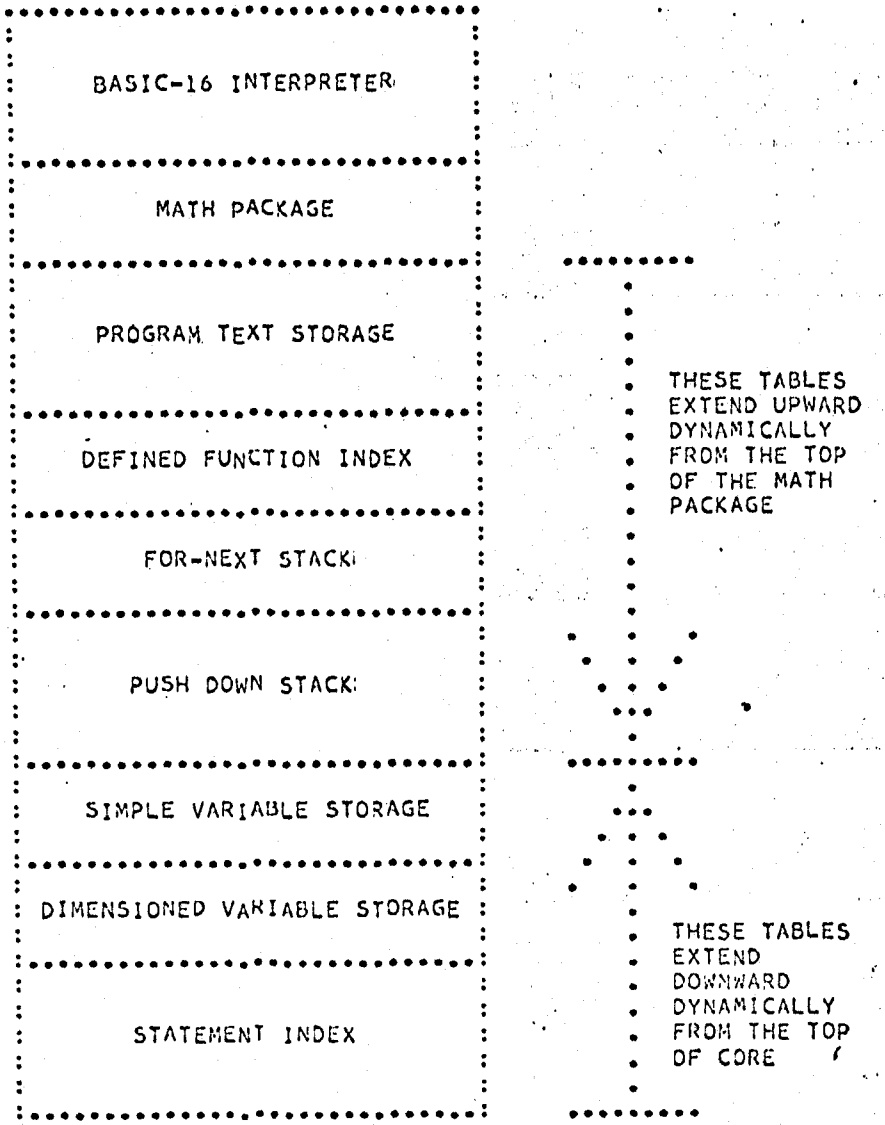
```
0001                *          BASIC-16
0002                *
0003                *
0004                *
0005                *
0006                *
0007                *
0008                *          BBBBBBB   AAAAAA   SSSSSS   IIIIII   CCCCCC
0009                *          BBBBBBBB AAAAAAAA SSSSSSSS IIIIII  CCCCCCCC
0010                *          BB    BB AA    AA SS    SS    II    CC    CC
0011                *          BBBBBBB  AA    AA SSS         II    CC
0012                *          BBBBBBB  AAAAAAAA  SSSSS      II    CC
0013                *          BB    BB AAAAAAAA     SSSS    II    CC
0014                *          BB    BB AA    AA SS    SS    II    CC    CC
0015                *          BBBBBBBB AA    AA SSSSSSSS IIIIII  CCCCCCCC
0016                *          BBBBBBB  AA    AA  SSSSSS  IIIIII   CCCCCC
0017                *
0018                *
0019                *
0020                *
0021                *
0022                     EJCT
```

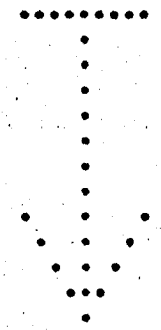
MEMORY ORGANIZATION AND TABLE FORMATS

GENERAL ORGANIZATION

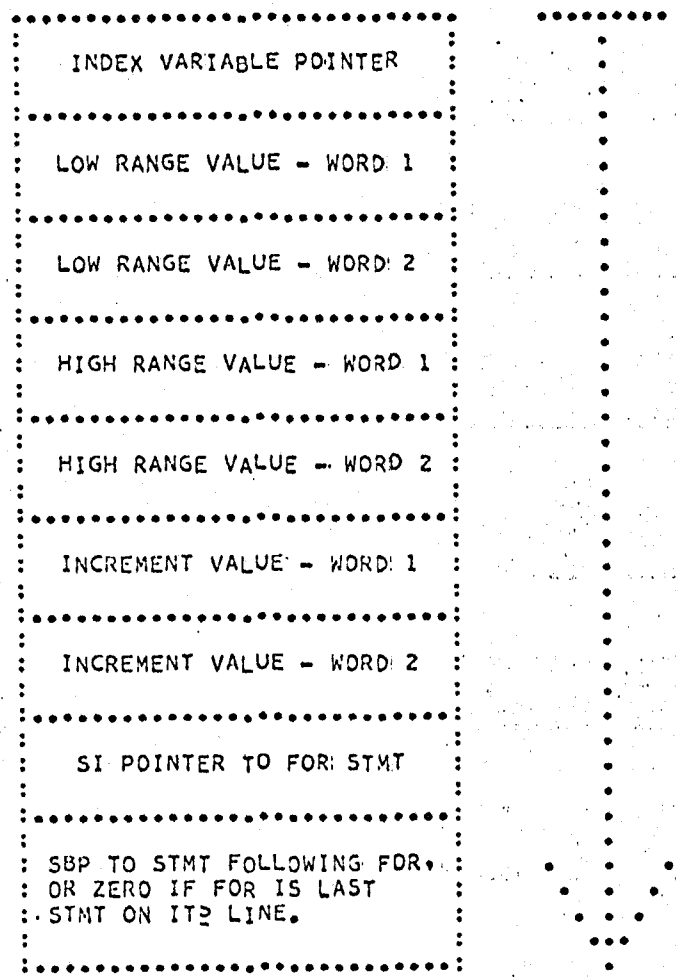


```
0073  *        PROGRAM TEXT STORAGE FORMAT
0074  *
0075  *
0076  *
0077  *    PURPOSE:  TO STORE THE USERS SOURCE PROGRAM.
0078  *
0079  *    ENTRY LENGTH:  VARIABLE, ONE WORD MINIMUM.
0080  *
0081  *    ENTRY FORMAT:  EACH ENTRY IS A BYTE STRING CORRESPONDING
0082  *                   TO THE USERS SOURCE STATEMENT, WITH THE
0083  *                   LINE NUMBER REMOVED.  IF A STATEMENT HAS
0084  *                   AN ODD NUMBER OF BYTES, AN EXTRA BYTE WILL
0085  *                   BE ADDED TO IT TO FILL UP THE LAST WORD
0086  *                   THAT THE STRING OCCUPIES.
0087  *
0088  *    POINTERS:
0089  *
0090  *                PTB - ADDRESS OF LOWEST WORD IN THE TABLE,
0091  *                      INITIALIZED TO THE ADDRESS OF THE
0092  *                      FIRST WORD ABOVE THE MATH PACKAGE.
0093  *                      PTB IS NEVER ALTERED.
0094  *
0095  *                PTH - ADDRESS OF THE LAST WORD IN THE TABLE
0096  *                      +1, IS INITIALLY SET TO THE SAME
0097  *                      VALUE AS PTB.
0098  *
0099  *
0100  *
0101        EJCT
```

```
0102  *        DEFINED FUNCTION INDEX
0103  *
0104  *
0105  *   PURPOSE:  TO SAVE THE ADDRESS OF THE PROTOTYPE EXPRESSION
0106  *             AND THE DUMMY VARIABLE NAME FOR THE DEFINED
0107  *             EXPRESSION.
0108  *
0109  *   ENTRY LENGTH:  THREE WORDS
0110  *
0111  *   ENTRY FORMAT:
0112  *
0113  *
0114  *        .....................................
0115  *        :  00000000  :  FUNCTION NAME       :
0116  *        :.....................:.............:
0117  *        :  DUMMY VARIABLE NAME              :
0118  *        :.....................:.............:
0119  *        :  POINTER TO FIRST BYTE OF EXPRESSION :
0120  *        :.....................:.............:
0121  *
0122  *   POINTERS:
0123  *
0124  *        DFB - POINTER TO FIRST WORD OF FIRST
0125  *              ENTRY IN THE INDEX.  IF INDEX
0126  *              IS EMPTY, DFB=0.
0127  *
0128  *        DFT - POINTER TO LAST WORD OF LAST
0129  *              ENTRY IN THE INDEX.  IF INDEX
0130  *              IS EMPTY, DFT=0.
0131  *
0132  *
0133  *
0134  *
0135  *
0136  *
0137  *
0138  *
0139  *
0140  *
0141      EJCT
```



```
0142                    *       FOR-NEXT STACK FORMAT
0143                    *
0144                    *
0145                    *   PURPOSE:  TO SAVE LOOP CONTROLI INFORMATION FOR
0146                    *             FOR-NEXT LOOPS.
0147                    *
0148                    *   ENTRY LENGTH:  NINE WORDS
0149                    *
0150                    *   ENTRY FORMAT:
0151                    *
0152                    *
0153                        EJCT
```



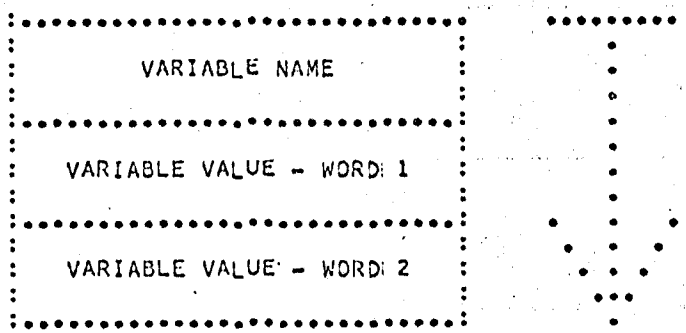
NOTE: THE INDEX VARIABLE POINTER IS THE DIFFERENCE BETWEEN THE TOP OF THE SIMPLE VARIABLE STORAGE TABLE AND THE FIRST WORD CONTAINING THE VALUE OF THE INDEX VARIABLE.

EJCT

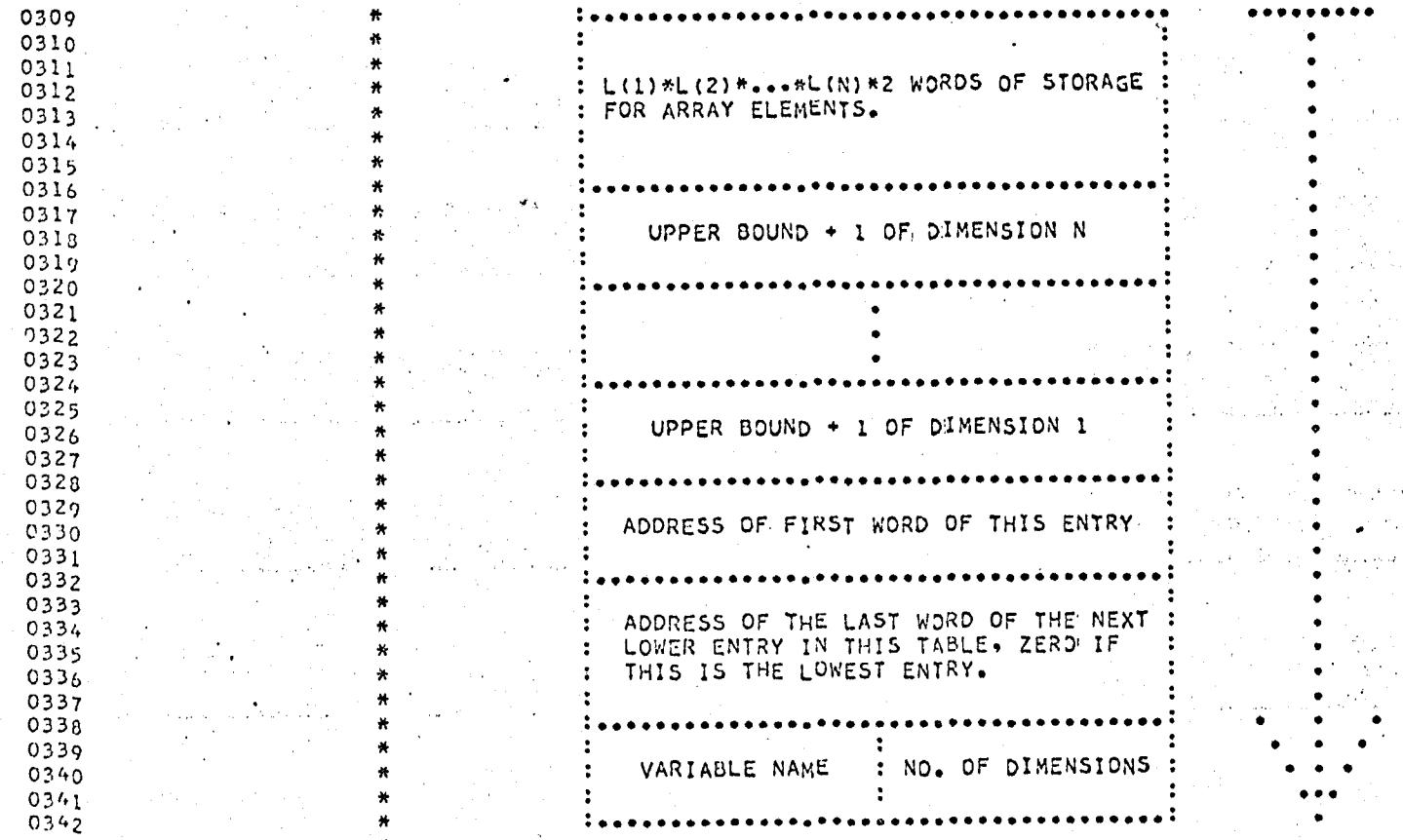
```
0204                *   POINTERS:
0205                *
0206                *           FNB - ADDRESS OF THE FIRST WORD OF THE
0207                *                 FIRST ENTRY IN THE FOR-NEXT
0208                *                 TABLE.  IF THE TABLE IS EMPTY,
0209                *                 FNB=0.
0210                *
0211                *           FNT - ADDRESS OF THE LAST WORD OF THE
0212                *                 LAST ENTRY IN THE FOR-NEXT
0213                *                 TABLE.  IF THE TABLE IS EMPTY,
0214                *                 FNT=0.
0215                *
0216                *
0217                *
0218                    EJCT
```

```
0219                *          PUSH DOWN STACK FORMAT
0220                *
0221                *
0222                *
0223                *     PURPOSE:  TO HOLD RETURN ADDRESSES AND INTERMEADIATE
0224                *               RESULTS FOR RE-ENTRANT SUBROUTINES.
0225                *
0226                *     ENTRY LENGTH:  ONE WORD
0227                *
0228                *     ENTRY FORMAT:  THERE IS NO FOXED FORMAT FOR PUSH
0229                *                    DOWN STACK ENTRIES, OTHER TAHN THE
0230                *                    FACT THAT EACH ENTRY OCCUPIES ONE
0231                *                    WORD.
0232                *
0233                *     POINTERS:
0234                *
0235                *                 PDLP - POINTS TO THE FREE WORD
0236                *                        AT THE TOP OF THE STACK.  IT
0237                *                        IS INITIALIZED TO THE ADDRESS
0238                *                        OF THE FIRST FREE WORD ABOVE
0239                *                        THE TABLES BELOW THE PUSH
0240                *                        DOWN STACK.
0241                *
0242                *
0243                *
0244                      EJCT
```

```
0245  *
0246  *        SIMPLE VARIABLE STORAGE FORMAT
0247  *
0248  *  PURPOSE:  TO HOLD THE VALUES OF SIMPLE VARIABLES
0249  *            USED IN A BASIC PROGRAM.
0250  *
0251  *  ENTRY LENGTH:  THREE WORDS
0252  *
0253  *  ENTRY FORMAT:
0254  *
0255  *
0256  *
0257  *
0258  *
0259  *
0260  *
0261  *
0262  *
0263  *
0264  *
0265  *
0266  *
0267  *
0268  *
0269  *
0270  *
0271  *  POINTERS:
0272  *
0273  *            SVB - POINTS TO THE FIRST WORD OF THE
0274  *                  FIRST ENTRY IN THE TABLE.  IF
0275  *                  THE TABLE IS EMPTY, SVB=0.
0276  *
0277  *            SVT - POINTS TO THE LAST WORD OF THE
0278  *                  LAST ENTRY IN THE TABLE.  IF
0279  *                  THE TABLE IS EMPTY, SVT=0.
0280  *
0281  *
0282  *
0283      EJCT
```



```
0284                *          DIMENSIONED VARIABLE STORAGE FORMAT
0285                *
0286                *
0287                *     PERPOSE:  TO HOLD SUBSCRIPT BOUNDS AND TO PROVIDE
0288                *               STORAGE FOR ARRAY ELEMENTS OF DIMENSIONED
0289                *               VARIABLES.
0290                *
0291                *     ENTRY LENGTH:
0292                *           THE ENTRY LENGTH FOR A GIVEN ARRAY MAY BE
0293                *           CALCULATED BY THE FOLLOWING FORMULAR:
0294                *
0295                *               GIVEN:
0296                *
0297                *                    N = NUMBER OF DIMENSIONS
0298                *                    L(X) = UPPER BOUND OF DIMENSION X + 1
0299                *
0300                *                 THEN:
0301                *
0302                *                    ENTRY LENGTH = L(1)*L(2)*...*L(N)*2+N+3
0303                *
0304                *
0305                *     ENTRY FORMAT:
0306                *
0307                *
0308                      EJCT
```



POINTERS:

DVB - ADDRESS OF THE FIRST WORD OF THE FIRST ENTRY IN THE TABLE. IF THE TABLE IS EMPTY, DVB=0.

DVT - ADDRESS OF THE LAST WORD OF THE LAST ENTRY IN THE TABLE. IF THE TABLE IS EMPTY, DVT=0.

EJCT

```
0358  *        STATEMENT INDEX FORMAT
0359  *
0360  *
0361  *    PURPOSE:  TO MAINTAIN A SEQUENTIAL LIST OF ALL
0362  *              STATEMENT NUMBERS IN A BASIC PROGRAM
0363  *              AND POINTERS TO THE CORRESPONDING
0364  *              STATEMENT TEXT IN THE PROGRAM TEXT
0365  *              TABLE.
0366  *
0367  *    ENTRY LENGTH:  TWO WORDS
0368  *
0369  *    ENTRY FORMAT:
0370  *
0371  *
0372  *              .............................      .........
0373  *              :                           :      :       .
0374  *              :  BINARY REPRESENTATION OF :      :       .
0375  *              :  STATEMENT NUMBER         :      :       .
0376  *              :                           :      :       .
0377  *              :...........................:      :       .
0378  *              :                           :      :       .
0379  *              :  POINTER TO FIRST BYTE OF :      :       .
0380  *              :  THIS STATEMENT IN THE    :      :     .   .   .
0381  *              :  PROGRAM TEXT TABLE.      :      :       .   .
0382  *              :                           :      :       ...
0383  *              :...........................:      :       .
0384  *
0385  *
0386  *    POINTERS:
0387  *
0388  *              SIB - POINTER TO THE FIRST WORD OF THE FIRST
0389  *                    ENTRY IN THE INDEX.  IF THE INDEX IS
0390  *                    EMPTY, SIB=SIT+1.
0391  *
0392  *              SIT - POINTS TO THE LAST WORD OF THE LAST
0393  *                    ENTRY IN THE INDEX.  SIT IS INITIALLY
0394  *                    SET TO THE ADDRESS OF THE LAST AVAILABLE
0395  *                    MEMORY LOCATION, AND IS NEVER ALTERED.
0396  *
0397  *              SIP - POINTS TO THE FIRST WORD OF THE
0398  *                    INDEX ENTRY FOR THE STATEMENT
0399  *                    CURRENTLY BEING EXECUTED.  IF,
0400  *                    HOWEVER, THE CURRENT STATEMENT
0401  *                    IS BEING EXECUTED IN THE IMMEADIATE
0402  *                    MODE, SIP=0.
0403  *
0404  *
0405  *
0406        EJCT
```

```
0407  *        RETURN STACK FORMAT
0408  *
0409  *
0410  *    PURPOSE:  TO HOLD RETURN INFORMATION FOR
0411  *              GOSUB-RETURN SEQUENCES.
0412  *
0413  *    TABLE SIZE:  FIXED LENGTH, 16 WORDS
0414  *
0415  *    ENTRY LENGTH:  TWO WORDS
0416  *
0417  *    ENTRY FORMAT:
0418  *
0419  *
0420  *          .................................
0421  *          :                               :
0422  *          : SI POINTER TO GOSUB STATEMENT :
0423  *          :                               :
0424  *          :...............................:
0425  *          :                               :
0426  *          :  SBP TO STATEMENT FOLLOWING   :
0427  *          :  GOSUB, OR ZERO IF GOSUB IS   :
0428  *          :  LAST STATEMENT ON ITS LINE   :
0429  *          :                               :
0430  *          :...............................:
0431  *
0432  *
0433  *    POINTERS:
0434  *
0435  *              RTB - POINTER TO THE FIRST WORD OF
0436  *                    THE FIRST AVAILABLE ENTRY,
0437  *                    NEVER IS ALTERED.
0438  *
0439  *              RTM - POINTER TO THE LAST WORD OF
0440  *                    THE LAST AVAILABLE ENTRY,
0441  *                    NEVER IS ALTERED.
0442  *
0443  *              RTP - POINTER TO THE FIRST WORD OF
0444  *                    THE NEXT FREE ENTRY.  IF THE
0445  *                    TABLE IS EMPTY, RTP=RTB.  IF
0446  *                    THE TABLE IS FULL, RTP=RTM+1.
0447  *
0448  *
0449  *
0450      EJCT
```

```
0451                *       EXTERNAL LINKAGE DECLARATIONS
0452                *
0453                *
0454                *
0455                        ENT   ERR       WE SUPPLY THE ERROR REPORTING ROUTINE
0456                        ENT   TYPE      MESSAGE TYPER
0457                        ENT   LODF      LOAD MODE FLAG
0458                        ENT   LSTF      PUNCH MODE FLAG
0459                        ENT   CPOS      ASR POSITION COUNTER
0460                        ENT   BRKF      PROGRAM BREAK FLAG
0461                        ENT   SBP       SOURCE BYTE POINTER
0462                        ENT   DBP       DESTINATION BYTE POINTER
0463                        ENT   SCHR      CHARACTER STORE ROUTINE
0464                        ENT   JOB       JOB COMMAND PROCESSOR
0465                        ENT   GNBC      GETS NEXT NON-BLANK CHARACTER
0466                        ENT   PTB       PROGRAM TEXT BASE ADDRESS (LOW POINTER)
0467                        ENT   SIT       STATEMENT INDEX TABLE ADDRESS(HIGH POINTER)
0468                        ENT   SBUF      USED AS INPUT BUFFER FOR INITIALIZATION
0469                        ENT   ATND      ADDRESS OF ARCTANGENT FUNCTION
0470                        ENT   TAND      ADDRESS OF TANGENT FUNCTION
0471                        ENT   SIND      ADDRESS OF SINE FUNCTION
0472                        ENT   COSD      ADDRESS OF COSINE FUNCTION
0473                        ENT   SQRD      ADDRESS OF SQUARE ROOT FUNCTION
0474                        ENT   DELT      ROUTINE TO FLAG ERROR OF
0475                        ENT   SCVL      STORES A AND B REGISTERS INTO THE FLOATING
0476                *                       ACCUMULATOR
0477                        ENT   PCVL      PRINTS CONTENTS OF THE FLOATING POINT
0478                *                       ACCUMULATOR
0479                        ENT   IDMS      ID MESSAGE
0480                        ENT   DFQ       MEESAGE FOR LIBRARY FUNCTION DELETION
0481                        ENT   SQRQ      MESSAGE ABOUT DELETING SQUARE ROOT FUNCTION
0482                        ENT   SCTQ      MESSAGE ABOUT DELETING SIN, COS, TAN
0483                        ENT   ATNQ      MESSAGE ABOUT DELETING ARCTANGENT
0484                        ENT   AYON      MESSAGE REQUESTS A YES OR NO ANSWER
0485                        ENT   HMAM      MESSAGE TO SET HIGH POINTER-SIT
0486                        ENT   AYOH      REQUESTS A YES OR A HIGH ADDRESS
0487                        ENT   USPM.     MESSAGE TO INDICATE AMOUNT OF USER SPACE
0488                        ENT   ISSM      FLAG INSUFFICIENT USER SPACE
0489                        ENT   C215
0490                        ENT   C300
0491                        ENT   C337
0492                        ENT   C1
0493                        ENT   C10
0494                        ENT   C221
0495                        ENT   C223
0496                        ENT   C240
0497                        ENT   C241
0498                        ENT   C260
0499                        ENT   C307
0500                        ENT   M1
```

```
0501                    ENT   M12
0502                    ENT   M100
0503                    ENT   F1
0504                    ENT   FM1
0505                *
0506                    EXT   INIT
0507                    EXT   LFCR      ASR LINE ADVANCE
0508                    EXT   IPUT      INPUT LINE
0509                    EXT   INA1      INPUT ONE CHARACTER
0510                    EXT   OTA1      OUTPUT ONE CHARACTER
0511                    EXT   BRKC      PROGRAM BREAK CHECK
0512                    EXT   ITAPE     INTIIALIZE PAPER TAPE PUNCHING
0513                    EXT   ETAPE     TERMINATE PAPER TAPE PUNCHING
0514                    EXT   SINF      SINE FUNCTION
0515                    EXT   COSF      COSINE FUNCTION
0516                    EXT   TANF      TANGENT FUNCTION
0517                    EXT   ATNF      ARCTANGENT FUNCTION
0518                    EXT   EXPF      EXPONENTIATION FUNCTION
0519                    EXT   ABSF      ABSOLUTE VALUE FUNCTION
0520                    EXT   LOGF      LOGARITHM FUNCTION
0521                    EXT   SQRF      SQUARE ROOT FUNCTION
0522                    EXT   RNDF      RANDOM NUMBER FUNCTION
0523                    EXT   SGNF      SIGN FUNCTION
0524                    EXT   L$22      FLOATING POINT LOAD
0525                    EXT   H$22      FLOATING POINT STORE
0526                    EXT   M$11      INTEGER MULTIPLY
0527                    EXT   A$22      FLOATING POINT ADDITION
0528                    EXT   S$22      FLOATING POINT SUBTRACTION
0529                    EXT   M$22      FLOATING POINT MULTIPLY
0530                    EXT   D$22      FLOATING POINT DIVISION
0531                    EXT   N$22      FLOATING POINT TWOS COMPLEMENT
0532                    EXT   E$22      FLOATING POINT EXPONENTIATION
0533                    EXT   TINT      FLOATING POINT INTEGER TEST
0534                    EXT   IFLT      FLOATING POINT TO INTEGER CONVERSION
0535                    EXT   FINT      INTEGER TO FLOATING POINT CONVERSION
0536                *
0537                *
0538                *
0539                    EJCT
```

```
0540                    *          TEMPORARY STORAGE
0541                    *
0542                    *
0543                    *
0544                    *
0545                         ORG   '20              START WAY DOWN LOW
0546                    *
0547                    *
0548                    *    TABLE POINTERS
0549                    *
0550                    *
0551 00020   000000     PTB  BSZ   1                PROGRAM TEXT TABLE BASE POINTER
0552 00021   000000     PTH  BSZ   1                PROGRAM TEXT TABLE HIGH POINTER
0553 00022   000000     DFB  BSZ   1                DEFINED FUNCTION INDEX BASE POINTER
0554 00023   000000     DFT  BSZ   1                DEFINED FUNCTION INDEX HIGH POINTER
0555 00024   000000     FNB  BSZ   1                FOR-NEXT TABLE BASE POINTER
0556 00025   000000     FNT  BSZ   1                FOR-NEXT TABLE HIGH POINTER
0557 00026   000000     SVB  BSZ   1                SIMPLE VARIABLE STORAGE BASE POINTER
0558 00027   000000     SVT  BSZ   1                SIMPLE VARIABLE STORAGE HIGH POINTER
0559 00030   000000     DVB  BSZ   1                DIMENSIONED VARIABLE STORAGE BASE POINTER
0560 00031   000000     DVT  BSZ   1                DIMENSIONED VARIABLE STORAGE HIGH POINTER
0561 00032   000000     SIB  BSZ   1                STATEMENT INDEX BASE POINTER
0562 00033   000000     SIT  BSZ   1                STATEMENT INDEX HIGH POINTER
0563 00034   000000     SIP  BSZ   1                CURRENT STATEMENT POINTER
0564 00035   000000     PDLP BSZ   1                PUSH DOWN STACK POINTER
0565 00036   000000     RTP  BSZ   1                RETURN STACK POINTER
0566                    *
0567                    *    PRIMARY BYTE POINTERS
0568                    *
0569 00037   000000     SBP  BSZ   1                PRIMARY BYTE FETCH POINTER
0570 00040   000000     DBP  BSZ   1                PRIMARY BYTE STORE POINTER
0571                    *
0572                    *    FLOATING POINT ACCUMULATORS
0573                    *
0574 00041   000000     CVAL BSZ   2                PRIMARY FLOATING POINT ACCUMULATOR
0575 00043   000000     LVAL BSZ   2                SECONDARY FLOATING POINT ACCUMULATOR
0576                    *
0577                    *    MISCELLANEOUS TEMPORARY STORAGE
0578                    *
0579 00045   000000     PRT1 BSZ   1                PRINT STMT
0580 00046   000000     CPOS BSZ   1                CURRENT CARRIAGE POSITION COUNTER
0581 00047   000000     RDT1 BSZ   1                READ STMT
0582 00050   000000     FSC  BSZ   1                FREE SPACE COUNTER
0583 00051   000000     IFT1 BSZ   1                IF STMT
0584 00052   000000     IFT2 BSZ   1                DITTO
0585         000034     LTT1 EQU   SIP              LIST COMMAND
0586 00053   000000     LTT2 BSZ   1                DITTO
0587 00054   000000     LTT3 BSZ   1                DITTO
0588 00055   000000     LTT4 BSZ   1                DITTO
0589 00056   000000     LTT5 BSZ   1                DITTO
```

```
0590 00057   000000     SNUM  BSZ   1        STATEMENT NUMBER SAVE
0591 00060   000000     SEQI  BSZ   1        SEQUENCING INHIBITION FLAG
0592 00061   000000     STT1  BSZ   1        SOURCE TEXT EDITOR
0593 00062   000000     STT2  BSZ   1        DITTO
0594 00063   000000     STT3  BSZ   1        DITTO
0595 00064   000000     STT4  BSZ   1        DITTO
0596 00065   000000     STT5  BSZ   1        DITTO
0597 00066   000000     STT6  BSZ   1        DITTO
0598 00067   000000     CLT1  BSZ   1        CALL STATEMENT
0599 00070   000000     CLT2  BSZ   1        DITTO
0600 00071   000000     CLT3  BSZ   1        DITTO
0601 00072   000000     DEFN  BSZ   1        DUMMY VARIABLE NAME
0602 00073   000000     DEFV  BSZ   2        DUMMY VARIABLE VALUE
0603 00075   000000     LOP   BSZ   1        LAST OPERATOR PRECEDENCE
0604 00076   000000     ILT1  BSZ   1        INPUT EDITOR
0605 00077   000000     ILT2  BSZ   1        DITTO
0606 00100   000000     ILT3  BSZ   1        DITTO
0607 00101   000000     ADT1  BSZ   1        ASSIGN DIMENSIONED VARIABLE
0608 00102   000000     ADT2  BSZ   1        DITTO
0609 00103   000000     ADT3  BSZ   1        DITTO
0610 00104   000000     ADT4  BSZ   1        DITTO
0611 00105   000000     ADT5  BSZ   1        DITTO
0612 00106   000000     ADT6  BSZ   1        DITTO
0613 00107   000000     ADT7  BSZ   1        DITTO
0614 00110   000000     ADT8  BSZ   1        DITTO
0615 00111   000000     DPTR  BSZ   1        LOCATE DIMENSIONED VARIABLE
0616 00112   000000     DCT1  BSZ   1        DITTO
0617 00113   000000     DCT2  BSZ   1        DITTO
0618 00114   000000     CHAR  BSZ   1        LAST CHARACTER FETCHED
0619 00115   000000     LCHR  BSZ   1        LAST CHARACTER STORED
0620 00116   000000     TMP1  BSZ   1        MISCELLANEOUS
0621 00117   000000     TMP2  BSZ   1        MISCELLANEOUS
0622 00120   000000     TMP3  BSZ   1        MISCELLANEOUS
0623 00121   000000     SIGN  BSZ   1        OUTPUT EDITOR
0624 00122   000000     EXP   BSZ   1        DITTO
0625 00123   000000     ECTR  BSZ   1        DITTO
0626 00124   000000     INHC  BSZ   1        DITTO
0627 00125   000000     CON1  BSZ   1        PROGRAM BREAK POINT SAVE
0628 00126   000000     CON2  BSZ   1        PROGRAM BREAK POINT SAVE
0629 00127   000000     BRKF  BSZ   1        PROGRAM BREAK FLAG
0630 00130   000000     DIMF  BSZ   1        DIMENSION STATEMENT FLAG
0631 00131   000001     LODF  OCT   1        MUST BE INITIALLY NONZERO OR ELSE*****
0632 00132   000000     LSTF  BSZ   1        LIST MODE FLAG
0633 00133             VARN  BSS   1        VARIABLE NAME STORAGE
0634 00134   000000     XON   BSZ   1        XON CHARACTER CONTROL FLAG
0635                    *
0636                    *    BUFFERS
0637                    *
0638 00135   000000     SBUF  BSZ   80       COMMAND INPUT BUFFER
0639 00255   000000     IBUF  BSZ   80       DATA INPUT BUFFER
```

```
0640           000256     WORKI EQU   IBUF+1          OUTPUT EDITOR SCRATCH AREA
0641 00375     000000     RSTKI BSZ   16              RETURN STACK
0642                      *
0643                      *
0644                      *
0645                           EJCT
```

```
0646                        *       THE FOLLOWING MESSAGES ARE USED BY THE INITIALIZATION
0647                        *   ROUTINE WHICH SETS THE HIGH AND LOW POINTERS(SIT AND PTB).  THE
0648                        *   SPACE USED BY THE MESSAGES IS LATER USED FOR INPUT AND OUTPUT
0649                        *   BUFFERS.
0650          000415    HERE SET   *
0651                       ORG   *-125
0652 00220    141301    IDMS BCI   9,BASIC-16  11-19-70
     00221    151711
     00222    141655
     00223    130666
     00224    120240
     00225    130661
     00226    126661
     00227    134655
     00230    133660
0653 00231    000000       OCT   0
0654 00232    120301    ATNQ BCI   2, ATN
     00233    152316
0655 00234    137640       VFD   8,'277,8,'240
0656 00235    124301    AYON BCI   9,(ANSWER YES OR NO)
     00236    147323
     00237    153705
     00240    151240
     00241    154705
     00242    151640
     00243    147722
     00244    120316
     00245    147651
0657 00246    000000       OCT   0
0658 00247    151640    SCTQ BCI   8,5 SIN, COS, TAN
     00250    151711
     00251    147254
     00252    120303
     00253    147723
     00254    126240
     00255    152301
     00256    147240
0659 00257    137400       VFD   8,'277,8,0
0660 00260    120323    SQRQ BCI   2, SQR
     00261    150722
0661 00262    137400       VFD   8,'277,8,0
0662 00263    142317    DFQ  BCI   19,DO YOU WISH TO DELETE LIBRARY FUNCTION
     00264    120331
     00265    147725
     00266    120327
     00267    144723
     00270    144240
     00271    152317
     00272    120304
     00273    142714
```

```
                00274   142724
                00275   142640
                00276   146311
                00277   141322
                00300   140722
                00301   154640
                00302   143325
                00303   147303
                00304   152311
                00305   147716
0663            00306   000000          OCT   0
0664            00307   120317  HMAM:   BCI   11,  OK TO USE ALL OF CORE
                00310   145640
                00311   152317
                00312   120325
                00313   151705
                00314   120301
                00315   146314
                00316   120317
                00317   143240
                00320   141717
                00321   151305
0665            00322   137400          VFD   8,'277,8,0
0666            00323   124301  AYOH:   BCI   20,(ANSWER YES OR GIVE HIGH OCTAL ADDRESS)
                00324   147323
                00325   153705
                00326   151240
                00327   154705
                00330   151640
                00331   147722
                00332   120307
                00333   144726
                00334   142640
                00335   144311
                00336   143710
                00337   120317
                00340   141724
                00341   140714
                00342   120301
                00343   142304
                00344   151305
                00345   151723
                00346   124640
0667            00347   000000          OCT   0
0668            00350   144716  ISSM:   BCI   12,INSUFFICIENT USER SPACE
                00351   151725
                00352   143306
                00353   144703
                00354   144705
                00355   147324
```

```
                00356   120325
                00357   151705
                00360   151240
                00361   151720
                00362   140703
                00363   142640
0669            00364   000000          OCT   0
0670            00365   120314  USPM    BCI   19, LOCATIONS FOR USER STORAGE AND TABLES
                00366   147703
                00367   140724
                00370   144717
                00371   147323
                00372   120306
                00373   147722
                00374   120325
                00375   151705
                00376   151240
                00377   151724
                00400   147722
                00401   140707
                00402   142640
                00403   140716
                00404   142240
                00405   152301
                00406   141314
                00407   142723
0671            00410   000000          OCT   0
0672                                    ORG   HERE
0673                                    EJCT
```

```
0674                    *       CONSTANTS AND FIXED POINTERS
0675                    *
0676                    *
0677                    *
0678 00415   177776     B16T HEX  FFFE          MASK TO CLEAR BIT 16
0679 00416   000000     C0   OCT  0
0680 00417   000001     C1   OCT  1
0681 00420   000010     C10  OCT  10
0682 00421   000011     C11  OCT  11
0683 00422   000110     C110 OCT  110
0684 00423   000012     C12  OCT  12
0685 00424   000016     C16  OCT  16
0686 00425   000017     C17  OCT  17
0687 00426   000002     C2   OCT  2
0688 00427   000020     C20  OCT  20
0689 00430   000221     C221 OCT  221
0690 00431   000023     C23  OCT  23
0691 00432   000212     C212 OCT  212
0692 00433   000215     C215 OCT  215
0693 00434   000220     C220 OCT  220
0694 00435   000223     C223 OCT  223
0695 00436   000240     C240 OCT  240
0696 00437   000241     C241 OCT  241
0697 00440   000242     C242 OCT  242
0698 00441   000250     C250 OCT  250
0699 00442   000251     C251 OCT  251
0700 00443   000252     C252 OCT  252
0701 00444   000253     C253 OCT  253
0702 00445   000254     C254 OCT  254
0703 00446   000255     C255 OCT  255
0704 00447   000256     C256 OCT  256
0705 00450   000257     C257 OCT  257
0706 00451   000260     C260 OCT  260
0707 00452   000261     C261 OCT  261
0708 00453   000272     C272 OCT  272
0709 00454   000273     C273 OCT  273
0710 00455   000275     C275 OCT  275
0711 00456   000277     C277 OCT  277
0712 00457   000003     C3   OCT  3
0713 00460   000300     C300 OCT  300
0714 00461   000305     C305 OCT  305
0715 00462   000307     C307 OCT  307
0716 00463   000333     C333 OCT  333
0717 00464   000336     C336 OCT  336
0718 00465   000337     C337 OCT  337
0719 00466   000004     C4   OCT  4
0720 00467   000043     C43  OCT  43
0721 00470   000044     C44  OCT  44
0722 00471   000005     C5   OCT  5
0723 00472   000050     C50  OCT  50
```

```
0724 00473   000052     C52   OCT   52
0725 00474   000054     C54   OCT   54
0726 00475   000013     CMAX  DEC   11          MAXIMUM SUBROUTINE IDENTIFIER + 1
0727 00476   000021     DTAC  OCT   21          'DATA'
0728 00477   040300     F1    DEC   1.0
     00500   000000
0729 00501   041120     F10   DEC   10.0
     00502   000000
0730 00503   023710     F10R  DEC   10.0E-16    PREVENTS UNDERFLOW IN NUMERIC CONVERSION
     00504   007276
0731 00505   137500     FM1   DEC   -1.0
     00506   000000
0732 00507   000006     GTC   OCT   6           'GOTO'
0733 00510   000176     INTF  OCT   176         INTEGER CONSTANT FLAG
0734 00511   000177     RELF  OCT   177         REAL CONSTANT FLAG
0735 00512   040511     LE10  OCT   040511,127307  NATURAL LOG OF 10
     00513   127307
0736 00514   0 003734   LSBP  DAC   IDNT+IDNT    POINTER TO RESERVED NAME LIST
0737 00515   177777     M1    OCT   -1
0738 00516   177770     M10   OCT   -10
0739 00517   177700     M100  OCT   -100
0740 00520   177766     M12   OCT   -12
0741 00521   177776     M2    OCT   -2
0742 00522   177760     M20   OCT   -20
0743 00523   177757     M21   OCT   -21
0744 00524   177775     M3    OCT   -3
0745 00525   177773     M5    OCT   -5
0746 00526   177727     M51   OCT   -51
0747 00527   177725     M53   OCT   -53
0748 00530   177772     M6    OCT   -6
0749 00531   000105     MCOL  DEC   69          MAX NO. OF ASR COLUMNS - 2
0750 00532   033723     ROND  DEC   .000005     ROUNDING FACTOR
     00533   161326
0751 00534   0 000375   RTB   DAC   RSTK        RETURN STACK BASE POINTER
0752 00535   0 000414   RTM   DAC   RSTK+15     RETURN STACK HIGH POINTER
0753 00536   177751     SMAX  OCT   -27         POINTER TO IDENTIFIERS THAT REQUIRE SPACES
0754 00537   023417     SNMX  DEC   9999.       LARGEST LEGAL STATEMENT NUMBER
0755 00540   000025     STPC  OCT   25          'STEP'
0756 00541   000043     TABC  OCT   43          'TAB('
0757 00542   000026     THNC  OCT   26          'THEN'
0758 00543   000024     TOC   OCT   24          'TO'
0759 00544   000022     DIMC  OCT   22          'DIM'
0760 00545   000042     FNC   OCT   42          'FN'
0761         000545     DEFF  EQU   FNC
0762 00546   000026     SYSL  OCT   26          PNTR TO FIRST SYS FUNCTION - 1
0763 00547   000042     SYSH  OCT   42          PNTR TO LAST SYS FUNCTION + 1
0764 00550   0 000256   WRKD  DAC   WORK        POINTER TO OUTPUT EDITOR SCRATCH AREA
0765 00551   0 000264   WKD7  DAC   WORK+6      POINTER TO 7TH WORD OF SCRATCH AREA
0766 00552   000016     REMF  OCT   16          'REM'
0767 00553   0 000073   DFVD  DAC   DEFV        POINTER TO DUMMY VARIABLE VALUE
```

```
0768 00554  -0 10 00554 CJST JST*  *          PROTOTYPE CALL'
0769 00555   0 004035         DAC   CL01-1     SUBROUTINE LINKAGE ADDRESS
0770 00556   0 004035         DAC   CL01-1     SUBROUTINE LINKAGE ADDRESS
0771 00557   0 004035         DAC   CL01-1     SUBROUTINE LINKAGE ADDRESS
0772 00560   0 004035         DAC   CL01-1     SUBROUTINE LINKAGE ADDRESS
0773 00561   0 004035         DAC   CL01-1     SUBROUTINE LINKAGE ADDRESS
0774 00562   0 004035         DAC   CL01-1     SUBROUTINE LINKAGE ADDRESS
0775 00563   0 004035         DAC   CL01-1     SUBROUTINE LINKAGE ADDRESS
0776 00564   0 004035         DAC   CL01-1     SUBROUTINE LINKAGE ADDRESS
0777 00565   0 004035         DAC   CL01-1     SUBROUTINE LINKAGE ADDRESS
0778 00566   0 004035         DAC   CL01-1     SUBROUTINE LINKAGE ADDRESS
0779 00567   0 01 03765 CJMP. JMP   CL06       SUBROUTINE RETURN COMMAND
0780                    *
0781                    *
0782                    *
0783                    EJCT
```

```
0784                        *          SYSTEM FUNCTION ADDRESS LIST
0785                        *
0786                        *
0787                        *
0788                        *
0789          000567        SFNL: EQU  *-1
0790 00570    0 000000      SIND  DAC  SINF          SINE FUNCTION
0791 00571    0 000000      COSD  DAC  COSF          COSINE FUNCTION
0792 00572    0 000000      TAND  DAC  TANF          TANGENT FUNCTION
0793 00573    0 000000      ATND  DAC  ATNF          ARCTANGENT FUNCTION
0794 00574    0 000000            DAC  EXPF          EXPONENTIAL FUNCTION
0795 00575    0 000000            DAC  ABSF          ABSOLUTE VALUE FUNCTION
0796 00576    0 000000            DAC  LOGF          NATURAL LOGARITHM FUNCTION
0797 00577    0 000000      SQRD  DAC  SQRF          SQUARE ROOT FUNCTION
0798 00600    0 000000            XAC  INTF          INTEGER PART FUNCTION
0799 00601    0 000000            DAC  RNDF          RANDOM NUMBER FUNCTION
0800 00602    0 000000            DAC  SGNF          SIGN FUNCTION
0801                        *
0802                        *
0803                        *
0804 00603    0 000604      DELT  DAC  *+1
0805 00604    0 000000            DAC  **            DELETED FUNCTION ERROR ROUTINE
0806 00605    0 10 05206          JST  ERR           FLAG ERROR
0807 00606    142306              BCI  1,DF          DF
0808                        *
0809                        *
0810                        *
0811                              EJCT
```

```
0812                    *       COMMAND INPUT DISCRIMINATOR
0813                    *
0814                    *
0815                    *       THIS ROUTINE WILL OUTPUT A QUESTION MARK AND
0816                    *   THEN INPUT A USER RESPONSE LINE.  DEPENDING ON THE
0817                    *   USERS INPUT, THE FOLLOWING ACTION WILL BE TAKEN:
0818                    *
0819                    *       1)  IF A LINE NUMBER STARTS THE INPUT LINE,
0820                    *   CONTROL WILL BE PASSED TO THE SOURCE TEXT EDITOR.
0821                    *
0822                    *       2)  IF THE INPUT IS A SYSTEM COMMAND, CONTROL
0823                    *   WILL BE PASSED TO THE APPROPRIATE PROCESSING ROUTINE.
0824                    *
0825                    *   3)  IF NONE OF THE ABLOVE, IT WILL BE EXECUTED AS AN
0826                    *   IMMEADIATE MODE STATEMENT.
0827                    *
0828                    *
0829                        ORG   '1000             MAIN BODY OF PROGRAM STARTS HERE
0830                    *
0831 01000   0 10 00000 CMOD JST   INIT             FIRST TIME INITIALIZATION
0832                    *
0833                    *   THE PREVIOUS INSTRUCTION WILL BE REPLACED BY A 'CRA' BY
0834                    *   THE INITIALIZATION ROUTINE.
0835                    *
0836 01001   0 04 00034      STA   SIP              SET SIP TO INDICATE A COMMAND LINE
0837 01002   0 02 00046      LDA   CPOS             IF NOT AT START OF ASR LINE,
0838 01003   100040         SZE                    OUTPUT CARRIAGE RETURN, LINE FEED
0839 01004   0 10 00000      JST   LFCR             X
0840 01005   0 02 00456      LDA   C277             REQUEST USER INPUT
0841 01006   0 10 01467      JST   ILIN             X
0842 01007   0 000272   SSBP DAC   SBUF+SBUF       BYTE POINTER TO SYSTEM INPUT BUFFER
0843 01010   0 02 01007      LDA   SSBP             SET POINTER TO FIRST CHARACTER OF
0844 01011   0 04 00037      STA   SBP              THE INPUT BUFFER
0845 01012   0 10 03020      JST   XCHR             EXAMINE FIRST INPUT ITEM
0846 01013   0 11 00510      CAS   INTF             TEST FOR LINE NUMBER
0847 01014   100000         SKP                    NO
0848 01015   0 01 01306      JMP   STMT             YES ... GO TO TEXT EDITOR
0849 01016   0 04 00131      STA   LODF             NO ... RESET LOAD MODE FLAG
0850 01017   0 10 03137      JST   DLCK             NULL LINE ?
0851 01020   100000         SKP                    NO
0852 01021   0 01 01000      JMP   CMOD             YES ... GET ANOTHER INPUT LINE
0853 01022   0 11 00467      CAS   C43              TEST FOR SYSTEM COMMAND
0854 01023   0 11 00474      CAS   C54              X
0855 01024   0 01 03157      JMP   ESMT             NO....EXECUTE IMMEADIATE COMMAND
0856 01025   0 01 03157      JMP   ESMT             NO....EXECUTE IMMEADIATE COMMAND
0857 01026   0 04 00000      STA   0                POINTER TO SYSTEM FUNCTION IN INDEX
0858 01027   -1 01 00764     JMP*  *-'43,1          BRANCH TO PROCESS A SYSTEM FUNCTION
0859 01030   0 001040        DAC   JOB              JOB COMMAND
0860 01031   0 001045        DAC   CLER             CLEAR COMMAND
0861 01032   0 001047        DAC   RUN              RUN COMMAND
```

```
0862 01033   0 001132        DAC   LIST        LIST COMMAND
0863 01034   0 001114        DAC   CONT        CONTINUE COMMAND
0864 01035   0 001123        DAC   QUIT        QUIT COMMAND
0865 01036   0 001125        DAC   LOAD        LOAD COMMAND
0866 01037   0 001130        DAC   PNCH        PUNCH COMMAND
0867                     *
0868                     *
0869                     *
0870                         EJCT
```

```
0871                        *         JOB COMMAND PROCESSOR
0872                        *
0873                        *
0874                        *     THE JOB COMMAND CLEARS ALL TABLES INCLUDING THE
0875                        *   PROGRAM STORAGE AREA.
0876                        *
0877                        *
0878                        *
0879 01040   0 02 00020 JOB  LDA   PTB              CLEAR THE PROGRAM TEXT TABLE
0880 01041   0 04 00021      STA   PTH              X
0881 01042   0 02 00033      LDA   SIT              DELETE ALL ENTRIES IN THE
0882 01043   141206          AOA                    STATEMENT INDEX
0883 01044   0 04 00032      STA   SIB              X
0884                        *
0885                        *   FALL THROUGH TO 'CLEAR' PROCESSOR
0886                        *
0887                        *
0888                        *
0889                        *
0890                            EJCT
```

```
                        *
                        *       CLEAR COMMAND PROCESSOR
0891                    *
0892                    *
0893                    *
0894                    *       THE CLEAR COMMAND INITIALIZES ALL TABLES EXECPT
0895                    *   THE PROGRAM TEXT STORAGE AREA AND THE STATEMENT INDEX.
0896                    *
0897                    *
0898 01045   0 10 02734 CLER1 JST   CLRT            CLEAR THE TABLES
0899 01046   0 01 01000       JMP   CMOD            GO PROCESS NEXT COMMAND
0900                    *
0901                    *
0902                    *
0903                        EJCT
```

```
0904                    *          RUN COMMAND PROCESSOR
0905                    *
0906                    *
0907                    *          ALL USERS TABLES WILL BE CLEARED, ALL DIM STATEMENTS
0908                    *     WILL BE PROCESSED, AND THEN PROGRAM EXECUTION WILL BE STARTED
0909                    *     AT THE LOWEST NUMBERED STATEMENT.
0910                    *
0911                    *
0912                    *
0913 01047   0 10 02734 RUN   JST   CLRT          CLEAR USER TABLES
0914 01050   0 04 00060       STA   SEQI          RESET SEQUENCING INIBITION FLAG
0915 01051   0 04 00130       STA   DIMF          SET DIMENSION STATEMENT FLAG
0916 01052   0 10 02755       JST   IPDS          SET UP THE PUSH DOWN STACK
0917 01053   0 02 00032       LDA   SIB           SETUP TO PROCESS ALL DIM STMTS
0918 01054   0 07 00426       SUB   C2            X
0919 01055   0 04 00034       STA   SIP           X
0920 01056   0 02 00544 RN01  LDA   DIMC          FIND THE NEXT DIMENSION STATEMENT
0921 01057   0 10 04433       JST   SSR           X
0922 01060   0 01 01073       JMP   RN02          ALL DIMENSION STMTS HAVE BEEN PROCESSED
0923 01061   0 10 04606 RN03  JST   PVN           ISOLATE VARIABLE NAME
0924 01062   0 10 04714       JST   ADV           CREATE TABLE ENTRY FOR THIS VARIABLE
0925 01063   0 01 01112       JMP   RN04          ERROR...ILLEGAL DIMENSIONED VARIABLE NAME
0926 01064   0 10 03013       JST   GCHR          FETCH ITEM TERMINATOR
0927 01065   0 11 00445       CAS   C254          IF COMMA, THEN MORE NAMES FOLLOW
0928 01066   100000           SKP                 NO
0929 01067   0 01 01061       JMP   RN03          GO PROCESS NEXT VARIABLE
0930 01070   0 10 03031       JST   UCHR          NOT COMMA, MUST BE : OR C/R
0931 01071   0 10 03062       JST   GDLM          X
0932 01072   0 01 01056       JMP   RN01          GO CHECK FOR MORE DIMENSION STMTS
0933                    *
0934 01073   0 02 01007 RN02  LDA   SSBP          RESTORE POINTER TO COMMAND LINE
0935 01074   0 04 00037       STA   SBP           X
0936 01075   0 02 00032       LDA   SIB           SET SI POINTER TO START
0937 01076   0 07 00426       SUB   C2            EXECUTION AT LOWEST NUMBERED
0938 01077   0 04 00034       STA   SIP           X
0939 01100   0 02 00423       LDA   C12           RESET DIMENSION STMT FLAG
0940 01101   0 04 00130       STA   DIMF          X
0941 01102   0 10 03013       JST   GCHR          STEP OVER 'RUN'
0942 01103   0 10 03020       JST   XCHR          TEST FOR STARTING LINE NUMBER
0943 01104   0 10 03137       JST   DLCK          X
0944 01105   100000           SKP                 MAYBE
0945 01106   0 01 04554       JMP   ASO           NO ... START AT LOWEST NUMBERED LINE
0946 01107   140040           CRA                 SIP=0 SO THAT ERROR DIAGNOSTICS
0947 01110   0 04 00034       STA   SIP           WILL COME OUT RIGHT
0948 01111   0 01 03270       JMP   GOTO          EXECUTE A 'GOTO' STATEMENT
0949                    *
0950 01112   0 10 05206 RN04  JST   ERR           REPORT DIMENSION VARIBALE NAME ERROR
0951 01113   142316           BCI   1,DN
0952                    *
0953                    *
```

```
0954                *
0955                *   EJCT
```

```
0956                        *          CONTINUE COMMAND PROCESSOR
0957                        *
0958                        *
0959                        *      IF A PREVIOUS BREAK OPERATION HAS BEEN
0960                        *    PERFORMED AND IF SEQUENCING HAS NOT BEEN INHIBITED,
0961                        *    STATEMENT EXECUTION WILL BE RESUMED AT THE POINT
0962                        *    WHERE IT WAS INTERRUPTED.
0963                        *
0964                        *
0965                        *
0966 01114   0 02 00125 CONT LDA   CON1          IS THERE A BREAK OUTSTANDING C
0967 01115   101040         SNZ                 X
0968 01116   0 01 01000      JMP   CMOD          NO....GO GET NEXT COMMAND
0969 01117   0 04 00034      STA   SIP           YES...SET POINTER TO STMT WE WERE AT
0970 01120   0 02 00126      LDA   CON2          SET BYTE POINTER TO THE FIRST
0971 01121   0 04 00037      STA   SBP           CHARACTER OF THAT STATEMENT
0972 01122   0 01 03157      JMP   ESMT          GO CONTINUE EXECUTION
0973                        *
0974                        *
0975                        *
0976                             EJCT
```

```
0977                    *       QUIT COMMAND PROCESSOR.
0978                    *
0979                    *
0980                    *           THIS ROUTINE WILL HALT THE CENTRAL PROCESSOR.  IF
0981                    *       THE START BUTTON IS DEPRESSED, CONTROL WILL BE RETURNED
0982                    *       TO COMMAND MODE.
0983                    *
0984                    *
0985                    *
0986 01123   000000     QUIT HLT                    STOP THE COMPUTER
0987 01124   0 01 01000      JMP   CMOD             ON RESTART, GET NEXT COMMAND
0988                    *
0989                    *
0990                    *
0991                         EJCT
```

```
0992                        *       LOAD COMMAND PROCESSOR
0993                        *
0994                        *
0995                        *       WHEN IN LOAD MODE, THE PAPER TAPE READER WILL BE
0996                        *   TURNED ON WHEN EACH LINE IS REQUESTED, AND THE INPUT
0997                        *   PREFIX CHARACTER WILL NOT BE PRINTED.
0998                        *
0999                        *
1000                        *
1001 01125   140040     LOAD CRA                    SET THE LOAD FLAG
1002 01126   0 04 00131      STA   LODF              X
1003 01127   0 01 01000      JMP   CMOD              START READING IN TEXT
1004                        *
1005                        *
1006                        *
1007                        EJCT
```

```
1008                        *          PUNCH COMMAND PROCESSOR
1009                        *
1010                        *
1011                        *
1012 01130   0 10 00000 PNCH: JST   ITAPE          INITIALIZE TAPE OUTPUT
1013 01131   0 12 00132       IRS   LSTF           TURN ON PUNCH FLAG
1014                        *
1015                        *     FALLI THROUGH TO LIST PROCESSORI
1016                        *
1017                        *
1018                        *
1019                             EJCT
```

```
1020                    *          LIST COMMAND PROCESSOR
1021                    *
1022                    *
1023                    *     THIS ROUTINE PRINTS A SOURCE LISTING
1024                    *   OF THE PROGRAM THAT IS CURRENTLY STORED. THE LISTING
1025                    *   IS PRECEEDED AND FOLLOWED BY TWO BLANK LINES.
1026                    *
1027                    *
1028 01132   0 02 00432 LIST LDA   C212          PRINT A COUPLE OF BLANK LINES
1029 01133   0 10 00000      JST   OTA1          X
1030 01134   0 10 00000      JST   OTA1          X
1031 01135   140040          CRA
1032 01136   0 04 00055      STA   LTT4          INITIALIZE LOW LIST RANGE POINTER
1033 01137   0 02 00477      LDA   F1            INITIALAIZE HIGH LIST RANGE POINTER TO
1034 01140   0 04 00056      STA   LTT5          SOME VALUE > 9999
1035 01141   0 10 03013      JST   GCHR          STEP OVER 'LIST'
1036 01142   0 10 03020      JST   XCHR          TEST FOR LIST PARAMETERS
1037 01143   0 10 03137      JST   DLCK          X
1038 01144   100000          SKP                 ......YES......
1039 01145   0 01 01163      JMP   LT11          NO ... START LISTING
1040 01146   0 11 00445      CAS   C254          IS FIRST PARAMETER MISSING ¢
1041 01147   100000          SKP                 NO
1042 01150   0 01 01154      JMP   LT13          YES ... LIST UPTO SECOND PARAMETER
1043 01151   0 10 04532      JST   ISN           INPUT LOW RANGE PARAMETER
1044 01152   0 07 00417      SUB   C1            ADJUST AS RANGE CHECK IN EXCLUSIVE
1045 01153   0 04 00055      STA   LTT4          SAVE IT
1046 01154   0 10 03013 LT13 JST   GCHR          GET PARAMETER DELIMITER
1047 01155   0 10 03137      JST   DLCK          SECOND PARAMETER MISSING ¢
1048 01156   100000          SKP                 NO
1049 01157   0 01 01163      JMP   LT11          YES ... START LISTING
1050 01160   0 10 04532      JST   ISN           INPUT THE SECOND PARAMETER
1051 01161   141206          AOA                 ADJUST AS RANGE CHECK IS EXCLUSIVE
1052 01162   0 04 00056      STA   LTT5          SAVE IT
1053 01163   0 02 00032 LT11 LDA   SIB           START SCAN AT LOWEST NUMBERED STATEMENT
1054 01164   0 11 00033 LT01 CAS   SIT           ARE WE PAST END OF TABLE¢
1055 01165   0 01 01300      JMP   LT02          YES...GO FINISH UP
1056 01166   000000          OCT   0             NEVER CAN EXECUTE THIS WORD
1057 01167   0 04 00034      STA   LTT1          SAVE POINTER IN SAFE PLACE
1058 01170  -0 02 00034      LDA*  SIP           GET NO. OF CURRENT STATEMENT
1059 01171   0 11 00055      CAS   LTT4          IS IT IN LISTING RANGE ¢
1060 01172   0 11 00056      CAS   LTT5          X
1061 01173   0 01 01275      JMP   LT14          ....NO....MOVE ON TO NEXT STATEMENT
1062 01174   0 01 01275      JMP   LT14          ....NO....MOVE ON TO NEXT STATEMENT
1063 01175   0 10 02702      JST   PLN           PRINT LINE NUMBER OF THIS STATEMENT
1064 01176   0 35 00034      LDX   LTT1          X POINTS TO SI ENTRY FOR THIS STATEMENT
1065 01177   1 02 00001      LDA   1,1           SET SBP TO START OF SOURCE FOR THIS
1066 01200   0 04 00037      STA   SBP           STATEMENT
1067 01201   0 10 03020      JST   XCHR          IF NEXT CHARACTER IS NOT A
1068 01202   0 11 00433      CAS   C215          SPECIAL IDENTIFIER, PRINT A
1069 01203   0 10 02730      JST   SPAC          SPACE TO MAKE THE LISTING LINE UP
```

```
1070 01204   101000              NOP
1071 01205   0 10 02730 LT12  JST   SPAC          X
1072 01206   0 10 03013 LT03  JST   GCHR          GET THE NEXT SOURCE CHARACTER
1073 01207   0 11 00510       CAS   INTF          TEST FOR INTEGER CONSTANT
1074 01210   100000            SKP                 NO
1075 01211   0 01 01252       JMP   LT04          YES...GO PRINT IT
1076 01212   0 11 00511       CAS   RELF          NO...TEST FOR REAL CONSTANT
1077 01213   100000            SKP                 NO
1078 01214   0 01 01255       JMP   LT05          YES...GO PRINT IT
1079 01215   0 11 00433       CAS   C215          TEST FOR NORMAL CHARACTER
1080 01216   0 10 00000       JST   OTA1          YES...PRINT IT
1081 01217   0 01 01267       JMP   LT06          GO SEE IF IT WAS CARRIAGE RETURN
1082 01220   140407            TCA                 SPECIAL IDENTIFIER...NEGATE FOR TABLE SCAN
1083 01221   0 04 00053       STA   LTT2          SAVE FOR LATER REFERENCE
1084 01222   0 04 00054       STA   LTT3          SAVE FOR COUNTING
1085 01223   0 11 00536       CAS   SMAX          DOES THIS IDENTIFIER REQUIRE A LEADING SPACE ¢
1086 01224   0 10 02730       JST   SPAC          YES...PRINT ONE
1087 01225   101000            NOP                 NO
1088 01226   0 35 00037       LDX   SBP           SAVE POINTER TO CURRENT STATEMENT
1089 01227   0 02 00514       LDA   LSBP          SET POINTER TO START OF THE
1090 01230   0 04 00037       STA   SBP           IDENTIFIER LIST
1091 01231   0 12 00054 LT09  IRS   LTT3          ARE WE AT THE CORRECT ENTRY¢
1092 01232   0 01 01246       JMP   LT07          NO...GO FEED THROUGH CURRENT ENTRY
1093 01233   0 10 03013 LT08  JST   GCHR          GET CHARACTER OF EXPANSION
1094 01234   100040            SZE                 DO NOT PRINT IT IF IDENTIFIER MARK
1095 01235   0 10 00000       JST   OTA1          PRINT CHARACTER OF IDENTIFIER EXPANSION
1096 01236   100040            SZE                 HAS ALL OF THE IDENTIFIER BEEN PRINTED¢
1097 01237   0 01 01233       JMP   LT08          NO...GO PRINT NEXT CHARACTER OF IDENTIFIER
1098 01240   0 15 00037       STX   SBP           RESTORE POINTER TO STMT WE ARE PRINTING
1099 01241   0 02 00053       LDA   LTT2          PRINT TRAILING SPACE
1100 01242   0 11 00536       CAS   SMAX          IF REQUIRED
1101 01243   0 01 01205       JMP   LT12          REQUIRED...GO PRINT SPACE
1102 01244   0 01 01206       JMP   LT03          NOT REQUIRED
1103 01245   0 01 01206       JMP   LT03          NOT REQUIRED
1104 01246   0 10 03013 LT07  JST   GCHR          STEP THROUGH CURRENT IDENTIFIER
1105 01247   100040            SZE                 END OF IDENTIFIER¢
1106 01250   0 01 01246       JMP   LT07          NO...CONTINUE SCAN
1107 01251   0 01 01231       JMP   LT09          HIT THE END OF IT
1108                    *
1109                    *     HERE TO PRINT CONSTANTS
1110                    *
1111 01252   0 10 03043 LT04  JST   GCPK          PACK THE INTEGER CONSTANT
1112 01253   0 10 00000       JST   FINT          FLOAT IT
1113 01254   0 01 01261       JMP   LT10          NOW TREAT AS REAL
1114 01255   0 10 03043 LT05  JST   GCPK          PACK FIRST WORD OF REAL CONSTANT
1115 01256   000201            IAB                 X
1116 01257   0 10 03043       JST   GCPK          PACK SECOND WORD OF THE CONSTANT
1117 01260   000201            IAB                 PUT NUMBER IN NORMAL FORM
1118 01261   0 10 03153 LT10  JST   SCVL          PRINT ROUTINE LOOKS FOR NUMBER IN CVAL
1119 01262   140040            CRA                 CLEAR CARRIAGE POSITION COUNTER TO
```

```
1120 01263   0 04 00046        STA   CPOS          PREVENT A C/R-L/F FROM BEING GENERATED
1121 01264   0 02 00436        LDA   C240          SURPASS ALL SPACES IN NUMBER
1122 01265   0 10 02123        JST   PCVL          PRINT THE NUMBER
1123 01266   0 01 01206        JMP   LT03          GO WORK ON NEXT CHARACTER
1124                          *
1125                          *     HERE FOR END FOR LINE TEST
1126                          *
1127 01267   0 11 00433 LT06   CAS   C215          IS CURRENT CHAR A C/R ¢
1128 01270   0 01 01206        JMP   LT03          NO...GO PROCESS NEXT ITEM
1129 01271   0 10 00000        JST   LFCR          ADVANCE ASR LINE
1130 01272   0 13 00127        IMA   BRKF          HAS THE USER HAD ENOUGH ¢
1131 01273   100040            SZE                 X
1132 01274   0 01 01300        JMP   LT02          YES ... RETURN TO COMMAND MODE
1133 01275   0 02 00034 LT14   LDA   LTT1          UPDATE SI POINTER TO
1134 01276   0 06 00426        ADD   C2            NEXT ENTRY
1135 01277   0 01 01164        JMP   LT01          GO TEST FOR END OF LISTING
1136                          *
1137                          *     HERE AT END OF LISTING
1138                          *
1139 01300   0 10 00000 LT02   JST   LFCR          PRINT A COUPLE OF BLANK LINES
1140 01301   0 10 00000        JST   LFCR          X
1141 01302   0 13 00132        IMA   LSTF          FETCH AND CLEAR PUNCH FLAG
1142 01303   100040            SZE                 WERE WE PUNCHING ¢ ?
1143 01304   0 10 00000        JST   ETAPE         YES ... PUNCH TRAILER
1144 01305   0 01 01000        JMP   CMOD          GO PROCESS NEXT COMMAND
1145                          *
1146                          *
1147                                EJCT
```

```
                    *
                    *        SOURCE TEXT EDITOR
                    *
                    *
                    *    FUNCTION:
                    *
                    *         THIS ROUTINE IS CALLED BY THE COMMAND LINE
                    *    RESPONSE ROUTINE WHEN A STATEMENT NUMBER IS DETECTED AT
                    *    THE BEGINNING OF A RESPONSE LINE. DEPENDING ON THE
                    *    LINE ENTERED, THE FOLLOWING ACTIONS MAY BE TAKEN:
                    *
                    *         1. IF IT IS A LINE NUMBER IMMEDIATELY FOLLOWED BY A
                    *    CARRIAGE RETURN, ANY EXISTING LINE WITH THAT NUMBER WILL
                    *    BE DELETED EROM STORAGE.
                    *         2. IF A CARRIAGE RETURN DOES NOT IMMEDIATELY FOLLOW
                    *    THE LINE NUMBER, AND A STATEMENT WITH THAT NUMBER ALREADY
                    *    EXISTS, THE PREVIOUS COPY WILL BE DELETED AND THE NEW TEXT
                    *    WILL BE INSERT IN ITS PLACE.
                    *         3. IF A CARRIAGE RETURN DOES NOT IMMEDIATELY FOLLOW
                    *    THE LINE NUMBER, AND A STATEMENT WITH THAT NUMBER DOES
                    *    NOT EXIST, THE NEW LINE WILL BE ADDED TO THE PROGRAM.
                    *
                    *
                    *
01306    0 10 02734 STMT JST    CLRT          CLEAR ALL TABLES EXECPT PROGRAM TEXT AND SI
01307    0 10 04532      JST    ISN           GET THE STATEMENT NUMBER IN SNUM
01310    0 02 00037      LDA    SBP           SAVE POINTER TO NEW STATEMENT
01311    0 04 00061      STA    STT1          UNTIL WE ARE READY FOR IT
01312    0 10 04451      JST    SISR          DO WE ALREADY HAVE A STMT WITH THAT NUMBER¢
01313    0 01 01435      JMP    STO1          NO...GO APPEND A STATEMENT
01314    0 10 04514 STO2 JST    SES           FIND END OF STATEMENT
01315    0 11 00433      CAS    C215          IS IT ACTUAL END OF LINE¢
01316    0 01 01314      JMP    STO2          NO... CONTINUE SCAN
01317    0 02 00037      LDA    SBP           CALCULATE THE NUMBER OF
01320   141206           AOA                  BYTES IN THE LINE TO BE
01321    1 07 00001      SUB    1,1           DELETED, ROUNDING UP IF IT ENDS
01322    0 03 00415      ANA    B16T          ON A HALF WORD BOUNDRY
01323    0 04 00062      STA    STT2          X
01324    1 06 00001      ADD    1,1           GET ADDRESS OF FIRST WORD OF
01325   0404 77          LGR    1             NEXT STATEMENT
01326    0 07 00417      SUB    C1
01327    0 04 00064      STA    STT4          IT IS FIRST SOURCE ADDRESS-1 FOR TABLE MOVE
01330    1 02 00001      LDA    1,1           GET ADDRESS OF FIRST WORD OF STATEMENT
01331   0404 77          LGR    1             IT IS FIRST DESTINATION ADDRESS FOR
01332    0 04 00065      STA    STT5          TABLE MOVE
01333    0 02 00064 STO4 LDA    STT4          ANY MORE TEXT TO
01334    0 11 00021      CAS    PTH           BE MOVED¢
01335   000000           OCT    0             NEVER CAN EXECUTE THIS WORD
01336    0 01 01344      JMP    STO3          NO...HOLE IN TABLE IS FILLED
01337    0 12 00064      IRS    STT4          GET ADDRESS OF NEXT SOURCE WORD
01340   -0 02 00064.     LDA*   STT4          MOVE NEXT WORD
```

```
1198 01341  -0 04 00065      STA*  STT5          X
1199 01342   0 12 00065      IRS   STT5          BUMP DESTINATION ADDRESS POINTER
1200 01343   0 01 01333      JMP   ST04          GO CHECK FOR COMPLETION
1201 01344   0 02 00065 ST03 LDA   STT5          UPDATE PROGRAM TEXT HIGH POINTER
1202 01345   0 04 00021      STA   PTH           X
1203 01346   0 15 00065      STX   STT5          SAVE INDEX AS CLRT DESTROYS IT
1204 01347   0 10 02734      JST   CLRT          ACCOUNT FOR THE SPACE WE OPENED UP
1205 01350   0 35 00065      LDX   STT5          RESTORE INDEX AS CLRT HAS DONE ITS THING
1206                        *
1207                        *     UPDATE STATEMENT INDEX POINTERS
1208                        *
1209                        *          ALL POINTERS TO STATEMENTS THAT WERE PHYSICALLY
1210                        *     LOCATED ABOVE THE STATEMENT THAT WAS DELETED MUST BE
1211                        *     ADJUSTED TO POINT TO THE NEW LOCATION OF THE STATEMENTS.
1212                        *
1213 01351   0 02 00032      LDA   SIB           START SCAN AT BASE OF TABLE
1214 01352   0 11 00033 ST06 CAS   SIT           PAST END OF INDEX ¢
1215 01353   0 01 01367      JMP   ST05          YES...STATEMENT INDEX IS UPDATED
1216 01354   000000          OCT   0             NEVER CAN EXECUTE THIS WORD
1217 01355   141206          ADA                 GET POINTER TO SBP OF CURRENT ENTRY
1218 01356   0 04 00064      STA   STT4          X
1219 01357  -0 02 00064      LDA*  STT4          DOES SBP IN CURRENT ENTRY POINT TO A
1220 01360   1 11 00001      CAS   1,1           STATEMENT ABOVE THE DELETED ONE¢
1221 01361   0 07 00062      SUB   STT2          YES...CORRECT THE POINTER
1222 01362   101000          NOP                 X
1223 01363  -0 04 00064      STA*  STT4          SET CORRECT POINTER VALUE
1224 01364   0 02 00064      LDA   STT4          UPDATE STATEMENT INDEX POINTER
1225 01365   141206          ADA                 X
1226 01366   0 01 01352      JMP   ST06          CONTINUE SCAN
1227                        *
1228                        *     DELETION/REPLACEMENT CHECK
1229                        *
1230 01367   0 02 00061 ST05 LDA   STT1          RESTORE POINTER TO THE NEW
1231 01370   0 04 00037      STA   SBP           LINE
1232 01371   0 10 03020      JST   XCHR          DOES A CARRIAGE RETURN
1233 01372   0 05 00433      ERA   C215          IMMEDIATELY FOLLOW THE
1234 01373   100040          SZE                 LINE NUMBER¢
1235 01374   0 01 01413      JMP   ST07          NO...GO APPEND A STATEMENT
1236                        *
1237                        *     HERE TO DELETE STATEMENT INDEX ENTRY
1238                        *
1239 01375   1 02 77777 ST08 LDA   -1,1          MOVE ENTRY BELOW CURRENT
1240 01376   1 04 00001      STA   1,1           ENTRY INTO POSITION OF
1241 01377   1 02 77776      LDA   -2,1          THE CURRENT ENTRY
1242 01400   1 04 00000      STA   0,1           X
1243 01401   0 02 00000      LDA   0             HAVE ALL ENTRIES BELOW THE
1244 01402   0 11 00032      CAS   SIB           DELETED ENTRY BEEN MOVED UP¢
1245 01403   100000          SKP                 NO...KEEP GOING
1246 01404   0 01 01410      JMP   ST09          YES...WRAP UP
1247 01405   0 07 00426      SUB   C2            DECREMENT THE TABLE POINTER
```

```
1248 01406   0 04 00000       STA   0              X
1249 01407   0 01 01375       JMP   ST08           CONTINUE WITH TABLE COMPRESSION
1250 01410   0 12 00032 ST09  IRS   SIB            CORRECT THE STATEMENT INDEX
1251 01411   0 12 00032       IRS   SIB            BASE POINTER
1252 01412   0 01 01433       JMP   ST10           GO CLOSE OUT THIS OPERATION
1253                          *
1254                          *     ADD STATEMENT TEXT TO TOP OF PROGRAM TEXT TABLE
1255                          *
1256 01413   0 02 00021 ST07  LDA   PTH            PUT BYTE POINTER TO NEW STATEMENT IN
1257 01414   0414 77          LGL   1              SECOND WORD OF THE
1258 01415   1 04 00001       STA   1,1            STATEMENT INDEX ENTRY
1259 01416   0 02 00040       LDA   DBP            CALCULATE NUMBER OF
1260 01417   0 07 00037       SUB   SBP            WORDS IN THE STATEMENT
1261 01420   0 07 00417       SUB   C1             X
1262 01421   0404 77          LGR   1              X
1263 01422   140401           CMA                  X
1264 01423   0 04 00061       STA   STT1           SAVE FOR COUNTING
1265 01424   140407           TCA                  MAKE SURE THERE IS ENOUGH FREE
1266 01425   0 10 03005       JST   UFSC           SPACE FOR THE STATEMENT
1267 01426   0 10 03043 ST11  JST   GCPK          MOVE TWO CHARACTERS TO
1268 01427  -0 04 00021       STA*  PTH            THE PROGRAM TEXT TABLE
1269 01430   0 12 00021       IRS   PTH            BUMP THE TABLE POINTER
1270 01431   0 12 00061       IRS   STT1           BUMP THE WORD COUNTER
1271 01432   0 01 01426       JMP   ST11           MORE TO MOVE
1272                          *
1273                          *     HERE FOR FINAL WRAP UP
1274                          *
1275 01433   0 04 00060 ST10  STA   SEQI           SET EXECUTION INHIBITION TRIGGER
1276 01434   0 01 01045       JMP   CLER           GO PROCESS NEXT COMMAND LINE
1277                          *
1278                          *     HERE FOR NEW STATEMENT ADDITION
1279                          *
1280 01435   0 02 00426 ST01  LDA   C2             MAKE SURE THAT THEREIS ROOM FOR
1281 01436   0 10 03005       JST   UFSC           A NEW STATEMENT INDEX ENTRY
1282 01437   0 35 00032       LDX   SIB            START SEARCH TO FIND POSITION OF THIS ENTRY
1283 01440   0 02 00000 ST13  LDA   0              ARE WE AT THE END OF THE TABLE
1284 01441   0 11 00033       CAS   SIT            X
1285 01442   0 01 01456       JMP   ST12           YES...ADD ENTRY TO TOP OF TABLE
1286 01443   000000           OCT   0              NEVER CAN EXECUTE THIS WORD
1287 01444   1 02 00000       LDA   0,1            COMPARE NUMBER OF THIS ENTRY
1288 01445   0 11 00057       CAS   SNUM           WITH NUMBER OF LINE TO BE INSERTED
1289 01446   0 01 01456       JMP   ST12           NEW ENTRY GOES JUST BEFORE THIS ENTRY
1290 01447   000000           OCT   0              NEVER CAN EXECUTE THIS WORD
1291 01450   1 04 77776       STA   -2,1           NEW ENTRY GOES ABOVE THIS...MOVE THIS
1292 01451   1 02 00001       LDA   1,1            ENTRY INTO VACATED POSITION
1293 01452   1 04 77777       STA   -1,1           BELOW IT.
1294 01453   0 12 00000       IRS   0              BUMP TABLE POINTER TO NEXT ENTRY
1295 01454   0 12 00000       IRS   0              X
1296 01455   0 01 01440       JMP   ST13           GO CHECK IT
1297 01456   0 02 00000 ST12  LDA   0              SET POINTER TO ENTRY
```

```
1298 01457   0 07 00426      SUB   C2          BEING CREATED
1299 01460   0 04 00000      STA   0           X
1300 01461   0 02 00057      LDA   SNUM        STATEMENT NUMBER GOES INTO
1301 01462   1 04 00000      STA   0,1         THE FIRST WORD OF THE ENTRY
1302 01463   0 02 00032      LDA   SIB         UPDATE STATEMENT INDEX BASE
1303 01464   0 07 00426      SUB   C2          POINTER TO ACCOUNT FOR
1304 01465   0 04 00032      STA   SIB         THE NEW ENTRY
1305 01466   0 01 01367      JMP   ST05        GO ADD LINE TO TEXT STORAGE
1306                    *
1307                    *
1308                         EJCT
```

```
1309                    *          USER INPUT PROCESSOR
1310                    *
1311                    *
1312                    *    CALLING SEQUENCE:
1313                    *
1314                    *          LDA    CHAR          INPUT REQUEST CHAR...SEE BELOW
1315                    *          JST    ILIN
1316                    *          DAC    BUF+BUF       55 WORD BUFFER ... SEE BELOW
1317                    *          ......RETURN
1318                    *
1319                    *               THIS ROUTINE WILL PRINT THE CHARACTER CONTAINED
1320                    *     IN THE A REGISTER ON ENTRY AND THEN INPUT CHARACTERS FROM
1321                    *     THE ASR UNTIL A CARRIAGE RETURN IS RECEIVED.  IF A
1322                    *     LEFT ARROW CHARACTER IS RECEIVED, THE LAST CHARACTER PLACED
1323                    *     IN THE BUFFER WILL BE DELETED.  IF A COMMERIAL AT
1324                    *     SIGN IS RECEIVED, A CARRIAGE RETURN/LINE FEED WILL BE ISSUED,
1325                    *     AND THE INPUT LINE WILL BE RESTARTED.
1326                    *               AFTER THE CARRIAGE RETURN HAS BEEN RECEIVED,
1327                    *     A LEXICAL SCAN OF THE INPUT LINE WILL BE PERFORMED.
1328                    *     ALL RESERVED IDENTIFIERS IN THE LINE WILL BE REPLACED
1329                    *     BY THEIR ONE BYTE EQUIVILANTS, AND ALL CONSTANTS WILL
1330                    *     BE CONVERTED TO BINARY AND PLACED IN THE BUFFER PRECEEDED
1331                    *     BY A ONE BYTE CONSTANT TYPE IDENTIFIER (INTEGER OR REAL).
1332                    *               THE EXTRA 18 WORDS OF BUFFER SPACE IS REQUIRED
1333                    *     FOR CASES WHEN THE BINARY VALUE OF A CONSTANT REQUIRES
1334                    *     MORE SPACE THAN IT'S ASCII EQUIVILANT.  THE ASCII SOURCE
1335                    *     IS PLACED IN WORDS 19-55 OF THE BUFFER, AND THE COMPRESSED
1336                    *     TEXT IS STORED STARTING IN THE FIRST WORD OF THE BUFFER.
1337                    *               ALL CHARACTER CODES LESS THAN '215 OR GREATER
1338                    *     THAN '337 ARE IGNORED BY THIS ROUTINE AND ARE NOT
1339                    *     PLACED IN THE BUFFER.
1340                    *
1341                    *
1342                    *
1343 01467   0 000000   ILIN DAC   **
1344 01470   000201          IAB                    PREFIX CHARACTER TO B
1345 01471  -0 02 01467      LDA*  ILIN             GET WORD ADDRESS OF
1346 01472   0404 77         LGR   1                41ST WORD OF
1347 01473   0 06 00472      ADD   C50              PROVIDED BUFFER
1348 01474   0 04 01477      STA   *+3              X
1349 01475   000201          IAB                    RETRIEVE THE PREFIX CHARACTER
1350 01476   0 10 00000      JST   IPUT             INPUT LINE
1351 01477   0 000000        DAC   **               X
1352                    *
1353                    *    HERE FOR LEXICAL SCAN OF NEW LINE
1354                    *
1355 01500  -0 02 01467      LDA*  ILIN             SET POINTER TO ACTUAL START OF BUFFER
1356 01501   0 04 00040      STA   DBP              X
1357 01502   0 10 03071 IL32 JST   GNBC             GET NEXT NON-BLANK CHARACTER
1358 01503   0 11 00440      CAS   C242             IS IT START OF TEXT STRING C
```

```
1359 01504   100000          SKP                NO
1360 01505   0 01 01525      JMP   IL05         YES...GO PACK IT WITHOUT COMPRESSION
1361 01506   0 11 00447      CAS   C256         IS IT A DECIMAL POINT ?
1362 01507   100000          SKP                NO
1363 01510   0 01 01536      JMP   IL06         YES...GO CONVERT A NUMBER
1364 01511   0 10 03130      JST   NUMC         IS IT A DIGIT (0-9) ?
1365 01512   100000          SKP                NO
1366 01513   0 01 01536      JMP   IL06         YES...GO CONVERT A NUMBER
1367 01514   0 10 03121      JST   ALFA         IS IT A ALPHABETIC CHARACTER
1368 01515   100000          SKP                NO
1369 01516   0 01 01701      JMP   IL07         YES...GO SEE IF IT'S A RESERVED WORD
1370 01517   0 10 03077 IL22 JST   SCHR         PUT THE CHARACTER IN THE BUFFER
1371 01520   0 05 00433      ERA   C215         TEST FOR END OF LINE
1372 01521   100040          SZE                X
1373 01522   0 01 01502      JMP   IL32         NO...GO PROCESS NEXT CHARACTER
1374 01523   0 12 01467      IRS   ILIN         YES...INCREMENT RETURN ADDRESS
1375 01524  -0 01 01467      JMP*  ILIN         AND EXIT
1376                         *
1377                         *     HERE TO PACK LITERAL TEXT STRING
1378                         *
1379 01525   0 10 03077 IL05 JST   SCHR         PLACE THE CHARACTER IN THE BUFFER
1380 01526   0 10 03013      JST   GCHR         GET THE NEXT CHARACTER
1381 01527   0 10 03137      JST   DLCK         ; AND C/R NOT ALLOWED IN A TEXT STRING
1382 01530   100000          SKP                OK
1383 01531   0 01 01754      JMP   IL08         ERROR...UNCLOSED TEXT STRING
1384 01532   0 11 00440      CAS   C242         TEST FOR END OF STRING
1385 01533   0 01 01525      JMP   IL05         NO...CONTINUE PACKING
1386 01534   0 01 01517      JMP   IL22         YES...REVERT TO NORMAL MODE
1387 01535   0 01 01525      JMP   IL05         NO....CONTINUE PACKING
1388                         *
1389                         *     HERE TO CONVERT CONSTANTS
1390                         *
1391 01536   0 02 00115 IL06 LDA   LCHR         IT LAST CHARACTER WAS ALPHABETIC,
1392 01537   0 10 03121      JST   ALFA         THEN THE CURRENT DIGIT
1393 01540   0 01 01543      JMP   *+3          IS PART OF A VARIABLE NAME
1394 01541   0 02 00114      LDA   CHAR         GO STORE PART OF A NAME
1395 01542   0 01 01517      JMP   IL22.        X
1396                         *
1397 01543   0 10 03031      JST   UCHR         BACK UP OVER FIRST CHAR OF NUMBER
1398 01544   0 02 00515      LDA   M1           INITIALIZE DECIMAL POINT POSITION FLAG
1399 01545   0 04 00077      STA   ILT2         X
1400 01546   140040          CRA                CLEAR DECIMAL POINT DETECTED FLAG
1401 01547   0 04 00076      STA   ILT1         X
1402 01550   0 04 00041      STA   CVAL         CLEAR NUMERIC ACCUMULATOR
1403 01551   0 04 00042      STA   CVAL+1       X
1404 01552   0 10 03071 IL10 JST   GNBC         GET NEXT NON-BLANK CHARACTER
1405 01553   0 10 03130      JST   NUMC         TEST FOR NUMERIC
1406 01554   0 01 01570      JMP   IL11         NO....GO CHECK FOR SPECIAL CHARACTER
1407 01555   0 07 00451      SUB   C260         CONVERT DIGIT TO BINARY
1408 01556   0 10 00000      JST   FINT         FLOAT IT
```

```
1409 01557   0 10 00000       JST   AS22          ADD TO PREVIOUS ACCUMULATION
1410 01560   0 000041          DAC   CVAL          X
1411 01561   0 10 00000       JST   MS22          UPDATE THE SUMMATION
1412 01562   0 000501          DAC   F10           X
1413 01563   0 10 03153       JST   SCVL          SAVE THE NEW RESULT
1414 01564   0 02 00076       LDA   ILT1          UPDATE DECIMAL POINT LOCATION COUNTER
1415 01565   0 06 00077       ADD   ILT2          X
1416 01566   0 04 00077       STA   ILT2
1417 01567   0 01 01552       JMP   IL10          GO PROCESS NEXT CHARACTER
1418                     *
1419 01570   0 11 00447 IL11  CAS   C256          TEST FOR DECIMAL POINT
1420 01571   100000           SKP                 NO
1421 01572   0 01 01627       JMP   IL12          YES...GO RECORD ITS LOCATION
1422 01573   0 05 00461       ERA   C305          TEST FOR 'E' (START OF EXPONENT)
1423 01574   100040           SZE                 X
1424 01575   0 01 01642       JMP   IL13          NO...GO CLOSE UP
1425                     *
1426                     *     HERE TO PROCESS EXPONENT
1427                     *
1428 01576   0 04 00100       STA   ILT3          CLEAR EXPONENT SIGN FLAG
1429 01577   0 13 00076 IL15  IMA   ILT1          CLEAR EXPONENT VALUE, GET DECIMAL POINT FLAG
1430 01600   101040           SNZ                 IF NO DEC PNT, THEN EXPONENT NOT ALLOWED
1431 01601   0 01 01642       JMP   IL13          THIS 'E' MUST BE PART OF SOMETHING ELSE   NOT NCSS.!
1432 01602   0 10 03071 IL31  JST   GNBC          SEE IF EXPLICIT EXPONENT SIGN
1433 01603   0 11 00444       CAS   C253          A '+' ¢
1434 01604   100000           SKP                 NO
1435 01605   0 01 01612       JMP   IL39          YES ... IGNORE IT
1436 01606   0 11 00446       CAS   C255          A '-' ¢
1437 01607   100000           SKP                 NO
1438 01610   0 01 01624       JMP   IL14          YES ... SET FLAG FOR LATER USE
1439 01611   0 10 03031       JST   UCHR          NO SIGN ... CHR IS PART OF SOMETHING ELSE
1440 01612   0 10 03071 IL39  JST   GNBC          GET CHARACTER
1441 01613   0 10 03130       JST   NUMC          IS IT NUMERIC DIGIT ¢
1442 01614   0 01 01635       JMP   IL16          NO....WE'VE HIT THE END OF THE EXPONENT
1443 01615   0 07 00451       SUB   C260          YES...CONVERT TO INTEGER
1444 01616   0 13 00076       IMA   ILT1          SWAP WITH PREVIOUS ACCUMULATION
1445 01617   0 10 00000       JST   MS11          UPDATE PREVIOUS SUMMATION
1446 01620   0 000423          DAC   C12           X
1447 01621   0 06 00076       ADD   ILT1          INSERT NEW DIGIT
1448 01622   0 04 00076       STA   ILT1          SAVE NEW ACCUMULATION
1449 01623   0 01 01612       JMP   IL39          CONTINUE SCAN
1450                     *
1451 01624   0 02 00515 IL14  LDA   M1            SET NEGITIVE EXPONENT FLAG
1452 01625   0 04 00100       STA   ILT3          X
1453 01626   0 01 01612       JMP   IL39          NOW GET EXPONENT VALUE
1454                     *
1455 01627   0 02 00515 IL12  LDA   M1            SET DECIMAL POINT DETECTED FLAG
1456 01630   0 13 00076       IMA   ILT1          X
1457 01631   101040           SNZ                 WAS IT ALREADY SET ¢
1458 01632   0 01 01552       JMP   IL10          NO....OK!
```

```
1459 01633   0 10 05206       JST   ERR              YES...REPORT ERROR
1460 01634   142320           BCI   1,DP             TOO MANY DECIMAL POINTS
1461                         *
1462                         *   HERE TO CLOSE OUT EXPONENT PROCESSING
1463                         *
1464 01635   0 02 00076  IL16 LDA   ILT1             GET EXPONENT VALUE
1465 01636   0 12 00100       IRS   ILT3             TEST FOR NEGITIVE EXPONENT
1466 01637   100000          SKP                    NO
1467 01640   140407          TCA                    YES...NEGATE THE EXPONENT
1468 01641   100000          SKP                    GO FINISH UP
1469                         *
1470                         *   HERE TO CLOSE OUT CONSTANT CONVERSION
1471                         *
1472 01642   140040     IL13 CRA                    ZERO EXPONENT BY DEFAULT
1473 01643   0 06 00077      ADD   ILT2             PUT IN DECIMAL POINT DISPLACEMENT
1474 01644   0 10 00000      JST   FINT             FLOAT THE RESULT
1475 01645   0 10 00000      JST   H$22             SAVE THE TRUE EXPONENT
1476 01646   0 000043        DAC   LVAL             X
1477 01647   0 10 00000      JST   L$22             RAISE 10 TO THE PROPER POWER
1478 01650   0 000501        DAC   F10              X
1479 01651   0 10 00000      JST   E$22             X
1480 01652   0 000043        DAC   LVAL             X
1481 01653   0 10 00000      JST   M$22             GET TRUE VALUE OF THE CONSTANT
1482 01654   0 000041        DAC   CVAL             X
1483 01655   0 10 00000      JST   TINT             IS THE NUMBER AN INTEGER ?
1484 01656   0 01 01676      JMP   IL17             YES...GO STORE IT
1485                         *
1486 01657   0 04 00041      STA   CVAL             NO...STORE REAL NUMBER
1487 01660   0 02 00511      LDA   RELF             PLACE REAL CONSTANT FLAG IN BUFFER
1488 01661   0 10 03077      JST   SCHR
1489 01662   0 02 00041      LDA   CVAL             GET FIRST WORD OF CONSTANT
1490 01663   141340          ICA                    POSITION FIRST BYTE
1491 01664   0 10 03077      JST   SCHR             STORE IT
1492 01665   141340          ICA                    POSITION 2ND BYTE
1493 01666   0 10 03077 IL18 JST   SCHR             STORE IT
1494 01667   000201          IAB                    GET NEXT WORD OF VALUE
1495 01670   141340          ICA                    POSITION FIRST BYTE
1496 01671   0 10 03077      JST   SCHR             STORE IT
1497 01672   141340          ICA                    POSITON SECOND BYTE
1498 01673   0 10 03077      JST   SCHR             STORE IT
1499 01674   0 10 03031      JST   UCHR             BACK UP OVER UNPROCESSED CHARACTER
1500 01675   0 01 01502      JMP   IL32             GO CONTINUE SCAN
1501                         *
1502 01676   000201     IL17 IAB                    SAVE INTEGER VALUE IN B
1503 01677   0 02 00510      LDA   INTF             PUT INTEGER CONSTANT FLAG IN TEXT
1504 01700   0 01 01666      JMP   IL18             GO PUT VALUE IN TEXT
1505                         *
1506                         *
1507                         *   HERE TO CHECK FOR RESERVED IDENTIFIER
1508                         *
```

```
1509 01701   0 10 03031 IL07 JST   UCHR          BACK UP OVER LEADING CHARACTER
1510 01702   0 02 00514       LDA   LSBP          SET POINTER TO START OF
1511 01703   0 04 00100       STA   ILT3          RESERVED IDENTIFIER LIST
1512 01704   0 02 00037       LDA   SBP           SAVE POINTER TO CURRENT POSITION IN
1513 01705   0 04 00076       STA   ILT1          THE SOURCE STREAM
1514 01706   0 35 00527       LDX   M53           SET COUNTER TO - NO. OF ENTRIES IN LIST
1515 01707   0 02 00037 IL20 LDA   SBP           SWAP SOURCE POINTER WITH RESERVED
1516 01710   0 13 00100       IMA   ILT3          NAME LIST POINTER
1517 01711   0 04 00037       STA   SBP           X
1518 01712   0 10 03013       JST   GCHR          GET NEXT CHAR FROM RESERVED NAME LIST
1519 01713   0 13 00037       IMA   SBP           SWAP BACK THE POINTERS LEAVING
1520 01714   0 13 00100       IMA   ILT3          THE CHAR IN A
1521 01715   0 13 00037       IMA   SBP           X
1522 01716   101040           SNZ                 HAS A NAME BEEN RECOGNIZED ¢
1523 01717   0 01 01741       JMP   IL21          YES#####
1524 01720   0 04 00077       STA   ILT2          SAVE TARGET CHAR WHILE GETTING SOURCE CHAR
1525 01721   0 10 03071       JST   GNBC          GET THE NEXT SOURCE CHAR (WE IGNORE SPACES)
1526 01722   0 05 00077       ERA   ILT2          COMPARE WITH TARGET
1527 01723   101040           SNZ                 DO THEY MATCH ¢
1528 01724   0 01 01707       JMP   IL20          YES...CONTINUE CHECK
1529                    *
1530                    *     RUNOUT NON-MATCHING LIST ENTRY
1531                    *
1532 01725   0 02 00100       LDA   ILT3          SET BYTE POINTER TO NEXT
1533 01726   0 04 00037       STA   SBP           BYTE IN NON-MATCHING NAME
1534 01727   0 10 03013       JST   GCHR          GET NEXT CHARACTER OF IT
1535 01730   100040           SZE                 END OF THE NAME ¢
1536 01731   0 01 01727       JMP   *-2           NO...KEEP TRYING
1537 01732   0 02 00076       LDA   ILT1          RESTORE POINTER TO START OF NAME
1538 01733   0 13 00037       IMA   SBP           IN SOURCE STREAM
1539 01734   0 04 00100       STA   ILT3          SAVE POINTER TO NEXT NAME IN TARGET LIST
1540 01735   0 12 00000       IRS   0             BUMP COUNT OF IDENTIFIERS SKIPPED
1541 01736   0 01 01707       JMP   IL20          STILL MORE TO CHECK!
1542 01737   0 10 03013       JST   GCHR          NOT A RESERVED NAME...STORE
1543 01740   0 01 01517       JMP   IL22          THE CHARACTER LITERALLY
1544                    *
1545                    *     HERE WHEN RESERVED NAME IS RECOGNIZED
1546                    *
1547 01741   0 02 00000 IL21 LDA   0             GET LIST POSITION OF
1548 01742   0 06 00474       ADD   C54           THE IDENTIFIER
1549 01743   0 11 00552       CAS   REMF          IS IT START OF A REMARK ¢
1550 01744   0 01 01517       JMP   IL22          NO...GO STORE THE IDENTIFIER CODE
1551 01745   100000           SKP                 YES...NO COMPRESSION UNTIL NEXT : OR C/R
1552 01746   0 01 01517       JMP   IL22          NO....GO STORE THE IDENTIFIER CODE
1553                    *
1554                    *     HERE TO RUN THROUGH A REMARK STATEMENT
1555                    *
1556 01747   0 10 03077       JST   SCHR          STORE CHARACTER OF THE REMARK
1557 01750   0 10 03013       JST   GCHR          GET THE NEXT CHARACTER
1558 01751   0 10 03137       JST   DLCK          END OF REMARK ¢
```

```
1559 01752   0 01 01747       JMP   *-3              NO...CONTINUE SCAN
1560 01753   0 01 01517       JMP   IL22             YES...REVERT TO NORMAL COMPRESSION
1561                     *
1562 01754   0 10 05206 IL08  JST   ERR              UNCLOSED TEXT STRING
1563 01755   152330          BCI   1,TX             X
1564                     *
1565                     *
1566                     *
1567                           EJCT
```

```
1568                    *          RESERVED IDENTIFIER LIST
1569                    *
1570                    *
1571                    *       THE FOLLOWING LIST CONTAINS ALL RESERVED
1572                    *    ID ENTFIERS PERMITTED IN BASIC-16. WHEN THESE
1573                    *    ID ENTIERS ARE DETECTED IN THE SOURCE STREAM,
1574                    *    THEY ARE REPLACED BY A SINGLE BYTE THAT CONTAINS
1575                    *    THE POSITION IN THE TABLE OF THE PARTICULAR IDENTIFIER.
1576                    *    THE FOLLOWING TABLE GIVES THE SEQUENCE OF THE IDENTIFIERS
1577                    *    IN THE LIST:
1578                    *
1579                    *        1  01. LET      14 12. RETURN    23. SIN     34. FN
1580                    *        2  02. READ     15 13. CALL      24. COS     35. TAB(
1581                    *        3  03. INPUT    16 14. REM       25. TAN     36. JOB
1582                    *        4  04. RESTORE  17 15. STOP      26. ATN     37. CLEAR
1583                    *        5  05. PRINT    20 16. END       27. EXP     38. RUN
1584                    *        6  06. GOTO        17. DATA      28. ABS     39. LIST
1585                    *        7  07. IF          18. DIM       29. LOG     40. CONTINUE
1586                    *       10  08. ON          19. DEF       30. SQR     41. QUIT
1587                    *       11  09. FOR         20. TO        31. INT     42. LOAD
1588                    *       12  10. NEXT        21. STEP      32. RND     43. PUNCH
1589                    *       13  11. GOSUB       22. THEN      33. SGN
1590                    *
1591                    *
1592 01756   146305   IDNT HEX   CCC5          LE
1593 01757   152000        HEX   D400          T@
1594 01760   151305        HEX   D2C5          RE
1595 01761   140704        HEX   C1C4          AD
1596 01762   000311        HEX   00C9          @I
1597 01763   147320        HEX   CED0          NP
1598 01764   152724        HEX   D5D4          UT
1599 01765   000322        HEX   00D2          @R
1600 01766   142723        HEX   C5D3          ES
1601 01767   152317        HEX   D4CF          TO
1602 01770   151305        HEX   D2C5          RE
1603 01771   000320        HEX   00D0          @P
1604 01772   151311        HEX   D2C9          RI
1605 01773   147324        HEX   CED4          NT
1606 01774   000307        HEX   00C7          @G
1607 01775   147724        HEX   CFD4          OT
1608 01776   147400        HEX   CF00          O@
1609 01777   144706        HEX   C9C6          IF
1610 02000   000317        HEX   00CF          @O
1611 02001   147000        HEX   CE00          N@
1612 02002   143317        HEX   C6CF          FO
1613 02003   151000        HEX   D200          R@
1614 02004   147305        HEX   CEC5          NE
1615 02005   154324        HEX   D8D4          XT
1616 02006   000307        HEX   00C7          @G
1617 02007   147723        HEX   CFD3          OS
```

```
1618 02010   152702          HEX   D5C2          UB
1619 02011   000322          HEX   00D2          cR
1620 02012   142724          HEX   C5D4          ET
1621 02013   152722          HEX   D5D2          UR
1622 02014   147000          HEX   CE00          N¢
1623 02015   141701          HEX   C3C1          CA
1624 02016   146314          HEX   CCCC          LL
1625 02017   000322          HEX   00D2          cR
1626 02020   142715          HEX   C5CD          EM
1627 02021   000323          HEX   00D3          cS
1628 02022   152317          HEX   D4CF          TO
1629 02023   150000          HEX   D000          P¢
1630 02024   142716          HEX   C5CE          EN
1631 02025   142000          HEX   C400          D¢
1632 02026   142301          HEX   C4C1          DA
1633 02027   152301          HEX   D4C1          TA
1634 02030   000304          HEX   00C4          cD
1635 02031   144715          HEX   C9CD          IM
1636 02032   000304          HEX   00C4          cD
1637 02033   142706          HEX   C5C6          EF
1638 02034   000324          HEX   00D4          cT
1639 02035   147400          HEX   CF00          O¢
1640 02036   151724          HEX   D3D4          ST
1641 02037   142720          HEX   C5D0          EP
1642 02040   000324          HEX   00D4          cT
1643 02041   144305          HEX   C8C5          HE
1644 02042   147000          HEX   CE00          N¢
1645 02043   151711          HEX   D3C9          SI
1646 02044   147000          HEX   CE00          N¢
1647 02045   141717          HEX   C3CF          CO
1648 02046   151400          HEX   D300          S¢
1649 02047   152301          HEX   D4C1          TA
1650 02050   147000          HEX   CE00          N¢
1651 02051   140724          HEX   C1D4          AT
1652 02052   147000          HEX   CE00          N¢
1653 02053   142730          HEX   C5D8          EX
1654 02054   150000          HEX   D000.         P¢
1655 02055   140702          HEX   C1C2          AB
1656 02056   151400          HEX   D300          S¢
1657 02057   146317          HEX   CCCF          LO
1658 02060   143400          HEX   C700          G¢
1659 02061   151721          HEX   D3D1          SQ
1660 02062   151000          HEX   D200          R¢
1661 02063   144716          HEX   C9CE          IN
1662 02064   152000          HEX   D400          T¢
1663 02065   151316          HEX   D2CE          RN
1664 02066   142000          HEX   C400          D¢
1665 02067   151707          HEX   D3C7          SG
1666 02070   147000          HEX   CE00          N¢
1667 02071   143316          HEX   C6CE          FN
```

```
1668 02072   000324          HEX   00D4      cT
1669 02073   140702          HEX   C1C2      AB
1670 02074   124000          HEX   A800      (c
1671 02075   145317          HEX   CACF      JO
1672 02076   141000          HEX   C200      Bc
1673 02077   141714          HEX   C3CC      CL
1674 02100   142701          HEX   C5C1      EA
1675 02101   151000          HEX   D200      Rc
1676 02102   151325          HEX   D2D5      RU
1677 02103   147000          HEX   CE00      Nc
1678 02104   146311          HEX   CCC9      LI
1679 02105   151724          HEX   D3D4      ST
1680 02106   000303          HEX   00C3      cC
1681 02107   147716          HEX   CFCE      ON
1682 02110   152311          HEX   D4C9      TI
1683 02111   147325          HEX   CED5      NU
1684 02112   142400          HEX   C500      Ec
1685 02113   150725          HEX   D1D5      QU
1686 02114   144724          HEX   C9D4      IT
1687 02115   000314          HEX   00CC      cL
1688 02116   147701          HEX   CFC1      OA
1689 02117   142000          HEX   C400      Dc
1690 02120   150325          HEX   D0D5      PU
1691 02121   147303          HEX   CEC3      NC
1692 02122   144000          HEX   C800      Hc
1693                 *
1694                 *
1695                 EJCT
```

```
1696                    *       FLOATING POINT OUTPUT EDITOR
1697                    *
1698                    *
1699                    *   CALLING SEQUENCE:
1700                    *
1701                    *       LDA   CHAR        SURPRESSION CHARACTER...SEE BELOW
1702                    *       JST   PCVL
1703                    *       ......RETURN
1704                    *
1705                    *       THE FLOATING POINT VALUE CONTAINED IN THE ACCUMULATOR
1706                    *   CVAL IS PRINTED IN THE FOLLOWING FORMAT:
1707                    *
1708                    *       1)  IF CVAL = 0 OR .1<=ABS(CVAL)<10000
1709                    *
1710                    *               SXXXX.XXXXXX
1711                    *
1712                    *       2)  IF NOT IN ABOVE RANGE
1713                    *
1714                    *               S.XXXXXXESYY
1715                    *
1716                    *   LEADING AND TRAILING ZEROS ARE REPLACED WITH SPACES.  THE
1717                    *   SIGN IS FLOATED RIGHT TO THE LEFT OF THE FIRST SIGNIFICANT
1718                    *   DIGIT OR THE DECIMAL POINT, WHICHEVER OCCURS FIRST.  IF IN
1719                    *   FORMAT 1 AND THE FRACTIONAL PART IS ZERO, THE DECIMAL
1720                    *   POINT WILL BE REPLACED WITH
1721                    *   A SPACE.  IF IN FORMAT 2, THE EXPONENT FIELD WILL BE
1722                    *   FLOATED LEFT TO THE RIGHT OF THE LAST NONZERO DIGIT.  THE
1723                    *   SIGN WILL BE PRINTED AS ' ' IS CVAL > 0, OR '-' IF CVAL < 0.
1724                    *   WHEN PRINTING, EACH CHARACTER TO BE OUTPUT WILL BE COMPARED
1725                    *   WITH THE INITIAL A REGISTER CONTENTS.  IF THEY MATCH, THE
1726                    *   CHARACTER WILL NOT BE PRINTED.  THIS WILL NORMALLY BE USED
1727                    *   TO SURPRESS SPACES FOR A PACKED LISTING.
1728                    *
1729                    *
1730                    *
1731 02123   0 000000   PCVL  DAC   **
1732 02124   0 04 00124       STA   INHC          SAVE THE SURPRESSION CHARACTER
1733 02125   0 35 00523       LDX   M21           FILL THE WORK
1734 02126   0 02 00451       LDA   C260          AREA WITH ASCII
1735 02127   1 04 00277       STA   WORK+'21,1    ZEROS ('261)
1736 02130   0 12 00000       IRS   0             X
1737 02131   0 01 02127       JMP   *-2           X
1738 02132   0 04 00123       STA   ECTR          INITIALIZE CNTR FOR EXP CONVERSION
1739 02133   140040           CRA                 CLEAR EXPONENT STORAGE IN
1740 02134   0 04 00122       STA   EXP           CASE CVAL=0
1741 02135   0 10 03147       JST   LCVL          GET VALUE TO BE PRINTED
1742 02136   0 04 00121       STA   SIGN          SAVE ORIGINAL SIGN
1743 02137   100400           SPL                 GET ABSOULUTE VALUE OF IT
1744 02140   0 10 00000       JST   N$22          X
1745 02141   101040           SNZ                 IF VALUE IS ZERO,
```

```
1746 02142   0 01 02222       JMP   ED01      DO NOT GO THROUGH BREAKDOWN LOOP
1747 02143   0 10 03153       JST   SCVL      SAVE THE VALUE
1748 02144   0 02 00530       LDA   M6        SET UP LOOP TO EXTRACT
1749 02145   0 04 00116       STA   TMP1      SIX DIGITS
1750 02146   0 02 00551       LDA   WKD7      POINTER TO 7TH WORD OF WORK AREA
1751 02147   0 04 00117       STA   TMP2      SIX ASCII DIGITS TO WORK(6-11.
1752 02150   0 10 00000       JST   LOGF      GET LOG BASE 10 OF VALUE
1753 02151   0 000041         DAC   CVAL      FOR EXPONENT
1754 02152   0 10 00000       JST   D$22      X
1755 02153   0 000512         DAC   LE10      =LN(10)
1756 02154   0 10 00000       JST   IFLT      INTERGERIZE BASE 10 EXPONENT
1757 02155   000000           OCT   0         OVERFLOW IS IMPOSSIBLE
1758 02156   141206           ADA             ADJUST IT SO THINGS COME OUT RIGHT
1759 02157   0 04 00122       STA   EXP       AND SAVE IT.
1760 02160   0 07 00417       SUB   C1        RESTORE TO UNADJUSTED VALUE
1761 02161   0 10 00000       JST   FINT      FLOAT IT
1762 02162   0 10 00000       JST   H$22      USE IT TO REDUCE CVAL
1763 02163   0 000043         DAC   LVAL      TO RANGE OF 1 TO 10 BY
1764 02164   0 10 00000       JST   L$22      CVAL=CVAL/10.**A, WHERE
1765 02165   0 000501         DAC   F10       A IS INT(ALOG10(CVAL))
1766 02166   0 10 00000       JST   E$22      X
1767 02167   0 000043         DAC   LVAL      X
1768 02170   0 10 00000       JST   H$22      X
1769 02171   0 000043         DAC   LVAL      X
1770 02172   0 10 03147       JST   LCVL      X
1771 02173   0 10 00000       JST   D$22      X
1772 02174   0 000043         DAC   LVAL      X
1773 02175   0 10 00000       JST   A$22      ADD IN .000005 TO ROUND UP
1774 02176   0 000532         DAC   ROND      X
1775 02177   0 10 03153 ED02  JST   SCVL      SAVE WORKING VALUE
1776 02200   0 10 00000       JST   IFLT      EXTRACT INTEGER PART
1777 02201   000000           OCT   0         OVERFLOW IS IMPOSSIBLE
1778 02202   0 04 00120       STA   TMP3      SAVE FOR LATER REFERENCE
1779 02203   0 11 00421       CAS   C11       SEE IF WE ROUNDED FROM 9.999... TO 10
1780 02204   0 01 02367       JMP   ED10      YES...SPECIAL ACTION REQUIRED
1781 02205   101000           NOP             NO PROBLEM
1782 02206   0 06 00451 ED11  ADD   C260      CONVERT TO ASCII
1783 02207  -0 04 00117       STA*  TMP2      PLACE DIGIT IN WORK AREA
1784 02210   0 02 00120       LDA   TMP3      RETRIEVE CURRENT DIGIT
1785 02211   0 10 00000       JST   FINT      FLOAT IT
1786 02212   0 10 00000       JST   S$22      EXTRACT IT FROM CVAL
1787 02213   0 000041         DAC   CVAL      X
1788 02214   0 10 00000       JST   N$22      X
1789 02215   0 10 00000       JST   M$22      POSITION NEXT DIGIT
1790 02216   0 000501         DAC   F10       X
1791 02217   0 12 00117       IRS   TMP2      BUMP POINTER TO WORK AREA
1792 02220   0 12 00116       IRS   TMP1      BUMP ITERATION COUNTER
1793 02221   0 01 02177       JMP   ED02      CONTINUE EXTRACTION.
1794                  *
1795 02222   0 10 02374 ED01  JST   FRNG      SEE IF IT WILL FIT IN F FORMAT
```

```
1796 02223   0 01 02235       JMP   ED03          NO
1797 02224   140407           TCA                 GET NO. OF PLACES TO SHIFT THE DIGIT FIELD
1798 02225   0 06 00551       ADD   WKD7          THIS GIVES DESTINATION ADDRESS
1799 02226   0 04 00116       STA   TMP1          SAVE IT
1800 02227   0 35 00520       LDX   M12           MOVE 10 DIGITS LEFT EXP POSITIONS
1801 02230   1 02 00276       LDA   WORK+*20,1    SHIFT A DIGIT
1802 02231  -0 04 00116       STA*  TMP1          X
1803 02232   0 12 00116       IRS   TMP1          BUMP THE DESTINATION POINTER
1804 02233   0 12 00000       IRS   0             BUMP THE SOURCE POINTER
1805 02234   0 01 02230       JMP   *-4           GO MOVE NEXT DIGIT
1806 02235   0 35 00525 ED03  LDX   M5            MAKE ROOM FOR THE DECIMAL POINT
1807 02236   1 02 00264       LDA   WORK+6,1      BY MOVING WORK(1-5) TO
1808 02237   1 04 00263       STA   WORK+5,1      WORK(0-4)
1809 02240   0 12 00000       IRS   0             BUMP THE COUNTER
1810 02241   0 01 02236       JMP   *-3           MORE TO MOVE
1811 02242   0 02 00447       LDA   C256          INSERT THE DECIMAL POINT
1812 02243   0 04 00263       STA   WORK+5        X
1813 02244   0 35 00525       LDX   M5            ZERO SURPRESS FROM LEFT
1814 02245   1 02 00263 ED05  LDA   WORK+5,1      PICK UP DIGIT FROM ARRAY
1815 02246   0 05 00451       ERA   C260          TEST FOR '0'
1816 02247   100040           SZE                 X
1817 02250   0 01 02255       JMP   ED04          FOUND 1ST NON '0' CHAR
1818 02251   0 02 00436       LDA   C240          REPLACE LEADING ZERO
1819 02252   1 04 00263       STA   WORK+5,1      WITH A SPACE
1820 02253   0 12 00000       IRS   0             SEE IF WE'RE DONE
1821 02254   0 01 02245       JMP   ED05          NO...CONTINUE SCAN
1822 02255   0 02 00121 ED04  LDA   SIGN          INSET SIGN IN THE ARRAY
1823 02256   0 10 02404       JST   SGN           X
1824 02257   1 04 00262       STA   WORK+4,1      X
1825 02260   0 35 00427       LDX   C20           START TRAILING ZERO SURPRESSION
1826 02261   1 02 00256 ED07  LDA   WORK,1        PICK UP DIGIT FROM HIGH END OF ARRAY
1827 02262   0 05 00451       ERA   C260          TEST FOR '0'
1828 02263   100040           SZE                 X
1829 02264   0 01 02273       JMP   ED06          FOUND LAST NONZERO CHAR
1830 02265   0 02 00436       LDA   C240          REPLACE TRAILING ZERO WITH A
1831 02266   1 04 00256       STA   WORK,1        SPACE
1832 02267   0 02 00000       LDA   0             STEP BACK ONE CHARACTER
1833 02270   0 07 00417       SUB   C1            X
1834 02271   0 04 00000       STA   0             X
1835 02272   0 01 02261       JMP   ED07          GO LOOK AT NEXT CHARACTER
1836 02273   1 02 00256 ED06  LDA   WORK,1        SEE IF EVERYTHING PAST
1837 02274   0 05 00447       ERA   C256          THE DECIMAL POINT HAS
1838 02275   100040           SZE                 BEEN SURPRESSED
1839 02276   0 01 02301       JMP   *+3           NO
1840 02277   0 02 00436       LDA   C240          YES...SURPRESS THE DECIMAL
1841 02300   1 04 00256       STA   WORK,1        POINT ALSO
1842 02301   1 02 00255       LDA   WORK-1,1      TEST FOR ALL BLANK BUFFER (VALUE ZERO)
1843 02302   0 11 00436       CAS   C240          X
1844 02303   100000           SKP                 NO
1845 02304   0 02 00451       LDA   C260          YES...LEAVE ONE ZERO
```

```
1846 02305   1 04 00255        STA   WORK-1,1        REPLACE THE CHARACTER
1847 02306   0 10 02374        JST   FRNG            IS 'E' PROCESSING REQUIRED ?
1848 02307   100000            SKP                   YES
1849 02310   0 01 02336        JMP   ED08            NO...SKIP THIS
1850 02311   0 10 02404        JST   SGN             GET SIGN OF EXPONENT
1851 02312   1 04 00260        STA   WORK+2,1        X
1852 02313   0 02 00461        LDA   C305            PUT IN 'E'
1853 02314   1 04 00257        STA   WORK+1,1        X
1854 02315   0 02 00122        LDA   EXP             CONVERT EXPONENT TO ASCII
1855 02316   100400            SPL                   MAKE SURE IT'S PLUS FIRST
1856 02317   140407            TCA                   X
1857 02320   0 07 00423        SUB   C12             DIVIDE BY 10
1858 02321   100400            SPL                   X
1859 02322   0 01 02325        JMP   *+3             X
1860 02323   0 12 00123        IRS   ECTR            X
1861 02324   0 01 02320        JMP   *-4             X
1862 02325   0 06 00453        ADD   C272            REMAINDER IS 2ND DIGIT
1863 02326   1 04 00262        STA   WORK+4,1        OF EXPONENT
1864 02327   0 02 00123        LDA   ECTR            PUT IN FIRST DIGIT
1865 02330   1 04 00261        STA   WORK+3,1        X
1866 02331   0 02 00000        LDA   0               MAKE INDEX POINT TO
1867 02332   0 06 00466        ADD   C4              END OF EXPONENT
1868 02333   0 04 00000        STA   0               X
1869 02334   0 02 00466        LDA   C4              START PRINTING AT 5TH CHAR OF WORK1
1870 02335   100000            SKP                   BUFFER1
1871 02336   140040    ED08    CRA                   1ST CHAR OF WORK1 FOR F FORMAT
1872 02337   0 06 00550        ADD   WRKD            GET POINTER TO 1ST PRINTING CHAR
1873 02340   0 12 00000        IRS   0               X POINTS TO LAST CHAR TO PRINT
1874 02341   0 04 00122 ED09   STA   EXP             SAVE POINTER TO FIRST CHARACTER
1875 02342   0 02 00000        LDA   0               CALCULATE ADDRESS OF LAST CHARACTER
1876 02343   0 06 00550        ADD   WRKD            X
1877 02344   0 04 00123        STA   ECTR            SAVE IT
1878 02345   0 07 00122        SUB   EXP             CALCULATE NUMBER OF CHARACTERS IN STRING
1879 02346   141206            ADA                   X
1880 02347   0 06 00046        ADD   CPOS            CHECK FOR LINE FIT
1881 02350   0 11 00531        CAS   MCOL            X
1882 02351   0 10 00000        JST   LFCR            NO...ADVANCE ASR TO NEXT LINE
1883 02352   101000            NOP                   IT WILL FIT
1884 02353  -0 02 00122 ED12   LDA*  EXP             GET NEXT CHARACTER TO BE PRINTED
1885 02354   0 11 00124        CAS   INHC            SEE IF IT'S SURPRESSED
1886 02355   100000            SKP                   NO
1887 02356   100000            SKP                   YES
1888 02357   0 10 00000        JST   OTA1            NO...PRINT IT
1889 02360   0 02 00122        LDA   EXP             ADVANCE THE CHARACTER POINTER
1890 02361   141206            ADA                   X
1891 02362   0 04 00122        STA   EXP             SAVE ADDRESS OF NEXT CHARACTER
1892 02363   0 11 00123        CAS   ECTR            TEST FOR COMPLETION
1893 02364  -0 01 02123        JMP*  PCVL            YES...EXIT
1894 02365   0 01 02353        JMP   ED12            NO....LOOP BACK1
1895 02366   0 01 02353        JMP   ED12            NO....LOOP BACK1
```

```
1896                          *
1897 02367   0 02 00452 ED10 LDA   C261        INSERT '10' IN WORK1 ARRAY
1898 02370  -0 04 00117       STA*  TMP2       X
1899 02371   0 12 00122       IRS   EXP        BUMP EXPONENT TO ACCOUNT FOR EXTRA DIGIT
1900 02372   101000           NOP              NOP IN CASE EXP GOES TO ZERO
1901 02373   0 01 02222       JMP   ED01       EXTRACTION COMPLETE, NOW FORMAT THE RESULT
1902                          *
1903                          *
1904 02374   0 000000  FRNG DAC   **           TEST FOR F FORMAT
1905 02375   0 02 00122       LDA   EXP        GET EXPONENT
1906 02376   0 11 00515       CAS   M1         EXPONENTS IN RANGE 0-4
1907 02377   0 11 00471       CAS   C5         ARE PRINTED IN F FORMAT
1908 02400  -0 01 02374       JMP*  FRNG       E FORMAT
1909 02401  -0 01 02374       JMP*  FRNG       E FORMAT
1910 02402   0 12 02374       IRS   FRNG       F FORMAT
1911 02403  -0 01 02374       JMP*  FRNG       X
1912                          *
1913                          *
1914 02404   0 000000  SGN  DAC   **           GET SIGN CHARACTER
1915 02405   140320           CSA              SET C IF NO. IS MINUS
1916 02406   0 02 00436       LDA   C240       SPACE FOR PLUS
1917 02407   100001           SRC              TEST SIGN
1918 02410   0 02 00446       LDA   C255       '-' FOR NEGITIVE
1919 02411  -0 01 02404       JMP*  SGN        RETURN
1920                          *
1921                          *
1922                          *
1923                          *
1924                               EJCT
```

```
1925                    *          EXPRESSION ANALYZER
1926                    *
1927                    *
1928                    *     CALLING SEQUENCE:
1929                    *
1930                    *          LDA    PTRG         A CONTAINS PRECEDENCE TRIGGER...SEE BELOW
1931                    *          JST    EXPA
1932                    *          ......RETURN       RESULT IN CVAL
1933                    *
1934                    *          THIS ROUTINE WILL EVALUATE THE EXPRESSION IN THE
1935                    *     SOURCE STREAM UNTIL AN OPERATOR OF THE SAME
1936                    *     OR LOWER PRECEDENCE LEVEL AS INDICATED BY THE
1937                    *     INITIAL A REGISTER CONTENTS IS ENCOUNTERED.
1938                    *     THE FOLLOWING ARE THE DEFINED PRECEDENCE LEVELS:
1939                    *          0 - DELIMITERS          <EXPRESSION>
1940                    *          1 - +,-                 <MULTIPLY FACTOR>
1941                    *          2 - *,/                 <INVOLUTION FACTOR>
1942                    *          3 - ¢                   <TERM>
1943                    *     TO EVALUATE A COMPLETE EXPRESSION, THE INITIAL
1944                    *     A REGISTER VALUE WOULD BE 0.  THE OTHER POSSIBLE
1945                    *     VALUES ARE USED PRIMARILY IN RECURSIVE CALLS TO
1946                    *     THIS ROUTINE MADE WITHIN IT.  THE RESLUT OF THE
1947                    *     EXPRESSION IS LEFT IN THE FLOATING POINT ACCUMULTOR,
1948                    *     CVAL.  THE PREVIOUS CONTENTS OF CVAL ARE LEFT
1949                    *     IN LVAL AND THE A + B REGISTERS ON RETURN.
1950                    *
1951                    *
1952                    *     EXPRESSION SYNTAX:
1953                    *
1954                    *          <EXPRESSION>:=<MULTIPLY FACTOR>¢<SIGN><EXPRESSION>¢
1955                    *                       <EXPRESSION>¤+¢-¤<INVOLUTION FACTOR>
1956                    *
1957                    *          <MULTIPLY FACTOR>:=<INVOLUTION FACTOR>¢<MULTIPLY FACTOR>
1958                    *                          ¤*¢/¤<INVOLUTION FACTOR>
1959                    *
1960                    *          <INVOLUTION FACTOR>:=<TERM>¢<INVOLUTION FACTOR>¤¢¤<TERM>
1961                    *
1962                    *          <TERM>:=<NUMBER>¢<VARIABLE>¢<FUNCTION TERM>¢(<EXPRESSION>)
1963                    *
1964                    *          <FUNCTION TERM>:=<FUNCTION NAME>(<EXPRESSION>)
1965                    *
1966                    *          <FUNCTION NAME>:=SIN¢COS¢TAN¢ATN¢EXP¢ABS¢LOG¢SOR¢INT¢RND¢
1967                    *                          SGN¢FN<ALPHABETIC CHARACTER>
1968                    *
1969                    *          <VARIABLE>:=<SIMPLE VARIABLE>¢<SUBSCRIPTED VARIABLE>
1970                    *
1971                    *
1972                    *
1973                    *
1974 02412    0 000000  EXPA DAC  **
```

```
1975 02413   0 13 00075        IMA   LOP      SET NEW OPERATOR PRECEDENCE TRIGGER
1976 02414   0 10 02766        JST   PUSH     PUT OLD VALUE ON STACK
1977 02415   0 02 02412        LDA   EXPA     SAVE RETURN ADDRESS IN CASE
1978 02416   0 10 02766        JST   PUSH     WE'RE RE-ENTERED
1979 02417   0 02 00041        LDA   CVAL     PUT THE VALUE OF THE
1980 02420   0 10 02766        JST   PUSH     TERM ON
1981 02421   0 02 00042        LDA   CVAL+1   THE STACK
1982 02422   0 10 02766        JST   PUSH     X
1983 02423   0 35 00521        LDX   M2       SET TRIGGER FOR UNARY MINUS PROCESSING
1984 02424   0 10 03013 EX06   JST   GCHR     LOOK AT NEXT SOURCE CHARACTER
1985 02425   0 11 00446        CAS   C255     IS IT '-' ?
1986 02426   100000           SKP            NO
1987 02427   0 01 02443        JMP   EX01     YES...AT THIS POINT, IT MUST BE UNARY
1988 02430   0 12 00000        IRS   0        HAS A UNARY MINUS BEEN DETECTED ?
1989 02431   0 01 02456        JMP   EX04     NO...GO CONTINUE ITEM DISCRIMINATION
1990 02432   0 02 00075        LDA   LOP      UNARY MINUS ONLY LEGAL WHEN
1991 02433   100040           SZE            LOP IS EQUAL TO ZERO
1992 02434   0 10 05206 EX07   JST   ERR      ERROR...ILLEGAL UNARY MINUS
1993 02435   152715           BCI   1,UM     (THIS WILL EXECUTE AS AN ERA)
1994 02436   0 10 00000        JST   L$22     FORCE A '-1.0*' INTO SOURCE
1995 02437   0 000505         DAC   FM1      X
1996 02440   0 10 03153        JST   SCVL     X
1997 02441   0 10 03031        JST   UCHR     PROCESS CURRENT CHARACTER LATER
1998 02442   0 01 02606        JMP   EX05     GO WORK ON '*' THAT WE FORCED
1999                  *
2000 02443   0 12 00000 EX01   IRS   0        BUMP THE MINUS COUNT
2001 02444   0 01 02424        JMP   EX06     OK...THIS IS THE FIRST
2002 02445   0 01 02434        JMP   EX07     TWO IN A ROW...ERROR
2003                  *
2004 02446   0 10 03043 EX02   JST   GCPK     FORM THE INTEGER
2005 02447   0 10 00000        JST   FINT     FLOAT IT
2006 02450   0 01 02512        JMP   EX22     GO SAVE VALUE AND WORK ON OPERATOR
2007                  *
2008 02451   0 10 03043 EX03   JST   GCPK     HIGH WORD OF REAL TO A
2009 02452   000201           IAB            SAVE IT IN B
2010 02453   0 10 03043        JST   GCPK     GET LOW ORDER WORD
2011 02454   000201           IAB            PUT IT IN NORMAL FORM
2012 02455   0 01 02512        JMP   EX22     GO SAVE VALUE AND WORK ON OPERATOR
2013                  *
2014 02456   0 11 00510 EX04   CAS   INTF     TEST FOR INTEGER CONSTANT
2015 02457   100000           SKP            NO
2016 02460   0 01 02446        JMP   EX02     YES...GO FORM IT
2017 02461   0 11 00511        CAS   RELF     TEST FOR FLOATING POINT CONSTANT
2018 02462   100000           SKP            NO
2019 02463   0 01 02451        JMP   EX03     YES...GO FORM IT
2020 02464   0 11 00441        CAS   C250     TEST FOR PARENTHETICAL TERM
2021 02465   100000           SKP            NO
2022 02466   0 01 02547        JMP   EX10     YES...GO EVALUATE IT
2023 02467   0 11 00546        CAS   SYSL     TEST FOR SYSTEM FUNCTION
2024 02470   0 11 00547        CAS   SYSH     ( SUCH AS SIN, COS, ECT.)
```

```
2025 02471   0 01 02474        JMP   *+3              NO
2026 02472   100000           SKP                    NO
2027 02473   0 01 02554        JMP   EX11             YES...GO PROCESS IT
2028 02474   0 11 00545        CAS   DEFF             TEST FOR CALL TO DEFINED FUNCTION
2029 02475   100000           SKP                    NO
2030 02476   0 01 02635        JMP   EX30             YES...GO EVALUATE IT
2031 02477   0 10 03031        JST   UCHR             IT MUST BE A VARIABLE
2032 02500   0 10 04606        JST   PVN              PROCESS THE VARIABLE NAME
2033 02501   0 01 02506        JMP   EX13             GO LOCATE SUBSCRIPTED VARIABLE
2034 02502   0 10 04664        JST   LSV              LOOK UP NAME IN SIMPLE VARIABLE TABLE
2035 02503   0 10 05206 EX15   JST   ERR              ERROR...USED BEFORE ASSIGNED
2036 02504   152726           BCI   1,UV             (THIS EXECUTES AS AN ERA)
2037 02505   0 01 02510        JMP   EX14             HAVE ADDRESS OF VALUE IN X...GO GET IT
2038 02506   0 10 05100 EX13   JST   LDV              PROCESS SUBSCRIPT AND GET ADDR OF VALUE
2039 02507   0 01 02503        JMP   EX15             ERROR...USED BEFORE ASSIGNED
2040 02510   0 10 00000 EX14   JST   L$22             GET VALUE OF VARIABLE
2041 02511   -0 000000        DAC*  0                X CONTAINS THE ADDRESS
2042 02512   0 10 03153 EX22   JST   SCVL             LEAVE VALUE IN CVAL
2043                          *
2044                          *     HERE TO LOOK FOR AN OPERATOR
2045                          *
2046 02513   0 10 03013 EX09   JST   GCHR             GET NEXT CHAR (IT MUST BE AN OPERATOR)
2047 02514   0 11 00444        CAS   C253             TEST FOR '+'
2048 02515   100000           SKP                    NO
2049 02516   0 01 02571        JMP   EX16             YES...GO PROCESS ADDITION OP
2050 02517   0 11 00446        CAS   C255             TEST FOR '-'
2051 02520   100000           SKP                    NO
2052 02521   0 01 02600        JMP   EX17             YES...GO PROCESS SUBTRACTION OP
2053 02522   0 11 00443        CAS   C252             TEST FOR '*'
2054 02523   100000           SKP                    NO
2055 02524   0 01 02606        JMP   EX05             YES...GO PROCESS MULTIPLICATION OP
2056 02525   0 11 00450        CAS   C257             TEST FOR '/'
2057 02526   100000           SKP                    NO
2058 02527   0 01 02614        JMP   EX19             YES...GO PROCESS DIVISION OP
2059 02530   0 11 00464        CAS   C336             TEST FOR '↑'
2060 02531   000000           OCT   0                NEVER CAN EXECUTE THIS WORD
2061 02532   0 01 02622        JMP   EX20             YES...GO PROCESS EXPONENTIATION OP
2062                          *
2063                          *          BY PROCESS OF ELIMINATION, CURRENT CHARACTER
2064                          *     MUST BE A TERMINAL DELIMITER (OR USER SYNTAX ERROR, IN
2065                          *     WHICH CASE THE LAST ZERO LEVEL CALLER WILL DETECT IT).
2066                          *
2067 02533   0 10 03031        JST   UCHR             BACK UP 1 CHAR FOR EASE ELSEWHERE
2068                          *
2069                          *     EXIT PROCESSING
2070                          *
2071 02534   0 10 02775 EX21   JST   POP              LEAVE VALUE THAT WAS PUT ON STACK
2072 02535   0 04 00044        STA   LVAL+1           ON ENTRY IN LVAL
2073 02536   0 10 02775        JST   POP              X
2074 02537   0 04 00043        STA   LVAL             X
```

```
2075 02540    0 10 02775        JST   POP              GET RETURN ADDRESS
2076 02541    0 04 02412        STA   EXPA             X
2077 02542    0 10 02775        JST   POP              GET PRECEDENCE TRIGGER FOR OPERATOR
2078 02543    0 04 00075        STA   LOP              PRECEEDING CURRENT ONE
2079 02544    0 10 00000        JST   L$22             LEAVE LVAL IN A + B
2080 02545    0 000043          DAC   LVAL             X
2081 02546   -0 01 02412        JMP*  EXPA             AND EXIT
2082                           *
2083                           *    EVALUATE PARENTHETICAL TERM
2084                           *
2085 02547    140040       EX10 CRA                   EVALUATE EXPRESSION WITHIN
2086 02550    0 10 02412        JST   EXPA             THE PARENTHESIS
2087 02551    0 02 00442        LDA   C251             MAKE SURE DELIMITER IS ')'
2088 02552    0 10 03050        JST   GCCK             X
2089 02553    0 01 02513        JMP   EX09             NOW WORK ON OPERATOR THAT FOLLOWS
2090                           *
2091                           *    HERE FOR PROCESSING SYSTEM FUNCTION
2092                           *
2093 02554    0 10 02766   EX11 JST   PUSH             CHAR IS FUNCTION IDENTIFIER
2094 02555    0 02 00441        LDA   C250             MAKE SURE '(' AFETER FUNCTION NAME
2095 02556    0 10 03050        JST   GCCK             X
2096 02557    140040            CRA                    EVALUATE THE ARGUMENT
2097 02560    0 10 02412        JST   EXPA             X
2098 02561    0 02 00442        LDA   C251             MAKE SURE IT ENDS WITH A ')'
2099 02562    0 10 03050        JST   GCCK             X
2100 02563    0 10 02775        JST   POP              FETCH THE FUNCTION IDENTIFIER
2101 02564    0 07 00546        SUB   SYSL             REDUCE IT FOR TABLE ACCESSING
2102 02565    0 04 00000        STA   0                X
2103 02566   -1 10 00567        JST*  SFNL,1           EVALAUTE THE FUNCTION
2104 02567    0 000041          DAC   CVAL             ARGUMENT WAS LEFT IN CVAL
2105 02570    0 01 02512        JMP   EX22             GO SAVE CVAL AND CONTINUE
2106                           *
2107                           *    HERE FOR ADDITION OPERATOR
2108                           *
2109 02571    0 02 00417   EX16 LDA   C1               + IS AT PRECEDENCE LEVEL 1
2110 02572    0 10 02630        JST   EXPC             TEST FOR PRECEEDING HIGHER OP
2111 02573    0 10 02412        JST   EXPA             EVALUATE UNTIL = OR < PRTY. OP
2112 02574    0 10 00000        JST   A$22             CVAL=LVAL+CVAL
2113 02575    0 000041          DAC   CVAL             X
2114 02576    0 10 03153   EX24 JST   SCVL             SAVE THE RESULT
2115 02577    0 01 02513        JMP   EX09             CONTINUE
2116                           *
2117                           *    HERE FOR SUBTRACTION OPERATOR
2118                           *
2119 02600    0 02 00417   EX17 LDA   C1               - IS AT PRECEDENCE LEVEL 1
2120 02601    0 10 02630        JST   EXPC             TEST FOR PRECEEDING HIGHER OP
2121 02602    0 10 02412        JST   EXPA             EVALUATE UNTIL = OR < PRTY. OP
2122 02603    0 10 00000        JST   S$22             CVAL=LVAL-CVAL
2123 02604    0 000041          DAC   CVAL             X
2124 02605    0 01 02576        JMP   EX24             GO SAVE RESULT AND CONTINUE
```

```
2125                        *
2126                        *     HERE FOR MULTIPLICATION OPERATOR
2127                        *
2128 02606   0 02 00426 EX05 LDA   C2              * IS AT PRECEDENCE LEVEL 2
2129 02607   0 10 02630      JST   EXPC            TEST FOR PRECEEDING HIGHER OP
2130 02610   0 10 02412      JST   EXPA            EVALUATE UNTIL = OT < PRTY. OP
2131 02611   0 10 00000      JST   M$22            CVAL=LVAL*CVAL
2132 02612   0 000041        DAC   CVAL            X
2133 02613   0 01 02576      JMP   EX24            GO SAVE RESULT AND CONTINUE
2134                        *
2135                        *     HERE FOR DIVISION OPERATOR
2136                        *
2137 02614   0 02 00426 EX19 LDA   C2              / IS AT PRECEDENCE LEVEL 2
2138 02615   0 10 02630      JST   EXPC            TEST FOR PRECEEDING HIGHER OP
2139 02616   0 10 02412      JST   EXPA            EVALAUATE UNTIL = OT < PRTY. OP
2140 02617   0 10 00000      JST   D$22            CVAL=LVAL/CVAL
2141 02620   0 000041        DAC   CVAL            X
2142 02621   0 01 02576      JMP   EX24            SAVE RESULT AND CONTINUE
2143                        *
2144                        *     HERE FOR EXPONENTIATION OPERATOR
2145                        *
2146 02622   0 02 00457 EX20 LDA   C3              ↑ IS AT PRECEDENCE LEVEL 3
2147 02623   0 10 02630      JST   EXPC            TEST FOR PRECEEDING ↑
2148 02624   0 10 02412      JST   EXPA            EVALAUTE UNTIL = OR < PRTY. OP
2149 02625   0 10 00000      JST   E$22            CVAL=LVAL**CVAL
2150 02626   0 000041        DAC   CVAL            X
2151 02627   0 01 02576      JMP   EX24            SAVE THE RESULT AND CONTINUE
2152                        *
2153                        *     PRECEDENCE COMPARISON ROUTINE
2154                        *
2155                        *          IF THE PRECEDENCE OF THE PRECEEDING
2156                        *     OPERATOR IS < THAT OF THE CURRENT OPERATOR,
2157                        *     RETURN IS MADE FOLLOWING THE CALL.  IF NOT,
2158                        *     CONTROL IS TRANSFERED TO THE NORMAL EXPA
2159                        *     EXIT SEQUENCE.
2160                        *
2161 02630   0 000000   EXPC DAC   **
2162 02631   0 11 00075      CAS   LOP             COMPARE THE PRECEDENCE LEVELS
2163 02632  -0 01 02630      JMP*  EXPC            CURRENT OP IS GREATER
2164 02633   0 01 02533      JMP   EX21-1          LAST OP WAS THE SAME
2165 02634   0 01 02533      JMP   EX21-1          LAST OP WAS HIGHER
2166                        *
2167                        *
2168                        *     PROCESS CALL TO PROGRAMMER DEFINED FUNCTION
2169                        *
2170                        *
2171 02635   0 10 04562 EX30 JST   SDFI            LOOK UP FUNCTION NAME IN TABLE
2172 02636   0 10 05206      JST   ERR             ERROR...FUNCTION HAS NOT BEEN DEFINED
2173 02637   152706         BCI   1,UF            (THIS EXECUTES AS AN ERA)
2174 02640   0 02 00441      LDA   C250            '(' MUST FOLLOW FUNCTION NAME
```

```
2175 02641   0 10 03050      JST   GCCK     X
2176 02642   0 02 00000      LDA   0        SAVE TABLE POINTER IN CASE
2177 02643   0 10 02766      JST   PUSH     THERE IS A PDF IN ARGUMENT
2178 02644   140040         CRA             EVALUATE THE FUNCTION ARGUMENT
2179 02645   0 10 02412      JST   EXPA     X
2180 02646   0 02 00442      LDA   C251     MAKE SURE IT'S FOLLOWED BY ')'
2181 02647   0 10 03050      JST   GCCK     X
2182 02650   0 10 02775      JST   POP      FETCH THE TABLE POINTER
2183 02651   0 04 00000      STA   0        X
2184 02652   1 02 00001      LDA   1,1      GET DUMMY ARGUMENT NAME
2185 02653   0 13 00072      IMA   DEFN     SWAP WITH CURRENT DUMMY NAME
2186 02654   0 10 02766      JST   PUSH     LEAVE PREVIOUS NAME ON STACK.
2187 02655   0 02 00041      LDA   CVAL     CVAL IS NEW DUMMY VARIABLE VALUE.
2188 02656   0 13 00073      IMA   DEFV     OLD VALUE GOES ON STACK
2189 02657   0 10 02766      JST   PUSH     X
2190 02660   0 02 00042      LDA   CVAL+1   X
2191 02661   0 13 00074      IMA   DEFV+1   X
2192 02662   0 10 02766      JST   PUSH     X
2193 02663   1 02 00002      LDA   2,1      GET POINTER TO FUNCTION EXPRESSION
2194 02664   0 13 00037      IMA   SBP      SWAP WITH CURRENT EXPRESSION POINTER
2195 02665   0 10 02766      JST   PUSH     LEAVE OLD POINTER ON STACK
2196 02666   140040         CRA             EVALUATE THE FUNCTION
2197 02667   0 10 02412      JST   EXPA     X
2198 02670   0 10 03062      JST   GDLM     IT MUST END IN : OR C/R
2199 02671   0 10 02775      JST   POP      RESTORE POINTER TO ORIGINAL EXPR.
2200 02672   0 04 00037      STA   SBP      X
2201 02673   0 10 02775      JST   POP      RESTORE PREVIOUS DUMMY
2202 02674   0 04 00074      STA   DEFV+1   VARIABLE VALUE
2203 02675   0 10 02775      JST   POP      X
2204 02676   0 04 00073      STA   DEFV     X
2205 02677   0 10 02775      JST   POP      RESTORE PREVIOUS DUMMY
2206 02700   0 04 00072      STA   DEFN     VARIABLE NAME
2207 02701   0 01 02513      JMP   EX09     NOW CONTINUE WITH ORIGINAL EXPRESSION
2208                         *
2209                         *
2210                         *
2211                         EJCT
```

```
2212                        *          PRINT LINE NUMBER
2213                        *
2214                        *
2215                        *    CALLING SEQUENCE:
2216                        *
2217                        *          JST   PLN
2218                        *          ......RETURN
2219                        *
2220                        *          THE LINE NUMBER OF THE CURRENT STATEMENT
2221                        *    (OR ZERO IF EXECUTING AN IMMEADIATE MODE STMT)
2222                        *    IS PRINTED WITH BLANK SURPRESSION.
2223                        *
2224                        *
2225                        *
2226 02702    0 000000    PLN  DAC   **
2227 02703    0 02 00034       LDA   SIP          GET SI POINTER TO CURRENT STMT
2228 02704    100040           SZE                ZERO IF IN IMMEADIATE MODE
2229 02705   -0 02 00034       LDA*  SIP          NOT IMMEADIATE MODE...GET LINE NO. FROM SI
2230 02706    0 10 00000       JST   FINT         FLOAT IT FOR THE OUTPUT EDITOR
2231 02707    0 10 03153       JST   SCVL         IT PRINTS THE FLT PNT NO. IN CVAL
2232 02710    0 02 00436       LDA   C240         SURPRESS BLANKS
2233 02711    0 10 02123       JST   PCVL         PRINT THE NUMBER
2234 02712   -0 01 02702       JMP*  PLN          AND RETURN
2235                        *
2236                        *
2237                        *
2238                             EJCT
```

```
                                *
                                *       INPUT/OUTPUT ROUTINES
                                *
                                *
                                *    TYPE - OUTPUT MESSAGE
                                *
                                *
                                *    CALLING SEQUENCE:
                                *
                                *         JST   TYPE
                                *         DAC   MESG          ADDRESS OF MESSAGE
                                *         ......RETURN        A REGISTER ZERO
                                *
                                *         THIS ROUTINE WILL OUTPUT THE MESSAGE UNTIL A BYTE
                                *    CONTAINING ZERO IS ENCOUNTERED.
                                *
                                *
02713    0 000000   TYPE DAC    **
02714   -0 02 02713      LDA*   TYPE                FETCH MESSAGE POINTER
02715    0414 77         LGL    1                   MAKE IT A BYTE POINTER
02716    0 13 00037      IMA    SBP                 SWAP WITH CURRENT SOURCE BYTE POINTER
02717    0 04 00000      STA    0                   LEAVE SBP IN A SAFE PLACE
02720    0 12 02713      IRS    TYPE                SET PROPER RETURN ADDRESS
02721    0 10 03013 TYP1 JST    GCHR                GET NEXT CHARACTER OF MESSAGE
02722    101040          SNZ                        END OF MESSAGE ?
02723    0 01 02726      JMP    TYP2                YES...GO RESTORE SBP AND EXIT
02724    0 10 00000      JST    OTA1                OUTPUT THE CHARACTER
02725    0 01 02721      JMP    TYP1                CONTINUE
                                *
02726    0 15 00037 TYP2 STX    SBP                 RESTORE SBP
02727   -0 01 02713      JMP*   TYPE                AND EXIT
                                *
                                *
                                *    SPAC - OUTPUT SPACE CHARACTER
                                *
                                *
                                *    CALLING SEQUENCE:
                                *
                                *         JST   SPAC
                                *         ......RETURN
                                *
                                *
02730    0 000000   SPAC DAC    **
02731    0 02 00436      LDA    C240                GET SPACE CODE
02732    0 10 00000      JST    OTA1                OUTPUT IT
02733   -0 01 02730      JMP*   SPAC                RETURN
                                *
                                *
                                *
                                EJCT
```

```
                                *
                                *          TABLE POINTER INITIALIZATION ROUTINE
                                *
                                *
                                *    CALLING SEQUENCE:
                                *
                                *          JST   CLRT
                                *          .....RETURN
                                *
                                *          THE FOLLOWING OPERATIONS ARE PERFORMED BY
                                *    THIS ROUTINE:
                                *
                                *          1)  THE DEFINED FUNCTION INDEX, THE FOR-NEXT STACK,
                                *    THE SIMPLE VARIABLE TABLE, THE DIMENSIONED VARIABLE TABLE,
                                *    AND THE RETURN STACK ARE CLEARED.
                                *
                                *          2)  THE 'CONTINUE' POINTER IS RESET
                                *
                                *          3)  THE READ STATEMENT DATA POINTER IS INITIALIZED.
                                *
                                *          4)  THE AMOUT OF FREE SPACE IS CALCULATED.
                                *
                                *
                                *
02734    0 000000   CLRT DAC    **
02735    0 35 00516      LDX    M10              CLEAR DFI, FN, SV,
02736   140040           CRA                     AND DV TABLES.
02737    1 04 00032      STA    DFB+10,1         X
02740    0 12 00000      IRS    0                X
02741    0 01 02737      JMP    *-2              X
02742    0 04 00125      STA    CON1             CLEAR 'CONTINUE' POINTER
02743    0 07 00032      SUB    SIB              CALUCLATE NO. OF WORDS
02744    0 06 00021      ADD    PTH              OF USER SPACE ARE AVAILABLE
02745    0 06 00426      ADD    C2               ALLOW A SLIGHT MARGIN
02746   101400           SMI                     ANY WORDS AVAILABLE ?
02747    0 01 02773      JMP    MEMO             .....NO..... REPORT MEMORY OVERFLOW
02750    0 04 00050      STA    FSC              SET THE COUNTER
02751    0 02 00534      LDA    RTB              INITIALIZE THE RETURN STACK
02752    0 04 00036      STA    RTP              X
02753    0 10 04156      JST    REST             GO INITIALIZE DATA STMT POINTERS
02754   -0 01 02734      JMP*   CLRT             RETURN
                                *
                                *
                                *
                                EJCT
```

```
2332                    *       INITIALIZE PUSH DOWN STACK!
2333                    *
2334                    *
2335                    *   CALLING SEQUENCE:
2336                    *
2337                    *       JST   IPDS
2338                    *       ......RETURN
2339                    *
2340                    *       THE PUSH DOWN STACK POINTER, PDSP, IS SET TO THE
2341                    *   FIRST FREE WORD ABOVE THE TABLES THAT EXTEND FROM THE TOP
2342                    *   OF THE COMPILER.
2343                    *
2344                    *
2345                    *
2346 02755    0 000000   IPDS DAC   **
2347 02756    0 02 00025      LDA   FNT             USE FOR-NEXT TABLE TOP
2348 02757    101040          SNZ                   UNLESS IT IS EMPTY, IN WHICH CASE
2349 02760    0 02 00023      LDA   DFT             USE DEFINED FUNCTION TABLE TOP
2350 02761    101040          SNZ                   UNLESS IT IS EMPTY, IN WHICH CASE
2351 02762    0 02 00021      LDA   PTH             USE PROGRAM TEXT TOP.
2352 02763    141206          ADA                   +1 TO GET ADDRESS OF FREE WORD
2353 02764    0 04 00035      STA   PDLP            SET THE POINTER
2354 02765   -0 01 02755      JMP*  IPDS            AND RETURN.
2355                    *
2356                    *
2357                    *
2358                         EJCT
```

```
2359                        *          PUSH DOWN STACK HANDLING ROUTINES
2360                        *
2361                        *
2362                        *    PUSH - ADD ENTRY TO STACK
2363                        *
2364                        *
2365                        *    CALLING SEQUENCE:
2366                        *
2367                        *          LDA    WORD         A CONTAINS WORD TO BE ADDED TO STACK
2368                        *          JST    PUSH
2369                        *          ......RETURN        UNLESS STACK OVERFLOW
2370                        *
2371                        *
2372 02766    0 000000     PUSH DAC    **
2373 02767   -0 04 00035          STA*   PDLP              PLACE ENTRY IN THE STACK
2374 02770    0 12 00035          IRS    PDLP              UPDATE THE STACK POINTER
2375 02771    0 12 00050          IRS    FSC               UPDATE THE FREE STORAGE COUNT
2376 02772   -0 01 02766          JMP*   PUSH              NO OVERFLOW...RETURN
2377 02773    0 10 05206   MEMO   JST    ERR               REPORT MEMORY OVERFLOW
2378 02774    146717              BCI    1,MO              X
2379                        *
2380                        *
2381                        *    POP  - REMOVE ENTRY FROM THE PUSH DOWN STACK
2382                        *
2383                        *
2384                        *    CALLING SEQUENCE85
2385                        *
2386                        *          JST    POP
2387                        *          ......RETURN        TOP STACK ENTRY IN A
2388                        *
2389                        *
2390 02775    0 000000     POP  DAC    **
2391 02776    0 02 00035          LDA    PDLP              DECREMENT THE STACK POINTER
2392 02777    0 07 00417          SUB    C1                X
2393 03000    0 04 00035          STA    PDLP              X
2394 03001    0 02 00515          LDA    M1                UPDATE THE FREE STORAGE COUNT
2395 03002    0 10 03005          JST    UFSC              X
2396 03003   -0 02 00035          LDA*   PDLP              REMOVE TOP ENTRY FROM THE STACK
2397 03004   -0 01 02775          JMP*   POP               AND RETURN
2398                        *
2399                        *
2400                        *
2401                              EJCT
```

```
2402                        *          UPDATE FREE STORAGE COUNT
2403                        *
2404                        *
2405                        *     CALLING SEQUENCE:
2406                        *
2407                        *          LDA    NWRD         A CONTAINS NO. OF WORDS REQUIRED
2408                        *          JST    UFSC
2409                        *          ......RETURN       IF REQUIRED NO. OF WORDS ARE AVAILABLE
2410                        *
2411                        *          THIS ROUTINE MAY BE USED TO REMOVE WORDS FROM THE
2412                        *     FREE STORAGE AREA (IF A IS +) OR TO RELEASE WORDS TO
2413                        *     THE FREE STORAGE AREA (IF A IS -).
2414                        *
2415                        *
2416 03005    0 000000    UFSC DAC    **
2417 03006    0 06 00050       ADD    FSC              UPDATE THE FREE STORAGE COUNT
2418 03007    101400           SMI                     TEST FOR MEMORY OVERFLOW
2419 03010    0 01 02773       JMP    MEMO             YES...GO TELL THE USER
2420 03011    0 04 00050       STA    FSC              NO....SAVE THE UPDATED COUNT
2421 03012   -0 01 03005       JMP*   UFSC             AND RETURN.
2422                        *
2423                        *
2424                        *
2425                             EJCT
```

```
                    *          CHARACTER HANDLING ROUTINES
                    *
                    *     GCHR - GET NEXT SOURCE CHARACTER
                    *
                    *     CALLING SEQUENCE:
                    *
                    *          JST   GCHR
                    *          ......RETURN
                    *
                    *          THE NEXT SOURCE CHARACTER IS LEFT IN
                    *     THE A REGISTER AND CHAR, AND THE SOURCE
                    *     BYTE POINTER, SBP, IS INCREMENTED.
                    *
                    *
03013    0 000000   GCHR  DAC   **
03014    0 10 03020       JST   XCHR            GET CHARACTER POINTED TO BY SBP
03015    0 04 00114       STA   CHAR            LEAVE IT IN CHAR
03016    0 12 00037       IRS   SBP             BUMP THE SOUCE BYTE POINTER
03017   -0 01 03013       JMP*  GCHR            AND RETURN
                    *
                    *
                    *     XCHR - EXAMINE NEXT SOURCE CHARACTER
                    *
                    *
                    *     CALLING SEQUENCE:
                    *
                    *          JST   XCHR
                    *          ......RETURN
                    *
                    *          THE CHARACTER POINTED TO BY SBP IS RETURNED
                    *     IN THE A REGISTER, SBP, AND CHAR ARE NOT ALTERED.
                    *
                    *
03020    0 000000   XCHR  DAC   **
03021    0 02 00037       LDA   SBP             GET THE BYTE POINTER
03022    0404 77          LGR   1               /2 TO GET WORD NO., BYTE INDICATOR IN C
03023    0 04 00116       STA   TMP1            X
03024   -0 02 00116       LDA*  TMP1            PICK UP WORD CONTAINING BYTE WE WANT
03025    101001           SSC                   SKIP IF LOW ORDER BYTE IS REQUESTED
03026    141340           ICA                   HIGH ORDER BYTE...POSITION IT
03027    141050           CAL                   ISOLATE THE BYTE
03030   -0 01 03020       JMP*  XCHR            RETURN
                    *
                    *
                    *     UCHR - BACK UP ONE CHARACTER
                    *
                    *
                    *     CALLING SEQUENCE:
```

```
2476                    *
2477                    *           JST   UCHR
2478                    *           ......RETURN
2479                    *
2480                    *           THE SOURCE BYTE POINTER, SBP, IS DECREMENTED
2481                    *     BY ONE.
2482                    *
2483                    *
2484 03031    0 000000    UCHR: DAC   **
2485 03032    0 13 00037        IMA   SBP               DECREMENT THE SOURCE BYTE
2486 03033    0 07 00417        SUB   C1                POINTER BY ONE
2487 03034    0 13 00037        IMA   SBP               AND REPLACE IT
2488 03035   -0 01 03031        JMP*  UCHR              RETURN
2489                    *
2490                    *
2491                    *     GCHX - EXCLUSIVE OR NEXT CHARACTER TO A
2492                    *
2493                    *
2494                    *     CALLING SEQUENCE:
2495                    *
2496                    *           JST   GCHX
2497                    *           ......RETURN
2498                    *
2499                    *           THE NEXT CHARACTER IS XOR'ED TO A AND
2500                    *     THE RESULT IS LEFT IN A.  SBP IS INCREMENTED
2501                    *     AND THE CHARACTER IS LEFT IN CHAR.
2502                    *
2503                    *
2504 03036    0 000000    GCHX  DAC   **
2505 03037    0 04 00117        STA   TMP2              SAVE A WHILE GETTING CHARACTER
2506 03040    0 10 03013        JST   GCHR              GET THE NEXT SOURCE CHARACTER
2507 03041    0 05 00117        ERA   TMP2              XOR PREVIOUS A TO IT
2508 03042   -0 01 03036        JMP*  GCHX              RETURN
2509                    *
2510                    *
2511                    *     GCPKI - PACK NEXT TWO SOURCE CHARACTERS
2512                    *
2513                    *
2514                    *     CALLING SEQUENCE:
2515                    *
2516                    *           JST   GCPK
2517                    *           ......RETURN
2518                    *
2519                    *           THE NEXT TWO CHARACTERS ARE FETCHED AND
2520                    *     LEFT IN A (FIRST CHARACTER IN A(1-8), SECOND
2521                    *     CHARACTER IN A(9-16)).
2522                    *
2523                    *
2524 03043    0 000000    GCPKI DAC   **
2525 03044    0 10 03013        JST   GCHR              GET THE FIRST CHARACTER
```

```
2526 03045   141240          ICR                     PUT IT A(1-8)
2527 03046   0 10 03036      JST   GCHX              PUT NEXT CHARACTER IN A(9-16)
2528 03047   -0 01 03043     JMP*  GCPK              RETURN
2529                        *
2530                        *
2531                        *    GCCK - GET CHARACTER AND CHECK
2532                        *
2533                        *
2534                        *    CALLING SEQUENCE:
2535                        *
2536                        *          LDA   CHAR      A CONTAINS CHARACTER EXPECTED
2537                        *          JST   GCCK
2538                        *          ......RETURN    IF CHAR FOUND (A ZERO)
2539                        *
2540                        *          THE NEXT CHARACTER IS FETCHED, AND IF IT
2541                        *    DOES NOT MATCH THE CONTENTS OF A, A 'MX' ERROR
2542                        *    IS REPORTED, WHERE X IS THE CHARACTER IN A ON ENTRY.
2543                        *
2544                        *
2545 03050   0 000000       GCCK  DAC   **
2546 03051   0 05 03061     GCC   ERA   GCC1            FORM ERROR DIAGNOSTIC
2547 03052   0 04 03060     STA   GCC2            SAVE IT IN CASE IT'S NEEDED
2548 03053   0 05 03061     ERA   GCC1            RESTORE A
2549 03054   0 10 03036     JST   GCHX            XOR NEXT CHARACTER WITH IT
2550 03055   101040         SNZ                   A ZERO IF THEY MATCH
2551 03056   -0 01 03050    JMP*  GCCK            MATCH!!!!...RETURN
2552 03057   0 10 05206     JST   ERR             DIFFERENT...REPORT ERROR
2553 03060   146730         GCC2  BCI   1,MX            X
2554                        *
2555 03061   146400         GCC1  HEX   CD00            ASCII M IN BITS 1-8, ZERO IN BITS 9-16
2556                        *
2557                        *
2558                        *    GDLM - GET TERMINAL DELIMITER
2559                        *
2560                        *
2561                        *    CALLING SEQUENCE:
2562                        *
2563                        *          JST   GDLM
2564                        *          ......RETURN    IF NEXT CHARACTER IS : OR C/R
2565                        *
2566                        *          THE NEXT CHARACTER IS FETCHED AND IF IT IS
2567                        *    EITHER A : OR C/R, RETURN IS MADE WITH THE
2568                        *    DELIMITER CODE IN A.  IF IT IS NOT A : OR C/R,
2569                        *    A 'DL' ERROR IS REPORTED.
2570                        *
2571                        *
2572 03062   0 000000       GDLM  DAC   **
2573 03063   0 10 03013     JST   GCHR            GET THE NEXT CHARACTER
2574 03064   0 10 03137     JST   DLCK            TEST FOR : OR C/R
2575 03065   100000         SKP                   NO....ERROR
```

```
2576 03066  -0 01 03062      JMP*  GDLM              YES...RETURN
2577 03067   0 10 05206      JST   ERR               REPORT THE ERROR
2578 03070   142314          BCI   1,DL              X
2579                     *
2580                     *
2581                     *    GNBC - GET NEXT NON-BLANK CHARACTER
2582                     *
2583                     *
2584                     *    CALLING SEQUENCE:
2585                     *
2586                     *          JST   GNBC
2587                     *          ......RETURN      CHARACTER IN A AND CHAR
2588                     *
2589                     *          THE NEXT NONUBLANK CHARACTER (<> '240) IS
2590                     *    FETCHED AND LEFT IN A AND CHAR.
2591                     *
2592                     *
2593                     *
2594 03071   0 000000   GNBC DAC   **
2595 03072   0 10 03013      JST   GCHR              GET THE NEXT CHARACTER
2596 03073   0 11 00436      CAS   C240              TEST FOR BLANK
2597 03074  -0 01 03071      JMP*  GNBC              NO...RETURN
2598 03075   0 01 03072      JMP   *-3               YES...TRY NEXT CHARACTER
2599 03076  -0 01 03071      JMP*  GNBC              NO...RETURN
2600                     *
2601                     *
2602                     *
2603                     *    SCHR - STORE CHARACTER
2604                     *
2605                     *
2606                     *    CALLING SEQUENCE:
2607                     *
2608                     *          LDA   CHAR        A CONTAINS CHARACTER TO BE STORED
2609                     *          JST   SCHR
2610                     *          ......RETURN      A UNCHANGED
2611                     *
2612                     *          THE CHARACTER IN A IS TRUNCATED TO 8 BITS AND LEFT
2613                     *    IN THE LOCATION LCHR.  IT IS THEN INSERTED IN THE BYTE
2614                     *    POINTED TO BY DBP WITHOUT DESTROYING THE OTHER BYTE IN
2615                     *    THE TARGET WORD.   DBP IS INCREMENTED FOLLOWING THE
2616                     *    INSETION.  THE INITIAL A REGISTER CONTENTS ARE RETURNED
2617                     *    UNCHANGED.
2618                     *
2619                     *
2620 03077   0 000000   SCHR DAC   **
2621 03100   0 04 00117      STA   TMP2              SAVE INITIAL A CONTENTS
2622 03101   141050          CAL                     TRUNCATE TO 8 BITS
2623 03102   0 04 00115      STA   LCHR              SAVE FOR LATER REFERENCE
2624 03103   0 02 00040      LDA   DBP               GET POINTER TO TARGET BYTE
2625 03104   0404 77         LGR   1                 WORD NO. IN A, BYTE INDICATOR IN C
```

```
2626 03105   0 04 00116      STA   TMP1        SAVE TARGET WORD POINTER
2627 03106  -0 02 00116      LDA*  TMP1        GET CURRENT CONTENTS OF TARGET WORD
2628 03107   101001           SSC              PUT TARGET BYTE IN A(9-16)
2629 03110   141340           ICA              X
2630 03111   141044           CAR              CLEAR POSITION FOR NEW BYTE
2631 03112   0 05 00115      ERA   LCHR        INSERT THE NEW CHARACETR
2632 03113   101001           SSC              REPOSITION THE BYTES IF NECESSARY
2633 03114   141340           ICA              X
2634 03115  -0 04 00116      STA*  TMP1        RESTORE THE BYTE PAIR
2635 03116   0 12 00040      IRS   DBP         INCREMENT THE DESTINATION BYTE POINTER
2636 03117   0 02 00117      LDA   TMP2        RESTORE A
2637 03120  -0 01 03077      JMP*  SCHR        AND EXIT
2638                    *
2639                    *
2640                    EJCT
```

```
2641                        *          CHARACTER CHECK ROUTINES
2642                        *
2643                        *
2644                        *    ALFA. - ALPHABETIC CHARACTER TEST ROUTINE
2645                        *
2646                        *
2647                        *    CALLING SEQUENCE:
2648                        *
2649                        *          LDA    CHAR         A CONTAINS CHARACTER TO BE TESTED
2650                        *          JST    ALFA
2651                        *          ......RETURN       IF NOT ALPHABETIC CHARACTER
2652                        *          ......RETURN       IF ALPHABETIC CHARACTER
2653                        *
2654                        *
2655 03121    0 000000    ALFA DAC    **
2656 03122    0 11 00460       CAS    C300               LOW ALPHABETCI RANGE - 1
2657 03123    0 11 00463       CAS    C333               HIGH ALPHABETIC TANGE + 1
2658 03124   -0 01 03121       JMP*   ALFA               RETURN 1 ... NOT ALPHABETIC
2659 03125   -0 01 03121       JMP*   ALFA               RETURN 1 ... NOT ALPHABETIC
2660 03126    0 12 03121       IRS    ALFA               ALPHABETIC ... TAKE SECOND RETURN
2661 03127   -0 01 03121       JMP*   ALFA               X
2662                        *
2663                        *
2664                        *    NUMC - NUMERIC CHARACTER TEST ROUTINE
2665                        *
2666                        *
2667                        *    CALLING SEQUENCE:
2668                        *
2669                        *          LDA    CHAR         A CONTAINS CHARACTER TO BE TESTED
2670                        *          JST    NUMC
2671                        *          ......RETURN       IF NOT NUMERIC CHARACTER
2672                        *          ......REOURN       IF NUMERIC CHARACTER
2673                        *
2674                        *
2675 03130    0 000000    NUMC DAC    **
2676 03131    0 11 00450       CAS    C257               LOW NUMERIC RANGE - 1
2677 03132    0 11 00453       CAS    C272               HIGH NUMERIC RANGE + 1
2678 03133   -0 01 03130       JMP*   NUMC               RETURN 1 ... NOT NUMERIC
2679 03134   -0 01 03130       JMP*   NUMC               RETURN 1 ... NOT NUMERIC
2680 03135    0 12 03130       IRS    NUMC               NUMERIC ... TAKE SECOND RETURN
2681 03136   -0 01 03130       JMP*   NUMC               X
2682                        *
2683                        *
2684                        *    DLCKI - TERMINAL DELIMITER TEST ROUTINE
2685                        *
2686                        *
2687                        *    CALLING SEQUENCE:
2688                        *
2689                        *          LDA    CHAR         A CONTAINS CHARACTER TO BE TESTED
2690                        .*         JST    DLCK
```

```
2691                        *          ......RETURN      IF NOT TERMINAL DELIMITER (: OR C/R)
2692                        *          ......RETURN      IF TERMINAL DELIMITER (: OR C/R)
2693                        *
2694                        *
2695 03137   0 000000    DLCK1 DAC    **
2696 03140   0 11 00453         CAS    C272              TEST FOR COLON
2697 03141  -0 01 03137         JMP*   DLCK1             NO....TAKE FIRST RETURN
2698 03142   0 12 03137         IRS    DLCK1             YES...INCREMENT TO TAKE SECOND RETURN
2699 03143   0 11 00433         CAS    C215              TEST FOR CARRIAGE RETURN
2700 03144  -0 01 03137         JMP*   DLCK1             NO....RETURN
2701 03145   0 12 03137         IRS    DLCK1             YES...INCREMENT FOR SECOND RETURN
2702 03146  -0 01 03137         JMP*   DLCK1             RETURN
2703                        *
2704                        *
2705                             EJCT
```

```
2706                        *          FLOATING POINT ACCUMULATOR LOAD AND STORE ROUTINES
2707                        *
2708                        *
2709                        *     LCVLI - LOAD A+B FROM FLOATING POINT ACCUMULATOR
2710                        *
2711                        *
2712                        *     CALLING SEQUENCE:
2713                        *
2714                        *          JST    LCVL
2715                        *          ......RETURN      CONTENTS OF CVALI IN A+B
2716                        *
2717                        *
2718 03147    0 000000   LCVLI DAC    **
2719 03150    0 10 00000       JST    L$22              LOAD A+B FROM CVALI
2720 03151    0 000041         DAC    CVAL              X
2721 03152   -0 01 03147       JMP*   LCVLI             RETURN
2722                        *
2723                        *
2724                        *     SCVLI - STORE A+B IN FLOATING POINT ACCUMULATOR
2725                        *
2726                        *
2727                        *     CALLING SEQUENCE:
2728                        *
2729                        *          JST    SCVL
2730                        *          ......RETURN      CVALI SET TO CONTENTS OF A+B
2731                        *
2732                        *
2733 03153    0 000000   SCVLI DAC    **
2734 03154    0 10 00000       JST    H$22              STORE A+B IN CVALI
2735 03155    0 000041         DAC    CVAL              X
2736 03156   -0 01 03153       JMP*   SCVLI             RETURN
2737                        *
2738                        *
2739                        *
2740                             EJCT   NOT MUCH TO THESE ROUTINES, IS THERE ¢
```

```
2741                    *       EXECUTE NEXT SOURCE STATEMENT
2742                    *
2743                    *
2744                    *       THIS ROUTINE WILL DETERMINE THE TYPE OF THE
2745                    *  STATEMENT POINTED TO BY SBP AND BRANCH TO THE PROPER
2746                    *  STATEMENT PROCESSER.  THE POINTER SIP IS ASSUMED TO
2747                    *  BE SET TO POINT TO THE STATEMENT INDEX ENTRY FOR THE
2748                    *  STATEMENT BEING PROCESSED, UNLESS IT IS AN IMMEADIATE
2749                    *  MODE STATEMENT, IN WHICH CASE SIP SHOULD BE ZERO.
2750                    *  BEFORE THE STATEMENT IS EXECUTED, A CHECK FOR PROGRAM
2751                    *  IS MADE.  IF A BREAK CHARACTER HAS BEEN ENTERED,
2752                    *  THE MESSAGE 'XXXX BREAK' IS PRINTED, WHERE XXXX IS THE
2753                    *  CURRENT LINE NUMBER, AND THE POINTERS TO THE STATEMENT
2754                    *  ARE SAVED FOR USE BY A FUTURE CONTINUE COMMAND.
2755                    *
2756                    *
2757                    *
2758 03157   140040     ESMT CRA                    INITIALIZE THE DUMMY VARIABLE NAME CELL
2759 03160   0 04 00072       STA   DEFN            X
2760 03161   0 10 00000       JST   BRKC            CHECK FOR PROGRAM BREAK
2761 03162   0 13 00127       IMA   BRKF            GET THE BREAK FLAG, ALSO CLEAR IT
2762 03163   100040           SZE                   X
2763 03164   0 01 03221       JMP   ES01            GO PROCESS PROGRAM BREAK
2764 03165   0 10 02755       JST   IPDS            INITIALIZE THE PUSH DOWN STACK
2765 03166   0 10 03013       JST   GCHR            LOOK AT 1ST CHAR OF STMT TO GET STMT TYPE
2766 03167   0 11 00431       CAS   C23             TEST FOR IMPLIED 'LET' STATEMENT
2767 03170   0 01 03217       JMP   ES02            YES...GO PROCESS IT
2768 03171   101000           NOP                   NO
2769 03172   0 04 00000       STA   0               SET STATEMENT BRANCH LIST POINTER
2770 03173  -1 01 03173       JMP*  *,1             BRANCH TO THE PROPER STATEMENT PROCESSOR
2771                    *
2772 03174   0 003240         DAC   ASNM            LET STATEMENT
2773 03175   0 004044         DAC   READ            READ STATEMENT
2774 03176   0 004040         DAC   INPT            INPUT STATEMENT
2775 03177   0 004153         DAC   RSTR            RESTORE STATEMENT
2776 03200   0 004165         DAC   PRNT            PRINT STATEMENT
2777 03201   0 003270         DAC   GOTO            GOTO STATEMENT
2778 03202   0 003331         DAC   IF              IF STATEMENT
2779 03203   0 003304         DAC   ON              ON STATEMENT
2780 03204   0 003436         DAC   FOR             FOR STATEMENT
2781 03205   0 003566         DAC   NEXT            NEXT STATEMENT
2782 03206   0 003664         DAC   GOSB            GOSUB STATEMENT
2783 03207   0 003704         DAC   RTRN            RETURN STATEMENT
2784 03210   0 003724         DAC   CALL            CALL STATEMENT
2785 03211   0 004401         DAC   REM             REM STATEMENT
2786 03212   0 004403         DAC   EXIT            STOP STATEMENT
2787 03213   0 004403         DAC   EXIT            END STATEMENT
2788 03214   0 004401         DAC   REM             DATA STATEMENT
2789 03215   0 004401         DAC   REM             DIM STATEMENT
2790 03216   0 004301         DAC   DEF             DEF STATEMENT
```

LÄGG MERKT PÅ HAT VÄRDE HÄR FÖR PROG OCH SEDAN JMP HÄR

```
2791                        *
2792 03217   0 10 03031 ES02 JST   UCHR          BACK UP OVER THE FIRST CHAR OF THE STMT
2793 03220   0 01 03240      JMP   ASNM          GO PROCESS ASSIGNMENT STATEMENT
2794                        *
2795                        *     HERE IF PROGRAM BREAK
2796                        *
2797 03221   0 02 00034 ES01 LDA   SIP           DO NOT RECORD BREAK INFO IF IN COMMAND MODE
2798 03222   101040         SNZ                 X
2799 03223   0 01 01000      JMP   CMOD          RETURN TO PROCESS NEXT SYSTEM COMMAND
2800 03224   0 04 00125      STA   CON1          SAVE POINTERS TO INTERRUPT POINT
2801 03225   0 02 00037      LDA   SBP           X
2802 03226   0 04 00126      STA   CON2          X
2803 03227   0 10 00000      JST   LFCR          PRINT 'XXXX BREAK'
2804 03230   0 10 02702      JST   PLN           X
2805 03231   0 10 02713      JST   TYPE          X
2806 03232   0 003234        DAC   BKMS
2807 03233   0 01 01000      JMP   CMOD          RETURN TO COMMAND MODE
2808                        *
2809                        *
2810 03234   120302   BKMS BCI   3, BREAK       BREAK MESSAGE
     03235   151305
     03236   140713
2811 03237   000000         OCT   0             MESSAGE TERMINATOR
2812                        *
2813                        *
2814                        *
2815                        EJCT
```

```
2816                    *       ASSIGNMENT STATEMENT PROCESSOR
2817                    *
2818                    *
2819                    *   STATEMENT SYNTAX:
2820                    *
2821                    *       <ASSIGNMENT STATEMENT>:=¤LET<VARIABLE>=<EXPRESSION>¢
2822                    *                              <VARIABLE>=<EXPRESSION>¤¤:¢C/R¤
2823                    *
2824                    *
2825                    *
2826 03240   140040     ASNM CRA             PUT EOL MARK ON THE STACK
2827 03241   0 10 02766      JST   PUSH      X
2828 03242   0 10 04606 ASN2 JST   PVN       INPUT/CLASSIFY VARIABLE NAME
2829 03243   0 10 04714      JST   ADV       LOCATE/ASSIGN DIMENSIONED VARIABLE
2830 03244   0 10 04634      JST   ASV       LOCATE/ASSIGN SIMPLE VARIABLE
2831 03245   0 02 00000      LDA   0         PUT VARIABLE ADDRESS ON THE STACK
2832 03246   0 10 02766      JST   PUSH      X
2833 03247   0 10 03013      JST   GCHR      HAS ASSIGNMENT LIST BEEN PROCESSED ¢
2834 03250   0 11 00445      CAS   C254      (COMMA MEANS NO)
2835 03251   100000          SKP             X
2836 03252   0 01 03242      JMP   ASN2      NO ... GO PROCESS NEXT NAME
2837 03253   0 10 03031      JST   UCHR      NOT COMMA, SO IT MUST BE '=' OR ELSE
2838 03254   0 02 00455      LDA   C275      X
2839 03255   0 10 03050      JST   GCCK      X
2840 03256   0 10 02412      JST   EXPA      EVALUATE THE EXPRESSION
2841 03257   0 10 03062      JST   GDLM      MAKE SURE IT END WITH A C/R OR :
2842 03260   0 10 02775 ASN3 JST   POP       GET VARIABLE ADDRESS FROM THE STACK
2843 03261   101040          SNZ             IS IT END OF STACK MARKER ¢
2844 03262   0 01 04550      JMP   SEX       .....YES..... PROCEED TO NEXT STATEMENT
2845 03263   0 04 03266      STA   ASN1      SET ADDRESS IN STORE CALLING SEQUENCE
2846 03264   0 10 03147      JST   LCVL      GET VALUE OF THE EXPRESSION
2847 03265   0 10 00000      JST   HS22      ASSIGN THE VALUE TO LIST VARIABLE
2848 03266   0 000000   ASN1 DAC   **        X
2849 03267   0 01 03260      JMP   ASN3      GO TRY FOR ANOTHER LIST ELEMENT
2850                    *
2851                    *
2852                    *
2853                         EJCT
```

```
2854                    *          GOTO STATEMENT PROCESSOR
2855                    *
2856                    *
2857                    *     STATEMENT SYNTAX:
2858                    *
2859                    *          <GOTO STATEMENT>::=GOTO<LINE NUMBER>□:CC/R■
2860                    *
2861                    *
2862                    *
2863 03270   0 10 04532 GOTO  JST   ISN          GET THE STATEMENT NUMBER
2864 03271   0 10 03062       JST   GDLM         MAKE SURE NEXT CHARACTER IS : OR C/R
2865 03272   0 35 00034 GOT2  LDX   SIP          PROTECT SI PNTR DURING STMT SEARCH
2866 03273   0 10 04451       JST   SISR         LOOK UP THE NUMBER IN THE STATEMENT INDEX
2867 03274   0 01 03301       JMP   GOT3         ERROR ... UNDEFINED STATEMENT NUMBER
2868 03275   0 02 00060       LDA   SEQI         SEE IF STMT SEQUENCING IS INHIBITED
2869 03276   101040           SNZ                X
2870 03277   0 01 03157       JMP   ESMT         NO...GO EXECUTED THE REQUESTED STATEMENT
2871 03300   0 01 01000       JMP   CMOD         YES...IGNORE AND RETURN TO COMMAND MODE
2872                    *
2873 03301   0 15 00034 GOT3  STX   SIP          RESTORE SI PNTR TO CURRENT LINE
2874 03302   0 10 05206       JST   ERR          REPORT UNDEFINED STATEMENT NO. ERROR
2875 03303   152723           BCI   1,US         X
2876                    *
2877                    *
2878                    *
2879                         EJCT
```

```
2880                        *         ON STATEMENT PROCESSOR
2881                        *
2882                        *
2883                        *    STATEMENT SYNTAX:
2884                        *
2885                        *         .<ON STATEMENT>::=ON<EXPRESSION>GOTO<LINE NUMBER>
2886                        *                          ¤,<LINE NUMBER>¤(0,*)¤:¢C/R¤
2887                        *
2888                        *
2889                        *         AN 'ON' ERROR MESSAGE WILL BE ISSUED
2890                        *    ON THE FOLLOWING CONDITIONS:
2891                        *
2892                        *         1)  'GOTO' MISSING
2893                        *         2)  EXPRESSION ZERO OR MINUS
2894                        *
2895                        *
2896 03304   140040     ON    CRA                    EVALUATE THE EXPRESSION
2897 03305   0 10 02412       JST   EXPA             X
2898 03306   0 10 03013       JST   GCHR             TEST FOR 'GOTO'
2899 03307   0 05 00507       ERA   GTC              ('GOTO' IS COMPRESSED TO 1 BYTE)
2900 03310   100040           SZE                    X
2901 03311   0 10 05206 ON1   JST   ERR              'ON' STATEMENT ERROR
2902 03312   147716           BCI   1,ON             (THIS WILL EXECUTE AS AN ANA)
2903 03313   0 10 03147       JST   LCVL             TRUNCATE EXPRESSION TO AN INTEGER
2904 03314   0 10 00000       JST   IFLT             X
2905 03315   0 01 03311       JMP   ON1              ERROR...NUMBER TOO LARGE
2906 03316   140407     ON3   TCA                    TEST FOR VALUE < 1
2907 03317   101400           SMI                    X
2908 03320   0 01 03311       JMP   ON1              ERROR ... EXPRESSION <= 0
2909 03321   0 04 00000       STA   0                SAVE VALUE FOR STEPPING THROUGH LIST
2910 03322   0 10 04532 ON2   JST   ISN              GET NEXT STMT NO. FROM THE LIST
2911 03323   0 12 00000       IRS   0                IS THIS THE ONE WE WANT ¢
2912 03324   100000           SKP                    NO
2913 03325   0 01 03272       JMP   GOT2             YES...NOW PROCESS LIKE A GOTO
2914 03326   0 02 00445       LDA   C254             TEST FOR COMMA SEPERATING THE
2915 03327   0 10 03050       JST   GCCK             STATEMENT NUMBERS
2916 03330   0 01 03322       JMP   ON2              CONTINUE SCAN
2917                        *
2918                        *
2919                        *
2920                             EJCT
```

```
2921                    *       IF STATEMENT PROCESSOR
2922                    *
2923                    *
2924                    *   STATEMENT SYNTAX:
2925                    *
2926                    *       <IF STATEMENT>::=<LOGICAL IF STATEMENT>¢
2927                    *                        <ARITHMETIC IF STATEMENT>
2928                    *
2929                    *       <LOGICAL IF STATEMENT>::=IF<EXPRESSION><RELATIONAL OPERATOR>
2930                    *                                 <EXPRESSION>¤TEN<STATEMENT BODY>
2931                    *                                 ¤:<STATEMENT BODY>¤¢TEN<LINE NUMBER>
2932                    *                                 ¢GOTO<LINE NUMBER>¤
2933                    *
2934                    *       <RELATIONAL OPERATOR>::=>¢>=¢=¢=<¢<¢<>
2935                    *
2936                    *       <ARITHMETIC IF STATEMENT>::=IF<EXPRESSION>,<LINE NUMBER>,
2937                    *                                   <LINE NUMBER>,<LINE NUMBER>¤:¢C/R¤
2938                    *
2939                    *
2940 03331  140040      IF    CRA                     CLEAR TEMP. REGISTERS USED IN
2941 03332  0 04 00051        STA   IFT1              PROCESSING RELATIONAL OPS
2942 03333  0 04 00052        STA   IFT2              X
2943 03334  0 10 02412        JST   EXPA              EVALUATE FIRST EXPRESSION
2944 03335  0 10 03020        JST   XCHR              TEST FOR ARITHMETIC IF
2945 03336  0 05 00445        ERA   C254             BY LOOKING FOR COMMA FOLLOWING
2946 03337  101040            SNZ                   FIRST EXPRESSION
2947 03340  0 01 03427        JMP   IF01              GO PROCESS ARITHMETIC IF
2948                    *
2949                    *   HERE FOR LOGICAL IF
2950                    *
2951 03341  0 10 03013 IF02   JST   GCHR              TEST NEXT CHARACTER FOR RELATIONAL OP.
2952 03342  0 11 00454        CAS   C273             < ('274), = ('275), AND > ('276) ARE
2953 03343  0 11 00456        CAS   C277             LEGAL REL. OPS.
2954 03344  0 01 03362        JMP   IF03             NOT RELATIONAL OP
2955 03345  0 01 03362        JMP   IF03             DITTO
2956 03346  0 07 00454        SUB   C273             TRANSPOSE CODE TO 1 FOR <, 2 FOR =,
2957 03347  0 11 00426        CAS   C2               OR 4 FOR >
2958 03350  0 02 00466        LDA   C4               X
2959 03351  101000            NOP                   X
2960 03352  0 13 00051        IMA   IFT1             CODE TO FIRST REL. OP. ACCUM.
2961 03353  101040            SNZ                   WAS THIS THE FIRST REL. OP. ¢
2962 03354  0 01 03341        JMP   IF02             YES...GO CHECK FOR ANOTHER
2963 03355  0 13 00052        IMA   IFT2             MOVE 1ST REL. OP. CODE TO 2ND ACCUM.
2964 03356  101040            SNZ                   WAS THIS THE 2ND REL. OP. ¢
2965 03357  0 01 03341        JMP   IF02             YES...MAKE SURE ANOTHER DOES NOT FOLLOW
2966 03360  0 10 05206 IF04   JST   ERR              IF STATEMENT CONDITION ERROR
2967 03361  144703            BCI   1,IC             X
2968                    *
2969 03362  0 10 03031 IF03   JST   UCHR             MOVE BACK TO 1ST NON REL. OP. CHAR.
2970 03363  0 02 00051        LDA   IFT1             COMBINE THE REL. OP. CODES
```

```
2971 03364   0 05 00052      ERA   IFT2          X
2972 03365   101040          SNZ                 TEST FOR NO CONDITION SELECTED OR
2973 03366   0 01 03360      JMP   IF04          THE SAME CONDITION SELECTED TWICE
2974 03367   0 04 00051      STA   IFT1          SAVE THE CONDITION CODE
2975 03370   140040          CRA                 EVALUATE THE SECOND EXPRESSION
2976 03371   0 10 02412      JST   EXPA          X
2977 03372   0 10 00000      JST   SS22          COMPARE THE TWO EXPRESSIONS
2978 03373   0 000041        DAC   CVAL          X
2979 03374   0 10 03153      JST   SCVL          REDUCE TO -1 IF 1ST EXPR IS >,
2980 03375   0 10 00000      JST   SGNF          -2 IF 2ST EXPR EQUALS 2ND EXPR, OR
2981 03376   0 000041        DAC   CVAL          -3 IF 2ND EXPR IS >
2982 03377   0 10 00000      JST   IFLT          X
2983 03400   000000          OCT   0             NEVER CAN EXECUTE THIS WORD
2984 03401   0 07 00426      SUB   C2            X
2985 03402   0 04 00000      STA   0             X
2986 03403   0 02 00051      LDA   IFT1          MOVE REL. OP. CODE BIT FOR THE RESULT
2987 03404   0414 64         LGL   12            TO A(1)
2988 03405   0414 77         LGL   1             X
2989 03406   0 12 00000      IRS   0             X
2990 03407   0 01 03405      JMP   *-2           X
2991 03410   101400          SMI                 WAS THE RESULT CONDITION SELECTED C
2992 03411   0 01 04554      JMP   ASQ           NO....ADVANCE TO NEXT LINE
2993 03412   0 10 03013      JST   GCHR          YES...TEST FOR 'GOTO'
2994 03413   0 11 00507      CAS   GTC           X
2995 03414   100000          SKP                 NO
2996 03415   0 01 03270      JMP   GOTO          YES...PROCESS GOTO STATEMENT
2997 03416   0 05 00542      ERA   THNC          NO....THIS CHARACTER MUST BE 'THEN'
2998 03417   100040          SZE                 X
2999 03420   0 10 05206      JST   ERR           NO...REPORT SYNTAX ERROR
3000 03421   152310          BCI   1,TH          COULD NOT FIND 'THEN'
3001 03422   0 10 03020      JST   XCHR          SEE WHETHER LINE NO. OF STATEMENT
3002 03423   0 11 00510      CAS   INTF          FOLLOWS
3003 03424   0 01 03157      JMP   ESMT          STATEMENT...GO PROCESS IT
3004 03425   0 01 03270      JMP   GOTO          LINE NUMBR...EXECUTE 'GOTO'
3005 03426   0 01 03157      JMP   ESMT          STATEMENT...GO PROCESS IT
3006                     *
3007                     *     HERE FOR ARITHMETIC IF STATEMENT
3008                     *
3009 03427   0 10 03013 IF01 JST   GCHR          STEP PAST THE COMMA
3010 03430   0 10 00000      JST   SGNF          SET A = -1 IF EXPT < 0, 0 IF
3011 03431   0 000041        DAC   CVAL          EXPR = 0, +1 IF EXPR > 0
3012 03432   0 10 00000      JST   IFLT          X
3013 03433   000000          OCT   0             NEVER CAN EXECUTE THIS WORD
3014 03434   0 06 00426      ADD   C2            MAKE THIS LOOK LIKE AN 'ON' STATEMENT
3015 03435   0 01 03316      JMP   ON3           AND GO PROCESS IT
3016                     *
3017                     *
3018                     *
3019                     EJCT
```

```
3020                    *          FOR STATEMENT PROCESSOR
3021                    *
3022                    *
3023                    *    STATEMENT SYNTAX:
3024                    *
3025                    *          <FOR STATEMENT>::=<SIMPLE VARIABLE>=<EXPRESSION>
3026                    *                            ␣TO␣␣=<EXPRESSION>␣␣STEP␣␣
3027                    *                            <EXPRESSION>␣(0,1)
3028                    *
3029                    *
3030 03436   0 02 00421 FOR  LDA   C11          MAKE SURE NINE WORDS ARE
3031 03437   0 10 03005      JST   UFSC         AVAILABLE FOR STACK ENTRY
3032 03440   0 02 00024      LDA   FNB          IS FOR STACK INITIALIZATION
3033 03441   100040          SZE                REQUIRED
3034 03442   0 01 03451      JMP   FR01         NO...GO APPEND ENTRY
3035 03443   0 02 00023      LDA   DFT          START FOR STACK ABOVE LOW
3036 03444   101040          SNZ                CORE TABLES
3037 03445   0 02 00021      LDA   PTH          X
3038 03446   141206          ADA                X
3039 03447   0 04 00024      STA   FNB          X
3040 03450   0 01 03453      JMP   FR06         X
3041 03451   0 02 00025 FR01 LDA   FNT          GET PREVIOUS STACK TOP
3042 03452   141206          ADA                GET FIRST WORD FOR NEW ENTRY
3043 03453   0 04 03562 FR06 STA   FRT1         SAVE IT
3044 03454   0 06 00420      ADD   C10          GET NEW STACK TOP
3045 03455   0 04 00025      STA   FNT          SAVE IT
3046 03456   0 10 02755      JST   IPDS         SET UP THE PDS ABOVE NEW ENTRY
3047 03457   0 10 04606      JST   PVN          PROCESS INDEX VARIABLE NAME
3048 03460   0 01 03556      JMP   FR07         CANNOT HAVE SUBSCRIPTED VARIABLE AS INDEX
3049 03461   0 10 04634      JST   ASV          LOCATE/ASSIGN THE VARIABLE
3050 03462   0 15 03474      STX   FRT2         SAVE THE POINTER
3051 03463   0 02 00000      LDA   0            GET ITS DISPLACEMENT FROM SVT
3052 03464   0 07 00027      SUB   SVT          X
3053 03465  -0 04 03562      STA*  FRT1         PUT IT IN FIRST WORD OF FN ENTRY
3054 03466   0 12 03562      IRS   FRT1         X
3055 03467   0 02 00455      LDA   C275         MAKE SURE '=' IS NEXT
3056 03470   0 10 03050      JST   GCCK         X
3057 03471   0 10 02412      JST   EXPA         EVALUATE THE FIRST EXPRESSION
3058 03472   0 10 03147      JST   LCVL         RESULT TO A
3059 03473   0 10 00000      JST   HS22         ASSIGN THE VALUE TO THE VARIABLE
3060 03474   0 000000   FRT2 DAC   **           X
3061 03475   0 10 03013      JST   GCHR         NEXT CHARACTER MUST BE
3062 03476   0 11 00445      CAS   C254         ',' OR 'TO'
3063 03477   100000          SKP                X
3064 03500   0 01 03506      JMP   FR02         IT'S A COMMA
3065 03501   0 11 00543      CAS   TOC          NOT COMMA, MAYBE 'TO'
3066 03502   100000          SKP                NO
3067 03503   0 01 03506      JMP   FR02         IT'S 'TO'
3068 03504   0 10 05206      JST   ERR          ERROR...FOR DELIMITER
3069 03505   143304          BCI   1,FD
```

```
3070                        *
3071 03506   140040     FR02 CRA                  EVALUATE THE SECOND EXPRESSION
3072 03507   0 10 02412      JST   EXPA           X
3073 03510   0 10 00000      JST   Ss22           COMPARE SECOND WITH FIRST
3074 03511   0 000041       DAC   CVAL           X
3075 03512   100400         SPL                  X
3076 03513   0 01 03522     JMP   FR03           FIRST IS SMALLER, OK
3077 03514   0 02 00041     LDA   CVAL           SWAP THE VALUES
3078 03515   0 13 00043     IMA   LVAL           X
3079 03516   0 04 00041     STA   CVAL           X
3080 03517   0 02 00042     LDA   CVAL+1         X
3081 03520   0 13 00044     IMA   LVAL+1         X
3082 03521   0 04 00042     STA   CVAL+1         X
3083 03522   0 10 00000 FR03 JST   Ls22           PUT LOW VALUE EXPRESSION IN 2ND AND 3RD
3084 03523   0 000043       DAC   LVAL           WORDS OF ENTRY
3085 03524   0 10 03560     JST   FRST           X
3086 03525   0 10 03147     JST   LCVL           PUT HIGH VALUE EXPRESSION IN 4TH AND 5TH
3087 03526   0 10 03560     JST   FRST           WORDS OF ENTRY
3088 03527   0 10 03013     JST   GCHR           SEE IF A THIRD EXPRESSION IS COMING
3089 03530   0 11 00445     CAS   C254           TEST FOR ','
3090 03531   100000         SKP                  NO
3091 03532   0 01 03542     JMP   FR04           YES
3092 03533   0 11 00540     CAS   STPC           MAYBE 'STEP'
3093 03534   100000         SKP                  NO
3094 03535   0 01 03542     JMP   FR04           YES
3095 03536   0 10 03031     JST   UCHR           LOOK AT CURRENT CHARACTER LATER
3096 03537   0 10 00000     JST   Ls22           SET STEP =1 BY DEFAULT
3097 03540   0 000477       DAC   F1             X
3098 03541   0 01 03545     JMP   FR05           X
3099 03542   140040     FR04 CRA                  EVALUATE THE THIRD EXPRESSION
3100 03543   0 10 02412     JST   EXPA           X
3101 03544   0 10 03147     JST   LCVL           GET THE RESULT
3102 03545   0 10 03560 FR05 JST   FRST           STEP VALUE TO WORDS 6 + 7 OF ENTRY
3103 03546   0 10 03062     JST   GDLM           DELIMITER MUST BE NEXT (: OR C/R)
3104 03547   0 07 00433     SUB   C215           IF C/R, THEN LAST WORD OF ENTRY IS
3105 03550   100040         SZE                  ZERO, ELSE IT IS CURRENT SBP
3106 03551   0 02 00037     LDA   SBP            X
3107 03552   000201         IAB                  X
3108 03553   0 02 00034     LDA   SIP            8TH WORD OF ENTRY IS POINTER TO THIS LINE
3109 03554   0 10 03560     JST   FRST           SET 8TH AND 9TH WORDS
3110 03555   0 01 04550     JMP   SEX            GO PROCESS NEXT STATEMENT
3111                        *
3112                        *
3113 03556   0 10 05206 FR07 JST   ERR            REPORT INDEX VARIABLE ERROR
3114 03557   144726         BCI   1,IV           X
3115                        *
3116                        *
3117                        *    FRST: STORE WORD PAIR
3118                        *
3119                        *          THE WORD PAIR IN A + B ARE STORED IN THE
```

```
3120                        *     TABLE STARTING AT THE ADDRESS IN FRT1. FRT1 IS THEN
3121                        *     INCREMENTED BY TWO.
3122                        *
3123                        *
3124 03560    0 000000     FRST  DAC   **
3125 03561    0 10 00000         JST   H$22              STORE THE WORD PAIR
3126 03562    0 000000     FRT1  DAC   **                X
3127 03563    0 12 03562         IRS   FRT1              BUMP THE TABLE POINTER
3128 03564    0 12 03562         IRS   FRT1              X
3129 03565   -0 01 03560         JMP*  FRST              RETURN
3130                        *
3131                        *
3132                        *
3133                              EJCT
```

```
3134                    *          NEXT STATEMENT PROCESSOR
3135                    *
3136                    *
3137                    *    STATEMENT SYNTAX:
3138                    *
3139                    *          <NEXT STATEMENT>::=NEXT<SIMPLE VARIABLE>¤:¢CR¤
3140                    *
3141                    *
3142 03566  0 10 04606 NEXT JST   PVN          PROCESS INDEX VARIABLE NAME
3143 03567  0 01 03556       JMP   FR07         SUBSCRIPTED VARIABLE IS ILLEGAL INDEX
3144 03570  0 10 04664       JST   LSV          FIND NAME IN TABLE
3145 03571  0 01 03644       JMP   NX01         ERROR ... UNDEFINED VARIABLE MEANS
3146                    *                       THAT THIS IS AN UNMATCHED 'NEXT'
3147 03572  0 10 03062       JST   GDLM         CHECK FOR : OR C/R'
3148 03573  0 02 00025 NX04  LDA   FNT          IS THE TABLE EMPTY ¢
3149 03574  101040           SNZ
3150 03575  0 01 03644       JMP   NX01         YES....NO 'FOR' FOR NEXT
3151 03576  0 07 00420       SUB   C10          GET ADDRESS OF FIRST WORD OF TOP ENTRY
3152 03577  0 13 00000       IMA   0            SWAP WITH INDEX VARIABLE ADDRESS
3153 03600  0 04 03607       STA   NXT1         SET LINKAGE TO VARIABLE IN A COUPLE
3154 03601  0 04 03613       STA   NXT2         OF LOAD AND STORE CALLING SEQUENCES
3155 03602  0 07 00027       SUB   SVT          GET THIS VARIABLES DISP. FROM SVT
3156 03603  1 07 00000       SUB   0,1          DOES THIS 'NEXT' MATCH THE
3157 03604  100040           SZE                LAST 'FOR'¢
3158 03605  0 01 03641       JMP   NX03         ....NO.... GO ABORT MOST RECENT FOR
3159                    *                       AND TRY THE NEXT LOWER LEVEL
3160 03606  0 10 00000       JST   L$22         GET CURRENT VALUE OF INDEX VARIABLE
3161 03607  0 000000   NXT1  DAC   **           X
3162 03610  0 10 00000       JST   A$22         ADD THE INCREMENT
3163 03611  1 000005         DAC   5,1          X
3164 03612  0 10 00000       JST   H$22         SAVE THE NEW INDEX VARIABLE VALUE
3165 03613  0 000000   NXT2  DAC   **           X
3166 03614  0 10 03153       JST   SCVL         IN CVAL ALSO FOR LATER REFERENCE
3167 03615  0 10 00000       JST   S$22         IS IT GREATER THAN LOW
3168 03616  1 000001         DAC   1,1          RANGE VALUE¢
3169 03617  100400           SPL                X
3170 03620  0 01 03637       JMP   NX02         NO....GO TERMINATE THE LOOP
3171 03621  0 10 00000       JST   L$22         IS IT LESS THAN HIGH
3172 03622  1 000003         DAC   3,1          RANGE VALUE¢
3173 03623  0 10 00000       JST   S$22         X
3174 03624  0 000041         DAC   CVAL         X
3175 03625  100400           SPL                X
3176 03626  0 01 03637       JMP   NX02         NO....GO TERMINATE LOOP
3177 03627  0 10 00000       JST   L$22         GET POINTERS TO BEGINNING OF RANGE
3178 03630  1 000007         DAC   7,1          X
3179 03631  0 04 00034       STA   SIP          SET START INDEX POINTER
3180 03632  000201           IAB                SET SOURCE BYTE POINTER
3181 03633  0 04 00037       STA   SBP          X
3182 03634  101040           SNZ                IS THE 'FOR' THE LAST START ON ITS LINE¢
3183 03635  0 01 04554       JMP   ASQ          YES.. GO PROCESS AT LINE FOLLOWING THE FOR
```

```
3184 03636   0 01 03157       JMP   ESMT          NO..GO EXECUTE NEXT STMT ON SAME LINE
3185                       *
3186 03637   0 10 03646 NX02 JST   DFSE          DELETE USED UP FOR STACK ENTRY
3187 03640   0 01 04550       JMP   SEX           PROCEED TO THE NEXT STATEMENT
3188                       *
3189                       *
3190 03641   0 10 03646 NX03 JST   DFSE          ABORT HIGHEST 'FOR'
3191 03642   0 35 03607       LDX   NXT1          RESTORE X TO AVOID TROUBLE
3192 03643   0 01 03573       JMP   NX04          GO TEST THIS FOR LEVEL FOR A MATCH
3193                       *
3194                       *
3195 03644   0 10 05206 NX01 JST   ERR           REPORT 'NEXT' ERROR!
3196 03645   147330          BCI   1,NX          X
3197                       *
3198                       *
3199                       *     DFSE - DELETE HIGHEST FOR STACK ENTRY
3200                       *
3201                       *
3202 03646   0 000000    DFSE DAC   **
3203 03647   0 02 03655       LDA   M11           RETURN 9 WORDS TO FREE STORAGE
3204 03650   0 10 03005       JST   UFSC          X
3205 03651   0 02 00025       LDA   FNT           DECREMENT FOR STACK HIGH POINTER
3206 03652   0 07 00421       SUB   C11           X
3207 03653   0 11 00024       CAS   FNB           IS THE STACK NOW EMPTY ?
3208 03654   0 01 03657       JMP   *+3           .........NO.........
3209 03655   177767      M11  OCT   -11          NEVER CAN EXECUTE THIS WORD
3210 03656   140040           CRA                 YES ... POINTERS MUST BE CLEARED
3211 03657   0 04 00025       STA   FNT           SET STACK HIGH POINTER
3212 03660   101040           SNZ                 IF STACK NOW EMPTY, CLEAR STACK
3213 03661   0 04 00024       STA   FNB           LOW POINTER
3214 03662   0 10 02755       JST   IPDS          RE-INITIALIZE THE PUSH DOWN STACK
3215 03663  -0 01 03646       JMP*  DFSE          RETURN FROM WHENCE WE CAME
3216                       *
3217                       *
3218                       *
3219                            EJCT
```

```
                        *
                        *       GOSUB STATEMENT PROCESSOR
                        *
                        *
                        *   'STATEMENT SYNTAX:
                        *
                        *       <GOSUB STATEMENT>:=GOSUB<LINE NUMBER>¤:¢C/R¤
                        *
                        *
                        *
03664    0 02 00036 GOSB LDA   RTP             SEE IF RETURN STACK IS FULL
03665    0 11 00535      CAS   RTM             X
03666    0 10 05206      JST   ERR             YES...>8 OUTSTANDING GOSUBS
03667   143723           BCI   1,GS            X
03670    0 04 00000      STA   0               NO....SET INDEX FOR TABLE ACCESSING
03671    0 10 04532      JST   ISN             GET STMT NUMBER AND LEAVE IT IN SNUM
03672    0 10 03062      JST   GDLM            MAKE SURE : OR C/R IS NEXT
03673    0 05 00433      ERA   C215            TEST FOR CARRIAGE RETURN
03674   100040           SZE                   IF IT IS, SECOND WORD OF ENTRY = 0
03675    0 02 00037      LDA   SBP             OTHERWISE SBP TO NEXT STMT ON THIS LINE
03676    1 04 00001      STA   1,1             X
03677    0 02 00034      LDA   SIP             FIRST WORD OF ENTRY IS POINTER
03700    1 04 00000      STA   0,1             TO LINE WITH GOSUB
03701    0 12 00036      IRS   RTP             BUMP THE RETURN STACK POINTER
03702    0 12 00036      IRS   RTP             X
03703    0 01 03272      JMP   GOT2            NOW HANDLE LIKE A 'GOTO'
                        *
                        *
                        *
                        EJCT
```

```
3249                    *          RETURN STATEMENT PROCESSOR
3250                    *
3251                    *
3252                    *     STATEMENT SYNTAX:
3253                    *
3254                    *          <RETURN STATEMENT>:=RETURN□:CC/R■
3255                    *
3256                    *
3257                    *
3258 03704   0 10 03062 RTRN JST   GDLM          MAKE SURE : OR C/R IS NEXT
3259 03705   0 02 00036      LDA   RTP           GET RETURN STACK POINTER
3260 03706   0 11 00534      CAS   RTB           CHECK FOR EMPTY STACK
3261 03707   0 01 03712      JMP   *+3           TABLE NOT EMPTY
3262 03710   0 10 05206      JST   ERR           ERROR...RETURN WITHOUT A GOSUB
3263 03711   151324          BCI   1,RT          X
3264 03712   0 07 00426      SUB   C2            DELETE LAST ENTRY
3265 03713   0 04 00036      STA   RTP           SAVE UPDATED POINTER
3266 03714   0 04 00000      STA   0             IN INDEX ALSO FOR TABLE ACCESS
3267 03715   1 02 00000      LDA   0,1           RESET SI POINTER TO START
3268 03716   0 04 00034      STA   SIP           EXECUTION AFTER LAST GOSUB
3269 03717   1 02 00001      LDA   1,1           RETRIEVE SBP TO GOSUB STMT (IT'S ZERO
3270 03720   0 04 00037      STA   SBP           IF GOSUB WAS LAST STMT ON LINE, NONZERO
3271 03721   101040          SNZ                 IF NOT)
3272 03722   0 01 04554      JMP   ASQ           ADVANCE TO LINE FOLLOWING GOSUB
3273 03723   0 01 03157      JMP   ESMT          EXECUTE NEXT STMT ON SAME LINE AS GOSUB
3274                    *
3275                    *
3276                    *
3277                         EJCT
```

```
3278                    *       CALL STATEMENT PROCESSOR
3279                    *
3280                    *
3281                    *   STATEMENT SYNTAX:
3282                    *
3283                    *       <CALL STATEMENT>::=CALL(<SUBROUTINE IDENTIFIER<¤,
3284                    *                           <SUBROUTINE PARAMETER>¤(0,*))
3285                    *
3286                    *       <SUBROUTINE IDENTIFIER>::=<EXPRESSION>
3287                    *
3288                    *       <SUBROUTINE PARAMETER>::=<VARIABLE NAME>¢<EXPRESSION>
3289                    *
3290                    *
3291                    *
3292 03724   0 02 00441 CALL  LDA   C250            GET LEFT PAREN
3293 03725   0 10 03050       JST   GCCK            X
3294 03726   0 04 00067       STA   CLT1            CLEAR ARGUMENT COUNTER
3295 03727   0 04 00070       STA   CLT2            CLEAR <EXPRESSION> COUNTER
3296 03730   0 10 02412       JST   EXPA            EVALUATE SUBROUTINE IDENTIFIER
3297 03731   0 10 03147       JST   LCVL            CONVERT RESULT TO INTEGER
3298 03732   0 10 00000       JST   IFLT            X
3299 03733   0 01 04036       JMP   CL01            ERROR ... IDENTIFIER OUT OF RANGE
3300 03734   0 11 00416       CAS   C0              PERFORM EXCLUSIVE RANGE CHECK
3301 03735   0 11 00475       CAS   CMAX            X
3302 03736   0 01 04036       JMP   CL01            .....ERROR..... OUT OF RANGE
3303 03737   0 01 04036       JMP   CL01            .....ERROR..... OUT OF RANGE
3304 03740   0 06 00554       ADD   CJST            FORM SUBROUTINE CALL
3305 03741   0 04 00255       STA   IBUF            PUT IT AT START OF CALLING SEQUENCE
3306 03742   0 10 03013 CL05  JST   GCHR            CHECK EXPRESSION DELIMITER
3307 03743   0 11 00445       CAS   C254            COMMA ¢
3308 03744   100000           SKP                   NO
3309 03745   0 01 03777       JMP   CL02            ......YES......  MORE ARGUMENTS FOLLOW
3310 03746   0 10 03031       JST   UCHR            NOT COMMA, MUST BE RIGHT PAREN
3311 03747   0 02 00442       LDA   C251            X
3312 03750   0 10 03050       JST   GCCK            X
3313 03751   0 10 03062       JST   GDLM            C/R OR : MUST FOLLOW RIGHT PAREN
3314 03752   0 02 00067       LDA   CLT1            GET ARGUMENT COUNT
3315 03753   0 04 00000       STA   0               IN X TO GET AT END OF PARAMETER LIST
3316 03754   0 07 00426       SUB   C2              MORE THAN ONE ARGUMENT ¢
3317 03755   100400           SPL                   X
3318 03756   0 01 03762       JMP   *+4             ...NO... LIST TERMINATOR NOT NEEDED
3319 03757   140040           CRA                   INSERT ZERO WORD AT END OF PARAMETER LIST
3320 03760   1 04 00256       STA   WORK,1          FOR FORTRAN COMPATIBILITY
3321 03761   0 12 00000       IRS   0               ACCOUNT FOR WORD JUST INSERTED
3322 03762   0 02 00567       LDA   CJMP            INSERT RETURN JUMP AT END
3323 03763   1 04 00256       STA   WORK,1          OF CALLING SEQUENCE
3324 03764   0 01 00255       JMP   IBUF            GO CALL THE SUBROUTINE
3325                    *
3326                    *   HERE WHEN SUBROUTINE RETURNS
3327                    *
```

```
3328 03765   0 02 00070 CL06 LDA   CLT2        ANY JUNK ON THE PUSH DOWN STACK ?
3329 03766   101040          SNZ               X
3330 03767   0 01 04550       JMP   SEX         ...NO... PROCEED TO THE NEXT STATEMENT
3331 03770   140407          TCA               ESTABLISH COUNTER TO CLEAR THE STACK
3332 03771   0 04 00000       STA   0           X
3333 03772   0 10 02775       JST   POP         REMOVE A REAL NUMBER FROM THE STACK
3334 03773   0 10 02775       JST   POP         X
3335 03774   0 12 00000       IRS   0           BUMPT THE COUNTER
3336 03775   0 01 03772       JMP   *-3         MORE TO GO
3337 03776   0 01 04550       JMP   SEX         PROCEED TO THE NEXT STATEMENT
3338                  *
3339 03777   0 10 03020 CL02 JST   XCHR        SEE IF ISOLATED VARIABLE NAME
3340 04000   0 10 03121       JST   ALFA        X
3341 04001   0 01 04020       JMP   CL03        NO ... IT MUST BE AN EXPRESSION
3342 04002   0 02 00037       LDA   SBP         MARK START OF NAME
3343 04003   0 04 00071       STA   CLT3        X
3344 04004   0 10 04606       JST   PVN         ISOLATE/CLASSIFY VARIABLE NAME
3345 04005   0 10 04714       JST   ADV         LOCATE/ASSIGN DIMENSIONED VARIABLE
3346 04006   0 10 04634       JST   ASV         LOCATE/ASSIGN SIMPLE VARIABLE
3347 04007   0 10 03020       JST   XCHR        SEE IF VARIABLE NAME IS PART OF A
3348 04010   0 11 00445       CAS   C254        BIGGER EXPRESSION
3349 04011   100000          SKP               POSSIBLE
3350 04012   0 01 04030       JMP   CL04        NO ... WE HAVE ISOLATED A PARAMETER
3351 04013   0 05 00442       ERA   C251        TRY FOR ')'
3352 04014   101040          SNZ               X
3353 04015   0 01 04030       JMP   CL04        VARIABLE NAME STANDS ALONE
3354 04016   0 02 00071       LDA   CLT3        IT'S PART OF AN EXPRESSION ....
3355 04017   0 04 00037       STA   SBP         ....RESET TO START OF NAME
3356 04020   140040     CL03 CRA               EVALUATE EXPRESSION
3357 04021   0 10 02412       JST   EXPA        X
3358 04022   0 35 00035       LDX   PDLP        SET PARAMETER ADDRESS IN X
3359 04023   0 10 03147       JST   LCVL        PUT PARAMETER VALUE ON THE STACK
3360 04024   0 10 02766       JST   PUSH        X
3361 04025   000201          IAB               X
3362 04026   0 10 02766       JST   PUSH        X
3363 04027   0 12 00070       IRS   CLT2        BUMP <EXPRESSION> COUNT
3364 04030   0 02 00000 CL04 LDA   0           GET PARAMETER ADDRESS
3365 04031   0 35 00067       LDX   CLT1        X POINTS TO NEXT WORD IN CALLING SEQ
3366 04032   1 04 00256       STA   WORK,1      SET PARAMETER ADDRESS IN CALLING SEQ.
3367 04033   0 12 00067       IRS   CLT1        BUMP PARAMETER COUNT
3368 04034   0 01 03742       JMP   CL05        GO TEST DELIMITER
3369                  *
3370 04035   0 000000        DAC   **          UNDEFINED SUBROUTINE ENTRY
3371 04036   0 10 05206 CL01 JST   ERR         REPORT SUBROUTINE SELECTION ERROR
3372 04037   151723          BCI   1,SS        X
3373                  *
3374                  *
3375                  *
3376                  *
3377                  .     EJCT
```

```
3378                    *       READ AND INPUT STATEMENT PROCESSORS
3379                    *
3380                    *
3381                    *   STATEMENT SYNTAX:
3382                    *
3383                    *       <READ STATEMENT>:=READ<READ LIST>¤:¢C/R¤
3384                    *
3385                    *       <INPUT STATEMENT>:=INPUT<READ LIST>¤:¢C/R¤
3386                    *
3387                    *       <READ LIST>:=<VARIABLE>¤,<VARIABLE>¤(0,*)
3388                    *
3389                    *
3390                    *
3391 04040   140040     INPT CRA                CLEAR DATA AVAILIBILITY POINTER
3392 04041   0 04 04111       STA   ISBP        X
3393 04042   0 02 04110       LDA   IDAC        SET LINKAGE TO INPT DATA AND TO DATA FETCH
3394 04043   100000           SKP               X
3395 04044   0 02 04124 READ  LDA   RDAC        SET LINKAGE TO READ DATA AND DATA FETCH
3396 04045   0 04 00047       STA   RDT1        SAVE PARAMETER LINKAGE
3397 04046   0 10 04606 RD02  JST   PVN         PROCESS LIST VARIABLE NAME
3398 04047   0 10 04714       JST   ADV         LOCATE/ASSIGN SUBSCRIPTED VARIABEL
3399 04050   0 10 04634       JST   ASV         LOCATE/ASSIGN SIMPLE VARIABLE
3400 04051   0 15 04073       STX   RDT2        SAVE THE VARIABLE POINTER
3401 04052   0 35 00047       LDX   RDT1        RDT1 POINTS TO DATA SBP
3402 04053   1 02 00000       LDA   0,1         IS DATA AVAILABLE ¢
3403 04054   101040           SNZ               X
3404 04055   -1 01 00001      JMP*  1,1         NO...GO GET SOME MORE
3405 04056   0 02 00037 RD04  LDA   SBP         SWAP STMT POINTER WITH
3406 04057   -0 13 00047      IMA*  RDT1        DATA POINTER
3407 04060   0 04 00037       STA   SBP         X
3408 04061   140040           CRA               EVALUATE THE NEXT DATA ITEM
3409 04062   0 10 02412       JST   EXPA        X
3410 04063   0 10 03013       JST   GCHR        TEST FOR END OF DATA LIST
3411 04064   0 10 03137       JST   DLCK        X
3412 04065   0 01 04103       JMP   RD01        NO....GO INSURE THAT SEPERATOR IS A COMMA
3413 04066   140040           CRA               YES...CLEAR DATA AVAILIBILITY INDICATOR
3414 04067   -0 13 00047 RD03 IMA*  RDT1        X
3415 04070   0 04 00037       STA   SBP         AND RESTORE STATEMENT POINTER
3416 04071   0 10 03147       JST   LCVL        GET VALUE OF DATA ITEM
3417 04072   0 10 00000       JST   H22         ASSIGN IT TO READ LIST ITEM
3418 04073   0 000000   RDT2  DAC   **          X
3419 04074   0 10 03013       JST   GCHR        AT END OF STATEMENT ¢
3420 04075   0 11 00445       CAS   C254        X
3421 04076   100000           SKP               X
3422 04077   0 01 04046       JMP   RD02        NO....GO PROCESS NEXT LIST ITEM
3423 04100   0 10 03031       JST   UCHR        YES...MAKE SURE DELIMITER IS : OR C/R
3424 04101   0 10 03062       JST   GDLM        X
3425 04102   0 01 04550       JMP   SEX         GO PROCESS NEXT STATEMENT
3426                    *
3427 04103   0 10 03031 RD01  JST   UCHR        IF NOT DELIMITER, IT MUST BE A COMMA
```

```
3428 04104   0 02 00445       LDA   C254              X
3429 04105   0 10 03050       JST   GCCK              X
3430 04106   0 02 00037       LDA   SBP               UPDATE DATA POINTER
3431 04107   0 01 04067       JMP   RD03              X
3432                        *
3433 04110   0 004111    IDAC DAC   *+1               LINKAGE TO INPUT DATA PARAMETERS
3434 04111   000000      ISBP OCT   0                 POINTER TO INPUT DATA
3435 04112   0 004113          DAC   *+1               POINTER TO INPUT DATA FETCH ROUTINE
3436                        *
3437                        *     HERE TO READ DATA LIST FROM ASR
3438                        *
3439 04113   0 02 00037       LDA   SBP               SAVE CURRENT STATEMENT BYTE POINTER DURING
3440 04114   0 04 04111       STA   ISBP              THE DATA FETCH
3441 04115   0 02 00437       LDA   C241              PERFIX LINE WITH '%'
3442 04116   0 10 01467       JST   ILIN              INPUT LINE FROM ASR
3443 04117   0 000532    INT1 DAC   IBUF+IBUF        READ INTO DATA INPUT BUFFER
3444 04120   0 02 04117       LDA   INT1              CALCULATE SBP OF DATA LINE
3445 04121   0 13 04111       IMA   ISBP              SWAP WITH CURRENT STMT BYTE POINTER
3446 04122   0 04 00037       STA   SBP               RESTORE CURRENT STATEMENT POINTER
3447 04123   0 01 04056       JMP   RD04              RETURN TO INPUT PROCESSING
3448                        *
3449                        *     HERE FOR READ STATEMENT DATA FETCH
3450                        *
3451 04124   0 004126    RDAC DAC   *+2               LINKAGE TO READ PARAMETERS
3452 04125   0 000000    RSIP DAC   **                POINTER TO CURRENT DATA STATEMENT
3453 04126   0 000000    RSBP DAC   **                POINTER TO CURRENT DATA ITEM
3454 04127   0 004130          DAC   *+1               LINKAGE TO DATA FETCH
3455                        *
3456 04130   0 10 04143       JST   RD06              SET STMT PNTRS FOR DATA STMT SEARCH
3457 04131   0 02 00476       LDA   DTAC              FIND THE NEXT DATA STATEMENT
3458 04132   0 10 04433       JST   SSR               X
3459 04133   0 01 04136       JMP   RD07              RAN OUT OF DATA
3460 04134   0 10 04143       JST   RD06              SWAP BACK THE POINTERS
3461 04135   0 01 04056       JMP   RD04              CONTINUE WITH READ STATEMENT
3462                        *
3463 04136   0 10 04143 RD07 JST   RD06              RESTORE THE POINTERS
3464 04137   140040          CRA                     RESET DATA AVAILABILITY POINTER
3465 04140   0 04 04126       STA   RSBP              X
3466 04141   0 10 05206       JST   ERR               REPORT THE ERROR
3467 04142   142301          BCI   1,DA              X
3468                        *
3469                        *     HERE TO SWAP DATA POINTERS
3470                        *
3471 04143   0 000000    RD06 DAC   **
3472 04144   0 02 04125       LDA   RSIP              SWAP SIP WITH RSIP
3473 04145   0 13 00034       IMA   SIP               X
3474 04146   0 04 04125       STA   RSIP              X
3475 04147   0 02 04126       LDA   RSBP              SWAP SBP WITH RSBP
3476 04150   0 13 00037       IMA   SBP               X
3477 04151   0 04 04126       STA   RSBP              X
```

```
3478 04152  -0 01 04143      JMP*  RD06          RETURN
3479                         *
3480                         *
3481                         *
3482                         EJCT
```

```
3483                        *          RESTORE STATEMENT PROCESSOR
3484                        *
3485                        *
3486                        *    STATEMENT SYNTAX:
3487                        *
3488                        *          <RESTORE STATEMENT>:=RESTORE¤:¢C/R¤
3489                        *
3490                        *
3491                        *
3492 04153   0 10 03062 RSTR  JST   GDLM          MAKE SURE : OR C/R IS NEXT
3493 04154   0 10 04156       JST   REST          INITIALIZE DATA STMT POINTERS
3494 04155   0 01 04550       JMP   SEX           GO PROCESS NEXT STATEMENT
3495                        *
3496                        *
3497                        *    REST - RESET DATA STATEMENT POINTERS
3498                        *
3499                        *
3500 04156   0 000000    REST DAC   **
3501 04157   0 02 00032       LDA   SIB           SET DATA STMT POINTER TO JUST BELOW START
3502 04160   0 07 00426       SUB   C2            OF STMT INDEX, SO THAT THE FIRST DATA
3503 04161   0 04 04125       STA   RSIP          STMT WILL BE FOUND DURING THE NEXT READ
3504 04162   140040           CRA                 SET RSBP TO INDICATE THAT A SEARCH
3505 04163   0 04 04126       STA   RSBP          FOR A DATA STMT WILL HAVE TO BE PERFORMED
3506 04164  -0 01 04156       JMP*  REST          RETURN
3507                        *
3508                        *
3509                        *
3510                             EJCT
```

```
                    *
                    *       PRINT STATEMENT PROCESSOR
                    *
                    *
                    *   STATEMENT SYNTAX:
                    *
                    *       <PRINT STATEMENT>::=PRINT¤<PRINT LIST>¤(0,1)
                    *
                    *       <PRINT LIST>::=<PRINT ITEM>¤¤;¢;¤<PRINT ITEM>¤(0,*)¤;¢;¤(0,1)
                    *
                    *       <PRINT ITEM>::=<EXPRESSION>¢<MESSAGE>¢<MESSAGE><EXPRESSION>¢
                    *                   TAB(<EXPRESSION>)
                    *
                    *       <MESSAGE>::="¤<ALPHABETIC CHARACTER>¢<DIGIT>¢
                    *                   <SPECIAL CHARACTER>¤(1,*)"
                    *
                    *
                    *
04165   0 10 03013  PRNT JST   GCHR         TEST FOR EMPTY PRINT LIST
04166   0 10 03137       JST   DLCK         (: OR C/R IMMEADIATLY FOLLOWING 'PRINT')
04167   0 01 04176       JMP   PR11         NOT EMPTY
04170   0 10 00000  PR04  JST   LFCR         ADVANCE TO NEXT LINE ON ASR
04171   0 01 04550       JMP   SEX          GO PROCESS NEXT STATEMENT
                    *
04172   0 10 03013  PR01  JST   GCHR         GET 1ST CHARACTER OF PRINT ITEM
04173   0 10 03137       JST   DLCK         WAS THERE A DANNGLING COMMA OR ;
04174   100000           SKP                NO
04175   0 01 04550       JMP   SEX          YES...EXIT WITHOUT LINE ADVANCE
04176   0 11 00440  PR11  CAS   C242         TEST FOR QUOTED TEXT STRING
04177   100000           SKP                NO
04200   0 01 04257       JMP   PR02         YES...GO PRINT THE STRING
04201   0 05 00541       ERA   TABC         TEST FOR TAB OPERATION
04202   101040           SNZ                X
04203   0 01 04230       JMP   PR03         YES...GO PROCESS TAB
                    *
                    *   HERE TO PRINT VALUE OF AN EXPRESSION
                    *
04204   0 10 03031       JST   UCHR         STEP BACK OVER LEADING CHAR OF EXPR
04205   140040           CRA                EVALUATE THE EXPRESSION
04206   0 10 02412       JST   EXPA         X
04207   140040           CRA                NO CHARACTER SURPRESSION FOR THIS
04210   0 10 02123       JST   PCVL         PRINT THE RESULT
                    *
                    *   HERE TO WORK ON PRINT ITEM DELIMITER
                    *
04211   0 10 03013  PR10  JST   GCHR         FETCH THE DELIMITING CHARACTER
04212   0 10 03137       JST   DLCK         IS IT TERMINAL (: OR C/R) ¢
04213   100000           SKP                NO
04214   0 01 04170       JMP   PR04         YES...ADVANCE TO NEXT LINE AND EXIT
04215   0 11 00454       CAS   C273         IS IT A SEMI-COLON ¢
04216   100000           SKP                NO
```

```
04217   0 01 04172        JMP    PR01        YES...NO SPECIAL SPACEING REQUIRED
04220   0 05 00445        ERA    C254        IT MUST BE COMMA,
04221   100040            SZE                OR ELSE AN ERROR
04222   0 01 04277        JMP    PR05        GO REPORT FIELD DELIMITER ERROR
04223   0 06 00424        ADD    C16         FIND NEXT TAB POSITION
04224   0 11 00046        CAS    CPOS        IS COLUMN IN A > CURRENT CARRIAGE POSITION ¢
04225   0 01 04242        JMP    PR06        YES...GO ADVANCE TO POSITION IN A
04226   0 01 04172        JMP    PR01        NO TAB REQUIRED...PROCESS NEXT LIST ITEM
04227   0 01 04223        JMP    *-4         NO...TRY NEXT TAB POSITION
                    *
                    *     HERE TO HANDLE 'TAB' FUNCTION
                    *
04230   0 10 02412  PR03  JST    EXPA        EVALUATE THE EXPRESSION
04231   0 02 00442        LDA    C251        MAKE SURE IT ENDS WITH A
04232   0 10 03050        JST    GCCK        RIGHT PAREN
04233   0 10 03013        JST    GCHR        IF NEXT CHARACTER IS : OR C/R, THEN NO POINT
04234   0 10 03137        JST    DLCK        IN DOING A 'TAB', AS HE
04235   100000            SKP                WILL GET A C/R ANYWAY
04236   0 01 04170        JMP    PR04        X
04237   0 10 03147        JST    LCVL        GET RESULT OF THE EXPRESSION
04240   0 10 00000        JST    IFLT        CONVERT TO INTEGER
04241   0 01 04255        JMP    PR07        TOO BIG...GO TO NEXT LINE
04242   0 11 00531  PR06  CAS    MCOL        IS IT TAB TO BEYOND END OF LINE ¢
04243   0 01 04255        JMP    PR07        YES...ADVANCE TO NEXT LINE
04244   101000            NOP                NO
04245   0 04 00045        STA    PRT1        SAVE TARGET COLUMN NUMBER
04246   0 11 00046  PR09  CAS    CPOS        COMPARE WITH CURRENT POSITION
04247   0 01 04252        JMP    *+3         NOT THERE YET
04250   0 01 04172        JMP    PR01        AT PROPER COLUMN
04251   0 01 04172        JMP    PR01        PAST IT...SO WHAT
04252   0 10 02730        JST    SPAC        MOVE OVER A SPACE
04253   0 02 00045        LDA    PRT1        RETRIEVE TARGET COLUMN NUMBER
04254   0 01 04246        JMP    PR09        GO TEST FOR COMPLETION
                    *
04255   0 10 00000  PR07  JST    LFCR        ADVANCE TO NEXT LINE
04256   0 01 04172        JMP    PR01        GO LOOK AT NEXT ITEM
                    *
                    *     HERE TO OUTPUT TEXT STRING
                    *
04257   0 10 03013  PR02  JST    GCHR        GET THE NEXT CHARACTER OF THE STRING
04260   0 11 00440        CAS    C242        END OF STRING ¢
04261   100000            SKP                NO
04262   0 01 04265        JMP    PR08        YES...GO CLOSE UP
04263   0 10 00000        JST    OTA1        OUTPUT THE CHARACTER
04264   0 01 04257        JMP    PR02        CONTINUE
                    *
04265   0 10 03013  PR08  JST    GCHR        LOOK AT CHARACTER FOLLOWING '"'
04266   0 11 00445        CAS    C254        IF COMMA OR SEMI COLON,
04267   100000            SKP                PERFORM STANDARD INTERITEM PROCESSING
04270   0 01 04212        JMP    PR10+1      IT'S A COMMA
```

Förslag
JMP PRO7

```
PROT  LDA   C215  CR
      JST   OTA1
      CRA
      STA   CPOS
      LDA   PRT1
      JMP   PRO9
```

78-04-04

```
3611 04271   0 11 00454       CAS   C273          MAYBE A SEMI COLON
3612 04272   100000           SKP                 NO
3613 04273   0 01 04212       JMP   PR10+1        YES
3614 04274   0 10 03137       JST   DLCK          IS IT A TERMINAL DELIMITER C
3615 04275   0 01 04176       JMP   PR11          NO....IT MUST BE START OF NEXT ITEM
3616 04276   0 01 04170       JMP   PR04          YES...ADVANCE ASR LINE AND EXIT
3617                      *
3618                      *
3619 04277   0 10 05206 PR05  JST   ERR           REPORT ITEM DELIMITER ERROR
3620 04300   150304           BCI   1,PD          X
3621                      *
3622                      *
3623                      *
3624                           EJCT
```

```
                        *       DEF STATEMENT PROCESSOR
                        *
                        *
                        *   STATEMENT SYNTAX:
                        *
                        *       <DEF STATEMENT>:=DEFFN<ALPHABETIC CHARACTER>
                        *                       (<SIMPLE VARIABLE>)=<EXPRESSION>¤:¢C/R¤
                        *
                        *
3634 04301  0 02 00034 DEF  LDA   SIP          ARE WE IN COMMAND MODE ¢
3635 04302  101040          SNZ                X
3636 04303  0 01 04401      JMP   REM          ......YES......   IGNORE THE DEFINITION
3637 04304  0 10 03013      JST   GCHR         TEST FOR 'FN'
3638 04305  0 05 00545      ERA   DEFF         X
3639 04306  100040          SZE                X
3640 04307  0 01 04375      JMP   DF01         NOT HTERE...REPORT ERROR
3641 04310  0 10 04562      JST   SDFI         IS NAME ALREADY IN FUNCTION INDEX ¢
3642 04311  100000          SKP                NO....AN ENTRY MUST BE APPENDED
3643 04312  0 01 04353      JMP   DF02         YES...REDEFINE THE FUNCTION
3644 04313  0 02 00457      LDA   C3           MAKE SURE THERE IS ROOM
3645 04314  0 10 03005      JST   UFSC         FOR ANOTHER ENTRY
3646 04315  0 02 00025      LDA   FNT          DOES FOR-NEXT TABLE HAVE TO BE MOVED ¢
3647 04316  101040          SNZ                X
3648 04317  0 01 04340      JMP   DF03         NO
3649 04320  0 04 00000      STA   0            SET X TO ADDRESS OF FIRST WORD TO BE MOVED
3650 04321  0 06 00457      ADD   C3           UPDATE THE FOR-NEXT TABLE HIGH POINTER
3651 04322  0 04 00025      STA   FNT          X
3652 04323  0 07 00024      SUB   FNB          GET NO. OF WORDS TO BE MOVED
3653 04324  0 07 00426      SUB   C2           X
3654 04325  140407          TCA                X
3655 04326  0 04 00116      STA   TMP1         X
3656 04327  1 02 00000 DF04 LDA   0,1          MOVE A WORD UP 3 LOCATIONS
3657 04330  1 04 00003      STA   3,1          X
3658 04331  0 02 00000      LDA   0            DECREMENT THE RABLE POINTER
3659 04332  0 07 00417      SUB   C1           X
3660 04333  0 04 00000      STA   0            X
3661 04334  0 12 00116      IRS   TMP1         BUMP THE WORD COUNTER
3662 04335  0 01 04327      JMP   DF04         MORE TO BE MOVED
3663 04336  0 12 00000      IRS   0            GET ADDRESS OF LOWEST WORD IN FOR TABLE
3664 04337  0 15 00024      STX   FNB          UPDATE FOR TABLE BASE POINTER
3665 04340  0 02 00022 DF03 LDA   DFB          IS FUNCTION INDEX EMPTY ¢
3666 04341  100040          SZE                X
3667 04342  0 01 04372      JMP   DF05         NO...GO APPEND ENTRY
3668 04343  0 02 00021      LDA   PTH          START FUNCTION INDEX ON TOP
3669 04344  141206          ADA                OF PROGRAM TEXT STORAGE
3670 04345  0 04 00022      STA   DFB          X
3671 04346  0 06 00426      ADD   C2           GET TABLE HIGN ADDRESS
3672 04347  0 04 00023 DF06 STA   DFT          SET FUNCTION TABLE HIGH
3673 04350  0 07 00426      SUB   C2           GET ADDRESS OF FIRST WORD OF THIS ENTRY
3674 04351  0 04 00000      STA   0            AND LEAVE IT IN THE INDEX FOR TABLE ACCESSING
```

```
3675 04352   0 10 02755       JST   IPDS        REINITIALIZE THE PUSH DOWN STACK!
3676 04353   0 02 00114 DF02  LDA   CHAR        PUT FUNCTION NAME IN FIRST WORD OF ENTRY
3677 04354   1 04 00000       STA   0,1         X
3678 04355   0 02 00441       LDA   C250        MAKE SURE '(' FOLLOWS
3679 04356   0 10 03050       JST   GCCK        X
3680 04357   0 10 04606       JST   PVN         FORM THE DUMMY VARIABLE NAME
3681 04360   0 01 04377       JMP   DF07        SUBSCRIPTED VARIABLES CANNOT BE DUMMY NAMES
3682 04361   0 02 00133       LDA   VARN        PLACE NAME IN SECOND WORD OF TBALE ENTRY
3683 04362   1 04 00001       STA   1,1         X
3684 04363   0 02 00442       LDA   C251        MAKE SURE ')' IS NEXT
3685 04364   0 10 03050       JST   GCCK        X
3686 04365   0 02 00455       LDA   C275        '=' MUST OCCUR NEXT
3687 04366   0 10 03050       JST   GCCK        X
3688 04367   0 02 00037       LDA   SBP         PLACE EXPRESSION POINTER IN
3689 04370   1 04 00002       STA   2,1         THIRD WORD OF ENTRY
3690 04371   0 01 04401       JMP   REM         GO PROCESS NEXT STATEMENT
3691                  *
3692 04372   0 02 00023 DF05  LDA   DFT         APPEND ENTRY TO FUNCTION INDEX
3693 04373   0 06 00457       ADD   C3          X
3694 04374   0 01 04347       JMP   DF06        X
3695                  *
3696 04375   0 10 05206 DF01  JST   ERR         REPORT MISSING 'FN'
3697 04376   143316           BCI   1,FN
3698                  *
3699                  *
3700 04377   0 10 05206 DF07  JST   ERR         REPORT DUMMY NAME ERROR
3701 04400   142326           BCI   1,DV        X
3702                  *
3703                  *
3704                  *
3705                  EJCT
```

```
3706                    *       REMARK STATEMENT PROCESSOR
3707                    *
3708                    *
3709                    *   STATEMENT SYNTAX:
3710                    *       <REM STATEMENT>::=REM□<ALPHABETIC CHARACTER>◊<DIGIT>◊
3711                    *                         <SPECIAL CHARACTER>*(0,*)□:◊C/R■
3712                    *
3713                    *
3714                    *
3715 04401   0 10 04514 REM JST  SES             SCAN UNTIL : OR C/R
3716 04402   0 01 04550     JMP  SEX             GO PROCESS NEXT STATEMENT
3717                    *
3718                    *
3719                    *
3720                        EJCT
```

```
3721                        *       STOP AND END STATEMENT PROCESSORS
3722                        *
3723                        *
3724                        *   STATEMENT SYNTAX:
3725                        *
3726                        *       <STOP STATEMENT>:=STOP□:¢C/R■
3727                        *       <END STATEMENT>:=END□:¢C/R■
3728                        *
3729                        *
3730                        *
3731 04403   0 10 03062  EXIT JST   GDLM            GET TERMINAL DELIMITER
3732 04404   0 10 00000  EXT1 JST   LFCR            ADVANCE ASR LINE
3733 04405   0 10 02702       JST   PLN             PRINT LINE NUMBER
3734 04406   0 10 02713       JST   TYPE            PRINT 'EXIT'
3735 04407   0 004411         DAC   EXT2            MESSAGE POINTER
3736 04410   0 01 01000       JMP   CMOD            RETURN TO COMMAND MODE
3737                        *
3738 04411   120305     EXT2 BCI   2, EXI
     04412   154311
3739 04413   152000           VFD   8,'324,8,'000   'T', MESSAGE TERMINATOR
3740                        *
3741                        *
3742                        *
3743                             EJCT
```

```
3744                    *       ADVANCE TO NEXT SOURCE LINE
3745                    *
3746                    *
3747                    *   CALLING SEQUENCE:
3748                    *
3749                    *       JST   ADVS
3750                    *       ......RETURN      IF BEYOND END OF SOURCE PROGRAM
3751                    *       ......RETURN      IF NOT PAST END OF PROGRAM
3752                    *
3753                    *       THIS ROUTINE WILL UPDATE THE STATEMENT INDEX
3754                    *   POINTER (SIP), AND SET THE SOURCE BYTE POINTER (SBP)
3755                    *   TO POINT TO THE FIRST BYTE IN THE NEW LINE.
3756                    *
3757                    *
3758 04414   0 000000   ADVS DAC   **
3759 04415   0 02 00034      LDA   SIP           UPDATE THE STATEMENT INDEX POINTER
3760 04416   0 06 00426      ADD   C2            (2 WORDS PER ENTRY)
3761 04417   0 04 00034      STA   SIP           X
3762 04420   0 04 00000      STA   0             SET INDEX FOR ACCESSING THE TABLE
3763 04421   0 11 00033      CAS   SIT           TEST AGAINST TABLE TOP
3764 04422   0 01 04430      JMP   AV01          TAKE OUT OF SOURCE RETURN
3765 04423   000000          OCT   0             NEVER CAN EXECUTE THIS WORD
3766 04424   1 02 00001      LDA   1,1           SET SBP FROM SECOND WORD OF ENTRY
3767 04425   0 04 00037      STA   SBP           X
3768 04426   0 12 04414      IRS   ADVS          BUMP THE RETURN POINTER
3769 04427  -0 01 04414      JMP*  ADVS          AND EXIT
3770                    *
3771 04430   140040     AV01 CRA                 CLEAR SIP ON THE WAY OUT TO MAKE
3772 04431   0 04 00034      STA   SIP           THE MESSAGE '0 EXIT' WORK
3773 04432  -0 01 04414      JMP*  ADVS          AND LEAVE MOST HASTILY
3774                    *
3775                    *
3776                    *
3777                        EJCT
```

```
3778                    *       STATEMENT SEARCH ROUTINE
3779                    *
3780                    *
3781                    *   CALLING SEQUENCE:
3782                    *
3783                    *       LDA   CODE      A CONTAINS 8 BIT STATEMENT IDENTIFIER
3784                    *       JST   SSR
3785                    *       ......RETURN    IF STMT NOT FOUND
3786                    *       ......RETURN    IF STMT FOUND
3787                    *
3788                    *       THE STATEMENT IDENTIFIER IS THE NUMBER CORESPONDING TO
3789                    *   THE STATEMENTS NAME IS THE RESERVED IDENTIFIER LIST.  THE
3790                    *   SEARCH IS STARTED AT THE LINE FOLLOWING THE LINE POINTED TO BY
3791                    *   THE STATEMENT INDEX POINTER, SIP.
3792                    *
3793                    *
3794 04433   0 000000   SSR  DAC   **
3795 04434   0 04 00120      STA   TMP3          SAVE THE TARGET
3796 04435   0 10 04414 SSR1 JST   ADVS          ADVANCE TO NEXT STATEMENT
3797 04436  -0 01 04433      JMP*  SSR           PAST END OF PROGRAM...RETURN 1
3798 04437   0 10 03013 SSR2 JST   GCHR          GET STMT IDENTIFIER
3799 04440   0 05 00120      ERA   TMP3          COMPARE WITH SEARCH TARGET
3800 04441   101040         SNZ                 SKIP IF NOT A MATCH
3801 04442   0 01 04447      JMP   SSR3          FOUND STMT OF TYPE WE'RE LOOKING FOR %%%
3802 04443   0 10 04514      JST   SES           SCAN FOR END OF CURRENT STATEMENT
3803 04444   0 11 00433      CAS   C215          TEST FOR END OF PHYSICAL LINE
3804 04445   0 01 04437      JMP   SSR2          NO....TEST NEXT STMT ON THIS LINE
3805 04446   0 01 04435      JMP   SSR1          YES...GO LOOK AT NEXT LINE
3806 04447   0 12 04433 SSR3 IRS   SSR           BUMP RETURN POINTER TO RETURN 2
3807 04450  -0 01 04433      JMP*  SSR           AND EXIT
3808                    *
3809                    *
3810                    *
3811                        EJCT
```

```
                    *
                    *        STATEMENT INDEX SEARCH ROUTINE
                    *
                    *
                    *    CALLING SEQUENCE:
                    *
                    *          JST   SISR
                    *          ......RETURN      IF ENTRY NOT FOUND
                    *          ......RETURN      IF FOUND
                    *
                    *          THIS ROUTINE WILL SEARCH THE STATEMENT INDEX
                    *    FOR AN ENTRY WHOSE VALUE IS THE SAME AS SNUM.  IF
                    *    A MATCH IS FOUND, SIP AND SBP ARE SET, AND THE SECOND
                    *    RETURN IS TAKEN.
                    *
                    *
                    *
04451   0 000000    SISR: DAC   **
04452   0 02 00033        LDA   SIT          SET TMP1 = END OF TABLE + 1
04453   141206            ADA               X
04454   0 04 00116        STA   TMP1        X
04455   0 02 00032        LDA   SIB         GET TABLE LOW ADDRESS
04456   0 11 00033        CAS   SIT         SEE IF TABLE IS EMPTY
04457  -0 01 04451        JMP*  SISR        EXIT...NO MATCH IN AN EMPTY TABLE
04460   000000            OCT   0           NEVER CAN EXECUTE THIS WORD
04461   0 04 00117        STA   TMP2        TMP2 = LOW ADDRESS OF TABLE
04462   140040            CRA               ZERO OUT SIP TO ADVOID CHANCE
04463   0 04 00034        STA   SIP         OF A FALSE EXIT
04464   0 02 00116 SIS3   LDA   TMP1        NEXT ENTRY TO CHECK!
04465   0 07 00117        SUB   TMP2        IS (TMP1-TMP2)/4*2+TMP2
04466   0404 76           LGR   2           X
04467   0414 77           LGL   1           X
04470   0 06 00117        ADD   TMP2        X
04471   0 13 00034        IMA   SIP         NEW ADDRESS IN SIP, LAST ADDR IN A
04472   0 11 00034        CAS   SIP         IF SAME ENTRY IS CHECKED TWICE, THEN
04473   100000            SKP               TARGET IS NOT IN TABLE
04474  -0 01 04451        JMP*  SISR        TAKE NOT FOUND RETURN
04475  -0 02 00034        LDA*  SIP         GET VALUE OF THIS ENTRY
04476   0 11 00057        CAS   SNUM        COMPARE WITH TARGET
04477   0 01 04504        JMP   SIS1        TOO HIGH
04500   0 01 04507        JMP   SIS2        FOUND IT !!!!
                    *
04501   0 02 00034        LDA   SIP         TOO LOW...TRY REGION BETWEEN
04502   0 04 00117        STA   TMP2        SIP AND TMP1 NEXT
04503   0 01 04464        JMP   SIS3        X
                    *
04504   0 02 00034 SIS1   LDA   SIP         TOO HIGH...TRY REGION BETWEEN
04505   0 04 00116        STA   TMP1        TMP2 AND SIP NEXT
04506   0 01 04464        JMP   SIS3        X
                    *
04507   0 35 00034 SIS2   LDX   SIP         FOUND ENTRY ... PULL SOURCE BYTE
```

```
3862 04510   1 02 00001      LDA   1,1        POINTER FROM THE ENTRY
3863 04511   0 04 00037      STA   SBP        X
3864 04512   0 12 04451      IRS   SISR       BUMP THE RETURN ADDRESS
3865 04513  -0 01 04451      JMP*  SISR       AND EXIT
3866                    *
3867                    *
3868                    *
3869                    *     EJCT
```

```
3870                        *         SCAN FOR END OF STATEMENT
3871                        *
3872                        *
3873                        *    CALLING SEQUENCE:
3874                        *
3875                        *         JST   SES
3876                        *         ......RETURN      A CONTAINS STMT DELIMITER (: OR C/R)
3877                        *
3878                        *         THIS ROUTINE WILL READ THROUGH THE CURRENT
3879                        *    SOURCE LINE UNTIL EITHER A 85 OR C/R IS DETECTED.
3880                        *
3881                        *
3882 04514    0 000000    SES  DAC   **
3883 04515    0 10 03013       JST   GCHR              GET NEXT CHARACTER OF STMT
3884 04516    0 10 03137       JST   DLCK              TEST FOR : OR C/R
3885 04517    100000           SKP                     NO
3886 04520   -0 01 04514       JMP*  SES               YES...EXIT
3887 04521    0 11 00510       CAS   INTF              TEST FOR INTEGER CONSTANT
3888 04522    100000           SKP                     NO
3889 04523    0 01 04530       JMP   SES1              YES...GO SKIP 2 CHARACTERS
3890 04524    0 05 00511       ERA   RELF              TEST FOR REAL CONSTANT
3891 04525    100040           SZE                     X
3892 04526    0 01 04515       JMP   SES+1             NO...GO BACK TO LOOK AT NEXT CHARACTER
3893 04527    0 10 03043       JST   GCPK              SKIP TWO CHARACTERS
3894 04530    0 10 03043  SES1 JST   GCPK              DITTO
3895 04531    0 01 04515       JMP   SES+1             GO LOOK AT THE NEXT CHARACTER
3896                        *
3897                        *
3898                        *
3899                             EJCT
```

```
3900                    *         INPUT STATEMENT NUMBER
3901                    *
3902                    *
3903                    *    CALLING SEQUENCE:
3904                    *
3905                    *         JST   ISN
3906                    *         ......RETURN      STMT NO. IN A AND SNUM
3907                    *
3908                    *         THIS ROUTINE CHECKS TO INSURE THAT A STATEMENT
3909                    *    NUMBER IS NEXT IN THE SOURCE TEXT, AND IF SO FORMS
3910                    *    THE SINGLE WORD BINARY REPRESENTATION AND LEAVES
3911                    *    IT IN A AND SNUM.  IF NOT, A 'SN' ERROR IS REPORTED
3912                    *    TO THE USER.
3913                    *
3914                    *
3915 04532   0 000000  ISN  DAC   **
3916 04533   0 10 03013      JST   GCHR          TEST NEXT CHARACTER FOR INTEGER FLAG
3917 04534   0 05 00510      ERA   INTF          (STMT NUMBERS ALWAYS LOOK LIKE INTEGERS)
3918 04535   100040          SZE                 SKIP IF IT IS
3919 04536   0 01 04546      JMP   ISN1          NOT STMT NO. ... SOMETHING IS WRONG
3920 04537   0 10 03043      JST   GCPK          PACK THE INTEGER
3921 04540   0 04 00057      STA   SNUM          LEAVE IT IN SNUM
3922 04541   100040          SZE                 ZERO IS NOT A LEAGL STMT NO.
3923 04542   0 11 00537      CAS   SNMX          MAKE SURE NUMBER IS NOT TOO BIG
3924 04543   0 01 04546      JMP   ISN1          TOO BAD ... NUMBER = 0 OR > 9999
3925 04544  -0 01 04532      JMP*  ISN           ...OK... RETURN
3926 04545  -0 01 04532      JMP*  ISN           ...OK... RETURN
3927 04546   0 10 05206 ISN1 JST   ERR           REPORT STATEMENT NUMBER ERROR
3928 04547   151716          BCI   1,SN          X
3929                    *
3930                    *
3931                    *
3932                         EJCT
```

```
                                *          COMMON STATEMENT EXIT
                                *
                                *
                                *
04550    0 10 03031 SEX   JST   UCHR       REFETCH LAST CHARACTER TO BE SAFE
04551    0 10 03013       JST   GCHR       SEE IF AT END OF LINE
04552    0 11 00433       CAS   C215       X
04553    0 01 03157       JMP   ESMT       NO....GO EXECUTE NEXT STMT ON THIS LINE
04554    0 02 00034 ASQ   LDA   SIP        SEE IF WE ARE IN COMMAND MODE
04555    101040           SNZ              X
04556    0 01 01000       JMP   CMOD       YES...GET NEXT COMMAND FROM CONSOLE
04557    0 10 04414       JST   ADVS       ADVANCE POINTERS TO NEXT LINE
04560    0 01 04404       JMP   EXT1       RAN OUT OF PROGRAM
04561    0 01 03157       JMP   ESMT       GO EXECUTE FIRST STMT ON NEW LINE
                                *
                                *
                                EJCT
```

```
3950                        *          SEARCH DEFINED FUNCTION INDEX
3951                        *
3952                        *
3953                        *          CALLING SEQUENCE:
3954                        *
3955                        *          JST    SDFI
3956                        *          ......RETURN       IF FUNCTION NOT FOUND
3957                        *          ......RETURN       IF FUNCTION FOUND, X POINTS TO
3958                        *                             FIRST WORD OF FUNCTION INDEX ENTRY
3959                        *
3960                        *          THE NEXT CHARACTER IS FETCHED AND USED AS
3961                        *     THE FUNCTION NAME ( IT IS CHECKED TO INSURE
3962                        *     THAT IT IS ALPHABETIC).
3963                        *
3964                        *
3965                        *
3966 04562   0 000000    SDFI DAC    **
3967 04563   0 10 03013       JST    GCHR              GET THE FUNCTION NAME
3968 04564   0 10 03121       JST    ALFA              IS IT ALPHABETIC ¢
3969 04565   0 01 04375       JMP    DF01              NO ... REPORT FUNCTION NAME ERROR
3970 04566   0 02 00022       LDA    DFB               GET FUNCTION INDEX BASE POINTER
3971 04567   101040           SNZ                      IS THE TABLE EMPTY ¢
3972 04570  -0 01 04562       JMP*   SDFI              YES ... TAKE NOT FOUND RETURN
3973 04571   0 11 00023 SD01  CAS    DFT               ARE WE PAST TOP OF THE INDEX ¢
3974 04572  -0 01 04562       JMP*   SDFI              YES ... TAKE NOT FOUND RETURN
3975 04573   000000           OCT    0                 NEVER CAN EXECUTE THIS WORD
3976 04574   0 04 00000       STA    0                 X POINTS TO FIRST WORD OF CURRENT ENTRY
3977 04575   1 02 00000       LDA    0,1               TEST ENTRY NAME AGAINST
3978 04576   0 05 00114       ERA    CHAR              SEARCH TARGET
3979 04577   100040           SZE                      DO THAY MATCH ¢
3980 04600   0 01 04603       JMP    SD02              NO....GO ADVANCE TO NEXT ENTRY
3981 04601   0 12 04562       IRS    SDFI              YES...TAKE NAME FOUND RETURN
3982 04602  -0 01 04562       JMP*   SDFI              X
3983                        *
3984 04603   0 02 00000 SD02  LDA    0                 UPDATE INDEX POINTER TO FIRST
3985 04604   0 06 00457       ADD    C3                WORD OF NEXT ENTRY
3986 04605   0 01 04571       JMP    SD01              CONTINUE SEARCH
3987                        *
3988                        *
3989                        *
3990                             EJCT
```

```
                                *
                                *       VARIABLE NAME ISOLATION ROUTINE
                                *
                                *
                                *   CALLING SEQUENCE:
                                *
                                *       JST   PVN
                                *       ......RETURN       IF SUBSCRIPTED VARIABLE
                                *       ......RETURN       IF SIMPLE VARIABLE
                                *
                                *       THE VARIABLE NAME IS ILOLATED AND LEFT
                                *   IN THE LOCATION VARN.
                                *
                                *
                                *
04606    0 000000    PVN  DAC   **
04607    0 10 03013       JST   GCHR          GET FIRST CHARACTERI OF VARIABLE NAME
04610    0 04 00133       STA   VARN          SAVE FOR NOW
04611    0 10 03121       JST   ALFA          FIRST CHARACTER MUST BE ALPHABETIC
04612    0 01 04630       JMP   PV01          ITS NOT...REPORT IDENTIFIER ERROR
04613    0 10 03013       JST   GCHR          NEXT CHARACTER MAY BE PART OF NAME
04614    0 11 00441       CAS   C250          TEST FOR START OF SUBSCRIPT
04615   100000            SKP                 NO
04616   -0 01 04606       JMP*  PVN           YES...TAKE FIRST RETURN
04617    0 12 04606       IRS   PVN           BUMP FOR SIMPLE VARIABLE RETURN
04620    0 10 03130       JST   NUMC          CURRENT CHAR MUST BE DIGIT IF PART OF NAME
04621    0 01 04626       JMP   PV02          NO....MAYBE WE HAVE GONE TOO FAR
04622   141340            ICA                 PUT SECOND DIGIT OF NAME IN A(1-8)
04623    0 05 00133       ERA   VARN          PUT FIRST CHARACTERI IN A(9-16)
04624    0 04 00133       STA   VARN          SAVE THE COMPLETED NAME
04625   -0 01 04606       JMP*  PVN           AND RETURN
04626    0 10 03121  PV02  JST   ALFA          TWO ALPHABETIC CHARS IN A ROW IS ILLEGAL
04627    0 01 04632       JMP   PV03          NO...GO STEP BACK OVER CURRENT CHARACTER
04630    0 10 05206  PV01  JST   ERR           REPORT IDENTIFIER ERROR
04631   144704            BCI   1,ID          X
04632    0 10 03031  PV03  JST   UCHR          THIS CHAR MUST BE PART OF SOMETHING ELSE
04633   -0 01 04606       JMP*  PVN           RETURN...SINGLE CHAR NAME IN VARN
                                *
                                *
                                *
                                EJCT
```

```
4031                    *       LOCATE/ASSIGN SIMPLE VARIABLE
4032                    *
4033                    *
4034                    *   CALLING SEQUENCE:
4035                    *
4036                    *       JST   ASV
4037                    *       ......RETURN     X POINTS TO WORD PAIR FOR VALUE
4038                    *                        OF THE VARIABLE
4039                    *
4040                    *   REQUIRED SETUP:
4041                    *
4042                    *       THE VARIABLE NAME MUST HAVE PREVIOUSLY BEEN
4043                    *   ISOLATED AND LEFT IN THE LOCATION VARN.
4044                    *
4045                    *
4046                    *
4047 04634   0 000000   ASV  DAC   **
4048 04635   0 10 04664      JST   LSV          IS NAME ALREADY IN THE TABLE ¢
4049 04636   100000          SKP                NO....WE MUST ADD IT
4050 04637  -0 01 04634      JMP*  ASV          YES...RETURN
4051 04640   0 02 00457      LDA   C3           MAKE SURE THERE IS ROOM FOR
4052 04641   0 10 03005      JST   UFSC         ANOTHER ENTRY
4053 04642   0 02 00027      LDA   SVT          IS THE TABLE EMPTY ¢
4054 04643   100040          SZE                X
4055 04644   0 01 04654      JMP   AS01         NO ... GO APPEND ENTRY
4056 04645   0 02 00030      LDA   DVB          GET HIGHEST UNUSED CORE LOCATION
4057 04646   101040          SNZ                X
4058 04647   0 02 00032      LDA   SIB          X
4059 04650   0 07 00417      SUB   C1           X
4060 04651   0 04 00027      STA   SVT          SET TABLE TOP
4061 04652   141206          AOA                ADJUST SO THE TABLE
4062 04653   100000          SKP                BASE COMES OUT RIGHT
4063 04654   0 02 00026 AS01 LDA   SVB          GET PREVIOUS TABLE BASE
4064 04655   0 07 00457      SUB   C3           APPEND THREE WORDS
4065 04656   0 04 00026      STA   SVB          SET NEW BASE
4066 04657   0 04 00000      STA   0            X POINTS TO WORD FOR VARIABLE NAME
4067 04660   0 02 00133      LDA   VARN         PUT VARIABLE NAME
4068 04661   1 04 00000      STA   0,1          IN FIRST WORD OF ENTRY
4069 04662   0 12 00000      IRS   0            S POINTS TO FIRST WORD OF VARIABLE VALUE
4070 04663  -0 01 04634      JMP*  ASV          RETURN
4071                    *
4072                    *
4073                    *
4074                         EJCT
```

```
                    *
                    *        LOCATE SIMPLE VARIABLE
                    *
                    *
                    *    CALLING SEQUENCE:
                    *
                    *         JST   LSV
                    *         ......RETURN       IF NAME NOT FOUND
                    *         ......RETURN       IF NAME FOUND
                    *
                    *    REQUIRED SETUP:
                    *
                    *         THE VARIABLE NAME MUST HAVE PREVIOUSLY
                    *    BEEN ISOLATED AND LEFT IN THE LOCATION VARN.
                    *
                    *
                    *    RETURN STATUS:
                    *
                    *         IF THE NAME IS FOUND, THE INDEX IS
                    *    LEFT POINTING TO THE FIRST WORD OF THE
                    *    WORD PAIR CONTAINING THE VALUE OF THE
                    *    VARIABLE.
                    *
                    *
04664    0 000000   LSV  DAC   **
04665    0 02 00133      LDA   VARN          TEST FOR REFERENCE TO THE DUMMY
04666    0 11 00072      CAS   DEFN          VARIABLE IN A PROGRAMMER DEFINED FUNCTION
04667    100000          SKP                 NO
04670    0 01 04712      JMP   LS01          YES...ITS VALUE IS IN A SPECIAL PLACE
04671    0 02 00026      LDA   SVB           START TABLE SEARCH
04672    101040          SNZ                 IS THE TABLE EMPTY ¢
04673   -0 01 04664      JMP*  LSV           YES...TAKE THE NOT FOUND RETURN
04674    0 11 00027 LS04 CAS   SVT           ARE WE PAST END OF TABLE ¢
04675   -0 01 04664      JMP*  LSV           YES ... TAKE NOT FOUND RETURN
04676    000000          OCT   0             NEVER CAN EXECUTE THIS WORD
04677    0 04 00000      STA   0             X POINTS TO 1ST WORD OF CURRENT ENTRY
04700    1 02 00000      LDA   0,1           COMPARE NAME OF CURRENT ENTRY WITH
04701    0 05 00133      ERA   VARN          THE SEARCH TARGET
04702    100040          SZE                 DO THEY MATCH ¢
04703    0 01 04707      JMP   LS02          NO....GO CHECK NEXT ENTRY
04704    0 12 00000      IRS   0             YES...X POINTS TO VARIABLE VALUE
04705    0 12 04664 LS03 IRS   LSV           TAKE NAME FOUND RETURN
04706   -0 01 04664      JMP*  LSV           X
                    *
04707    0 02 00000 LS02 LDA   0             NOT THIS ENTRY,
04710    0 06 00457      ADD   C3            TRY THE NEXT
04711    0 01 04674      JMP   LS04          X
                    *
04712    0 35 00553 LS01 LDX   DFVD          X POINTS TO DUMMY VARIABLE VALUE
04713    0 01 04705      JMP   LS03          TAKE NAME FOUND RETURN
                    *
```

LS04 (not label; cross-ref list)

```
4125                *
4126                *
4127                *       EJCT
```

```
                    *
                    *        LOCATE/ASSIGN DIMENSIONED VARIABLE
                    *
                    *
                    *    CALLING SEQUENCE:
                    *
                    *          JST   ADV
                    *          ..........
                    *          ......RETURN      SEE BELOW FOR RETURN INFORMATION
                    *
                    *          THIS ROUTINE FIRST CHECKS TO SEE IF THE NAME
                    *    IS ALREADY IN THE TABLE.  IF IT IS, RETURN IS MADE
                    *    WITH THE ADDRESS OF THE ELEMENT IN THE INDEX REGISTER.
                    *    IF NOT, AN ENTRY IS ADDED TO THE DIMENSIONED VARIABLE
                    *    TABLE.  IF THE DIM STMT FLAG IS SET (=0), THEN
                    *    THE VALUE OF THE SUBSCRIPTS AS THEY APPEAR IN THE SOURCE ARE
                    *    USED AS THE DIMENSIONS.  IF THE DIM STMT FLAG IS
                    *    RESET (='12), THEN A VALUE OF 10 IS USED FOR
                    *    EACH DIMENSION.  AFTER THE ENTRY HAS BEEN ADDED,
                    *    THE DIM STMT FLAG IS CHECKED.  IF IT IS SET, RETURN IS
                    *    MADE IMMEADIATLY.  IF NOT, THE NAME IS REPROCESSED
                    *    AND THE LOCATION OF THE SELECTED ARRAY ITEM IS LEFT
                    *    IN THE INDEX REGISTER.
                    *
                    *
                    *
04714   0 000000    ADV  DAC   **
04715   0 12 04714       IRS   ADV          STEP OVER UNUSED WORD IN CALLING SEQ.
04716   0 10 05100 AD07 JST   LDV          IS NAME ALREADY IN TABLE ¢
04717   100000           SKP               NO....IT MUST BE ADDED
04720  -0 01 04714       JMP*  ADV         YES...RETURN, ENTRY ADDRESS IN X
04721   0 02 00133       LDA   VARN        SAVE THE NAME OF THE VARIABLE
04722   0 04 00103       STA   ADT3        X
04723   140040           CRA               CLEAR THE DIMENSION COUNTER
04724   0 04 00104       STA   ADT4        X
04725   141206           ADA               INITIALIZE THE ARRAY SIZE COUNTER
04726   0 04 00106       STA   ADT6        X
04727   0 15 00105       STX   ADT5        SAVE X AS IT MAY HAVE THREAD TO LAST ENTRY
04730   0 02 00037       LDA   SBP         SAVE BYTE POINTER TO START OF SUBSCRIPT
04731   0 04 00110       STA   ADT8        IN CASE IT HAS TO BE REPROCESSED
04732   140040     AD01  CRA               EVALUATE SUBSCRIPT EXPRESSION
04733   0 10 02412       JST   EXPA        X.
04734   0 10 03147       JST   LCVL        GET THE RESULT
04735   0 10 00000       JST   IFLT        CONVERT TO INTEGER
04736   0 01 05076       JMP   AD08        ERROR...SUBSCRIPT SIZE
04737   000201           IAB               SAVE RESULT
04740   0 02 00130       LDA   DIMF        IS THIS A DIMENSION STMT ¢
04741   0 11 00416       CAS.  CO          X
04742   100000           SKP               NO...USE DEFAULT VALUE OF 10
04743   000201           IAB               YES...USE EXPRESSION VALUE
04744   100400           SPL               IS IT A LEGAL SIZE ¢
```

```
4178 04745   0 01 05076        JMP   AD08        NO...REPORT ERROR!
4179 04746   141206           AOA               SET TO LIMIT + 1
4180 04747   0 10 02766       JST   PUSH        LEAVE IT ON THE STACK
4181 04750   0 10 00000       JST   M$11        UPDATE THE ARRAY SIZE COUNTER
4182 04751   0 000106         DAC   ADT6        X
4183 04752   0 04 00106       STA   ADT6        X
4184 04753   0 12 00104       IRS   ADT4        UPDATE DIMENSION COUNT
4185 04754   000201           IAB               TEST FOR GROSS SIZE ERROR
4186 04755   100040           SZE               X
4187 04756   0 01 02773       JMP   MEMO        NO MACHINE WE SELL CAN HOLD THIS SIZE ARRAY
4188 04757   0 10 03013       JST   GCHR        TEST FOR END OF SUBSCRIPT LIST
4189 04760   0 11 00445       CAS   C254        X
4190 04761   100000           SKP               X
4191 04762   0 01 04732       JMP   AD01        NO...GO PROCESS NEXT SUBSCRIPT
4192 04763   0 05 00442       ERA   C251        NOT COMMA, MUST BE ')'
4193 04764   100040           SZE               X
4194 04765   0 01 05076       JMP   AD08        IT'S NOT...REPORT ERROR
4195 04766   0 02 00106       LDA   ADT6        CALCULATE SIZE OF TABLE ENTRY
4196 04767   0414 77          LGL   1           2 WORDS PER ARRAY ELEMENT
4197 04770   0 06 00104       ADD   ADT4        1 WORD PER DIMENSION
4198 04771   0 06 00457       ADD   C3          3 WORDS FOR GENERAL OVERHEAD
4199 04772   0 04 00107       STA   ADT7        SAVE FOR LATER REFERENCE
4200 04773   0 10 03005       JST   UFSC        MAKE SURE THERE IS ENOUGH FREE SPACE
4201 04774   0 02 00026       LDA   SVB         IS SIMPLE VARIABLE TABLE EMPTY ?
4202 04775   101040           SNZ               X
4203 04776   0 01 05016       JMP   AD02        YES...NO TABLE MOVE NEEDED
4204 04777   0 04 00101       STA   ADT1        ADDRESS OF FIRST WORD TO BE MOVED
4205 05000   0 07 00107       SUB   ADT7        GET NEW TABLE BASE
4206 05001   0 04 00026       STA   SVB         SET SIMPLE VARIABLE BASE POINTER
4207 05002   0 04 00102       STA   ADT2        SET DESTINATION POINTER FOR MOVE
4208 05003  -0 02 00101 AD03  LDA*  ADT1        MOVE ONE WORD
4209 05004  -0 04 00102       STA*  ADT2        X
4210 05005   0 12 00101       IRS   ADT1        BUMP THE SOURCE AND DESTINATION POINTERS
4211 05006   0 12 00102       IRS   ADT2        X
4212 05007   0 02 00027       LDA   SVT         TEST FOR MOVE COMPLETE
4213 05010   0 07 00101       SUB   ADT1        X
4214 05011   101400           SMI               X
4215 05012   0 01 05003       JMP   AD03        NO...GO MOVE ANOTHER WORD
4216 05013   0 02 00102       LDA   ADT2        UPDATE SIMPLE VARIABLE TABLE
4217 05014   0 07 00417       SUB   C1          TOP POINTER
4218 05015   0 04 00027       STA   SVT         X
4219 05016   0 02 00031 AD02  LDA   DVT         IS THIS FIRST ENTRY IN DIMENSIONED
4220 05017   100040           SZE               VARIABLE TABLE ?
4221 05020   0 01 05071       JMP   AD04        NO....LINK IN LAST ENTRY MUST BE THREADED
4222 05021   0 02 00032       LDA   SIB         START DIMENSIONED VARIABLE TABLE
4223 05022   0 07 00417       SUB   C1          JUST BELOW THE BASE OF THE
4224 05023   0 04 00031       STA   DVT         STATEMENT INDEX
4225 05024   0 07 00107 AD05  SUB   ADT7        GET NEW TABLE BASE ADDRESS
4226 05025   141206           AOA               X
4227 05026   0 04 00030       STA   DVB         X
```

```
4228 05027   0 04 00000      STA   0          X
4229 05030   0 02 00106      LDA   ADT6       CALCULATE -NO. OF WORDS OF ELEMENT STORAGE
4230 05031   0414 77         LGL   1          X
4231 05032   140407          TCA              X
4232 05033   0 04 00101      STA   ADT1       X
4233 05034   140040          CRA              SET ENTIRE ARRAY TO ZERO
4234 05035   1 04 00000      STA   0,1        X
4235 05036   0 12 00000      IRS   0          X
4236 05037   0 12 00101      IRS   ADT1       X
4237 05040   0 01 05035      JMP   *-3        X
4238 05041   0 02 00104      LDA   ADT4       SETUP TO MOVE DIMENSION LIMITS FROM
4239 05042   140407          TCA              THE STACK TO THE ENTRY
4240 05043   0 04 00101      STA   ADT1       X
4241 05044   0 10 02775      JST   POP        REMOVE DIMENSION LIMIT FROM THE STACK
4242 05045   1 04 00000      STA   0,1        PUT IT IN TABLE ENTRY
4243 05046   0 12 00000      IRS   0          BUMP TABLE ENTRY POINTER
4244 05047   0 12 00101      IRS   ADT1       BUMP DIMENSION COUNT
4245 05050   0 01 05044      JMP   *-4        GO MOVE ANOTHER LIMIT
4246 05051   0 02 00030      LDA   DVB        PUT IN ADDRESS OF FIRST WORD OF ARRAY
4247 05052   1 04 00000      STA   0,1        STORAGE FOR THIS VARIABLE
4248 05053   140040          CRA              CLEAR ETNRY THREAD
4249 05054   1 04 00001      STA   1,1        X
4250 05055   0 02 00104      LDA   ADT4       FIRST WORD OF ENTRY CONTAINS
4251 05056   0400 70         LRL   8          NAME IN BITS 1 TO 8 AND NUMBER
4252 05057   0 02 00103      LDA   ADT3       OF DIMENSIONS IN BITS 9-16
4253 05060   0 04 00133      STA   VARN       X
4254 05061   0410 70         LLL   8          X
4255 05062   1 04 00002      STA   2,1        X
4256 05063   0 02 00130      LDA   DIMF       IS THIS A DIMENSION STATEMENT ?
4257 05064   101040          SNZ              X
4258 05065  -0 01 04714      JMP*  ADV        YES...DO NOT NEED TO LOCATE A SPECIFIC ITEM
4259 05066   0 02 00110      LDA   ADT8       RESTORE BYTE POINTER TO START OF SUBSCRIPT
4260 05067   0 04 00037      STA   SBP        X
4261 05070   0 01 04716      JMP   AD07       GO LOCATE SPECIFIC ARRAY ITEM
4262                        *
4263 05071   0 35 00105 AD04 LDX   ADT5       SET THREAD IN LAST TABLE ENTRY TO
4264 05072   0 02 00030      LDA   DVB        POINT TO THIS ENTRY
4265 05073   0 07 00417      SUB   C1         X
4266 05074   1 04 77777      STA   -1,1       X
4267 05075   0 01 05024      JMP   AD05       GO SETUP THIS ENTRY
4268                        *
4269 05076   0 10 05206 AD08 JST   ERR        REPORT SUBSCRIPT ERROR
4270 05077   140723          BCI   1,AS
4271                        *
4272                        *
4273                        *
4274                            EJCT
```

```
4275                    *       LOCATE DIMENSIONED VARIABLE
4276                    *
4277                    *
4278                    *   CALLING SEQUENCE:
4279                    *
4280                    *       JST   LDV
4281                    *       ......RETURN      IF NAME NOT IN TABLE
4282                    *       ......RETURN.     IF NAME FOUND, X POINTS TO ARRAY ELEMENT
4283                    *
4284                    *   THIS ROUTINE ASSUMES:
4285                    *
4286                    *       1)  THE CHARACTER IN CHAR IS ALPHABETIC
4287                    *       2)  THE NEXT SOURCE CHARACETR IS A '('
4288                    *
4289                    *       IF THE NAME IS IN THE TABLE, THE SUBSCRIPT
4290                    *   WILL BE EVALUATED AND THE INDEX REGISTER WILL BE SET
4291                    *   TO THE ADDRESS OF THE SELECTED ARRAY ELEMENT.
4292                    *
4293                    *
4294                    *
4295 05100   0 000000   LDV  DAC   **
4296 05101   0 02 00031      LDA   DVT         PICK UP THREAD TO FIRST ENTRY
4297 05102   101040     LD02 SNZ               AT END OF TABLE ?
4298 05103  -0 01 05100      JMP*  LDV         YES...TAKE NOT FOUND RETURN
4299 05104   0 04 00000      STA   0           X POINTS TO TOP WORD OF ENTRY
4300 05105   1 02 00000      LDA   0,1         GET WORD CONTAINING NAME
4301 05106   0400 70         LRL   8           NO. OF DIMS TO B, NAME IN A
4302 05107   0 05 00133      ERA   VARN        COMPARE WITH TARGET
4303 05110   101040          SNZ               X
4304 05111   0 01 05114      JMP   LD01        FOUND ENTRY !!!
4305 05112   1 02 77777      LDA   -1,1        PICK UP LINK TO NEXT ENTRY
4306 05113   0 01 05102      JMP   LD02        GO CHECK IT
4307                    *
4308 05114   0 12 05100 LD01 IRS   LDV         BUMP RETURN ADDRESS TO FOUND RETURN
4309 05115   0 02 05100      LDA   LDV         PUT RETURN ADDRESS ON THE STACK
4310 05116   0 10 02766      JST   PUSH        X
4311 05117   0 02 00111      LDA   DPTR        PUT EXISTING VALUES OF THE TEMP
4312 05120   0 10 02766      JST   PUSH        STORAGE LOCATIONS THAT THIS ROUTINE USES
4313 05121   0 02 00112      LDA   DCT1        ON THE STACK, AS THIS IS A RE-ENTRANT
4314 05122   0 10 02766      JST   PUSH        ROUTINE
4315 05123   0 02 00113      LDA   DCT2        X
4316 05124   0 10 02766      JST   PUSH        X
4317 05125   140040          CRA               NO. OF DIMENSIONS TO A
4318 05126   0410 70         LLL   8           X
4319 05127   140407          TCA               NEGATE FOR COUNTING
4320 05130   0 04 00112      STA   DCT1        SAVE FOR TWO LOOPS THROUGH ENTRY
4321 05131   0 04 00113      STA   DCT2        X
4322 05132   0 02 00000      LDA   0           SET TABLE POINTER TO LIMIT OF
4323 05133   0 07 00457      SUB   C3          DIMENSION 1
4324 05134   0 04 00111      STA   DPTR        X
```

```
4325 05135    0 01 05140        JMP   LD03      GO EVALUATE THE SUBSCRIPTS
4326 05136    0 02 00445  LD04  LDA   C254      MAKE SURE ',' SEPERATES
4327 05137    0 10 03050        JST   GCCK      THE SUBSCRIPT EXPRESSIONS
4328 05140   140040         LD03  CRA           EVALUATE THE NEXT SUBSCRIPT EXPRESSION
4329 05141    0 10 02412        JST   EXPA      X
4330 05142    0 10 03147        JST   LCVL      RESULT TO A+B
4331 05143    0 10 00000        JST   IFLT      MAKE IT AN INTEGER
4332 05144    0 01 05076        JMP   AD08      ERROR...TOO LARGE
4333 05145    0 11 00515        CAS   M1        MAKE SURE VALUE IS BETWEEN
4334 05146   -0 11 00111        CAS*  DPTR      ZERO AND LIMIT FOR THIS DIMENSION
4335 05147    0 01 05076        JMP   AD08      ERROR...OUT OF RANGE
4336 05150    0 01 05076        JMP   AD08      ERROR...OUT OF RANGE
4337 05151    0 10 02766        JST   PUSH      LEAVE THE SUBSCRIPT ON THE STACK
4338 05152    0 02 00111        LDA   DPTR      DECREMENT THE TABLE POINTER
4339 05153    0 07 00417        SUB   C1        X
4340 05154    0 04 00111        STA   DPTR      X
4341 05155    0 12 00112        IRS   DCT1      BUMP DIMENSION COUNT
4342 05156    0 01 05136        JMP   LD04      GO PROCESS NEXT SUBSCRIPT
4343 05157    0 02 00442        LDA   C251      MAKE SURE ')' ENDS SUBSCRIPT LIST
4344 05160    0 10 03050        JST   GCCK      X
4345 05161    0 12 00111        IRS   DPTR      BUMP TABLE POINTER TO LIMIT OF LAST DIM.
4346 05162    0 10 00000  LD06  JST   M&11     PREVIOUS ACCUM. * LIMIT OF CURRENT DIM.
4347 05163   -0 000111         DAC*  DPTR      X
4348 05164    0 12 00111        IRS   DPTR      UPDATE ENTRY POINTER
4349 05165    0 04 00112        STA   DCT1      SAVE PARTIAL RESULT
4350 05166    0 10 02775        JST   POP       GET SUBSCRIPT VALUE FOR THIS POSITION
4351 05167    0 06 00112        ADD   DCT1      X
4352 05170    0 12 00113        IRS   DCT2      HAVE ALL THE SUBSCRIPTS BEEN PROCESSED
4353 05171    0 01 05162        JMP   LD06      NO...GO ENTER NEXT ONE
4354 05172    0414 77           LGL   1         *2 AS TWO WORDS PER ELEMENT
4355 05173   -0 06 00111        ADD*  DPTR      ADD TO BASE OF ARRAY STORAGE
4356 05174    0 04 00000        STA   0         LEAVE ELEMENT PNTR IN X FOR CALLER
4357 05175    0 10 02775        JST   POP       RESTORE THE TEMP. LOCATIONS
4358 05176    0 04 00113        STA   DCT2      X
4359 05177    0 10 02775        JST   POP       X
4360 05200    0 04 00112        STA   DCT1      X
4361 05201    0 10 02775        JST   POP       X
4362 05202    0 04 00111        STA   DPTR      X
4363 05203    0 10 02775        JST   POP       RETRIEVE THE RETURN ADDRESS
4364 05204    0 04 05100        STA   LDV       X
4365 05205   -0 01 05100        JMP*  LDV       AND EXIT
4366                        *
4367                        *
4368                            EJCT
```

```
                                *         BASIC-16
4369                            *          ERROR REPORTING ROUTINE
4370                            *
4371                            *
4372                            *    CALLING SEQUENCE:
4373                            *
4374                            *         JST   ERR
4375                            *         BCI   1,XX        XX IS ERROR CODE
4376                            *
4377                            *         AFTER THE MESSAGE IS PRINTED, CONTROL
4378                            *    IS RETURNED TO COMMAND MODE.
4379                            *
4380                            *
4381 05206    0 000000    ERR  DAC   **
4382 05207    0 10 00000       JST   LFCR             ADVANCE TO NEXT LINE ON ASR
4383 05210   -0 02 05206       LDA*  ERR              PUT ERROR CODE IN MESSAGE STRING
4384 05211    0 04 05221       STA   EMST             X
4385 05212    0 10 02713       JST   TYPE             PRINT THE MESSAGE
4386 05213    0 005216         DAC   EMSG             X
4387 05214    0 10 02702       JST   PLN              PRINT THE CURRENT LINE NUMBER
4388 05215    0 01 01000       JMP   CMOD             RETURN TO COMMAND MODE
4389                            *
4390 05216    142722      EMSG BCI   3,ERROR
     05217    151317
     05220    151240
4391 05221    154330      EMST BCI   1,XX           ERROR CODE GOES HERE
4392 05222    120314           BCI   3, LINE
     05223    144716
     05224    142640
4393 05225    000000           OCT   0              MESSAGE TERMINATOR
4394                            *
4395                            *
4396                            *
4397                             EJCT
```

4398 END (cmd)

A$22	000000E	ADSF	000000E	AD01	004732A	AD02	005016A
AD03	005003A	AD04	005071A	AD05	005024A	AD07	004716A
AD08	005076A	ADT1	000101A	ADT2	000102A	ADT3	000103A
ADT4	000104A	ADT5	000105A	ADT6	000106A	ADT7	000107A
ADT8	000110A	ADV	004714A	ADVS	004414A	ALFA	003121A
AS01	004654A	ASN1	003266A	ASN2	003242A	ASN3	003260A
ASNM	003240A	ASQ	004554A	ASV	004634A	ATND	000573A
ATNF	000000E	ATNQ	000232A	AV01	004430A	AYOM	000323A
AYON	000235A	B16T	000415A	BKMS	003234A	BRKC	000000E
BRKF	000127A	C0	000416A	C1	000417A	C10	000420A
C11	000421A	C110	000422A	C12	000423A	C16	000424A
C17	000425A	C2	000426A	C20	000427A	C212	000432A
C215	000433A	C220	000434A	C221	000430A	C223	000435A
C23	000431A	C240	000436A	C241	000437A	C242	000440A
C250	000441A	C251	000442A	C252	000443A	C253	000444A
C254	000445A	C255	000446A	C256	000447A	C257	000450A
C260	000451A	C261	000452A	C272	000453A	C273	000454A
C275	000455A	C277	000456A	C3	000457A	C300	000460A
C305	000461A	C307	000462A	C333	000463A	C336	000464A
C337	000465A	C4	000466A	C43	000467A	C44	000470A
C5	000471A	C50	000472A	C52	000473A	C54	000474A
CALL	003724A	CHAR	000114A	CJMP	000567A	CJST	000554A
CL01	004036A	CL02	003777A	CL03	004020A	CL04	004030A
CL05	003742A	CL06	003765A	CLER	001045A	CLRT	002734A
CLT1	000067A	CLT2	000070A	CLT3	000071A	CMAX	000475A
CMOD	001000A	CON1	000125A	CON2	000126A	CONT	001114A
COSD	000571A	COSF	000000E	CPOS	000046A	CVAL	000041A
D$22	000000E	DBP	000040A	DCT1	000112A	DCT2	000113A
DEF	004301A	DEFF	000545A	DEFN	000072A	DEFV	000073A
DELT	000603A	DF01	004375A	DF02	004353A	DF03	004340A
DF04	004327A	DF05	004372A	DF06	004347A	DF07	004377A
DFB	000022A	DFQ	000263A	DFSE	003646A	DFT	000023A
DFVD	000553A	DIMC	000544A	DIMF	000130A	DLCK	003137A
DPTR	000111A	DTAC	000476A	DVB	000030A	DVT	000031A
E$22	000000E	ECTR	000123A	ED01	002222A	ED02	002177A
ED03	002235A	ED04	002255A	ED05	002245A	ED06	002273A
ED07	002261A	ED08	002336A	ED09	002341A	ED10	002367A
ED11	002206A	ED12	002353A	EMSG	005216A	EMST	005221A
ERR	005206A	ES01	003221A	ES02	003217A	ESMT	003157A
ETAPE	000000E	EX01	002443A	EX02	002446A	EX03	002451A
EX04	002456A	EX05	002606A	EX06	002424A	EX07	002434A
EX09	002513A	EX10	002547A	EX11	002554A	EX13	002506A
EX14	002510A	EX15	002503A	EX16	002571A	EX17	002600A
EX19	002614A	EX20	002622A	EX21	002534A	EX22	002512A
EX24	002576A	EX30	002635A	EXIT	004403A	EXP	000122A
EXPA	002412A	EXPC	002630A	EXPF	000000E	EXT1	004404A
EXT2	004411A	F1	000477A	F10	000501A	F10R	000503A
FINT	000000E	FM1	000505A	FNB	000024A	FNC	000545A

FNT	000025A	FOR	003436A	FR01	003451A	FR02	003506A
FR03	003522A	FR04	003542A	FR05	003545A	FR06	003453A
FR07	003556A	FRNG	002374A	FRST	003560A	FRT1	003562A
FRT2	003474A	FSC	000050A	GCC1	003061A	GCC2	003060A
GCCK	003050A	GCHR	003013A	GCHX	003036A	GCPK	003043A
GDLM	003062A	GNBC	003071A	GOSB	003664A	GOT2	003272A
GOT3	003301A	GOTO	003270A	GTC	000507A	H$22	000000E
HERE	000415A	HMAM	000307A	IBUF	000255A	IDAC	004110A
IDMS	000220A	IDNT	001756A	IF	003331A	IF01	003427A
IF02	003341A	IF03	003362A	IF04	003360A	IFLT	000000E
IFT1	000051A	IFT2	000052A	IL05	001525A	IL06	001536A
IL07	001701A	IL08	001754A	IL10	001552A	IL11	001570A
IL12	001627A	IL13	001642A	IL14	001624A	IL15	001577A
IL16	001635A	IL17	001676A	IL18	001666A	IL20	001707A
IL21	001741A	IL22	001517A	IL31	001602A	IL32	001502A
IL39	001612A	ILIN	001467A	ILT1	000076A	ILT2	000077A
ILT3	000100A	INA1	000000E	INHC	000124A	INIT	000000E
INPT	004040A	INT1	004117A	INTF	000510A	IPDS	002755A
IPUT	000000E	ISBP	004111A	ISN	004532A	ISN1	004546A
ISSM	000350A	ITAPE	000000E	JOB	001040A	LS22	000000E
LCHR	000115A	LCVLI	003147A	LD01	005114A	LD02	005102A
LD03	005140A	LD04	005136A	LD06	005162A	LDV	005100A
LE10	000512A	LFCR	000000E	LIST	001132A	LOAD	001125A
LODF	000131A	LOGF	000000E	LOP	000075A	LS01	004712A
LS02	004707A	LS03	004705A	LS04	004674A	LSBP	000514A
LSTF	000132A	LSV	004664A	LT01	001164A	LT02	001300A
LT03	001206A	LT04	001252A	LT05	001255A	LT06	001267A
LT07	001246A	LT08	001233A	LT09	001231A	LT10	001261A
LT11	001163A	LT12	001205A	LT13	001154A	LT14	001275A
LTT1	000034A	LTT2	000053A	LTT3	000054A	LTT4	000055A
LTT5	000056A	LVALI	000043A	M$11	000000E	M$22	000000E
M1	000515A	M10	000516A	M100	000517A	M11	003655A
M12	000520A	M2	000521A	M20	000522A	M21	000523A
M3	000524A	M5	000525A	M51	000526A	M53	000527A
M6	000530A	MCOLI	000531A	MEMO	002773A	N$22	000000E
NEXT	003566A	NUMC	003130A	NX01	003644A	NX02	003637A
NX03	003641A	NX04	003573A	NXT1	003607A	NXT2	003613A
ON	003304A	ON1	003311A	ON2	003322A	ON3	003316A
OTA1	000000E	PCVLI	002123A	PDLP	000035A	PLN	002702A
PNCH	001130A	POP	002775A	PR01	004172A	PR02	004257A
PR03	004230A	PR04	004170A	PR05	004277A	PR06	004242A
PR07	004255A	PR08	004265A	PR09	004246A	PR10	004211A
PR11	004176A	PRNT	004165A	PRT1	000045A	PTB	000020A
PTH	000021A	PUSH	002766A	PV01	004630A	PV02	004626A
PV03	004632A	PVN	004606A	QUIT	001123A	RD01	004103A
RD02	004046A	RD03	004067A	RD04	004056A	RD06	004143A
RD07	004136A	RDAC	004124A	RDT1	000047A	RDT2	004073A
READ	004044A	RELF	000511A	REM	004401A	REMF	000552A
REST	004156A	RN01	001056A	RN02	001073A	RN03	001061A
RN04	001112A	RNDF	000000E	ROND	000532A	RSBP	004126A

RSIP	004125A	RSTKI	000375A	RSTR	004153A	RTB	000534A
RTM	000535A	RTP	000036A	RTRN	003704A	RUN	001047A
S522	000000E	SBP	000037A	SBUF	000135A	SCHR	003077A
SCTQ	000247A	SCVLI	003153A	SD01	004571A	SD02	004603A
SDFI	004562A	SEDI	000060A	SES	004514A	SES1	004530A
SEX	004550A	SFNLI	000567A	SGN	002404A	SGNF	000000E
SIB	000032A	SIGN	000121A	SIND	000570A	SINF	000000E
SIP	000034A	SIS1	004504A	SIS2	004507A	SIS3	004464A
SISR	004451A	SIT	000033A	SMAX	000536A	SNMX	000537A
SNUM	000057A	SPAC	002730A	SORD	000577A	SORF	000000E
SQRQ	000260A	SSBP	001007A	SSR	004433A	SSR1	004435A
SSR2	004437A	SSR3	004447A	ST01	001435A	ST02	001314A
ST03	001344A	ST04	001333A	ST05	001367A	ST06	001352A
ST07	001413A	ST08	001375A	ST09	001410A	ST10	001433A
ST11	001426A	ST12	001456A	ST13	001440A	STMT	001306A
STPC	000540A	STT1	000061A	STT2	000062A	STT3	000063A
STT4	000064A	STT5	000065A	STT6	000066A	SVB	000026A
SVT	000027A	SYSH	000547A	SYSL	000546A	TABC	000541A
TAND	000572A	TANF	000000E	THNC	000542A	TINI	000000E
TMP1	000116A	TMP2	000117A	TMP3	000120A	TOC	000543A
TYP1	002721A	TYP2	002726A	TYPE	002713A	UCHR	003031A
UFSC	003005A	USPM	000365A	VARN	000133A	WKD7	000551A
WORK	000256A	WRKD	000550A	XCHR	003020A	XON	000134A

0000 WARNING OR ERROR FLAGS
DAP-16 MOD 2 REV. A 03-16-70

2 parts

		1518 1534 1542 1557 1984 2046 2261 2446
		2506 2525 2573 2595 2765 2833 2898 2951
		2993 3009 3061 3088 3306 3410 3419 3528
		3534 3555 3576 3600 3607 3637 3798 3883
		3916 3938 3967 4006 4010
2504	GCHX	2508 2527 2549
2524	GCPK	1111 1114 1116 1267 2004 2008 2010 2528
		3893 3894 3920
2572	GDLM	931 2198 2576 2841 2864 3103 3147 3235
		3258 3313 3424 3492 3731
2594	GNBC	465 1357 1404 1432 1440 1525 2597 2599
3229	GOSB	2782
2865	GOT2	2913 3244
2873	GOT3	2867
2863	GOTO	948 2777 2996 3004
732	GTC	2899 2994
	H$22	525 1475 1762 1768 2734 2847 3059 3125
		3164 3417
650	HERE	672
664	HMAM	485
639	IBUF	640 3305 3324 3443 3443
3433	IDAC	3393
652	IDMS	479
1592	IDNT	736 736
2940	IF	2778
3009	IF01	2947
2951	IF02	2962 2965
2969	IF03	2954 2955
2966	IF04	2973
	IFLT	534 1756 1776 2904 2982 3012 3298 3581
583	IFT1	2941 2960 2970 2974 2986
584	IFT2	2942 2963 2971
1379	IL05	1360 1385 1387
1391	IL06	1363 1366
1509	IL07	1369
1562	IL08	1383
1404	IL10	1417 1458
1419	IL11	1406
1455	IL12	1421
1472	IL13	1424 1431
1451	IL14	1438
1429	IL15	
1464	IL16	1442
1502	IL17	1484
1493	IL18	1504
1515	IL20	1528 1541
1547	IL21	1523
1370	IL22	1386 1395 1543 1550 1552 1560
1432	IL31	
1357	IL32	1373 1500

62 SYMBOLS
149 REFERS
4398 RECORDS
FLAGS OMITTED

016-XREF 24 OCT 69