041          73700           LXI     H,WORDS         ;TYPE THE HEADING
4592  003763' 000000  004146'
4593  003764' 000000  003762'
4594  003765' 001000  000315          73720           CALL    STROUT          ;"ALTAIR BASIC VERSION#####"
4595  003766' 000000  003630*
4596  003767' 000000  003763'
4597  003770' 001000  000041          73740           LXI     H,STROUT
4598  003771' 000000  003766*
4599  003772' 000000  003766'
4600  003773' 001000  000042          73760           SHLD    REPINI##+1
4601  003774' 000000  000001*
4602  003775' 000000  003771'
4603  003776' 001000  000315          73780           CALL    SCRTCH          ;NOW SET UP EVERYTHING ELSE
4604  003777' 000000  000000*
4605  004000' 000000  003774'
4606                                   73800   IFN     LPTSW,<
4607                                   73820           MVI     A,4
4608                                   73840           OUT     2               ;CLEAR THE LPT
4609                                   73860           XRA     A
4610                                   73880           STA     PRTFLG##
4611                                   73900           STA     LPTPOS##>
4612  004001' 001000  000041          73920   IFE     CONSSW,<LXI     H,READY>
4613  004002' 000000  000000*
4614  004003' 000000  003777'
4615                                   73940   IFN     CONSSW,<
4616                                   73960           LXI     H,CONSOO##>
4617  004004' 001000  000042          73980           SHLD    SCODE+2
4618  004005' 000000  000002*
4619  004006' 000000  004002'
4620  004007' 001000  000351          74000           PCHL
4621
4622                                   74040   IFE     EXTFNC,<
4623                                   74060   TBLOO:  ADR(INITSA)
4624                                   74080           ADR(FNS)
4625                                   74100           ADR(SINFIX)
4626                                   74120           ADR(SIN)
4627                                   74140           ADR(FNS2)
4628                                   74160           ADR(RNDFIX)
4629                                   74180           ADR(RND)
4630                                   74200           ADR(FNS3)
4631                                   74220           ADR(SQRFIX)
4632                                   74240           ADR(SQR)
4633                                   74260   TBDASK:                         ;END OF ASK TABLE
4634                                   74280   FNS:    DC"WANT SIN"
4635                                   74300           0
4636                                   74320   FNS2:   DC"WANT RND"
4637                                   74340           0
4638                                   74360   FNS3:   DC"WANT SQR"
4639                                   74380           0>
4640                                   74400   IFN     EXTFNC,<
```



```
4641  004010' 000000  000127          74420   FNS:    DC"WANT SIN-COS-TAN-ATN"
4642  004011' 000000  000101
4643  004012' 000000  000116
4644  004013' 000000  000124
4645  004014' 000000  000040
4646  004015' 000000  000123
4647  004016' 000000  000111
4648  004017' 000000  000116
4649  004020' 000000  000055
4650  004021' 000000  000103
4651  004022' 000000  000117
4652  004023' 000000  000123
4653  004024' 000000  000055
4654  004025' 000000  000124
4655  004026' 000000  000101
4656  004027' 000000  000116
4657  004030' 000000  000055
4658  004031' 000000  000101
4659  004032' 000000  000124
4660  004033' 000000  000116
4661  004034' 000000  000316
4662  004034' 000000  000000          74440           0>
4663  004035' 000000  000015          74460   AUTTXT: ACRLF
4664  004036' 000000  000012
4665  004037' 000000  000012          74480           "D10
4666  004040' 000000  000127          74500           DC"WRITTEN BY BILL GATES & PAUL ALLEN & MONTE DAVIDOFF,"
4667  004041' 000000  000122
4668  004042' 000000  000111
4669  004043' 000000  000124
4670  004044' 000000  000124
4671  004045' 000000  000105
4672  004046' 000000  000116
4673  004047' 000000  000040
4674  004050' 000000  000102
4675  004051' 000000  000131
4676  004052' 000000  000040
4677  004053' 000000  000102
4678  004054' 000000  000111
4679  004055' 000000  000114
4680  004056' 000000  000114
4681  004057' 000000  000040
4682  004060' 000000  000107
4683  004061' 000000  000101
4684  004062' 000000  000124
4685  004063' 000000  000105
4686  004064' 000000  000123
4687  004065' 000000  000040
4688  004066' 000000  000046
4689  004067' 000000  000040
4690  004070' 000000  000120
4691  004071' 000000  000101
4692  004072' 000000  000125
4693  004073' 000000  000114
```

```
4694  004074' 000000  000040
4695  004075' 000000  000101
4696  004076' 000000  000114
4697  004077' 000000  000114
4698  004100' 000000  000105
4699  004101' 000000  000116
4700  004102' 000000  000040
4701  004103' 000000  000046
4702  004104' 000000  000040
4703  004105' 000000  000115
4704  004106' 000000  000117
4705  004107' 000000  000116
4706  004110' 000000  000124
4707  004111' 000000  000105
4708  004112' 000000  000040
4709  004113' 000000  000104
4710  004114' 000000  000101
4711  004115' 000000  000126
4712  004116' 000000  000111
4713  004117' 000000  000104
4714  004120' 000000  000117
4715  004121' 000000  000105
4716  004122' 000000  000106
4717  004123' 000000  000056
4718  004123' 000000  000256
4719  004124' 000000  000015          74520           ACRLF
4720  004125' 000000  000012
4721  004126' 000000  000000          74540           0
4722
4723  004127' 000000  000124          74580   TTYWID: DC"TERMINAL WIDTH"
4724  004130' 000000  000105
4725  004131' 000000  000122
4726  004132' 000000  000115
4727  004133' 000000  000111
4728  004134' 000000  000116
4729  004135' 000000  000101
4730  004136' 000000  000114
4731  004137' 000000  000040
4732  004140' 000000  000127
4733  004141' 000000  000111
4734  004142' 000000  000104
4735  004143' 000000  000124
4736  004144' 000000  000110
4737  004144' 000000  000310
4738  004145' 000000  000000          74600           0
4739  004146' 000000  000040          74620   WORDS:  DC" BYTES FREE"
4740  004147' 000000  000102
4741  004150' 000000  000131
4742  004151' 000000  000124
4743  004152' 000000  000105
4744  004153' 000000  000123
4745  004154' 000000  000040
4746  004155' 000000  000106
```



```
4747  004156' 000000  000122
4748  004157' 000000  000105
4749  004160' 000000  000105
4750  004160' 000000  000305
4751  004161' 000000  000015          74640           ACRLF
4752  004162' 000000  000012
4753  004163' 000000  000015          74660           ACRLF
4754  004164' 000000  000012
4755  004165' 000000  000101          74680           DC"ALTAIR BASIC VERSION 3.0"
4756  004166' 000000  000114
4757  004167' 000000  000124
4758  004170' 000000  000101
4759  004171' 000000  000111
4760  004172' 000000  000122
4761  004173' 000000  000040
4762  004174' 000000  000102
4763  004175' 000000  000101
4764  004176' 000000  000123
4765  004177' 000000  000111
4766  004200' 000000  000103
4767  004201' 000000  000040
4768  004202' 000000  000126
4769  004203' 000000  000105
4770  004204' 000000  000122
4771  004205' 000000  000123
4772  004206' 000000  000111
4773  004207' 000000  000117
4774  004210' 000000  000116
4775  004211' 000000  000040
4776  004212' 000000  000063
4777  004213' 000000  000056
4778  004214' 000000  000060
4779  004214' 000000  000260
4780  004215' 000000  000015          74700           ACRLF
4781  004216' 000000  000012
4782                                   74720   IFE     LENGTH,<DC"[FOUR-K VERSION]">
4783  004217' 000000  000133          74740   IFE     LENGTH-1,<DC"[EIGHT-K VERSION]">
4784  004220' 000000  000105
4785  004221' 000000  000111
4786  004222' 000000  000107
4787  004223' 000000  000110
4788  004224' 000000  000124
4789  004225' 000000  000055
4790  004226' 000000  000113
4791  004227' 000000  000040
4792  004230' 000000  000126
4793  004231' 000000  000105
4794  004232' 000000  000122
4795  004233' 000000  000123
4796  004234' 000000  000111
4797  004235' 000000  000117
4798  004236' 000000  000116
4799  004237' 000000  000135
```

```
MATHPK FOR BASIC MCS 8080  GATES/ALLEN/DAVIDOFF MACRO 47(113) 06:09 27-AUG-75 PAGE 19-11
F4       MAC     23-AUG-64 06:08          SYSTEM INITIALIZATION CODE

  4800  004237'  000000   000335
  4801                                74760   IFE     LENGTH=2,<DC"[81G VERSION]">
  4802  004240'  000000   000015     74780           ACRLF
  4803  004241'  000000   000012
  4804  004242'  000000   000000     74800           0
  4805
  4806  004243'  000000   000115     74840   MEMORY: DC"MEMORY SIZE"
  4807  004244'  000000   000105
  4808  004245'  000000   000115
  4809  004246'  000000   000117
  4810  004247'  000000   000122
  4811  004250'  000000   000131
  4812  004251'  000000   000040
  4813  004252'  000000   000123
  4814  004253'  000000   000111
  4815  004254'  000000   000132
  4816  004255'  000000   000105
  4817  004255'  000000   000305
  4818  004256'  000000   000000     74860           0
  4819  004257'                     74880   LASTWR::                        ;LAST WORD OF SYSTEM CODE+1
  4820  004257'                     74900           BLOCK   "D30            ;SPACE FOR TEMP STACK
  4821  004313'                     74920   TSTACK::
  4822                              74940   IFE     LENGTH,<
  4823                              74960           BLOCK   "D13000>
  4824                              74980   IFN     LENGTH,<
  4825  004313'                     75000           BLOCK   "D10000>
  4826                  004005'     75020           ,C2==:,P
  4827                              75040   END

NO ERRORS DETECTED

PROGRAM BREAK IS 027735

9K CORE USED
```

```
MATHPK FOR BASIC MCS 8080  GATES/ALLEN/DAVIDOFF MACRO 47(113) 06:09 27-AUG-75 PAGE 20
F4       MAC     23-AUG-64 06:08          SYMBOL TABLE

A                 000007          DMULT           001476' INT     FMULT3          000632'
ABS               001173' INT     DMULT1          001506'         FMULT4          000571'
AFFF              000165          DMULT2          001524'         FMULT5          000607'
ASKAGN            003624'         DSKFUN          000000  SPD     FMULTA          000605'
ATN               003152' INT     DV0ERR          000657' EXT     FMULTB          000600'
ATN2              003205'         E               000003          FMULTT          000515' INT
ATNCON            003217'         ERROR           000272' EXT     FNS             004010'
ATNFIX            003667' EXT     ERROV           000270' EXT     FUNE            000400' INT
AUTTXT            004035'         EXP             002452' INT     FOUT            002001' INT
B                 000000          EXPCON          002552'         FOUT1           002015'
BSERR             001522' EXT     EXTFNC          000001  SPD     FOUT10          002151'
BUF               003536' EXT     FAC             003162' EXT     FOUT11          002216'
C                 000001          FACLO           002430' EXT     FOUT12          002232'
CASSW             000001  SPD     FADD            000025' INT     FOUT14          002252'
CNLCA1            000000  EXT     FADD1           000057'         FOUT15          002254'
CNLCA2            000000  EXT     FADD3           000125'         FOUT17          002271'
CNLCA3            000000  EXT     FADDA           000274'         FOUT19          002266'
CNLCAA            000000  EXT     FADDH           000000'         FOUT3           002234'
CNTHFL            003417' EXT     FADDS           000003' INT     FOUT5           002271'
CONSSW            000000  SPD     FADDT           000023' INT     FOUT6           002124'
CONTRW            000001  SPD     FADFLT          000143'         FOUT8           002132'
COS               002762' INT     FBUFFR          002002' EXT     FOUT9           002060'
COSFIX            003677' EXT     FCERR           000423' EXT     FOUT8L          002316'
CRDO              003755' EXT     FCOMP           001317' INT     FOUTCB          002274'
CURLIN            003404' EXT     FCOMPE          001347'         FPWR            002360' INT
D                 000002          FCOMPU          001344'         FPWR1           002422'
DCRART            001720'         FCOMPS          001141'         FPWR2           002440'
DFLENT            003604'         FDIV            000655' INT     FPWRT           002356' INT
DIV10             000637'         FDIV1           000720'         FR4             003074'
                                  FDIV2           000755'         FRETOP          003614' EXT
                                  FDIVA           000733'         FSUB            000017' INT
                                  FDIVB           000727'         FSUB3           000011'
                                  FDIVC           000723'         FSUBT           000015' INT
                                  FDIVG           000736'         FUNIO           145001  SPD
                                  FDIVT           000653' INT     H               000004
                                  FHALF           002312'         HAVFNS          003712'
                                  FIN             001531' INT     ICOMPS          001142'
                                  FIN1            001545'         ILLFUN          003661' EXT
                                  FINC            001555'         INIT            003304' INT
                                  FINDIG          001702'         INITAT          003276'
                                  FINDP           001635'         INITSA          003264'
                                  FINE            001641'         INPRT           001756' INT
                                  FINE2           001644'         INRART          001145' INT
                                  FINE3           001660'         INT             001445' INT
                                  FINEC           001621'         INTXT           001760' EXT
                                  FINEDG          001742'         INXHRT          001252' INT
                                  FINLOG          001731'
                                  FINMLT          001673'
                                  FINMUL          001672'
                                  FLOAT           001156' INT
                                  FLOATR          001155' INT
                                  FMULT           000517' INT
                                  FMULT2          000557'
```

```
L                000005          OKCHAR          003660'          SHIFTR          000334'
LASTWR           004257' INT     OMERR           003727' EXT      SHRADD          000362'
LENGTH           000001  SPD     OVERR           000267' INT      SIGN            001322' EXT
LINGET           003541' EXT     PI2             003070'          SIGNC           001135' INT
LINPRT           001766' INT     PLUSTK          001615' EXT      SIGNS           001143' INT
LINPT1           003557' EXT     POLY            002632'          SIN             002770' INT
LINPT2           003562' EXT     POLY1           002643'          SIN1            003044'
LINPT3           003565' EXT     POLYX           002613'          SINCON          003100'
LINPT4           003602' EXT     POPHRT          000630'          SINFIX          003705' EXT
LOG              000421' INT     POPPRT          001474'          SNERR           003511' EXT
LOGCN2           000404'         PSHNEG          002340'          SQR             002345' INT
LOOPMM           003460'         PSW             000006  SPD      STKTOP          003734' EXT
LPTSW            000000  SPD     PUSHF           001205' INT      STRING          000001  SPD
M                000006  SPD     QINLIN          003633' EXT      STROUI          001776' EXT
MEMORY           004245'         QINT            001372' INT      STROUT          003771' EXT
MEMSIZ           003611' EXT     QINTA           001436'          TAN             003125' INT
MINUTK           001603' EXT     READY           004002' EXT      TANFIX          003702' EXT
MLOVEX           001073'         REALIO          000001  SPD      TEMPPT          003430' EXT
MORCPS           003567'         REASON          003740' EXT      TEMPST          003425' EXT
MOVE             001257' INT     REPIN1          000000  EXT      TSTACK          004315' INT
MOVE1            001261'         RND             002673' INT      TTYW            003517'
MOVFM            001222' INT     RND1            002725'          TTYWID          004127'
MOVFR            001225' INT     RNDX            002752'          TXTTAB          003717' EXT
MOVMF            001254' INT     ROUND           000233'          UNPACK          001272'
MOVRF            001240' INT     ROUNDA          000255'          USEDE9          003501'
MOVRM            001243' INT     ROUNDB          000234'          USEDEF          003515'
MUL10            001146'         SCRTCH          003777' EXT      WORDS           004146'
MULDIM           000001  SPD     SGN             001147' INT      ZERO            000173' INT
MULDIV           001035'         SHFTR1          000336'          ZERO8           000174'
MULDV1           001076'         SHFTR2          000353'          SCODE           002543' EXT
MULDV2           001077'         SHFTR3          000356'          .C2             004005' SIN
MULLN2           000586'                                          .P              004005' SPD
NEG              001175' INT
NEGR             000310'
NORM1            000151'
NORM2            000200'
NORM3            000210'
NORMAL           000146'
NOTPIO           003400'
```

```
A                212     213     218     231     251     299     301     302     304     306     308     310     312     344
                 345     346     347     355     356     372     381     383     384     385     386     390     393     401
                 412     416     423     427     472     474     476     478     480     482     492     494     495     496
                 498     499     501     502     504     505     507     537     538     544     546     547     549     550
                 552     553     555     613     684     709     712     739     740     761     762     764     765     767
                 768     770     772     784     827     832     837     866     867     868     869     875     878     879
                 882     883     886     887     904     905     906     915     917     918     920     921     923     925
                 927     935     966     967     971     977     980     986     993     1000    1029    1030    1037    1068
                1070    1096    1126    1129    1247    1265    1269    1274    1275    1279    1333    1334    1343    1349
                1359    1363    1367    1371    1530    1531    1532    1533    1534    1555    1559    1565    1580    1582
                1603    1614    1622    1624    1695    1698    1714    2342    2343    2344    2346    2402    2404    2416
                2429    2559    2579    2580    2582    2686    2694    2716    2771    2801    2812    2838    2839    2840
                2843    2866    2868    2870    2872    2874    2876    2924    2933    2949    3637    3638    3644    3665
                3839    3848    3925    3927    4023    4318    4340    4356    4358    4363    4364    4378    4398    4408
                4409    4413    4438    4575    4577    4578    4580
ABS               89    1111#
AFFF            4642#   4643#   4644#   4645#   4646#   4647#   4648#   4649#   4650#   4651#   4652#   4653#   4654#   4655#
                4656#   4657#   4658#   4659#   4660#   4661#   4661    4667#   4668#   4669#   4670#   4671#   4672#   4673#
                4674#   4675#   4676#   4677#   4678#   4679#   4680#   4681#   4682#   4683#   4684#   4685#   4686#   4687#
                4688#   4689#   4690#   4691#   4692#   4693#   4694#   4695#   4696#   4697#   4698#   4699#   4700#   4701#
                4702#   4703#   4704#   4705#   4706#   4707#   4708#   4709#   4710#   4711#   4712#   4713#   4714#   4715#
                4716#   4717#   4718#   4710    4724#   4725#   4726#   4727#   4728#   4729#   4730#   4731#   4732#   4733#
                4734#   4735#   4736#   4737#   4737    4740#   4741#   4742#   4743#   4744#   4745#   4746#   4747#   4748#
                4749#   4750#   4750    4756#   4757#   4758#   4759#   4760#   4761#   4762#   4763#   4764#   4765#   4766#
                4767#   4768#   4769#   4770#   4771#   4772#   4773#   4774#   4775#   4776#   4777#   4778#   4779#   4779
                4784#   4785#   4786#   4787#   4788#   4789#   4790#   4791#   4792#   4793#   4794#   4795#   4796#   4797#
                4798#   4799#   4800#   4800    4807#   4808#   4809#   4810#   4811#   4812#   4813#   4814#   4815#   4816#
                4817#   4817
ASKAGN          4496#   4522
ATN               92    4140#   4528
ATN2            4155    4167#
ATNCON          4168    4179#
ATNFIX          4531#   4531
AUTTXT          4231    4663#
B                198     210     212     226     240     300     301     342     345     356     379     390     392     412
                 420     497     498     516     527     553     555     620     622     624     628     630     674     692
                 695     696     768     770     781     806     864     874     883     886     902     925     927     966
                 976     977     979     980    1029    1037    1083    1097    1098    1199    1225    1241    1250    1333
                1359    1530    1561    1570    1695    1710    2342    2417    2580    2582    2583    2602    2717    2823
                2839    2850    2856    2863    2865    2891    2893    2898    2935    2937    2947    3618    3637    3643
                3654    3658    3682    3686    3745    3763    3846    3851    3931    3986    3997    4120    4155    4581
                4584
BSERR           1702#   1702    1712
BUF             4225#   4373    4405
C                311     312     346     352     384     386     426     427     454     456     481     482     506     507
                 529     530     544     546     684     740     761     783     784     864     867     904     921     923
                 935    1096    1200    1223    1275    1279    1343    1349    1355    1363    1531    1585    1624    1696
                2346    2405    2408    2581    2828    2874    2876    2894    2951    3644    3764    3926    3927    4158
                4159
CASSW             6#
CNLCA1          4269#   4269    4301
CNLCA2          4251#   4251    4283
CNLCA3          4256#   4256    4288
```

```
CNLCA4   4262#   4262    4294
CNTWFL   4320#   4320
CONSSW      9#   4612    4615
CONTRW     11#   4317
COS        92   3958#   4130    4548
COSFIX  4539#   4539
CRUD    4223#   4323    4586
CURLIN  4308#   4308
D         199    211     241     307     308     352     353     381     383     452     477     478     503     504
          528    529     547     549     623     629     675     695     739     746     747     750     751     772
          782    783     807     868     918     920     936    1085    1221    1232    1246    1248    1367    1532
         1579   1580    2343    2370    2380    2398    2436    2578    2579    2585    2603    2847    2858    2870
         2872   3619    3642    3653    3659    3681    3687    3746    3760    3764    3815    3818    3847    3850
         3987   3998    4158    4408    4445    4455    4508    4527    4547    4559    4566    4578
DCRART   2559#
DFLENT   4400   4443#
DIV10      79#   2426    2798
DMULT      94   1692#
DMULT1   1700#   1716
DMULT2   1700   1714#
DSKFUN      8#
DV0ERR   106#    815
E         303    304     343     391     450     463     473     474     500     501     527     528     550     552
          696    721     771     781     782     869     915     917     937    1219    1371    1533    1581    1622
         2394   2403    2404    2410    2686    2689    2694    2866    2868    3925    3926    4159    4413    4439
         4575
ERROR    106#    466
ERROV    106#    464
EXP        9E   3635    3704#
EXPCON   3755   3770#
EXTFNC     30#     91     191     580     670     996    1608    1613    1621    1626    1628    2592    2597    2667
         3554   3598    3694#   3805    3904    3906    3955    4002    4007    4028    4033    4046    4069    4077
         4087   4111    4137    4461    4495    4622    4640
FAC      106#    216     374     398     414     490     611     943     973    1042    1063    1094    1124    1175
         1202   1263    1346    1603    3669    3717    4148
FACLO    106#    270     707    1170    1197    1211    1233    1610    3673
FADD       86    186     212#    626     653    1039    3864    3920
FADD1     220    242#
FADD3     273    299#
FADDA     276    472#   2883
FADDM     179#   2810    4020
FADDS      86    182#   3739    3962    4040
FADDT      98    210#   2677
FADFLT    315#   1105    1630
FBUFFR   106#   2743
FCERR    106#    606
FCOMP      89   1333#   2783    2963    3662
FCOMP2   1352   1358#
FCOMPD   1354#
FCOMPS   1066#   1339
FDIV       86    632     813#   3989    4161
FDIV1    873#    939     948
FDIV2    893    902#
```

```
FDIVA    829    884#
FDIVB    834    880#
FDIVC    839    876#
FDIVG    871    887#    896     929     933
FDIVT     98    806#   4133
FMALF    180   2972#   3613
FIN       86   2322#
FIN1    2329   2334    2337#
FINC    2348#   2410    2605
FINDIG  2350   2576#
FINDP   2355   2408#
FINE    2360   2400    2415#
FINE2   2418#   2431
FINE3   2422   2430#
FINEC   2367   2394#   2696
FINEDG  2396   2686#
FINLOG   657   2599    2669#
FINMLT  2546#   2791
FINMUL  2419   2545#
FLOAT      86   1083#   2673
FLOATR     86   1093#   2720
FMULT      88    676#   3689    3714    3766    3824    3855    3911
FMULT2    702    709#
FMULT3    714    781#
FMULT4    735#    774
FMULT5    742    760#
FMULTA    686    752#
FMULTB    690    747#
FMULTT     96    674#   3816
FNS      4497   4641#
FONE       92    586#    635     3736
FOUT       86   2742#
FOUT1   2751   2755#
FOUT10  2865#   2880
FOUT11  2903#   2908
FOUT12  2900   2916#
FOUT14  2928   2935#
FOUT15  2937#   2941
FOUT17  2918   2951#
FOUT19  2759   2948#
FOUT3   2776#   2795
FOUT5   2787   2809#
FOUT6   2831   2836    2843#
FOUT8   2850#   2896
FOUT9   2797#   2967
FOUTBL  2848   2977#
FOUTC8  2774   2804    2956#
FPWR       92   3633#
FPWR1   3651   3667#
FPWR2   3681#
FPWRT     100   3618#
FR4      4009   4035    4082#
FRETOP   4453#   4453
```

```
RNC          246    1612
RNZ          451     453     455     459    1045    1069    1361    1365    1369    1373    1583
RP          1116
RST          604     676     813     997    1077    1115    1342    2348    2363    2394    2747    3633    3886    4014
            4022    4140    4342    4397    4505    4562
RZ           214     405     540     677    1031    1344    1535    1697    2376    2379    2384    2390    2545    3849
            3903
SBB          303     307     311     500     503     506    1068    2871    2875    4579
SBI          880     884     889
SHLD         689    1196    1201    3668    3672    4250    4255    4261    4268    4282    4287    4293    4300    4307
            4314    4329    4449    4452    4530    4538    4541    4544    4555    4568    4600    4617
SPHL        4313    4567
STA          373     685     828     833     838     870     895     932    4319    4419    4423    4427    4440
STC          900    4015
SUB          226     300     497     620    1372    1561    2403    2417    2867    4576
SUI          357     521     876    2590    2692    2938    4430
SYNCHK       39#
XCHG         232     236     688     720     777     865    1168    1179    1195    1204    1704    1706    2441    2715
            2889    4125    4382    4551    4571
XRA          299     344     372     426     495     538     866     975     979    1271    1281    1348    1355    1554
            2402    2716    2771    4318
XRI         1127
XTHL        1172    1177    2369    3603    4558
```

```
SPOOLER RUNTIME 14 SECONDS, 44 KCS, 462 DISK READS, 3 DISK WRITES, 118 PAGES
 **END** USER MITS  [6000,6000] JOB  F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**
```

END USER MITS [6000,6000] JOB F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

END USER MITS [6000,6000] JOB F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

END USER MITS [6000,6000] JOB F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

END USER MITS [6000,6000] JOB F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

END USER MITS [6000,6000] JOB F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

END USER MITS [6000,6000] JOB F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

END USER MITS [6000,6000] JOB F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

END USER MITS [6000,6000] JOB F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

END USER MITS [6000,6000] JOB F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

END USER MITS [6000,6000] JOB F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

END USER MITS [6000,6000] JOB F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

END USER MITS [6000,6000] JOB F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

END USER MITS [6000,6000] JOB F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

END USER MITS [6000,6000] JOB F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000011111111111111111111111111111111